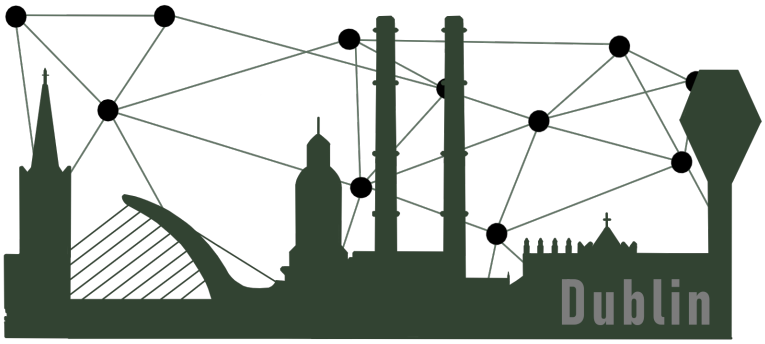
International Network Optimization Conference 2024

Dublin, 11-13 March 2024



INOC - 2024

Book of Abstracts

All times are Greenwich Mean Time (GMT)

Contents

From green networking to network virtualization: some interesting problems arising from telecommunication applications.

Bernardetta Addis

Universitè de Lorraine

Abstract

Telecommunication networks and, more broadly, information and communication technology (ICT), are integral components to the optimization community.
In this talk, we introduce optimization problems we have tackled in collaboration with telecommunication experts over the past decade. The first half focuses on energy-aware network management, highlighting the importance of efficient routing and dynamic device switching. This approach is crucial for reducing energy consumption without compromising quality of service (QoS).
The second part of the presentation explores a problem arising from the convergence of network and computing systems: the placement and routing of Virtual Network Functions (VNFs). This second application involves a novel combination of network design and facility location optimization. Our contributions include mathematical programming models, a comprehensive analysis of their performance in a realistic setting, and a comparative analysis of its computational complexity compared to current state-of-the-art formulations.

Date: Monday 11 March 2024
Time: 9:30–10:30
Location: Q014

Self-adjusting networks

Stefan Schmid

TU Berlin

Abstract

In this talk, I will present the vision of self-adjusting networks: networks ("graphs") which are optimized towards, and "match", the traffic workload they serve. These networks find applications for example in datacenters: Over the last years, the bandwidth and latency requirements of modern datacenter applications have led researchers to propose various datacenter topology designs using static, dynamic demand-oblivious (rotor), and/or dynamic demand-aware switches. However, given the diverse nature of datacenter traffic, there is little consensus about how these designs would fare against each other. We will discuss information-theoretic metrics to quantify the structure in communication traffic as well as the achievable performance in datacenter networks matching their demands, present network optimization principles accordingly, and identify open research challenges. I will also show how the notions of self-adjusting networks and demand-aware graphs relate to classic optimization problems in theoretical computer science

Date: Tuesday 12 March 2024
Time: 9:30–10:30
Location: Q014

Benders Adaptive-Cuts Method Applied to Network Design and Facility Location Problems Under Uncertainty

Ivana Ljubic

ESSEC Business School

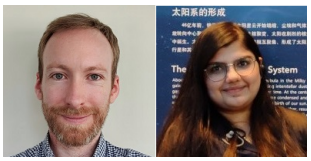
Abstract

Benders decomposition is one of the most applied methods to solve two-stage stochastic problems (TSSP) with a large number of scenarios. The main idea behind the Benders decomposition is to solve a large problem by replacing the values of the second-stage subproblems with individual variables and progressively forcing those variables to reach the optimal value of the subproblems, dynamically inserting additional valid constraints, known as Benders cuts. Most traditional implementations add a cut for each scenario (multicut) or a single cut that includes all scenarios. In this talk, we present a novel Benders adaptive-cuts method, where the Benders cuts are aggregated according to a partition of the scenarios, which is dynamically refined using the LP-dual information of the subproblems. This scenario aggregation/disaggregation is based on the Generalized Adaptive Partitioning Method, which has been successfully applied to TSSPs. Our new method can be interpreted as a compromise between the Benders single-cuts and multicuts methods, drawing on the advantages of both sides, by rendering the initial iterations faster (as for the single-cuts Benders) and ensuring the overall faster convergence (as for the multicuts Benders). We will demonstrate how Benders adaptive-cuts can be applied to the Stochastic Multi-Commodity Network Design Problem and the conditional value-at-risk (CVaR) Facility Location Problem. The new method outperforms the other implementations of Benders methods, as well as other standard methods for solving TSSPs, in particular when the number of scenarios is very large. Moreover, our study demonstrates that the method is not only effective for the risk-neutral decision makers, but also that it can be used in combination with the risk-averse CVaR objective.

Reference: C. Ramirez-Pico, I. Ljubic, E. Moreno: Benders Adaptive-Cuts Method for Two-Stage Stochastic Programs, Transportation Science, online first, (2023), https://doi.org/10.1287/trsc.2022.0073

Date: Wednesday 13 March 2024
Time: 9:30–10:30
Location: Q014

Buildings as Smart Grid Network Components



Scott McDonald and Kanika Sharma

Eaton

Abstract

Tutorial Abstract: Buildings are becoming energy hubs. Building owners and operators need to be prepared for the future and meet new regulations â€“ design future buildings, integrate EV chargers or leverage renewable energy produced on-site while managing the energy flows and planning power capacity. Reducing emissions to mitigate climate change is fast becoming law across Europe. The switch to electric vehicles and on-site energy generation are part of the response, and this means integrating assets such as EV chargers, solar PV, and energy storage systems into buildings.
As buildings become energy hubs, Building Energy Management Software is needed to help the building manager reduce energy costs and CO2 emissions by collecting and analyzing data, and predicting and managing energy flows. This talk will specifically focus on analyzing the available data from hardware assets and using an optimizer to solve for the best control strategy of the controllable assets including battery energy storage system and diesel generators. We will discuss the modeling process of forecasting building load, PV production and EV load along with the design of the optimizer. The optimizer produces an optimized schedule for a customised horizon (e.g., next 24 hours) by constructing and solving mathematical models while still meeting energy demands and the objectives.

Date: Monday 11 March 2024
Time: 14:00–15:00
Location: Q014

Optimization Methods for Large-scale Cell-free Massive MIMO

Nam Tran, University College Dublin, University College Dublin

Abstract

The speedy rollout of 5G networks across the globe over the past two years simply means now is the right time to envision what 6G looks like. Although 5G networks can support high data rates, beyond-5G/6G networks will need a paradigm shift in wireless access technologies to keep pace with the anticipated traffic explosion. In this talk I will introduce some fundamentals of cell-free massive multiple-input multiple-output (CFmMIMO) technology, which has been proposed as a solution to the inherent limitations of cellular systems and is considered to be a disruptive technology for beyond-5G/6G wireless networks. Without cell boundaries, CFmMIMO can offer uniformly good service for all users across the network.
This presentation will specifically focus on the design of CFmMIMO system from an optimization perspective. Due to the relatively small number of antennas per access point (AP), CFmMIMO typically requires a substantially large number of APs. This leads to large-scale resource allocation problems, calling for novel scalable methods to make CFmMIMO more practically feasible. In this context, I will introduce specific power control problems arising from CFmMIMO, describe associated challenges, and then showcase some of the current methodologies developed to achieve high-performance solutions.

Date: Tuesday 12 March 2024
Time: 14:00–15:00
Location: Q014

INOC - 2024

Session Session 1A: Telecommunication Networks

Monday 11 March 2024, 11:00-12:30

Q011

Enhancing the resilience of telecommunication networks through geodiversification

José Alves[1], Maria Teresa Godinho[2], and Marta Pascoal[3]

[1]Department of Engineering, Polytechnic Institute of Beja, Portugal, ✉ 22424@stu.ipbeja.pt
[2]Department of Mathematical and Physical Sciences, Polytechnic Institute of Beja, Portugal; Cmaf-cIO, University of Lisbon, Portugal, ✉ mtgodinho@ipbeja.pt
[3]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy; Universidade de Coimbra; INESC-Coimbra, Portugal, ✉ marta.brazpascoal@polimi.it

The resilience of a system can be defined as the ability to prepare for and adapt to changing conditions and withstand and recover rapidly from disruptions. Resilience includes the ability to withstand and recover from deliberate attacks, accidents, or naturally occurring threats or incidents [3]. A resilient telecommunications network is, therefore, a network that is both prepared and quick to recover from disruption. A survey on this topic can be found in [4]. One way to increase the preparedness of a resilient network is to guarantee the existence of alternative paths linking origins and destinations. In fact, if at least one of the alternative paths survives the disruption, it is ensured that the flow of communication prevails. When dealing with disasters that may impact on large areas of a terrain, such as natural disasters, this translates into imposing a safety geographical distance between the paths.

Recently, Godinho and Pascoal [2] addressed the problem of finding K shortest paths between two nodes, such that each pair of these paths is separated by a distance of at least D, for a given integer $K \geq 2$ and a given $D > 0$. This problem is named the K-D Geodiverse Shortest Path Problem (`KD-GPaths` problem). However, when dealing with real world networks, it is not rare that no feasible solution of this problem exists for some target value of D. In such case, a set of K alternative paths as geodiverse as possible should be sought, alternatively. This new problem, already studied in [1], aims at determining the maximum value of D for which a feasible solution of the `KD-GPaths` problem exists. The new problem is named the Maximum Distance between $K \geq 2$ Paths Problem (`MG-KPaths` problem).

In this talk we address the `MG-KPaths` problem by means of both exact and approximated methods. Two new Integer Linear Programming models are introduced and compared theoretically, and a new improvement heuristic is described. Computational results are presented for the three approaches.

References

[1] A. de Sousa, D. Santos, and P. Monteiro. Determination of the minimum cost pair of D-geodiverse paths. In *DRCN 2017-Design of Reliable Communication Networks; 13th International Conference*, pages 1–8, 2017.

[2] M. T. Godinho and M. Pascoal. Implementation of geographic diversity in resilient telecommunication networks. In J. P. Almeida et al., editor, *Operational Research*, page To appear, Cham, 2024. Springer International Publishing.

[3] Committee on National Security Systems. Committee on national security systems (CNSS) glossary. `https://www.niap-ccevs.org/Ref/CNSSI_4009.pdf`, March 2022. Accessed on 2023-11-30.

[4] J. Rak, R. Girão-Silva, T. Gomes, G. Ellinas, B. Kantarci, and M. Tornatore. Disaster resilience of optical networks: State of the art, challenges, and opportunities. *Optical Switching and Networking*, 42:100619, 2021.

Survivable Traffic Grooming with Practical Constraints in Large-Scale Optical Network

Jianwei Niu
Theory Lab, 2012 Labs, Huawei Technologies, Co. Ltd
Shenzhen, Guangdong, China
niujianwei7@huawei.com

Junyan Liu
Theory Lab, 2012 Labs, Huawei Technologies, Co. Ltd
Sha Tin, Hong Kong SAR, China
liu.junyan@huawei.com

Fan Zhang
Theory Lab, 2012 Labs, Huawei Technologies, Co. Ltd
Sha Tin, Hong Kong SAR, China
zhang.fan2@huawei.com

Fabo Sun
Optical Product Research Department, Huawei Technologies, Co. Ltd
Dongguan, Guangdong, China
sunfabo@huawei.com

Kerong Yan
Optical Product Research Department, Huawei Technologies, Co. Ltd
Dongguan, Guangdong, China
yankerong@huawei.com

Junqi Ma
Optical Product Research Department, Huawei Technologies, Co. Ltd
Dongguan, Guangdong, China
mark.majunqi@huawei.com

Jie Sun
Theory Lab, 2012 Labs, Huawei Technologies, Co. Ltd
Sha Tin, Hong Kong SAR, China
j.sun@huawei.com

Tieyong Zeng
Department of Mathematics, The Chinese University of Hong Kong
Sha Tin, Hong Kong SAR, China
zeng@math.cuhk.edu.hk

ABSTRACT

This paper investigates the problem of the survivable traffic grooming, routing, and wavelength assignment (GRWA) subject to the no-loop and optical channel (OCh) length constraints. To the best of our knowledge, this complex problem has not been previously studied in the literature. We propose a novel length-constraint and no-loop Dijkstra algorithm, which helps to compute the primary and backup paths for traffic demands based on the augmented-layer graph (ALG). We conduct numerical experiments on both small and large-scale networks and demonstrate the superior efficiency of our approach compared to the results obtained from the integer linear programming formulation, which is solved by the SCIP solver.

KEYWORDS

Survivable Traffic Grooming, No-loop Constraint, OCh Length Constraint, Augmented-Layer Graph, LCNL-Dijkstra Algorithm, Tabu Search

1 INTRODUCTION

The wavelength division multiplexing (WDM) [13],[15] in an optical network is a widely used technology that multiplexes a number of traffic demands onto a single fiber by using different wavelengths. To further improve the bandwidth utilization of the WDM network, the traffic grooming technique has been proposed. The traffic grooming technique packs different traffic demands onto a wavelength using optical-electrical-optical (OEO) converters. Therefore, it can significantly enhance the fiber capacity and pave the way for the development of optical networks. In the WDM network with the traffic grooming, an optical channel (OCh) is a lightpath, which consists of a collection of consecutive fibers that share the same wavelength channel. An OCh originates and terminates at stations with OEO converters.

In an optical network, the failure of a single link can disrupt numerous OChs and connections. Therefore, the survivability in the optical network, which assigns not only the primary path but also the backup path for traffic demands, has been developed as referenced in [13], where the authors have introduced a comprehensive set of protection schemes, including the dedicated link protection, the shared link protection, the dedicated path protection, and the shared path protection. For the high utility of resources, we focus on the shared path-based protection scheme.

In the context of the industry, unnecessary resource wastage, transmission delay, and signal attenuation are intolerable to enterprise customers. Hence, we must consider two important constraints, including the no-loop constraint and the Och length constraint. To further explain, the no-loop constraint also called the simple path constraint in [10] is that the physical path for each demand does not contain any cycles to avoid unnecessary transmission delays. Moreover, we need to consider the length constraint because of the signal attenuation that occurs within an OCh, and the total attenuation or loss is and nonlinearly related to the distance of fibers composing the OCh [7]. To model the attenuation more conveniently, we could set a threshold for the length of an och to represent an accepted attenuation degree.

Due to the high cost of OEO converters and the fact that each OCh requires two converters, our objective is to minimize the number of OChs needed to satisfy a given set of traffic demands while adhering to the shared path protection scheme under the no-loop constraint and the OCh length constraint. We denote the above problem as the survivable multi-constraint traffic grooming (SMTG) problem. Numerous studies have demonstrated a strong interest

in the area of the survivable traffic grooming. Authors of [1, 6, 12] have proposed different survivable schemes and objectives with no constraints. In the work of Naser et al. [9], a delay-constrained and survivable scheme is proposed, but it cannot apply to our problem as it focuses on minimizing delay time and resource utilization.

In order to solve the SMTG problem, an integer programming problem (ILP) is formulated. However, obtaining the exact solution is challenging since the scale of variables and constraints in the problem can be as large as billions or even more. As a result, we present a novel heuristic algorithm, specifically the augmented-layer graph-based heuristic algorithm, herein referred to as the SMTG-ALGB algorithm. The main framework comprises the construction of the augmented-layer graph (ALG), the primary grooming procedure, the primary tabu search algorithm, the backup grooming procedure, and the backup tabu search algorithm. A crucial component of our grooming algorithm is the length-constraint and no-loop Dijkstra (LCNL-Dijkstra) algorithm based on the ALG, which efficiently tackles the constraints in the SMTG problem. Ultimately, we performed multiple experiments to compare the results of both the ILP formulation and the heuristic algorithm on two artificially-constructed networks and two large-scale, practical networks. The results of our experiments indicate that our novel heuristic algorithm is notably efficient and effective, even when confronted with extensive networks containing hundreds of demands, nodes, and links.

2 PROBLEM STATEMENT

The two-layer network structure[5, 8] models the entire WDM network, consisting of both operational OChs and nodes with OEO converting ability. The physical layer represents the original network, including the physical links and nodes, while the virtual layer models the virtual nodes with OEO converting ability, OEO converters installed on the virtual nodes and established OChs. In Fig.1, an OCh connecting the virtual nodes 1 and 3 is routed by physical nodes 1, 4, and 3 with the red wavelength in the physical layer. From the perspective of the two-layer network, we can formulate the ILP for the SMTG problem below.

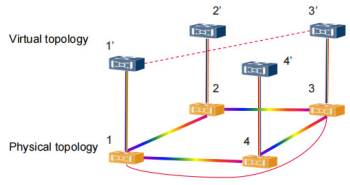


Figure 1: Two-layer network example

2.1 Notation used

In our formulations, we will use the following symbols and variables in TABLE 1 and TABLE 2.

Table 1: Notations

Symbols	Description
V_1	The set of physical nodes in the network.
E	The set of bidirectional physical links.
$E(v_1)$	The set of physical links whose nodes contains v_1.
Λ	The set of available wavelengths, e.g. $\{1, 2, \cdots, 80\}$.
V_2	The set of virtual nodes.
P_{ij}	The set of physical paths connecting virtual nodes i and j, where the overall physical length of each element of P_{ij} adheres to the length constraint.
p, P	The physical path and the whole set, $p \in P = \cup P_{ij}$.
h_e^p	$h_e^p = 1$ if a path p uses the link e; $h_e^p = 0$ if not.
$r^+(p), r^-(p)$	Two directional paths contained in p.
R	The set of directional paths.
$InR(v_2)$	The set of directional paths terminating at $v_2 \in V_2$.
$OutR(v_2)$	The set of directional paths originating at $v_2 \in V_2$.
(p, λ)	Bidirectional OCh with path p and wavelength λ.
(r, λ)	Directional OCh with path r and wavelength λ.
d	A bidirectional demand.
D	The set of bidirectional demands.
s_d	The original node of the demand d.
t_d	The terminal node of the demand d.
B_d	The bandwidth of the demand d.
b_j^d	$b_j^d = -1$, if $j = s_d$; $b_j^d = 1$, if $j = t_d$; $b_j^d = 0$, otherwise.
C	The bandwidth for a wavelength channel.

Table 2: Variables

Var.	Description
$z^{p,\lambda}$	$z^{p,\lambda} = 1$, if OCh (p, λ) is established; $z^{p,\lambda} = 0$, otherwise.
$x_{p,\lambda}^d$	$x_{p,\lambda}^d = 1$, if the primary path of the bidirectional demand d uses the OCh (p, λ); $x_{p,\lambda}^d = 1$, otherwise.
$x_{r,\lambda}^d$	$x_{r,\lambda}^d = 1$, if the primary path of a directional demand which is contained in the bidirectional demand d and from s_d to t_d uses the directional OCh (r, λ); $x_{r,\lambda}^d = 0$, otherwise.
$y_{p,\lambda}^d$	$y_{p,\lambda}^d = 1$, if the backup path of a bidirectional demand d uses the OCh (p, λ). $y_{p,\lambda}^d = 0$, otherwise.
$y_{r,\lambda}^d$	$y_{r,\lambda}^d = 1$, if the backup path of a directional demand which is contained in the bidirectional demand d and from s_d to t_d uses the directional OCh (r, λ); $y_{r,\lambda}^d = 0$, otherwise.
$w_{p,\lambda}^{d,e}$	$w_{p,\lambda}^{d,e} = 1$, if physical link e has a failure, the demand d is influenced and its backup path uses OCh (p, λ); $w_{p,\lambda}^{d,e} = 0$, otherwise.

2.2 ILP formulation for minimizing the OCh usage

Objective function:

$$\underset{\substack{\{z^{p,\lambda}\},\{x_{p,\lambda}^d\},\{x_{r,\lambda}^d\} \\ \{y_{p,\lambda}^d\},\{y_{r,\lambda}^d\},\{w_{p,\lambda}^{d,e}\}}}{\text{Minimize}} : \sum_{p \in P} \sum_{\lambda \in \Lambda} z^{p,\lambda} \tag{1}$$

Constraints:

$$\sum_{\substack{r\in InR(v_2)\\ \lambda\in\Lambda}} x^d_{r,\lambda} - \sum_{\substack{r\in OutR(v_2)\\ \lambda\in\Lambda}} x^d_{r,\lambda} = b^d_{v_2}, \forall d\in D, v_2\in V_2,$$
$$x^d_{p,\lambda} = x^d_{r^+(p),\lambda} + x^d_{r^-(p),\lambda}, \forall d\in D, p\in P, \lambda\in\Lambda. \tag{2}$$

$$\sum_{\substack{r\in InR(v_2)\\ \lambda\in\Lambda}} y^d_{r,\lambda} - \sum_{\substack{r\in OutR(v_2)\\ \lambda\in\Lambda}} y^d_{r,\lambda} = b^d_{v_2}, \forall d\in D, v_2\in V_2,$$
$$y^d_{p,\lambda} = y^d_{r^+(p),\lambda} + y^d_{r^-(p),\lambda}, \forall d\in D, p\in P, \lambda\in\Lambda. \tag{3}$$

$$\sum_{\lambda\in\Lambda}\sum_{p\in P}\sum_{e\in E(v_1)} h^p_e x^d_{p,\lambda} \le 2, \forall d\in D, v_1\in V_1, \tag{4}$$

$$\sum_{\lambda\in\Lambda}\sum_{p\in P}\sum_{e\in E(v_1)} h^p_e y^d_{p,\lambda} \le 2, \forall d\in D, v_1\in V_1, \tag{5}$$

$$\sum_{\lambda\in\Lambda}\sum_{p\in P} h^p_e x^d_{p,\lambda} + \sum_{\lambda\in\Lambda}\sum_{p\in P} h^p_e y^d_{p,\lambda} \le 1 \tag{6}$$

for $\forall d\in D, e\in E$,

$$\sum_{p\in P} h^p_e z^{p,\lambda} \le 1, \forall e\in E, \lambda\in\Lambda, \tag{7}$$

$$\sum_{d\in D} B_d(1-h^p_e)x^d_{p,\lambda} + \sum_{d\in D} B_d w^{d,e}_{p,\lambda} \le C(1-h^p_e)z^{p,\lambda} \tag{8}$$

for $\forall e\in E, p\in P, \lambda\in\Lambda$,

$$w^{d,e}_{p,\lambda} \le \sum_{p'\in P}\sum_{\lambda'\in\Lambda} h^{p'}_e x^d_{p',\lambda'},\ w^{d,e}_{p,\lambda} \le y^d_{p,\lambda},$$
$$w^{d,e}_{p,\lambda} \ge \left(\sum_{p'\in P}\sum_{\lambda'\in\Lambda} h^{p'}_e x^d_{p',\lambda'}\right) + y^d_{p,\lambda} - 1 \tag{9}$$

for $\forall e\in E, d\in D, p\in P, \lambda\in\Lambda$.

The objective function (1) will count the total used OChs. Flow constraints (2) and (3) ensure that the paths of demands are feasible in the virtual layer. No-loop constraints (4) and (5) are required for physical paths of demands. Even though the OCh paths of demands have no loops, the whole physical path can still have them due to the diverse routing of each OCh. In order to ensure that the primary and backup paths are disjoint in the physical layer, constraint (6) is imposed. Furthermore, different OChs must utilize distinct wavelength channels on the same link, as represented by constraint (7). To ensure that traffic demands do not exceed the bandwidth allotted to each OCh, constraint (8) is included. The linearization of the auxiliary variables is given by constraint (9) according to the description of $w^{d,e}_{p,\lambda}$.

Even though the ILP is a precise formulation and will give an exact solution, practical difficulties may arise when dealing with a large-scale network due to numerous variables and constraints. As an illustration, in a network comprised of 65 nodes, 96 links, 20 wavelength channels, and 122 demands, the combined number of variables and constraints generated by the associated ILP may exceed five billion.

3 HEURISTIC APPROACH

The general survivable traffic GRWA is NP-hard [12], and the additional constraints imposed in our proposed SMTG problem, namely the no-loop and OCh length constraints, increase its computational complexity. In light of this, the ILP formulation also suggests that the problem will be challenging to solve optimally. Consequently, a heuristic approach becomes necessary, and we devise the SMTG-ALGB algorithm.

In the upcoming subsections, any symbols that have not been given explicit definitions can be referenced in TABLE 3.

Table 3: Notations

Symbols	Description
G_p	The physical topology.
G_a	The auxiliary graph.
L	The length constraint for the OCh.
π_w	Weight for WLEs.
π_t	Weight for TrEs.
P_d	The path information for demand d, which only contains OChs. We use P^p_d for the primary path and P^b_d for the backup path.
$H(\cdot)$	Get physical hops for a OCh.
$W(\cdot)$	Get the weight of an edge in ALG.
Num_{opt}	The number of nodes in the physical layer.

3.1 Main framework

To begin with, the main framework will be presented, where the primary process is denoted by 'p' while the backup process is denoted by 'b'. We also simplify inputs and outputs for the functions in SMTG-ALGB.

Algorithm 1: SMTG-ALGB

Input: Physical topology G_p, wavelength set Λ, length constraint L, demands D and so on

Output: G_a, primary paths $\{P^p_d\}$ and backup paths $\{P^b_d\}$ for D

Initialize the ALG G_a according to G_p and Λ;
Sort the demands in a non-increasing order ;
$G_a, \{P^p_d\} \leftarrow$ Grooming(G_a, D, 'p') ;
$G_a, \{P^p_d\} \leftarrow$ Tabu-search(G_a, D, $\{P^p_d\}$,'p') ;
Sort the demands in a non-decreasing order ;
$G_a, \{P^b_d\} \leftarrow$ Grooming(G_a, D, 'b') ;
$G_a, \{P^b_d\} \leftarrow$ Tabu-search(G_a, D, $\{P^b_d\}$, 'b') ;

3.2 Construction of the augmented-layer graph

Heuristic algorithms for solving GRWA problems involve the graph construction based on the previous two-layer network structure as the first step. For example, the link bundled auxiliary graph (LBAG) in [11] can be effective with traditional grooming and routing algorithms, albeit with some challenges such as wavelength consistency constraints that need special consideration. While Layered-AG (LAG) [14] provides clear representations of all processes, the

abundance of edges, such as mux, demux, and converter edges, may complicate the representation and hinder the efficiency of the grooming algorithm. To address this concern and incorporate the advantages of both approaches, we propose the ALG model. An illustrative example of the ALG corresponding to Fig. 1 is presented in Fig. 2, where we assume that there are two available wavelengths. In order to provide a comprehensive understanding of ALG, a detailed explanation is provided below.

3.2.1 Edges in the ALG. In ALG, we classify the edges into three categories as follows.

- WLE: Assuming that each physical link has $|\Lambda|$ wavelengths, $|\Lambda|$ wavelength layers will be generated accordingly. The notation $E_w(i, j, m)$ denotes the wavelength edge that forms the connection between nodes i and j in the m^{th} wavelength layer, while its weight is represented by $W_w(i, j, m)$.
- TrE: Edges representing transmitters and receivers are a crucial component of ALG. The notation $E_t(i, \lambda_m)$ is used to denote the transceiver edges connecting the virtual layer to the m^{th} wavelength layer, with $W_t(i, \lambda_m)$ representing their respective weights.
- OCh: Edges within the virtual layer are crucial in the establishment of an OCh, utilizing designated physical links and a specific wavelength. If an OCh denoted as $E_o(i, j, \lambda_m)$ is established to interconnect nodes i and j using wavelength channel λ_m, the corresponding wavelength edges in the m^{th} wavelength-layer will be removed. The weight for $E_o(i, j, \lambda_m)$ is characterized by $W_o(i, j, \lambda_m)$. The number of physical hops when traversing an OCh through the physical layer can be represented by $H(i, j, \lambda_m)$.

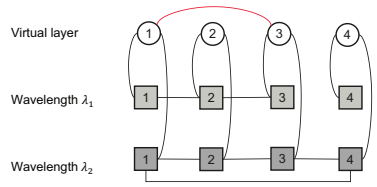


Figure 2: ALG

3.3 Grooming algorithm based on the ALG

After the construction of ALG, we can groom the traffic demands one by one based on it. For example, if we want to establish a connection between nodes 3 and 4, we can find a possible path $E_t(3, \lambda_2)$-$E_w(3, 4, 2)$-$E_t(4, \lambda_2)$, after which we should add an OCh $E_o(3, 4, \lambda_2)$ and delete $E_w(3, 4, 2)$ in the ALG. Basically, there exist the primary and the backup routing algorithms, and these two versions exhibit minor differences in their approach towards updating the graph for the LCNL-Dijkstra.

3.3.1 Grooming algorithm. The main grooming procedure is shown in Algorithm 2. It should be noted that after the LCNL-Dijkstra algorithm section, we perform iterations over both the edge set and the node set of the path. However, this path may not necessarily be the final OCh path for the given demand. We need to transfer the wavelength sub-paths with TrEs into a newly established OCh and save it in P_d. Besides, WLEs used by the new OCh will be deleted from G_a.

Algorithm 2: Grooming

Input: G_a, D, the type of grooming T, 'p' for primary and 'b' backup.
Output: G_a, $\{P_d\}$
for *all* $d \in D$ **do**
  if $T ==$ 'p' **then**
    Delete the OChs where $B_a < B_d$;
  else if $T ==$ 'b' **then**
    Delete all WLEs and OChs wich have joint parts with the primary path of d ;
    Delete the OCh edges where $B_s + B_a < B_d$;
    Update G_a ;
  end
  if *LCNL-Dijkstra*(G_a, s_d,t_d, Num_{opt}, L, $Path$) **then**
    Get the returned edge set $Edge$;
    for *all* $e \in Edge$ **do**
      Determine the type of e ;
      Update G_a and P_d ;
    end
  else
    Failed grooming process for d ;
  end
  Restore the deleted edges before LCNL-Dijkstra ;
end

3.3.2 LCNL-Dijkstra algorithm. In the LCNL-Dijkstra shown in Algorithm 3, we endeavor to explicate the variables and functions that will be mentioned. $dist[\cdot]$ denotes the distance from a node to the source node, which is determined by the weights in G_a. Additionally, $len[\cdot]$ keeps track of the physical length of the most recent contiguous subpath that exists within a single wavelength layer. Furthermore, $prev[\cdot]$ identifies the parent node and edge of a given node. It should be noted that the nodes with smaller distances are prioritized in Q, which serves as a priority queue. The vector $omsNodes[v]$ stores all non-repeating physical nodes from the original node s_d to v. The function $\texttt{CheckNoOverlap}(omsNodes[u], l)$ returns $true$ if edge l has no repeating physical nodes with $omsNodes[u]$ except the joint node; it returns $false$ otherwise. The function $\texttt{CheckLength}(L, u, l)$ returns $true$ if $len[u] + \texttt{OmsLen}(l) \geq L * (1 + \texttt{floor}(len[u]/L))$ and false otherwise. Here, $\texttt{OmsLen}(\cdot)$ is used to denote the physical length of a wavelength edge. If node u is relaxed with edge l, and the resulting subpath exceeds the designated limit, this sub-path will be flagged for further splitting. Its weight may be subject to increase during the subsequent $\texttt{Relax}()$ part if required.

Next, we briefly describe the relaxing part in Algorithm 4, where we ignore some necessary input variables.

Within the Algorithm 4, we make use of the $\texttt{ceil}()$ function when $newOch \neq 0$. It aims to make $len[\cdot]$ accurately reflect the true number of OCh if we consider the physical length constraint. Once a path has been determined, the function $\texttt{Split}()$ will be employed to examine the TrEs and the physical length of WLEs between

Algorithm 3: LCNL-Dijkstra

Input: Auxiliary graph G_a, original node s_d, terminal node t_d, Num_{opt}, L, $Path$ (including $Edge$ and $Node$)
Output: $flag$
$flag \leftarrow false$;
$newOch \leftarrow 0$;
for *each node* $v \in G_a$ **do**
  $dist[v] \leftarrow INF, len[v] \leftarrow 0$;
  $prev[v].node \leftarrow -1, prev[v].edge \leftarrow -1$;
end
$dist[s_d] \leftarrow 0$;
Q.insert(s_d) ;
$omsNodes[s_d]$.push(mod(s_d, Num_{opt})) ;
while $Q \neq \emptyset$ **do**
  $u \leftarrow Q$.pop() ;
  if $u == t_d$ **then**
    $flag \leftarrow true$;
    break ;
  end
  for *each* l *from adjacent edges of* u **do**
    if CheckNoOverlap($omsNodes[u], l$) **then**
      continue ;
    end
    if CheckLength(len, L, u, l) **then**
      $newOch \leftarrow 1$;
    end
    Get the other endpoint of l and denoted as v ;
    Relax($dist, u, v, l, newOch, \pi_t$) ;
  end
end
if $flag$ **then**
  Generate a complete path $tmpPath$ based on $prev$;
  $Path \leftarrow$ Split($tmpPath$) ;
end

Algorithm 4: Relax

Input: $dist, u, v, l, newOch, \pi_t$
if $dist[v] \neq INF$ *and*
 $dist[u] +$ W(l) $+ 2 * newOch * \pi_t < dist[v]$ **then**
  $dist[v] \leftarrow dist[u] + W(l) + 2 * newOch * \pi_t$;
  $prev[v].node \leftarrow u, prev[v].edge \leftarrow l$;
  Q.insert(v) ;
end
if $newOch \neq 0$ **then**
  $len[v] \leftarrow$ ceil($len[u]/L$) + OmsLen(l) ;
else if $l \in E_w$ **then**
  $len[v] \leftarrow len[u] +$ OmsLen(l) ;
else if $l \in E_t$ **then**
  $len[v] \leftarrow 0$;
end
Update $len[v]$ and $omsNodes[v]$;

them. In the event that a decision is made to split the corresponding wavelength subpath, the splitting node can be chosen flexibly, taking into account factors such as the reusability and availability of wavelength channels. It is important to note, however, that the number of splits should not exceed ceil($len[u]/L$).

3.3.3 Grooming policy and weights setting. Before we talk about the setting of weight, we go through the bandwidth of a channel first. The bandwidth for a wavelength channel can be categorized into free bandwidth B_a, dedicated bandwidth, and spare bandwidth. The spare bandwidth can further be divided into sharable spare bandwidth B_s and non-sharable bandwidth.

In our survivable protection scheme, we will identify a risk-disjoint backup path for the primary path of each demand, taking link failure scenarios into consideration. Given that the backup routing process entails the utilization of sharable bandwidth, it is judicious to prescribe distinct weight settings for the primary and backup grooming, respectively. Besides, In order to minimize the usage of OChs, which constitute the cost component of our task, it is imperative to enhance the sharable ability of the routing algorithm. Considering the aforementioned background and the insights provided in [12], we propose the following weight setting and diverse grooming processes can be delineated by varying the weights in ALG.

Table 4: Weight setting for primary and backup routing

Primary, edge e	WLE	$W_w(e) = \begin{cases} \pi_w, & \text{if } E_w(e) \text{ is unused} \\ \infty, & \text{otherwise} \end{cases}$
	TrE	$W_t(e) = \pi_t$
	OCh	$W_o(e) = \begin{cases} \alpha \times H(e), & \text{if } B_a \geq B_d \\ \infty, & \text{otherwise} \end{cases}$, where $0 < \alpha \leq 1$ and α is tunable.
Backup, edge e	WLE	$W_w(e) = \begin{cases} \pi_w, & \text{if } E_w(e) \text{ is unused} \\ \infty, & \text{otherwise} \end{cases}$
	TrE	$W_t(e) = \pi_t$
	OCh	$W_o(e) = \begin{cases} \alpha \times H(e), & \text{if } B_s \geq B_d \\ (\beta + \alpha(1-\beta)) \times H(e), & \text{if } B_s < B_d \leq B_s + B_a \\ \infty, & \text{otherwise} \end{cases}$, where $0 < \alpha \leq 1$ and $\beta = (B_d - B_s)/B_d$.

3.4 Tabu Search Algorithm

Tabu search is a widely employed meta-heuristic method. For a comprehensive understanding of it, interested readers are advised to refer to [4]. Our proposed tabu search algorithm entails the removal of a pre-existing OCh and subsequently rerouting its associated demands to other extant OChs, the complete removal of an OCh may not be a feasible option in every instance. Consequently, we adopt a strategy which accepts a move that either preserves the extant number of OChs or augments it. We delineate the following significant measures.

Firstly, thoroughly examining every OCh in each iteration would be exceedingly time-consuming. To overcome this issue, we propose generating a relatively small set of OCh edges, then selecting a

single edge for manipulation in each subsequent iteration. Secondly, to enhance the algorithm's flexibility, we adopt a hybrid approach involving both fixed and variable tabu steps. Moreover, we integrate the OCh into the tabu list and remove the corresponding OCh from the list after the later tabu steps. Thirdly, when it is necessary to remove an OCh and re-groom the associated demands, it is probable that they will be assigned to the same physical links that were previously allocated for the deleted OCh. To circumvent this issue, the weights of all relevant WLEs can be raised after removing the OCh. Lastly, an important measure is inspired by previous work in [2] and [3], where a cost function is defined for each move of re-grooming demand. In the end, we opt to choose the slightest cost move. We omit the intricate details, which resemble those of the preceding algorithm.

3.5 Experiments

In this section, we provide the details of our experiments involving four distinct networks. The first two are synthetic small-scale networks, while the latter two correspond to large practical networks of cities in China. Table 5 presents the statistical characteristics of networks and demands. Specifically, in the network under consideration, each physical link can accommodate up to 80 wavelengths, and each wavelength channel has a bandwidth of 100G. The traffic demands being considered have diverse bandwidth requirements, e.g. $\{2.5, 10, 100\}$G; The physical length of links in Network 1 and 2 varies between 1, 2, and 3 km, whereas in Network 3, it is mainly 1 km. The physical length of links in Network 4 is much longer, from 1 to 800 km. For the weights of WLEs and TrEs in our heuristic algorithm, we set $\pi_w = 1$ and $\pi_t = 10$, respectively. Besides, we limit each vertex pair to a maximum of 20 paths for the computation of the ILP. Our experiments are running in an Ubuntu server with an Intel(R) Xeon(R) Gold 6266C CPU of 3.00GHz and a RAM of 64GB.

Table 5: Network topology

Network	$\lvert V_1\rvert$	$\lvert E\rvert$	$\lvert V_2\rvert$	$\lvert\Lambda\rvert$	$\lvert D\rvert$	L	$\lvert P\rvert$
1	4	4	4	3	4	10	8
2	6	8	6	4	6	10	73
3	65	96	65	20	122	6	6429
4	227	409	227	40	578	1000	33537

Table 6 presents the ILP and heuristic results (OCh number and time consumption). For the heuristic algorithm with tabu search, we achieve the same optimal values as we get from the ILP for small-scale networks. Moreover, the time consumption of the heuristic algorithms for small-scale networks is almost negligible. Besides, it was not possible to construct the ILP using the SCIP solver for networks 3 and 4, since the variables and constraints involved in network 3 are estimated to be approximately 1.5 billion and 4.5 billion, and they will be large for the network 4. However, even when conducting hundreds of rounds of tabu search on large-scale networks, the heuristic remains effective and highly competitive.

4 CONCLUSION

In this paper, we have presented the SMTG-ALGB algorithm, which tackles the survivable GRWA problem with the no-loop and OCh length constraints and aims to minimize the usage of OCHs. We first construct the ALG and then devise grooming algorithms. Since our primary and backup grooming algorithms' effectiveness is highly dependent on the weights configured within the ALG, a detailed exposition on weight setting is also provided. Lastly, a tabu search algorithm is proposed to further improve the grooming result.

The experiments have demonstrated that our algorithm can achieve the optimal solution for a small network and provide effective grooming results for large practical networks, while the ILP-based method can only be applied to small networks.

Table 6: Overall Results

Network	ILP		Our algorithm (without TS)		Our algorithm (with TS)	
	OCh	Time	OCh	Time	OCh	Time
1	4	1s	5	0.001s	4	0.001s
2	8	4221s	9	0.001s	8	0.003s
3	None	None	113	0.2s	102	20s
4	None	None	1104	8s	1060	766s

REFERENCES

[1] Asima Bhattacharya, Malabika Sarder, Monish Chatterjee, and Diganta Saha. 2020. Efficient integration of logical topology design and survivable traffic grooming for throughput enhancement in WDM optical networks. *IEEE Access* 8 (2020), 58155–58170.
[2] Ilfat Ghamlouche, Teodor Gabriel Crainic, and Michel Gendreau. 2003. Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Oper. Res.* 51, 4 (Aug. 2003), 655–667.
[3] Ilfat Ghamlouche, Teodor Gabriel Crainic, and Michel Gendreau. 2004. Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Ann. Oper. Res.* 131, 1-4 (Oct. 2004), 109–133.
[4] Fred Glover and Manuel Laguna. 1998. *Tabu search*. Springer.
[5] Arunita Jaekel, Ataul Bari, and Subir Bandyopadhyay. 2008. Resilient traffic grooming for WDM networks. *J. Opt. Netw.* 7, 5 (May 2008), 378.
[6] Arunita Jaekel, Ataul Bari, Quazi Rahman, Ying Chen, Subir Bandyopadhyay, and Yash Aneja. 2012. Resource efficient network design and traffic grooming strategy with guaranteed survivability. *Optical Switching and Networking* 9, 4 (2012), 271–285.
[7] Gerd Keiser. 2021. *Optical Signal Attenuation and Dispersion.* Springer Singapore, Singapore, 93–145. https://doi.org/10.1007/978-981-33-4665-9_3
[8] Junyan Liu, Fan Zhang, Kerong Yan, Junqi Ma, and Bo Bai. 2022. An ILP-based approach for dual-layer optimization in large-scale optical networks. In *2022 31st Wireless and Optical Communications Conference (WOCC).* IEEE.
[9] Hassan Naser and Ming Gong. 2007. Link-disjoint shortest-delay path-pair computation algorithms for shared mesh restoration networks. In *2007 12th IEEE Symposium on Computers and Communications.* IEEE, 269–274.
[10] Xinyun Wu, Tao Ye, Qi Guo, and Zhipeng Lü. 2015. GRASP for traffic grooming and routing with simple path constraints in WDM mesh networks. *Comput. Netw.* 86 (July 2015), 27–39.
[11] Wang Yao and B Ramamurthy. 2005. A link bundled auxiliary graph model for constrained dynamic traffic grooming in WDM mesh networks. *IEEE J. Sel. Areas Commun.* 23, 8 (Aug. 2005), 1542–1555.
[12] Wang Yao and B Ramamurthy. 2005. Survivable traffic grooming with path protection at the connection level in WDM mesh networks. *J. Lightwave Technol.* 23, 10 (Oct. 2005), 2846–2853.
[13] Dongyun Zhou and Suresh Subramaniam. 2000. Survivability in optical networks. *IEEE network* 14, 6 (2000), 16–23.
[14] Hongyue Zhu, Hui Zang, Keyao Zhu, and B Mukherjee. 2003. A novel generic graph model for traffic grooming in heterogeneous WDM mesh networks. *IEEE ACM Trans. Netw.* 11, 2 (April 2003), 285–299.
[15] K Zhu and B Mukherjee. 2003. A review of traffic grooming in WDM optical networks: Architectures and challenges. *Optical Networks Magazine* 4, 2 (2003), 55–64.

Feasibility of Near Term Quantum Optimisation of Communication Networks

Catherine White[1]

[1]BT Research, Adastral Park, Ipswich, UK , ✉ catherine.white@bt.com

Present day and near-term Quantum Computers are of the Noisy, Intermediate-Scale Quantum (NISQ)[1] type: they are not fully error corrected against imperfections (such as noise, decoherence, stray coupling and open boundary effects) in the physical encoding and processing of quantum information, and they have a fairly limited number of qubits per Quantum Processing Unit (QPU). In this class of machines, we may include present-day quantum annealers, as well as gate-based quantum computers. For the latter type of system, hybrid variational algorithms such as Variational Quantum Eigensolver (VQE)[2] have been invented.

Real world optimisation problems in the field of Communication Networks may fall into the class NP, particularly when constrained to discrete (integer) solutions. Some real world scenarios do present instances which are practically computationally hard. For example, when multiple different demands are placed on a network such that it is close to capacity, or when other constraints and optimisation goals also exist (and are hard to satisfy), such as absolute limits on latency, or a target of cost optimisation. I will present examples of such hard instances, and discuss their characteristics.

All though noise cannot be fully compensated in the NISQ era, there is a range of evidence and opinion about whether such systems may still offer some quantum advantage for problem solving (e.g. optimisation or satisfaction problems) for practical systems. Such systems may be divided into quantum systems (optimisation of quantum information) and classical systems (optimisation of classical information). The optimisation of communication networks primarily falls into the latter category (although with development in quantum communication networks, examples of the former might exist.)

I will provide an update on ongoing work at BT to explore the potential, including highlighting a collaborative UKRI project into the feasibility of NISQ era advantages to real world communication network optimisation. Current work, and a discussion of future work and challenges will be presented.

References

[1] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[2] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1):4213, July 2014.

INOC - 2024

Session Session 1B: Robust Optimization

Monday 11 March 2024, 11:00-12:30

Q012

Robust optimization for the Segment Routing Traffic Engineering Problem

Hugo Callebaut[1, 3], Jérôme De Boeck[2], and Bernard Fortz[2, 1, 3]

[1]Départment d'Informatique, Université libre de Bruxelles, Brussels, 1050, Belgium
[2]HEC Liège, Management school of the University of Liège, Liège, 4000, Belgium
[3]INOCS, INRIA, 59650 Villeneuve d'Ascq, France

1 Introduction

Segment Routing (SR) introduces a flexible approach to IP network routing, addressing limitations present in conventional protocols. SR is to be used on top of an existing routing protocol and allows traffic to take detours through nodes and links called node segments and adjacency segments.

The segment routing traffic engineering problem (SRTEP) uses shortest path based underlying protocols such as OSPF and assumes that the link weights are set. Only the SR-paths can be changed, and there can only be one unique SR-path between two (ordered) pairs of nodes. The goal of the SRTEP is to minimise the maximum link utilisation with respect to one traffic matrix (TM) [4].

This paper focuses on robust optimization within the SRTEP. Unlike traditional approaches that rely on a single traffic matrix for optimization, we deal with an infinite set of matrices defined by linear constraints. The objective is to identify routing strategies robust to the worst-case scenario within this set.

We build upon prior work by introducing new linear formulations inspired by Bertsimas and Sim [1], exploring their efficiency in comparison to exhaustive enumeration methods and constraint generation methods. Our findings reveal promising results, showcasing the practicality and speed advantages of our approach.

Keywords: Segment Routing, Traffic Engineering, Robust Optimization, Mixed Integer Linear Problem.

2 Methodology

The goal of robust optimization is to obtain the best possible solution in the worst case but this is generally seen as rather conservative. In the SRTEP if only one traffic matrix was used and we assumed that for each measure there was an error rate of for example 20%, then the worst possible matrix would be the one where all demands are at 120% of their initial capacity. One can then prove that in the case of the SRTEP, because the maximum link utilisation (MLU) is minimised, the resulting paths would not change and the solution of the robust problem would be 20% worse than the original problem.

Bertsimas and Sim tackled this problem in [1]. They assume that even though all measurements could be off by some amount, it is extremely unlikely that all measurements take their worst value at the same time. They then introduce two parameters to their formulation Γ that indicates the amount of values in the traffic matrix that are allowed to deviate from their original value and a vector e_{st} that for each demand $\mathbf{D}(s,t)$ indicates how much this value can deviate from the original demand. For conciseness we will assume that Γ is a natural number and the values of $e_{s,t}$ will be proportional to $\mathbf{D}(s,t)$ but none of these assumptions need to be true.

Due to space limitations we provide now a high level overview of our approach. The inputs to our problem are a weighted and capacitated graph $G(N,A)$ with node set N and directed arc set A, a Traffic Matrix $\mathbf{D}$, Γ and a maximum deviation parameter which defines $e_{s,t}$ using $\mathbf{D}$.

We first find a continuous linear formulation that maximises the additional load for each edge. The dual of this problem can then be derived and by strong duality theorem it will have the same optimal

value. This new minimisation problem can then be integrated in the formulation of the original SRTEP problem, resulting in the minimisation of the maximal MLU on each edge with respect to our infinite set of traffic matrices. This new formulation has $|A| + |A| * |\mathbf{D}|$ new variables and $|A| * |\mathbf{D}|$ new constraints. The size of the problem is unaffected by the parameter Γ.

One should note that the formulation solved does not correspond to our original problem. This is because we maximise the MLU of each link individually which implies that the flow on each link is not conserved. Fortunately, because the minimisation of the MLU is used as objective function, the solution obtained will be the same for both problems. This can be proved by contradiction, starting from an optimal solution of one problem one can see that an optimal solution of the other problem cannot be worse than the optimal solution of the first problem.

Other methods to solve the robust problem exist and do not need to be limited to the MLU objective function. Since the traffic matrices are defined by linear constraints, it is possible to enumerate all extreme points of these constraints and give them as input to a modified formulation of the SRTEP accepting multiple TMs, this would amount to adding $\binom{|\mathbf{D}|}{\Gamma} \times |A|$ new constraints to the problem. Another option is to solve the SRTEP using one extreme TM. Generating the new worst TM with respect to the new solution, adding new constraints to add this TM to the problem and iterating this until no new worst TM is found. The solution found will then be optimal and this method is finite as a new extreme TM is generated at each step and there are a finite number of extreme TMs.

3 Results

We compared the dual approach, the enumeration of all extreme points and the iterative generation of TM. The parameters used are $\Gamma = 2$ and the maximum deviation is 100% meaning that at most two demands can change and their volume can up to double. The instances used come from Repetita [3] and used inverse capacity weights. We also used a preprocessing technique [2] to remove SR-paths from the formulation while still ensuring an optimal solution.

In Figure 1 we compare how many instances were solved optimally (on the y-axis) within a certain amount of time (on the x-axis) an see that the dual approach is generally the most efficient although the constraint generation approach is sometimes faster and the enumeration of all extreme points is always the worst option.

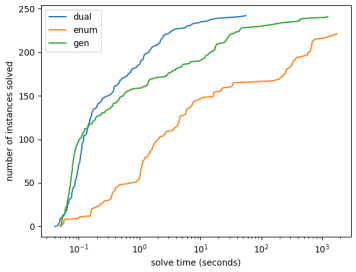


Figure 1: The cumulative number of instances solved over time.

References

[1] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98:49–71, 2003.

[2] H. Callebaut, J. De Boeck, and B. Fortz. Preprocessing for segment routing optimization. *Networks*, 82(4):459–478, 2023.

[3] S. Gay, P. Schaus, and S. Vissicchio. Repetita: Repeatable experiments for performance evaluation of traffic-engineering algorithms. *arXiv:1710.08665v1*, 2017.

[4] R. Hartert, S. Vissicchio, P. Schaus, O. Bonaventure, C. Filsfils, T. Telkamp, and P. François. A declarative and expressive approach to control forwarding paths in carrier-grade networks. *SIGCOMM*, 2015.

Two-Stage Robust b-matchings under uncertain capacities

Jenny Segschneider[1] and Arie M.C.A. Koster[2]

[1]Lehrstuhl II für Mathematik, RWTH Aachen, Aachen, Germany, ✉ segschneider@math2.rwth-aachen.de
[2]Lehrstuhl II für Mathematik, RWTH Aachen, Aachen, Germany, ✉ koster@math2.rwth-aachen.de

1 Introduction

The b-matching problem is a well-known variant of the famous matching problem with many applications. Given an undirected graph, each vertex v can be matched up to b_v times and edges can be used multiple times. It is solvable in polynomial time. Applications include matching workers or produce to customers, as in the Chinese Postman Problem, but it is also used in heuristics for other well-known Problems like TSP. In other applications, the capacity b_v has to be fulfilled exactly by a feasible matching. This variant of the problem is called a perfect b-matching problem. In theory, the capacity b_v is fixed and known for each vertex. However, in practice, the capacity is often subject to uncertainties, as workers get sick or customers drop out. In this talk, we present a new variant of both the b-matching and the perfect b-matching problem under uncertain capacity ensuring robustness, called the directed robust (perfect) b-matching problem, and analyze their complexity on different graph classes.

2 Related Work

Recent literature considering the matching problem under uncertain capacity mostly focused on online variants of the problem. This is due to an application linked to online advertisements, the Ad Allocation Problem. We refer to Mehta et al. [4] for more details on the topic. To the best of our knowledge, the research on the robust matching or b-matching problem is very sparse. There are only three publications considering variants of this problem, none of which consider the same problem we do. Katriel et al. [3] propose a randomized algorithm for the robust recoverable matching problem with uncertain costs. Housni et al. [2] consider a robust version of the ride-hailing problem where riders are matched to drivers. They propose a two stage model where the available drivers and a first batch of riders are known while the second batch of drivers are subject to uncertainty and only revealed in the second stage. Schmitz and Büsing [1] consider a version of the robust b-matching problem under consistent selection constraints. They also propose a two stage approach. In the first stage, the b-matching is only set on a given subset of all edges, while the b-matching on the remaining edges is set in the second stage when the scenario is known.

3 Problem Description and Summary of Results

An instance of the directed robust b-matching problem consists of a directed graph $D = (V, A)$ with arc value $c \in \mathbb{Z}^{|A|}$ and a set of scenarios $\mathcal{B}$ for the uncertain capacities on each vertex. The problem has two stages: first, we set a so-called pre-matching x_v for each vertex $v \in V$, the sum of the matching over all outgoing arcs. Second, after realization of the worst-case scenario, the b-matching with maximum value

satisfying both the pre-matching and the scenario is chosen. Formally, the problem is then given as

$$\max_{x \geq 0} \min_{b \in \mathcal{B}} \max_{m^b \geq 0} \sum_{a \in A} c_a m_a^b \tag{1}$$

$$\text{s.t.} \sum_{(v,w) \in A} m^b_{(v,w)} = x_v \qquad \forall v \in V \tag{2}$$

$$\sum_{(v,w) \in A} m^b_{(v,w)} + \sum_{(w,v) \in A} m^b_{(w,v)} \leq b_v \qquad \forall v \in V \tag{3}$$

$$m^b_a \in \mathbb{Z}_+ \qquad \forall a \in A \tag{4}$$

For the directed robust perfect b-matching problem, the matching in the second stage has to be a perfect b-matching. Thus, the matching constraints (3) have to be satisfied with equality.

Due to the two-stage approach, this problem has multiple practical applications. The problem originated from an application related to healthcare, where a schedule for administering vaccinations requiring two doses for each patient with limited, uncertain supply had to be devised. In this application, the pre-matching would set appointments for a first dose for each patient. This simplifies the planning of the complete schedule, as second doses can be planned for spontaneously in the presence of the patient. We refer to [5] for details. Furthermore, this problem can be used in applications linked to supply chain management problems, where limited goods have to be shipped to different locations with uncertain demand. An example would be a manufacturer selling a limited item in several stores. They have to decide how many items to send to each store while the number of buyers in each store is uncertain. The pre-matching then fixes the delivered goods for each store while the matching models the customers picking one of their preferred stores. Depending on the model, robustness would then ensure that each customer is satisfied or that all items are sold in each scenario.

Both variants of the problem turn out to be strongly NP-hard on arbitrary graphs with $|\mathcal{B}| \in \mathcal{O}(n)$ and weakly NP-hard even with $|\mathcal{B}| = 2$ (see [5] for NP-hardness of the perfect variant). Because of this, we examine the problems on different graph classes. We show that the directed robust b-matching problem is weakly NP-hard even on semi-directed paths, which are connected directed graphs whose underlying undirected graph is a path. The problem only becomes solvable in polynomial time on directed paths. In contrast, it turns out that the directed robust perfect b-matching problem can be solved in polynomial time on directed trees. A directed tree is a connected, directed graph whose underlying undirected graph is a tree. Since each semi-directed path is also a directed tree, this result shows that the perfect variant of the problem is easier to solve on some graph classes. Finally, we will discuss some ideas for a polynomial-time algorithm for the directed robust perfect b-matching problem on directed series-parallel graphs.

References

[1] Christina Büsing and Sabrina Schmitz. Robust two-stage combinatorial optimization problems under discrete demand uncertainties and consistent selection constraints. *Discrete Applied Mathematics*, 347:187–213, 2024.

[2] Omar El Housni, Vineet Goyal, Oussama Hanguir, and Clifford Stein. Matching drivers to riders: A two-stage robust approach. preprint on webpage at `https://arxiv.org/abs/2011.03624`, 2021.

[3] Irit Katriel, Claire Kenyon-Mathieu, and Eli Upfal. Commitment under uncertainty: Two-stage stochastic matching problems. *Theoretical Computer Science*, 408:213–223, 2008.

[4] Aranyak Mehta et al. Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science*, 8(4):265–368, 2013.

[5] Jenny Segschneider and Arie M.C.A. Koster. Robust two-dose vaccination schemes and the directed b-matching problem. preprint on webpage at `https://optimization-online.org/?p=22009`, 2023.

Mixed-integer linearity in nonlinear optimization: a trust region approach

Alberto De Marchi

Institute of Applied Mathematics and Scientific Computing,
Department of Aerospace Engineering,
University of the Bundeswehr Munich,
85577 Neubiberg, Germany
✉ alberto.demarchi@unibw.de

The operation of complex, integrated process systems demands efficient use of resources whilst imposing tight safety constraints. Mixed-integer optimization provides a powerful and flexible mathematical template for modeling many tasks that involve discrete and continuous variables. Our contribution [2] is dedicated to the minimization of a smooth nonlinear objective function subject to polyhedral and integrality constraints:

$$\text{minimize} \quad f(x) \qquad \text{over} \quad x \in \mathcal{X} := \{x \in \mathbb{R}^n \mid A_\mathrm{e} x = b_\mathrm{e},\ x_\mathrm{l} \leq x \leq x_\mathrm{u},\ x_i \in \mathbb{Z} \quad \forall i \in \mathcal{I}\}\,.$$

Bringing together nonlinear optimization with mixed-integer linear constraints enables versatile modeling, but poses significant computational challenges. Despite the vast literature on mixed-integer programming, this broad problem class remains relatively unexplored. With a focus on affordable methods, as opposed to a global optimization perspective, we address such mixed-integer optimization problems, without any convexity assumptions, following a trust region approach.

First of all, we define a suitable "criticality" concept, providing a localized tool to monitor first-order optimality. Then, we develop a numerical scheme that exploits the objective function's gradient and comes with convergence guarantees (and optional acceleration steps). In particular, we investigate a method based on sequential mixed-integer linear approximations with a trust region safeguard, computing feasible iterates via calls to a generic mixed-integer linear solver, possibly tailored to the structure and feasible set of the problem at hand. We build upon the trust region and sequential linearization rationales, providing a detailed analysis which allows to successfully address nonconvex objectives. Convergence to critical, possibly suboptimal, feasible points is established for arbitrary starting points. This contribution goes beyond classical methods which seem to rely on the feasible set or the objective function being convex, e.g., in sequential linear-quadratic programming [1] and Frank-Wolfe methods [3].

Finally, we present numerical applications in nonsmooth optimal control and optimal network design and operation. While the intended outreach of this work is mainly theoretical, our findings constitute a solid basis for the development of more efficient solvers, based for instance on the combination with Newton-type algorithms. Numerical results illustrate the algorithmic behavior on two nontrivial examples, with encouraging results, and demonstrate the potential benefit of heuristic acceleration steps.

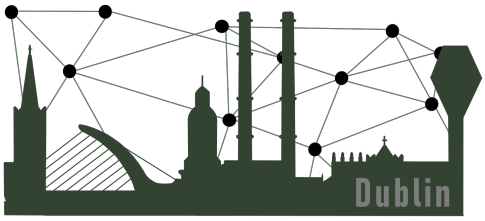
A research preprint with accompanying code is available online [2].

References

[1] Richard H. Byrd, Nicholas I. M. Gould, Jorge Nocedal, and Richard A. Waltz. On the convergence of successive linear-quadratic programming algorithms. *SIAM Journal on Optimization*, 16(2):471–489, 2005.

[2] Alberto De Marchi. Mixed-integer linearity in nonlinear optimization: a trust region approach, 2023. arXiv:2310.17285.

[3] Deborah Hendrych, Hannah Troppens, Mathieu Besançon, and Sebastian Pokutta. Convex mixed-integer optimization with Frank-Wolfe methods, 2023. arXiv:2208.11010v5.



INOC - 2024

Session Session 1C: P-Median

Monday 11 March 2024, 11:00-12:30

Q013

On the nested p-center problem

Christof Brandstetter
Institute of Business Analytics and Technology
Transformation/JKU Business School, Johannes Kepler
University Linz
Linz, Austria
christof.brandstetter@jku.at

Markus Sinnl
Institute of Business Analytics and Technology
Transformation/JKU Business School, Johannes Kepler
University Linz
Linz, Austria
markus.sinnl@jku.at

ABSTRACT

In recent years, the concept of nesting gained renewed interest within the location science community. Nesting allows to model temporal aspects in planning, which are highly relevant in practice. In this paper, we introduce the nested p-center problem, which is an extension of the classic p-center problem. In this problem we are given a finite time horizon and at each time period, we are allowed to open a given number of facilities. The sets of open facilities at each time period must fulfill the nesting property, i.e., the open facilities at an earlier time period must be a subset of the open facilities at a later time period. The objective function is the sum of the objective function values of the individual periods and the goal is to minimize this objective function. The objective function value of each period is the maximal distance between a customer and its closest open facility. We present two mixed integer programming formulations for this problem. We provide a computational study on well-known p-center instances from literature to assess the performance of the two formulations and also to analyse the effect of nesting.

1 INTRODUCTION

Consistency is very important in long term planning and in particular in the location of facilities. However, many facility location problems potentially provide inconsistent solutions for varying numbers of open facilities, i.e., for different numbers of allowed open facilities the optimal locations can be vastly different. In practice, this could result in opening and closing of facilities when facing a long term planning project, where initially some facilities are to be built, and at some later time steps additional facilities are to be built. This is of course undesirable for a variety of reasons such as monetary cost or environmental cost.

The first ideas of modeling a facility location problem with such consistency in mind appeared in the 1970s in works by Scott [20] and Roodman and Schwarz [19]. In these works, the authors describe certain *multi-period facility location problems*, where the number of open facilities is changed over time, but the open facilities cannot be relocated. We note that location problems needing to fulfill constraints of this type such as the particular *nesting property* which is considered in our work (see Definition 1 for details) can be categorized as one problem family within the area of the multi-period problems (see Chapter 11 of [15] for a general overview of this area). However, most of the existing recent work regarding such multi-period location problems usually focuses on varying demand, distances, or cost over time, see e.g., [4, 7]. In 2022, McGarvey and Thorsen [16] revisited nesting and applied the concept to the p-median problem. They also posed the question about applying the nesting property to other classical location problems, such as the maximum coverage problem or the p-center problem. In this paper, we follow up on this open question by considering the latter.

The *(discrete) nested p-center problem (n-pCP)* can be defined the following way:

Definition 1. We are given a set $\mathcal{I}$ of customer demand points, a set $\mathcal{J}$ of potential facility locations and distances $d_{ij} \geq 0$ between each $i \in \mathcal{I}$ and $j \in \mathcal{J}$. Additionally, we are given a time horizon $\mathcal{H} = \{1, \dots, H\}$ and a set of integers $\mathcal{P} = \{p^1, \dots p^H\}$ where $p^h \leq p^{h+1}$ for $h = 1, \dots, H-1$ and $p^H \leq |\mathcal{J}|$.

A feasible solution to the nested p-center problem consists of a set $\mathcal{J}^h \subseteq \mathcal{J}$ with $\left|\mathcal{J}^h\right| = p^h$ for $h \in \mathcal{H}$. Moreover, the *nesting property* must be fulfilled by these sets, i.e., $\mathcal{J}^h \subseteq \mathcal{J}^{h+1}$ must hold for $h = 1, \dots, H-1$.

For a given time period $h \in \mathcal{H}$ and set $\mathcal{J}^h$, let $d_h(\mathcal{J}^h) = \max_{i \in I} \min_{j \in \mathcal{J}^h} d_{ij}$. The objective function value of a given feasible solution is defined as $\sum_{h=1}^{H} d_h(\mathcal{J}^h)$ and the goal is to find a feasible solution with minimal objective function value.

Observation 1. The objective function of the n-pCP can be viewed as minimizing the sum of the *regrets* over the time periods, where the regret of a given n-pCP-solution for a time period h is defined as the difference between the objective function value of the n-pCP-solution for the time period and the optimal p-center value for $p = p^h$. We note that the minimization of regret is a popular concept when dealing with uncertainty, see, e.g., [21] and was also considered in [16].

Observation 2. For $|\mathcal{P}| = 1$ the problem reduces to the (classical) p-center problem (pCP) which was introduced by Hakimi [12] in 1964. The pCP is NP-hard for $p \geq 2$ [14].

Observation 3. In our definition of the n-pCP we have that the number of facilities to be allowed open is non-decreasing over the time horizon. The optimal solution to this problem is also the optimal solution to the problem variant, where the number of facilities to be allowed open is non-increasing over the time horizon (and the nesting property is accordingly adapted to $\mathcal{J}^{h+1} \subseteq \mathcal{J}^h$), as these problems are equivalent.

Figure 1 shows an instance of the (nested) p-center problem (the `eil51` instance of the `TSPlib`[18]). In this instance, we have that $\mathcal{I} = \mathcal{J}$, i.e., each point (visualized as gray dot) is a demand point

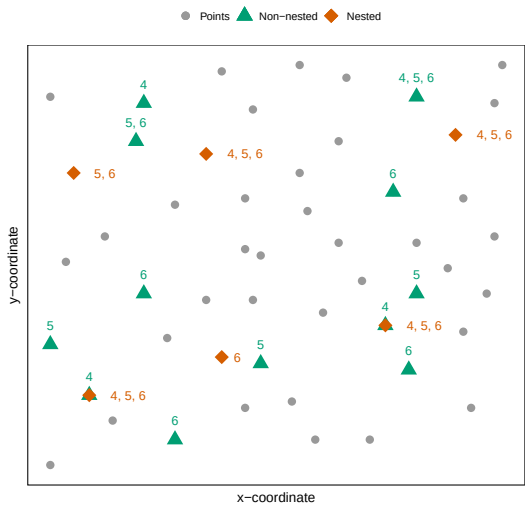


Figure 1: Instance: `eil51` and optimal solutions for n-pCP with $\mathcal{P} = \{4, 5, 6\}$ and pCP for $p = 4, 5, 6$

and can also be used as potential facility location. The distance between two points in this instance is the Euclidean distance. In this figure, next to illustrating an optimal solution of the n-pCP for $H = 3$ with $\mathcal{P} = \{4, 5, 6\}$, we also illustrate optimal solutions for the pCP when solving it for $p = 4, 5, 6$ individually. The nested solution is visualized using green triangles, with a green number above a triangle indicating that the facility is open at this location in the time period where p^h is this number. For example, if the numbers four, five, and six are besides the diamonds, it means that this location is used in all three time periods for the nested solution. The solutions for the pCP are visualized using the orange rectangles, with the orange numbers indicating that a facility is open at the location in the optimal solution where p is this number. Note that the optimal solutions for the pCP open twelve different facilities in total and only one facility which was in the solution for $p = 4$ was also in the solution for $p = 6$. The optimal objective function value for the n-pCP is 61, while the objective function values for the pCP for $p = 4, 5, 6$ are 22, 19 and 17, giving a value of 58 in total. The sum of the regrets is 61-58=3.

1.1 Contribution and outline

In this paper, we introduce the nested p-center problem and provide two mixed-integer linear programming (MILP) formulations for it. Based on these formulations, we conduct a computational study on well-known pCP instances from literature to assess the performance of the two approaches and also identify their limitations and potential areas for their improvement.

The paper is structured as follows: In the remainder of this section, we discuss previous and related work to the pCP and the nesting property. In Section 2 we present our two MILP formulations and Section 3 contains the computational study. Section 4 concludes the paper with an outlook to potential improvements to be considered in future work.

1.2 Literature review

For the pCP there exists considerable amount of work on heuristic and exact solution methods, as well as different adaptations and variants. We focus our literature review on existing exact methods, as our work is about the design of exact solution approaches. For the existing work on heuristic methods and approximation algorithms for the pCP we refer to the recent survey [11]. In 1970 the first exact solution approach for the pCP was developed [17] using the relationship to the set cover problem. Some recent state-of-the-art algorithms for the pCP also use this connection [5, 6]. The classical MILP formulation for the problem can be found in textbooks like in Chapter 5 of [8]. More recently, a compact formulation has been introduced in [9] and further extension of this formulation are presented in [1]. Their formulation has a binary variable y_j for $j \in \mathcal{J}$ to indicate at which point a facility opens and a binary variable u_k for each distinct distance D_k indicating whether the optimal value of pCP is less or equal than D_k. The authors show that their formulation has stronger linear programming (LP) relaxation bounds than the classical formulation. In [3] another compact formulation with the same strength of the LP-relaxation is presented.

In 2022 a new projection based formulation was introduced [10], which can be obtained by applying Benders Decomposition to the classical formulation [8]. Their formulation only uses binary variables y_j for $j \in \mathcal{J}$ indicating open facilities and a continuous variable which measures the distance in the objective function. The authors also present a lifting procedure for the inequalities in their formulation and show that the LP-relaxation bounds of their lifted formulation are the same as the bounds obtained by the formulations of [1, 3, 9].

Location problems, where facilities are opened iteratively over a certain time horizon, has been discussed firstly in the 70s by [20]. The authors compared a dynamic programming system, which takes into account the complete time horizon, with a myopic system, which optimizes the next period without considering any later periods. Their dynamic programming system outperformed the myopic system for larger time horizons. A first MILP formulation for such problems has been introduced in [19], where the authors try to minimize the operational cost of closing facilities iteratively over a certain time horizon and where the first to use a nesting-type constraint which enforces open facilities to be open until the end of the time horizon. Further, they presented a generalization of the formulation where they start with a set of facilities which are open at the beginning and a set of potential facilities which can be opened. The starting facilities can be closed at any point in the time horizon, but once closed must remain closed, while the potential facilities can be opened, but not closed again. This allows for a restricted redistribution of facilities. More recently, the nesting concept was revisited and applied it to the p-median problem for two objective functions, minimizing the sum of the regrets and minimizing maximum regret by [16].

Incremental facility location and network design problems, f.e. [2, 13] are multi-period problems, in which a network is incrementally extended by the means of adding arcs, facilities or nodes to maintain incrementally increasing coverage requirements in each period while optimizing some objective like minimization of total cost or maximizing the cumulative flow over the planning horizon.

2 MIXED INTEGER LINEAR PROGRAMMING FORMULATIONS

In this section, we begin by presenting a first formulation of the nested p-center problem based on the textbook pCP formulation (see, e.g., [8]). Afterward, we present a second formulation based on the pCP formulation presented in Section 2.1 of [1].

2.1 First formulation

Our first formulation (nPC1) for the n-pCP uses two sets of binary variables, denoted as x and y. The variable x_{ij}^h is indicating if customer demand point $i \in \mathcal{I}$ is assigned to potential facility location $j \in \mathcal{J}$ in time period h and the variable y_j^h is indicating if a facility is opened at the potential facility location j in time period h. The continuous variables R^h measure the maximum distance from any customer demand point to its nearest open facility in time period h.

$$\text{(nPC1)} \quad \min \quad \sum_{h \in \mathcal{H}} R^h \tag{1a}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}} y_j^h = p^h \qquad \forall h \in \mathcal{H} \tag{1b}$$

$$\sum_{j \in \mathcal{J}} x_{ij}^h = 1 \qquad \forall i \in \mathcal{I}, \forall h \in \mathcal{H} \tag{1c}$$

$$\sum_{j \in \mathcal{J}} d_{ij} x_{ij}^h \leq R^h \qquad \forall i \in \mathcal{I}, \forall h \in \mathcal{H} \tag{1d}$$

$$x_{ij}^h \leq y_j^h \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, h \in \mathcal{H} \tag{1e}$$

$$y_j^h \leq y_j^{h+1} \qquad \forall j \in \mathcal{J}, \forall h \in \mathcal{H} \setminus \{H\} \tag{1f}$$

$$x_{ij}^h \in \{0,1\} \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, h \in \mathcal{H} \tag{1g}$$

$$y_j^h \in \{0,1\} \qquad \forall j \in \mathcal{J}, h \in \mathcal{H} \tag{1h}$$

$$R^h \in \mathbb{R}_{\geq 0} \qquad \forall h \in \mathcal{H} \tag{1i}$$

The objective function (1a) minimizes the sum over the distances R^h over all time periods. The constraints (1b) ensure that p_h facilities are opened in time period h. Constraints (1c) ensure that each customer is only assigned to one facility in each time period. The constraints (1d) are pushing the decision variables R^h to the largest distance of any assigned customer-facility combination in each time period. Each customer can only be assigned to an open facility, which is ensured by constraints (1e). The nesting constraints (1f) ensure that each facility, which is opened in time period h, is also open in time period $h+1$, so once a facility is opened in a time period, it cannot be closed in later time periods. Without this constraint, the formulation would just represent the sum over the individual p-center problems for each time period. The remaining constraints (1g) - (1i) are the binary constraints for the variables x_{ij}^h and y_j^h and the non-negativity constraint for variables R^h.

2.2 Second formulation

Our second formulation uses the binary variable y_j^h for $j \in \mathcal{J}$ and $h \in \mathcal{H}$ to indicate the open facilities analogously to the formulation (nPC1). Furthermore, let $\mathcal{D} = \{d_{ij} : i \in \mathcal{I}, j \in \mathcal{J}\}$ denote the set of all possible distances and let $D_1 \leq \ldots \leq D_K$ be the values contained in $\mathcal{D}$, so $\mathcal{D} = \{D_1, \ldots, D_K\}$. Let $\mathcal{K}$ be the set of indices in $\mathcal{D}$.

For a $k \in \mathcal{K}$ the binary variables u_k^h indicate if the objective function value in time period h (measured by continuous variable R^h) is greater or equal than D_k. For customer $i \in \mathcal{I}$ let the set $\mathcal{S}_i$ be the set of indices $k \in \mathcal{K}$ for which there exists a facility $j \in \mathcal{J}$ with $d_{ij} = D_k$.

$$\text{(nPC2)} \quad \min \quad \sum_{h \in H} R^h \tag{2a}$$

$$\text{s.t.} \quad \sum_{j \in J} y_j^h = p^h \qquad \forall h \in \mathcal{H} \tag{2b}$$

$$D_0 + \sum_{k=1}^{K} (D_k - D_{k-1}) \; u_k^h \leq R^h \qquad \forall h \in \mathcal{H} \tag{2c}$$

$$u_k^h + \sum_{j : d_{ij} < D_k} y_j^h \geq 1 \qquad \forall i \in \mathcal{I}, \; \forall k \in \mathcal{S}_i \cup \{K\}, \forall h \in \mathcal{H} \tag{2d}$$

$$u_k^h \geq u_{k+1}^h \qquad \forall k \in \mathcal{K} \setminus \{K\}, \forall h \in \mathcal{H} \tag{2e}$$

$$y_j^h \leq y_j^{h-1} \qquad \forall j \in \mathcal{J}, \forall h \in \mathcal{H} \tag{2f}$$

$$y_j^h \in \{0,1\} \qquad \forall j \in \mathcal{J}, \forall h \in \mathcal{H} \tag{2g}$$

$$u_k^h \in \{0,1\} \qquad \forall k \in \mathcal{K}, \forall h \in \mathcal{H} \tag{2h}$$

$$R^h \in \mathbb{R} \qquad \forall h \in \mathcal{H} \tag{2i}$$

The objective function (2a) minimizes the sum over the distances R^h over all time periods. The correct value of the R^h-variables is ensured by constraints (2c). The constraints (2b) ensure that no more than p^h facilities are opened in each time period. Constraint (2d) is ensuring that if for any customer i in time period h no facility j with smaller distance than D_k is opened u_k^h has to be one. Since (2d) is not defined for all $k \in \mathcal{K}$ but only for the subsets based on $\mathcal{S}_i \cup \{K\}$, constraints (2e) are necessary in order to ensure that no u_k^h can equal zero if u_{k+1}^h is one (otherwise constraints (2c) would not measure the distance correctly). The inequalities (2f) are for the nesting and are the same as (1f). The remaining constraints are the binary and non-negativity constraints, respectively. This formulation has $O((|\mathcal{I}|+|\mathcal{K}|)\,|\mathcal{H}|)$ variables and $O(\min\{|\mathcal{I}|\,|\mathcal{J}|, |\mathcal{I}|\,|\mathcal{K}|\}\,|\mathcal{H}|)$ constraints instead of $O(|\mathcal{I}|\,|\mathcal{J}|\,|\mathcal{H}|)$ variables and constraints in (nPC1). Depending on $|\mathcal{K}|$ this can be a significant reduction in both variables and constraints.

Observation 4. There exist instances of the n-pCP where the LP relaxation bound of (nPC2) is stronger than the LP relaxation bound of (nPC1). This is a direct consequence of the fact that for instances with $|\mathcal{P}| = 1$ (i.e., the pCP) both formulations reduce to their classical pCP-formulation counterparts and that such a result is known for these pCP-formulations, see, e.g., [1, 9].

3 COMPUTATIONAL RESULTS

The formulations from Section 2 have been implemented in C++ with CPLEX 20.1 as MILP-solver and were run on a single core of an Intel Xeon X5570 machine with 2.93 GHz with all CPLEX settings left on default values. The time limit was set to 3600 seconds and the memory limit to 9 GB.

We used two well-known instance sets in our computational study:

- pmed: This set contains 40 instances and was used for the pCP in e.g., [1, 3, 5, 6, 10]. For all instances, $\mathcal{I} = \mathcal{J} = \mathcal{V}$ holds, so all customer demand points are also potential facility locations. The number of $|\mathcal{V}|$ ranges between 100 and 900, and p is between 5 and 200. The instances in this set are given as graphs, and the distances d_{ij} are the shortest-path distances between $i, j \in \mathcal{V}$ in the graph.
- TSPlib: This is an instance set originally introduced in [18] for the traveling salesperson problem. Subsets of this instance set have been used in many works for the pCP, see, e.g., [3, 5, 6, 9, 10]. The number of potential facilities equals again the number of customer demand points, so $|\mathcal{V}| = |\mathcal{I}| = |\mathcal{J}|$. In the subset we consider for our study $|\mathcal{V}|$ ranges between 51 and 1002. The exact values of $|\mathcal{V}|$ can be found in the name of the instances in Table 2 as part of the instance name. The instances contain the two-dimensional coordinates for each point to calculate the Euclidean distance. Following previous literature, we rounded the distances to the nearest integer.

3.1 Comparison of formulations

In Table 1 we present the runtime (column t[s]), lower and upper bounds (columns lb and ub) and optimality gap (column g[%]) obtained for the pmed instances with $\mathcal{P} = \{p, p+1, p+2\}$, where p is given by the instance. If no optimal solution has been found within the time limit, it is indicated with the abbreviation TL in the runtime-column. The optimality gap is calculated as $\frac{ub-lb}{ub}100$ and the best obtained gap considering both formulations is indicated in bold. The formulation (nPC2) manages to solve eleven instances to optimality within the time limit, whereas formulation (nPC1) does not manage to solve any instance. Moreover, there is no instance where the obtained optimality gap of (nPC1) at the time limit is better than the time limit of (nPC2). We note that for formulation (nPC1) in 19 instances it was not possible to finish building the model for CPLEX to finish solving the initial LP relaxation within the given time limit. These issues in scalability between the two formulations can be explained by the fact, that the distances in the pmed-instance are based on shortest-path distances in a graph, thus there are not so many distinct distances, hence K is small, which is good for (nPC2).

For the instance set TSPlib, we used $\mathcal{P} = \{4, 5, 6\}$. The results (reported in Table 2) paint a similar overall picture, i.e., (nPC2) outperforms (nPC1) most of the time. With formulation (nPC2) 14 out of 50 instances of this set can be solved to optimality within the timelimit, whereas for formulation (nPC1) only 3 out of 50 instances can be solved to optimality. Regarding the optimality gap for instances not solved to optimality, the picture is not as clear as for instance set pmed. However, for nearly all instance for which (nPC1) had a lower gap than (nPC2), the lower bound of (nPC2) is better, but the upper bound is worse. This indicates that CPLEX primal heuristics seem to work better with (nPC1) while the theoretical strength of the LP relaxation of (nPC2) compared to (nPC1) seems to pay off also in practice. We note that for these instances there are much more distinct distances, and thus larger values of K compared to instance set pmed, as in this instance set the distances are based on the Euclidean distance. This can contribute to make them more difficult for formulation (nPC2) and is also reflected in the results, as for some instances of this set, the formulation does not manage to solve the root relaxation within the time limit (indicated by a value of zero in the column lb in the table).

Table 1: Comparison of (nPC1) and (nPC2) for the pmed instances with $\mathcal{P} = \{p, p+1, p+2\}$

			(nPC1)				(nPC2)			
Inst.	p	|V|	t[s]	lb	ub	g[%]	t[s]	lb	ub	g[%]
1	100	5	TL	267	381	29.95	1833.939	356	356	**0.00**
2	100	10	TL	196	345	43.14	313.387	292	292	**0.00**
3	100	10	TL	207	310	33.27	111.594	278	278	**0.00**
4	100	20	TL	139	235	40.85	85.634	220	220	**0.00**
5	100	33	TL	85	139	38.64	61.483	138	138	**0.00**
6	200	5	TL	178	276	35.50	TL	220	252	**12.70**
7	200	10	TL	134	201	33.42	TL	180	188	**4.05**
8	200	20	TL	100	180	44.31	1028.47	161	161	**0.00**
9	200	40	TL	59	488	87.84	1022.54	109	109	**0.00**
10	200	67	TL	29	63	53.23	416.797	58	58	**0.00**
11	300	5	TL	132	297	55.51	TL	160	175	**8.79**
12	300	10	TL	111	303	63.25	TL	140	154	**9.03**
13	300	30	TL	69	349	80.34	2964.903	107	107	**0.00**
14	300	60	TL	42	406	89.73	TL	67	78	**13.49**
15	300	100	TL	24	204	88.24	2756.437	52	52	**0.00**
16	400	5	TL	108	223	51.51	TL	131	140	**6.50**
17	400	10	TL	87	216	59.55	TL	106	117	**9.68**
18	400	40	TL	0	103239	100.00	TL	74	423	**82.44**
19	400	80	TL	30	256	88.34	TL	49	303	**83.92**
20	400	133	TL	18	106	83.04	3488.017	39	39	**0.00**
21	500	5	TL	0	92343	100.00	TL	107	130	**17.39**
22	500	10	TL	0	125994	100.00	TL	101	339	**70.13**
23	500	50	TL	0	95016	100.00	TL	58	282	**79.31**
24	500	100	TL	0	101730	100.00	TL	39	300	**86.89**
25	500	167	TL	14	228	93.64	TL	32	245	**86.98**
26	600	5	TL	0	106278	100.00	TL	101	113	**10.88**
27	600	10	TL	0	120651	100.00	TL	87	96	**9.30**
28	600	60	TL	0	148722	100.00	TL	48	330	**85.57**
29	600	120	TL	0	109215	100.00	TL	32	264	**87.74**
30	600	200	TL	13	220	94.13	TL	24	288	**91.82**
31	700	5	TL	0	96330	100.00	TL	82	195	**58.05**
32	700	10	TL	0	210198	100.00	TL	77	372	**79.25**
33	700	70	TL	0	106851	100.00	TL	41	222	**81.60**
34	700	140	TL	0	149175	100.00	TL	28	294	**90.52**
35	800	5	TL	0	119076	100.00	TL	82	91	**9.50**
36	800	10	TL	0	161628	100.00	TL	75	261	**71.44**
37	800	80	TL	0	139044	100.00	TL	40	234	**83.09**
38	900	5	TL	0	160359	100.00	TL	78	252	**69.16**
39	900	10	TL	0	262383	100.00	TL	62	345	**81.89**
40	900	90	TL	0	127518	100.00	TL	34	207	**83.61**

3.2 Results in context to the pCP

Next, we take a closer look at the performance of (nPC2) and also put the results obtained for the (nPC2) in context with results obtained for the pCP. In Table 3 we report the objective function values at termination (columns ub), the lower bound at the root node (columns root lb) and at termination (columns lb) for the n-pCP with $\mathcal{P} = \{p, p+1, p+2\}$ and also for the pCP with p. The pCP is solved using the formulation of [1], i.e., (nPC2) for $\mathcal{P} = \{p\}$. We denote this formulation by (PC). Table 4 reports the same values for TSPlib.

In Table 3 and Table 4, we can see that the nesting does not seem to have a huge effect on the obtained root lower bounds, as for both (PC), and (nPC2) the root lower bounds are close to the lower bounds at termination and also close to the upper bounds for instances solved to optimality. Thus, also for the (nPC2) this

Table 2: Comparison of (nPC1) and (nPC2) for the TSPlib instances with $\mathcal{P} = \{4, 5, 6\}$ for (nPC2). The number in the instance names indicate the numbers of point in each instance.

	(nPC1)				(nPC2)			
Inst.	t[s]	lb	ub	g[%]	t[s]	lb	ub	g[%]
eil51	869.843	61	61	**0.00**	46.905	61	61	**0.00**
berlin52	2105.984	1215	1215	**0.00**	31.623	1215	1215	**0.00**
st70	2558.426	90	90	**0.00**	99.939	90	90	**0.00**
eil76	TL	50	70	29.20	409.317	64	64	**0.00**
pr76	TL	13060	16850	22.49	1416.761	16330	16330	**0.00**
rat99	TL	98	163	39.91	1399.774	144	144	**0.00**
kroA100	TL	1976	2973	33.54	2465.575	2812	2812	**0.00**
kroB100	TL	1967	3426	42.57	TL	2496	2965	**15.83**
kroC100	TL	1937	3199	39.45	TL	2402	2886	**16.76**
kroD100	TL	1842	3062	39.85	2863.816	2862	2862	**0.00**
kroE100	TL	1939	3097	37.38	2582.52	2893	2893	**0.00**
rd100	TL	669	1208	44.64	TL	911	993	**8.22**
eil101	TL	50	75	33.42	537.34	66	66	**0.00**
lin105	TL	1384	2352	41.14	2419.2	2067	2067	**0.00**
pr107	TL	3258	5238	37.80	1763.96	5170	5170	**0.00**
pr124	TL	5637	7682	26.62	2752.469	7370	7370	**0.00**
bier127	TL	12541	18279	31.39	2791.905	15936	15936	**0.00**
ch130	TL	487	776	37.25	TL	589	715	**17.63**
pr136	TL	7251	10674	32.07	TL	8591	9318	**7.80**
pr144	TL	6392	11866	46.13	TL	7774	12221	**36.39**
ch150	TL	441	873	**49.44**	TL	514	2547	79.81
kroA150	TL	1872	3523	**46.85**	TL	2198	12654	82.63
kroB150	TL	1881	3663	**48.66**	TL	2200	12561	82.49
pr152	TL	6831	15625	56.28	TL	8756	14493	**39.58**
u159	TL	3155	6593	52.14	TL	4073	5119	**20.44**
rat195	TL	132	280	**52.71**	TL	152	859	82.29
d198	TL	1013	4531	**77.65**	TL	1260	12780	90.14
kroA200	TL	1905	4092	**53.43**	TL	2215	12879	82.80
kroB200	TL	1862	3602	**48.32**	TL	2179	12510	82.58
ts225	TL	8548	41811	79.56	TL	9602	44826	**78.58**
tsp225	TL	227	543	**58.23**	TL	271	1557	82.58
pr226	TL	7733	14448	**46.47**	TL	9467	50796	81.36
gil262	TL	132	620	78.70	TL	152	703	**78.33**
pr264	TL	3104	24933	87.55	TL	3755	26069	**85.60**
a280	TL	140	726	**80.76**	TL	159	843	81.09
pr299	TL	2760	18220	**84.85**	TL	3136	20877	84.98
lin318	TL	2395	10980	**78.18**	TL	2774	14598	80.99
linhp318	TL	2395	10980	**78.18**	TL	2774	14598	80.99
rd400	TL	0	898956	100.00	TL	767	4059	**81.10**
fl417	TL	0	0	100.00	TL	1381	7029	**80.36**
pr439	TL	0	0	100.00	TL	7780	38460	**79.77**
pcb442	TL	0	0	100.00	TL	2530	14523	**82.58**
d493	TL	0	0	100.00	TL	1900	12888	**85.26**
u574	TL	0	0	**100.00**	TL	0	10314	**100.00**
rat575	TL	0	493497	**100.00**	TL	0	493497	**100.00**
p654	TL	0	0	**100.00**	TL	0	0	**100.00**
d657	TL	0	0	**100.00**	TL	0	0	**100.00**
u724	TL	0	0	**100.00**	TL	0	0	**100.00**
rat783	TL	0	782220	**100.00**	TL	0	782220	**100.00**
pr1002	TL	0	0	**100.00**	2014.429	0	0	**100.00**

modeling approach seems to give strong lower bounds. However, the heuristics of CPLEX seem to struggle to find good feasible solutions for larger instances of the n-pCP. This can be inferred from the fact that we can construct a valid upper bound solution for the n-pCP-instances by just putting the solution obtained for the pCP also as solution for $p+1$ and $p+2$ (together with one, resp., two random additional open facilities). The objective function value of solutions constructed in such a way for the n-pCP is at most three times the objective function value obtained the pCP. Thus, for example, for the instance *rat195* we would obtain a solution for n-pCP with value at most 216 while using (nPC2) we obtained

Table 3: Root bound comparison on instance set pmed with $\mathcal{P} = \{p, p+1, p+2\}$. Optimal objective function values are printed in **bold**.

			(PC)			(nPC2)		
Inst.	|V|	p	ub	root lb	lb	ub	root lb	lb
1	100	5	**127**	118	127	**356**	306	356
2	100	10	**98**	98	98	**292**	243	292
3	100	10	**93**	93	93	**278**	237	278
4	100	20	**74**	74	74	**220**	188	220
5	100	33	**48**	43	48	**138**	104	138
6	200	5	**84**	79	84	252	214	220
7	200	10	**64**	61	64	188	167	180
8	200	20	**55**	51	55	**161**	140	161
9	200	40	**37**	36	37	**109**	92	109
10	200	67	**20**	25	20	**58**	47	58
11	300	5	**59**	59	59	175	154	160
12	300	10	**51**	52	51	154	139	140
13	300	30	**36**	32	36	**107**	95	107
14	300	60	**26**	31	26	78	65	67
15	300	100	**18**	14	18	**52**	41	52
16	400	5	**47**	46	47	140	128	131
17	400	10	**39**	36	39	117	105	106
18	400	40	**28**	25	28	423	74	74
19	400	80	**18**	15	18	303	46	49
20	400	133	**13**	11	13	**39**	31	39
21	500	5	**40**	38	40	130	107	107
22	500	10	**38**	35	38	339	101	101
23	500	50	**22**	20	22	282	58	58
24	500	100	**15**	13	15	300	38	39
25	500	167	**11**	9	11	245	26	32
26	600	5	**38**	36	38	113	100	101
27	600	10	**32**	30	32	96	87	87
28	600	60	**18**	16	18	330	48	48
29	600	120	**13**	11	13	264	32	32
30	600	200	**9**	7	9	288	22	24
31	700	5	**30**	31	30	195	82	82
32	700	10	**29**	27	29	372	77	77
33	700	70	**15**	14	15	222	41	41
34	700	140	**11**	9	11	294	28	28
35	800	5	**30**	29	30	91	82	82
36	800	10	28	25	27	261	74	75
37	800	80	59	13	15	234	40	40
38	900	5	**29**	29	29	252	77	78
39	900	10	**23**	21	23	345	62	62
40	900	90	**13**	11	13	207	34	34

an upper bound of 859 at termination. The situation is similar for many other instances.

4 CONCLUSION AND OUTLOOK

In this work, we introduced the nested p-center problem and presented two mixed-integer linear programming formulations for it, together with a computational study to evaluate the effectiveness

Table 4: Root bound comparison on instance set TSPlib with $\mathcal{P} = \{4, 5, 6\}$. TSPlib. Optimal objective function values are printed in **bold**.

	(PC)			(nPC2)		
Inst.	ub	root lb	lb	ub	root lb	lb
eil51	**22**	22	22	**61**	52	61
berlin52	**426**	426	426	**1215**	1100	1215
st70	**33**	33	33	**90**	78	90
eil76	**23**	23	23	**64**	52	64
pr76	**6082**	6082	6082	16330	12862	16329
rat99	**51**	46	51	**144**	109	144
kroA100	**1001**	832	1001	**2812**	2198	2812
kroB100	**989**	857	989	2965	2241	2496
kroC100	**977**	839	977	2886	2197	2402
kroD100	**995**	840	995	**2862**	2188	2862
kroE100	**1030**	826	1030	**2893**	2164	2893
rd100	**349**	305	349	993	794	911
eil101	**23**	92	23	**66**	53	66
lin105	**717**	615	717	**2067**	1619	2067
pr107	**1746**	1746	1746	**5170**	3730	5170
pr124	**2588**	2497	2588	**7370**	6592	7370
bier127	**5578**	5051	5578	**15936**	13004	15936
ch130	**237**	228	237	715	544	589
pr136	**3225**	2880	3225	9318	7742	8591
pr144	**3375**	2961	3375	12221	7758	7774
ch150	**225**	196	225	2547	512	514
kroA150	**1024**	822	1024	12654	2192	2198
kroB150	**1042**	830	1042	12561	2190	2200
pr152	**5100**	3732	5100	14493	8730	8756
u159	**1655**	1461	1655	5119	3754	4073
rat195	**72**	57	72	859	152	152
d198	**623**	541	623	12780	1255	1260
kroA200	1011	834	975	12879	2214	2215
kroB200	1008	826	835	12510	2178	2179
ts225	**4243**	3751	4243	44826	9601	9602
tsp225	**124**	104	124	1557	271	271
pr226	**4104**	3704	4104	50796	9438	9467
gil262	**66**	58	66	703	152	152
pr264	**1610**	1537	1610	26069	3754	3755
a280	**79**	60	79	843	159	159
pr299	6959	1192	1194	20877	3136	3136
lin318	4866	1065	1065	14598	2774	2774
linhp318	1331	1065	1065	14598	2774	2774
rd400	441	296	296	4059	767	767
fl417	676	536	537	7029	1381	1381
pr439	12820	12820	2958	38460	7780	7780
pcb442	4280	980	980	14523	2530	2530
d493	3957	749	749	12888	1900	1900
u574	3438	3438	692	10314	0	0
rat575	534	100	113	493497	0	0
p654	6083	1438	1444	7594660	0	0
d657	4771	4771	878	6044890	0	0
u724	3198	3198	614	3927170	0	0
rat783	628	628	117	782220	0	0
pr1002	0	0	0	0	0	0

of the proposed formulations. Based on the findings of the computational study, we currently work on the following topics to obtain an improved solution framework for the problem:

- design of starting and primal heuristics, as CPLEX seems to struggle to find good primal solutions on its own
- transferring the projection-based p-center formulation and constraint lifting ideas of [10] to the nested setting for better overall scaleability

Another interesting avenue for further research could be to combine the nested problem with some form of uncertainty.

ACKNOWLEDGMENTS

This research was funded in whole, or in part, by the Austrian Science Fund (FWF) [P 35160-N]. For the purpose of open access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

[1] Zacharie Ales and Sourour Elloumi. 2018. Compact MILP Formulations for the p-Center Problem. *Combinatorial Optimization* (2018), 14–25.
[2] Ashwin Arulselvan, Andreas Bley, and Ivana Ljubić. 2019. The incremental connected facility location problem. *Computers & Operations Research* 112 (2019), 104763.
[3] Hatice Calik and Barbaros C Tansel. 2013. Double bound method for solving the p-center location problem. *Computers & Operations Research* 40, 12 (2013), 2991–2999.
[4] Tobia Calogiuri, Gianpaolo Ghiani, Emanuela Guerriero, and Emanuele Manni. 2021. The multi-period p-center problem with time-dependent travel times. *Computers & Operations Research* 136 (2021), 105487.
[5] Doron Chen and Reuven Chen. 2009. New relaxation-based algorithms for the optimal solution of the continuous and discrete p-center problems. *Computers & Operations Research* 36, 5 (2009), 1646–1655.
[6] Claudio Contardo, Manuel Iori, and Raphael Kramer. 2019. A scalable exact algorithm for the vertex p-center problem. *Computers & Operations Research* 103 (2019), 211–220.
[7] Isabel Correia and Teresa Melo. 2016. Multi-period capacitated facility location under delayed demand satisfaction. *European Journal of Operational Research* 255, 3 (2016), 729–746.
[8] Mark S. Daskin. 2013. *Network and Discrete Location: Models, Algorithms, and Applications, Second Edition.* John Wiley & Sons, Ltd.
[9] Sourour Elloumi, Martine Labbé, and Yves Pochet. 2004. A new formulation and resolution method for the p-center problem. *INFORMS Journal on Computing* 16, 1 (2004), 84–94.
[10] Elisabeth Gaar and Markus Sinnl. 2022. A scaleable projection-based branch-and-cut algorithm for the p-center problem. *European Journal of Operational Research* 303, 1 (2022), 78–98.
[11] Jesus Garcia-Diaz, Rolando Menchaca-Mendez, Ricardo Menchaca-Mendez, Saúl Pomares Hernández, Julio César Pérez-Sansalvador, and Noureddine Lakouari. 2019. Approximation algorithms for the vertex k-center problem: Survey and experimental evaluation. *IEEE Access* 7 (2019), 109228–109245.
[12] S Louis Hakimi. 1964. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research* 12, 3 (1964), 450–459.
[13] Thomas Kalinowski, Dmytro Matsypura, and Martin W.P. Savelsbergh. 2015. Incremental network design with maximum flows. *European Journal of Operational Research* 242, 1 (2015), 51–62.
[14] Oded Kariv and S Louis Hakimi. 1979. An algorithmic approach to network location problems. I: The p-centers. *SIAM journal on applied mathematics* 37, 3 (1979), 513–538.
[15] Gilbert Laporte, Stefan Nickel, and Francisco Saldanha-da Gama. 2019. *Introduction to location science.* Springer.
[16] Ronald G McGarvey and Andreas Thorsen. 2022. Nested-solution facility location models. *Optimization letters* 16, 2 (2022), 497–514.
[17] Edward Minieka. 1970. The m-center problem. *Siam Review* 12, 1 (1970), 138–139.
[18] Gerhard Reinelt. 1991. TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing* 3, 4 (1991), 376–384.
[19] Gary M Roodman and Leroy B Schwarz. 1975. Optimal and heuristic facility phase-out strategies. *AIIE transactions* 7, 2 (1975), 177–184.
[20] Allen J Scott. 1971. Dynamic location-allocation systems: some basic planning strategies. *Environment and planning A* 3, 1 (1971), 73–82.
[21] Lawrence V. Snyder. 2006. Facility location under uncertainty: a review. *IIE Transactions* 38, 7 (2006), 547–564.

Revisiting a Cornuéjols-Nemhauser-Wolsey formulation for the p-median problem

Cristina Requejo[1] and Agostinho Agra[2]

[1]ISEG, University of Lisbon, Lisbon, Portugal, ✉ crequejo@iseg.ulisboa.pt
[2]Department of Mathematics, University of Aveiro, Aveiro, Portugal, ✉ aagra@ua.pt

In the p-median problem we are given a set of locations $V = \{1, \ldots, n\}$ and non-negative costs $w_{ij}, i \in V, j \in V$, representing the cost of serving location i from a facility installed in location j (we assume $w_{ii} = 0, i \in V$). The p-median problem consists in selecting a subset of p locations where to establish facilities and allocate each location to these facilities in order to minimize the total servicing cost.

Here we revisit a formulation for the simple facility location and p-median problems introduced by Cornuéjols, Nemhauser and Wolsey (1980) [2]. This formulation uses two sets of variables: binary variable $y_j, j \in V$ that indicate whether location j is selected or not to install a facility, and nonnegative continuous variables c_j for the cost of serving location j, that is,

$$c_j = \min_{i \in V} \{w_{ji} | y_i = 1\}, \qquad j \in V. \tag{1}$$

The p-median problem can be modeled as follows:

$$\min \quad \sum_{j \in V} c_j \tag{2}$$

$$\text{s. t.} \quad \sum_{j \in V} y_j = p, \tag{3}$$

$$c_j \geq w_j^{j_k} - \sum_{t=1}^{k-1} (w_j^{j_k} - w_j^{j_t}) y_{j_t}, \qquad j \in V, k \in \{1, \ldots, n-p+1\}, \tag{4}$$

$$c_j \geq 0, \qquad j \in V, \tag{5}$$

$$y_j \in \{0, 1\}, \qquad j \in V. \tag{6}$$

This formulation, named CNW, can be seen as the intersection of a selection set ((3), (6)) with an additional family of optimality constraints (4) to define the costs correctly. We establish connections to the classical formulation and to the Radius formulation [1].

Despite being the smallest known formulation regarding the number of variables, this formulation is barely used in the literature. Recently, it has been employed as the result of Benders decomposition of other formulations for large scale p-median problems, see [3].

By exploring the optimality constraints we discuss approaches to derive bounds for large-size instances. These approaches are based on relaxations obtained by eliminating optimality constraints and can be seen as simple matheuristic to solve large size instances. In particular, we characterize relaxations which provide the optimal solution, and therefore, can be seen as new formulations for the p-medium problem. One such relaxation is obtained by dropping half of the optimality constraints (4). For those cases where the relaxation is not exact we propose a row generation algorithm to solve the instances to optimality.

Computational tests are reported [1] showing that the renewed formulation can be used efficiently to solve p-medium instances. Indeed, by considering a small number of constraints (we report results with two selection approaches and ten constraints associated with each node) provide very good solutions and bounds. Moreover, separation algorithms based on these formulations are very efficient. The formulation and the separation schemes can be easily implemented and used in large size instances. The results also show that this formulation is a good alternative to the Radius formulation when the values of p are not too large. Overall, we showed that this formulation can be quite competitive to solve certain location

problems. Nevertheless, for each particular related problem,it may be necessary to adapt the relaxations resulting from discarding optimality constraints and it may also be necessary to adjust the separation schemes. Moreover, in order to devise competitive exact algorithms, the separation scheme needs to be embedded in a general framework where additional issues (e.g. branching rules, use of heuristics, etc.) need to be taken into consideration.

References

[1] A. Agra, C. Requejo. Revisiting a Cornuéjols-Nemhauser-Wolsey formulation for the p-median problem. *EURO Journal on Computing*, 2023.

[2] G. Cornuéjols, G.L. Nemhauser, L.A. Wolsey. A canonical representation of simple plant location problems and its applications. *SIAM Journal on Algebraic and Discrete Methods*, 1(3), 261–272, 1980.

[3] C. Duran-Mateluna, Z. Ales, S. Elloumi, An efficient Benders decomposition for the p-median problem, *European Journal of Operational Research*, 308, 84–96, 2023.

Extensions of Node-Depot Assignment Formulations for the Hamiltonian p-Median Problem

Luís Gouveia[1] and Francisco Canas[1]

[1]CMAFCIO, Faculdade de Ciências da Universidade de Lisboa, Lisbon, Portugal, ✉ legouveia@fc.ul.pt

Given a weighted complete undirected graph $G = (V, E)$ with weights on the edges and a positive integer p, the symmetric Hamiltonian p-Median Problem (HpMP) on G is to find a minimum weight set of p elementary cycles partitioning the vertices of G. We focus on a variant of the HpMP (the HpMP$_{\geq}$) where solutions with strictly more than p cycles are also feasible.

We present new extensions of known node-depot assignment compact formulations. Node-depot assignment models are characterized by the inclusion of node-depot assignment variables h_i^d, which are 1 if and only if node d acts as the depot of the cycle node i is in (or alternatively, if node i is "assigned" to node d), in addition to edge variables u_{ij}, which are 1 if and only if edge $\{i, j\}$ is in some cycle. The standard known node-depot assignment formulation is as follows (see [1], [2], [3]):

$$\text{Min.} \quad \sum_{\{i,j\}\in E} d_{ij}u_{ij} \tag{1a}$$

$$\text{s.t.} \quad \sum_{j\in V} u_{ij} = 2, \qquad \forall i \in V \tag{1b}$$

$$\sum_{d\in V} h_d^d = p, \tag{1c}$$

$$\sum_{d\in V} h_i^d = 1, \qquad \forall i \in V \tag{1d}$$

$$h_i^d \leq h_d^d, \qquad \forall i, d \in V \tag{1e}$$

$$h_i^d + u_{ij} \leq h_j^d + 1, \quad \forall i, j, d \in V : i \neq j \tag{1f}$$

$$h_i^d = 0, \qquad \forall i, d \in V : d > i \tag{1g}$$

$$u_{ij} \in \{0, 1\}, \qquad \forall \{i, j\} \in E \tag{1h}$$

$$h_i^d \in \{0, 1\}, \qquad \forall i, d \in V \tag{1i}$$

The objective function corresponds to minimizing the sum of the weights of the edges which define the p cycles. Equalities (1b) are the usual assignment constraints stating that each node is included in exactly one cycle. Constraints (1h) define the u_{ij} variables as binary. A solution to the formulation described by the u_{ij} variables alone is composed of one or more disjoint elementary cycles (with at least three nodes each) covering all nodes of the graph. Equality (1c) indicates the number of nodes that play the role of depots of the cycles. Constraints (1d) state that any node i must be assigned to exactly one node d, and (1e) state that if some node i is assigned to some node d, then, that node must act as a depot. Constraints (1g) enforce a symmetry breaking strategy (by stating that a node i cannot be assigned to any node d such that $d > i$), and finally, (1i) define these variables as binary (although they can be defined as continuous without altering the validity of any models presented here).

Observe that solutions representing sets of less than p cycles are still feasible for the model including only the constraints described so far. That is why constraints (1f) are also required: these constraints state that if two nodes i and j are adjacent, they must be assigned to the same depot, and by preventing the existence of multiple depots in a single cycle, prevent solutions with less than p cycles.

We extend these formulations with edge-depot assignment variables z_{ij}^d, which are 1 if and only if node d acts as the depot of the cycle edge $\{i, j\}$ is in. In this talk, we:

i) propose new formulations which use these variables;

ii) relate these models with formulations including only the u_{ij} and h_i^d variables;

iii) present new inequalities in the space of the u_{ij} and h_i^d variables that are derived from the new models using edge-based variables;

iv) present computational results obtained with the new models.

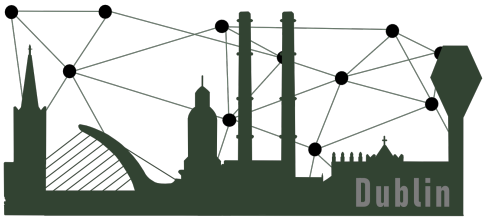
Preliminary computational results show that the new models produce very strong LP relaxation bounds and, when used in combination with a modern ILP solver, allow for the resolution of instances with over one-hundred nodes.

References

[1] Michele Barbato, Francisco Canas, Luís Gouveia, and Pierre Pesneau. Node based compact formulations for the hamiltonian p-median problem. *Networks*, 82(4):336–370, 2023.

[2] Güneş Erdoğan, Gilbert Laporte, and Antonio M Rodríguez Chía. Exact and heuristic algorithms for the hamiltonian p-median problem. *European Journal of Operational Research*, 253(2):280–289, 2016.

[3] Stefan Gollowitzer, Luis Gouveia, Gilbert Laporte, Dilson Lucas Pereira, and Adam Wojciechowski. A comparison of several models for the hamiltonian p-median problem. *Networks*, 63(4):350–363, 2014.



INOC - 2024

Session Session 2A: Smart Grids
Monday 11 March 2024, 15:30-17:00
Q011

Unit Commitment problem with uncertain demand and renewable energy availability

Cristian Aguayo[1,3] and Bernard Fortz[2,3]

[1]Département d'Informatique, Université libre de Bruxelles, Brussels, Belgium
[2]HEC Liège, Management School of the University of Liège, Liège, Belgium
[3]INOCS, INRIA, Villeneuve d'Ascq, France

Abstract: Energy generation is an area where four of the many problems facing humanity today - the energy crisis, climate change, scarcity of natural resources and global warming - interact in a vicious circle. In this scenario, it is not surprising that governments around the world are reviewing their energy policies. Agents such as Renewable Energy Sources (RES) and Local Energy Communities (LEC) can play a fundamental role in the energy transition. However, the optimization of operational costs continues to be a relevant factor. The Unit Commitment (UC) problem is one of the classical approaches to optimize these costs. This problem involves decisions related to the schedule of generating units as well as the power they must produce in order to meet the total power demand, where the last one can be deterministic or uncertain. By integrating RES, a new source of uncertainty is added to the problem. In this work, various formulations for the UC problem with uncertainty are presented and solved using benchmark data.

Keywords: Robust optimization, mixed integer linear programming, energy

1 Introduction

The Unit Commitment (UC) problem is one of the classical approaches to determine which generators should be switched on or off in a given period and how much power they would dispatch over a short or medium-term time horizon, subject to sets of constraints such as demand meeting, minimum up and down times, and minimum and maximum power output. Usually, the UC problem is used to optimize the power generation of thermal and hydrothermal units [7, 5, 3], but there are also applications of this model for cases with renewable energy sources, either with or without the presence of hydrothermal generating units [4, 6]. The UC problem is often formulated as a mixed integer program (MIP), which can be linear (MILP) or non-linear (MINLP). The UC problem involves decisions about turning on or off generators belonging to a set $\mathcal{J} = \{1, 2, ..., J\}$ during a certain period, and how much power they will produce. On and off decisions are represented by binary decision variables, while power generation in represented by continuous decision variables. Usually, the time horizon is represented by a discrete set $\mathcal{K} = \{0, 1, ..., K\}$ of periods. In a very general way, some of the constraints of the UC problem are:

- Power demand meeting for each period: each period, the power produced by all the generating units that are on should be able to satisfy the power demand.
- Minimum up and down constraints: depending on the nature of the generating units, they may have a minimum operating time after being turned on, as well as a minimum idle time after being turned off.
- Ramping constraints: these constraints model the fact that a generating unit cannot drastically increase or decrease the amount of generated power.

Since the UC problem must ensure that energy demand is met, energy demand forecasts play a fundamental role in solving real-life economic dispatch problems. When considering Renewable Energy Sources (RES), forecasts related to the availability of these intermittent energy sources must also be considered, so it is necessary to have tools to efficiently handle the randomness in the UC problem. This paper focuses on presenting and solving different UC models with uncertainty in energy demand and availability of dispatchable renewable energy.

2 Methodology

Two groups of UC models can be found in literature. The first one is the Single Bus UC (SB-UC), in which the aggregate energy demand must be satisfied, and the second is the Multi Bus UC (MB-UC), where the demand of each bus must be satisfied separately. The main difference between these groups of models is that MB-UC considers that energy flows through transmission lines, which have a limited capacity of energy flow,which implies that the UC problem acquires a network structure by adding the decision of how much energy flows from one bus to another.

To deal with uncertainty, the robust optimization approach was adopted. Robust optimization considers a set of uncertainty over which the objective function is optimized, taking into account that feasibility must be maintained for all possible values of the uncertain parameters. To avoid that the solutions obtained are too conservative, an uncertainty budget is usually considered, which allows adjusting the level of conservatism of the solution [2]. For the UC problem with uncertain demand and RES availability studied in this paper, each source of uncertainty has its own uncertainty budget parameter.

Two techniques were implemented to solve de robust model: Affine Decision Rules (ADR) [1], and Column and Constraint Generation (CCG) [8]. The proposed models and the proposed techniques were tested on instances based on the IEEE multi bus instances available in the MATPOWER package repository for `MATLAB`.

References

[1] Aharon Ben-Tal et al. “Adjustable robust solutions of uncertain linear programs”. In: *Mathematical programming* 99.2 (2004), pp. 351–376.

[2] Dimitris Bertsimas and Melvyn Sim. “The price of robustness”. In: *Operations research* 52.1 (2004), pp. 35–53.

[3] M. Carrion and J.M. Arroyo. “A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem”. In: *IEEE Transactions on Power Systems* 21.3 (2006), pp. 1371–1378. DOI: 10.1109/TPWRS.2006.876672.

[4] Shantanu Chakraborty et al. “Optimal Thermal Unit Commitment Integrated with Renewable Energy Sources Using Advanced Particle Swarm Optimization”. In: *IEEJ Transactions on Electrical and Electronic Engineering* 4.5 (2009), pp. 609–617. DOI: https://doi.org/10.1002/tee.20453. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/tee.20453. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/tee.20453.

[5] H. Habibollahzadeh and J. A. Bubenko. “Application of Decomposition Techniques to Short-Term Operation Planning of Hydrothermal Power System”. In: *IEEE Transactions on Power Systems* 1.1 (1986), pp. 41–47. DOI: 10.1109/TPWRS.1986.4334842.

[6] Germán Morales-España, Álvaro Lorca, and Mathijs M de Weerdt. “Robust unit commitment with dispatchable wind power”. In: *Electric Power Systems Research* 155 (2018), pp. 58–66.

[7] M.V.F. Pereira and L.M.V.G. Pinto. “Application of Decomposition Techniques to the Mid - and Short - Term Scheduling of Hydrothermal Systems”. In: *IEEE Transactions on Power Apparatus and Systems* PAS-102.11 (1983), pp. 3611–3618. DOI: 10.1109/TPAS.1983.317709.

[8] Bo Zeng and Long Zhao. “Solving two-stage robust optimization problems using a column-and-constraint generation method”. In: *Operations Research Letters* 41.5 (2013), pp. 457–461.

A Model for Local Energy Community Management in the Presence of Distribution Network Time-of-use Tariffs

James Fitzpatrick[1], Juan Sepúlveda[2], Hélène Le Cadre[2], Luce Brotcorne[2], Victor Astapov[3], Paula Carroll[1], and Anna Mutule[3]

[1]School of Business, Univeristy College Dublin, Dublin, Ireland, ✉ james.fitzpatrick1@ucd.ie
[2]Inria, Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, Lille, France,
[3]Institute of Physical Energetics, Riga, Latvia,

Local Energy Communities (LECs) are seen as a mechanism for citizens to actively engage in the energy transition. The integration of LECs into the existing distribution networks is key to a future of clean, secure, and equitable energy. Current research, including studies like [3, 4], delves into critical questions about the construction, management, and operation of low-voltage electricity networks to facilitate LEC participation. There is an increasing need for detailed models that consider the diverse, and sometimes conflicting interests of various energy system stakeholders. These models should account for factors like energy loss, stochastic generation and demand, and infrastructure specific to LECs and the types of renewable generation and energy storage assets they may choose to invest in.

In this changing landscape, it is key to consider the point of view and interests of both the distribution network operator (DSO) and LECs and managers who may act on their behalf. The challenge for the DSO is not just to ensure the reliability of the system but also to enhance operational efficiency metrics, a task that increases in difficulty in the presence of LECs. On the other hand, LECs management systems should ensure the maximisation of the LEC's own performance metrics, while taking into account the selfish interests of their members some of whom are prosumers i.e., customers who both produce and consume electricity. Members of the LEC have their own heterorgeneous preferences and aim to maximise their own utility functions. Ultimately, the effect of the LEC in the operation of the distribution system materialises in the resulting power flows in the lines and transformers of the grid system that hosts the LEC.

Different LEC business models and grid topology design decisions may affect the resulting power flows, allowing different points of view to tackle the problem of supporting the development and operation of LECs. From the planning perspective, studies in the context of smart grids, such as [2], have explored approaches such as network reinforcement or reconfiguration to achieve better distribution of flows. The authors use a MILP approach to minimise the costs of additional network cables and to minimise flows between the LEC members, hence promoting self sufficiency and controlling the LEC's electricity flows. This approach allows the technical requirements from a power systems network operation perspective to be explicitly included in the model. Emerging smart grid technologies manage flows at the physical and/or logical level. Another perspective involves the design of business models of LECs, eventually determining their energy consumption and production patterns, as in [3]. In response to the decision-making at the LEC management level, demand response models could be implemented to align the interests of members of an LECs and the DSO.

The interaction problem between the distribution network managed by the DSO and the LEC involving prosumers can be modeled as a noncooperative game [5] with coupling constraints capturing the bilateral reciprocity of the energy trades among the prosumers. Stackelberg games, and reformulations as bilevel optimization problems, have been used to capture the hierarchical decision process between DSO, LEC, and prosumers. For instance, the versatility of multi-level models is demonstrated in [1].

In this work, we propose a bilevel model that hierarchically incorporates, at the lower level, the LEC management system, and, at the upper level, the DSO. Resulting equilibria determine pricing strategies for the DSO, in the form of tariffs, which can be interpreted as signals for the active management of the LEC and the operation of the distribution network, and ultimately, determine the resulting network power flows. In this way, we aim to capture the complex interplay between LEC and DSO, and the real-time distribution network operation.

A branch-and-cut approach is employed to solve efficiently this complex, multi-level problem. This method allows us to systematically explore the solution space, cutting off non-promising branches to focus computational efforts on more likely solutions. By incorporating elements such as demand response, energy storage, and variable renewable energy sources into our model, we aim to reflect the actual conditions and challenges of modern electricity networks.

In summary, our bilevel model presents a strategic approach to integrating LECs into distribution networks, balancing the objectives of the participants of the LECs with the efficient operation of the distribution network. By applying a game-theoretic approach, this model adeptly manages the complex interactions between the DSO and LECs, enhancing its practicality by taking into account the strategic behaviors of both DSOs and LECs. This makes our model a significant contribution towards developing sustainable, efficient energy policies for the future. However, bridging the gap between the proposed model and its actual implementation presents significant challenges, indicating numerous opportunities for future research in this area. Among them, we highlight the need for policies to periodically estimate the parameters of the model based on online measurements, particularly under highly uncertain meteorological conditions (wind, temperature, solar irradiance, cloud cover etc.). That direction could be tackled on the algorithmic side by taking advantage of the structure of the underlying network abstracted as a graph, to provide algorithmic approach for equilibrium computation. Decentralised (machine learning) based approaches could be proposed to compute equilibrium, allowing both the speeding up of the convergence rate and protecting data privacy.

Finally, taking a broader perspective, distribution grid tariffs are a key element in coordinating the operation and network investment decisions, and conventional constant volumetric grid usage tariffs do not provide an adequate response to the challenge of increasing peak loads, caused by the electrification of heating and transport, which may require significant grid reinforcements. This change of paradigm requires us to investigate the impact of different tariff structures on the operation and investment decisions, as well as their indirect impact on the LEC. Interesting interpretations and tariff structure policy design might result from the long-term extension of our bilevel model.

Acknowledgements: This work emanates from research supported by the ERA-NET Cofund grant under the CHIST-ERA IV Joint Call on Novel Computational Approaches for Environmental Sustainability (CES), project "Supporting Energy Communities - Operational Research and Energy Analytics" (SEC-OREA). James Fitzpatrick is funded by the Irish Research Council.

References

[1] Didier Aussel, Luce Brotcorne, Sébastien Lepaul, and Léonard von Niederhäusern. A trilevel model for best response in energy demand-side management. *European Journal of Operational Research*, 281(2):299–315, 2020.

[2] Paula Carroll and Cristina Requejo. Smart grid topology designs. In *Darties, B., Poss, M.(eds.). Proceedings of the International Network Optimization Conference (INOC), June 12-14, 2019*. Open-Proceedings. org, 2019.

[3] Tarmo Korõtko, Freddy Plaum, Tobias Häring, Anna Mutule, Roberts Lazdins, Olegs Borščevskis, Argo Rosin, and Paula Carroll. Assessment of power system asset dispatch under different local energy community business models. *Energies*, 16(8):3476, 2023.

[4] Fabio Moret and Pierre Pinson. Energy collectives: A community and fairness based approach to future electricity markets. *IEEE Transactions on Power Systems*, 34(5):3994–4004, 2018.

[5] Ilia Shilov, Hélène Le Cadre, and Ana Busic. A generalized nash equilibrium analysis of the interaction between a peer-to-peer financial market and the distribution grid. In *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*. 2021, 2021.

Risk Measures in Equilibrium Energy Markets

Dáïre Byrne[1] and Mel T. Devine[2]

[1]College of Business, University College Dublin, Dublin, Ireland, ✉ daire.byrne@ucdconnect.ie
[2]College of Business, University College Dublin, Dublin, Ireland, ✉ mel.devine@ucd.ie

December 8, 2023

The amount of stochasticity faced by firms operating in energy markets has dramatically increased in recent years due to sources of uncertainty stemming from legislative changes, geopolitical unrest, and the high penetration of renewable energy sources.[8][10] This increased stochasticity exposes players in these markets to large amounts of risk and the threat of costly losses. When facing such potentially volatile scenarios, players will act with risk aversion, and it is therefore necessary that mathematical models describing the behaviour of these players be equipped with risk measures capable of quantifying this risk-averse decision-making.[2] This work will consider the impact of applying risk measures to energy markets, and in particular to multi-player equilibrium models, wherein firms compete with each other and possess market power by acting in the Cournot sense.[7] It seeks to explore the impact which arises as a consequence of incorporating risk measures into equilibrium models. We are particularly concerned with the conditional value-at-risk, second-order stochastic dominance constraints, and concave utility functions, each of which have been frequently employed in existing literature.[9][4][11] We will consider sources of uncertainty caused by stochastic costs and by stochastic demand; constructing illustrative equilibrium models for each by adapting, enhancing, and expanding the scope of existing toy models.[2][6] This analysis will encompass a discussion of the risk measures both computationally and analytically, including comparison between the risk measures and discussion of their respective advantages. It will furthermore consider the circumstances under which the equilibrium models can be converted to more tractable optimisation problems.[5] Attention is additionally paid towards Arrow-Debreu securities, which provide a prospective mechanism of completing the market.[3][10] This work seeks to build upon similar previous inquiries in a single-player context,[1] and aims to ameliorate understanding of these risk measures and the impact that their implementation has in energy markets. Such analyses will become only more relevant as the ongoing energy transition enforces the need for flexible and stochastic energy markets.

Keywords: Energy markets; Equilibrium modelling; Risk Measures; Optimization

References

[1] D. Byrne and M.T. Devine. Risk measures in energy markets. In *Optimization in Green Sustainability and Ecological Transition - ODS, Ischia, Italy, September 4–7, 2023*, AIRO Springer Series, 2024 (forthcoming).

[2] A.J. Conejo, M. Carríon, and J.M. Morales. *Decision Making Under Uncertainty in Electricity Markets*. Springer, 2010.

[3] G.d.M. d'Aertrycke, A. Ehrenmann, and Y Smeers. Investment with incomplete markets for risk: The need for long-term contracts. *Energy Policy*, 105:571–583, 2007.

[4] R. Domínguez, S. Vitali, M. Carrión, and V. Moriggia. Analysing decarbonizing strategies in the european power system applying stochastic dominance constraints. *Energy Economics*, 101:105438, 2021.

[5] R. Egging-Bratseth, T. Baltensperger, and A. Tomasgard. Solving oligopolistic equilibrium problems with convex optimization. *European Journal of Operational Research*, 284(1):44–52, 2020.

[6] R. Egging-Bratseth and A.S. Siddiqui. Stochastic equilibria with capacity expansion: Increasing expected profit with risk aversion. *Decision Analytics Journal*, 7:100234, 2023.

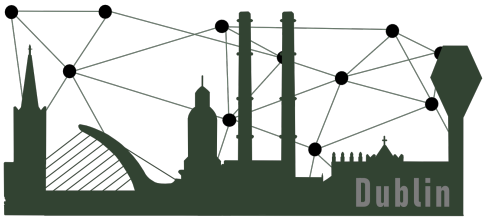
[7] S.A. Gabriel, A.J. Conejo, J.D. Fuller, B.F. Hobbs, and C. Ruiz. *Complementarity Modelling in Energy Markets*. Springer, 2013.

[8] S.J. Kazempour and P. Pinson. Effects of risk aversion on market outcomes: A stochastic two-stage equilibrium model, 1995. International Conference on Probabilistic Methods Applied to Power Systems.

[9] T. Möbius, I. Riepin, F. Müsgens, and A.H. van der Weijde. Risk aversion and flexibility options in electricity markets. *Energy Economics*, 126:106767, 2023.

[10] A. Philpott, M. Ferris, and R. Wets. Equilibrium, uncertainty and risk in hydro-thermal electricity systems. *Mathematical Programming*, 157(3):483–513, 2016.

[11] M. Shamsi and P. Cuffe. Prediction markets for probabilistic forecasting of renewable energy source. *IEEE Transactions on Sustainable Energy*, 13(2):1244–1253, April 2022.



INOC - 2024

Session Session 2B: Fairness and decision trees

Monday 11 March 2024, 15:30-17:00

Q012

Resource Planning and Equitable Work Assignment for On-site Services

Yash Kumar[1], Anantaram Balakrishnan[2], and Prakash Mirchandani[3]

[1]Operations Research and Industrial Engineering, University of Texas at Austin, TX, USA, ✉ yashkumar1803@utexas.edu
[2]McCombs School of Business, University of Texas at Austin, TX, USA, ✉ anantb@mail.utexas.edu
[3]Katz Graduate School of Business, University of Pittsburgh, Pittsburgh, PA, USA, ✉ pmirchan@pitt.edu

1 Introduction

Many service settings including healthcare, infrastructure maintenance, emergency services, and education entail providing various services at spatially distributed sites using flexible resources. The underlying planning problem is complex because it entails deciding the number of resources of each type (with varying capabilities) to use, assigning an appropriate resource (from among the different types) to each task, and sequencing and routing every resource. We seek a solution that not only minimizes the sum of resource usage, assignment, and travel costs but also allocates the work equitably among the resources. Even the simplest variations of this problem are NP-hard. The goals of our paper are to (i) develop a general formulation that effectively models this problem, (ii) compare alternate ways to model fairness, (iii) develop and test solution approaches for the problem, and (iv) evaluate the cost of imposing fairness requirements.

This problem spans two main streams of literature—resource scheduling and routing, and fairness in work assignment. Paraskevopoulos et al. [1] provide a comprehensive literature review on resource-constrained scheduling and routing. Cappanera et al. [2] study the skill vehicle routing problem to service clients with different requirements. Unlike these models which assume a fixed set of resources, our model incorporates the decisions of how many resources of each type to deploy. Also, past literature has focused on modeling the movements of individual resources, whereas by indexing variables by resource type our model can handle a large number of resources of each type as is common in practice. Matl et al. [3] review fairness models in the routing and task assignment literature with criteria such as route- and load-balancing. For the home healthcare context, Bonomi et al. [4] discuss various resource- and client-based fairness measures and develop models to include these metrics. Our model represents fairness in alternative ways, such as by constraining the highest or the lowest value across all resources for each metric, or limiting the gap between the highest and lowest values.

2 Model Development

We consider a problem setting with spatially distributed clients who have distinct service requirements. Each service type requires particular resource capabilities or skills. Resource types vary in their capabilities; some can only perform one type of service, whereas others are more flexible and can perform multiple services. We incur fixed costs for using resources, routing costs for each deployed resource to travel from location to location, and resource-to-client assignment costs. These costs can vary by resource type and service location. Resources are constrained by the overall workload that they handle and the duration of their duty. The objective is to minimize the total cost. Our model incorporates various fairness requirements such as limits, for each resource, on (i) total working time, (ii) duty time, (iii) number of desirable or or difficult tasks, and (iv) total distance travelled.

Our flow-based model uses a graph, $G = (V \cup B, A)$, where V is the set of client nodes, B is the set of nodes where resources are based, and A is the set of arcs on which resources can travel. Since there are multiple resources of each type, we index our model variables and associated parameters based on resource type $r \in R$ (versus individual resources). This approach not only vastly reduces the number of variables in the model but also overcomes solution-symmetry issues. Node $b^r \in B$, denotes resource type

r's base, and μ^r is the maximum number of type r resources that are available for deployment. Resource type r can flow over arcs in A^r and service clients in V^r. Using a resource of type r incurs a fixed cost F^r; if this resource flows over arc (i,j) to service the client at j, the cost is c^r_{ij}. We index fairness metrics by k, $k \in K$. v^{rk}_{ij} is the value of metric k when a type-r resource services node i and traverses arc (i,j). We define binary resource routing variables, x^r_{ij}, that equal 1 if resource type r traverses arc (i,j), and 0 otherwise. To model fairness, we introduce variables u^k_{ij} that denote the cumulative value of metric k if any resource traverses arc (i,j) before it services node j, and is 0 otherwise.

We use l, $l \in L$, to index the fairness criteria, and define $\Phi^{lk}(\mathbf{x},\mathbf{u})$ for each metric k and criteria l. The function $\Phi^{lk}(\mathbf{x},\mathbf{u})$ can capture many different types of fairness requirements, and depends on the assignment and routing decisions. For example, $\Phi^{lk}(\mathbf{x},\mathbf{u})$ can model the maximum or minimum of the u^k_{ij} values across all resources, or the range between these values. The parameter ζ^{lk} is the bound imposed on this function. The model is given below.

$$\min \quad \Sigma_{(i,j)\in A}\Sigma_{r\in R:(i,j)\in A^r} c^r_{ij}x^r_{ij} + \Sigma_{(b^r,j)\in A}\Sigma_{r\in R:(b^r,j)\in A^r} F^r x^r_{b^r j} \tag{1}$$

$$\text{s.t.} \quad \Sigma_{i:(i,j)\in A}\Sigma_{r\in R:(i,j)\in A^r} x^r_{ij} = 1 \qquad \forall j \in V, \tag{2}$$

$$\Sigma_{i:(j,i)\in A^r} x^r_{ji} = \Sigma_{i:(i,j)\in A^r} x^r_{ij} \qquad \forall r \in R, j \in V^r \cup \{b^r\}, \tag{3}$$

$$\Sigma_{(b^r,j)\in A^r} x^r_{b^r j} \le \mu^r \qquad \forall r \in R, \tag{4}$$

$$\Sigma_{i:(j,i)\in A} u^k_{ji} - \Sigma_{i:(i,j)\in A} u^k_{ij} = \Sigma_{i:(j,i)\in A}\Sigma_{r\in R:(j,i)\in A^r} v^{rk}_{ji} x^r_{ji} \qquad \forall k \in K, \forall j \in V, \tag{5}$$

$$u^k_{ij} \le \Sigma_{r\in R:(i,j)\in A^r} U^{rk}_j x^r_{ij} \qquad \forall k \in K, \forall (i,j) \in A : i \notin B, \tag{6a}$$

$$u^k_{bj} = \Sigma_{r\in R:(b,j)\in A^r} v^{rk}_{bj} x^r_{bj} \qquad \forall k \in K, \forall (b,j) \in A : b \in B, \tag{6b}$$

$$\Phi^{lk}(\mathbf{x},\mathbf{u}) \le \zeta^{lk} \qquad \forall k \in K, \forall l \in L, \tag{7}$$

$$x^r_{ij} \in \{0,1\} \qquad \forall r \in R, (i,j) \in A^r, \tag{8a}$$

$$u^k_{ij} \ge 0 \qquad \forall k \in K, (i,j) \in A. \tag{8b}$$

The objective (1) minimizes the sum of resource usage, assignment, and routing costs. Constraints (2) ensure that each client is served by exactly one resource. Constraints (3) enforce conservation of resource flows, while constraints (4) specify the maximum number of available resources of each type $r \in R$. Constraints (5) compute the cumulative fairness metric value for the resource arriving at client j. Constraints (6a, 6b) link these values to the routing variables x^r_{ij}, where U^{rk}_j is metric k's upper bound on the route for resource type r to node j before servicing it. These constraints also ensure that the solution does not contain any subtours that do not span a base node. Finally, constraints (7) bound the value of fairness metric k for criteria l. Constraints (8) are the binary and nonnegativity constraints.

To strengthen the linear programming relaxation model, we develop some valid inequalities. We conduct various computational tests to: (a) identify the main drivers of computational performance; (b) compare solution effectiveness for alternative representations of fairness requirements; and (c) assess the cost impact of imposing fairness requirements.

References

[1] D. C. Paraskevopoulos, G. Laporte, P. P. Repoussis, and C. D. Tarantilis. Resource constrained routing and scheduling: Review and research prospects. *Eur. J. Oper. Res.*, 263(3):737–754, 2017.

[2] P. Cappanera, L. Gouveia, and M. G. Scutellà. Models and valid inequalities to asymmetric skill-based routing problems. *EURO J. Transp. Logist.*, 2(1-2):29–55, 2013.

[3] P. Matl, R. F. Hartl, and T. Vidal. Workload Equity in Vehicle Routing Problems: A Survey and Analysis. *Transp. Sci.*, 52(2):239–260, 2018.

[4] V. Bonomi, R. Mansini, and R. Zanotti. Mediating governance goals with patients and nurses satisfaction: a multi-actor multi-objective problem including fairness. *Int. J. Prod. Res.*, pages 1–28, 2023.

Cardinality and fairness constrained clustering using k-means

Antoine Obled[1] and Marta Pascoal[2]

[1]ENSTA Paris, Institut Polytechnique de Paris, France, ✉ antoine.obled@ensta-paris.fr
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy; University of Coimbra; INESC-Coimbra, Portugal, ✉ marta.brazpascoal@polimi.it

1 Introduction

Clustering consists in partitioning a given set of data points in such a way that the points in a cluster are more similar than those within the remaining ones. Assuming that $k \in \mathbb{N}$ is the intended number of clusters, in the k-clustering problem each group is represented by a center point and the most common criterion to optimize is the mean square error, given by:

$$\sum_{j=1}^{k} \sum_{i=1}^{n} ||X_{i\cdot} - Y_{j\cdot}||^2$$

where $X_{i\cdot}$ represents the data point i, $i = 1, \ldots, n$, and $Y_{j\cdot}$ represents the cluster j's center, $j = 1, \ldots, k$. The k-means algorithm [4] is a classical method used to solve this type of problem. It is simple to apply and provides a local minimum for the problem, upon two steps that alternate until the solution stops changing:

1. assigning the data points to the current clusters (that is, to the current center points);
2. updating the center points according to the new assignment.

The solution for the k-clustering problem may contain empty clusters or clusters with few points. This issue can be overcome by limiting the cardinality of the clusters above and below according to bounds known in advance. This variant of the problem is know as the cardinality constrained k-clustering problem. Balanced k-clustering is a particular case of this variant, for which the sought partition of the points should be as uniform as possible in terms of cardinality [2, 3]. Therefore, the sizes of the clusters can vary by at most one data point.

2 Resolution method

We first address the cardinality constrained k-clustering problem with lower and upper bounds on the number of points in each cluster, and describe a k-means-based algorithm to solve it. The assignment step of this algorithm is formulated as a minimum cost flow problem defined on a suitable graph. Like in the traditional approach, the data points correspond to supply nodes in the graph whereas the clusters correspond to demand nodes. In addition, a dummy node is considered to balance the problem and simultaneously to impose the minimum cardinality for each cluster. Naturally, this applies also to the balanced k-clustering problem.

We then extend the same approach to solve the fairness balanced k-clustering problem [1]. In this variant the data points are classified into different groups/types, and the goal of the problem is to identify clusters that are balanced with respect to the cardinality in such a way that the different groups/types are represented in the various clusters in a balanced manner.

3 Numerical results

The proposed methods were implemented in Julia, using Gurobi as the solver. Comparative empirical tests ran on an 11$^{\text{th}}$ Gen Intel®Core™i7-1195G7 with 2.9 GHz, 1805 MHz, 4 cores and 16 Go RAM. The

experiments used the benchmark data sets listed on Table 1. The points in the data sets `Adult_2` and `Adult_5` are distinguished into two and five groups/types, respectively.

Name	n	k	Source
si, $i = 1, 2, 3, 4$	5 000	15	`http://cs.uef.fi/sipu/datasets/`
`MNIST_test`	10 000	10	`http://cs.uef.fi/sipu/datasets/`
`MNIST_train`	60 000	10	`http://cs.uef.fi/sipu/datasets/`
`ConfLong`	164 860	11	`http://cs.uef.fi/sipu/datasets/`
`Adult_`i, $i = 2, 5$	32 561	10	`https://archive.ics.uci.edu/dataset/2/adult`

Table 1: Characteristics of the data sets and the experiments

Different intervals between the upper and the lower bounds for the sizes of the clusters were considered for the cardinality constrained k-clustering problem. All these instances were solved in less than 2 seconds. The results report that wider intervals require more iterations and bigger run times. For the balanced case our method was compared with the regularized k-means method presented in [2], `RKM`, which penalizes the objective function whenever the cardinality balance constraint is violated. In particular, the objective function values output by both approaches is the same. In terms of the run time, the speed-up of our method over `RKM` varied between 58% and 77%. In particular, the biggest instance was solved in approximately 100 seconds. Finally, fairness balanced constrained k-clustering was compared with the problem that does not consider the fairness of the clusters for the data sets `Adult_`i, $i = 2, 5$. As expected, fairness indices increase with the new method. Furthermore, the instances were solved in up to 20 seconds. The results of the computational experiments are discussed in more detail.

Acknowledgment The work of Marta Pascoal was partially supported by the Portuguese Foundation for Science and Technology (FCT) under project grants UID/MAT/00324/2020 and UID/MULTI/00308/-2020.

References

[1] M. Ghadiri, S. Samadi, and S. S.Vempala. Fair k-means clustering. *CoRR*, abs/2006.10085, 2020.

[2] W. Lin, Z. He, and M. Xiao. Balanced clustering: A uniform model and fast algorithm. In *IJCAI*, pages 2987–2993, 2019.

[3] Y. Lin, H. Tang, Y. Li, C. Fang, Z. Xu, Y. Zhou, and A. Zhou. Generating clusters of similar sizes by constrained balanced clustering. *Applied intelligence*, 52:5273–5289, 2022.

[4] S. Lloyd. Least squares quantization in PCM. *IEEE transactions on information theory*, 28:129–137, 1982.

Soft regression trees: a model variant and a decomposition training algorithm

E. Amaldi[1], A. Consolo[1], and A. Manno[2]

[1]DEIB, Politecnico di Milano, Milano, Italy , ✉ edoardo.amaldi/antonio.consolo@polimi.it
[2]DISIM, Università degli Studi dell'Aquila, L'Aquila, Italy, ✉ andrea.manno@univaq.it

Decision trees are popular supervised Machine Learning (ML) methods for classification and regression tasks. They are widely used in a variety of application fields ranging from Business Analytics to Medicine and Biology. The success of decision trees lies mainly in their interpretability and good accuracy. Unlike most black-box ML models, they reveal the decisions leading to the tree response for any input vector. Interpretability is of particular importance in applications where the ML models support domain experts decisions and justifiable predictions are required, such as for instance in medical diagnosis.

A decision tree is a directed binary tree with a set of internal nodes, referred to as branch nodes, including the root, a set of leaf nodes, and two outgoing arcs (branches) for each branch node associated to a splitting rule applied to the input space (feature space). Once values are assigned to the tree parameters (those associated to the branch nodes and those associated to the leaf nodes), any input vector is routed from the root along the tree according to the splitting rules at the branch nodes, eventually falling into a leaf node where the output (the linear prediction or the class number) is determined. Hard or soft splitting rules can be considered at branch nodes. In hard (deterministic) splits, the left branch is followed if a single feature (univariate case) or a linear combination of the features (multivariate case) exceeds a given threshold value. In soft splits, both left and right branches are followed with complementary probabilities given by a continuous sigmoid function applied to a linear combination of the features.

In the seminal CART method for building univariate classification and regression trees [3], a topdown and greedy approach is adopted: at each branch node a split is determined by minimizing a local impurity measure. In the later variants C4:5 and ID3, the greedy approach is combined with a pruning phase to decrease the tree size and to improve the testing accuracy by reducing overfitting. Since training decision trees is known to be NP-hard, it is a challenging problem to globally optimize them over all tree parameters.

During the past decade, growing attention has been devoted to globally optimized (trained) decision trees with deterministic or soft splitting rules at branch nodes, leveraging on the remarkable advances in mixed integer optimization and unconstrained nonlinear optimization.

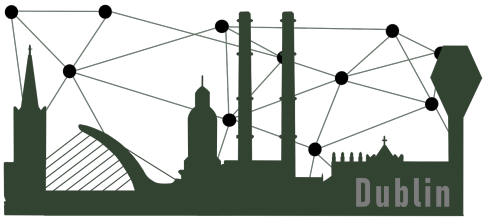
In this work, we propose a variant of soft multivariate regression trees (SRTs) where, for every input vector, a potential linear prediction is available at each leaf node as a linear regression of the features but the actual tree prediction is the one associated to a single specific leaf node. Our nonlinear optimization formulation for training such soft trees is well-suited to decomposition and to impose fairness constraints. After showing a universal approximation result for SRTs, we present a node-based decomposition training algorithm which includes a specific initialization procedure and a heuristic for reassigning the data points along the tree. Under mild assumptions, we also establish asymptotic convergence guarantees. Experiments on 15 data sets from the UCI and KEEL repositories indicate that our model variant and decomposition algorithm yield improved testing accuracy compared with the soft regression trees discussed in [2] by Blanquero et al., and significant speed-up in training time and similar testing accuracy compared with the mixed integer optimization approach described in [1] by Bertsimas and Dunn.

References

[1] Bertsimas D, Dunn J (2019) Machine learning under a modern optimization lens (Dynamic Ideas LLC).

[2] Blanquero R, Carrizosa E, Molero-Rıo C, Morales DR (2022) On sparse optimal regression trees. European Journal of Operational Research 299(3):1045–1054.

[3] Breiman L, Friedman J, Stone CJ, Olshen RA (1984) Classification and regression trees (CRC press).



INOC - 2024

Session Session 2C: Sustainable Mobility and Transportation

Monday 11 March 2024, 15:30-17:00

Q013

Digraphs and k-Domination Models for Facility Location Problems in Road Networks: Greedy Heuristics

Lukas Dijkstra
School of Mathematics, Cardiff University
Cardiff, Wales, UK
DijkstraL@cardiff.ac.uk

Andrei Gagarin
School of Mathematics, Cardiff University
Cardiff, Wales, UK
GagarinA@cardiff.ac.uk

Padraig Corcoran
School of Computer Science & Informatics, Cardiff University
Cardiff, Wales, UK
CorcoranP@cardiff.ac.uk

Rhyd Lewis
School of Mathematics, Cardiff University
Cardiff, Wales, UK
LewisR9@cardiff.ac.uk

ABSTRACT

We consider modelling the placement of refuelling facilities for alternative fuel vehicles in road networks by using directed graphs and k-dominating sets. The concept of a reachability digraph corresponding to a road network is introduced, and three greedy heuristics are proposed and experimentally tested to search for k-dominating sets in two types of digraphs, including the reachability digraphs of road networks. These simple and efficient heuristics show that refined greedy strategies usually provide better results for large as well as small digraphs, and their results are reasonably close to exact solutions for small digraphs. Combining the greedy strategies with some randomized heuristic ideas helps to improve the results even further in the case of digraphs associated with the road networks.

1 INTRODUCTION

1.1 Motivation

Dominating sets in simple graphs and networks have attracted a lot of attention from different perspectives, be they theoretical [1, 4, 12] or more applied [8, 19–21] in nature. However, directed graphs, or digraphs, which are more general abstract models in comparison to the simple graphs, are often overlooked. For example, the classic book on digraphs [2] does not pay much attention to dominating sets, and the classic book on dominating sets [12] does not pay much attention to digraphs. Digraphs offer advantages of more subtle modelling tools though, like representing one-way streets in road networks. Also, digraphs allow us to account for separate costs or differences in fuel consumption depending on which direction a road is travelled. These properties of digraphs are very useful when modelling road networks, a major area of application of graph theory [3, 5, 7, 8, 21]. On the other hand, in simple graphs it is not clear how to represent one-way streets or roads that are on an incline, causing fuel consumption to differ dramatically depending on whether a vehicle is going up- or down-hill.

In this paper, we consider the concept of k-dominating sets in the context of digraphs, in particular, from the perspective of the facility location problems in large-scale road networks (e.g., for this application context, see results for the reachability simple graph model defined in [8]). One of the classic NP-complete problems is to find a smallest size dominating set in a simple graph [10]. This problem is also known to be APX-hard [18] and, in general, is not fixed-parameter tractable [6]. Naturally, these complexity issues apply to finding smallest size k-dominating sets in digraphs as generalizations of the corresponding concepts and structures. Here, we focus on simple, yet effective heuristics that produce efficient results for large digraphs. To this end, we put forward a number of greedy heuristics to solve the k-dominating set problem in digraphs for small values of k. We run computational experiments to illustrate their effectiveness, considering both randomly generated digraphs as well as reachability digraphs corresponding to real world road networks. Some potential randomized extensions of the greedy strategies are also considered.

1.2 Basic Definitions, Notions, and Notation

A digraph D is defined as $D = (V, A)$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of vertices and $A = \{e_1, e_2, ..., e_m\}$ is a set of ordered pairs of vertices called arcs. So, each arc $e \in A$ is of the form $e = (v_i, v_j)$ for some $i, j \in \{1, ..., n\}$, $i \neq j$. An arc $e = (v_i, v_j)$ as well as its inverse $e^{-1} = (v_j, v_i)$ may both be included in A and, in this case, are treated as independent entities. Thus, $0 \leq m \leq n(n-1)$.

The out-neighborhood of a vertex $v \in V$ is defined as the set $N^+(v) = \{u \in V \,|\, (v, u) \in A\}$, i.e. the set of vertices that are directly reachable from v by traversing exactly one arc. On the other hand, the in-neighbourhood of v, $N^-(v) = \{u \in V \,|\, (u, v) \in A\}$, is the set of vertices that have an arc leaving them that leads to v. Additionally, the closed out-neighbourhood of v is defined to be $N^+[v] = N^+(v) \cup \{v\}$. Similarly, the closed in-neighbourhood of v is defined to be $N^-[v] = N^-(v) \cup \{v\}$. The out-degree of v is $d^+(v) = |N^+(v)|$, and the in-degree of v is $d^-(v) = |N^-(v)|$. These are the numbers of vertices directly reachable from v and such that v is directly reachable from them, respectively. The minimum out- and in-degrees of D are denoted by $\delta^+ = min\{d^+(v) \,|\, v \in V\}$ and $\delta^- = min\{d^-(v) \,|\, v \in V\}$, respectively. These are the smallest degrees found across all vertices in the digraph.

Given a set of vertices $X \subseteq V$, a vertex v is said to be *covered* by X if $N^-[v] \cap X \neq \emptyset$, i.e. when v is either in the set X or can be directly reached via an arc from a vertex in X. The set of vertices covered by X is denoted by $C(X) = \{v \in V \mid N^-[v] \cap X \neq \emptyset\}$. The set X is called a *dominating set* of D if $C(X) = V$, i.e. when every vertex of D is either in X or directly reachable from a vertex in X. More generally, for any integer $k \geq 1$, v is said to be *k-covered* by X if either $v \in X$ or $|N^-(v) \cap X| \geq k$. In other words, v is either in X or directly reachable by arcs from at least k vertices in X. The set of vertices that are k-covered by X in D is denoted by $C_k(X)$, and X is a *k-dominating set* of D if $C_k(X) = V$. Note that, in these terms, covering by a set of vertices and a dominating set are simply the case of $k = 1$.

A k-dominating set X of D is *minimal* (by inclusion) if no vertex can be removed from X without the resulting set losing the k-dominating set property, i.e. if we have $C(X \setminus \{v\}) \neq V$ for every $v \in X$. A k-dominating set X of D is a *minimum* k-dominating set if there does not exist a k-dominating set Y of D of a smaller size. The k-domination number of a digraph D is the size of a minimum k-dominating set of D, which is denoted by $\gamma_k(D)$. Some basic theoretical results for the k-domination number of digraphs can be found in [16].

Thus, given a digraph D, we are interested in the problem of finding small-sized k-dominating sets in D, while using $\gamma_k(D)$ as a quality benchmark, whenever possible. Also, we want to find such sets of vertices in D quickly.

2 HEURISTICS

Recent research focused on efficient bespoke heuristics to search for small k-dominating sets in simple graphs [5, 8]. Here we propose and consider three different greedy heuristic methods to tackle a similar problem in digraphs. These heuristics are called Basic Greedy (Algorithm 2), Deficiency Coverage Greedy (Algorithm 3), and Two-Criteria Greedy (Algorithm 4). A fourth heuristic method, relying on a combination of greedy and randomized ideas, is also proposed. Each of the heuristics starts by finding a k-dominating set of the digraph, which is usually not minimal (by inclusion). Therefore, at the end of the four main heuristics, an additional greedy heuristic is run to remove unnecessary vertices and to reduce the initially found set to a minimal k-dominating subset, or to check minimality of the initially found set. This Minimal k-Dominating Subset greedy heuristic is described by Algorithm 1.

2.1 Main Greedy Heuristics

An intuitive basic greedy strategy to find a k-dominating set in a simple graph is to start with an empty set X and to add vertices into X, one at a time, by choosing iteratively a vertex with the most vertices in its closed neighbourhood that are not yet k-covered by X. This recursive procedure can be repeated until all vertices of the graph are k-covered by X, at which point X is a k-dominating set. This strategy has been studied previously in the context of domination [1, 17] as well as k-domination [5, 8] in simple graphs. This basic greedy strategy generalizes to digraphs by checking specifically the closed out-neighbourhood of vertices at each step in iteration. This is described in the pseudocode of Algorithm 2.

Algorithm 1: Minimal k-Dominating Subset

Input: A digraph $D = (V, A)$, an integer $k \geq 1$, a k-dominating set X of D.
Output: A minimal k-dominating set Y of D.

begin
  foreach $v \in X$ **do**
    Determine $x_v = |N^+(v) \setminus X|$
  end
  Initialize $Y = X$
  while $X \neq \emptyset$ **do**
    Find a vertex $v \in U = \arg\min_{u \in X} x_u$
    if $C_k(Y \setminus \{v\}) = V$ **then**
      Put $Y = Y \setminus \{v\}$
    end
    Put $X = X \setminus \{v\}$
  end
  return Y
end

Algorithm 2: Basic Greedy

Input: A digraph $D = (V, A)$, an integer $k \geq 1$.
Output: A minimal k-dominating set Y of D.

begin
  Initialize $X = \emptyset$
  while $C_k(X) \neq V$ **do**
    Find a vertex $v \in U = \arg\max_{u \in V \setminus X} |N^+[u] \setminus C_k(X)|$
    Put $X = X \cup \{v\}$
  end
  Find a minimal k-dominating set $Y \subseteq X$
  return Y
end

It is important to note that, in the case of $k > 1$, when searching for a k-dominating set of a (di)graph, vertices that are not yet k-covered by some vertex set X can have different numbers of (in-) neighbours already in X. As a consequence, it can be more difficult to k-cover these vertices by their (in-)neighbours while expanding X in the (di)graph. On the other hand, since including a vertex into X results in k-covering this vertex regardless of how many (in-)neighbours the vertex has in X, it maybe more interesting to prioritize adding into X the vertices that are not well k-covered yet to reduce the amount of vertices ((in-)neighbours) needed for their k-covering later.

Therefore, given a set of vertices $X \subseteq V$ of a digraph $D = (V, A)$ and an integer $k \geq 1$, the deficiency of a vertex $v \in V \setminus X$ is defined as $l_k(v, X) = \max\{0,\ k - |N^-(v) \cap X|\}$. This represents the amount of in-neighbours that are still needed to completely k-cover v in the digraph. Algorithm 3, called Deficiency Coverage Greedy, is described below. It follows a modified greedy strategy of Basic Greedy of Algorithm 2. In contrast to Basic Greedy, Deficiency Coverage Greedy finds k-dominating sets by selecting vertices not only by their number of not k-covered out-neighbours, but also by the remaining deficiency of the vertex itself.

Algorithm 3: Deficiency Coverage Greedy

Input: A digraph $D = (V, A)$, an integer $k \geq 1$.
Output: A minimal k-dominating set Y of D.

begin
  Initialize set $X = \emptyset$
  while $C_k(X) \neq V$ **do**
    Find a set $U = \arg\max_{u \in V \setminus X} |N^+(u) \setminus C_k(X)| + l_k(u, X)$
    Select $v \in U$ `/* uniformly at random */`
    Put $X = X \cup \{v\}$
  end
  Find a minimal k-dominating set $Y \subseteq X$
  return Y
end

Another greedy strategy, introduced and computationally tested as a part of this research, can be considered as a refinement of Deficiency Coverage Greedy. In the Deficiency Coverage Greedy strategy, when several vertices can be used as the best candidates to be included into a set X under construction, the algorithm chooses one of them uniformly at random. Instead, it is possible to make choice of the best candidate by considering the out-neighbours of each of these equally-ranked vertices.

Given a vertex $v \in V$ of a digraph $D = (V, A)$, we define the total out-neighbour in-degree of v to be $f_D(v) = \sum_{u \in N^+(v)} d^-(u)$. The additional greedy strategy uses the following heuristic assumption and observations. Since a vertex of low in-degree has fewer possible ways to be eventually k-covered in the digraph by its in-neighbours, if there is no efficient way to k-cover it, such a vertex is likely to be included in the k-dominating set in a later iteration, in particular, if its out-degree is much higher than its in-degree. Therefore, Algorithm 4, called Two-Criteria Greedy, prioritizes vertices v with a higher total out-neighbour in-degree $f_D(v)$ in iteration. This is to discourage adding vertices with lower in-degree out-neighbours, because such out-neighbours are likely to be included themselves into the set under construction at a later point of time, which would reduce effectiveness of including the original vertex during the process. This Two-Criteria Greedy method is described in Algorithm 4.

Algorithm 4: Two-Criteria Greedy

Input: A digraph $D = (V, A)$, an integer $k \geq 1$.
Output: A minimal k-dominating set Y of D.

begin
  Initialize set $X = \emptyset$
  while $C_k(X) \neq V$ **do**
    Find a set $U = \arg\max_{u \in V \setminus X} |N^+(u) \setminus C_k(X)| + l_k(u, X)$
    Find a vertex $v \in U' = \arg\max_{u \in U} f_D(u)$
    Put $X = X \cup \{v\}$
  end
  Find a minimal k-dominating set $Y \subseteq X$
  return Y
end

The worst-case complexity analysis shows that all these greedy heuristics can be implemented to run in $O(nm)$ time. This agrees with our implementation, for which the worst-case analysis provides a more detailed upper bound of $O(n(n + m))$.

2.2 Combining with a Randomized Heuristic

Although the algorithms above have some flexibility for the choice of a vertex at each iteration, they are very restrictive by their greedy selection nature. To fix this issue and to make them more flexible, one can try to use analytical tools and add more randomized components to the greedy strategies. In other words, we can combine the greedy strategies, for example, with a basic randomized technique.

A simple and efficient approach to make the greedy strategies above more flexible can consist in determining an initial random subset of vertices of a digraph for the greedy heuristics to start with (instead of an empty set). To do this in a more subtle and justified way, one can use a probabilistic method and corresponding analytical tools. Suppose we find an initial subset X of vertices for a k-dominating set by including (or not) each vertex of the digraph into X with some fixed probability p (respectively, $1 - p$). One way to optimize this probability p is to use ideas from the probabilistic method.

The basic probabilistic method is a well-studied analytical tool [1, 9, 13], which can be used, for example, to find an upper bound for the domination number of a simple graph $G = (V, E)$. It can be summarized as follows. Suppose we have some probability $p \in [0, 1]$ to be specified or optimized later. First, find a random subset S of vertices of G by including each vertex of G into S independently with probability p. Then, we have the subset $R = V \setminus C(S)$ of vertices which are not covered by S in G. Now, $S \cup R$ is a dominating set of G, as all the vertices not covered by S have simply been included into the set. The expected cardinality $\mathbb{E}(|S \cup R|) = \mathbb{E}(|S|) + \mathbb{E}(|R|)$ of this set can be computed explicitly in terms of p and is an upper bound for the domination number $\gamma(G)$. The justification is straightforward: there must exist at least one dominating set obtained by using this method which has its cardinality at most the expected value. Since the expected cardinality can be considered as a function of p, it can be optimized with respect to p to give the best possible upper bound for $\gamma(G)$.

This approach has been generalized and applied to k-dominating sets in simple graphs. One of the best known results is as follows.

Theorem 2.1 ([9]). *Given a simple graph $G = (V, E)$ with minimum vertex degree δ and some integer k, $1 \leq k \leq \delta$,*

$$\gamma_k(G) \leq \left(1 - \frac{\delta'}{\binom{\delta}{k-1}^{1/\delta'} \cdot (1 + \delta')^{1+1/\delta'}}\right) n,$$

where $\delta' = \delta - k + 1$.

The probability used to find this optimized upper bound in general simple graphs is $p = 1 - \frac{1}{\sqrt[\delta']{\binom{\delta}{k-1}(1+\delta')}}$. However, as shown by the computational experiments in [8], this probability is too high for the reachability graphs of road networks. Therefore, instead of the minimum vertex degree $\delta = \delta(G)$, alternative degree parameters of G, such as the mean and median vertex degrees, have been considered and used in the above algebraic expression.

Although the probabilistic method has been mainly used with simple graphs, it can be applied to digraphs as well. Lee [13] generalized the basic result of [1] to the domination number of digraphs, i.e. for the case of $k = 1$. We have obtained the corresponding result for the k-domination number of digraphs in general.

Theorem 2.2. *Given a digraph $D = (V, A)$ with minimum in-degree δ^- and some integer k, $1 \leq k \leq \delta^-$,*

$$\gamma_k(D) \leq \left(1 - \frac{\delta^{-\prime}}{\binom{\delta^-}{k-1}^{1/\delta^{-\prime}} \cdot (1+\delta^{-\prime})^{1+1/\delta^{-\prime}}}\right) n,$$

where $\delta^{-\prime} = \delta^- - k + 1$.

After optimization, the probability used to find this upper bound is $p = 1 - \frac{1}{\sqrt[\delta^{-\prime}]{\binom{\delta^-}{k-1}(1+\delta^{-\prime})}}$. Therefore, we use this algebraic expression for probability p to find an initial random subset of vertices in a digraph in an attempt to improve the results of greedy heuristics of Section 2.1. The complexity of finding an initial random subset of vertices in such a way is $O(n^2)$. It takes a linear time to decide with probability p for each vertex v_i whether to include or not v_i in the subset, $i = 1, 2, \ldots, n$. However, computing p involves computing the binomial coefficient, which can be done in $O(n^2)$ time in this case.

3 EXPERIMENTAL RESULTS

A number of computational experiments were run to test the heuristics from Section 2 and to compare and analyze the results. Two different types of digraphs were used in these experiments. Digraphs of the first type are randomly generated by using the so-called Erdős–Rényi (ER) random digraph model. This consists in taking a set of vertices, and, by using some fixed probability p, independently for each ordered pair of vertices, we decide whether the corresponding arc is in the digraph or not. Note that the two possible arcs between a pair of vertices are considered separately and therefore included or not into the digraph independently from each other.

The second type of digraphs are so-called reachability digraphs derived from actual road networks. A similar concept of a reachability graph is defined in [8] for simple graphs. In the reachability digraph model, vertices represent some locations in the road network. An arc from one vertex to another is included into the reachability digraph if it is possible to travel from the location corresponding to the first vertex to the location of the other vertex within a certain predefined road distance. The maximum travelling distance to have an arc is called the reachability radius (r) in the road network. In comparison to the ER random digraphs, the reachability digraphs are more similar to simple graphs, because many streets support two-way traffic. However, although most connections in a reachability digraph are two-way, there is still a non-negligible number of one-way connections that would be ignored in a simple graph model. This is illustrated in Figure 1, where the arcs originating from the blue vertex are leading only to the red veritices.

Before running computational experiments on large digraph instances, we considered small size digraphs to be able to obtain some exact solutions. This allows us to compute $\gamma_k(D)$ to compare the

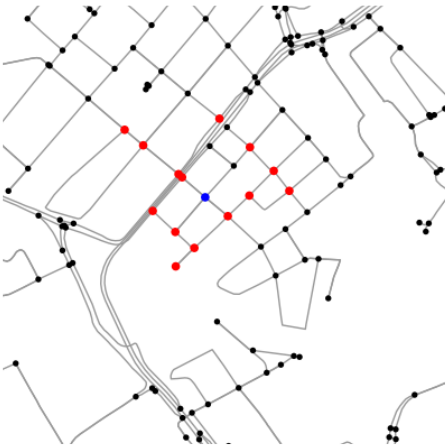


Figure 1: A vertex (blue) and its out-neighbours (red) in a reachability digraph.

greedy heuristics results. The exact deterministic solutions were obtained by solving an integer-linear programming (ILP) formulation of the problem by using Gurobi 10.0.1 [11]. We used the following ILP formulation:

$$\begin{aligned} \text{minimize} \quad & z(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} x_i \\ \text{subject to:} \quad & kx_i + \sum_{v_j \in N^-(v_i)} x_j \geq k, \quad i = 1, \ldots, n \\ & x_i \in \{0, 1\}, \qquad\qquad i = 1, \ldots, n, \end{aligned}$$

where x_i is a binary variable indicating whether vertex v_i of the digraph is included in the k-dominating set or not, $i = 1, \ldots, n$. When it was not possible to solve the problem of computing $\gamma_k(D)$ in a reasonable amount of time, the generic ILP solver was run as an alternative heuristic solver using a substantial amount of CPU time resources.

We have considered k-domination for $k = 1, 2, 4, 8$. For some small-size digraphs, the deterministic method (ILP) was able to return an optimal solution within a reasonable timeframe for lower values of k, but started experiencing infeasibly large runtimes for higher values of k. Therefore, a time limit of 24 hours was imposed and, if it was reached, the best solution found so far (i.e. heuristic) would be recorded instead of the exact solution. The experiments were conducted by using C++ on a PC with a 3.00 GHz Intel Core i5 processor and 16 GB of RAM, running Windows 10 Education, version 21H2. In the tables, *max* denotes the 24-hour time limit for the ILP solver; BG, DCG, and TCG stand for Basic Greedy, Deficiency

n	k	ILP		BG		DCG		TCG	
		Size	Time (s)	Size	Time (s)	Size	Time (s)	Size	Time (s)
100	1	12	0.2	14	0.00057	12	0.00059	12	0.00055
	2	21	1.98	24	0.00067	24	0.00073	24	0.00069
	4	36	5.72	45	0.0009	38	0.00091	40	0.00084
	8	64	1.46	71	0.0011	66	0.0012	66	0.0011
200	1	15	564.9	19	0.0014	17	0.0014	16	0.0014
	2	25	*max*	28	0.0015	27	0.0016	27	0.0015
	4	44	*max*	47	0.003	47	0.0021	46	0.0019
	8	76	*max*	85	0.0027	83	0.0041	83	0.0071

Table 1: k-Dominating sets in small Erdős–Rényi digraphs.

n	k	BG		DCG		TCG	
		Size	Time (s)	Size	Time (s)	Size	Time (s)
50, 000	1	52	21.95	51	22.34	52	25.47
	2	73	23.27	72	21.68	72	25.95
	4	105	23.21	104	21.65	105	26.29
	8	165	23.72	164	23.16	163	25.9
100, 000	1	58	1654	58	1392	57	1407
	2	78	1470	78	1422	78	1453
	4	113	1521	113	1576	112	1587
	8	174	1574	173	1646	172	1598

Table 2: k-Dominating sets in large Erdős–Rényi digraphs.

Coverage Greedy, and Two-Criteria Greedy, respectively. Preliminary experiments with the greedy algorithms combined with a randomized heuristic of Section 2.2 have not shown any improvements in the case of Erdős–Rényi digraphs yet. However, they do show some improvements in the case of reachability digraphs of road networks (not included in the tables).

3.1 Erdős–Rényi digraphs

All of the Erdős–Rényi digraphs used in these experiments were generated using arc inclusion probability of $p = 0.1$. The digraphs for the small-scale experiments had $n = 100, 150$, and 200 vertices, whilst the large-size digraphs contained 25, 50, 75, and 100 thousand vertices. An experiment on a digraph with 125 thousand vertices was also attempted, but the computer ran out of memory. Some of the results of these experiments are presented in Tables 1 and 2.

Table 1 shows that the proposed greedy heuristics solve the small-size instances of the problem in milliseconds, while the solution quality is comparable to the exact or heuristic ILP solutions after running the generic ILP solver on these test instances for a much longer period time (from 10^3 to 10^5 times longer to obtain an exact solution, and 10^7 times longer to obtain alternative heuristic solutions). Also, Tables 1 and 2 show that, among the three greedy solvers, DCG and TCG usually provide better results, and their runtimes are always within a reasonable time limit (less than 30 min for a digraph on 100, 000 vertices).

3.2 West Midlands Conurbation road networks

We constructed digraphs corresponding to road networks by using OpenStreetMap (OSM) geographic information system data [15]. The corresponding road networks are comprised of all roads contained within a square box, the center of which is the Birmingham New Street train station in the United Kingdom (exact coordinates: 52.478691, -1.89984). The digraphs for the small-scale experiments are given by the box side-lengths of 1, 1.25, 1.5, 1.75, and 2 kilometers, with reachability radii of $r = 300, 325, 350, 375$, and 400 meters, respectively. The large size digraphs are given by the box side-lengths of 10, 20, 30, 40, and 50 kilometers, with the reachability radii of 3, 4, 5, 6, and 7 kilometers, respectively. An experiment on a digraph corresponding to a road network of the box side-length of 60 kilometers and with the reachability radius of 8 kilometers was attempted, but the computer ran out of memory. Some of the results of these computational experiments are presented in Tables 3 and 4 below.

Similarly to the small Erdős–Rényi digraphs, Table 3 shows that the solution quality of the proposed greedy heuristics is comparable to the exact or heuristic ILP solutions after running the generic ILP solver on these small reachability digraphs for a much longer time (one to three orders of magnitude more time to obtain exact ILP solutions, and five to six orders of magnitude more time to obtain alternative heuristic solutions). Also, Table 3 shows that, for the small reachability digraphs, when $k > 1$, TCG usually provides better results than the other two greedy heuristics, and the advantages of DCG and TCG over BG become more visible for the larger values of k. For the large reachability digraphs, Table 4 shows that TCG provides the best results for all but two problem instances (out of twelve), which are better solved by DCG. BG is still competitive for $k = 1$, but for larger values of $k > 1$, the advantages of DCG and TCG are more visible again. Notice that, for $k = 1$, DCG would normally produce the same results as BG (TCG has the secondary selection criterion, which comes into play even when $k = 1$). However, the random choice of a vertex among the equally most suitable candidates in iteration of DCG produces slightly different from BG results and introduces the option of running the algorithm several times to potentially obtain better results. The runtimes of greedy heuristics on the same digraph instance are always comparable, and within a reasonable time limit (less than 30 min for the reachability digraph on 225289 vertices).

4 CONCLUDING REMARKS

In this research, we have considered and accentuated using digraphs for modelling problems in road networks, introduced the concept of a reachability digraph corresponding to a road network, proposed modelling and optimization of facility locations in road networks by considering k-dominating sets in digraphs. By refining some greedy criteria, we have devised and computationally tested three different greedy heuristics, shown and discussed their performance with respect to some exact (or near-exact) solutions and each other by using two types of digraphs. To make the greedy heuristics more flexible and to improve their performance further, some randomization ideas are proposed as well.

Current and future research will focus on the randomization techniques to make these greedy heuristics more flexible and effective, and to be able to improve the obtained results for large-scale digraphs efficiently. We also plan to consider more subtle domination models in digraphs and more involved heuristic solution strategies, for example, applications and modifications of the local search. To help with exact solutions for small size problem instances in digraphs, we plan to consider devising customized deterministic

Square size	Radius, r	#Vertices, n	#Arcs, m	k	ILP		BG		DCG		TCG	
					Size	Time (s)	Size	Time (s)	Size	Time (s)	Size	Time (s)
1km x 1km	300m	614	8,546	1	61	0.11	66	0.0057	66	0.0069	67	0.0063
				2	108	0.24	118	0.0085	114	0.011	113	0.0086
				4	195	5.9	220	0.012	211	0.014	210	0.012
				8	341	170.8	387	0.017	355	0.019	358	0.015
2km x 2km	400m	2,354	56,306	1	136	0.16	154	0.044	152	0.055	148	0.045
				2	259	7.35	294	0.064	287	0.076	283	0.06
				4	485	*max*	561	0.11	538	0.14	525	0.11
				8	896	*max*	1044	0.17	963	0.19	963	0.15

Table 3: k-Dominating sets in small reachability digraphs of Road Networks.

Square size	Radius, r	#Vertices, n	#Arcs, m	k	BG		DCG		TCG	
					Size	Time (s)	Size	Time (s)	Size	Time (s)
10km x 10km	3km	38,506	52,571,274	1	88	3.95	91	3.9	86	4.36
				2	151	3.88	149	4.05	145	4.61
				4	260	4.27	259	4.4	257	4.92
				8	452	4.91	450	5.2	439	5.79
30km x 30km	5km	131,969	423,758,564	1	232	49.82	235	58.65	232	50.81
				2	398	36.54	400	31.91	393	36.55
				4	708	44.61	691	39.77	696	49.5
				8	1275	49.55	1217	47.55	1213	49.82
50km x 50km	7km	225,289	1,063,778,792	1	338	411.1	334	413.7	331	371.2
				2	570	334.5	571	417.6	560	398.9
				4	982	455.5	962	401.2	964	449
				8	1752	494.5	1695	543.1	1689	494.1

Table 4: k-Dominating sets in large reachability digraphs of Road Networks.

algorithms. Notice that some recent research (see [14]) focused on greedy heuristics to search for small weight dominating sets in vertex-weighted digraphs.

ACKNOWLEDGMENTS

Lukas Dijkstra acknowledges funding from the Maths DTP 2020, EPSRC grant EP/V520159/1.

REFERENCES

[1] N. Alon, J.H. Spencer, *The Probabilistic Method*, John Wiley & Sons Inc., New York, 1992.
[2] J. Bang-Jensen, G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer, London, 2009.
[3] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, R. Werneck, Route planning in transportation networks, In: *Algorithm Engineering: Selected Results and Surveys*, LNCS 9220, Springer, 2016, pp. 19–80.
[4] E.J. Cockayne, S.T. Hedetniemi, Towards a theory of domination in graphs, *Networks* **7**(3) (1977), 247–261.
[5] P. Corcoran, A. Gagarin, Heuristics for k-domination models of facility location problems in street networks, *Comput. Oper. Res.* **133** (2021), paper no. 105368.
[6] R.G. Downey, M.R. Fellows, *Parameterized Complexity*, Springer, 1999.
[7] S. Funke, A. Nusser, S. Storandt, Placement of Loading Stations for Electric Vehicles: No Detours Necessary!, *J. Artif. Intell. Res.* **53** (2015), 633–658.
[8] A. Gagarin, P. Corcoran, Multiple domination models for placement of electric vehicle charging stations in road networks, *Comput. Oper. Res.* **96** (2018), 69–79.
[9] A. Gagarin, A. Poghosyan, V. Zverovich, Randomized algorithms and upper bounds for multiple domination in graphs and networks, *Discrete Appl. Math.* **161** (2013), 604–611.
[10] M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman & Co., New York, 1979.
[11] Gurobi Optimization software, https://www.gurobi.com/
[12] T.W. Haynes, S.T. Hedetniemi, P.J. Slater, *Fundamentals of Domination in Graphs*, Marcel Dekker, New York, 1998.
[13] C. Lee, Domination in digraphs, *J. Korean Math. Soc.* **35**(4) (1998), 843–853.
[14] M.R. Nakkala, A. Singh, Heuristics for Minimum Weight Directed Dominating Set Problem, In: *Futuristic Trends in Networks and Computing Technologies*, FTNCT 2019, CCIS 1206, Springer, 2020, pp. 494–507.
[15] OpenStreetMap geographic information system, https://www.openstreetmap.org/
[16] L. Ouldrabah, M. Blidia, A. Bouchou, On the k-domination number of digraphs, *J. Comb. Optim.* **38** (2019), 680–688.
[17] A.K. Parekh, Analysis of a greedy heuristic for finding small dominating sets in graphs, *Inf. Process. Lett.* **39**(5) (1991), 237–240.
[18] R. Raz, S. Safra, A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP, In: *Proc. 29th Annual ACM Symposium on Theory of Computing*, STOC '97, ACM Press, 1997, pp. 475–484.
[19] F. Wang, H. Du, E. Camacho, K. Xu, W. Lee, Y. Shi, S. Shan, On positive influence dominating sets in social networks, *Theor. Comput. Sci.* **412**(3) (2011), 265–269.
[20] J. Yu, N. Wang, G. Wang, D. Yu, Connected dominating sets in wireless ad hoc and sensor networks – A comprehensive survey, *Comput. Commun.* **36**(2) (2013), 121–134.
[21] V. Zverovich, *Modern Applications of Graph Theory*, Oxford University Press, UK, 2021.

Heuristics for improving bicycle networks

Repusseau Felix[1,2], Rault Tifenn[1], and Néron Emmanuel[1]

[1]Laboratoire d'Informatique Fondamentale et Appliquée, Université de Tours, Tours, France, ✉ felix.repusseau@etu.univ-tours.fr
[2]La compagnie des Mobilité,Tours, France

1 Introduction

In this paper we are interested in the improvement of a bicycle network in order to increase the overall safety of cyclists. At a time when the development of low-GHG mobility is a key issue for all cities, it now seems necessary to deploy cycling facilities intelligently in order to increase the safety of cyclists [3, 2, 6].

The problem we address here is to use a set of GPS tracks corresponding to the routes taken by cyclists to propose improvements to the bicycle network within a given budget. The tool is designed to help local authorities plan the expansion of their bicycle network. Previous work has focused on a mathematical model whose application to real instances has proved to be limited [4].

2 Problem presentation

We have a *graph* describing the network (road network and its bicycle facilities), and for each arc *costs representing the distance and insecurity of the arc before and after possible improvements*. We assume that the cost of an improvement is proportional to the distance of the modified arc. The coefficient of proportionality depends on the type of improvement and the maximum speed of the vehicles if the infrastructure is shared between cyclists and cars. We also consider a set of *user paths*, i.e. paths taken by cyclists on the graph. These paths are obtained from real sets of GPS-traces. For each path a *user preference* between distance and insecurity is determined. This preference is recalculated according to the path taken by the user and the shortest and safest paths between his origin and his destination. Finally, to carry out these improvements we have a maximal *budget*, corresponding to the maximum length of lanes that can be improved, e.g, building bike-lane.

The aim is to propose a set of arcs that, improve (1) the user's effort, i.e. that allow the cyclists to benefit from a better path in terms of the preference established (trade-off between distance and safety) from their starting point to their destination, (2) the intrinsic properties of the graph. For these later criteria we will consider a number of indicators of the cyclability of the network, such as the coverage of the network by bicycle facilities, the number of connected components, the size of the connected components and the resilience of the network (the possibility of connecting two vertices of connected components when an arc is deleted)[7].

An Integer Linear Programming (ILP) formulation has already been proposed for this problem minimizing the overall effort of cyclists. This ILP model will allow us to evaluate the behavior of the heuristics on small instances, or on real graphs, considering a reduced number of traces [4] (see Tab.1 for details).

3 Heuristics overview

We present here a few heuristics based on works in the literature adapted to our problem.

Heuristics based on network topology. These methods seek to improve the connectivity of the bicycle network by connecting connected components as in [5]. Two versions are proposed: *Largest to Second* which aims to connect the largest connected component to the second largest one and *Largest to Closest* which aims to connect the largest connected component to the closest one. In both cases, all paths connecting the two connected components are tested, within the limits of the remaining budget. Once

	# Nodes	# Arcs	# Traces	# already upgraded arcs
Synthetic	1246	3819	1000	479
Tours	43 709	108 473	4194	10 192
Paris	53 222	126 280	52 949	24 566

Table 1: Size of synthetic and real instances

the two components are connected, the size of the merged component is recalculated and the heuristic iterates as long as the remaining budget is non-zero.

Heuristics based on cyclists' use of the network. From the user traces we can associate each arc with a frequency, i.e. the number of traces passing through this arc. The heuristic inspired by [1] seeks to make bicycle networks grow by preserving the characteristic of connectivity while taking into account this frequency. Improvable arcs are therefore arcs connected to an existing bicycle infrastructure. Among those arcs that can be upgraded, the one with the highest ridership and whose upgrading cost is less than the remaining budget is chosen. Arcs connected to this new upgraded arc become improvable. The process is repeated as long as the budget is not zero.

Heuristics based on cyclists' traces. This heuristic aims to evaluate, for each arc, the potential gain that users can derive from its improvement. For this purpose, for each user, the best path, integrating user preference, is recalculated assuming the arc is upgraded. If the user's effort improves considering this upgrade, the gain for the user is considered. The gain of an arc is the sum of gains from all users. The chosen arc for improvement is the one with the highest gain within the remaining budget limit. This process iterates as long as the budget is not zero.

4 Conclusion and Future works

The results of these heuristics will be presented at the conference. Their performances will be compared with the results obtained by the ILP formulation, which only considers the criterion of total user effort, on reduced instances (either in terms of network size or number of traces considered). We will also present results on real instances (cities of Tours and Paris, with a significant number of traces). Finally, prospects for improving the bicycle network will be discussed, both in terms of the fairness of the improvements adopted in relation to different users, and in terms of resolving discontinuities in the bicycle network: the continuity of an end-to-end network, with a certain level of safety, is often considered a key factor in users' decisions to bicycle.

References

[1] Jie Bao, Tianfu He, Sijie Ruan, Yanhua Li, and Yu Zheng. *Planning Bike Lanes based on Sharing-Bikes' Trajectories*. August 2017.

[2] Sheng Liu, Max Shen, and Xiang Ji. Urban Bike Lane Planning with Bike Trajectories: Models, Algorithms, and a Real-World Case Study. *Manufacturing & Service Operations Management*, 24, October 2021.

[3] Luis Guillermo Natera Orozco, Federico Battiston, Gerardo Iñiguez, and Michael Szell. Data-driven strategies for optimal bicycle network growth. *Royal Society Open Science*, 7:201130, December 2020.

[4] Emmanuel Néron, Tifenn Rault, and Félix Repusseau. Bicycle Network Improvements. In *ROADEF 2023 (24ème édition du congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision)*, Rennes, France, February 2023.

[5] Luis Orozco, Federico Battiston, Gerardo Iñiguez, and Michael Szell. Data-driven strategies for optimal bicycle network growth. *Royal Society Open Science*, 7:201130, 12 2020.

[6] Juan P. Ospina, Juan C. Duque, Verónica Botero-Fernández, and Alejandro Montoya. The maximal covering bicycle network design problem. *Transportation Research Part A: Policy and Practice*, 159:222–236, May 2022.

[7] Michael Szell, Sayat Mimar, Tyler Perlman, Gourab Ghoshal, and Roberta Sinatra. Growing urban bicycle networks. *Scientific Reports*, 12(1):6765, Apr 2022.

A Guided Insertion Mechanism for Solving the Dynamic Large-Scale Dial-a-Ride Problem

Chijia Liu
chijia.liu@uca.fr
Université Clermont Auvergne, CNRS, Clermont Auvergne INP, Mines Saint-Etienne, UMR 6158 LIMOS
Clermont-Ferrand, France

Quilliot Alain
quilliot.alain@uca.fr
Université Clermont Auvergne, CNRS, Clermont Auvergne INP, Mines Saint-Etienne, UMR 6158 LIMOS
Clermont-Ferrand, France

Hélène Toussaint
helene.toussaint@uca.fr
Université Clermont Auvergne, CNRS, Clermont Auvergne INP, Mines Saint-Etienne, UMR 6158 LIMOS
Clermont-Ferrand, France

Dominique Feillet
feillet@emse.fr
Mines Saint-Etienne, Univ Clermont Auvergne, Clermont Auvergne INP, CNRS, UMR 6158 LIMOS, F
Saint-Etienne, France

ABSTRACT

We study a prospective transportation system where vehicles provide dial-a-ride services to fulfill a very large scale of passenger requests (around 300,000). The system operates dynamically, with newly submitted requests needing immediate processing. A crucial aspect of this system's viability in real-time situations is therefore the implementation of an efficient routing algorithm that can deliver high-quality solutions. We address a dynamic large-scale dial-a-ride problem through a best-fit greedy insertion algorithm. Furthermore, large-scale requests are assumed to be dominated by daily commuting needs and thus should be similar and repetitive from one day to another. Therefore, there might exist recurring and similar patterns in vehicle trajectories if similar requests can be served in the same manner. We introduce a Guided Insertion Mechanism that relies on a representative reference resolution and guides the insertion of dynamic requests while maintaining high-quality solutions.

1 INTRODUCTION

In recent years, we have witnessed the emergence of novel transportation systems, offering efficient and sustainable alternatives to traditional modes of transportation. Among these, *vehicle-sharing* and *ride-sharing* systems are two predominant categories. In vehicle-sharing systems, vehicles positioned at stations are left for access by users. Related problems can be about the design of the systems, such as where to position the stations and how to relocalize vehicles [8]. Ride-sharing services allow users to share common journeys in the same vehicle. The most common problems are about the effective routing and scheduling of vehicles that take advantage of the mutualization of different services [1, 7]. Both systems promote reduced congestion and emissions while maximizing vehicular utilization. *Dail-A-Ride* (DAR) system can be regarded as a hybrid of the above two categories, where vehicles are owned by operators and provide demand-responsive transportation services to fulfill the diverse needs of various communities and enhance accessibility across urban transit systems.

In this paper, we consider a prospective transportation system where a mid-capacity vehicle fleet offers Dial-A-Ride (DAR) services to a very large volume of passengers (around 300, 000 per day), catering to the transportation needs of a vast user base. The system operates in a dynamic context, where user requests are submitted and processed on-the-fly. Therefore, to ensure the viability of the system, we need to implement a routing algorithm that is both efficient and capable of delivering high-quality solutions. For that, we address a dynamic large-scale dial-a-ride problem. Due to the large-scale aspect, it is difficult to solve the problem through exact methods(e.g., branch and bound, branch and cut, etc.) or local search approaches (e.g., Large Neighborhood Search, Build and Destroy, etc). We rely on the classic greedy insertion heuristic for its simplicity and effectiveness. Furthermore, we consider that large-scale requests should be dominated by daily commute needs which exhibit recurring characteristics. Therefore, daily requests should be globally similar and repetitive. Under this assumption, we should be able to recreate similar travel patterns in vehicle trajectories. Based on this idea, as our main contribution to this work, we propose a *Guided Insertion Mechanism (GIM)* which utilizes the travel patterns constructed from a representative reference resolution to efficiently insert dynamic requests while maintaining the high quality of the routing solution.

The remainder of this paper is organized as follows: In Section 2 we introduce some related works in the literature. Section 3 formally defines our problem and introduces some important notation. Section 4 introduces the notion of insertion of a request. In Section 5, the solution scheme is outlined. Then, in Section 6, we detail the *GIM*. Experimental results are presented and discussed in Section 7 and we conclude in Section 8.

2 RELATED WORKS

On-demand DAR transportation systems provide personalized and efficient transportation services to customers. These systems utilize digital platforms and shared mobility concepts, offering convenient, flexible, and cost-effective travel choices. As the need for efficient urban transportation increases, there is a rising prevalence of studies focusing on large-scale systems. Studies are actively studying

various system characteristics, encompassing the management of congestion issues resulting from the substantial vehicle fleet size [13], addressing recharging challenges in scenarios involving electric vehicles [2], and exploring the potential advantages of introducing ride-sharing into these systems [6]. In customer-facing systems like these, the rapid processing of user demands during dynamic scenarios is crucial. Hence, expediting the routing and scheduling process holds vital significance. The authors in [16] study a large-scale taxi system that serves at least 330, 000 requests per day. To provide instant feedback in response to dynamic requests, they introduce a filtering algorithm *dual-side taxi searching* that rapidly retrieves some interesting (nearest) vehicles to be candidates to service the given request. Similarly, in [14], the authors propose a filtering system that not only provides candidate vehicles but corresponding candidate insertion positions within the routes as well.

Within the context of DAR transportation systems, we naturally rely on the Dial-A-Ride Problem (DARP) to formulate the problem. Designing a fleet of vehicles' routes and schedules to accommodate a set of passenger requests—typically defined by a pickup location, a drop-off location, pickup and/or drop-off time windows, and a maximum ride time—is the basis of DARP. Objectives like maximizing customer service quality or reducing vehicle operating costs guide decisions [10]. DARP is NP-hard as it admits the Vehicle Routing Problem (VRP) as a special case. Very few studies seek to solve the DARP using exact optimization methods, unless for solving small-sized static problems [4, 15]. More studies tend to propose approximate methods capable of solving larger instances, such as tabu search meta-heuristics [5], Large Neighborhood Search (LNS) [9]. For example, [3] proposes a two-phase scheduling heuristic that first builds an auxiliary graph and then solves an assignment problem on this graph.

Furthermore, in a very large-scale context, daily travel requests are supposed to obey a certain repetitive pattern because they should be dominated by regular commuting needs. This makes capturing the daily mobility patterns of requests a hot topic in the literature. If sufficient historical daily instances are available, this task can be achieved through some machine learning or deep learning approaches [18, 19]. On the other hand, due to the similarity in passengers' travel requests, the trajectories of vehicles supporting these requests should also exhibit some regular travel patterns. In [12], the authors propose a graph-based analysis framework that characterizes spatial and temporal patterns of network-wide traffic flows. In [11], a trajectory clustering method is presented to discover spatial and temporal travel patterns in a traffic network. In this paper, we do not take care of extracting travel patterns from historical events and assume that a sufficiently representative reference request instance in our DAR system is available. We introduce a *Guided Insertion Mechanism (GIM)* that uses a set of simplified routes that we call *vehicle travel patterns* established by solving the related reference problem to help solve a large-scale dynamic DARP more quickly. To the best of our knowledge, we are the first to consider the correlation between vehicle travel patterns and the resolution of DARP. Additionally, we offer a formal representation of these vehicle travel patterns, providing a framework applicable in resolving DARP.

3 PROBLEM STATEMENT

In this section, we define our Large-Scale Dial-A-Ride Problem (**LSDARP**).

We consider a transit network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ contains all the intersections, and $\mathcal{A}$ contains all the arcs in the network. We only consider one depot, denoted by $n_0 \in \mathcal{N}$ in $\mathcal{G}$. The travel time of an arc $(i, j) \in \mathcal{A}$, $i, j \in \mathcal{N}^2$, is denoted by $t(i, j)$. By extension, we use $t(u, v)$ to denote the shortest travel time from any node u to any node v in the network.

A request $r \in \mathcal{R}$ is submitted at time t^r_{sub} with the following information: a pickup service requirement O^r, a drop-off service requirement D^r and the number of involved passengers q^r. The pickup service O^r includes an origin $o^r \in \mathcal{N}$ and a pickup time window $[e_{O^r}, l_{O^r}]$. And the drop-off service D^r includes a destination $d^r \in \mathcal{N}$ and a maximum ride time T^r. We note that service times are not considered, and all requests are supposed to be feasible and non-preemptive, which means that each request must be fulfilled exactly once by exactly one vehicle.

Passengers are serviced by a fleet $\mathcal{V}$ of vehicles of capacity Q. A route $\theta^v \in \Theta$ followed by a vehicle v is a list of *key points* K that aggregates services happening at the same location at the same time. Typically, a key point K contains: $n_K \in \mathcal{N}$, the location of the service; q_K, the load of v before departing from n_K; R^+_K, the list of requests scheduled to get onboard at K; R^-_K, the list of requests scheduled to get off at K; $[e^a_K, l^a_K]$, the arrival time window at n_K; and $[e^d_K, l^d_K]$, the departure time window from n_K. Let $succ(K)$ denote the successive key point of K. For any request r assigned to v, we use $K(O^r) \in \theta^v$ (resp. $K(D^r)$) to denote the key point where O^r (resp. D^r) is inserted.

When providing services, v follows the earliest arrival time e^a_K and earliest departure time e^d_K. Vehicle routes must start, end at the depot n_0, and have a load that never exceeds vehicle capacity. And for every request r assigned to the vehicle, O^r precedes D^r, and the schedule must not violate the pickup time window and the maximum ride time constraints.

We consider a lexicographic objective function. We assume that the number of vehicles is unlimited, so minimizing the fleet size is the primary objective. The total drive time of vehicles is considered the second criterion.

4 INSERTION OF A REQUEST

Given a triplet of *insertion parameters* (θ^v, K^o, K^d), we introduce the procedure $INSERTION(r, \theta^v, K^o, K^d)$ that inserts r into the route θ^v at positions K^o and K^d.

We first insert the pickup service O^r. If $n_{O^r} = n_{K^o}$, we *aggregate* O^r to K^o and $K(O^r) = K^o$; otherwise, a new key point $K(O^r)$ supporting O^r will be inserted between K^o and $succ(K^o)$. In both cases, the load $q_{K(O^r)}$, in inbound request list $R^+_{K(O^r)}$, and the pickup and drop-off time windows on $K(O^r)$ should be updated while considering the constraints about the vehicle and requests mentioned previously. Same rules applied for the insertion of the drop-off service D^r at K^d. We note that when updating the arrival time windows of $K(D^r)$, we need to take into account the maximum ride time T^r of r.

Once O^r and D^r are inserted, we increase the load of key points between $K(O^r)$ and $K(D^r)$ by q^r while ensuring that the new loads never exceed the vehicle capacity Q. We should also update the time windows of the key points along θ^v and checking that these time windows are never empty. For that, we implement a classic *constraint propagation procedure* ([17]) considering the above time constraints. The procedure has a complexity of $O(|\theta^v|^2)$, where $|\theta^v|$ is the number of key points in θ^v.

5 ALGORITHM FRAMEWORK

We define a set of decision epochs $\mathcal{E} = \{E_0, E_1, \ldots, E_i, \ldots, E_{|E|-1}\}$. Each decision epoch lasts I_e (for example, $I_e = 10$ min) time units. The starting time of E_i is $t_{E_i} = i \times I_e$. For each decision epoch $E_i \in \mathcal{E}$,

(1) during the time slot $[t_{E_i}, t_{E_i} + \tau]$, where τ defines the maximum decision duration, the system makes the routing decisions for requests submitted during the previous epoch E_{i-1}, denoted by $\mathcal{R}_{E_{i-1}}$;
(2) at time $t = t_{E_i} + \tau$, the system updates the vehicles' schedules, and informs unserved passengers whose requests have already been inserted about the updated information about their pickup (the vehicle's passage time);
(3) following the update, vehicles start implementing the new routes until reaching $t = t_{E_{i+1}} + \tau$.

Regarding the very large size of the problem and the need to make decisions on-the-fly, we address the problem to be solved for each decision epoch E based on the classic best-fit insertion heuristic. During epoch E, requests $\mathcal{R}_E$ are inserted one by one following a specific order, and for each request r, we try to insert it according to the best-fitted insertion parameters that minimize the insertion cost measured by the detour to service r. Depending on the implementation, searching for insertion parameters among the whole search space Θ would generally require a computational effort of $O(|\theta^v|^2)$. In addition, testing the insertion feasibility and the *INSERTION* process are also computationally expensive, as mentioned in Section 4. Due to the large-scale effect, the search space would contain thousands of vehicles along with hundreds of thousands of insertion positions to explore, which can be too large to fit the dynamic need. For that, we introduce a *guided insertion mechanism* upstream of the best-fit insertion that rapidly and wisely selects well-fitted insertion parameters and tries several valuable insertions.

Then, given a request $r \in \mathcal{R}$ and the current route collection $\Theta = \{\theta^v, v \in \mathcal{V}\}$:

- We first invoke the *Guided Insertion Mechanism (GIM)* which uses knowledge learned from representative historical instances and solutions to guide the insertion of r. If a feasible insertion is found, we keep it.
- If *GIM* fails, we invoke a best-fit insertion heuristic over the entire search space Θ, and try to insert r into the best-fitted vehicle route at the insertion positions (i.e., key points) that minimize the insertion cost.
- Finally, if both of the above steps fail, we activate a new vehicle v to serve r and add θ^v to the current route set Θ.

6 THE GUIDED INSERTION MECHANISM

In this section, we introduce the *Guided Insertion Mechanism (GIM)*. Let us use **LSDARP**($\mathcal{R}$) to denote the problem with an input instance $\mathcal{R}$. Consider two *similar* request sets $\mathcal{R}_1$ and $\mathcal{R}_2$ in a way that for most of the requests in $\mathcal{R}_1$, we can find a *similar* request in $\mathcal{R}_2$. Then we believe that the optimal routing solution Θ_1 to the problem **LSDARP**($\mathcal{R}_1$) should be similar to Θ_2, the optimal solution to the problem **LSDARP**($\mathcal{R}_2$). Because if $r_1 \in \mathcal{R}_1$ and $r_2 \in \mathcal{R}_2$ are similar, they should be able to be inserted in a similar manner.

GIM is conceived based on the above idea. Thanks to the large-scale aspect and the fact that requests should be dominated by recurring daily commute requests, we assume that requests to be processed in the DAR systems are similar from one day to another. Therefore, the travel patterns of vehicles should also be similar from one day to another. Assuming that we have a representative reference set $\bar{\mathcal{R}}$ which captures the basic distribution (origin and destination and pickup times) of requests, then the *static* (i.e., *off-line*) optimal solution $\bar{\Theta}$ to the problem **LSDARP**($\bar{\mathcal{R}}$) should be able to guide any *dynamic* (i.e., *on-line*) resolution of any real problem **LSDARP**($\mathcal{R}$), where $\mathcal{R}$ is the set of real dynamic requests to be processed in the service period.

Extracting the daily mobility patterns of requests in an underlined system and solving the associated static problem to optimal are two independent problems. As mentioned previously, in this paper, we do not take care of either of the above-mentioned problems and assume that a representative reference set $\bar{\mathcal{R}}$ generated based on the daily request distribution is in our possession, along with its off-line (near) optimal solution $\bar{\Theta}$. We are interested in how the references ($\bar{\mathcal{R}}$, $\bar{\Theta}$) can be informative and guide the insertion when solving a similar real dynamic problem **LSDARP**($\mathcal{R}$).

6.1 Preprocessing: Obtain Vehicle Travel Patterns

For each reference route $\bar{\theta} \in \bar{\Theta}$, we compute a specific *travel pattern* $\gamma(\bar{\theta}) \in \Gamma$ during the preprocessing phase. A travel pattern $\gamma(\bar{\theta})$ is a simplified route defined as a list of *pattern points*, where each *pattern point* $\bar{P}$ represents a *cluster* of key points in $\bar{\theta}$. The notion of *key point cluster* is defined as follows:

Definition 6.1 (key point cluster). Given a route $\bar{\theta} = \{\bar{K}_0, \ldots, \bar{K}_i, \ldots, \bar{K}_{M-1}\}$ with M key points, $\{\bar{K}_i, \bar{K}_{i+1}, \ldots, \bar{K}_{i+m}\}$ is *a key point cluster* if and only if:

$$t(\bar{K}_{i+j} - \bar{K}_{i+j-1}) \leqslant \delta^K, \text{ for } 1 \leqslant i \leqslant m,$$

$$t(\bar{K}_{i-1}, \bar{K}_i) > \delta^K, \text{ for } i > 0,$$

and

$$t(\bar{K}_{i+m}, \bar{K}_{i+m+1}) > \delta^K, \text{ for } i < M - 1 - m,$$

where δ^K is the clustering threshold indicating the maximum travel time from a key point to its successor that are in the same cluster.

For example, in Figure 1 where three partial routes are shown, we see that $\bar{K}_2$, $\bar{K}_3$ and $\bar{K}_4$ represent a key point cluster, so in the travel pattern $\gamma(\bar{\theta})$, they are represented by a pattern point $\bar{P}_2$.

The travel patterns are used in the guided insertion process. In *GIM*, each pattern guides the construction of at most one real route $\theta \in \Theta$. Specifically, if a real key point $K \in \theta$ is created via *GIM* under the guidance of a pattern point $\bar{P} \in \gamma$, then we call K a *child*

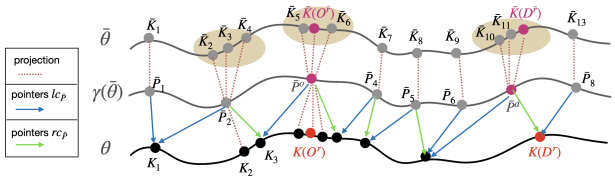


Figure 1: Illustration of the relationships between the reference route, the travel pattern, and the real route

of $\bar{P}$. Any point $\bar{P} \in \gamma$ may project to several *children* in θ. For each point $\bar{P} \in \gamma$, we define two pointers $lc_{\bar{P}}$ and $rc_{\bar{P}}$, where $lc_{\bar{P}}$ points at the preceding key point of the left-most child of $\bar{P}$, and $rc_{\bar{P}}$ points at the right-most child of $\bar{P}$. For example, in Figure 1, $lc_{\bar{P}_2}$ points at K_1, and $rc_{\bar{P}_2}$ points at K_3. For any $\bar{P}$, both $lc_{\bar{P}}$ and $rc_{\bar{P}}$ are initialized as null pointers during the preprocessing process.

6.2 Guided Insertion Process

Given a real request r, *GIM* operates according to the following steps.

6.2.1 Step 1: Retrieve similar reference requests. We first identify from $\bar{\mathcal{R}}$ all the reference requests $\bar{r}$ that are *similar* to r. The *similarity* between requests is defined as follows:

Definition 6.2 (similarity between requests). Two requests r_1 and r_2 are similar if and only if $t(o^{r_1}, o^{r_2}) \leqslant \delta^s$, $t(d^{r_1}, d^{r_2}) \leqslant \delta^s$, and $|e_{O^{r_1}} - e_{O^{r_2}}| \leqslant \delta^t$, where δ^s is a threshold indicating the maximum travel time between two locations, and the threshold δ^t indicates the maximum difference in the earliest pickup time between r_1 and r_2.

If r_1 and r_2 satisfy the above conditions, we use $|e_{O^{r_1}} - e_{O^{r_2}}|$ to define their similarity measure. Let $\bar{\mathcal{R}}^r$ denote the set of retrieved similar reference requests. The set $\bar{\mathcal{R}}^r$ is sorted according to the descending order of their value of similarity measure with r.

6.2.2 Step 2: Construct guide object set. Next, we construct a set of *guide objects*, denoted by GO^r. A *guide object* is a triplet $(\gamma, \bar{P}^o, \bar{P}^d)$ used to guide the guided insertion of r, where $\bar{P}^o$ and $\bar{P}^d$ are two pattern points in the travel pattern γ. For example, as illustrated in Figure 1, a reference request $\bar{r}$ inserted in $\bar{\theta}$ at $\bar{K}(O^r)$ and $\bar{K}(D^r)$ corresponds to the *guide object* $(\gamma(\bar{\theta}), \bar{P}^o, \bar{P}^d)$. Then, GO^r is constructed by sequentially capturing and organizing the corresponding *guide objects* of all the reference requests $\bar{r}$ in alignment with the order specified in $\bar{\mathcal{R}}^r$. We note that it is possible that several reference requests may correspond to the same *guide object*. In GO^r, we only keep one occurrence of the same *guide objects*.

6.2.3 Step 3: Insert r according to the guide object. As mentioned before, the concept of guided insertion is that similar requests should be able to be inserted in the same manner. Given a target request r and the set GO^r, we try to insert r under the guidance of elements in GO^r.

Given $(\gamma, \bar{P}^o, \bar{P}^d) \in GO^r$, we use $\bar{P}^o$ to guide the insertion of O^r, and $\bar{P}^d$ to guide the insertion of D^r.

If γ has not been related to any real route in Θ, then we activate a new vehicle v to service r. Its route θ^v is initialized as two key points K_1, the initial depot, and K_2, the final depot. Next, two key points $K(O^r)$ and $K(D^r)$ supporting the pickup and drop-off services are inserted between K_1 and K_2. Then we relate θ^v to γ by correctly setting the pointers $lc_{\bar{P}}$ and $rc_{\bar{P}}$ for all $\bar{P} \in \gamma$ (see Figure 2).

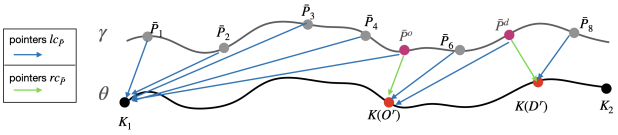


Figure 2: Illustration of the creation of the new route via GIM

If γ is already related to a real route $\theta \in \Theta$, then we utilize $\bar{P}^o$ to construct a list of candidate insertion positions for the pickup service. Specifically, in case $rc_{\bar{P}^o}$ is a *null* pointer (which means that $\bar{P}$ has no related child yet), the candidate list only contains the key point pointed by $lc_{\bar{P}^o}$; otherwise, the list contains all the key points between the two key points pointed by $lc_{\bar{P}^o}$ and $rc_{\bar{P}^o}$. We obtain a candidate list for the drop-off services in the same way. Then, we proceed with a *best-fit* scheme while trying the insertion feasibility of r at the selected candidate key points for the pickup and drop-off services. This means that if at least one feasible insertion is found, we keep the candidates that minimize the insertion cost.

Finally, every time r is inserted in θ at $K(O^r)$ and $K(D^r)$ via *GIM* with the guide object $(\gamma, \bar{P}^o, \bar{P}^d)$, we need to update the pointers $lc_{\bar{P}}$ and $rc_{\bar{P}}$ of pattern points $\bar{P}$ along γ.

We note that the goal of *GIM* when inserting r is to replicate the insertion mode of the most similar reference request. We know that the further forward positioned the guide object GO^r, the more similar the corresponding reference request is to r. Therefore, we implement a *first-fit* scheme to explore GO^r. If r can be inserted under the guidance of $(\gamma, \bar{P}^o, \bar{P}^d)$, we proceed with the insertion, stop the exploration of GO^r, and continue to insert the next request. Otherwise, we explore the next guide object in GO^r.

7 NUMERICAL EXPERIMENTS

We programmed the algorithms in C++ language and solved the problem on a 512 GB RAM machine with an AMD EPYC 7452 32-Core Processor.

7.1 Input Data

We take the transit network of the city of Clermont-Ferrand, France, and its peri-areas. The underlined area contains 13, 839 nodes and 31, 357 arcs. Among all nodes, 1, 469 are selected to be valid pickup and drop-off locations.

The system we study in this paper is still prospective, so there are no available real-life request instances. The instances tested in this paper are self-generated and simulate the intended use cases of the system: providing services to all kinds of travel requests, which are in addition dominated by daily commute demands.

The service period lasts $T = 24$ hours, from 00 : 00 to 24 : 00. We divide the entire period into five time slots: *MS* (Morning Slack, 00:00 ∼ 06:00), *MP* (Morning Peak, 06:00 ∼ 10:00), *NH* (Normal Hours, 10:00 ∼ 15:00), *EP* (Evening Peak, 15:00 ∼ 19:00) and *ES* (Evening Slack, 19:00 ∼ 24:00). We assume requests to be processed

in one day generally obey the following basic distribution: During *MP*, arond 50% are *typical* that travel from a residential location to a working location. Reversely, *EP* requests follow a symmetric pattern, with half of the requests being *typical* that move from a working position to a residential position. For the requests of *MS*, *NH* and *ES*, their origin and destination are randomly distributed over the network.

To guarantee the representative property of the reference request set $\bar{\mathcal{R}}$ used in *GIM*, we randomly construct it according to the above-introduced basic distribution.

Real requests should globally obey the basic distribution and be "similar" from one day to another while exhibiting some random variations. To simulate such a phenomenon, real request instances are decomposed into two parts: the "random" part and the "similar" part. Requests in the "random" part are randomly generated using the above-defined basic distribution. The "similar" part simulates the stable and recurring pattern of daily requests. We rely on $\bar{\mathcal{R}}$ to generate the corresponding requests. Typically, when generating a request r in this part, we randomly select a reference $\bar{r}$. Then o^r (resp. d^r) is randomly selected among the nodes that are reachable within 3 arcs from $o^{\bar{r}}$ (resp. $d^{\bar{r}}$). And the earliest pickup time e_{O^r} is a random value selected between $e_{O^{\bar{r}}} - 7.5$ minutes and $e_{O^{\bar{r}}} + 7.5$ minutes. In terms of the daily recurring pattern, we consider two scenarios: (*high*): "similar" part accounts for 90% of the requests; and (*moderate*): "similar" part accounts for 50% of the requests.

Five instances of 300, 000 requests are generated and final results are averaged. The time length of the pickup window is set at 15 minutes for all requests. The maximum ride time T^r is twice the shortest travel time from o^r to d^r. The load for each request is 1. And as real requests are supposed to be dynamic, we randomly set the submission time t^r_{sub} of r between 0 and e_{O^r}. Requests are submitted on average one hour before e_{O^r}. And in line with the no-rejection assumption, values of t^r_{sub} also satisfy that when processing r at epoch E, we can always activate a vehicle v currently parking at the depot to serve r.

7.2 Analyses of the Effectiveness of GIM

In this work, the static reference problem **LSDARP**($\bar{\mathcal{R}}$) is solved by a best-fit insertion heuristic. The resulting solution $\bar{\Theta}$ contains routes of 1, 621 vehicles. It is worth noting that there are other approximation algorithms (Large Neighborhood Search, Adaptive Large Neighborhood Search, etc.) capable of offering more optimal solutions, but at the expense of significantly increasing the processing time.

The length of each decision epoch e lasts $I_e = 10$ minutes, and for the reason of comparing the efficiency of different approaches, the maximum decision duration τ is not explicitly fixed, and we count the CPU time spent required for each decision epoch. The *GIM* parameters δ^K (key point cluster threshold), δ^s and δ^t (request similarity thresholds) are fixed at 2 minutes, 2 minutes, and 15 minutes, respectively.

In order to test the performance of the *GIM* , we consider three sets of baseline algorithms: **BF** (Best-Fit insertion heuristic), **PFS** (Partial Filtering System), and **FS** (Filtering System). **PFS** and **FS** are two approaches proposed in [14], in which a best-fit insertion heuristic is implemented over the candidate vehicles and insertion positions selected by a filtering system. Specifically, given a request r, in **PFS**, the search space contains all the filtered candidate vehicles and insertion positions; while in **FS**, the search space is further reduced to a subset of selected candidate vehicles along with their candidate insertion positions. The **FS** implemented in this study involves keeping the top 10% best vehicles from the candidate pool. The number of selected vehicles is also bounded between 40 and 140. Then, we can integrate the *GIM* on the upstream side of these three baseline algorithms. Accordingly, we propose three methods: **GIM-BF**, **GIM-PFS** and **GIM-FS**.

Table 1 shows the results of the final fleet size, the total drive time of vehicles, and passengers' average in-vehicle time. The variations represented in percentage are calculated based on the baseline approach **BF**. First, in terms of fleet size, we see that in both scenarios, all the approaches integrated with *GIM* outperform their corresponding baseline approaches. In addition, **GIM-BF** and **GIM-PFS** further reduce the fleet size established by **BF** by almost 6%, because they are guided by statically obtained travel patterns. Furthermore, thanks to the reduction in the number of used vehicles and the fact that *GIM* helps better organize the routes by following the pre-defined travel patterns, the total drive times of vehicles are also improved. Meanwhile, passenger travel comfort remains the same because better organized routes decrease detours and increase the ride-sharing. All evidence indicates that, by learning from the well-resolved reference solution, *GIM* has the potential to promote vehicle utilization and alleviate emissions. Finally, we also notice that *GIM* is not sensible to similarity scenarios *high* and *moderate*. A possible reason is that, due to the large-scale aspect, we have a great chance of finding a similar reference request, even though it was not deliberately generated as such.

Let us now focus on the scenario *high* to analyze the CPU times spent to process the requests during each decision epoch (see Figure3). First of all, we observe two peaks in the processing time for almost all approaches, corresponding to the high ratio of request submissions during the two peak periods *MP* and *EP*. During most of the epochs corresponding to the time slots *MS, MP, NH* and *ES*, all approaches using *GIM* outperforms their baseline methods. This advantage is especially pronounced during peak hours. During the first few epochs (corresponding to *MS*), methods with *GIM* tend to be less efficient. This is because *GIM* activates numerous vehicles at the beginning of the process, resulting in a larger search space compared to other baseline approaches.

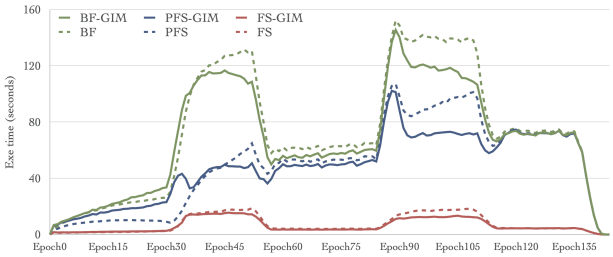


Figure 3: The execution time during each decision epoch

Generally speaking, compared to their baseline approaches, methods integrated with *GIM* showcase the advantages of reducing the

Table 1: Results with different approaches under different similarity scenarios with 300k requests

scenario	approach	fleet size	total drive time (h)	average in-vehicle time (min)
high	BF	2,183	25,717.5	16.9
	PFS	2,183 (-0.0%)	-0.0%	16.9
	FS	2,583 (+18.3%)	+22.7%	16.7
	GIM-BF	2,061 (-5.6%)	-4.9%	16.9
	GIM-PFS	2,053 (-6.0%)	-4.9%	16.9
	GIM-FS	2,326 (+6.6%)	+15.8%	16.7
moderate	BF	2,191	25,641.0	17.0
	PFS	2,176 (-0.7%)	+0.1%	17.0
	FS	2,548 (+16.3%)	+23.0%	16.8
	GIM-BF	2,063 (-5.8%)	-4.7%	17.0
	GIM-PFS	2,066 (-5.7%)	-4.7%	17.0
	GIM-FS	2,347 (+7.1%)	+16.8%	16.8

execution times during peak hours while providing better routing solutions in terms of the fleet size, total drive time and passenger comfort. This highlights the potential of the utilization of *GIM* in solving the dynamic **LSDARP**.

8 CONCLUSION

We introduce a *GIM* upstream of the classic best-fit insertion heuristic. The experiment results show that by learning and imitating a reference static solution, *GIM* stands out in dynamic scenarios by encouraging the fleet to follow optimal vehicle travel patterns. Moreover, its integration with state-of-the-art accelerating algorithms such as the *filtering system* substantially reduces processing time without compromising solution quality. We believe that, with a more representative reference problem and a more optimal reference solution, GIM should emerge as a highly effective algorithm, well-suited for addressing dynamic online problems. *GIM* is currently in its early stages as an emerging technology. Our upcoming focus aims to enhance its performance by maximizing the proportion of successful guided insertions within each epoch.

ACKNOWLEDGMENTS

This work was supported by the International Research Center "Innovation Transportation and Production Systems" of the I-SITE CAP 20-25.

REFERENCES

[1] Niels Agatz, Alan Erera, Martin Savelsbergh, and Xing Wang. 2012. Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research* 223, 2 (Dec. 2012), 295–303. https://doi.org/10.1016/j.ejor.2012.05.028

[2] Claudia Bongiovanni, Mor Kaspi, and Nikolas Geroliminis. 2019. The electric autonomous dial-a-ride problem. *Transportation Research Part B: Methodological* 122 (April 2019), 436–456. https://doi.org/10.1016/j.trb.2019.03.004

[3] Roberto Wolfler Calvo and Alberto Colorni. 2007. An effective and fast heuristic for the Dial-a-Ride problem. *4OR* 5, 1 (April 2007), 61–73. https://doi.org/10.1007/s10288-006-0018-0

[4] Jean-François Cordeau. 2006. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research* 54, 3 (2006), 573–586. Publisher: INFORMS.

[5] Jean-François Cordeau and Gilbert Laporte. 2003. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological* 37, 6 (2003), 579–594. https://doi.org/10.1016/S0191-2615(02)00045-0

[6] Daniel J. Fagnant and Kara M. Kockelman. 2018. Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in Austin, Texas. *Transportation* 45, 1 (Jan. 2018), 143–158. https://doi.org/10.1007/s11116-016-9729-z

[7] Masabumi Furuhata, Maged Dessouky, Fernando Ordóñez, Marc-Etienne Brunet, Xiaoqing Wang, and Sven Koenig. 2013. Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological* 57 (Nov. 2013), 28–46. https://doi.org/10.1016/j.trb.2013.08.012

[8] Damianos Gavalas, Charalampos Konstantopoulos, and Grammati E. Pantziou. 2015. Design and Management of Vehicle Sharing Systems: A Survey of Algorithmic Approaches. *CoRR* abs/1510.01158 (2015). arXiv:1510.01158

[9] Timo Gschwind and Michael Drexl. 2019. Adaptive Large Neighborhood Search with a Constant-Time Feasibility Test for the Dial-a-Ride Problem. *Transportation Science* 53, 2 (2019), 480–491. https://doi.org/10.1287/trsc.2018.0837

[10] Sin C. Ho, W.Y. Szeto, Yong-Hong Kuo, Janny M.Y. Leung, Matthew Petering, and Terence W.H. Tou. 2018. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological* 111 (May 2018), 395–421. https://doi.org/10.1016/j.trb.2018.02.001

[11] Jiwon Kim and Hani S. Mahmassani. 2015. Spatial and Temporal Characterization of Travel Patterns in a Traffic Network Using Vehicle Trajectories. *Transportation Research Procedia* 9 (2015), 164–184. https://doi.org/10.1016/j.trpro.2015.07.010

[12] Jiwon Kim, Kai Zheng, Jonathan Corcoran, Sanghyung Ahn, and Marty Papamanolis. 2022. Trajectory Flow Map: Graph-based Approach to Analysing Temporal Evolution of Aggregated Traffic Flows in Large-scale Urban Networks. arXiv:2212.02927 [cs].

[13] Xiao Liang, Gonçalo Homem de Almeida Correia, Kun An, and Bart van Arem. 2020. Automated taxis' dial-a-ride problem with ride-sharing considering congestion-based dynamic travel times. *Transportation Research Part C: Emerging Technologies* 112 (March 2020), 260–281. https://doi.org/10.1016/j.trc.2020.01.024

[14] Chijia Liu, Alain Quilliot, Hélène Toussaint, and Dominique Feillet. 2023. A Filtering System for the Large-Scale Dial-A-Ride Problem With Shared Autonomous Vehicles. In *Proceedings of the 12th International Symposium on Information and Communication Technology (SOICT '23)*. Association for Computing Machinery, New York, NY, USA, 679–686. https://doi.org/10.1145/3628797.3628871

[15] Yannik Rist and Michael Forbes. 2021. A New Formulation for the Dial-a-Ride Problem. *Transportation Science* 55 (08 2021). https://doi.org/10.1287/trsc.2021.1044

[16] Shuo Ma, Yu Zheng, and O. Wolfson. 2013. T-share: A large-scale dynamic taxi ridesharing service. In *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE, Brisbane, QLD, 410–421. https://doi.org/10.1109/ICDE.2013.6544843

[17] Jiafu Tang, Yuan Kong, Henry Lau, and Andrew W.H. Ip. 2010. A note on "Efficient feasibility testing for dial-a-ride problems". *Operations Research Letters* 38, 5 (2010), 405–407. https://doi.org/10.1016/j.orl.2010.05.002

[18] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. 2019. Origin-Destination Matrix Prediction via Graph Convolution: a New Perspective of Passenger Demand Modeling. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Anchorage AK USA, 1227–1235. https://doi.org/10.1145/3292500.3330877

[19] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. *arXiv:1610.00081 [cs]* (Jan. 2017). arXiv: 1610.00081.

INOC - 2024

Session Session 3A: Combinatorial Optimization

Tuesday 12 March 2024, 11:00-12:30

Q011

Ensemble pruning via an integer programming approach with diversity constraints*

†

Marcelo A. Mendes Bastos
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
marcelo.bastos@dcc.ufmg.br

Humberto Brandão
Universidade Federal de Alfenas
Alfenas, MG, Brazil
humberto.brandao@gmail.com

Cristiano Arbex Valle
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
arbex@dcc.ufmg.br

ABSTRACT

Ensemble learning combines multiple classifiers in the hope of obtaining better predictive performance. Empirical studies have shown that ensemble pruning, that is, choosing an appropriate subset of the available classifiers, can lead to comparable or better predictions than using all classifiers. In this paper, we consider a binary classification problem and propose an integer programming (IP) approach for selecting optimal classifier subsets. We propose a flexible objective function to adapt to different datasets as well as constraints to ensure minimum diversity levels in the ensemble. We are able to quickly obtain good solutions for datasets with up to 60,000 data points. Our approach yields competitive results when compared to some of the most used pruning methods in literature.

1 INTRODUCTION

Ensemble learning is a popular technique in the domain of machine learning. An ensemble is defined as the aggregation of multiple classifications into a single final decision. It is generally accepted in literature that the precision of an ensemble tends to improve when compared to the behaviour of individual classifiers [27].

Well-known approaches for efficiently generating ensembles include Bagging (bootstrap aggregating) [3] and Boosting [13], in which all classifiers are considered in the aggregation. There are, however, theoretical and empirical studies which have shown that pruning an ensemble by selecting a subset of the classifiers can lead to comparable or better predictions [19, 27].

In this work we tackle the ensemble pruning problem by introducing an integer programming (IP) approach for choosing an optimal subset of binary classifiers. Our formulation optimises a weighted function of the patterns in the binary confusion matrix. This flexible approach allows us to customise the objective function according to the properties of the underlying dataset. As our objective is based on performance we also introduce linear constraints that ensure minimum diversity levels in the ensemble.

Despite the existence of consolidated techniques for ensemble pruning, we believe that our approach contributes to the current knowledge in the field due to the flexibility of the IP paradigm, adaptable to particularities of different problems. One of its advantages is being able to combine performance and diversity criteria. Furthermore, our method is exact, while most algorithms in literature are suboptimal.

In this paper we show that with current solver technology we can find good solutions to relatively large problems in reasonable computational times. We compare our formulation to a full ensemble and six other well-known methods in literature. We report competitive results for publicly available datasets ranging from 195 to 60,000 data points.

The remainder of this paper is organised as follows. In Section 2 we give a brief overview of existing methods in ensemble learning. In Section 3 we present our optimisation approach and in Section 4 we amend it to enforce minimum diversity levels. Our computational experiments are shown in Section 5 and in Section 6 we present our concluding remarks.

2 LITERATURE REVIEW

The first step in an ensemble process is to generate a set of distinct classifiers that is precise and diverse. Highly correlated classifiers may hinder the potential benefit of using an ensemble. Several techniques for ensuring diversity in classifiers have been proposed [8, 10], such as randomisation, distinct tuning of hyperparameters and different classifiers. Other diversification techniques include training classifiers with different distributions of the training set and with distinct subsets of features.

The next step is selecting an appropriate classifier subset. This selection can be dynamic [8], where different subsets are chosen for different data points, or static, where a single subset is chosen. Static selection policies are based on ranking, clusters and optimisation.

Ranking methods sort classifiers according to a fitness function. In general they greedily increase the subset size. In Kappa pruning [20], every pair of classifiers is sorted according to a statistical measure of agreement. Reordering techniques [22] are used to build sub-ensembles of increasing size. In [26] classifiers are ranked according to a significance index.

Cluster methods first apply a clustering algorithm to separate classifiers according to some similarity measure and then prune each cluster separately to increase general diversity. Known clustering algorithms include k-means [17], where similarity is based on Euclidean distance, and hierarchical agglomerative clustering [15], which employs probabilities.

Several optimisation methods for ensemble pruning have also been proposed, with most offering approximate solutions. The most popular method is hill climbing, which has been applied with several different fitness functions. Some are based on performance [11] (e.g.

*Produces the permission block, and copyright information
†

accuracy), others on diversity [20, 24]. Three examples of diversity-based fitness functions are Complementariness [21], Concurrency [1] and Uncertainty Weighted Accuracy [25]. In [23], reinforcement learning was employed for a greedy method based on diversity. In [27] a semi-definite programming approach was proposed which considers trade-offs between accuracy and diversity.

The last step in the procedure is combining classifiers into a single prediction, which is usually done through majority voting. For further details we refer the reader to [16].

3 FORMULATION

Consider a binary classification problem where data points belong to classes 1 (positive) or 0 (negative). Let $\mathcal{K} = \{1, \ldots, K\}$ be the set of classifiers. Let $\mathcal{N}_0 = \{1, \ldots, N_0\}$ and $\mathcal{N}_1 = \{1, \ldots, N_1\}$ be the sets of negative and positive data points respectively, with $N = N_0 + N_1$ being the total number of data points. Consider a $N_1 \times K$ matrix B where $\beta_{ik} = 1$ if classifier $k \in \mathcal{K}$ correctly classified data point $i \in \mathcal{N}_1$ as positive, $\beta_{ik} = 0$ if it mistakenly classified i as negative. Accordingly consider a $N_0 \times K$ matrix A where $\alpha_{jk} = 0$ if classifier $k \in \mathcal{K}$ correctly classified data point $j \in \mathcal{N}_0$ as negative, $\alpha_{jk} = 1$ if j was mistakenly classified as positive.

Suppose $\mathcal{S} \subseteq \mathcal{K}$ is a set of S classifiers selected to compose a given pruned ensemble. For any data point $i \in \mathcal{N}_1$, $\sum_{s\in\mathcal{S}} \beta_{is}$ is the number of correct positive classifications within $\mathcal{S}$. Accordingly, for any data point $j \in \mathcal{N}_0$, $\sum_{s\in\mathcal{S}} \alpha_{js}$ represents the number of (wrong) positive classifications within $\mathcal{S}$.

We define a threshold $0 \leq L \leq S$ such that for a given data point $i \in \mathcal{N}_1$, $\sum_{s\in\mathcal{S}} \beta_{is} > L$ implies that the ensemble classifies i as positive. If $\sum_{s\in\mathcal{S}} \beta_{is} \leq L$, then i is classified by the ensemble as negative. Similarly for $j \in \mathcal{N}_0$, $\sum_{s\in\mathcal{S}} \alpha_{js} > L$ implies a positive ensemble classification and $\sum_{s\in\mathcal{S}} \alpha_{js} \leq L$ implies a negative ensemble classification. For instance, if $S = 10$ and $L = 5$, then the ensemble classifies a data point as positive if at least 6 individual classifications are positive. If 5 or less are positive, then the ensemble classifies that data point as negative.

In our formulation we let the optimisation define both $\mathcal{S}$ and L. Hence we include L as a general integer variable representing the classification threshold and binary variables $x_k = 1$ if classifier $k \in \mathcal{K}$ is chosen to compose the ensemble ($x_k = 0$ otherwise).

Table 1: Binary classification confusion matrix

		Predicted 1	Predicted 0
Actual	1	T^+	F^-
	0	F^+	T^-

Consider the binary confusion matrix shown in Table 1, where T^+, F^-, T^- and F^+ are the total number of classifications of each possible pattern. For each patterns we assign weights $W_T^+, W_T^-, W_F^+, W_F^- \in \mathbb{R}$, and the objective function is defined by the weighted sum $W_T^+T^+ + W_F^-F^- + W_T^-T^- + W_F^+F^+$.

For modelling this function we define binary variables t_i^+, f_i^- if the ensemble classification of $i \in \mathcal{N}_1$ is respectively a true positive or false negative. Similarly we define binary variables t_j^-, f_j^+ if the ensemble classification of $j \in \mathcal{N}_0$ is a true negative or false positive.

The IP formulation is given by:

$$\max \sum_{i=1}^{N_1} (W_T^+ t_i^+ + W_F^- f_i^-) + \sum_{j=1}^{N_0} (W_T^- t_j^- + W_F^+ f_j^+) \tag{1}$$

subject to

$$(L+1) - \sum_{k=1}^{K} x_k\, \beta_{ik} \leq (K+1)(1 - t_i^+), \qquad \forall i \in \mathcal{N}_1 \tag{2}$$

$$\sum_{k=1}^{K} x_k\, \beta_{ik} - L \leq (K+1) t_i^+, \qquad \forall i \in \mathcal{N}_1 \tag{3}$$

$$t_i^+ + f_i^- = 1, \qquad \forall i \in \mathcal{N}_1 \tag{4}$$

$$\sum_{k=1}^{K} x_k\, \alpha_{jk} - L \leq K(1 - t_j^-), \qquad \forall j \in \mathcal{N}_0 \tag{5}$$

$$(L+1) - \sum_{k=1}^{K} x_k\, \alpha_{jk} \leq K t_j^-, \qquad \forall j \in \mathcal{N}_0 \tag{6}$$

$$f_j^+ + t_j^- = 1, \qquad \forall j \in \mathcal{N}_0 \tag{7}$$

$$x_k \in \mathbb{B} \qquad \forall k \in \mathcal{K} \tag{8}$$

$$t_i^+, f_i^- \in \mathbb{B} \qquad \forall i \in \mathcal{N}_1 \tag{9}$$

$$t_j^-, f_j^+ \in \mathbb{B} \qquad \forall j \in \mathcal{N}_0 \tag{10}$$

$$0 \leq L \leq K, \tag{11}$$

$$L \in \mathbb{Z} \tag{12}$$

Constraints (2) ensure that a positive data point $i \in \mathcal{N}_1$ has $t_i^+ = 1$ if the number of individual positive classifications exceeds L. Conversely, constraints (3) ensure that $t_i^+ = 0$ if the number of individual positive classifications is no more than L. Constraints (4) ensure that either $t_i^+ = 1$ or $f_i^- = 1$. Constraints (5) guarantee that a negative data point $j \in \mathcal{N}_0$ has $t_j^- = 0$ if the number of positive classifications exceeds L. Otherwise, constraints (6) make sure that $t_j^- = 1$. Constraints (7) ensure that either $f_j^+ = 1$ or $t_j^- = 1$. Constraints (8)-(12) define variables bounds.

3.1 Objective function

For some classification problems, it may be desirable to optimise some patterns instead of others. For instance, in an investment decision, investing in the wrong project may cause bankruptcy while not investing in a promising project may be seen as a regretful but acceptable lost opportunity. In this case prioritising the minimisation of F^+ is desirable. The weights in Equation (1) provide flexibility for defining optimisation criteria depending on the characteristics of the dataset at hand (such as being highly imbalanced). A few examples are outlined below.

Accuracy is defined as $\frac{T^+ + T^-}{N}$. As N is constant we can maximise accuracy by defining weights $W_T^+ = W_T^- = 1$ and $W_F^+ = W_F^- = 0$. Notice that if we choose this objective then constraints (3) and (6) are redundant as maximising positive weights W_T^+ and W_T^- ensure that $t_i^+ = 1$ and $t_i^- = 1$ if allowed by constraints (2) and (5). Similarly, Recall is defined as $\frac{T^+}{T^+ + F^-} = \frac{T^+}{N_1}$ and can be maximised by setting $W_T^+ = 1$ and $W_T^- = W_F^+ = W_F^- = 0$ (with constraints (3) being redundant).

Accuracy may not be the most appropriate metric for the selected datasets since several are imbalanced. Let $\theta = \frac{N_1}{N}$ be the dataset imbalance level. If, for instance, $\theta \geq 1-\epsilon$ for small ϵ, a high accuracy can be achieved by simply classifying every data point as positive. For imbalanced datasets a possibly useful configuration is setting weights $W_T^+ = (1-\theta)$, $W_T^- = \theta$ and $W_F^+ = W_F^- = 0$. We refer to this function as θ-weighted).

Balanced Accuracy (BA) is an alternative metric which weighs equally the accuracy of positive data points and the accuracy of negative data points. BA is a more appropriate measure for imbalanced datasets [4] and is given by:

$$\text{BA} = \frac{\frac{T^+}{T^+ + F^-} + \frac{T^-}{T^- + F^+}}{2} = \frac{\frac{T^+}{N_1} + \frac{T^-}{N_0}}{2} \tag{13}$$

Theorem 1 shows that maximising BA is equivalent to maximising the θ-weighted function.

Theorem 1. *Maximising the θ-weighted configuration is equivalent to maximising balanced accuracy.*

Proof. Following the definition of the θ-weighted function in Section 3.1, objective function z can be written as:

$$\max z = \left(1 - \frac{N_1}{N}\right)T^+ + \frac{N_1}{N}T^-$$

where $T^+ = \sum_{i=1}^{N_1} t_i^+$, $T^- = \sum_{j=1}^{N_0} t_j^-$ and $\theta = \frac{N_1}{N}$. As $N = N_0 + N_1$ it follows that:

$$\begin{aligned} \max \quad & z = \frac{N_0}{N}T^+ + \frac{N_1}{N}T^- \\ \max \; & Nz = N_0 T^+ + N_1 T^- \\ \max \quad & cz = \frac{T^+}{N_1} + \frac{T^-}{N_0} \end{aligned}$$

where $c = \frac{N}{N_1 N_0} > 0$ is a scaling factor, and thus maximising the θ-weighted function is equivalent to maximising balanced accuracy. □

4 DIVERSITY

As mentioned before many ensemble pruning algorithms employ diversity criteria. Our proposed formulation optimises a performance measure, and in this section we introduce a way to control diversity with linear constraints. We consider a diversity measure called Pairwise Failure Crediting (PFC), proposed originally by [5], chosen due to well-known good performance in imbalanced datasets [2, 12]. PFC measures how diverse an individual classifier is from the remaining classifiers in the ensemble.

PFC is calculated as follows. For each classifier k, we compute a *failure pattern* (FP). A FP is a string of 0's and 1's with length N. A '0' in the string means that the classifier failed to correctly predict the corresponding data point and a '1' means that it predicted the data point correctly (irrespective of its real value). Once we have all failure patterns we take any two classifiers k and l and calculate their Hamming distance. The Hamming distance between same-length strings is the number of different characters in the same positions. For example, if $\text{FP}_k = \{0011011101\}$ and $\text{FP}_l = \{0110001110\}$, the Hamming distance between k and l is 5 (characters 2, 4, 6, 9 and 10 differ). Next, we sum all failures by both classifiers - that is, we sum the number of zeros in both strings which, in the example, is 9. The *failure credit* (FC) between k and l is obtained by dividing the Hamming distance by the sum of failures. In the example, $\text{FC}_{kl} = 5/9$. For every pair $k, l \in \mathcal{K}$ we compute FC_{kl}.

Consider again $\mathcal{S}$ as a set of $S \leq K$ classifiers selected to compose an ensemble. We assume without loss of generality that classifiers in $\mathcal{S}$ are indexed by $k = 1, \ldots, S$. PFC is defined as:

$$\text{PFC}_k = \frac{\sum_{l=1, l\neq k}^{S} \text{FC}_{kl}}{S-1} \qquad k \in \mathcal{S}$$

A (maximum) value of 1 in PFC_k means that k classifies all data points differently from every other classifier in the ensemble, and a (minimum) value of 0 means that k is identical to all other classifiers. Both extreme cases imply that all other classifiers are identical among themselves.

For ensuring minimum desired diversity levels, we propose two approaches: (i) the minimum PFC of any individual classifier is at least a certain threshold $0 \leq \tau \leq 1$ in order to prevent very similar pairs of classifiers and (ii) the average PFC of the ensemble must be at least a certain threshold $0 \leq \gamma \leq 1$ to ensure an overall good level of diversity. Clearly we must have $\gamma \geq \tau$.

We add the following new decision variables. Let $y_{kl} = 1$ if both classifiers k and l have been selected to be part of the ensemble, and $y_{kl} = 0$ if at most one of k and l is chosen to compose the ensemble. This adds $\binom{K}{2}$ extra variables (for every possible pair k, l). For simplicity, both y_{kl} and y_{lk} denote the exact same variable. The following constraints ensure that y_{kl} takes the correct values:

$$y_{kl} \geq x_k + x_l - 1 \qquad \forall k, l \in \mathcal{K}, k < l \tag{14}$$
$$y_{kl} \leq x_k \qquad \forall k, l \in \mathcal{K}, k < l \tag{15}$$
$$y_{kl} \leq x_l \qquad \forall k, l \in \mathcal{K}, k < l \tag{16}$$
$$y_{kl} \geq 0 \qquad \forall k, l \in \mathcal{K}, k < l \tag{17}$$

Notice that there is no need for the y_{kl} variables to be binary. Both x_k and x_l being binary ensure y_{kl} to be 0-1 in any integer solution.

We then rewrite the PFC equation using variables x_k and y_{kl}:

$$\text{PFC}_k = \frac{\sum_{l=1, l\neq k}^{K} \text{FC}_{kl}\ y_{kl}}{\sum_{m=1}^{K} x_m - 1} \qquad \forall k \in \mathcal{K}$$

The term $\sum_{m=1}^{K} x_m$ is the cardinality of the ensemble and any non-selected classifier k (with $x_k = 0$) has a PFC equal to zero (as all $y_{kl} = 0, l \neq k$).

The following linear constraints enforce that every classifier has a minimum PFC of τ:

$$\sum_{\substack{l=1 \\ l\neq k}}^{K} \text{FC}_{kl}\ y_{kl} \geq \tau\Big(\sum_{m=1}^{K} x_m - 1\Big) - K\tau(1 - x_k) \qquad \forall k \in \mathcal{K} \tag{18}$$

The term $K\tau(1-x_k)$ ensures that the constraints above are only enforced if classifier k is chosen to compose the ensemble.

The following nonlinear constraint ensures that the average PFC of the ensemble is at least γ:

$$\frac{1}{\sum_{m=1}^{K} x_m} \frac{\sum_{k=1}^{K} \sum_{l=1, l\neq k}^{K} \text{FC}_{kl}\ y_{kl}}{\sum_{m=1}^{K} x_m - 1} \geq \gamma \tag{19}$$

Observe that in Equation (19) the FCs of every pair are added twice. We use this fact to linearise this expression. For a given subset $\mathcal{S}$,

the average PFC μ_{PFC} is given by:

$$\begin{aligned}\mu_{\text{PFC}} &= \frac{1}{S}\sum_{k=1}^{S}\frac{\sum_{l=1,l\neq k}^{S}\text{FC}_{kl}}{S-1}\\ &= \frac{1}{S(S-1)}\sum_{k=1}^{S}\sum_{\substack{l=1\\l\neq k}}^{S}\text{FC}_{kl}\\ &= \frac{2}{S(S-1)}\sum_{k=1}^{S-1}\sum_{l=k+1}^{S}\text{FC}_{kl}\\ &= \frac{1}{\binom{S}{2}}\sum_{k=1}^{S-1}\sum_{l=k+1}^{S}\text{FC}_{kl} = \mu_{\text{FC}}\end{aligned}$$

where μ_{FC} denotes the average FC of all pairs in the ensemble. We have that the average PFC among all classifiers in the ensemble is equal to the average FC among all pairs.

If S classifiers are selected in the ensemble, then the number of y_{kl} variables that take value 1 is exactly $\binom{S}{2}$. Therefore we can ensure that the average PFC is at least γ with the following linear constraint:

$$\sum_{k=1}^{K-1}\sum_{l=k+1}^{K}\text{FC}_{kl}\ y_{kl} \geq \gamma\sum_{k=1}^{K-1}\sum_{l=k+1}^{K}y_{kl} \tag{20}$$

The expanded formulation with minimum diversity levels is given by maximising (1) subject to (2)-(18) and (20). It requires $\binom{K}{2}$ extra variables and a similar number of extra constraints. Even so, we observed empirically in Section 5.3 that the inclusion of such constraints causes a negligible decrease in solution quality.

5 COMPUTATIONAL EXPERIMENTS

In this section we outline the computational experiments used to evaluate the proposed formulation. We used 9 publicly available datasets, outlined in Table 2[1], ranging from $N = 195$ to $N = 60{,}000$. Imbalance parameter θ is shown in the table.

5.1 Description of the experiments

We prepared 10 different heterogeneous classifier models. Each model was instantiated a number of times with different random seeds and parameters. We set K as multiples of 10 in order to have an equal number of instantiations of each classifier. For instance, if $K = 60$, we have 6 classifiers of each model. In our experiments, reported below, we used $K = \{40, 60, 80, 100\}$. Each classifier produces, as output, a probability of a data point being positive. This probability is rounded to define matrices A and B.

For evaluating performance we used a stratified 10-fold cross-validation procedure. The N data points are initially shuffled randomly and the dataset is split into 10 folds. At each iteration, one of the folds is left out as an independent set. The results presented below are based solely on this set. The other 9 folds, comprising 90% of the original dataset, are joined and split into two sets: a training set, containing 63% of the data points, is used to optimise the individual classifiers. A validation set, comprising the remaining 27% data points, is used to optimise the ensemble algorithms.

The procedure above is repeated 10 times: in each we vary the random seeds required to both shuffle the dataset and initialise the individual classifiers. For each value of K and for each instance shown in Table 2, we run 100 experiments: 10 random initialisations $\times$ 10 folds. For ensuring reproducibility of our results, we have made all necessary data publicly available. A link and a description of the classifiers can be found in the supplementary material[2].

Table 2: Selected datasets from the UCI Machine Learning Repository [18]

Identifier	Dataset	Features	N	N_0	N_1	θ
PRK	Parkinsons	23	195	48	147	0.77
MSK	Musk (Version 1)	168	476	269	207	0.44
BCW	Breast Cancer Wisconsin	32	569	357	212	0.37
QSR	QSAR biodegradation	41	1055	356	699	0.66
DRD	Diabetic Retinopathy Debrecen	20	1151	540	611	0.53
SPA	Spambase	57	4601	2788	1813	0.39
DEF	Default of credit card clients	24	30000	23364	6636	0.22
BMK	Bank Marketing	21	41188	36548	4640	0.11
APS	APS Failure at Scania Trucks	171	60000	59000	1000	0.02

5.2 Benchmarks

We compare our formulation to seven other approaches: Full (non-pruned) Ensemble (FE), Reduced-Error Pruning with Backfitting [14] (hereby Backfitting or BFT), Kappa pruning [20] and four different hill climbing based methods. Here we report here results for the four approaches with the best overall out-of-sample performance. The full results are available in the supplementary material accompanying this paper. All benchmarks classify data points based on majority voting and are allowed to run for a maximum of 5 minutes.

Backfitting follows a greedy approach with revision. From an empty subset S, BFT iteratively adds to S a classifier s such that the accuracy of $S \cup s$ is maximised. This process is repeated until M classifiers are added to S, with ties broken arbitrarily. Whenever a classifier is added, the greedy choice is revised through a local search procedure. Each classifier in the ensemble is iteratively replaced by another previously left out. If the overall accuracy is improved, the method starts again with the new subset S. Kappa pruning is similar, but does not revise the greedy choice and optimises the κ-statistic [6]. Both methods require M to be fixed. For a fairer comparison, we varied M within 20% and 80% of K. The best in-sample results are used to evaluate the independent set.

The other benchmarks use the forward version of the hill climbing search algorithm, differing in the selected fitness function. In all four methods, the first iteration selects the individual classifier with maximum accuracy. Then classifiers are greedily added so as to maximise the selected fitness. This process is repeated until all classifiers are added to S. The chosen ensemble is the one with best fitness over all the ensembles iteratively created. As opposed to the other benchmarks, direct hill climbing does not define the ensemble size *a priori*. The fitness functions chosen are the same as tested by [25]: Accuracy, Complementariness [21], Concurrency [1] (HC-CON) and Uncertainty Weighted Accuracy [25] (HC-UWA).

We compare our method to BFT, HC-CON, HC-UWA and FE.

5.3 Solving the formulation

Due to limited space, in this paper we refrain from evaluating our proposed formulation with regards to the computational effort

[1] All datasets can be found at the UCI Machine Learning Repository [18]

[2] The supplementary material is available here.

required to solve it. We leave that for future work. We however observed in practice that, with a 5-minutes time limit, we were able to either optimally solve or terminate the algorithm with small optimality gaps for all instances.

The average gaps for the results reported in Section 5.4 for $K = 100$ are summarised in Table 3. The "No diversity" column corresponds to **F1** in that section, and only constraints (2)-(12) are used. The "With diversity" column corresponds to **F3**, which uses constraints (2)-(18) and (20). The largest instance, APS, had average gaps of only 0.1% in both cases. The hardest instance was DEF (6.7% and 6.9%). The only case where a difference was notable was for the DRD instance (4.3% and 6.1%).

In our view, even the hardest instances were still relatively close to optimality considering the short computational time. We used CPLEX 12.8 [7] with default parameters as the IP solver and we ran all experiments in an Intel Core(TM) I7-7700 @ 3.60GHz with 32GB of RAM, using 8 cores and having Linux as the operating system.

Table 3: Avg. optimality gaps and standard deviations (in %).

Instance	27% of N	No diversity		With diversity	
		Avg.	Std.	Avg.	Std.
PRK	26	0.0	0.0	0.0	0.0
BCW	129	0.0	0.0	0.0	0.0
MSK	154	0.0	0.0	0.0	0.0
QSR	285	0.0	0.0	0.0	0.1
DRD	311	4.3	2.0	6.1	1.8
SPA	1242	0.0	0.0	0.2	0.2
DEF	8100	6.7	0.4	6.9	0.4
BMK	11121	5.4	0.3	5.5	0.2
APS	16200	0.1	0.0	0.1	0.0

5.4 Accuracy

In the results reported in this section, we seek to maximise accuracy regardless of θ, by setting $W_T^+ = W_T^- = 1$ and $W_F^+ = W_F^- = 0$. We evaluate three different configurations.

In the first, **F1**, we employ only constraints (2)-(12), without enforcing diversity. The other two configurations, **F2** and **F3**, enforce minimum diversity levels in the hope of preventing possible overfitting. In **F2** we only constrain the overall average PFC by setting $\tau = 0$ and $\gamma = \frac{\text{PFC}_{\text{min}}+\text{PFC}_{\text{avg}}}{2}$, where PFC_{min} and PFC_{avg} are the minimum individual PFCs among all classifiers and the average PFC of the full ensemble. In **F3** we also set $\tau = \text{PFC}_{\text{min}}$.

Table 4 summarises the results with an average rank per value of K across all datasets. We use the ranking procedure of [9]. The full results are shown in the supplementary material.

Table 4: Average ranks of accuracies

K	**F1**	**F2**	**F3**	BFT	HC-CON	HC-UWA	FE
40	3.82	3.74	3.79	4.04	3.71	3.66	5.25
60	3.82	3.70	3.76	4.03	3.66	3.66	5.39
80	3.82	3.80	3.75	4.09	3.54	3.58	5.42
100	3.91	3.72	3.72	4.07	3.55	3.56	5.48
Avg:	**3.84**	**3.74**	**3.75**	**4.06**	**3.61**	**3.61**	**5.38**

The results suggest that while our proposed formulation is overall competitive, it was slightly outperformed by HC-CON and HC-UWA - both in terms of average accuracy (from the table in the supplementary material) and average rank. Still, with the exception of FE, the difference between BFT (worst performing) and HC-CON (best performing) was 0.33% in terms of average overall accuracy and 0.45 in terms of average rank. Adding diversity constraints to our formulation also had a small beneficial impact in improving average accuracy and reducing the average ranking. In 11 out of the 36 cases, **F2** outperformed all benchmarks.

Both HC benchmarks had a higher dispersion of accuracies than our methods. Also, adding diversity in **F2** and **F3** helped reduce dispersion. Further studies on either better enforcing these constraints or proposing new constraints based on alternative diversity measures remain as future work. Since our proposed method is exact in nature (although limited to 5 minutes), in the supplementary material we discuss in more detail the effects of overfitting.

5.5 Balanced accuracy

In this section, we evaluate the out-of-sample performance according to Balanced Accuracy (BA). We employ **F1** as defined earlier and a modified **F1** where we maximise the θ-weighted configuration suggested in Section 3.1. Tables 5 and 6 show the results. In Table 5, we show results for only $K = \{80, 100\}$ and for the five largest datasets, but the full results are available in the supplementary material. A bold value in the **Avg.** columns means that our formulation obtained a higher average than all benchmarks. The averages in the last row are for all results, not only those displayed in the table. We did not rerun the experiments for the accuracy version of **F1** nor for the benchmarks, rather we used the same ensemble subsets to calculate the corresponding balanced accuracies.

Here both configurations of our formulation outperformed the benchmarks. **F1** (θ-weighted) consistently outperformed **F1** and all benchmarks, with better ranks, overall average accuracies and lower dispersion, especially for the larger (and more imbalanced) datasets. **F1** (θ-weighted) outperformed all benchmarks in 22 out of 36 cases. It had worse performance than the benchmarks in MSK and DRD, which are the most balanced datasets. These results suggest that being able to configure the objective function according to the characteristics of the dataset at hand can be highly beneficial.

6 CONCLUSIONS AND FUTURE DIRECTIONS

In this work we proposed an IP approach for the problem of selecting a subset of classifiers in ensemble learning, with the goal of maximising a weighted function of the patterns in the confusion matrix. In order to combine performance and diversity criteria, we also proposed linear constraints to enforce minimum diversity levels. We observed that state-of-the-art solvers can find good solutions in reasonable computational times for relatively large datasets. The IP approach is, in our view, able to provide a flexible exact algorithm which can also be used as a heuristic if short computational time limits are required. This approach has the additional advantage of providing bounds on optimal values.

We compared our formulation to seven well-known benchmarks. We used a stratified 10-fold cross validation procedure and evaluated the effect of enforcing minimum diversity levels and varying

Table 5: Balanced Accuracy averages and standard deviations

Dataset	K	**F1**		**F1** (θ-weighted)		BFT		HC-CON		HC-UWA		FE	
		Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
DRD	80	0.7457	0.0064	0.7468	0.0046	0.7503	0.0057	0.7525	0.0065	0.7525	0.0072	0.7126	0.0077
	100	0.7404	0.0088	0.7466	0.0093	0.7479	0.0063	0.7506	0.0075	0.7551	0.0057	0.7148	0.0075
SPA	80	0.9504	0.0021	**0.9521**	0.0018	0.9488	0.0017	0.9512	0.0017	0.9493	0.0020	0.9403	0.0009
	100	0.9500	0.0022	**0.9517**	0.0025	0.9485	0.0020	0.9515	0.0015	0.9493	0.0018	0.9402	0.0008
DEF	80	**0.6662**	0.0014	**0.6985**	0.0016	0.6484	0.0020	0.6553	0.0009	0.6557	0.0011	0.6484	0.0011
	100	**0.6661**	0.0019	**0.6992**	0.0014	0.6482	0.0018	0.6542	0.0020	0.6560	0.0007	0.6473	0.0012
BMK	80	**0.7765**	0.0055	**0.8684**	0.0015	0.7322	0.0046	0.7477	0.0028	0.7469	0.0018	0.6822	0.0036
	100	**0.7794**	0.0053	**0.8694**	0.0012	0.7363	0.0040	0.7478	0.0029	0.7479	0.0018	0.6762	0.0031
APS	80	**0.8731**	0.0039	**0.9395**	0.0038	0.8398	0.0045	0.8535	0.0033	0.8500	0.0040	0.8021	0.0037
	100	**0.8735**	0.0053	**0.9416**	0.0041	0.8447	0.0064	0.8562	0.0040	0.8513	0.0032	0.8006	0.0032
Average:		**0.8433**	**0.0080**	**0.8665**	**0.0063**	**0.8313**	**0.0069**	**0.8397**	**0.0075**	**0.8383**	**0.0076**	**0.8111**	**0.0057**

Table 6: Average ranks of balanced accuracies

K	**F1**	**F1** (θ-weighted)	BFT	HC-CON	HC-UWA	FE
40	**2.98**	**2.37**	3.94	3.33	3.43	4.95
60	**2.88**	**2.45**	3.97	3.29	3.40	5.02
80	**2.92**	**2.39**	3.97	3.25	3.38	5.09
100	**2.97**	**2.41**	4.00	3.20	3.36	5.06
Avg:	**2.94**	**2.40**	**3.97**	**3.27**	**3.39**	**5.03**

the weights assignments of the objective function. The results suggest that our approach is competitive and its flexibility can be beneficial when dealing with different datasets. All data required to reproduce our results is made available as supplementary material.

As future work we intend to experiment with different criteria and larger datasets. We also plan to study alternative diversity constraints and to research IP techniques/matheuristics for both finding good solutions quickly and solving the formulation faster.

REFERENCES

[1] Robert E. Banfield, Lawrence O. Hall, Kevin W. Bowyer, and W.Philip Kegelmeyer. 2005. Ensemble diversity measures and their application to thinning. *Information Fusion* 6, 1 (3 2005), 49–62. https://doi.org/10.1016/j.inffus.2004.04.005

[2] Urvesh Bhowan, Mark Johnston, Mengjie Zhang, and Xin Yao. 2012. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Transactions on Evolutionary Computation* 17, 3 (2012), 368–386. https://doi.org/10.1109/TEVC.2012.2199119

[3] L. Breiman. 1996. Bagging predictors. *Machine Learning* 24, 2 (1996), 123–140. https://doi.org/10.1007/BF00058655

[4] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. 2010. The balanced accuracy and its posterior distribution. In *2010 20th international conference on pattern recognition*. IEEE, 3121–3124. https://doi.org/10.1109/ICPR.2010.764

[5] A. Chandra and X. Yao. 2006. Ensemble learning using multi-objective evolutionary algorithms. *Journal of Mathematical Modelling and Algorithms* 5, 1 (2006), 417–445. https://doi.org/10.1007/s10852-005-9020-3

[6] J. Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.

[7] CPLEX Optimizer. 2021. IBM. Available from https://www.cplex.com, last accessed April 1st.

[8] R. M. O. Cruz, R. Sabourin, and G. D. C. Cavalcanti. 2018. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion* 41, May (2018), 195–216. https://doi.org/10.1016/j.inffus.2017.09.010

[9] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research* 7 (2006), 1–30. https://doi.org/10.5555/1248547.1248548

[10] R. P. W. Duin. 2002. The combining classifier: to train or not to train? *Object recognition supported by user interaction for service robots* 2, c (2002), 765–770. https://doi.org/10.1109/ICPR.2002.1048415

[11] Wei Fan, Fang Chu, Haixun Wang, and Philip S. Yu. 2002. Pruning and dynamic scheduling of cost-sensitive ensembles. In *AAAI/IAAI*. 146–151.

[12] Everlandio RQ Fernandes, André CPLF de Carvalho, and André LV Coelho. 2015. An evolutionary sampling approach for classification with imbalanced data. In *IJCNN'15. International Joint Conference on Neural Networks*. IEEE, 1–7. https://doi.org/10.1109/IJCNN.2015.7280760

[13] Y. Freund and R. E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 1 (1997), 119–139. https://doi.org/10.1006/jcss.1997.1504

[14] J. H. Friedman and W. Stuetzle. 1981. Projection pursuit regression. *Journal of the American statistical Association* 76, 376 (1981), 817–823. https://doi.org/10.1080/01621459.1981.10477729

[15] G. Giacinto, F. Roli, and G. Fumera. 2000. Design of effective multiple classifier systems by clustering of classifiers. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, Vol. 2. IEEE, 160–163. https://doi.org/10.1109/ICPR.2000.906039

[16] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. 1998. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence* 20, 3 (1998), 226–239. https://doi.org/10.1109/ICPR.1996.547205

[17] Aleksandar Lazarevic and Zoran Obradovic. 2001. Effective pruning of neural network classifier ensembles. In *IJCNN'01. International Joint Conference on Neural Networks.*, Vol. 2. IEEE, 796–801. https://doi.org/10.1109/IJCNN.2001.939461

[18] M. Lichman. 2021. UCI Machine Learning Repository. Available from http://archive.ics.uci.edu/ml, last accessed April 1st.

[19] Z. Lu, X. Wu, X. Zhu, and J. Bongard. 2010. Ensemble pruning via individual contribution ordering. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 871–880. https://doi.org/10.1145/1835804.1835914

[20] D. D. Margineantu and T. G. Dietterich. 1997. Pruning adaptive boosting. In *Proceedings of the 14th International Conference of Machine Learning*, Vol. 97. Citeseer, 211–218. https://doi.org/10.1007/11875581_39

[21] Gonzalo Martınez-Muñoz and Alberto Suárez. 2004. Aggregation ordering in bagging. In *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*. Citeseer, 258–263. https://doi.org/10.1.1.146.3650

[22] G. Martínez-Muñoz and A. Suárez. 2006. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd International Conference on Machine Learning*. ACM, 609–616. https://doi.org/10.1145/1143844.1143921

[23] Ioannis Partalas, Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2006. Ensemble pruning using reinforcement learning. In *Hellenic Conference on Artificial Intelligence*. Springer, 301–310. https://doi.org/10.1007/11752912_31

[24] Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. 2008. Focused ensemble selection: A diversity-based method for greedy ensemble selection. *Frontiers in Artificial Intelligence and Applications* 178 (2008), 117–121. https://doi.org/10.3233/978-1-58603-891-5-117

[25] Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. 2010. An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. *Machine Learning* 81, 3 (2010), 257–282. https://doi.org/10.1007/s10994-010-5172-0

[26] G. Tsoumakas, I. Katakis, and I. Vlahavas. 2004. Effective voting of heterogeneous classifiers. In *European Conference on Machine Learning*. Springer, 465–476. https://doi.org/10.2147/JPR.S129139

[27] Y. Zhang, S. Burer, and W. N. Street. 2006. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research* 7, Jul (2006), 1315–1338. https://doi.org/10.1016/j.jasms.2006.06.007

Learning to Prune Instances of Steiner Tree Problem in Graphs

Full Paper

Jiwei Zhang
University College Dublin
Dublin, Ireland

Dena Tayebi
University College Dublin
Dublin, Ireland

Saurabh Ray
New York University
Abu Dhabi, United Arab Emirates

Deepak Ajwani
University College Dublin
Dublin, Ireland

ABSTRACT

We consider the Steiner tree problem on graphs where we are given a set of nodes and the goal is to find a tree sub-graph of minimum weight that contains all nodes in the given set, potentially including additional nodes. This is a classical NP-hard combinatorial optimisation problem. In recent years, a machine learning framework called learning-to-prune (L2P) has been successfully used for solving a diverse range of combinatorial optimisation problems. In this paper, we use this learning framework on the Steiner tree problem and show that even on this problem, the learning-to-prune framework results in computing near-optimal solutions on a large majority of the instances at a fraction of the time required by commercial ILP solvers. Furthermore, we show that on instances from the SteinLib and PACE Challenge datasets where the L2P framework does not find the optimal solution, the optimal solution can often be discovered by either using a lightweight local search algorithm to improve the L2P solution or using L2P solution as a warm start in an ILP solver. Our heuristic for Steiner tree problem leverages historical solutions of known solutions for past instances from the same distribution. Our results underscore the potential of the L2P framework in solving various combinatorial optimisation problems.

KEYWORDS

Minimum Steiner Tree, Combinatorial Optimisation, Machine Learning, Learning to Prune

1 INTRODUCTION

Steiner tree problem is a classical well-studied combinatorial optimisation problem. It is applied to various problems in research and industry [12] including various network design problems (see e.g., [9]).We consider the variant of this problem on graphs, where we are given an input weighted graph, a set of terminal nodes V and the goal is to compute a minimal connected subgraph that contains all nodes in V. This variant of the problem is NP-hard. Furthermore, it is even NP-hard to approximate it within a factor of 96/95 [3]. Given its applications and hardness, numerous approximation algorithms and heuristics have been developed to solve this problem efficiently. We refer the reader to the PACE challenge 2018 report [2] for a list and ranking of the various algorithms and heuristics developed for this problem.

Traditional approaches to solve combinatorial optimisation problems include the usage of integer linear programming solvers, constraint programming approaches, parameterised and approximation algorithms, various heuristics including nature based metaheuristics (e.g., genetic algorithms) and customising algorithms to specific input distribution. In recent years, machine learning techniques have been explored to speed-up the computation of solutions (c.f., [1] for a survey). Machine learning techniques are particularly useful in applications where the same optimisation problem is solved repeatedly on instances coming from the same distribution [6]. Many of these learning techniques aim to learn the optimal solution directly. An example of such an end-to-end machine learning technique on Steiner tree problem is the Cherrypick solution by Yan et al. [19], where a deep reinforcement learning technique called DQN is used together with an embedding to encode path-related information in order to predict the optimal solution directly. Such end-to-end approaches generally suffer from poor generalisation (resulting in poor solutions for larger and/or more complex problem instances) and large training requirement. To deal with these issues, these approaches would need to collect the training data by solving a large number of problem instances of the same size as the test instances. Furthermore, these end-to-end deep learning approaches also suffer from poor interpretability and explanability of the algorithms learnt. This is because the learnt algorithm is implicit in the millions of parameters of the deep learning architecture. Since in real industry setting, new constraints are routinely added to the problem, poor interpretability means that we do not know if the learnt model will still work with newly discovered constraints and thus, new models have to be learnt from scratch every time this occurs.

A key reason for the poor generalizability of end-to-end approaches is that they do not leverage any algorithmic insight into the problem, instead relying solely on the input and embedding vectors. This is also an important factor for them needing deep learning models with poor interpretability. To address some of these issues, a learning-to-prune approach [4, 10, 11, 16] has been proposed. Instead of trying to predict the optimal solution directly, it uses a supervised learning model to predict the elements (e.g., nodes/edges) that are unlikely to be part of optimal solution and prune them from further consideration. Once these elements are pruned out, the problem on the remaining elements (predicted to be in optimal solution by the classifier or where the classifier was

not confident) is usually quite small and can, thus, be solved using existing exact/approximate approaches. The supervised learning approach leverages a large number of features that can capture the algorithmic insights from the state-of-the-art literature on the problem. As such, classification techniques with significantly fewer parameters, such as random forest, SVM etc. work very well in this framework and there is no need for more complex deep learning models. An added advantage of this framework is that it requires far fewer labelled training instances for training, which is vital for NP-hard optimisation problems.

In this paper, we explore if the learning-to-prune framework can be used to solve the Steiner tree problem efficiently. Towards this end, we carefully select an ILP formulation with small integrality gap and the features used in the learning approach. We consider the instances from SteinLib dataset [7, 8] and the 2018 PACE challenge [2]. We show that on the instances where the LP relaxation doesn't return integral solutions, learning-to-prune framework is able to obtain optimal or near-optimal solution in orders of magnitude less time compared to solving the ILP formulation directly using Gurobi. Furthermore, we show that even on instances where the solution returned by learning-to-prune is not optimal, we can often achieve optimal solution by using a lightweight local search algorithm to improve the learning-to-prune solution. Using the learnt solution as a warm start in Gurobi is also quite effective in finding optimal or near-optimal solutions for many of the instances where the learning-to-prune solution is not already optimal.

We note that many state-of-the-art Steiner tree problem solvers [12–15] use preprocessing and problem reduction techniques and some of these techniques rely on features similar to ones used in our adaptation of learning-to-prune framework. While these solvers are very successful in practice, they need to be carefully redesigned for each new variant of the problem. In contrast, the effort required to adapt the learning-to-prune framework to the other variants is likely to be considerably less. Thus, our work paves the way for augmenting the ability of algorithm designers to develop preprocessing techniques that can leverage the specific application constraints and the input distributions. Our learnt preprocessing can be combined with well-known "exact" preprocessing techniques in different ways.

2 APPLYING LEARNING-TO-PRUNE TO STEINER TREE PROBLEM

In the context of minimum Steiner tree problem, the learning-to-prune framework learns a classification model to predict whether an edge will be part of the optimal solution or not. The training examples consists of edges from a set of training graphs and the classification model learns a mapping from a feature vector of an edge to its label. The edges, for which the classifier is highly confident that they are not part of the optimal solution, are pruned out and the remaining (hopefully much smaller) instance is then solved using an ILP solver.

The key decisions we need to make in order to apply the learning-to-prune framework involve (i) the choice of the ILP formulation for the Steiner tree problem, (ii) the choice of features and (iii) the classification models. We found that the choice of the ILP formulation was crucial for the success of the learning-to-prune framework on this problem. An ILP formulation with smaller integrality gap seems to be particularly suitable for this framework as its LP relaxation can be used as a highly discriminative feature in the process of search-space pruning. Thus, we carefully considered the various ILP formulations for this problem and opted for a formulation based on multicommodity flow transmission [18].

We note that there are other formulations such as those based on directed cut [13] that are algorithmically more efficient (see the survey article [12]). We expect that the gains from learning-to-prune approach will be similar for these other formulations as well.

2.1 Integer Linear Programming Formulation

In this formulation, we first convert an undirected Steiner tree problem into a directed version by replacing each edge $\{i, j\}$ with weight c_{ij} by two directed edges (i, j) and (j, i) of the same weight c_{ij}. Then, we consider the problem of connecting the set of terminal nodes in the undirected graph as sending a unit flow from an arbitrary terminal node (referred to as root and indexed as node 1) to the remaining terminal nodes in the corresponding directed graph. In particular, the k^{th} flow goes from the root to the k^{th} terminal node (the first flow goes from the root node to itself). Since all flows are moving away from root and towards the terminal nodes, the aggregation of these paths result in the Steiner tree in the undirected graph. Next, we describe this formulation in more detail:

2.1.1 *Sets and Indices.*

- $i \in N = \{1, 2, .., n\} = \{1\} \cup V \cup S$: The index number of root node is 1. Here, $\{1\} \cup V$ is the set of terminal nodes and S is the set of remaining nodes.
- $E = \{(i, j)\}$: Set of directed edges. Note that the size of E is double the size of the edge set of the undirected graph.
- $G = (N, E)$: A graph where the set N defines the set of nodes, the set E defines the set of directed edges and the set $\{1\} \cup V \subseteq N$ defines the set of terminals.
- $T \subset E$: Set of edges that represents a tree spanning $\{1\} \cup V$ in G.
- $c_{ij} \in R^+$: The cost of the arc (i, j), for all $(i, j) \in E$.

2.1.2 *Decision Variables.*

- $y_{ij} \in \{0, 1\}$: This variable is equal to 1, if edge (i, j) is in the set T. Otherwise, the decision variable is equal to zero.
- x_{ij}^k: This is the amount of commodity k (the amount of flow from node 1 to k) that goes through edge (i, j).

2.1.3 *Objective Function.* Minimize the total cost of T:

$$minimize \sum_{(i,j)\in E} c_{ij} \cdot y_{ij} \tag{1}$$

2.1.4 *Constraints.*

$$\sum_{h\in N} x_{ih}^k - \sum_{j\in N} x_{ji}^k = \begin{cases} 1, & i = 1, \\ -1, & i = k, \\ 0, & i \neq 1, k. \end{cases} \quad \forall k \in V, \tag{2}$$

$$x_{ij}^k \leq y_{ij}, \tag{3}$$

$$x_{ij}^k \geq 0, \quad \forall (i, j) \in E, k \in V, \tag{4}$$

$$y_{ij} \in \{0, 1\}. \quad (5)$$

2.1.5 Constraints Explanation. As described before, we use an embedded multi-commodity network flow problem to describe the connectivity of the Steiner tree problem. In constraint 2, one unit of commodity k must be transmitted from node 1 to node k. Constraint 3 indicates that when the flow is allowed in an edge, the edge must be in the solution. Constraint 4 enforces that the flow on any edge is non-negative while Constraint 5 enforces that the variables y_{ij} are binary. Constraints 2- 5 indicate that a feasible solution must have a directed path of edges (i. e. $y_{ij} = 1$) between node 1 and a node belonging to V.

2.2 Features

Another requirement for applying the learning-to-prune framework is to have a set of highly discriminative features that can separate the edges in the optimal solution from those that are not. In other words, we want to associate a set of features with each edge that will allow us to train a classifier separating the set of edges in the optimal solution from the other edges. For the Steiner tree problem, we consider features associated with the LP relaxation, weight of the edges with respect to other edge weights and centrality of associated nodes. Except for Eigenvector centrality, these features can be computed quite fast and they allow us to achieve a high degree of pruning with little loss in objective function value.

2.2.1 LP relaxation feature. The first feature is the value of edge variables y_{ij} in the LP relaxation of the problem. A high value of this variable suggests a higher likelihood of the edge appearing in the optimal solution. We didn't consider any signals from the dual of the problem and we also didn't utilise the variables x_{ij}^k in this study. Note that the computation of LP relaxation requires significantly less time compared to the ILP computation.

2.2.2 Weight-Related Features. We used the following weight-related features: (i) normalised weight $w_N(e) = (w(e) - w_{min})/(w_{max} - w_{min})$ where $w(e)$ is the weight of the edge e and w_{min} and w_{max} are the lightest and heaviest edge-weight in the graph, (ii) standardised weight $w_S(e) = (w(e) - \mu(w))/\sigma(w)$ where $\mu(w)$ and $\sigma(w)$ are the mean and standard deviation of the edge-weights, (iii) chi-square of normalised weight $w_C(e) = (w_N(e) - \mu(w_N))^2/\mu(w_N)$ where $\mu(w_N)$ is the mean of the normalised edge-weights and (iv, v) local rank of edge (i, j) at node i and j. Here, local rank refers to the rank of this edge in the sorted order of all edges (by weight) incident at the node.

2.2.3 Centrality Features. To capture the discriminative power of an edge further, we also use the centrality of the incident nodes. Intuitively, the edges between highly central nodes are more likely to be part of optimal Steiner trees as they are crucial for low-weight connectivity. Specifically, we use the following centrality features: degree centrality, betweenness centrality and Eigenvector centrality. Degree centrality is defined as the number of links incident upon a node, which is the simplest centrality feature to calculate. It simply measures the importance of a node by how many edges are incident to it. Betweenness centrality is widely used in weighted graphs as it captures the fraction of shortest paths passing through a given node [17]. The betweenness centrality of a node v is defined as $C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$ where σ_{st} is the total number of shortest paths between s and t and $\sigma_{st}(v)$ is the number of shortest paths between s and t that pass through v. Eigenvector centrality [20] is also an important centrality feature that captures the "influence" of a node in the network: A high eigenvector score means that a node is connected to many nodes who themselves have high scores.

For all centrality features, NetworkX is used to construct the graph and calculate the feature values [5]. As these centrality features are focusing on nodes instead of edges, each centrality metric results in two features for an edge (i, j) corresponding to the smaller and the bigger value of the two incident nodes i and j.

2.3 Classification

As noted by the previous work on learning-to-prune [4, 10, 11, 16], the exact classification model is not so crucial in this framework. We experimented with five different classification techniques: **Random forest** (RF), **Support vector machine** (SVM), **Logistic Regression** (LR), **K-nearest neighbour** and **Gaussian naive bayes**. While the SVM performs best on this problem, the main insights from the experimental results remain the same for all these classifiers. This provides us confidence that our results are not too specific to a particular classification model, but are more broad-based.

2.4 ILP on the pruned subgraph

We run the exact Steiner tree ILP formulation on the unpruned graph so obtained. This can be done by fixing all the edge variables of the pruned edges to 0 in the ILP formulation and solving the modified ILP using an ILP solver. The output of this modified ILP is then returned as the output of the learning-to-prune approach.

2.5 Ensuring Feasibility

One issue with the learning-to-prune framework is that the remaining set of edges may not contain any feasible solution of the problem that satisfies all constraints. In other words, the pruned graph (graph remaining after the pruned edges have been removed) may have multiple connected components with nodes in V divided between these components. To resolve this issue, we add back all edges for which the corresponding variable has a non-zero value in the LP relaxation. Assuming that the input graph was connected, the set of edges with non-zero values in LP relaxation solution will maintain connectivity among the terminal nodes and thus, with their addition, feasibility will be guaranteed.

Next, we evaluate the quality of this solution as well as the running time of this approach and the relative importance of the different features used in the framework.

3 RESULT

3.1 Experimental Setup

In the benchmark SteinLib [7] dataset, we found that there are only 55 problem instances for which the LP relaxation of the considered ILP formulation does not return integral solutions. Thus, we only focus on these instances and select 80% of them with the smallest running times as training and use the remaining 20% of the instances with the largest running time as the test dataset. This is because

we want to show that our model generalises from smaller instances to larger and more complex instances in this dataset.

3.1.1 Feature Importance. The feature importance for training the model is shown in Table 1 for a SVM classification model. Unsuprisingly, Table 1 shows that the LP relaxation feature is the most discriminative of all. An important observation here is that even though LP relaxation feature is important, it accounts for less than half of the discriminative power of all the features. This implies that this feature alone isn't enough, but other features also contribute significantly to improving the accuracy of the classification model and the entire learning framework is needed. We also considered the feature importance from a logistic regression classification model[1]. Similar to the case of SVM, it showed that while the LP relaxation feature is the most discriminative, the other features (such as the maximum degree centrality of the two incident nodes and the minimum local rank of the two incident nodes) also prove to be quite useful in the classification.

Feature	Importance
LP relaxation feature	0.462
Normalised Weight	0.108
Variance	0.083
Degree Centrality Max	0.052
Eigenvector Centrality Max	0.048
Betweenness Centrality Max	0.048
Degree Centrality Min	0.046
Local Rank j	0.041
Eigenvector Centrality Min	0.039
Betweenness Centrality Min	0.038
Local Rank i	0.036

Table 1: Relative feature importance of different features based on a SVM classification model

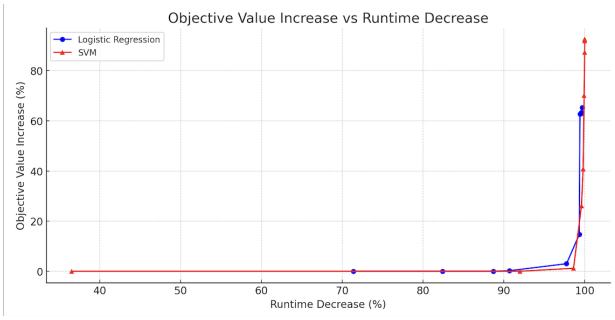


Figure 1: Trade-off obtained by varying the threshold of a SVM classifier

[1]calculated as the product of the feature coefficient with the standard deviation of feature values in the training set

3.1.2 Objective Function vs. Running Time Trade-off. As noted in Figure 1, both SVM classifier and logistic regression classifiers obtained a drastic reduction in running time at little loss in objective function value. In these plots, we vary the pruning thresholds. A pruning rate of 60-70% resulted in a significant reduction in running time while increasing the objective function only slightly. While the general trends are similar between SVM and logistic regression, SVM gives a better trade-off between the objective function value and running time. The drastic reduction in running time at little loss in objective function value is further illustrated in Table 2 and 3, which presents the results of the learning-to-prune approach on the 10 test instances using the SVM classification model. We first note that on these larger and more complex instances, the time to compute all the features including the LP values is very small compared to the time to run the original ILP. More importantly, the running time of the learning-to-prune approach (including the time to compute features and then running the ILP with hard pruning constraint) is around 99% less than the original ILP solver time on these instances (using the Gurobi solver). In 7 of these 10 instances, the hard pruning is able to find the optimal solution itself in significantly less time. In the remaining three instances, the resultant increase in the objective function value because of the mistakes in the pruning process is still very small (less than 0.6%).

	Objective (Original)	Objective (After Pruning)	Objective Increase %
i160-344	8307	8324	0.20
i160-244	5076	5103	0.53
i160-345	8327	8327	0
i160-343	8275	8275	0
i160-342	8348	8355	0.08
i160-313	9159	9159	0
i160-241	5086	5086	0
i160-341	8331	8331	0
i160-245	5084	5084	0
i160-242	5106	5106	0

Table 2: Objective function values returned by Gurobi ILP solver and the learning-to-prune framework (with SVM classifier) for different test instances

At this stage, a natural question to ask is how do these results compare with directly pruning based on the LP relaxation values with different thresholds. Next, we consider the three instances where the hard pruning doesn't get optimal results and compare the results of the hard pruning with pruning based on the LP relaxation value. Figures 2 and 3 presents the result of such a comparison. We observe that the hard pruning provides significantly better objective value vs running time trade-off compared to the Gurobi ILP solver. In particular, note that Gurobi requires considerably more time to reach a comparable objective function value. In all three instances, hard pruning based on a diverse range of features provides solutions with better objective function values compared to directly pruning based on LP relaxation values, even though it takes some more time. In these plots, the dashed orange horizontal line in these curves represent the objective function value of the

	Runtime (Original)	Time to Compute Features	ILP Solver Runtime	Runtime Decrease %
i160-344	27245.21	54.17	157.71	99.22
i160-244	7762.75	25.68	47.13	99.06
i160-345	70653.84	51.93	242.82	99.58
i160-343	20897.36	50.54	114.90	99.21
i160-342	91351.38	60.09	1384.39	98.42
i160-313	3832.54	15.25	84.73	97.39
i160-241	6446.48	24.78	32.78	99.11
i160-341	52473.68	53.65	104.74	99.70
i160-245	3014.05	28.05	15.95	98.54
i160-242	4817.80	27.34	42.81	98.54

Table 3: Running time taken by Gurobi ILP solver and the learning-to-prune framework (with SVM classifier) for different test instances

optimal ILP solution on the instance obtained by pruning all edges with zero LP relaxation value. Note that even this value is higher than the solutions from our hard pruning approach.

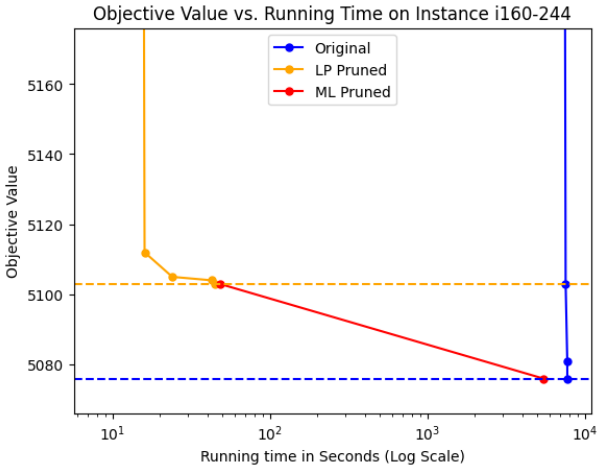


Figure 2: Comparing hard pruning (referred "ML Pruned") with LP-based pruning and Gurobi ILP solver on the original formulation of Steiner tree problem on instances i160-244. The dashed blue horizontal line represents the optimal ILP solution, while the dashed orange horizontal line represents the optimal ILP solution on the instance obtained by pruning all edges with zero LP values.

In applications where we wish to reduce the optimality gap even further, we can use the soft pruning approach. The idea here is that instead of adding a hard constraint that no edge can be taken from the set of pruned edges (fixing those edge variables to 0), we add a soft constraint that a small constant number of edges can be taken from the set of pruned edges in the returned solution. In other words, we add the constraint that sum of all edge variables corresponding to pruned edges has to be less than equal to a small

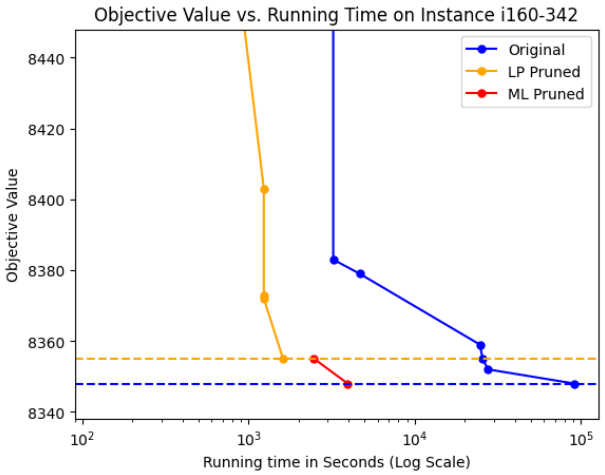


Figure 3: Comparing hard pruning (referred "ML Pruned") with LP-based pruning and Gurobi ILP solver on the original formulation of Steiner tree problem on instance i160-342.

constant. This is implemented by simply adding the corresponding constraint in the ILP formulation. While the soft pruning still retains all the edge variables in the ILP formulation, it prunes the search space considerably. When applied on the instance i160-344, the soft pruning approach, that allows just one edge from the pruned set, finds the optimal solution of the original problem. The running time of this approach on this instance is around 3000 seconds, which is still considerably less than the original ILP time of around 27000 seconds, but more than the time of the hard pruning approach (around 150 seconds).

3.2 Results on PACE Challenge datasets

Next, we consider the 200 instances from the track 1 of the 2018 PACE challenge [2]. This track provided the benchmark dataset for the exact search techniques. Of the 200 instances, the LP relaxation of our ILP formulation was able to compute the optimal integral solution on 148 instances. On another 17 instances, the LP solution was very close to the optimal integral solution. Thus, we focused on the remaining 35 instances. Out of these instances, we used 13 for training and tested on the remaining 22 instances. Based on the results on the SteinLib dataset, we decided to use SVM as the classification technique for this dataset. Table 4 shows that on all of these instances, the learning to prune approach provides optimal or near-optimal solutions.

On the few instances where the solution returned by learning-to-prune was not optimal, we use a lightweight local search algorithm to improve the solution. We define the local neighbourhood function to consist of solutions that can be obtained by removing one edge from the current solution and adding one replacement edge (from outside the current solution) in its place. We find the best solution in this local neighbourhood. This can be computed efficiently as the returned solution is a tree and removing an edge leaves the tree disconnected. Thus, to find a replacement edge, we only need to consider edges that connect the two components and find a

	Objective (Original)	Objective (After Pruning)	Objective Increase %
010	2338	2344	0.26
109	939	942	0.32
141	2200557	2200560	0.0001
160	1996	1996	0.0
161	5199	5209	0.19
162	5193	5193	0.0
164	5205	5205	0.0
165	5218	5218	0.0
171	42	42	0.0
172	7229	7304	0.07
173	71	72	1.4
176	10519	10519	0.0
195	54	54	0.0
196	100	101	1.0

Table 4: Objective function values returned by Gurobi ILP solver and the learning-to-prune framework for different test instances of PACE Challenge dataset

minimum weight replacement edge. We found that this local search algorithm was able to yield the optimal solution when initialised with the learning-to-prune solution. For instance, on instance 010, the objective function values of the optimal solution and learning-to-prune solution are 2338 and 2344, respectively. The local search was able to improve the learning-to-prune solution to the optimal solution. Similarly on instance 109, the optimal solution and the learning-to-prune solution had objective function values of 939 and 942 respectively. Again, the local search was able to obtain the optimal solution.

We found that using the learning-to-prune solution as a warm start in the Gurobi ILP solver is also quite effective. For instance, on instance 010 (one of the few instances where the learning-to-prune doesn't return an optimal solution), Gurobi warm start from learning-to-prune returns an optimal solution.

The track 2 of the PACE challenge was dedicated to instances that have small tree-width. Surprisingly, we discovered that our learning model that was trained on track 1 instances (exact algorithms track) was quite effective on track 2 instances as well. The solution obtained after pruning was optimal in 9 out of 15 track 2 instances where the LP and ILP solutions had different objective function values. On the remaining 6 test instances, the solution obtained using the learning-to-prune framework was within a multiplicative factor of 1.0002 of the optimal solution. Thus, we conclude that our learning model is also quite robust to changes in input distribution and instance sizes.

4 CONCLUSION

Our experiments show that the learning-to-prune framework provides optimal or near-optimal solutions on instances of the SteinLib and PACE challenge benchmarks at a fraction of the costs of the Gurobi ILP solver. While the feature based on LP relaxation is unsurprisingly the most discriminatory feature for classification, the hard pruning is able to achieve better objective function value compared to pruning directly based on LP relaxation values. It shows that combining the signal from different features using classification models is an effective strategy to prune the problem instances. On the few instances where the learning-to-prune solution is not optimal, we show that using it as a warm start for Gurobi ILP solver or using it as the initial solution for a lightweight local search heuristic can often yield the optimal solution. We expect that this framework will also be effective on many other network optimisation problems.

ACKNOWLEDGMENTS

The research of first and last author is partially supported by Science Foundation Ireland under Grant number 18/CRT/6183.

REFERENCES

[1] Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. 2021. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research* 290 (2021), 405–421. Issue 2.

[2] Édouard Bonnet and Florian Sikora. 2019. The PACE 2018 Parameterized Algorithms and Computational Experiments Challenge: The Third Iteration. In *13th International Symposium on Parameterized and Exact Computation (IPEC 2018) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 115. 26:1–26:15.

[3] Miroslav Chlebík and Janka Chlebíková. 2008. The Steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science* 406, 3 (2008), 207–214.

[4] James Fitzpatrick, Deepak Ajwani, and Paula Carroll. 2021. Learning to Sparsify Travelling Salesman Problem Instances. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Springer Cham.

[5] Aric Hagberg, Pieter Swart, and Daniel S Chult. 2008. *Exploring network structure, dynamics, and function using NetworkX*. Technical Report LA-UR-08-5495. Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

[6] Elias B. Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. 2017. Learning Combinatorial Optimization Algorithms over Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*. 6348–6358. https://proceedings.neurips.cc/paper/2017/hash/d9896106ca98d3d05b8cbdf4fd8b13a1-Abstract.html

[7] T. Koch, A. Martin, and S. Voß. 2000. *SteinLib: An Updated Library on Steiner Tree Problems in Graphs*. Technical Report. Konrad-Zuse-Zentrum für Informationstechnik Berlin ZIB-Report 00-37.

[8] Thorsten Koch, Alexander Martin, and Stefan Voß. 2001. *SteinLib: An Updated Library on Steiner Tree Problems in Graphs*. Springer US, Boston, MA, 285–325.

[9] Moyukh Laha and Raja Datta. 2023. A Steiner Tree based efficient network infrastructure design in 5G urban vehicular networks. *Computer Communications* 201 (2023), 59–71.

[10] Juho Lauri and Sourav Dutta. 2019. Fine-grained search space classification for hard enumeration variants of subset problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33.

[11] Juho Lauri, Sourav Dutta, Marco Grassia, and Deepak Ajwani. 2023. Learning fine-grained search space pruning and heuristics for combinatorial optimization. *Journal of Heuristics* 29, 2-3 (2023), 313–347.

[12] Ivana Ljubic. 2021. Solving Steiner trees: Recent advances, challenges, and perspectives. *Networks* 77, 2 (2021), 177–204.

[13] Tobias Polzin and Siavash Vahdati Daneshmand. 2003. On Steiner trees and minimum spanning trees in hypergraphs. *Operations Research Letters* 31, 1 (2003), 12–20.

[14] Daniel Rehfeldt and Thorsten Koch. 2022. On the Exact Solution of Prize-Collecting Steiner Tree Problems. *INFORMS J. Comput.* 34, 2 (2022), 872–889.

[15] Daniel Rehfeldt and Thorsten Koch. 2023. Implications, conflicts, and reductions for Steiner trees. *Math. Program.* 197, 2 (2023), 903–966.

[16] Dena Tayebi, Saurabh Ray, and Deepak Ajwani. 2022. Learning to Prune Instances of k-median and Related Problems. In *Proceedings of the ACM-SIAM symposium on algorithm engineering and experiments (ALENEX)*. 184–194.

[17] Huijuan Wang, Javier Martin Hernandez, and Piet Van Mieghem. 2008. Betweenness centrality in a weighted network. *Physical Review* 77, 4 (2008).

[18] Richard T. Wong. 1984. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical programming* 28 (1984), 271–287. Issue 3.

[19] Zong Yan, Haizhou Du, Jiahao Zhang, and Guoqing Li. 2021. Cherrypick: Solving the Steiner Tree Problem in Graphs using Deep Reinforcement Learning. In *IEEE 16th Conference on Industrial Electronics and Applications (ICIEA)*. 35–40.

[20] Mohammed J. Zaki and Wagner Meira Jr. 2014. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press. http://www.dataminingbook.info/

Utilizing Graph Sparsification for Pre-processing in Max Cut QUBO Solver: Extended Abstract

Vorapong Suppakitpaisarn[1] and Jin-Kao Hao[2]

[1]The University of Tokyo, Tokyo, Japan, ✉ vorapong@is.s.u-tokyo.ac.jp
[2]University of Angers, Angers, France, ✉ jin-kao.hao@univ-angers.fr

The maximum cut problem (max cut) is one of the most well-known NP-hard problems [12] and has garnered interest among researchers [19, 14, 2]. There are numerous heuristic, approximation, and exact algorithms proposed to solve the problem. Prominent among these are the well-known SDP relaxation algorithm [7, 17] and algorithms for specific graph types [18, 8, 16, 23].

Several studies, including those by [5, 4, 15], have concentrated on pre-processing techniques for the problem. These methods aim to either decrease the size of the problem instance or present simpler instances for the max cut algorithms to process. By doing so, they potentially reduce the computational time required for solving the max cut problem.

In our studies, we explore pre-processing strategies for the max cut problem, particularly when addressed using quantum or quantum-inspired computing. Quantum(-inspired) computing is considered to have the potential to enhance the efficiency of solving various computational problems [24, 10, 6]. In particular, these types of computers are expected to provide more effective algorithms for tackling the quadratic unconstrained binary optimization (QUBO) [9, 21]. Because max cut is a problem that can be easily expressed within the QUBO framework [3, 22], it serves as an exemplary case to illustrate the effectiveness of quantum(-inspired) approaches in QUBO contexts.

Although minimizing computation time is important for solving the max cut problem, there is an additional challenge in addressing the problem with quantum(-inspired) QUBO solvers. Since quantum(-inspired) computers will not be commercially available for the next several decades, we are compelled to utilize these solvers through cloud services. This requires us to transmit our problems to the service providers, the step which often results in communication becoming a significant bottleneck [20, 13]. Therefore, our focus in this paper is a pre-processing that diminishes the costs associated with this communication.

The communication cost of the max cut problem is strongly related to the number of edges in the input graph. We therefore propose to use the graph sparsification technique by the effective resistance edge sampling [11, 1, 25] to reduce the communication cost. The effective resistance technique has been demonstrated to significantly reduce the number of edges in a graph while preserving the cut size [25]. Expanding on these ideas, we suggest using graph sparsification before submitting the graph to QUBO solvers, with the aim of obtaining large-sized cuts from the QUBO solvers.

References

[1] András A Benczúr and David R Karger. Approximating s-t minimum cuts in $O(n^2)$ time. In *STOC'96*, pages 47–55, 1996.

[2] Una Benlic and Jin-Kao Hao. Breakout local search for the max-cutproblem. *Engineering Applications of Artificial Intelligence*, 26(3):1162–1173, 2013.

[3] Iain Dunning, Swati Gupta, and John Silberholz. What works best when? a systematic evaluation of heuristics for max-cut and qubo. *INFORMS Journal on Computing*, 30(3):608–624, 2018.

[4] Damir Ferizovic. *A Practical Analysis of Kernelization Techniques for the Maximum Cut Problem*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2019.

[5] Damir Ferizovic, Demian Hespe, Sebastian Lamm, Matthias Mnich, Christian Schulz, and Darren Strash. Engineering kernelization for maximum cut. In *ALENEX'20*, pages 27–41, 2020.

[6] Sevag Gharibian and François Le Gall. Dequantizing the quantum singular value transformation: hardness and applications to quantum chemistry and the quantum pcp conjecture. In *STOC'22*, pages 19–32, 2022.

[7] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[8] Martin Grötschel and George L Nemhauser. A polynomial algorithm for the max-cut problem on graphs without long odd cycles. *Mathematical Programming*, 29(1):28–40, 1984.

[9] Stuart Harwood, Claudio Gambella, Dimitar Trenev, Andrea Simonetto, David Bernal, and Donny Greenberg. Formulating and solving routing problems on quantum computers. *IEEE Transactions on Quantum Engineering*, 2:1–17, 2021.

[10] Miguel Herrero-Collantes and Juan Carlos Garcia-Escartin. Quantum random number generators. *Reviews of Modern Physics*, 89(1):015004, 2017.

[11] David R Karger. Global min-cuts in rnc, and other ramifications of a simple min-cut algorithm. In *SODA'93*, pages 21–30, 1993.

[12] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010.

[13] Shuta Kikuchi, Nozomu Togawa, and Shu Tanaka. Dynamical process of a bit-width reduced ising model with simulated annealing. *arXiv preprint arXiv:2304.12796*, 2023.

[14] Robbie King. An improved approximation algorithm for quantum max-cut. *arXiv preprint arXiv:2209.02589*, 2022.

[15] Sebastian Lamm. *Scalable Graph Algorithms using Practically Efficient Data Reductions*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2022.

[16] Frauke Liers and Gregor Pardella. Partitioning planar graphs: a fast combinatorial approach for max-cut. *Computational Optimization and Applications*, 51(1):323–344, 2012.

[17] Sanjeev Mahajan and Hariharan Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM Journal on Computing*, 28(5):1641–1663, 1999.

[18] S Thomas McCormick, M Rammohan Rao, and Giovanni Rinaldi. Easy and difficult objective functions for max cut. *Mathematical Programming*, 94:459–466, 2003.

[19] Renee Mirka and David P Williamson. An experimental evaluation of semidefinite programming and spectral algorithms for max cut. *ACM Journal of Experimental Algorithmics*, 28:1–18, 2023.

[20] Daisuke Oku, Masashi Tawada, Shu Tanaka, and Nozomu Togawa. How to reduce the bit-width of an ising model by adding auxiliary spins. *IEEE Transactions on Computers*, 71(1):223–234, 2020.

[21] Hiroki Oshiyama and Masayuki Ohzeki. Benchmark of quantum-inspired heuristic solvers for quadratic unconstrained binary optimization. *Scientific reports*, 12(1):2146, 2022.

[22] Daniel Rehfeldt, Thorsten Koch, and Yuji Shinano. Faster exact solution of sparse maxcut and qubo problems. *Mathematical Programming Computation*, pages 1–26, 2023.

[23] Wei-Kuan Shih, Sun Wu, and Yue-Sun Kuo. Unifying maximum cut and minimum cut of a planar graph. *IEEE Transactions on Computers*, 39(5):694–697, 1990.

[24] Peter W Shor. Introduction to quantum algorithms. In *Proceedings of Symposia in Applied Mathematics*, volume 58, pages 143–160, 2002.

[25] Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *STOC'08*, pages 563–568, 2008.

Session Session 3B: Network Optimization
Tuesday 12 March 2024, 11:00-12:30
Q012

Optimizing k-level facility location problem:a bipartite boolean quadratic programming model solved by tabu Search with random-key sequence

Bahram Alidaee[1], Haibo Wang[2], Jun Huang[3], and Lutfu S. Sua[4]

[1]School of Business Administration, University of Mississippi, University, MS 38677, United States, ✉ balidaee@bus.olemiss.edu
[2]A. R. Sánchez Jr. School of Business, Texas A&M International University, Laredo, TX 78041, United States, ✉ hwang@tamiu.edu
[3]Department of Management and Marketing, Angelo State University,San Angelo, TX 78041, United States, ✉ jun.huang@angelo.edu
[4]Department of Management and Marketing, Southern University and A&M College, Baton Rouge, LA, USA, ✉ lutfu.sagbansua@subr.edu

The uncapacitated facility location problem at the k-level involves ensuring that each client is serviced by a sequence of k different facilities. This scenario is commonly encountered in complex logistical systems, where central depots distribute goods through smaller depots to the clients. Given its inherent complexity, particularly when k $>$ 1, the use of approximation algorithms becomes crucial for swiftly obtaining high-quality feasible solutions [2, 4, 6, 3].

This study specifically addresses a four-level facility location problem encompassing plants (P), warehouses (W), distribution centers (D), and retail stores (S). In this context, the locations of plants and retail stores are already known, and the objective is to select suitable facilities for warehouses and distribution centers. The formulation of the four-level facility location problem is approached through a bipartite boolean quadratic programming (BBQP) model, and its solution is derived using a heuristic technique involving Tabu search with random-key sequence embedded as diversification. To provide clarity, the notations used in the formulation are presented next:

Input parameters

P	Set of potential plant locations with, indexed by n
W	Set of potential warehouse locations, by i
D	Set of potential distribution center locations, indexed by j
S	Set of potential retail stores (clients), indexed by m
$G=(V,A)$	is a graph with nodes V and directed arcs A
f_{mab}	Cost of moving a bundle of products for retail store$m$ along an arc (a,b) , Note: for $a=j$ and $b=m$, we can replace f_{mab} with f_{jm}
fw_i	Fixed cost of opening a warehouse in W
fd_j	Fixed cost of opening a distribution center in D
uw	Upper bound for number of warehouses to be opened
ud	Upper bound for number of distribution centers to be opened

Decision Variables

w_i	is 1 if warehouse i is opened, 0 otherwise
D_j	is 1 if distribution j is opened, 0 otherwise
x_{mni}	is 1 if an opened plant n delivers a truckload for retail store m to an open warehouse i, and 0 otherwise
y_{jm}	is 1 if an opened distribution center j delivers a truckload to an opened retail store m, and 0 otherwise

The BBP model for the objective of a four-level facility location problems to minimize the total cost

TC is:

$$\text{Minimize TC}= \sum_{i\in W}\sum_{n\in P}\sum_{m\in S}\sum_{j\in D} f_{mij}x_{mni}y_{jm} + \sum_{i\in W} fw_i w_i + \sum_{j\in D} fd_j d_j + \sum_{i\in W}\sum_{n\in P}\sum_{m\in S} f_{mni}x_{mni} + \sum_{m\in S}\sum_{j\in D} f_{jm}y_{jm} \quad (1)$$

$$\text{subject to} \quad \sum_{n\in P}\sum_{i\in W} x_{mni} = 1, \quad \forall m \in S \quad (2)$$

$$\sum_{j\in D} y_{jm} = 1, \quad \forall m \in S \quad (3)$$

$$\sum_{i\in W} w_i \leq uw \quad (4)$$

$$x_{mni} \leq w_i, \quad \forall n \in P, m \in S, i \in W \quad (5)$$

$$\sum_{j\in D} d_j \leq ud \quad (6)$$

$$y_{jm} \leq d_j, \quad \forall m \in S, j \in D \quad (7)$$

$$x_{mni}, y_{jm}, w_i, d_j \quad in\{0,1\}, \quad \forall n \in P, m \in S, i \in W, j \in D \quad (8)$$

Expanding on the research presented in [5, 1], a heuristic is implemented to address the BBQP model, utilizing the Tabu search with random-key sequence embedded as the diversification strategy. The diversification procedures outlined in this paper are contingent on the choice of the next improvement step. While existing diversification approaches in the literature involve various manipulations of the solution x, the initial demonstration of the effectiveness of improvement steps as a diversification approach for general UBQP was provided. The implementation of improvement processes can yield a substantial number of potential diversification approaches by employing different sequences. In this study, we adopt several methods to generate diverse sets of orders for the implementation of improvement procedures. To assess the heuristic's efficacy, the BBQP model is also solved using the Gurobi 10.0 QP solver across a range of benchmark instances provided in [4]. These instances, featuring four levels, consist of randomly generated scenarios with larger numbers of customers and potential facilities. The datasets encompass between 500 and 2,000 customers and between 100 and 200 potential facilities. Furthermore, a comparative analysis is conducted on four Mixed-Integer Linear Programming (MILP) formulations equivalent to the BBQP model using the Gurobi 10.0 LP solver. The proposed heuristic and the Gurobi 10.0 solvers were executed sequentially on a single core of an Intel Xeon Quad-core E5420 Harpertown processor, featuring a 2.5 GHz CPU with 8 GB memory. The initial findings will be reported at the conference.

References

[1] Bahram Alidaee and Haibo Wang. A note on heuristic approach based on ubqp formulation of the maximum diversity problem. *Journal of the Operational Research Society*, 68(1):102–110, 2017.

[2] Ningxuan Kang, Hao Shen, and Ye Xu. Jd.com improves delivery networks by a multiperiod facility location model. *INFORMS Journal on Applied Analytics*, 52(2):133–148, 2021.

[3] Camilo Ortiz-Astorquiza, Ivan Contreras, and Gilbert Laporte. Multi-level facility location as the maximization of a submodular set function. *European Journal of Operational Research*, 247(3):1013–1016, 2015.

[4] Camilo Ortiz-Astorquiza, Ivan Contreras, and Gilbert Laporte. Formulations and approximation algorithms for multilevel uncapacitated facility location. *INFORMS Journal on Computing*, 29(4):767–779, 2017.

[5] Haibo Wang and Bahram Alidaee. A new hybrid-heuristic for large-scale combinatorial optimization: A case of quadratic assignment problem. *Computers & Industrial Engineering*, 179:109220, 2023.

[6] Limin Wang, Zhao Zhang, Chenchen Wu, Dachuan Xu, and Xiaoyan Zhang. Approximation algorithms for the dynamic k-level facility location problems. *Theoretical Computer Science*, 853:43–56, 2021.

Bayesian Optimisation for Facility Location Problems

Niyati Seth[1] and Michael Fop[2]

[1]School of Mathematics and Statistics, University College Dublin, Dublin, Ireland, ✉ niyati.seth@ucdconnect.ie
[2]School of Mathematics and Statistics, University College Dublin, Dublin, Ireland, ✉ michael.fop@ucd.ie

1 Introduction

Facility location is a spatial optimisation problem concerning the identification of the optimal location of facilities under certain constraints. Typically, the aim is to find the locations of facilities so that demand on every node on a grid is fulfilled, and the maximum population is covered in the most cost-effective manner. The decisions on where to locate facilities often require estimating different parameters associated with the problem, for example costs, demand, distances, etc. These parameters are uncertain and may fluctuate with time and circumstances. Based on this, recent research has been aimed towards developing facility location models under uncertainty.

In uncertain environment problems, the parameters are variable with, usually unknown, probability distributions [8]. In the recent body of literature, a number of approaches have been proposed to incorporate uncertainty in the location problem at various levels. For example, in [2], uncertainty is incorporated via assumptions about the demand distributions and their relation to the location decisions. However, these distributions are not informed by observed data. In [5], a hybrid approach is presented, where uncertainty in the data and locations are accounted for using a Bayesian spatial interaction model. Nonetheless, this uncertainty is not directly incorporated within the optimisation procedure, which considers the objective functions to be deterministic.

Following and extending this branch of literature, our research aims at developing facility location procedures which account for uncertainty in the model and in the data. To do so, we extend Bayesian combinatorial optimisation methods [1][3] to optimal facility location problems. The motivating application of our research involves the design of hybrid wind-photovoltaic electrification systems [4], with the aim of accounting for uncertainty associated with time, weather, energy supply and demand.

2 Bayesian optimisation

Bayesian optimisation is used to solve optimisation problems when the objective function is expensive to evaluate. The problem of globally minimising an objective function f in a search space $\mathcal{X}$ can be formulated as $\arg\min_{\mathbf{x}} f(\mathbf{x})$. To facilitate the optimisation process, a surrogate function representing $f(\mathbf{x})$ is defined. To guide the search, Bayesian optimisation employs an acquisition function, $a(\mathbf{x})$ that strategically selects the next input point, $\mathbf{x}^{(t)}$ to evaluate. Upon selecting $\mathbf{x}^{(t)}$, the algorithm proceeds to evaluate the true function, $\mathbf{x}^{(t)}$). This evaluation result $\mathbf{x}^{(t)}, f(\mathbf{x}^{(t)})$, is integrated into the new dataset. This new updated data forms the training set for the next iteration of the optimisation procedure. [7][6].

3 Bayesian optimisation for Facility Location

The objective function of our problem is the simplified version of the model proposed in [4] and is defined at the cost of installing solar panels on a grid.

$$\underset{\mathbf{x}}{\text{minimise}} \quad f(\mathbf{x}) = \sum_{i=1}^{n} x_i g(C_i, W_i) \tag{1}$$

where, $f(\mathbf{x})$ is a generalised objective function, $\mathbf{x}$ is a discrete decision variable to install panels, $\mathbf{x} \in \{0,1\}^n$, g is a function that describes the relationship between various parameters, C_i is the cost

of installing a panel at location i, $\mathbf{W}_i$ is a generic vector of variables, associated with weather, energy supply and demand, grid layout, etc. The problem presented (1), is subject to constraints related to energy supply and demand, budget, coverage, choice of equipment. The parameters and variables need to be estimated themselves on the basis of available data and are subject to uncertainty, hence making the objective function expensive to evaluate.

Using a Bayesian optimisation approach for combinatorial optimisation over discrete search spaces (BOCS) [1], we present a method to define a surrogate for (1). The surrogate function $f_\alpha(\mathbf{x}) = \alpha_0 + \sum_j \alpha_j x_j + \sum_{i,j \geq i} \alpha_{ij} x_i x_j$ is a linear function in $\alpha = (\alpha_i, \alpha_{ij}) \in \mathbb{R}^n$ and quadratic in the binary decision variable $\mathbf{x}$. To guide the search the acquisition function, $a(\mathbf{x}) = \arg\min_{\mathbf{x}} f_\alpha(\mathbf{x}) + \lambda P(\mathbf{x})$, based on Thompson Sampling is employed with the addition of a penalty.

Our research presents a novel approach by extending the BOCS method to account for constraints, uncertainty, grid structure and interaction of the allocation nodes. We extend the general algorithm described in [1], with the surrogate model remaining the same linear function and modifying the acquisition function to include the problem constraints.

Incorporating constraints within a combinatorial domain for a facility location problem is challenging within the framework of BOCS. Hence, we extend this framework by introducing a probabilistic reparameterisation [3] for the decision variable $\mathbf{x}$ as $p(\mathbf{x} \mid \boldsymbol{\theta})$. This distribution is parameterized by a vector of continuous parameters $\boldsymbol{\theta}$. Given this reparametisation then, the acquisition function is redefined as the expected value $E_{\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})}[f_\alpha(\mathbf{x})]$. This acquisition function is then optimised over $\boldsymbol{\theta}$, to sample $\mathbf{x} \sim p(\mathbf{x} \mid \hat{\boldsymbol{\theta}})$ as the next evaluation point. This allows to define optimisation of the acquisition function on a continuous space.

By employing Bayesian optimisation we incorporate uncertainty in the optimisation procedure itself, while allowing for an integrated framework in which the estimation of optimisation variables is data driven. We demonstrate the performance of our method in comparison to other state of the art algorithms.

References

[1] Ricardo Baptista and Matthias Poloczek. Bayesian Optimization of Combinatorial Structures. In *Proceedings of the 35th International Conference on Machine Learning*, pages 462–471. PMLR, July 2018.

[2] Beste Basciftci, Shabbir Ahmed, and Siqian Shen. Distributionally robust facility location problem under decision-dependent stochastic demand. *European Journal of Operational Research*, 292(2):548–561, July 2021.

[3] Samuel Daulton, Xingchen Wan, David Eriksson, Maximilian Balandat, Michael A Osborne, and Eytan Bakshy. Bayesian Optimization over Discrete and Mixed Spaces via Probabilistic Reparameterization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 12760–12774. Curran Associates, Inc., 2022.

[4] L. Ferrer-Martí, B. Domenech, A. García-Villoria, and R. Pastor. A MILP model to design hybrid wind–photovoltaic isolated rural electrification projects in developing countries. *European Journal of Operational Research*, 226(2):293–300, April 2013.

[5] Shanaka Perera, Virginia Aglietti, and Theodoros Damoulas. On the competitive facility location problem with a Bayesian spatial interaction model. *Journal of the Royal Statistical Society Series C: Applied Statistics*, page qlad003, February 2023.

[6] Tony Pourmohamad and Herbert K. H. Lee. *Bayesian Optimization with Application to Computer Experiments*. SpringerBriefs in Statistics. Springer International Publishing, Cham, 2021.

[7] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148–175, January 2016.

[8] Lawrence V. Snyder. Facility location under uncertainty: a review. *IIE Transactions*, 38(7):547–564, June 2006.

Optimizing Charging Station Locations for Electric Vehicles: Catering to Diverse Driver Profiles

Jingyu Xiang[1], Paula Carroll[2], and Annunziata Esposito Amideo[3]

[1]UCD College of Business, University College Dublin, Dublin, IRELAND, ✉ jingyu.xiang@ucdconnect.ie
[2]UCD College of Business, University College Dublin, Dublin, IRELAND, ✉ paula.carroll@ucd.ie
[3]UCD College of Business, University College Dublin, Dublin, IRELAND, ✉ annunziata.espositoamideo@ucd.ie

As the electric vehicle (EV) market continues to expand, the efficient deployment of charging facilities has become imperative to support the increasing charging needs of electric vehicles. The charging station location problem (CSLP) arises to optimize the locations of charging stations, thereby refueling EVs to reach their destinations. This paper presents an integrated approach for the CSLP combining the Multi-Path Refueling Location Model (MPRLM) with considerations of different EV driver behaviours in relation to a variety of charging options and routing decisions. The study aims to ensure that the placement of charging stations aligns with the diverse charging needs and decision-making patterns of private EV drivers.

By incorporating deviation paths into a flow-based set-covering model, Huang, Li and Qian (2015) developed a novel CSLP model called the Multipath Refueling Location Model (MPRLM) to locate refueling stations at nodes within the network. The authors considered multiple potential routes between origin-destination pairs for users of alternative fuel vehicles (AFVs), aiming to minimizing overall system costs while satisfying all the charging needs of AFV drivers [3]. In this study, we first expand upon the MPRLM initially designed for alternative fueling stations, adapting it to different charging stations for EVs, including slow-charging, fast-charging, and battery-swapping stations. Every node in the network is permitted to have various types of charging stations installed. This extended MPRLM considers different charging rates of multiple charging stations, assuming a linear recharging rate over time. Our objective is to strategically position charging stations, minimizing the total cost while balancing EV user satisfaction and ensuring all EV users can reach their destinations efficiently. This involves two key objectives: firstly, minimizing the total installation and operational costs across various charging station types; and secondly, reducing the overall charging time for EV users. To prevent the unbalanced employment of most affordable charging stations, our MRPLM introduces a capacity constraint for each type of charging unit, which denotes the maximum number of various charging units that can be installed at each node.

To demonstrate the model's efficacy, we test our extended MPRLM and the original MPRLM without the integration of diverse charging options on the 25-node highway network. The 25-node network was first provided for traveling salesman location problems [4] and then adapted to be used in charging station location problems [2]. By conducting a comparative analysis of the outcomes, the results show improvements in balancing the overall system costs and total charging time of EV users, as well as optimizing charging station utilization when diverse charging options are taken into account.

Then, we continue to refine the extended MPRLM by incorporating the different charging profiles of EV users. Under the battery leasing/electric car sharing service business models, Yang et al. [5] identified four distinct customer types, each characterized by specific thresholds for range anxiety and loss anxiety. Considering "range anxiety" and "loss anxiety", the users' satisfaction function was developed for three stages according to the battery's remaining capacity, the range anxiety threshold, and the loss anxiety threshold. Furthermore, Guo et al. [1] introduced a range anxiety function and evenly divided the customers in each origin-destination (O-D) pair into two distinct groups with different thresholds of range anxiety. In our study, we categorize private EV drivers based on their sensitivity levels to range anxiety, charging costs and charging time, which encompasses high sensitive, range-anxious, cost-sensitive, time-sensitive and low sensitive profiles. For each O-D pair journey, we assume an equal number of private EV users from each of these groups.

Subsequently, we incorporate these distinct satisfaction functions for these varied diverse EV drivers into the expanded MPRLM which now includes a range of charging options. To incorporate personal be-

havioral insights, we develop satisfaction functions of different driver profiles and establish the satisfaction thresholds for each EV driver category, which is informed by the review of existing literature on the CSLP for EVs with the consideration of diverse driver types [1][5]. In this Integrated Behavioral Multipath Refueling Location Model (IB-MPRLM) , the charging station located at a certain node can only refuel an EV when the user's satisfaction is higher than the threshold of the EV user. The satisfaction function for various EV drivers during their journey is assumed to relate with the deviation from the shortest path, the type of charging stations and the battery's remaining capacity. EV users sensitive to range anxiety exhibit a more substantial requirement for the remaining fuel capacity, demanding more reliable and accessible charging options to alleviate their concerns. On the other hand, EV users identified as having low sensitivity to range anxiety are characterized by a lower threshold for satisfaction, indicating they are more easily satisfied with available charging facilities during their origin-destination (O-D) pair journeys. Additionally, the users highly sensitive to charging time tend to prefer routes that minimally deviate from the shortest path to a charging station and the fast-charging options, prioritizing the total traveling and charging time. Furthermore, cost-Sensitive EV users are more likely to be satisfied and get refueled by the slow-charging station than fast-charging stations at the same location when the remaining state of charge is relatively high, even if it entails longer charging time. By incorporating diverse profiles of EV drivers into the model, we aim to ensure that the selected charging station locations not only meet the charging demands of EVs but also align with the convenience and accessibility expectations of various drivers.

Our forthcoming step involves evaluating the IB-MPRLM on the established 25-node network [2][4] to validate the model's ability to handle the complexity and provide an efficient solution for the CSLP. Through sensitivity analysis which examines different distribution scenarios of diverse driver profiles, our objective is to understand the impact of diverse driver profiles and their charging preferences on the optimal siting of EV charging stations.

In conclusion, our paper introduces a novel approach to the CSLP for EVs, bridging the gap between technical optimization models and various driver profiles, to facilitate the transition towards sustainable urban mobility. Our findings validate the proficiency of the first extended MPRLM in accommodating various charging options and we aim to further test the enhanced IB-MPRLM's ability to cater to the heterogeneous driving public.

References

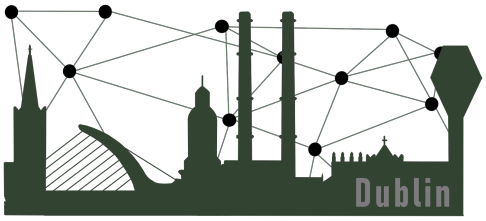
[1] Fang Guo, Jun Yang, and Jianyi Lu. The battery charging station location problem: Impact of users' range anxiety and distance convenience. *Transportation Research Part E: Logistics and Transportation Review*, 114:1–18, 2018.

[2] M. John Hodgson. A flow-capturing location-allocation model. *Geographical Analysis*, 22(3):270–279, 1990.

[3] Yongxi Huang, Shengyin Li, and Sean Qian. Optimal deployment of alternative fueling stations on transportation networks considering deviation paths. *Networks and Spatial Economics*, 15, 03 2015.

[4] David Simchi-Levi and Oded Berman. A heuristic algorithm for the traveling salesman location problem on networks. *Operations research*, 36(3):478–484, 1988.

[5] Jun Yang, Fang Guo, and Min Zhang. Optimal planning of swapping/charging station network with customer satisfaction. *Transportation Research Part E: Logistics and Transportation Review*, 103:174–197, 2017.



INOC - 2024

Session Session 3C: Exact Approaches

Tuesday 12 March 2024, 11:00-12:30

Q013

Cutting-plane algorithms for the stochastic diversion path problem

J. Cole Smith[1], N. Orkun Baycik[2], and Di Nguyen[3]

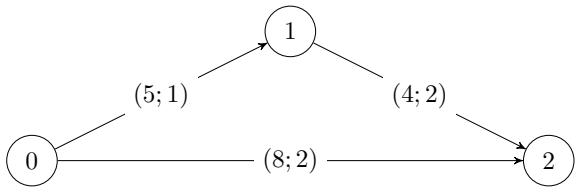
[1]Dept. of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY, USA, ✉ colesmit@syr.edu
[2]Questrom School of Business, Boston University, Boston, MA, USA, ✉ nobaycik@bu.edu
[3]School of Mechanical and Materials Engineering, University College Dublin, Dublin, Ireland, ✉ di.nguyen@ucd.ie

This talk examines a stochastic network optimization problem in which we modify arc lengths in order to guarantee that a specified path within the network will be optimal with sufficiently high probability. Modifying the arc lengths (by increasing or decreasing them from their original values) incurs a per-unit nonnegative modification cost.

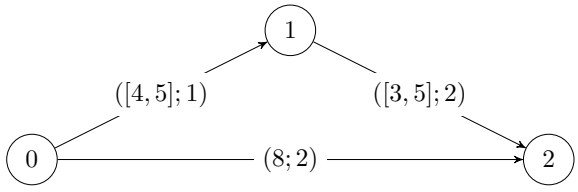
Our problem data involves a directed network G with nodes N and arcs A and a given *diversion path* (DP) starting at a source node $s \in N$ and ending in a sink node $t \in N$. For each arc $(i, j) \in A$, we let c_{ij} be the *arc length* for (i, j), and we let m_{ij} denote the *modification cost* for (i, j). In the deterministic version of this problem, all arc lengths are deterministic (and nonnegative), and hence we seek to minimize the sum of modification costs required to ensure that the diversion path is optimal. Moreover, the diversion path must be better than all other solutions by at least some margin, which is given by parameter δ. This criterion is known as the "δ condition." We call this problem the Diversion Path Problem (DPP).

In the stochastic version, each arc length is an independent, uniformly-distributed random variable. (The lower bound on the arc lengths is nonnegative.) Given a parameter $0 < \tau \leq 1$, the goal is to minimize the sum of modification costs required to guarantee that the DP satisfies the delta condition with probability at least τ. This problem is called the Chance-Constrained Diversion Path Problem (CC-DPP), and this problem is the subject of our talk.

A DPP instance is depicted in Figure 1(a) with source node 0 and sink node 2. There are just two paths in this network: $0 \to 2$, whose length is 8 units, and $0 \to 1 \to 2$, which is the diversion path and has a length of 9 units. In the examples of Figure 1, we set $\delta = 1$.



(a) Deterministic case



(b) Stochastic case

Figure 1: Small networks to illustrate DPP. The diversion path is $0 \to 1 \to 2$ and $\delta = 1$. Notation $(c_{ij}; m_{ij})$ on arcs represents the length and modification cost for arc (i, j). The lower and upper bounds for arc lengths are represented as $[c^L_{ij}, c^U_{ij}]$ in the stochastic case, respectively.

For the DPP, we can either increase the length of arc $(0, 2)$ by two units, or decrease the total length of arcs $(0, 1)$ and $(1, 2)$ by the same amount (or a combination thereof). Because the modification costs

are given as $m_{01} = 1$ and $m_{02} = m_{12} = 2$, an optimal solution to the DPP reduces the length of arc $(0, 1)$ by two units.

The CC-DPP is depicted in Figure 1(b), where the lengths of arcs $(0, 1)$ and $(1, 2)$ are now uniformly-distributed random variables within ranges $[4, 5]$ and $[3, 5]$, respectively. The length of path $0 \to 1 \to 2$ now follows a trapezoidal distribution between 7 units and 10 units. The cost of path $0 \to 2$ is 8 units, meaning that with $\delta = 1$, at least $\tau\%$ of the DP's trapezoidal distribution must be less than or equal to 7. Again, it is optimal to modify the length of arc $(0, 1)$. Decreasing the length of $(0, 1)$ by d units, so that its length is now distributed between $[4 - d, 5 - d]$, shifts the range of the trapezoidal distribution for the DP's cost to $[7 - d, 10 - d]$. In particular, the cumulative distribution function $g(u|d)$ of the DP's cost is given by

$$g(u|d) = \begin{cases} (1/4)(u - (7 - d))^2 \text{ for } 7 - d \leq u \leq 8 - d \\ (1/4) + (1/2)(u - (8 - d)) \text{ for } 8 - d \leq u \leq 9 - d \\ 1 - (1/4)((10 - d) - u)^2 \text{ for } 9 - d \leq u \leq 10 - d. \end{cases}$$

If, for instance, $\tau = 3/4$, then setting $d = 2$ turns out to be optimal (as $g(u|2) = 3/4$).

The DPP might represent a case in which the optimizing agent who modifies the arc lengths may be in conflict with the agent who travels in the network, as is the case in most diversion optimization problems. The diversion path may be preferred by the leader (i.e., the agent modifying the arc lengths) if the leader can best monitor and control those arcs. The follower (who traverses the path) may conduct illegal activities, communicate with hostile entities, or attack critical infrastructure somewhere along the path. If the follower deviates from this path and engages in these activities on some arc outside the diversion path, the leader would be unable to arrest the follower, intercept the follower's communications, or defend against the follower's disruptions as the case arises. The DPP is a version of the network diversion problem (NDP). One early NDP study seeks to remove links and nodes so as to route the flow through a specific set of arcs [4]. Cullenbine et al. [3] propose a polynomial-time algorithm to NDP as well as an improved mixed-integer programming formulation. Lee et al. [5] extend the network diversion problem to a multiple flows network diversion problem in which there exist multiple source-sink pairs on a given network.

The approach we take in our study is to find "induced sets" as a function of modification decisions. The induced set is the set of all data points in the uncertainty set, which itself is a hyperrectangle formed based on the uniform random variable distributions, shifted by arc cost modification decisions. (Many of these concepts also arise in inverse optimization, see, e.g., [1, 2].) The area of the induced set is thus the percentage of the uncertainty set for which the DP satisfies the δ condition. Our approach constructs restrictions and relaxations of the induced set, which can be used to bound the probability that DP satisfies the δ condition for a proposed set of modification decisions. We then provide a method for optimizing the modification decisions based on those bounds, and for then refining those bounds to provide convergence to an optimal solution for the CC-DPP.

References

[1] R. K. Ahuja and J. B. Orlin. Inverse optimization. *Operations Research*, 49(5):771–783, 2001.

[2] D. Burton and P. L. Toint. On an instance of the inverse shortest paths problem. *Mathematical Programming*, 53(1):45–61, 1992.

[3] C. Cullenbine, R. K. Wood, and A. M. Newman. Theoretical and computational advances for network diversion. *Networks*, 62(3):225–242, 2013.

[4] N. D. Curet. The network diversion problem. *Military Operations Research*, 6(2):35–44, 2001.

[5] C. Lee, D. Cho, and S. Park. A combinatorial Benders decomposition algorithm for the directed multiflow network diversion problem. *Military Operations Research*, 24(1):23–40, 2019.

A cutting-plane-based method for solving fixed-charge transportation problems using new valid inequalities for single-node flow polytope

Guneshwar Anand[1], Sachin Jayaswal[2], and B. Srirangacharyulu[3]

[1]Production and Operations Management, IIM Visakhapatnam, India, , ✉ guneshwar.anand-phd19@iimv.ac.in
[2]Operations and Decision Sciences, IIM Ahmedabad, India , ✉ sachin@iima.ac.in
[3]Decision Sciences, IIM Visakhapatnam, India, ✉ sriranga@iimv.ac.in

1 Introduction

The fixed charge transportation problem (FCTP) is a generalization of the well-known transportation problem, which includes a fixed cost of transportation between any source and destination, in addition to the variable cost per unit of transportation. However, while the transportation problem is polynomially solvable, FCTP is known to be NP-hard. There have been a few studies on solving FCTP [1] more efficiently using the current mixed-integer linear programme (MILP) solvers, but even the state-of-the-art method struggles to solve instances of even medium-size. In this paper, we discuss a cutting-plane-based solution approach to solve FCTP more efficiently. For this, we exploit the single-node fixed-charge flow polytope as a relaxation of the FCTP polytope, for which we propose two new classes of valid inequalities (VIs) based on generalizations of the well-known flow cover inequalities (FCIs) proposed by [3]. We further provide conditions under which our proposed VIs define facets for the single-node fixed-charge flow polytope. We show the effectiveness of the proposed VIs in efficiently solving the instances of FCTP from the benchmark data sets.

The single-node flow set is defined as follows [2]: $S = P \cap \mathbb{R}^n \times \mathbb{B}^n$, where

$$P = \left\{ (x, y) \in \mathbb{R}^n \times \mathbb{R}^n : \sum_{i \in N} x_i \leq b,\ x_i \leq a_i y_i,\ x \geq 0,\ 0 \leq y_i \leq 1\ \forall i \in N = \{1, \ldots, n\} \right\},$$

where $0 < a_i \leq b,\ \forall i \in N$. a_i represents the capacity of arc $i\ \forall i \in N$ entering a single node, while b is the capacity of a single arc going out.

2 Flow Cover-based Inequalities

A set $C \subseteq N$ is a flow cover of S if $\sum_{i \in C} a_i > b$. $\lambda = \sum_{i \in C} a_i - b$ is called the flow cover surplus. Then, the following is well well-known flow cover inequality (FCI):

$$\sum_{i \in C} x_i \leq b - \sum_{i \in C} (a_i - \lambda)^+ (1 - y_i) \tag{FCI}$$

In the following, we introduce two new classes of VIs for conv(S). The first class of VIs is a generalization of the well-known FCI using the idea of partitioning a flow cover. Hence, we refer to the resulting VIs as partition-based flow cover inequality (PBFCI). Our second class of VIs are specifically tailored for flow covers with very high surplus, i.e., $C : \lambda \geq \max_{i \in C}\{a_i\}$.

2.1 Partition-based Flow Cover Inequality

Given a flow cover C, let $C_1 = \{i \in C : \sum_i a_i < b\}$, $A_1 = \sum_{i \in C_1} a_i$, $C_2 = \{i \in C \setminus C_1 : a_i + A_1 > b\}$, $\lambda' = \sum_{i \in C_2}(a_i - (b - A_1)) = \lambda - (|C_2| - 1)(b - A_1)$, $b' = b + (|C_2| - 1)(b - A_1)$, $a_1 = \min_{i \in C}\{a_i\}$, $a_2 = \min_{i \in C : a_i > a_1}\{a_i\}$, $a_m = \max_{i \in C}\{a_i\}$.

Proposition 1. *Given a flow cover C and its partition C_1, C_2, the following is a VI for conv(S):*

$$\sum_{i \in C} x_i \leq b' - \sum_{i \in C_1} (a_i - \lambda')^+ (1 - y_i) - \sum_{i \in C_2} (b - A_1)(1 - y_i) \tag{PBFCI}$$

Proposition 2. *For a given flow cover C, let I be the singleton set of* (FCI)*, while I^{PB} be the set of* (PBFCI) *corresponding to all possible partitions C_1, C_2 of C. Then, $I \subseteq I^{PB}$ $\forall C \subset N$.*

Proposition 3. *Given a flow cover C and its partition C_1, C_2,* (PBFCI) *separates the following set of fractional points $(\bar{x}, \bar{y}) \in P$ from conv(S):*

(a) when $\lambda \leq a_1 : \bar{x}_i = a_i, \bar{y}_i = 1$ $\forall i \in C \setminus \{j\}, \bar{x}_j = a_j - \lambda, \bar{y}_j = \frac{(a_j - \lambda)}{a_j}$;; $\bar{x}_i = 0, \bar{y}_i = 0$ $\forall i \in N \setminus C$

(b) when $a_1 < \lambda \leq a_m$:

(i) $\bar{x}_i = a_i, \bar{y}_i = 1$ $\forall i \in C_1; \bar{x}_j = b - A_1, \bar{y}_j = \frac{b - A_1}{a_j}$ for each $j \in C_2; \bar{x}_i = \bar{y}_i = 0$ $\forall i \in (C_2 \setminus j) \cup (N \setminus C)$

(ii) $\exists j \in C : a_j > \lambda : \bar{x}_j = a_j - \lambda, \bar{y}_j = \frac{(a_j - \lambda)}{a_j}; \bar{x}_i = a_i, \bar{y}_i = 1$ $\forall i \in C \setminus \{j\}$; $\bar{x}_i = 0, \bar{y}_i = 0$ $\forall i \in N \setminus C$

Theorem 1. *Given a flow cover C,* (PBFCI) *defines a facet of conv(S) if $\lambda < \max_{i \in C}\{a_i\}$ when $|C_2| = 1$; $\lambda \leq \max_{i \in C}\{a_i\}$ when $|C_2| > 1$.*

2.2 Heavyweight Flow Cover Inequality

In this section, we focus on those flow covers $C : \lambda \geq a_m$. Under such conditions, (FCI) becomes trivial, while (PBFCI), although non-trivial, fails to generate facets of conv(S). We now propose a new class of VIs that can generate facets of conv(S) under such conditions. We refer to our new class of VIs as heavyweight FCI since it is based on flow covers that contain an item such that the sum of its weight with any other item in C exceeds the capacity outgoing arc (b), i.e., $i \in C : a_i + a_m > b$. We call such an item a heavyweight.

Proposition 4. *For a given flow cover C,*

(a) the following is a set of VIs of conv(S).

$$\sum_{i \in C} x_i - \sum_{i \in C \setminus \{j\}} \min\{(b - a_j), a_i\} y_i - \min\{(b - a_m), a_j\} y_j \leq (a_j - (b - a_m))^+ \quad \forall j \in C \qquad \text{(HWFCI)}$$

(b) at least one of the VIs in (HWFCI) *corresponding to $j \in C : a_j + a_m > b$ is non-trivial.*

Proposition 5. *Given a single-node flow set $S : a_i > b/2$ $\forall i \in N$ and its flow cover C, let I be the singleton set of* (FCI)*, while I^{HW} be the set of* (HWFCI) *corresponding to $\forall j \in C$. Then, $I \subseteq I^{HW}$ $\forall C \subseteq N$.*

Theorem 2. *Given a flow cover C,*

(a) if $a_i \geq b/2$ $\forall i \in C$ and $\exists k \in C : a_k > b/2$, then at least one (HWFCI) *defines a facet of conv(S).*

(b) if $a_i > b/2$ $\forall i \in C$, then (HWFCI) *defines a facet of conv(S) $\forall j \in C$.*

3 Conclusions

We presented a cutting plane-based solution method for FCTP based on two new classes of VIs for the single-node fixed-charge flow polytope, which appears as a relaxation of the FCTP polytope. Further, we derived the conditions under which our proposed VIs are facet-defining for the single-node fixed-charge flow polytope. Our limited computational results demonstrate the efficacy of our proposed cutting-plane method, which effectively reduces the computation time for solving instances of FCTP.

References

[1] Yogesh Agarwal and Yash Aneja. Fixed-charge transportation problem: Facets of the projection polyhedron. *Operations research*, 60(3):638–654, 2012.

[2] Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli, et al. *Integer programming*, volume 271. Springer, 2014.

[3] Manfred W Padberg, Tony J Van Roy, and Laurence A Wolsey. Valid linear inequalities for fixed charge problems. *Operations Research*, 33(4):842–861, 1985.

Valid Inequalities to Solve the Train Stop Scheduling Problem

Faiz Hamid[1] and Yogesh K Agarwal[2]

[1]Department of Management Sciences, Indian Institute of Technology Kanpur, Kanpur, India, ✉ fhamid@iitk.ac.in
[2]Decision Sciences Group, Jaipuria Institute of Management, Lucknow, India, ✉ yogesh.agarwal@jaipuria.ac.in

We consider the problem of minimizing the number stops for a set of trains T operating on a railway line (origin and destination stations specified) without branches. The objective is to assign passengers of each origin-destination (OD) pair to different trains in such a way that train capacity and passenger demand constraints are satisfied with minimal stoppages. The literature refers to this problem as the train stop scheduling problem (TSSP) [1, 2]. The problem has been extensively studied for decades, yet an exact solution approach has not been proposed. Several valid inequalities have been proposed in [3] to strengthen the mixed-integer programming formulation and solve the problem exactly in a reasonable amount of CPU time. This conference paper presents the study carried out in [3].

To formulate the mixed integer linear programming (MILP) model, we introduce the undirected track graph $G = (N, E)$, consisting of a finite set of stations (vertices) N and a finite set of sections (edges) E connecting adjacent stations. The stations are sequentially numbered $1, 2, \ldots, n$, with stations 1 and n denoting the origin and destination stations, respectively. We assume that passenger demand between pairs of stations is symmetric in both directions. Since lines are typically operated in both directions and passenger demand is symmetrical, we focus only on trains traveling in one direction. The notations used in the formulation and the rest of the paper are summarized in Table 1.

Table 1: Notations

T	set of trains, $T = \{1, 2, \ldots, m\}$ indexed by t
N	set of stations, $N = \{1, 2, \ldots, n\}$ indexed by i and j
E	set of railroad sections, $E = \{(i, i+1) \mid i, i+1 \in N\}$
O	set of OD pairs/station pairs, $O = \{(i, j) \mid i, j \in N, i < j\}$
L	set of classes, $L = \{1, 2, \ldots, l\}$ indexed by r
$C^{r,t}$	capacity of class $r \in L$ in train $t \in T$
$d^r_{i,j}$	passenger demand for the OD $(i, j) \in O$ in class $r \in L$
$x^{r,t}_{i,j}$	passenger flow variable indicating number of seats to be allocated for the OD $(i, j) \in O$ in class $r \in L$ of train $t \in T$
y^t_i	binary stop variable = 1 if train $t \in T$ stops at station $i \in N$, = 0 otherwise

Here $x^{r,t}_{i,j}$ and y^t_i are the decision variables, also referred to as seat allocation and stoppage variables respectively. The TSSP is formulated mathematically as the MILP problem (1) - (7).

The objective (1) is to minimize the total number of train stops. Constraints (2) enforce the train capacity restriction, which implies that for each section of a train's route, the total number of passengers in any class cannot exceed the capacity of that class. Demand constraints (3) for a given OD pair of a particular class require that the total number of seats allocated across all trains for this OD pair exceed the passenger demand, i.e., no passenger demand is rejected. Constraints (4) and (5) ensure that passengers traveling between stations i and j can only be assigned to train t if it stops at both stations. Conversely, if passengers are traveling from station i to station j on train t (i.e., $x^{r,t}_{ij} > 0$), then train t must make stops at both stations. The large constants $M^{r,t}_{i,j}$ can take on the values $= \min\{C^{r,t}, d^r_{i,j}\}$. Note that here m implies an upper bound on the number of trains required to serve the passenger demand.

$$\text{Minimize} \sum_{t\in T}\sum_{i\in N} y_i^t \tag{1}$$

subject to

$$\sum_{i'\le i}\sum_{j>i} x_{i',j}^{r,t} \le C^{r,t} \qquad \forall i \in N\setminus\{n\},\ \forall t\in T,\ \forall r\in L \tag{2}$$

$$\sum_{t\in T} x_{i,j}^{r,t} \ge d_{i,j}^{r} \qquad \forall (i,j)\in O,\ \forall r\in L \tag{3}$$

$$x_{i,j}^{r,t} \le M_{i,j}^{r,t} y_i^t \qquad \forall (i,j)\in O, \forall t\in T,\ \forall r\in L \tag{4}$$

$$x_{i,j}^{r,t} \le M_{i,j}^{r,t} y_j^t \qquad \forall (i,j)\in O, \forall t\in T,\ \forall r\in L \tag{5}$$

$$x_{i,j}^{r,t} \ge 0 \qquad \forall (i,j)\in O, \forall t\in T,\ \forall r\in L \tag{6}$$

$$y_i^t \in \{0,1\} \qquad \forall i\in N,\ \forall t\in T \tag{7}$$

The proposed valid inequalities are - (1) Symmetry Breaking (SB), (2) Minimum Stoppage (MS), (3) Station Subset (SS), (4) Chvátal-Gomory (CG) inequality, (4) Traffic Restriction (TR), and (5) Polar (PR) inequalities. The concept of polar duality [4] has been utilized to find more complex valid inequalities which may be hard to find otherwise. Despite the problem's high practical relevance, valid inequalities for the problem have not yet been studied in the literature. An aggregation procedure has been proposed to solve large size problem instances exactly. The reader is referred to [3] for further details.

Computational study on several randomly generated problem instances demonstrate the efficacy of the proposed valid inequalities. From Figure 1 it can be observed that the LP relaxation bound of (1) - (5) is only about 30% of the optimal solution. On addition of the proposed valid inequalities, the lower bound surges to about 87%. Therefore, the LP formulation is significantly strengthened using these inequalities.

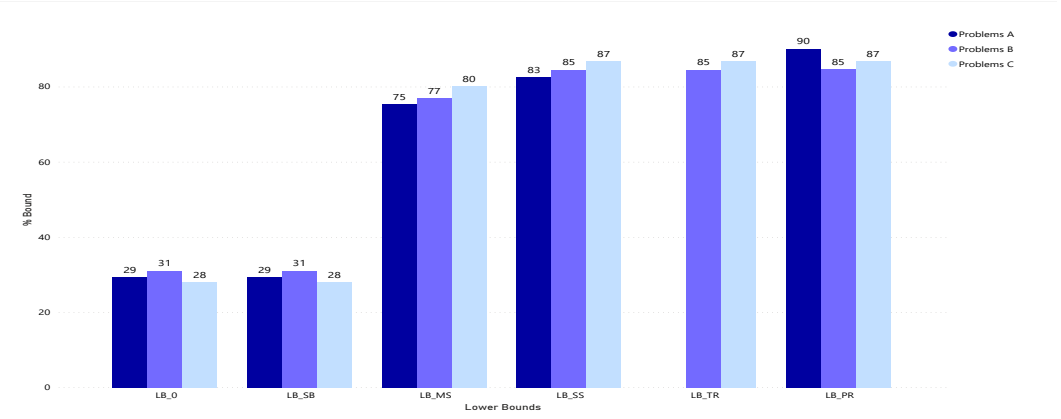


Figure 1: Lower bounds with problem category [3]

References

[1] Lianbo Deng, Feng Shi, and Wenliang Zhou. Stop schedule plan optimization for passenger train. *China Railway Science*, 30(4):102–107, 2009.

[2] Huiling Fu, Lei Nie, Benjamin R Sperry, and Zhenhuan He. Train stop scheduling in a high-speed rail network by utilizing a two-stage approach. *Mathematical Problems in Engineering*, 5:123–132, 2012.

[3] Faiz Hamid and Yogesh K Agarwal. Train stop scheduling problem: An exact approach using valid inequalities and polar duality. *European Journal of Operational Research*, 313(1):207–224, 2024.

[4] George L Nemhauser and Laurence A Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1999.

Session Session 4A: Routing Algorithms
Tuesday 12 March 2024, 15:30-17:00
Q011

A Triple Bottom Line optimization model for assignment and routing of on-demand home services

Debajyoti Biswas[1], Laurent Alfandari[2], and Claudia Archetti[3]

[1]UCD College of Business, Dublin, Ireland, ✉ debajyoti.biswas@ucd.ie
[2]ESSEC Business School, France, ✉ alfandari@essec.edu
[3]ESSEC Business School, France, ✉ archetti@essec.edu

There has been a significant rise in the consumption of on-demand services globally, ranging from ride-hailing, grocery, food delivery to crowd-shipping and other services (Taylor 2018). Platforms or online marketplaces which facilitate these services, act as an intermediary for matching service providers to customers and benefit from the revenue realised (Bai et al. 2019). These firms have disrupted traditional markets of services, transforming the global business landscape considerably. The on-demand industry has a strong growth projection, slated to reach a valuation of close to 335 billion USD globally by 2025, according to a report from PwC (Osztovits et al. 2015). One of the emerging segments of on-demand services is that of 'home services'. On-demand home service platforms like Handy, TaskRabbit, Homejoy, Thumbtack and Urban Company match independent service providers to consumers who need some kind of household service like cleaning, plumbing, electrical work, furniture repair etc. (Roose 2014).

While there has been considerable innovation and progress in adopting new technologies and business models in the recent past, firms have been gradually shifting towards more sustainable ways of doing business, guided by global regulations and policies around labour and the environment [1]. Sustainable development is succinctly defined in the Bruntland Commission (Brundtland et al. 1987) as "development that meets the needs of the present without compromising the ability of future generations to meet their own needs". Elkington (1997) characterises the concept of sustainability in business terms as the "triple bottom line", by which a firm adopts a holistic approach to measure and improve environmental and social performance besides economic performance.

We define the Home Services Assignment and Routing Problem with the Triple Bottom Line (HSARP-TBL) for modelling the assignment and routing of service professionals to customers of an on-demand home services platform incorporating the TBL criteria. We assign service professionals to customers based on their preferred time slots and the availability of professionals for those slots. The goal is to minimize the costs of penalties arising from slot time window violations (to reduce waiting time for customers and idle time for professionals) and uncovered customers for the professionals' visits to customers (economic). We apply additional constraints to capture different TBL criteria by - assigning professionals based on customer ratings and prioritising subscribed customers over non-subscribed ones for assigning higher-rated professionals, to improve customer satisfaction (economic), imposing a limit on overall emissions from the tours of professionals, allowing different modes of public transport as a more eco-friendly way of commuting (environment), ensuring a minimum net earning for professionals accounting for parking and travel costs (social).

The main contributions of this paper which distinguishes it from previous work in this area are the following:

1. This paper proposes an optimization model for assignment and routing of on-demand home services which addresses all three pillars of the TBL -economic, environmental, social. Contrary to previous formulations, we consider the possibility of combining multiple transport modes in the same tour for the environmental criteria.

2. We implement a Hybrid Genetic Search (HGS) metaheuristic for solving large instances. By applying HGS, we are able to find solutions with a better objective value compared to the MILP within a given cut-off time for a dataset representing a real urban transport network for the city of Paris with simulated demands.

[1]https://unfccc.int/process-and-meetings/the-paris-agreement

3. We show that imposing steep emission caps can lead to lesser customers being served, adversely impacting the primary economic performance of the platform, indicating that there is a strong trade-off between economic and environmental performance. We show that combining public transport and personal vehicles for serving customers can significantly reduce emissions (compared to using personal vehicles only) without impacting the primary economic criteria (penalties from time window violations).

4. For the social criteria, we find that ensuring fairness in terms of professional earning may not negatively impact the economic performance. Also, increasing the minimum earning threshold for service professionals cannot worsen the disparity in earnings for them.

5. We show that for some instances, environmental, social and (customer satisfaction-based) economic criteria can be improved without worsening the primary economic objective (compared to the case with no TBL criteria), when we impose constraints based on the TBL.

Very few papers have combined all the TBL pillars for routing and assignment of Home Services (HS) or Home Health Care (HHC). Although Fathollahi-Fard et al. (2022) address the TBL pillars in the context of HHC, there are many features that are unique in the HSARP-TBL model formulation that we develop. We define a separate profit function for service professionals including total revenue based on the service fee from each customer visit and transportation and parking costs for each visit based on the vehicle type and customer location and ensure a minimum profit for each professional. For customer satisfaction and retention, we use historical service ratings of professionals. We give higher priority to subscribed customers in assigning top-rated professionals (compared to non-subscribed customers) [2]. We model the choice of transport mode for service professionals with the provision of combining different forms of public transport in the same route (or using one's personal vehicle throughout the route) with a focus on controlling emissions across all professionals' tours.

We implement the HGS algorithm introduced by Vidal et al. (2012) and adapt it to our problem context. We introduce some additional novel operators for the initial solution generation process (before applying the genetic algorithm) for incorporating each of the TBL pillars, using a greedy approach. Similar to Vidal et al. (2012) we compute a composite score of each solution comprising of the objective value score and the diversity score based on the relative diversity of the solution with respect to the population of solutions. However, the choice of transport mode adds a layer of complexity to the algorithm, making our formulation different from previous ones. We apply the selection, crossover and mutation operators along with some local search operators to explore neighbourhoods of promising solutions in each iteration. Finally, we apply a refinement operator to check if the best solution from the genetic algorithm can be further improved. We performed our computational experiments on an M1 Mac OSX System with 16 GB RAM, 3.2 GHz, 8 core processor, using CPLEX 12.10. We conducted the tests for a range of 12 to 60 customers and 3 to 15 professionals and 3 modes of transport - bus, train, car.

References

Bai, Jiaru, Kut C So, Christopher S Tang, Xiqun Chen, Hai Wang. 2019. Coordinating supply and demand on an on-demand service platform with impatient customers. *Manufacturing & Service Operations Management* **21**(3) 556–570.

Brundtland, Gro Harlem, Mansour Khalid, Susanna Agnelli, Saleh A Al-Athel, BJNY Chidzero, LM Fadika, Volker Hauff, Istvan Lang, S Ma, Margarita Morino de Botero, et al. 1987. Our common future; by world commission on environment and development .

Elkington, John. 1997. The triple bottom line. *Environmental management: Readings and cases* **2** 49–66.

Fathollahi-Fard, Amir M, Abbas Ahmadi, Behrooz Karimi. 2022. Sustainable and robust home healthcare logistics: A response to the covid-19 pandemic. *Symmetry* **14**(2) 193.

Osztovits, Ádám, Árpád Kőszegi, Bence Nagy, Bence Damjanovics. 2015. Sharing or paring? growth of the sharing economy. URL `https://www.pwc.com/hu/en/kiadvanyok/assets/pdf/sharing-economy-en.pdf`. Last accessed Jan 21, 2023.

Roose, Kevin. 2014. Does silicon valley have a contract-worker problem. *New York Magazine* **18**.

Taylor, Terry A. 2018. On-demand service platforms. *Manufacturing & Service Operations Management* **20**(4) 704–720.

Vidal, Thibaut, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, Walter Rei. 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research* **60**(3) 611–624.

[2] https://www.timesprime.com/categories/essentials/urban-company

Exploring varied average speeds to assess energy consumption and charging profiles in EVRP benchmark instances

Clíodhna Ní Shé[1], Damian Flynn[2], and Paula Carroll[1]

[1]School of Business, University College Dublin, Dublin, Ireland, ✉ cliodhna.nishe@ucdconnect.ie
[2]School of Electrical and Electronic Engineering, University College Dublin, Dublin, Ireland, ✉ damian.flynn@ucd.ie

The energy consumption model used in electric vehicle routing problems (EVRP) is a topic that has different perspectives. This abstract presents computational experiments investigating the effect that vehicle speed has on energy consumption (EC) in EVRP benchmark instances, based on EC models found in the literature. Factors that have been shown to affect the EC of electric vehicles (EV) are rolling resistance, air resistance, slope resistance, total mass of the EV (and people in it) and the cargo (if any) [5, 3]. In addition to these factors, ambient temperature affects battery performance and charging time. On top of that, air conditioning has proven to be the most demanding auxiliary load [2]. Traffic congestion and road type affect the driving speed of the vehicles, which in turn affects the EC. There are well established energy factor models for the EC (kWh/km), for different road grades given the average speed of the vehicle (light duty EVs), equations 1-4 from [6] have been utilised and built upon in several EC models in different papers, where v is the average speed.

$$Expressway(v) = 0.247 + \frac{1.520}{v} - 0.004v + 2.992 \times 10^{-5}v^2 \tag{1}$$

$$ArterialRoad(v) = -0.179 + 0.004v + \frac{5.492}{v} \tag{2}$$

$$SecondaryRoad(v) = 0.21 - 0.001v + \frac{1.531}{v} \tag{3}$$

$$Branch(v) = 0.208 - 0.002v + \frac{1.553}{v} \tag{4}$$

The benchmark instances for the capacitated EVRP (E-CVRP) were developed in [4] and consist of a subset of modified instances from the benchmark CVRP instances. There are 24 instances in total from four different sets of benchmark CVRP instances. The algorithm used to convert the instances from CVRP to E-CVRP is provided in [4]. In these benchmark instances, a standard EC factor is used, which in all instances, for every road, is equal to 1. This is to say that in every instance, the vehicle uses 1 unit of energy to traverse one unit of distance. The energy capacity Q for each benchmark instance is defined as $Q = 2\bar{d}$ where $\bar{d}$ is the maximum euclidean distance between the depot and all other nodes.
It is clear from the literature that the EC of an EV depends on more factors than simply distance travelled. Evidence has shown that speed is a significant factor in the energy consumed by an EV. With this in mind, we investigate the impact of average speed by introducing more meaningful EC to the E-CVRP benchmark instances. We assume three separate cases of vehicle speed to identify how this will effect the solutions.

We use equations 1-4 and the supporting data from [6] to alter the EC factor in the benchmark instances. From the speed distributions on each road, we find an average speed for each road type. Using this average speed, we calculate the average EC per kilometre for each road grade. The EC per km for each road ranges between 0.157 kWh/km and 0.262 kWh/km which is significantly less than the energy consumption of 1 defined in the benchmark instances. With these average EC per km values for each road grade, we assume that there is a uniform distribution of road grades (25% of roads are of each grade) and we compute an average EC per km. The energy capacity Q of the EVs needs to be adapted to reflect the impact on battery performance due to driving speed. To reflect this, for each instance, Q from the benchmark instance is multiplied by this average energy consumption factor which is 0.203. Therefore, for each of the 24 instances we now have $Q_{mod} = EC_{mod} \times Q$ which is a modified energy capacity for the

vehicles in each instance.
To explore the effect that average speed has on the energy consumption in EVRP benchmark instances, we investigated three different average speed scenarios. For the speed travelled on each road, we defined a slow average speed, average speed and a fast average speed. EC rates for each road grade, in each scenario were obtained using equations 1-4. The road grade of each edge was assigned based on distance from the depot. Edges were categorised based on how far away the furthest vertex from the depot was. We used a modified Clarke-Wright savings heuristic in [1] to solve each of our modified instances for each scenario. The number of charging stations (CS) visited (Figure 1a), the total distance travelled (Figure 1b) and the number of extra vehicles (more than k defined in the instance) (Figure 1c) in each scenario is compared for every instance.

(a) CS visited (b) Distance (c) Vehicles more than predefined k

Figure 1: Comparison of the charging stations (CS) visited, distance travelled and extra vehicles needed for each instance for the three studied speed cases, and the case with no road grade (RG) or EC model

Figure 1 shows that travelling at a slow speed on average increases the number of CS visited, the number of vehicles needed and the distance travelled in nearly all instances. This reinforces the fact that traffic congestion plays a pivotal role in computing the EC of vehicles during trips. We predict that with a more sophisticated road grade assignment strategy and with more factors taken into account in our EC model, we will be able to understand more about the effect that vehicle speed has on energy consumption for these EVRP benchmark instances. Future work will include an analysis of where the depot is located relative to the customers and whether the road grades should reflect this.

References

[1] Sevgi Erdoğan and Elise Miller-Hooks. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114, 2012. Select Papers from the 19th International Symposium on Transportation and Traffic Theory.

[2] Jing Huang, Xiuli Wang, Chengcheng Shao, Zhenzi Song, Yifei Wang, and Xuanyue Shuai. An improved charging navigation strategy of electric vehicles via optimal time-of-use pricing. *Electric Power Systems Research*, 210:108077, 2022.

[3] Chuanyao Li and Yiting Chen. Evaluation and comparative analysis of the mixed traffic flow on urban roads considering the green cost. *Sustainable Energy Technologies and Assessments*, 57:103292, 2023.

[4] Michalis Mavrovouniotis, Charalambos Menelaou, Stelios Timotheou, Georgios Ellinas, Christos Panayiotou, and Marios Polycarpou. A benchmark test suite for the electric capacitated vehicle routing problem. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2020.

[5] Mengcheng Tang, Weichao Zhuang, Bingbing Li, Haoji Liu, Ziyou Song, and Guodong Yin. Energy-optimal routing for electric vehicles using deep reinforcement learning with transformer. *Applied Energy*, 350:121711, 2023.

[6] Enjian Yao, Zhiqiang Yang, Yuanyuan Song, and Ting Zuo. Comparison of electric vehicle's energy consumption factors for different road types. *Discrete Dynamics in Nature and Society*, 2013:Article ID 328757, 7 pages, 2013.

An Improved Single-Commodity Flow Formulation for the Vehicle Routing Problem with a Heterogeneous Fleet

Devanand
SnT, University of Luxembourg
Kirchberg, Luxembourg
devanand.devanand@uni.lu

ABSTRACT

The heterogeneous vehicle routing problem (HVRP), a variant of the classical vehicle routing problem (VRP), involves optimizing route planning for vehicles with different load capacities, each designed for specific tasks or constraints. Improving the HVRP model not only enhances the solution quality and reduces solving time but could also be useful for related extended versions such as HVRP with time windows, pickup and delivery, multiple depots, stochastic elements, and industry-specific constraints. In this paper, we present a novel formulation for HVRP that uses a single-commodity flow approach based on a 2-index formulation. In contrast to the conventional single-commodity flow formulation, our approach requires a significantly smaller number of variables. We performed a computational experiment to show the efficiency of our model by solving some HVRP instances and found a significant advantage.

1 INTRODUCTION

A classical vehicle routing problem (VRP) involves customers with specified item demands to be fulfilled by a identical fleet of vehicles. Here, all vehicles originate and conclude their routes at a common point, a depot. The primary aim is to minimize the combined distance covered by all vehicles while meeting the customers' demands. VRP has been extensively studied due to its direct economic and environmental importance in logistic and supply chain operations. The transportation process constitutes 10% to 20% of the ultimate cost of goods. Also, international freight transport accounts for around one-third of the total CO2 emissions [21]. Due to this, research on VRPs has always been demanding and growing exponentially [5]. VRP was first presented by Dantzig and Ramser [8]. The initial stage of VRP works often focused on developing mathematical models and exact algorithms for homogeneous fleets, serving as a foundation for later extensions. We refer reader [7, 14, 18] for various exact and heuristics techniques under such VRPs.

The VRP with a heterogeneous vehicle fleet, called heterogeneous VRP (HVRP), is a popular VRP that allows organizations to deploy various vehicles with different capacities, each tailored to specific tasks or constraints.

HVRP is formally described as follows: Suppose $G = (V, E)$ is a complete graph where $V = \{0, 1, \ldots, n\}$ is a set of nodes, and $E = \{(i, j) \mid i, j \in V,\ i\ != j\}$ is the set of all possible edges between nodes. Here, $0 \in V$ represents a central depot, where all the vehicles start and return by serving all the customers. $N = V \backslash \{0\}$ represents the index set of customer location. The depot has τ number of vehicles whose capacities are denoted by a set $T = \{t_1, t_2, \ldots, t_\tau\}$, where t_i denotes the maximum capacity i^{th} vehicle can carry. In the case of VRP with a homogeneous fleet of vehicles, capacities will be the same, $t_i = t_j$; otherwise, at least one vehicle will be different by capacity. For each arc $(i, j) \in E$, we have a transportation cost $c_{i,j}$ to travel from customer location i to location j. For simplicity, we consider $c_{i,j}$ as the distance between location i and j. For each $i \in V$, we represent associated pickup quantities by $p_i \geq 0$. Our objective is to determine the vehicle routes from the depot 0 to customer points so that

- total cost be as minimum as possible
- total number of vehicles used as small as possible
- each customer is visited exactly once
- all the vehicles start from and end at the depot
- all customer demands must served by the vehicle

The pickup demand at each demand point is known before departure, and it can not be split. We assume that any vehicle can serve any of the customers. It means the restriction that a specific sized vehicle only serves a particular customer is not considered in our problem. The objective is to determine the optimal vehicle routes to pick up goods after reaching the demand point without violating the vehicle's maximum carrying capacities. We limit our focus to the basic HVRP with a fixed number of vehicles, each with varying capacity constraints.

HVRPs have received greater attention in the literature. It was first studied in the seminal work of Golden, *et al.* [13] and has since developed into an extensive field of research. Detailed surveys of HVRP are conducted by [16] and cover the 30 years of development since HVRP was developed by Golden, *et al.*. There are works by [1, 2] that cover the solution strategy of HVRP. The model studied in the rich-VRP literature can be found in [6].

The HVRP is NP-hard as it is a natural generalization of the travelling salesman problem (TSP). Many heuristics and exact methods have been proposed in the literature. Classical heuristics leverage extensions from well-established heuristics for classical VRPs [12, 22, 23]. Tabu search-based heuristics, extensively tested and studied, have proven effective for HVRPs [10, 25]. For branch-and-cut and branch-price-and-cut are the main approaches that depend on the modeling of the HVRP. We can find the work of lower bounding and its variants for HVRP in [2]. [20] studies exact solving procedures using branch-and-price-and-cut for VRP.

For exact branch-and-bound algorithms, [17, 19] have made significant contributions; however, these algorithms tend to work optimally only on relatively small instances. In contrast, Baldacci and Mingozzi [4] present an exact algorithm for the HVRP, which

generalizes bounding procedures and exact methods described for the CVRP. They introduce novel bounding methods demonstrating particular effectiveness when the vehicle's fixed cost contribution to the total cost is significant. Various valid inequalities and delayed constraints specific to the problem are employed to speed up the solving procedure, such as capacity cuts, comb cuts, etc. [3, 26].

Researchers have proposed various mathematical models for the HVRP. Broadly, three variants of formulation for HVRPs are studied.

The first is based on single-commodity flow formulation [2]. In this, the entire vehicle fleet is considered as a single commodity. This formulation is based on the flow of the commodity from the depot to the customers and back to the depot. The objective is to minimize the total cost, often a combination of travel distances, vehicle fixed costs, and other relevant factors. The second type is the two-flow formulation of [3]. In this formulation, it is assumed that the vehicle types do not dominate and are ordered. The network of customers and depots is considered symmetric. In addition, a dummy depot is included in the modeling. Another type of formulation is the set partitioning-based formulation. In the set partitioning formulation of [4], given an undirected graph $G(V, E)$, each route is assumed to be a subset of the customer set V associated with cost, and the goal is to select the set of routes such that the union of all routes is V, subject to the associated VRP constraints. In this paper, we focus on the single-flow formulation.

Two-index vehicle flow models have found application in representing basic versions of classical symmetric and asymmetric-VRP (AVRP), including variants like the VRP with backhauls [24]. However, as the complexity of VRPs increases, these models may prove insufficient. To solve this problem, we explicitly specify the vehicle traveling through an arc so that more complex constraints can be imposed on the routes. This explicit representation allows the imposition of complicated constraints on the routes and provides a more flexible and robust solution for solving complex VRPs.

This paper contributes a novel formulation, a 2-index commodity flow model, which has fewer variables and constraints than the existing formulations of VRPSPDs, which are generally modeled as a 3-index commodity flow formulation. We discuss the existing model for VRPSPD in Section 2. Our new model is discussed in Section 3. In Section 4, we show computational results that highlight the advantages of our model. Finally, we summarize our work in Section 5 and give an outlook on future work.

2 EXISTING SINGLE-COMMODITY FLOW MODEL FOR HVRP

We follow the model of F. Gheysens *et al.*. [11]. It is a 3-index commodity flow formulation, the most widely used model, and is also suitable for modeling other complex classes of VRP.

The parameters and sets used in the model are the same as discussed above. For heterogeneity of vehicles, the capacity of some vehicles is non-identical. So, for some $t_i,\ t_j \in T,\ t_i \neq t_j$. Broadly, two categories of decision variables for the model are as follows:

- $x_{k,ij} = 1$ if vehicle k moves from customer i to j. Otherwise, it takes a 0 value.
- $pv_{i,j}$ is the decision variable that indicates the total load a vehicle carries while traversing from customer i to customer j for pickup of the items.

The problem is modeled as follows:

$$\text{3-HVRP} := \min \sum_{k=1}^{\tau} \sum_{i=0}^{n} \sum_{j=0}^{n} x_{k,ij}\ cost_{i,j}$$

$$\text{s.t.} \sum_{k=1}^{\tau} \sum_{i \in V, j \neq i} x_{k,ij} = 1\ \forall j \in N, \tag{1}$$

$$\sum_{j \in V} x_{k,ij} - \sum_{j \in V} x_{k,ji} = 0\ \forall i \in V,\ \forall k \in \{1 \ldots, \tau\}, \tag{2}$$

$$\sum_{i \in V} pv_{i,j} + p_j = \sum_{i \in V} pv_{j,i} \quad \forall j \in N, \tag{3}$$

$$\sum_{j \in N} x_{k,0j} \leq 1 \quad \forall k \in \{1, \ldots, \tau\}, \tag{4}$$

$$\sum_{i \in N} x_{k,i0} \leq 1 \quad \forall k \in \{1, \ldots, \tau\}, \tag{5}$$

$$pv_{i,j} \leq \sum_{k=1}^{\tau} t_k x_{k,ij}\ \ \forall i \in V,\ \forall j \in V, \tag{6}$$

$$x_{k,ij} \in \{0, 1\},\ \forall i \in V,\ \forall j \in V,\ \forall k \in \{1, \ldots, \tau\}, \tag{7}$$

$$pv_{i,j} \geq 0, \forall i \in V, \forall j \in V,$$

The model's objective function is to minimize the cost associated with the edges covered by vehicles. Constraint (1) ensures that each customer is served exactly once by one of the vehicles. Constraint (2) guarantees that the same vehicle enters and leaves the customer point. Flow conservation restrictions for pickup are mentioned in constraint (3). Constraints 4 and 5 represent flow-in and flow-out vehicles at the depot, which limits each vehicle to be used for a maximum of one route. Constraint (6) imposes the capacity constraint. It is a Miller-Tucker-Zemlin constraint [9] that handles the problem of subtour in the solution. Trivial constraints on binary and continuous variables are imposed in (7).

The model consists of variables having three indices. Clearly, such model requires $O(n^2 \cdot \tau)$ variables.

3 IMPROVED FORMULATION

3.1 Motivation

The existing model mentioned in the previous section explicitly tracks the vehicle type to serve a given customer. So, if $x_{k,ij} = 1$. It makes it very clear that the vehicle k served (reached) the customer (depot) j after serving (starting) from i.

Consider we have at least one feasible solution for a given HVRP problem. Without loss of generality, we can consider $0 - 1 - 2 - 3 \ldots, n - 0$ as a feasible path. The consequent solution obtained from the model 3-HVRP is $x_{1,01} = x_{1,12} = x_{1,23} = \ldots = x_{1,n0}$. It implies that vehicle 1 serves all the customers in the lexicographical order. For other vehicle, the variables $x_{t,n0}, \forall t \in \{2, \ldots, \tau\}$, take the value 0. If we replace $x_{k,ij}$ with a binary decision variable $x_{i,j}$, for the given feasible path, $x_{0,1} = x_{1,2} = x_{2,3} = \ldots = x_{n,0}$. However, we needed the information about the vehicle type which served all the customers. If $x_{n,0} = 1$ and vehicle k serves customer n associated

to the feasible path, then it will serve all the remaining customers $j = \{1, \ldots, n-1\}$ to this path.

We can generalize this idea by introducing a new binary decision variable $y_{j,k}$ that represents whether the vehicle k serves the feasible paths such that the last visited customer is j. Clearly, $y_{j,k} = 1$ if $x_{j,0}$ is 1 and vehicle number k is used to carry the load associated at customer points $j \in J$. Where J is the set of all last visited customer point by any vehicles. That is, $J = \{j \in N \mid x_{j,0} = 1\}$. So, $\sum_{k=1}^{\tau} y_{j,k} = 1,\ j \in J$. Since we do not have any information about J before computing the model, we can rewrite it as -

$$\sum_{k=1}^{\tau} y_{j,k} = x_{j,0},\ \forall j \in N. \tag{8}$$

Equation 8 ensures that if the last visited customer is not j then $y_{j,k} = 1$ for any $k \in \{1, \ldots, \tau\}$. One important condition related to $y_{j,k}$ is the following:

$$\sum_{j \in N} y_{j,k} \leq 1,\ \forall k \in \{1, \ldots, \tau\}. \tag{9}$$

This ensures that a vehicle $k \in \{1, \ldots, \tau\}$ can not be connected to more than one feasible path.

Now we can associate the variables $y_{j,k}$ and $pv_{j,0}$ that form the maximum carrying capacity of vehicle k,

$$p_{var_{i,0}}\ y_{i,k} \leq t_k,\ \forall i \in N,\ \forall\, k \in \{1, \ldots, \tau\}. \tag{10}$$

Equation (10) consists nonlinear terms. We can linearlize the product of binary and continuous variables $z = p_{var_{i,0}}\ y_{i,k} \leq t_k$ as follows:

$$\begin{aligned} &z \leq y_{i,k} M, \\ &z \leq p_{var_{i,0}}, \\ &z \geq p_{var_{i,0}} + (1 - y_{i,k}) M, \\ &0 \leq z \leq t_k. \end{aligned} \tag{11}$$

Here M is a suitable large number. Since most of the optimization solvers also handle logical constraints, an alternative to the systems of Equations (11), we can have the following constraints:

$$y_{i,k} = 1 \implies p_{var_{i,0}} \leq t_k. \tag{12}$$

3.2 Model

Using the above constraints and other HVRP constraints, we form the 2-index formulation as follows:

$$\text{2-HVRP} := \min \sum_{i=0}^{n} \sum_{j=0}^{n} x_{i,j}\ cost_{i,j}$$

$$\text{s.t.} \sum_{i \in V, j \neq i} x_{i,j} = 1 \quad \forall j \in N, \tag{13}$$

$$\sum_{j \in V, i \neq j} x_{i,j} = 1 \quad \forall i \in N, \tag{14}$$

$$\sum_{i \in V} pv_{i,j} + p_j = \sum_{I \in V} pv_{j,i} \quad \forall j \in N, \tag{15}$$

$$\sum_{i \in N} pv_{0,i} = 0, \tag{16}$$

$$\sum_{i \in N} pv_{i,0} = \sum_{j \in N} p_j, \tag{17}$$

$$\sum_{k=1}^{\tau} y_{j,k} = x_{j,0},\ \forall j \in N \tag{18}$$

$$\sum_{j \in N} y_{j,k} \leq 1,\ \forall k \in \{1, \ldots, \tau\}, \tag{19}$$

$$y_{i,k} = 1 \implies p_{var_{i,0}} \leq t_k,\ \forall i \in N,\ \forall\, k \in \{1, \ldots, \tau\}, \tag{20}$$

$$x_{i,j} \in \{0, 1\},\ \forall i \in V,\ \forall j \in V, \tag{21}$$

$$pv_{i,j} \geq 0, \forall i \in V, \forall j \in V,$$

$$y_{i,k} \in \{0, 1\},\ \forall i \in N, \forall k \in \{1, \ldots, \tau\}.$$

Here, similar to the model 3-HVRP , the objective function in the model is to minimize the cost associated with the edges covered by vehicles. The indegree and outdegree constraints (13) and (14) ensure that exactly one entry and exit is allowed at each customer. Constraint (15) guarantees flow conservation restrictions for pickup. Equation (16) and (17) make sure that vehicles start empty from the depot and return with all the picked-up items to the depot. The details of constraints (18), (19) and (20) are provided in Section 3.1. Trivial constraints on binary and continuous variables are imposed in (21). Note that the model consists of variables having only two indices - such a model requires $O(n^2)$ variables and $O(n^2)$ constraints.

The model can be helpful to other classes of VRP problems with more added constraints - we can have a similar formulation for HVRP with pickup and load. Since time-window-based restrictions in HVRPs do not require any variable related to the vehicle's capacity, our model can adapt to such time-window-based problems.

4 COMPUTATIONAL EXPERIMENTS

In this section, we give empirical evidence of the effectiveness of our 2-index-based model (2-HVRP) in some HVRP instances and provide computational details of them. We compare the performance of our model with that of the 3-HVRP by solving it with the 22.1.1.0 version of CPLEX, one of the fastest optimization solvers. Both models are coded in Python (version 3.8), utilizing the CPLEX Python API. This Python package within CPLEX facilitates access to the Callable Library from the Python programming language.

The hardware used for the computation is a Mac OS M2 chip, an 8-core CPU supporting a 10-core GPU with a 3.49 GHz CPU. To

avoid multiple processes sharing common resources, we run one job at a time with the default settings of CPLEX API.

We generate 18 test instances for our experiment, each with different customers and vehicles. The locations of the vehicles and customers are two-dimensional coordinate points (x, y) that are randomly generated, where x and y are from a uniform distribution such that $x \sim U[a_1, a_2]$ and $y \sim U[b_1, b_2]$.

It should be noted that there are HVRP instances that have already been well studied and tested [15]. Our experiments focused on smaller data sets. This was a deliberate choice, as our current work does not focus on refining solution strategies, but aims to motivate readers for the effectiveness of using an improved model. Consequently, we applied our model to the optimization solver avoiding delving into developing an exact method for its solution.

The vehicle's capacity is chosen randomly and is uniformly distributed between q_1 to q_2. The value of items to be picked up by vehicle at the customer location is also randomly generated, uniformly distributed between p_1 to p_2. Selection of the number of vehicles should ensure sufficient supply to serve all the customers. The simple approach to estimate this value is always to keep the number of vehicles more than the sum of total items to be picked divided by the average capacity of a vehicle.

Our data set is generated with the following suitable values: $a_1 = b_1 = 0$, $a_2 = 200$, $b_2 = 100$, $p_1 = 1$, $p_2 = 5$, $q_1 = 5$ and $q_2 = 10$. Out of 18 test instances, we show the detailed specification of the first 4 instances in Tables 1 and 2. In Table 1, for each instances, we list n, number of customers, τ, maximum number of vehicles available, $p_i,\ i = 1, \ldots, n$, pickup items at each customer locations , and $t_j,\ j = 1, \ldots, \tau$, the maximum carrying capacities of each vehicle. Note that the capacity of vehicles and items to be picked at the customer ends have the same units. Each $n_i,\ i = 1, \ldots, n$ in Table 2 is the $x-y$ coordinate that represents customer locations. For our experiments, we consider $cost_{i,j} = \|n_i - n_j\|_2$, the Euclidean distance between customers i and j. The remaining test instances consist of the following number of customer n and the number of vehicles τ:

$$(n, \tau) = \begin{cases} (8, 4), & \text{if } I = I5,\ I6,\ I7,\ I8, \\ (10, 6), & \text{if } I = I9,\ I10,\ I11,\ I12, \\ (15, 8), & \text{if } I = I13,\ I14,\ I15,\ I16, \\ (40, 20), & \text{if } I = I17,\ I18. \end{cases}$$

All test instances (in CSV)and models (in .LP format) are available on https://github.com/devanandR/HVRP.git.

Table 3 compares the performance of our 2-HVRP to the existing 3-HVRP . For both the models, we report optimal objective value, deterministic solving (wall) time taken by Cplex, and the number of vehicles used by the obtained optimal route, denoted by 'objval', 'time', and 'v-used', respectively. The unit for the time used is in seconds. We set the time limit of 600 seconds. The last column, 'improvement', compares the solving times. The first comparison, '%', reports the solving time benefit of using our method in terms of percentages. The second performance, 'times', measures how often our models are faster than the existing method. Compared to a 3-index-based model, our approach takes almost negligible time for small-sized instances. For a better picture of the effectiveness of our method, we refer to Figure 1. We use a log scale to show the solving time to illustrate the comparison. The blue column represents our model and the red column represents the existing model. The average time CPLEX takes to solve all the first 16 instances modeled as 2-HVRP is 0.56 seconds, much less than that of 3-HVRP , which takes 58 seconds.

The last two problem instances, I17 and I18, are chosen to be a difficult problem. Both the models hit the maximum time limit for I17 and I18. Interestingly, our model for such instances found a feasible integer solution with less than a 10 % optimality gap. However, the existing model could not find a feasible solution for such instances.

Table 1: Vehicles Capacities & Customer Demands

I	n	τ	p_1	p_2	p_3	p_4	p_5	t_1	t_2	t_3
I_1	5	3	1	1	2	2	1	6	9	7
I_2	5	3	2	2	4	4	3	5	7	7
I_3	5	3	4	1	2	1	3	8	9	9
I_4	5	3	1	2	2	3	3	9	6	8

Table 2: Customer Locations (the x-y-coordinate) for the Instances in Table 1

I		depot	n1	n2	n3	n4	n5
I1	x	7.946	199.811	122.772	24.721	125.448	167.456
	y	76.594	45.897	97.952	99.052	3.214	13.750
I2	x	192.64	140.37	170.73	82.36	114.75	59.00
	y	93.47	33.29	35.20	19.75	90.67	0.16
I3	x	88.51	134.67	148.54	126.48	180.12	94.98
	y	60.07	71.64	51.23	74.93	82.48	78.19
I4	x	62.95	141.06	15.64	113.02	64.45	17.54
	y	76.35	82.10	79.78	49.71	85.23	35.42

Table 3: Computational Summary of the Performance of our Model, 2-HVRP Compared to 3-HVRP

I	3-indexModel			2-index			improvement	
	objval	time	v-used	objval	time	v-used	%	times
I1	446.51	0.18	1	446.51	0.016	1	91.11	11.3
I2	654.37	0.023	3	654.37	0.02	3	13.04	1.2
I3	239.91	0.05	2	239.91	0.017	2	66.00	2.9
I4	327.52	0.1	2	327.52	0.022	2	78.00	4.5
I5	632.91	0.19	3	632.91	0.04	3	78.95	4.8
I6	353.16	0.31	3	353.16	0.076	3	75.48	4.1
I7	489.84	0.13	3	489.84	0.036	3	72.31	3.6
I8	431.13	0.15	3	431.13	0.06	3	60.00	2.5
I9	693.81	0.62	4	693.81	0.33	4	46.77	1.9
I10	665.15	1.32	4	665.15	0.3	4	77.27	4.4
I11	692.5	0.26	3	692.5	0.04	3	84.62	6.5
I12	618.5	0.66	3	618.52	0.12	3	81.82	5.5
I13	987.065	392.43	5	987.065	3.72	5	99.05	105.5
I14	818.7	45.5	5	818.7	0.31	5	99.32	146.8
I15	1319.46	6.46	5	1319.46	0.47	5	92.72	13.7
I16	1168.76	493.16	5	1168.76	3.45	5	99.30	142.9
I17	unsolved	timeout		2659.48 (9.88%)	timeout	15	-	-
I18	unsolved	timeout		2988(8.7%)	timeout	14	-	-

5 CONCLUSION AND FUTURE WORK

In this paper, we have presented a 2-index-based single-commodity flow model for the heterogeneous vehicle routing problem (HVRP). Our model has been shown to be a compelling alternative to the existing 3-index-based single-commodity flow model, as it requires significantly fewer variables. This reduction in complexity contributes to a computationally more efficient technique for solving the problem. When solving the models with CPLEX, an optimization solver, we saw remarkable speedups for smaller problem instances with up to 15 customers. In addition, for two larger instances with 40 customers, our model also showed impressive result by providing feasible integer solutions with a gap of less than 10% for instances where the existing model struggled to reach even a single integer feasible solution. Notably, our model can be extended to other classes of problems, including HVRP with time window constraints, pickup and delivery considerations, and other related HVRP extensions.

We have only tried to model the problem and focus on the basic version of the HVRP. Other complex constraints, such as HVRP with multiple depots, HVRP where some customers are only allowed to use certain types of vehicles, and related complex HVRP, are something that we are working towards.

The current work focuses only on modeling the problem. Our immediate research direction is to perform extensive computational experiments by exploiting valid inequalities and delayed constraints to solve large instances.

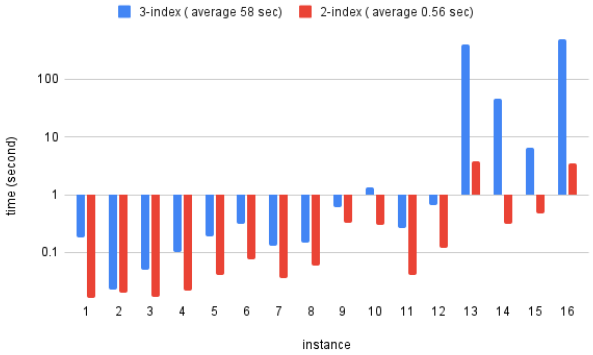


Figure 1: A Column Chart Comparing the Solving Time

REFERENCES

[1] Roberto Baldacci, Enrico Bartolini, Aristide Mingozzi, and Roberto Roberti. 2010. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science* 7 (2010), 229–268.

[2] Roberto Baldacci, Maria Battarra, and Daniele Vigo. 2008. Routing a heterogeneous fleet of vehicles. *The vehicle routing problem: latest advances and new challenges* (2008), 3–27.

[3] Roberto Baldacci, Maria Battarra, and Daniele Vigo. 2009. Valid inequalities for the fleet size and mix vehicle routing problem with fixed costs. *Networks: An International Journal* 54, 4 (2009), 178–189.

[4] Roberto Baldacci and Aristide Mingozzi. 2009. A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming* 120 (2009), 347–380.

[5] Kris Braekers, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. 2016. The vehicle routing problem: State of the art classification and review. *Computers & industrial engineering* 99 (2016), 300–313.

[6] Jose Caceres-Cruz, Pol Arias, Daniel Guimarans, Daniel Riera, and Angel A Juan. 2014. Rich vehicle routing problem: Survey. *ACM Computing Surveys (CSUR)* 47, 2 (2014), 1–28.

[7] Jean-François Cordeau, Gilbert Laporte, Martin WP Savelsbergh, and Daniele Vigo. 2007. Vehicle routing. *Handbooks in operations research and management science* 14 (2007), 367–428.

[8] George B Dantzig and John H Ramser. 1959. The truck dispatching problem. *Management science* 6, 1 (1959), 80–91.

[9] Martin Desrochers and Gilbert Laporte. 1991. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters* 10, 1 (1991), 27–36.

[10] Michel Gendreau, Gilbert Laporte, Christophe Musaraganyi, and Éric D Taillard. 1999. A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research* 26, 12 (1999), 1153–1173.

[11] Filip Gheysens, Bruce Golden, and Arjang Assad. 1984. A comparison of techniques for solving the fleet size and mix vehicle routing problem. *Operations-Research-Spektrum* 6 (1984), 207–216.

[12] Filip Gheysens, Bruce Golden, and Arjang Assad. 1986. A new heuristic for determining fleet size and composition. *Netflow at Pisa* (1986), 233–236.

[13] Bruce Golden, Arjang Assad, Larry Levy, and Filip Gheysens. 1984. The fleet size and mix vehicle routing problem. *Computers & Operations Research* 11, 1 (1984), 49–66.

[14] Bruce L Golden, Subramanian Raghavan, Edward A Wasil, et al. 2008. *The vehicle routing problem: latest advances and new challenges*. Vol. 43. Springer.

[15] Rosa Herrero, Alejandro Rodríguez, José Cáceres-Cruz, and Angel A Juan. 2014. Solving vehicle routing problems with asymmetric costs and heterogeneous fleets. *International Journal of Advanced Operations Management* 6, 1 (2014), 58–80.

[16] Çağrı Koç, Tolga Bektaş, Ola Jabali, and Gilbert Laporte. 2016. Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research* 249, 1 (2016), 1–21.

[17] Gilbert Laporte. 1981. Comb. Inequalities for the Vehicle Routing Problem. *Methods of Operations research* 51 (1981), 271–276.

[18] Gilbert Laporte. 2009. Fifty years of vehicle routing. *Transportation science* 43, 4 (2009), 408–416.

[19] Gilbert Laporte and Yves Nobert. 1987. Exact algorithms for the vehicle routing problem. In *North-Holland mathematics studies*. Vol. 132. Elsevier, 147–184.

[20] Artur Pessoa, Eduardo Uchoa, and Marcus Poggi de Aragão. 2009. A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks: An International Journal* 54, 4 (2009), 167–177.

[21] Amit Rauniyar, Rahul Nath, and Pranab K Muhuri. 2019. Multi-factorial evolutionary algorithm based novel solution approach for multi-objective pollution-routing problem. *Computers & Industrial Engineering* 130 (2019), 757–771.

[22] Said Salhi and Graham K Rand. 1993. Incorporating vehicle routing into the vehicle fleet composition problem. *European Journal of Operational Research* 66, 3 (1993), 313–330.

[23] Christos D Tarantilis, Chris T Kiranoudis, and Vassilios S Vassiliadis. 2004. A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *European Journal of Operational Research* 152, 1 (2004), 148–158.

[24] Paolo Toth and Daniele Vigo. 2002. *The vehicle routing problem*. SIAM.

[25] Niaz A Wassan and Ibrahim H Osman. 2002. Tabu search variants for the mix fleet vehicle routing problem. *Journal of the Operational Research Society* 53, 7 (2002), 768–782.

[26] Hande Yaman. 2006. Formulations and valid inequalities for the heterogeneous vehicle routing problem. *Mathematical programming* 106 (2006), 365–390.

Session Session 4B: Telecommunication Networks
Tuesday 12 March 2024, 15:30-17:00
Q012

In-Band Network Telemetry for Efficient Congestion Mitigation

Youcef Magnouche, Sébastien Martin, Jeremie Leguay, Paolo Medagliani
Huawei Technologies France, Paris Research Center, France.
firstname.lastname@huawei.com

ABSTRACT

Tactical traffic engineering solutions are a must to adapt traffic steering when unexpected congestions occur. While centralized solutions are already available to solve congestion issues, they can be too slow and not suitable for some deployment scenarios. To address this issue, a distributed congestion mitigation mechanism that leverages Segment Routing (SR) to offload traffic away from congested links over alternative paths has been proposed. However, to accurately re-route traffic while not inducing other congestions, it requires fresh information about link loads inside alternative paths. In this paper, we propose to rely on the computation of additional paths, called "monitoring paths", that can be used to collect link loads efficiently. We investigate the associated optimization problem to decrease the number of paths used for monitoring. Also, thanks to Bilevel optimization, we show that the load information of several links can be recovered without monitoring. Results show that the overhead can be drastically reduced.

KEYWORDS

OSPF, Shortest-Path, Monitoring, Optimization, Segment-Routing, Path-Tracing, Network Telemetry, Bilevel optimization.

1 INTRODUCTION

As IP networks are continuously increasing in traffic, scale and complexity, service providers must carefully plan and design their networks to anticipate network evolution, e.g. traffic or failure scenarios, and meet custom requirements, e.g. in terms of Quality of Service (QoS) or routing requirements. In order to keep the network management as simple as possible, most of the traffic is routed across the network following the shortest path given by the Open Shortest Path First protocol (OSPF) [8], a routing protocol that works by flooding Link State Advertisement (LSA) information throughout the network. This information includes the cost of each link, its capacity, as well as some performance metrics about the link utilization. Routers use these metrics to compute and update their routing strategies.

However, in the case of unexpected link failures, the original planning may no longer hold, as traffic may be redirected to the post-convergence paths, leading to excessive use of some links, i.e., to congestions. This calls for fast reaction mechanisms to mitigate these congestions in less than 50 ms. As the interaction with the centralized controller is not suitable, due to the slow communication with devices, it is preferable to take decisions locally at node level. In [1], the authors propose a distributed congestion mitigation (CM) mechanism using Segment Routing (SR) to load balance traffic from a congested link to different alternative paths, as soon as a router detects congestion over an outgoing interface. However, if the load balancing weights are not properly tuned, the rerouted traffic can introduce new congestion in other parts of the network. Therefore, routers need to get accurate loads of the links in each alternative path, to decide how much traffic to reroute over them.

A solution to get accurate information can be provided by in-network telemetry [18], which aims to collect data from devices at high speed and in real-time. In particular, In-band Network-wide Telemetry (INT), or In-situ Operations, Administration, and Maintenance (IOAM) [2, 12, 19] embeds the telemetry information in the header of user packets or probe packets [11] to perform end-to-end or hop-by-hop measurements. To collect link loads over alternative paths, we can use Path Tracing (SR-PT) [9, 14] or iFit [15], i.e., two candidate solutions to provide a record of end-to-end delay, per-hop delay, and load on each egress interface along the packet delivery path.

Unfortunately, the number of alternative paths may be very high as routers maintain k alternative paths per destination and outgoing link (potentially congested). As explained in [5], path-based measurements are prohibitive due to the massive number of existing paths inside a network. First, probing packets have a maximum size (e.g. 1500 B), and therefore, if the path to be monitored is too long, it is not possible to collect telemetry data within a single packet. Second, as paths often overlap, a significant amount of collected information is redundant. This calls for the design of a new mechanism to improve the efficiency of alternative paths monitoring so that a more accurate reaction can be taken, avoiding introducing cascade congestions in remote links in the network.

In this paper, we investigate the computation of a small additional subset of paths, referred to as *monitoring paths*. These paths, which are deployed by each node, are used to collect remote link load information over the links used by its alternative paths (i.e., for all the destinations). The number of monitoring paths must be (i) much smaller than the number of alternative paths, and (ii) allow monitoring of all links in the alternative paths. In addition, our solution allows using some alternative paths for monitoring. As they are already deployed inside the routing table of a node, they can be immediately used at congestion time for faster reaction. In order to further reduce the monitoring overhead, we also show that it is possible to skip the monitoring of some links, without any loss of information.

In this paper, we provide the following contribution:

- We introduce a new set of paths, referred to as *monitoring paths*, used for hop-by-hop measurements.
- We formulate the monitoring paths computation problem using an Integer Linear Programming (ILP) model.
- We show that the monitoring paths computation problem is NP-Hard.

- We exploit Bilevel optimization to design partial monitoring paths, still guaranteeing full measurement.
- We perform extensive computational experiments to evaluate the performance of our algorithm compared to a "Naive approach" that consists in monitoring all alternative paths. We show that our solution decreases, the average number of paths for monitoring by up to 68% and the average number of monitored links by up to 71%.

The paper is organized as follows. In Sec. 2, we introduce the state of the art, in Sec. 3, we detail the considered use case. In Sec. 4, we provide a mathematical model for the computation of the monitoring paths. In Sec. 5 we show how a partial monitoring is enough to recover the load of all links in the alternative paths. In Sec. 6, we show the efficiency of our algorithm. Finally, Sec. 7 concludes this paper.

2 RELATED WORK

Several works in the literature have proposed routing optimization to mitigate congestion. In very recent works, such as [3, 4], the authors suggest leveraging mid-point SR optimization to mitigate congestion in the network. This approach is based on a centralized controller to compute alternative SR policies for congestion mitigation, which may not be desirable for scalability, fault tolerance, or commercial reasons. In [1], authors proposed a distributed mechanism based on SR. When a router detects congestion on a link, a portion of the traffic can be automatically offloaded and load balanced over a set of alternative paths using UCMP (Unequal Cost Multi Paths). The goal is to select lightly loaded paths to reroute a maximum of traffic, mitigate the congestion, and avoid creating new congestion elsewhere in the network. Our paper provides a distributed solution to optimize the collection of link loads, in real-time, over alternative paths for this type of mechanism.

Several works in the literature have proposed solutions for the computation of monitoring paths computations. In [16], authors develop a heuristic called "Graph Partitioned INT" so that a centralized controller can organize path measurements to cover all the nodes in the network, guarantee the freshness of telemetry information, and minimize redundancy. In [13], authors developed an algorithm to generate, at the controller, non-overlapped INT paths that cover the entire network with a minimum path number. In [5], authors investigate the computations of probing cycles that collect telemetry information over time employing a MILP model and a mathematical-based heuristic.
Our paper proposes a distributed mechanism to organize the collection of link data from a given router. In addition, it leverages as many as possible alternative paths for a smooth transition when congestions happen and further reduces the overhead with the partial collection.

3 USE CASE: MONITORING FOR CONGESTION MITIGATION

Typical service provider networks are configured to support reliability (up to 1-link failure) and QoS satisfaction (low MLU). As shown in Fig. 1, the backbone network is composed of Provider (P) nodes, i.e., nodes belonging to the same Internet Service Provider (ISP). The backbone network receives traffic from different sites, which

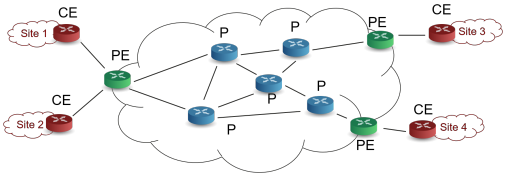


Figure 1: Example of a network with "CE" nodes attached to sites, "P" nodes in the ISP backbone, and "PE" nodes interconnecting "P" and "CE" nodes.

are interconnected to the service provider network via Customer Edge (CE) routers. The "PE" nodes interconnecting "P" and "CE" nodes are referred to as Provider Edge (PE) routers. In this backbone network, the "PE" nodes act as both sources and destinations of traffic aggregates. In order to keep the routing plane as simple as possible, flows follow the OSPF path, i.e., the shortest path between each pair of "PE" nodes. We point out that, in this paper, we only consider Segment Routing (SR) Best Effort (BE) traffic following the OSPF shortest path.

In the case of a link failure, the node detecting the failure broadcasts an LSA message to all the other links in the network to notify them of the network change. Each node independently computes a new OSPF shortest path tree to route the traffic avoiding the failed link. However, after that the network has reconverged, it may no longer guarantee low MLU and congestion, i.e. links whose load exceeds a given threshold, may appear. For this reason, it is necessary to implement efficient congestion mitigation mechanisms that locally react to congestions in less than 50ms, i.e. without requiring any interaction with an external controller.

A reference solution, that we will consider throughout this paper, is presented in [1], where the authors present a congestion mitigation mechanism that load balances traffic away from the link whose load is above 70%. The traffic is rerouted over k alternative paths that allow reaching the original destination. In this preliminary work, the authors only consider the link load of the congested interface. However, efficient congestion mitigation requires the knowledge of the load of all the links in the alternative paths, in order to avoid introducing remote congestions due to the unawareness of remote link loads.

As the reception of an LSA denotes that a network change has occurred, each node that receives an LSA message knows that congestion may happen. For this reason, after that it has updated its routing tables and converged to the new OSPF tree, it can deploy some extra paths, referred to as *monitoring* paths. As for SR-PT paths, the node sends probe packets through the monitoring paths to collect statistics about the load of all the links that belong to its alternative paths towards each destination. In this way, the node can collect all the information that it needs to make better load-balancing decisions in the case of congestion. In order to reduce the number of additional monitoring paths, some alternative paths can be chosen for monitoring purposes. As the failing link is not known a priori, the monitoring paths for each failure can be pre-computed offline, stored in the devices, and activated only when needed.

Once congestion mitigation is initiated, the node calculates the split ratio for each alternative path by considering both the local and the remote link loads. We neglect this computation as out of scope for this paper. In order to reroute traffic, an explicit Segment Routing

ID (SID) list is encapsulated in the header of the packets forwarded over alternative paths. This traffic, which is called "engineered", is no longer following the OSPF shortest path.

As soon as the congestion issue is resolved (i.e., the utilization of the egress link falls below a given threshold, i.e. 30%), the node stops the mitigation process and uninstalls the monitoring paths.

4 MONITORING PATHS COMPUTATION

The network can be modelled as a graph $G = (V, A)$, where V is the set of nodes and A is the set of arcs (links). Nodes of $R \subset V$ represent the set of "PE" nodes that generate the traffic in the backbone network, as they receive traffic from the "CE" nodes. The graph of "P" nodes, i.e., restricted to nodes of $V \setminus R$, corresponds to the *backbone network*.

The traffic, between two nodes $u, v \in V$ in the network, follows the shortest path with respect to OSPF weights. This path is called, *OSPF path*, denoted by p_u^v. Let S be the set of all shortest paths between every pair of "PE" nodes in R. Let $\bar{A}$ be the set of links belonging to OSPF paths, i.e., $\bar{A} = \bigcup_{u,v \in V} p_u^v$. In the following, we denote by $\delta^+(v) \subseteq A$ (resp. $\delta^-(v)$) the outgoing (resp. ingress) links of $v \in V$. Let $\delta(v) = \delta^+(v) \cup \delta^-(v)$. In this paper, we investigate the monitoring paths computation problem from the point of view of one arbitrary node $u^* \in V$. For a "PE" node $r \in R \setminus \{u^*\}$ and every arc $\bar{a} \in \delta(u^*)$, let $P_{\bar{a}}^r$ be the set of $k \in \mathbb{N}$ alternative paths between u^* and r avoiding $\bar{a}$. These paths are designed by the network operator (not necessarily shortest-paths) to reroute traffic in case of congestion. Let $P^r = \bigcup_{\bar{a} \in \delta(u^*)} P_{\bar{a}}^r$ be the set of all alternative paths to destination r and $A' = \bigcup_{r \in R} \bigcup_{p \in P^r} p$ be the set of all links in the alternative paths.

4.1 Problem definition

The monitoring paths computation problem consists in computing at most $q \in \mathbb{N}$ paths between u^* and "PE" nodes R such that:

- each link in the alternative paths, nonadjacent to u^* (links of $A' \setminus \delta(u^*)$), is monitored,
- the length (number of hops) of each monitoring path is at most $L_{\max} \in \mathbb{N}$,

under the following multi-objective function:

1) minimize the number of monitoring paths,
2) maximize the number of alternative paths used for monitoring,
3) minimize the total monitoring paths cost. The link costs may be used to prioritize some links for monitoring or to minimize the number of hops in the paths.

We consider a weighted sum of objective functions, by assigning a weight $w^1 \in \mathbb{R}^+$ to objective 1), $w_p^r \in \mathbb{R}^+$ to objective 2) and w_a in objective 3) for every link $a \in A$. Let $Q = \{1, \ldots, q\}$. Note that "Paths used for monitoring" represent the union of monitoring paths and a subset of alternative paths used for monitoring.

4.2 Complexity

Theorem 4.2.1. *The monitoring paths computation problem is NP-hard.*

Proof. We propose a polynomial reduction from the Hamiltonian path problem, known to be NP-complete [10], that consists, given a graph $G = (V, A)$, in computing a path, between a source s and a destination t, in G crossing all vertices in $V \setminus \{s, t\}$ exactly once. We construct a graph $G' = (V', A')$ from G, as follows:

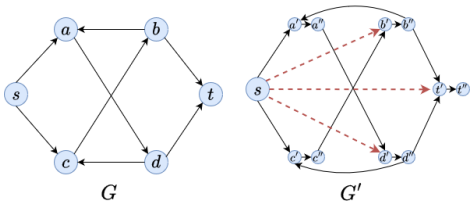


Figure 2: Graph transformation

- replace each node $v \in V \setminus \{s\}$ by two nodes v' and v'' connected by a link (v', v'') of weight 0,
- for each link $(u, v) \in A$, such that $u \in V \setminus \{s\}$ and $v \in V \setminus \{s\}$ add a link (u'', v') with $w_{(u'',v')} = 1$,
- for each $v \in V \setminus \{s\}$ add a link (s, v') of weight $w_{(s,v')} = |A| + |V|$, if $(s, v) \notin A$ and $w_{(s,v')} = 0$, otherwise.

Let $R = \{v'' \mid \forall v \in V \setminus \{s\}\}$ be the set of "PE" nodes. For all $r \in R$, let $p^r = \{(s, r'), (r', r'')\}$ be an alternative path. Consider weight $w_p^r = |A| + |V|$ and $w^1 = 0$. See Fig. 2. For $q = 1$, solving the monitoring paths computation problem in G' allows us to solve the Hamiltonian path problem in G. Indeed, since $\delta^+(t) = \emptyset$, the monitoring path crosses every link (r', r'') $\forall r \in V \setminus \{s\}$, and the result follows. □

4.3 Mathematical model

Let $z_p^r \in \{0, 1\}$ be a binary variable that equals 1 if alternative path $p \in P^r$ between $u^* \in V$ and $r \in R$ is used for monitoring and 0 otherwise. Let $y_i \in \{0, 1\}$ be a binary variable that equals 1 if monitoring path $i \in Q$ is considered in the solution, and 0 otherwise. Let x_a^i be a binary variable that equals 1 if link $a \in A$ belongs to monitoring path $i \in Q$ and 0 otherwise.

The monitoring paths computation problem is equivalent to the following Integer Linear Program (MPCP):

$$\min \quad \sum_{i \in Q} (w^1 y_i + \sum_{a \in A} w_a x_a^i) - \sum_{r \in R} \sum_{p \in P^r} w_p^r z_p^r \tag{1}$$

$$\sum_{a \in \delta^+(v)} x_a^i - \sum_{a \in \delta^-(v)} x_a^i = \begin{cases} y_i & \text{if } v = u^*, \\ -y_i & \text{if } v = r^*, \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in Q, v \in V, \tag{2}$$

$$\sum_{a \in A[W]} x_a^i \leq |W| - 1 \qquad \forall W \subseteq V, \forall i \in Q, \tag{3}$$

$$\sum_{i \in Q} x_a^i + \sum_{r \in R} \sum_{p \in P^r | a \in p} z_p^r \geq 1 \qquad \forall a \in A' \setminus \delta(u^*), \tag{4}$$

$$\sum_{a \in A} x_a^i \leq L_{\max} \qquad \forall i \in Q. \tag{5}$$

where r^* is a dummy node connected to each "PE" node $r \in R$ by the following dummy link (r, r^*). Constraints (2)-(3) represent the flow conservation equalities and sub-tour elimination inequalities. They allow computing the monitoring paths. Constraints (4) ensure monitoring every link in the alternative path by at least one path. Note that, links adjacent to u^* can be unmonitored. Finally, Constraints (5) bound the number of hops in the monitoring paths.

INOC 2024, March 11 - 13, 2024, Dublin, Ireland

5 PARTIAL LINKS MONITORING

In this section, we show that it is possible to get the measurements of all links in the alternative paths without monitoring all of them. This allows us to decrease the number of paths used for monitoring and the number of links to be monitored. For example, consider the graph in Fig. 3 where the weights on the links represent the link cost. The graph contains 8 nodes including 3 "PE" nodes (in blue).

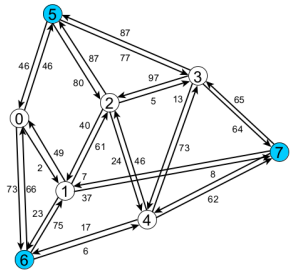


Figure 3: Network with 8 nodes. Nodes 5, 6 and 7 represent the "PE" nodes. Weights or links represent the TE Costs.

Fig. 4 represents the shortest paths between "PE" nodes. In this example, we consider node 3 for the congestion mitigation. Node 3 is aware of the link loads of all its ingress/outgoing links. Fig. 5 displays two alternative paths between node 3 for every "PE" node.

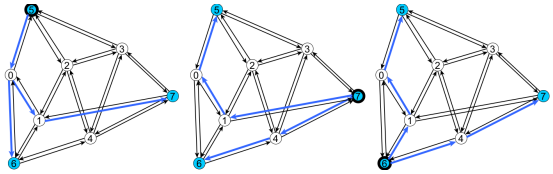


Figure 4: Shortest path trees between "PE" nodes.

Finally, Fig. 6 shows two monitoring paths, the first one is between nodes 3 and 6 and the second one is between 3 and 7.

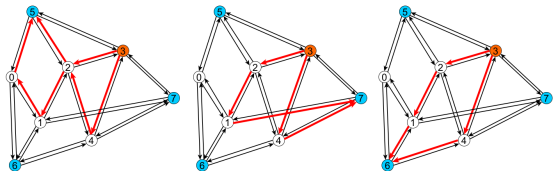


Figure 5: 2 Alternative paths between node 3 and each "PE".

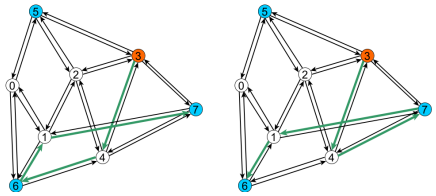


Figure 6: Monitoring paths (in green) to get loads of all links in the alternative paths given in Fig. 5 (in red).

It is easy to see that the number of monitoring paths is smaller than the number of alternative paths. Moreover, the number of links in the monitoring paths is smaller than those of the alternative paths. Although the monitoring paths do not cover all links in the alternative paths, they are enough to get the measurement of all links in the alternative paths. As links $(2,1)$, $(2,5)$, $(4,2)$ do not appear in the OSPF paths, they cannot route any traffic. Hence, they are not monitored even if they belong to alternative paths. Links $(1,0)$, $(0,5)$ are not monitored even if they appear in the alternative and OSPF paths. The traffic over these two links can be deduced thanks to the measurements of links $(6,1)$, $(7,1)$. Indeed, since link $(2,1)$ belongs to no OSPF path, and all OSPF paths crossing $(1,0)$ do not cross $(0,6)$ the traffic load over links $(1,0)$ and $(0,5)$ equals the loads sum of $(6,1)$ and $(7,1)$.

5.1 Mathematical model

The partial monitoring paths computation problem (PMPCP) is a variant of MPCP, described in the previous section. In contrast with MPCP, in this version, we relax Constraints (4) forcing the monitoring paths to cross over all links in the alternative paths. The monitoring on a link in the alternative path can be skipped if 1) it belongs to no OSPF path, or, 2) it is able to recover its link load only based on the monitoring of other links. We refer to the second type links as the *"Recovered links"*. For that, we need to ensure that all traffic matrices satisfying the loads on the monitored links, give the same load on every Recovered link. The PMPCP can, then, be tackled as a Bilevel optimization problem [17], where the leader selects the links to monitor and the follower tries to find two different traffic matrices giving the same load on monitored links (decided by the leader) but with different loads on at least one recovered link. From MPCP, we consider additional decision variables for the leader as follows: let $t_a \in \{0,1\}$ be a variable that equals 1 if link $a \in A$ is monitored and 0 otherwise. Moreover, we consider further variables for the follower: let $q_a \in \mathbb{R}^+$ be the load difference between the two traffic matrices on link $a \in A$. And, let $x_p^i \in \mathbb{R}^+$ be the amount of traffic over the OSPF path $p \in S$ associated with traffic matrix $i = \overline{1,2}$.

The partial monitoring paths computation problem is equivalent to the following Bilevel mathematical model (PMPCP)

$$\min \quad \sum_{a\in A} w^0 t_a + \sum_{i\in Q}\Big(w^1 y_i + \sum_{a\in A} w_a x_a^i\Big) - \sum_{r\in R}\sum_{p\in P^r} w_p^r z_p^r \tag{6}$$

$$\textit{Constraints} \qquad (2),(3),(5)$$

$$x_a^i \le t_a \qquad \forall a \in A, i \in Q, \tag{7}$$

$$z_p^r \le t_a \qquad \forall r \in R, p \in P^r, a \in p, \tag{8}$$

$$t_a \le \sum_{i\in Q} x_a^i + \sum_{r\in R}\sum_{p\in P^r | a\in p} z_p^r \qquad \forall a \in A, \tag{9}$$

$$\vartheta(t) \le 0, \tag{10}$$

$$\text{where} \qquad \vartheta(t) = \max \sum_{a\in A'\cap \bar{A}} q_a \tag{11}$$

$$\alpha_a : \; t_a \sum_{p\in S\ni a} x_p^1 = t_a \sum_{p\in S\ni a} x_p^2 \qquad \forall a \in A, \tag{12}$$

$$\beta_a : \; -\sum_{p\in S\ni a} x_p^1 + \sum_{p\in S\ni a} x_p^2 \le -q_a \qquad \forall a \in A, \tag{13}$$

$$\theta_a^i : \; \sum_{p\in S | a\in p} x_p^i \le b_a \qquad \forall a \in A, i = \overline{1,2}. \tag{14}$$

where $w^0 \in \mathbb{R}_+$ represents the weight given to the number of monitored links in the objective function, $b_a \in \mathbb{R}^+$ represents the

bandwidth capacity of link $a \in A$ and α, β and θ represent the dual variables associated with Constraints (12), (13) and (14), respectively. Constraints (7)-(9) guarantee that if a link is monitored it must belong to a new monitoring path or an alternative path used for monitoring. Thanks to the follower, Constraints (10) ensure that all traffic matrices satisfying the loads on the monitored links, give the same load on the recovered links. For the follower, Objective (11) maximizes the total link load differences between two traffic matrices. Note that only links belonging to both OSPF and alternative paths are considered in this objective function. Constraints (12) ensure that, for every monitored link (i.e., $t_a = 1$), the two matrices give the same load. Constraints (13) computes the load difference for every link and finally, Constraints (14) guarantee that each traffic matrix respects the link capacities.

5.2 Single-level model

In the literature, several methods have been developed to solve the Bilevel optimization problems. One approach, called "Single-Level Reduction", consists in including the KKT conditions of the follower as constraints of the leader problem. In our case, we can exploit the fact that the follower is a maximization problem and $\vartheta(t) \geq 0$ for any $t \in \{0,1\}^{|A|}$ (as $q(t) \geq 0$), to guarantee the optimality of the follower. Let us consider the dual of the follower, given as follows:

$$\min \quad \sum_{a \in A} \sum_{i \in Q} \theta_a^i b_a \tag{15}$$

$$\sum_{a \in p} (\alpha_a t_a - \beta_a + \theta_a^1) \geq 0 \qquad \forall p \in S, \tag{16}$$

$$\sum_{a \in p} (-\alpha_a t_a + \beta_a + \theta_a^2) \geq 0 \qquad \forall p \in S, \tag{17}$$

$$\beta_a \geq 1 \qquad \forall a \in A' \cap \bar{A}, \tag{18}$$

$$\beta_a, \theta_a \geq 0 \qquad \forall a \in A. \tag{19}$$

Claim 5.2.1. *All variables θ can be set to* 0.

Proof. From Constraints (10), $\vartheta(t) = \sum_{a \in A} \sum_{i \in Q} \theta_a^i b_a = 0$. Since $\theta \geq 0$ and $b_a \geq 0$ for all $a \in A$, the result follows. □

The single-level model can be obtained from PMPCP by replacing (10)-(14) by the constraints of the dual of the follower together with $\sum_{a \in A} \sum_{i \in Q} \theta_a^i b_a = 0$. By Claim 5.2.1, the single-level model is equivalent to the following the mixed integer non-linear model:

$$\min \quad \sum_{a \in A} w^0 t_a + \sum_{i \in Q} (w^1 y_i + \sum_{a \in A} w_a x_a^i) - \sum_{r \in R} \sum_{p \in P^r} w_p^r z_p^r \tag{20}$$

$$\textit{Constraints} \qquad (2), (3), (5)$$

$$\textit{Constraints} \qquad (7), (8), (9)$$

$$\sum_{a \in p} \alpha_a t_a = \sum_{a \in p} \beta_a \qquad \forall p \in S, \tag{21}$$

$$\beta_a \geq 1 \qquad \forall a \in A' \cap \bar{A}, \tag{22}$$

$$\beta_a \in \mathbb{R}^+ \qquad \forall a \in A. \tag{23}$$

The above model can be linearized easily as follows.

Claim 5.2.2. *For $a \in A$, let $f \in \mathbb{R}$ be a new variable. For an enough big value $M \in \mathbb{R}^+$, Constraints* (21) *can be replaced by*

$$\sum_{a \in p} f_a = \sum_{a \in p} \beta_a \qquad \forall p \in P,$$

$$\alpha_a - M(1 - t_a) \leq f_a \leq \alpha_a + M(1 - t_a) \qquad \forall a \in A,$$

$$-M t_a \leq f_a \leq M t_a \qquad \forall a \in A.$$

6 NUMERICAL EXPERIMENTS

We now evaluate the monitoring paths computation on randomly generated networks. For benchmarking, we compare our solution to the "Naive approach" which consists in monitoring all alternative paths with path-tracing. Except for Figure 9, all experiments are on the partial links monitoring given in Section 5. In our tests, the models are solved using IBM ILOG CPLEX 12.6 solver [6]. All implementations are in Python on a machine with an Intel(R) Xeon(R) CPU E5-4627 v2 at 3.30GHz and 504GB RAM, running Linux 64 bits. A maximum of 32 threads has been used for CPLEX, and a time-limit of 3 hours. The instances have been generated by varying the following parameters: the number of nodes: $\{50, 100, 200\}$, the number of alternative paths k: $\{2, 4\}$, the number of "PE" nodes: $\{30\%, 60\%\}$ of the number of nodes, the network density: $\{20\%, 40\%\}$, the maximum number of hops in the monitoring paths $L_{\max}$: $\{6, 12\}$ and the maximum number of monitoring paths q: $\{5, 10\}$.

We consider the following weights in the objective function

- $w^0 = 10^3, \quad w^1 = 10^2, \quad w_a = 10^2 \ \forall a \in A,$
- $w_p^r = 10^2 \times (|p| + 1), \ \forall r \in R$ and $p \in P^r$.

These weights give the highest priority to minimizing the number of monitored links. Throughout this section, some results are presented in the form of box plots that account for the points between the 1st (Q_1) and the 3rd quartile (Q_3), while the bar in the middle of the box plot represents the median (Q_2). The whiskers represent $Q_1 - 1.5IQR$ and $Q_3 + 1.5IQR$, where $IQR = Q_3 - Q - 1$. The points represent the outliers. The OSPF paths have been computed between each pair of nodes in the network using the Dijkstra algorithm [7]. For a given node $u \in V$, for every "PE" node $r \in R$, and for every outgoing arc $\bar{a} \in \delta(u)$, the k associated alternative paths are generated by solving mathematical model (ILP). The model maximizes the disjointness between the k paths without crossing $\bar{a}$.

Fig. 7 displays the reduction ratio on the number of monitored links when using the monitoring paths described in Section 5 compared to the "Naive approach". Each color corresponds to a distinct parameter, each of which has two values (see above). In Plot 7.(a), on instances with 50 nodes, we notice that with few "PE" nodes (i.e., 30% of number of nodes), the improvement on the number of monitored links is much higher (83% instead of 65%). This is to be expected, as the greater the number of "PE" nodes, the more alternative and OSPF paths there are. On the other hand, we notice that a high value of k gives a better improvement in the number of monitored links. Indeed, when the number of alternative paths is high, the links are crossed multiple times. This leads to redundant monitoring via the "Naive solution". These two behaviours are similar on networks with 100 and 200 nodes (Plots 7.(b) and 7.(c)). On 50 node instances, we also notice that the density, the maximum monitoring path length "P-Length" and the maximum number of monitoring paths "M-Paths" positively impact the number of monitored links. These three parameters allow to design of efficient monitoring paths, avoiding repeated crossed links. On instances with 100 and 200 nodes, The behaviour is ambiguous, and due to a significant optimality gap, a definitive conclusion cannot be drawn. Fig. 8 displays a comparison of the following ratios:

- Recovered Links "R": $\frac{\#\text{Recovered links}}{|A'|} \times 100$
- Optimality Gaps "G": $\frac{\text{Upper bound} - \text{Lower bound}}{\text{Lower bound}} \times 100$
- Paths for monitoring "M": $\frac{\#\text{Alternative paths} - \#\text{Paths for monitoring}}{\#\text{ Alternative paths}} \times 100$

for instances of 50, 100 and 200 nodes. The higher the number of nodes, the lower the ratio of recovered links. This is due to

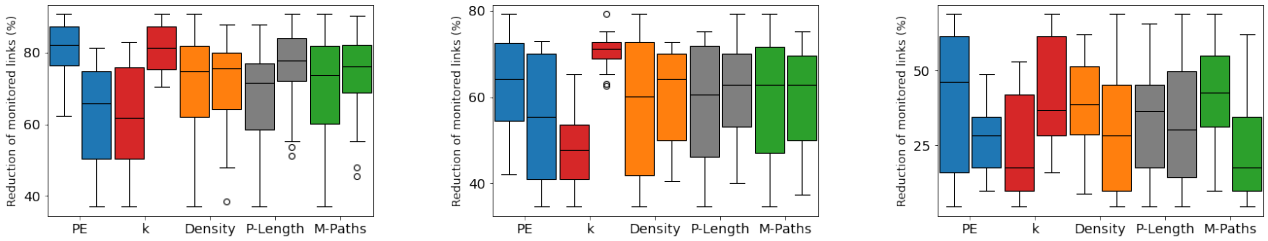


Figure 7: Reduction ratio on the number of monitored links (%) for 50, 100 and 200 nodes respectively.

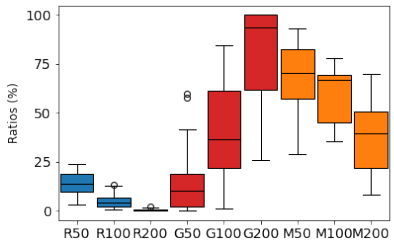


Figure 8: Ratio of Recovered links, Optimality gap and Reduction of paths used for monitoring.

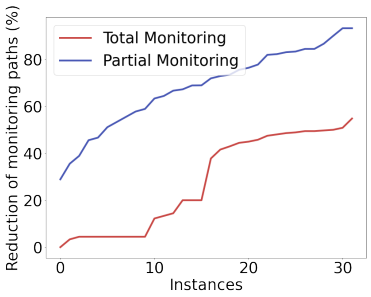


Figure 9: Reduction comparison of the number of paths used for monitoring between Total and Partial monitoring.

the optimality gap. Indeed, this latter increases with the number of nodes impacting the quality of the solutions. We see the same impact on the number of paths used for monitoring. We save 68%, 58% and 39% of paths, in average, for instances of 50, 100 and 200 nodes, respectively. These results depend mainly on the quality of the solutions obtained within the time limit. The last Fig. 9 shows a comparison of the reduction ratio on the number of paths used for monitoring between "Total Monitoring" version given in Section 4 and the "Partial Monitoring" given in Section 5. The second version performs much better than the first one. This was expected since in the second version, the paths used for monitoring can avoid crossing through Recovered links and Links out of OSPF paths.

7 CONCLUSION

In this paper, we propose a way to exploit the in-band telemetry for accurate congestion mitigation. We have shown that a lot of redundancies in the measurements appear when alternative paths are monitored. We proposed to design extra "monitoring paths" to help drastically decrease the number of monitored links. Moreover, we have shown that up to 25% of links in alternative paths can be recovered without monitoring. Indeed, the load over these links can be recovered from the measurement of others. From a practical perspective, an efficient heuristic needs to be developed to solve the problem in a short time in order to be used in practice.

REFERENCES

[1] Lirong Bai, Liang Zhang, Geng Zhang, Lin Zhang, Paolo Medagliani, and Sébastien Martin. 2023. A Distributed Congestion Mitigation Mechanism Based on Neighboring Nodes Traffic Steering. In Proc. IEEE NoF.

[2] Frank Brockners. 2017. Next-gen Network Telemetry is Within Your Packets: In-band OAM. http://events17.linuxfoundation.org/sites/events/files/slides/In-band_OAM.pdf.

[3] Alexander Brundiers, Timmy Schüller, and Nils Aschenbruck. 2022. Midpoint Optimization for Segment Routing. In Prox. IEEE INFOCOM. 1579–1588.

[4] Alexander Brundiers, Timmy Schuller, and Nils Aschenbruck. 2023. Tactical Traffic Engineering with Segment Routing Midpoint Optimization. In IFIP Networking.

[5] Ariel G. Castro, Arthur F. Lorenzon, Fábio D. Rossi, Roberto I. T. da Costa Filho, Fernando M. V. Ramos, Christian E. Rothenberg, and Marcelo C. Luizelli. 2021. Near-Optimal Probing Planning for In-Band Network Telemetry. IEEE Communications Letters 25, 5 (2021), 1630–1634.

[6] CPLEX. [n.d.]. https://www.ibm.com/products/ilog-cplex-optimization-studio.

[7] Edsger W Dijkstra. 2022. A note on two problems in connexion with graphs. In Edsger Wybe Dijkstra: His Life, Work, and Legacy. 287–290.

[8] Dennis Ferguson, Acee Lindem, and John Moy. 2008. OSPF for IPv6. RFC 5340. https://doi.org/10.17487/RFC5340

[9] Clarence Filsfils, Ahmed Abdelsalam, Pablo Camarillo, Mark Yufit, Thomas Graf, Yuanchao Su, Satoru Matsushima, Mike Valentine, and Amit Dhamija. 2023. Path Tracing in SRv6 networks. Internet-Draft draft-filsfils-spring-path-tracing-05. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-filsfils-spring-path-tracing/05/ Work in Progress.

[10] Michael R Garey and David S Johnson. 1979. Computers and intractability. A Guide to the Theory of NP-Completeness (1979).

[11] The P4.org Applications Working Group. 2020. In-band network telemetry (INT) dataplane specification version 2.1. https://github.com/p4lang/p4-applications/blob/master/docs/INT_latest.pdf.

[12] Youngho Kim, Dongeun Suh, and Sangheon Pack. 2018. Selective in-band network telemetry for overhead reduction. In Proc. IEEE CloudNet. IEEE, 1–3.

[13] Tian Pan, Enge Song, Zizheng Bian, Xingchen Lin, Xiaoyu Peng, Jiao Zhang, Tao Huang, Bin Liu, and Yunjie Liu. 2019. Int-path: Towards optimal path planning for in-band network-wide telemetry. In Proc. IEEE INFOCOM.

[14] Leonardo Rodoni. 2022. Delay Measurement, Path Tracing, and Telemetry Data Correlation in Segment Routed Networks. Technical Report. ETH and Swisscom.

[15] J Shin and SK Telecom. [n.d.]. In-situ Flow Information Telemetry Framework draft-song-opsawg-ifit-framework-00.

[16] Goksel Simsek, Doğanalp Ergenç, and Ertan Onur. 2021. Efficient network monitoring via in-band telemetry. In Proc. IEEE DRCN.

[17] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. 2018. A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. IEEE Transactions on Evolutionary Computation 22, 2 (2018), 276–295.

[18] Lizhuang Tan, Wei Su, Wei Zhang, Jianhui Lv, Zhenyi Zhang, Jingying Miao, Xiaoxi Liu, and Na Li. 2021. In-band Network Telemetry: A Survey. Computer Networks 186 (2021), 107763. https://doi.org/10.1016/j.comnet.2020.107763

[19] Shaofei Tang, Deyun Li, Bin Niu, Jianquan Peng, and Zuqing Zhu. 2019. Sel-INT: A runtime-programmable selective in-band network telemetry system. IEEE transactions on network and service management 17, 2 (2019), 708–721.

Exploring quantum optimization for solving the PCI planning problem in 5G networks

Full Paper

Adriano Borges
CENTRE FOR ADVANCED STUDIES AND SYSTEMS - CESAR School
Recife, Pernambuco, Brazil
aab3@cesar.school

Pamela Bezerra
CENTRE FOR ADVANCED STUDIES AND SYSTEMS - CESAR School
Recife, Pernambuco, Brazil
ptlb@cesar.school

Erico Teixeira
CENTRE FOR ADVANCED STUDIES AND SYSTEMS - CESAR School
Recife, Pernambuco, Brazil
est@cesar.school

ABSTRACT

Physical Cell IDs (PCIs) are numerical identifiers crucial for distinguishing various antennas, or cells, within telecommunication networks like 5G. They play a vital role in facilitating the efficient connection of mobile devices to different cells, preventing issues such as interference. However, the growing scale of 5G networks, coupled with a limited pool of unique PCIs, allocating different PCI to adjacent cells is a challenge known as the PCI planning problem.

In this scenario, this article explores the use of Quantum Computing (QC) to solve the PCI planning problem. With remarkable advancements in recent years, QC has shown great potential for solving complex optimization problems. To discern the advantages QC could bring to PCI planning, we analyzed the performance of classical and quantum methods across diverse network configurations. Our results show that quantum methods yield solutions equivalent to exhaustive search but with substantially reduced execution time, opening new research opportunities in QC and telecommunications.

1 INTRODUCTION

Mobile phones can transmit and receive data by connecting to antennas, also known as cells, at specific frequencies. These cells are often distributed among different telecommunications towers and are identified through a number known as Physical Cell ID (PCI). It is essential to assign distinct PCI values to nearby cells to mitigate Inter-Cell Interference (ICI), i.e., using the same frequency band by adjacent cells. An efficient PCI allocation provides a high-quality communication service to many users, avoiding, for example, long cell allocation time.

However, the number of available PCIs is limited to only 1008 in current 5G networks, making the efficient allocation of these identifiers challenging, especially with these networks' growing density. This problem, known as the **PCI planning problem**, is an NP-complete combinatorial optimization problem, with recent works in the area proposing the use of heuristics, such as reuse distance [9] and Glowworm swarm optimization (GSO)[11]. For example, commercial tools such as Atoll by Forsk use Monte Carlo simulation [6] to solve the PCI planning problem.

The innovative work of Gui et al. [7], for example, proposes a new combinatorial optimization model to describe collision, confusion, and *mod q* interference comprehensively and quantitatively. The PCI planning problem was mapped as a Binary Quadratic Programming (BQP) model, and a Greedy algorithm was developed to configure PCIs automatically for each cell in the whole network. To evaluate the optimization performance of the proposed algorithm, numerical simulations were performed compared with the scheme implemented in the current network and the classical graph coloring algorithm. The experimental results demonstrated that the Greedy algorithm had a significant advantage in reducing the collision, confusion, and *mod* 3 interference in scenarios using 1131 cells and 30 PCI. The Greedy algorithm not only eliminates conflict and confusion completely but also reduces the mod 3 interference by 26.213% more than the baseline scheme and far more than the improvement ratio of 4.436% given by the classical graph coloring algorithm.

On the other hand, Quantum Computing (QC) has seen rapid advancement in recent years, with companies like IBM and D-Wave launching new computers almost every year and an estimated investment of US$ 38.6 billion worldwide in just 2023[1]. Among the various application possibilities, QC has excellent potential for solving combinatorial optimization problems, such as PCI planning, due to using quantum mechanical phenomena, such as superposition. This phenomenon implies the main difference between classical and quantum computers: while the former use the bit as the basic unit of information, which can be 0 or 1, quantum computers use the quantum bit (qubit), a linear combination of the base states 0 and 1, allowing more information processing with fewer units.

The first work to propose using QC in PCI planning is Boella et al. Using a simpler Quadratic Unconstrained Binary Optimization (QUBO) formulation, [4] executed experiments using the quantum computer from D-Wave to solve a PCI planning. QUBO is a mathematical representation that provides a powerful tool for formulating and solving certain types of problems in computer science, particularly those that are NP-hard. The formulation was applied to the PCI planning of 5G and 4G and compared to the legacy procedure, Fast Greedy Algorithm. To analyze the algorithm's potential, a series of tests using a sample set of 450 cells of the TIM network were performed, decreasing the number of Secondary Synchronization Signal (SSS) compared to the maximum. In fact, as the number of SSS decreases, the probability of violating the constraints increases.

[1] https://qureca.com/overview-of-quantum-initiatives-worldwide-2023/

In light of the aforementioned considerations, this paper further explores the potential of QC in solving the PCI planning problem. Building upon the formulation introduced by [7], we have devised classical and quantum algorithms to scrutinize their performance across diverse network configurations characterized by varying numbers of cells and PCI. The experiments were also performed using a D-Wave system quantum annealing-based computer.

To the best of our knowledge, this study is one of the pioneering endeavors to investigate the PCI planning problem within quantum machines comprehensively. Our investigation meticulously examines the influences of different parameters on the execution time and the quality of the obtained results (i.e., how close to the optimal value), thus contributing novel insights to the evolving landscape of quantum computing applications in telecommunications planning.

We performed experiments using three methods: exact brute force, Steepest Descent heuristics, and the hybrid CQM method. We systematically varied the number of cells and distributed PCIs across 66, exploring diverse cell arrangements and yielding 64 unique scenarios for each method. Our experiments show clear empirical evidence in favor of quantum computing solutions when comparing their performance with the exact methods and heuristics.

This paper is organized as follows: **Section 2** details the PCI planning problem; **Section 3** explains the QUBO model required to solve the problem; **Section 4** describes the experimental environment and algorithms used, while **Section 5** discuss the obtained results; finally, **Section 6** summarise the main conclusions and futures works of this research.

2 THE PCI PLANNING PROBLEM

The PCI is crucial in assisting User Equipment (UEs - the technical term for mobile devices) in identifying which cell to connect to among various signals within the same frequency. It comprises two parts: Primary Synchronization Signal (PSS), which can have values of 0, 1, or 2, and Secondary Synchronization Signal (SSS), with values ranging from 0 to 355. These elements combine to form the PCI using the formula $PCI = 3 * SSS + PSS$. In the 5G context, this results in a total of 1008 possible PCI values, allowing for reuse when conducting PCI planning in scenarios with a higher number of cells.

Due to the finite number of available PCIs, reuse becomes inevitable in networks with a cell count exceeding 1008. However, the wrong allocation of PCI will significantly increase the occurrence of ICI. To mitigate these ICIs effectively, it is imperative to thoroughly examine scenarios involving collisions, confusion, and *mod q* interference within neighboring cells (i.e., adjacent cells less than 1 km apart) operating on the same frequency. This comprehensive analysis is pivotal for optimizing PCI allocation strategies and enhancing overall network performance.

In regular use, as described by Figure 1 (a), UE navigates from one cell to another with different PCI. Collision may occur if some neighboring cells have the same frequency and PCI, Figure 1 (b). In this case, it is difficult for the UE to select which cell to address, as there are two different cells with the same PCI.

Confusion may occur with two or more neighbor cells sharing the same frequency and PCI, shown in Figure 1 (c), when the UE leaves a cell with a PCI and goes to a region with two cells with the same PCI.

Similar to the collision scenario, *mod q* interference may occur in some neighboring cells with the same frequency, as shown in Figure 1 (d); the PCI *mod q* value of one cell is equal to the PCI *mod q* value of other cells.

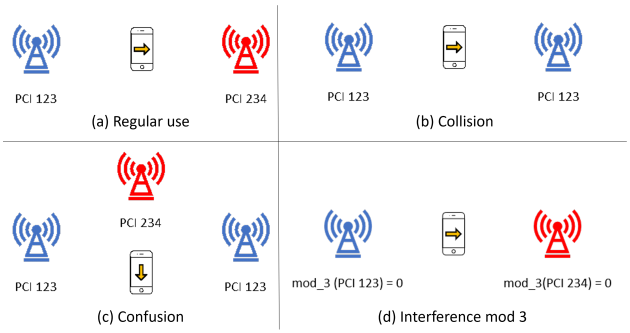


Figure 1: Types of Interferences

3 OPTIMIZATION MODEL

The Quadratic Unconstrained Binary Optimization (QUBO) model is applied to solve combinatorial optimization problems where the goal is to find the optimal binary values (0 or 1) for a set of variables to minimize a quadratic objective function. The optimize function can be formally defined by Equation 1:

$$\min F(x) = \sum_{i<j}^{N} Q_{i,j} x_i x_j + \sum_{i}^{N} Q_{i,i} x_i \tag{1}$$

, where Q is a NxN triangular matrix with real values, with the diagonal representing the linear weight terms, the off-diagonal the quadratic weight terms, and x is a vector of the binary variables.

This is an important formulation because real-world problems can be naturally transformed into QUBO form by providing a way to represent classical optimization problems as quantum problems. In addition, quantum annealer computers, like those offered by D-Wave[2], are designed to solve QUBO problems efficiently.

The QUBO formulation used in this work to address the PCI assignment problem was the same as that presented by Gui et al. [7], where the objective is to minimize collisions, confusion, and *mod q* interference.

This work assigns one binary variable for each pair composed by a cell i to PCI k:

$$x_{i,k} = \begin{cases} 1 & \text{if cell } i \text{ is associated to PCI } k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

For this formulation, a matrix A of dimensions $n \times n$, where n is the number of cells, is constructed, and each element $a_{i,j}$ represents the relationship between the cell in row i and the cell in column j. The value 1 will be assigned when they are neighbors and 0

[2]https://www.dwavesys.com/

otherwise. Furthermore, the value zero will be associated with the main diagonal.

In the same way, another matrix B is created, also of dimensions $n \times n$, which represents the state of confusion between the cells. Each element $b_{i,j}$ of B represents the relationship between the cell in row i and the cell in column j. The value 1 will be assigned when they are neighbors due to confusion, and 0 otherwise.

Finally, the matrix L of dimension $m \times q$ calculate *mod q* interferences of m PCIs. Each element $l_{i,j}$ indicates if the i-th PCI has *mod q* equal to j. The general expression of L is:

$$L = [\underbrace{I_q, I_q, \ldots, I_q}_{m/q\ items}]^T \tag{3}$$

, where I_q is the $q \times q$ identity matrix.

Thus, the QUBO model for PCI planning is described by the objective function

$$\begin{aligned} \min_{i,j} F = {} & \omega_1 \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} \sum_{k=1}^{m} x_{i,k}x_{j,k} \\ & + \omega_2 \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij} \sum_{k=1}^{m} x_{i,k}x_{j,k} \\ & + \omega_3 \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} \sum_{h=1}^{q} (\sum_{k=1}^{m} x_{i,k}l_{k,h}).(\sum_{k=1}^{m} x_{j,k}l_{k,h}) \\ \text{s.t.} \sum_{k=1}^{m} x_{i,k} = 1, \forall i \rightarrow & \sum_{i}^{n} \left(\sum_{k=1}^{m} x_{i,k} - 1\right)^2 \\ x_{i,k} \in \{0,1\}, \forall i,k & \end{aligned} \tag{4}$$

with the constraint

$$\sum_{k=1}^{m} x_{i,k} = 1, \forall i \rightarrow \sum_{i}^{n} \left(\sum_{k=1}^{m} x_{i,k} - 1\right)^2 \tag{5}$$

where ω_1, ω_2 and ω_3 are weighting factors, respectively, for the condition of collision, confusion, and *mod q* interference, n is the total of cells, m the number of PCI to be distributed, and the constraint that associates only one PCI to each cell leads to a penalty function to be added to the QUBO formulation.

4 ENVIRONMENT AND ALGORITHMS

As previously mentioned, PCI planning is an NP-complete combinatorial optimization problem, and several studies have proposed heuristics methods and other classical (i.e., non-quantum) strategies to solve it efficiently [9][11].

In our analysis, we used two classical optimization methods:

(1) **Exhaustive Search**, which is the easiest method for finding optimization solutions by testing all possible solutions. Although this method can always find the best solution, its use is not practical for large problems due to the exponential number of cases to be tested (combinatorial explosion). This study used the exhaustive search in small scenarios as a benchmark for the solution's value for the heuristic and quantum methods.
(2) **Gradient Descent**, a simple iterative heuristic that gradually moves toward a minimal local function based on its gradient. The gradient of any differentiable function represents how quickly it moves towards a local minimum. Therefore, the gradient descent method moves in the opposite direction, always selecting the largest step. For our analysis, this method was used as a benchmark for the algorithm execution time for the exhaustive and quantum methods.

4.1 Adiabatic Quantum Computing

Adiabatic Quantum Computing (AQC) [2] stands as a fundamental paradigm for quantum computation, drawing upon the adiabatic theorem [5] in quantum mechanics. According to this theorem, a quantum system will remain in its ground state if the associated Hamiltonian changes sufficiently slowly. In broad strokes, AQC commences with a simple Hamiltonian ($\hat{H}_0$) whose ground state is readily preparable. Over time, the Hamiltonian is smoothly changed to represent the problem Hamiltonian ($\hat{H}_f$) wherein the ground state encapsulates the solution to a computational problem. A parameterized time-dependent Hamiltonian describes the algorithms associated with AQC (Equation 6)

$$\hat{H} = A(t)\hat{H}_0 + B(t)\hat{H}_f \tag{6}$$

where $t \in [0, 1]$ and $A(t)$ and $B(t)$ are the functions that describe the interpolation between the Hamiltonians and that obey the generic boundary conditions given by

$$A(0) \neq 0, \quad B(1) \neq 0, \quad A(1) = B(0) = 0 \tag{7}$$

As the system undergoes evolution, $A(t)$ gradually decreases while $B(t)$ increases until, ultimately, the total Hamiltonian is solely defined by the term associated with $B(t)$. If the process is slow enough -achieving adiabatic conditions- the resultant state will correspond to the ground state of the final Hamiltonian of the system, which encodes a solution to the problem.

4.2 Quantum Annealing

Quantum annealing (QA) [10] represents a specific implementation of AQC, with the time evolution of the quantum system drawing inspiration from the classical annealing process in metallurgy, where a material is heated and slowly cooled to remove defects and optimize its structure. QA is a heuristic quantum approximation because the switch from $\hat{H}_0$ to $\hat{H}_f$ is determined heuristically, and the adiabatic conditions are not guaranteed. As a result, QA is particularly well-suited for encoding binary combinatorial optimization problems, expressed in Ising or QUBO form. These two representations are equivalent and can be readily transformed into each other through a simple change of basis. This flexibility makes QA a valuable approach for tackling a broad class of optimization challenges in quantum computing.

One of the most popular, widely used QA devices is the D-Wave System, featuring a quantum processor with a set of superconducting qubits arranged in a configuration analogous to a chain of Ising-type magnetic spins. In this platform, $\hat{H}_0$ is constructed by applying transverse magnetic fields, aligning the spins (qubits) in the direction of the field. The adiabatic interpolation (process $\hat{H}_0 \rightarrow \hat{H}_f$) unfolds by slowly reducing the intensity of the transverse field to zero. Simultaneously, the intensity of the couplings

between the qubits increases, facilitating the transition from the initial state to the final Hamiltonian.

5 COMPUTATIONAL EXPERIMENTS

As previously mentioned (Section 4.2), D-Wave Systems is the market leader in QA devices. Founded in 1999 in Canada, the company developed "the world's first commercially available quantum computer" with 128 qubits in 2011 [8]. Currently, its more powerful device, the Advantage[3], has 5000 qubits and an optimized topology, being capable of solving complex commercial problems with more than 1 million variables. Over the years, the company has accumulated more than 200 patents and 100 research publications in various areas, such as logistics and financial services.

D-Wave is not only a pioneer in quantum hardware but also in software and services. It developed its own Python-based open-source software development kit (SDK), the D-Wave Ocean [4]. Additionally, the company provides a cloud service, the D-Wave Leap[5], for real-time remote access to its devices. The experiments reported in this article were performed directly through the Leap interface.

The essential elements of a D-Wave solution are the *samplers* and *solvers*. A sampler is a process that samples low-energy states from objective functions to find the best solution. It takes a problem formulated as QUBO and returns a set of potential solutions represented as binary assignments (0s and 1s) for the variables in the model. These samplers run on a device known as a solver, which can be mainly classified into three types: (1) Classical (i.e., a classic computer), (2) Quantum (i.e., a QA computer), and (3) Hybrid, an architecture that explores the advantages of both types of resources (classical and quantum). In our experimental analyses, we've used two classical solvers, **Exact Solver** and **Steepest Decent**, which implement, respectively, the exhaustive search and a discrete version of gradient descent, and a Hybrid Solver Service (HSS) known as **Constrained Quadratic Model** (CQM).

5.1 Constrained Quadratic Models

CQM involves linear constraints, i.e., the constraint does not need to be translated into a penalty function. Thus, the restriction that associates only one PCI for each cell presented in Equation 5 can be used without the need to rewrite it as a penalty. In this framework, besides the quantum stage, a classical pre-processing involves generating an initial solution with a classical heuristic to the CQM that can be used as a starting point for the quantum annealing algorithm, potentially reducing the time required to find a good solution. The starting point will depend on the classic algorithm used in the heuristic. In our experiment, we used the default D-Wave configuration, which corresponds to Simulated Annealing.

Figure 2 illustrates how HSS CQM works [1]. The solver (blue) invokes some heuristics (threads) that run on classic CPUs and GPUs (green) and searches for good-quality starting point solutions. Each heuristic solver contains a quantum module (QM) that formulates and sends quantum queries to a D-Wave QPU (orange). The responses to these queries can guide heuristic search or improve the quality of a current set of solutions. In the end, each heuristic sends its best solutions to the solver.

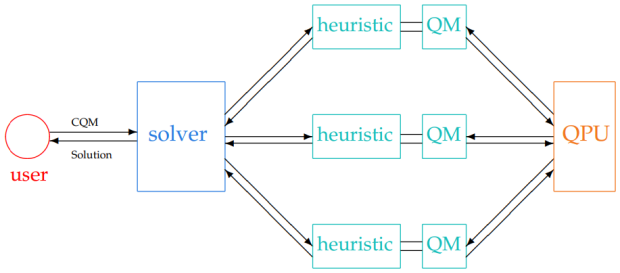


Figure 2: How CQM works at D-Wave.

5.2 The problem instances

Were created two groups of instances:

(1) **Synthetic Instances** - cell arrangements generated synthetically to simulate collision, confusion and mod q interference.
(2) **Real Instances** - cell arrangements based on real world datasets.

Each algorithm was expected to be tested through 64 instances (Qtty Exp), with varying cell neighborhood arrangements (Ins) and the quantity of PCI available for allocation (Qtty PCI), according to Table 1. Synthetic instances were arranged in four ways: a 5-cell instance, named X5, Figure 3, with matrix A configured in chess pattern, i.e., cells 1, 3, and 5 neighbors of cells 2 and 4; and 5, 6, and 9-cell instances, with all cells being neighbors between to each other, respectively named V5, Figure 4, V6, and V9. Real instances were created using public information of 5G network cells available from a telecommunications governmental agency in Brazil[6]. The circles in Figure 5 define the groups with 15, 27, 48, and 66 cells named R15 (yellow circle with a 0.5 km radius), R27 (white circle with a 0.7 km radius), R48 (red circle with a 0.98 km radius), and R66 (blue circle with a 1.25 km radius), respectively. Cells that are less than 1 km apart are considered neighbors. In Figures 3 and 4, who represent neighborhood matrix A, value 0, zero, indicates that the cell from that line is not neighbor to the cell in that column, the same way that the value 1, one, means that they are neighbors.

0	1	0	1	0
1	0	1	0	1
0	1	0	1	0
1	0	1	0	1
0	1	0	1	0

Figure 3: Instance X5, five cells as a chess pattern.

Initially, it was decided to assign the same weight to all penalties, that is, the factors ω_1, ω_2, and ω_3 will have values equal to 1. This promotes a better understanding of each algorithm, allowing them to find solutions that satisfy all constraints in a balanced way. In

[3]https://www.dwavesys.com/solutions-and-products/systems/
[4]https://www.dwavesys.com/solutions-and-products/ocean/
[5]https://www.dwavesys.com/solutions-and-products/cloud-platform/
[6]https://sistemas.anatel.gov.br/se/public/view/b/licenciamento.php

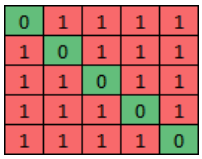


Figure 4: Instance V5, five cells all neighbors.

Table 1: Instances Distribution. Qtty PCI and Exp describe, respectively, the number of PCI and experiments per instance

Ins	Type	Qtty PCI	Qtty Exp
X5	Synth	1 to 5	5
V5	Synth	1 to 5	5
V6	Synth	1 to 6	6
V9	Synth	1 to 9	9
R15	Real	1 to 15	15
R27	Real	1,2,3,6,9,15,27	7
R48	Real	1,2,3,6,9,15,27,48	8
R66	Real	1,2,3,6,9,15,27,48,66	9
		TOTAL	64

Figure 5: Real instances, showing cell arrangements with 15 cells (yellow circle), 27 cell (white circle), 48 cell (red circle) and 66 cell (blue circle).

this scenario, of the three algorithms tested, the exact solver was the slowest to complete each experiment. Furthermore, the fact of having to traverse the entire search space made the execution of some scenarios unfeasible. Thus, the algorithm failed to generate results in 33 of the 64 predicted scenarios.

Table 2 shows the energy (i.e., the final Hamiltonian) of exact (EX), steepest descent (SD), and CQM methods to some of the synthetic instances simulated. Table 3 shows the energy of steepest descent (SD) and CQM methods for some real instances simulated. It can be seen that the exhaustive and CQM methods always find the lowest energy when compared to SD.

Figures 6, 7 and 8 show the execution time for V9 instances applying the exact algorithm, SD and CQM, respectively. It can be seen that SD has the shortest execution time, but, as highlighted in Table 2, this method does not always present the optimal solution (e.g, instances V5 with 4 and 5 PCI). This behavior is repeated in all other instances. These graphs also demonstrate the more rapid growth of the exhaustive search in comparison to other methods.

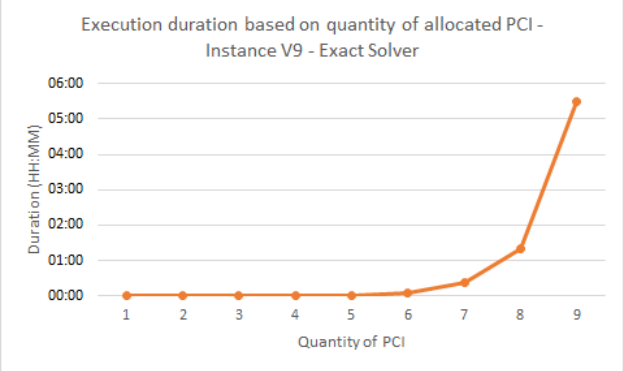


Figure 6: Execution duration based on the quantity of allocated PCI - Instance V9 - Exact Solver.

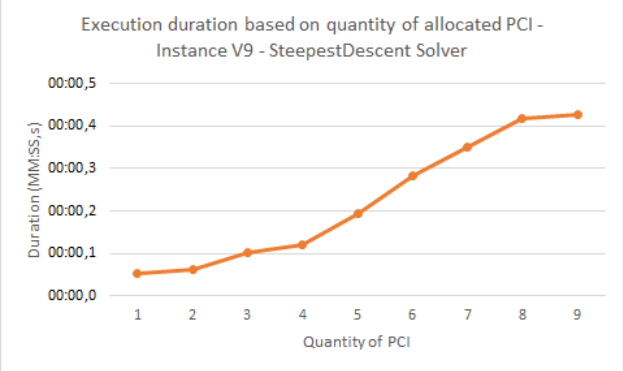


Figure 7: Execution duration based on the quantity of allocated PCI - Instance V9 - SteepestDescent Solver.

6 CONCLUSION

This paper explored the use of quantum computing to better solve the Physical Cell ID (PCI) planning problem. PCI planning is essential for minimizing the impacts of Inter-Cell Interference (ICI) in 5G Networks, such as collision, confusion, and *mod q* interference. Due to its complexity, this problem is usually solved with heuristics.

Our results show a clear advantage of quantum computing. CQM, the hybrid quantum-classical method, found solutions equivalent to the exact solver, but faster. In addition, besides the steepest descent has the shortest run, it does not produce optimal results.

As a future work, we plan to enhance our comparative analysis by exploring D-Wave Quantum Solvers, such as the Binary Quadratic Model (BQM)[7], other classic heuristics, such as Simulated

[7]https://docs.ocean.dwavesys.com/en/stable/concepts/bqm.html

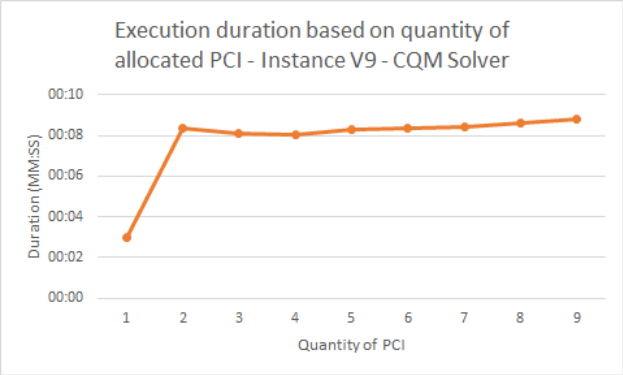


Figure 8: Execution duration based on the quantity of allocated PCI - Instance V9 - CQM Solver.

Table 2: Energy of Synthetic Instances

Ins	PCI	EX	SD	CQM
X5	1	32	32	32
X5	2	8	8	8
X5	3	4	4	4
X5	4	2	2	2
X5	5	0	0	0
V5	1	60	60	60
V5	2	24	24	24
V5	3	12	12	12
V5	4	8	**12**	8
V5	5	4	**8**	4
V6	1	90	90	90
V6	2	36	36	36
V6	3	18	18	18
V6	4	14	14	14
V6	5	10	10	10
V6	6	6	**12**	6
V9	1	216	216	216
V9	2	96	96	96
V9	3	54	54	54
V9	4	44	**48**	44
V9	5	36	36	36
V9	6	30	30	30
V9	7	26	**28**	26
V9	8	22	**30**	22
V9	9	18	**22**	18

Annealing, Tabu Search, and better exact algorithms for solving QUBO problems [3]. Additionally, we want to extend our current analysis beyond Quantum Annealing systems, testing, for example, gate-based systems such as the ones provided by IBM[8].

[8]https://www.ibm.com/quantum

Table 3: Energy of Real Instances

Ins	PCI	SD	CQM	Ins	PCI	SD	CQM
R15	2	266	242	R27	3	566	468
R15	3	148	134	R27	6	336	258
R15	4	124	98	R27	9	222	180
R15	5	104	66	R27	15	188	154
R15	6	84	62	R27	27	152	126
R15	7	70	58	R48	3	1392	1266
R15	8	56	54	R48	6	852	682
R15	9	54	50	R48	9	616	488
R15	10	50	46	R48	15	420	378
R15	11	62	44	R48	27	354	312
R15	12	62	42	R66	3	1896	1824
R15	13	44	40	R66	6	1220	920
R15	14	44	38	R66	9	910	650
R15	15	42	38	R66	15	614	468

ACKNOWLEDGMENTS

The authors would like to thank the Center of Advances Studies and Systems (CESAR) for investing in the Quantum Applications in Technology and Software (|QATS⟩), a research group that the authors are members, enabling access to the D-Wave computers as well as other relevant materials for developing this research paper.

REFERENCES

[1] 2022. Hybrid Solvers for Quadratic Optimization. https://api.semanticscholar.org/CorpusID:251834718
[2] Tameem Albash and Daniel A. Lidar. 2018. Adiabatic quantum computation. *Reviews of Modern Physics* 90, 1 (jan 2018), 015002. https://doi.org/10.1103/RevModPhys.90.015002 arXiv:arXiv:1611.04471v2
[3] Alain Billionnet and Sourour Elloumi. 2007. Using a Mixed Integer Quadratic Programming Solver for the Unconstrained Quadratic 0-1 Problem. *Mathematical Programming* 109, 1 (2007), 55–68. https://doi.org/10.1007/s10107-005-0637-9
[4] Andrea Boella, Michele Ludovico, Giuseppe Minerva, and Mauro Alberto Rossotto. 2021. *Application of Quantum Computing to Cell-ID Planning of 4.5G and 5G Networks (TIM).* Technical Report. 42–45 pages. https://www.gsma.com/newsroom/wp-content/uploads/IG-11-Quantum-Computing-Networking-and-Security.pdf
[5] M Born and V Fock. 1928. Beweis des Adiabatensatzes. *Zeitschrift für Physik* 51 (1928), 165–180. https://doi.org/10.1007/BF01343193
[6] Forsk. 2018. *Atoll - Wireless Network Engineering Software - version 3.4.* Retrieved November 27, 2023 from https://www.forsk.com/sites/default/files/atoll_34_sept2018-light_0.pdf
[7] Jihong Gui, Zhipeng Jiang, and Suixiang Gao. 2018. PCI Planning Based on Binary Quadratic Programming in LTE/LTE-A Networks. *IEEE Access* 7 (2018), 203–214. https://doi.org/10.1109/ACCESS.2018.2885313
[8] Mark W Johnson, Mohammad HS Amin, Suzanne Gildert, Trevor Lanting, Firas Hamze, Neil Dickson, Richard Harris, Andrew J Berkley, Jan Johansson, Paul Bunyk, et al. 2011. Quantum annealing with manufactured spins. *Nature* 473, 7346 (2011), 194–198.
[9] Anggita Putri Lestari, A Nachwan Mufti, and Uke Kurniawan Usman. 2018. Optimization of PCI conflict detection in LTE-advanced using collision and confusion methods considering reuse distance algorithm. In *2018 International Conference on Signals and Systems (ICSigSys)*. IEEE, 20–24.
[10] Satoshi Morita and Hidetoshi Nishimori. 2008. Mathematical foundation of quantum annealing. *J. Math. Phys.* 49, 12 (dec 2008), 125210. https://doi.org/10.1063/1.2995837 arXiv:0806.1859
[11] Cheng-fu Zhang, Navrati Saxena, Fan-guang Kong, and Jian-hua He. 2013. Self-Configuration of Physical Cell Identity in LTE Based on Improved GSO Algorithm. In *Proceedings of 20th International Conference on Industrial Engineering and Engineering Management: Theory and Apply of Industrial Engineering*. Springer, 377–387.

Generation of Industrial Protocol-Traffic via Enhanced Wasserstein GAN

Mikel Moreno Moreno[1], Lander Segurola Gil[1], Francesco Zola[1], Arantza del Pozo[1], and Iker Pastor López[2]

[1]Fundación Vicomtech, Research and Technology Alliance, San Sebastian, Spain, ✉ {mmoreno, lsegurola, fzola, adelpozo}@vicomtech.es
[2]Universidad de Deusto, Bilbao, Spain, ✉ iker.pastor@deusto.es

Machine learning (ML) has become a useful tool in cybersecurity for anomaly and attack detection tasks [3]. A large amount of training data is required to get the best performance of those models. However, training models using real user data often leads to privacy exposure and ethics problems. The option of real data anonymization has been proven unsuccessful in providing satisfactory privacy protection without causing data quality degradation. The option left is the generation of new realistic synthetic data that can substitute real data in ML training processes. Compared to generating images, learning distributions on multi-variate time-series data poses a different set of challenges. Multi-variate data is more diverse in the real world, and such data usually has more complex dependencies (temporal and spatial) as well as heterogeneous attribute types (continuous and discrete) [11]. Synthetic data generation models often treat each column as a random variable to model joint multivariate probability distributions. The modelled distribution is then used for sampling. Traditional modelling algorithms have the limitation of distribution data types, and due to computational issues, the dependability of synthetic data generated by these models is extremely limited [9]. Recently, Generative Adversarial Network (GAN)-based approaches have augmented the ability to generate data [5, 10, 7, 6]. However, they are either restricted to a static dependency without considering the temporal dependence usually prevalent in real-world data [10, 8], or only partially build temporal dependence inside GAN blocks [4].

The work aims to investigate if GAN architecture can learn the diverse types of industrial traffic instigated by users and machines on a network with multiple industrial protocols. For that, the TON_IoT open-source dataset [1] is used containing Modbus protocol traffic. The following research questions are tackled: 1) Are GANs able to overcome their usual limitations and be able to generalize and generate realistic industrial network traffic? 2) Can synthetic data generated by these models be injected and fool a genuine industrial service? A methodology comprised of two steps is proposed as an answer. The first is a generator module that can create synthetic network packets. The second is a simulator module that injects the synthetic data into the network to simulate a genuine Modbus client process.

On the one hand, a generator module is proposed composed of a GAN and a data loader in charge of feeding the model with network traffic in pcap format. Training is performed for each simulated network so that each model will be able to replicate the characteristics of the trained data traffic to be simulated by the injection module. It has already been proved that the output of GAN models suffers from mode-collapse, a decrease in the variability by continuously generating instances of the same data). To avoid this, a Wasserstein GAN (also known as WGAN is implemented) [2]. When training the traffic generator, the pipeline is comprised of three steps: the data acquisition, the data transformation (via an autoencoder), and the model fitting. First, the network traffic is extracted from the desired environment via tshark[1] commands. After a raw pcap is obtained, the categorical features of the packets (for example, source, destination, or protocol fields) are first passed through a one-hot-encoding process. After that, the vectorized data is used for training an autoencoder and both, the encoder and decoder parts are extracted. Then embedded into vectors of continuous space via the autoencoder's encoder. Afterwards, the parsed data values are ingested in the WGAN architecture for training. Once the stop conditions are fulfilled, the model can represent new data as continuous space vectors, to get them back to the original state, the inverse path is followed: The decoder is used to reconstruct the embedding-space-belonging-generated data, and then the inverse operation of the one-hot encoder is applied. The generated data is

[1]https://www.wireshark.org/docs/man-pages/tshark.html (Accessed on: 10/08/2023)

then saved in a CSV format for the simulator to read them. To be able to adapt to different sources, the presented pipeline starts from a traffic capture performed by tshark and stored in a pcap file, after which a script is executed that generates a CSV file containing the characteristics of the protocol in the captured traffic.

On the other hand, a simulator module is developed to validate the synthetic data generated, this module is composed of a decoder and a packet injector tool called Scapy [2]. Scapy will automatically generate the remaining header values to send the synthetic data as payload through the network. To verify the simulation matches the given input, the packet sniffing tool *tshark* is used. In case any packet is malformed, tshark will send an alert. The number of generated alerts will be useful to calculate the malformed packet generation rate of the proposed model.

The expected output should be a complete communication between a Modbus fake client (it does not possess the service or information of the processes) and a genuine server, with no malformed packets and payloads that maintain the context of the network. So, an individual observing the network traffic could not distinguish between a genuine and a fake client. In the future, this research could lead to fully simulated networks (both clients and servers) based on trained data from external networks, thus helping other researchers improve their ML models by being able to generate high-quality synthetic training data.

References

[1] Abdullah Alsaedi, Nour Moustafa, Zahir Tari, Abdun Mahmood, and Adnan Anwar. Ton_iot telemetry dataset: A new generation dataset of iot and iiot for data-driven intrusion detection systems. *IEEE Access*, 8:165130–165150, 2020.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[3] Anna L. Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, 2016.

[4] Steve TK Jan, Qingying Hao, Tianrui Hu, Jiameng Pu, Sonal Oswal, Gang Wang, and Bimal Viswanath. Throwing darts in the dark? detecting bots with limited data using neural data augmentation. In *2020 IEEE symposium on security and privacy (SP)*, pages 1190–1206. IEEE, 2020.

[5] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using GANs for sharing networked time series data. In *Proceedings of the ACM Internet Measurement Conference*. ACM, oct 2020.

[6] Santosh Kumar Nukavarapu, Mohammed Ayyat, and Tamer Nadeem. Miragenet - towards a gan-based framework for synthetic network traffic generation. pages 3089–3095. IEEE, 12 2022.

[7] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, jun 2018.

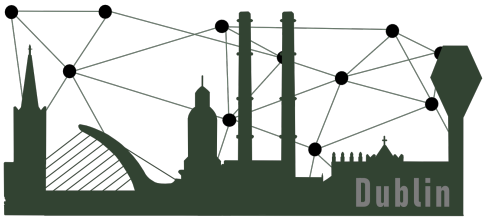
[8] Markus Ring, Daniel Schlör, Dieter Landes, and Andreas Hotho. Flow-based network traffic generation using generative adversarial networks. *Computers & Security*, 82:156–172, may 2019.

[9] Yi Sun, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Learning vine copula models for synthetic data generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5049–5057, Jul. 2019.

[10] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in Neural Information Processing Systems*, 32, 2019.

[11] Shengzhe Xu, Manish Marwah, Martin Arlitt, and Naren Ramakrishnan. Stan: Synthetic network traffic generation with generative neural models. *Communications in Computer and Information Science*, 1482 CCIS:3–29, 9 2020.

[2] https://scapy.readthedocs.io/ (Accessed on: 10/08/2023)



INOC - 2024

Session Session 4C: Routing
Tuesday 12 March 2024, 15:30-17:00
Q013

A nested Benders-Lagrange Approach to Delay Constrained Routing

Antonio Frangioni[1], Laura Galli[1], and Enrico Sorbera[2]

[1]Dipartimento di Informatica, Università di Pisa, Pisa, Italy, ✉ antonio.frangioni@unipi.it, laura.galli@unipi.it
[2]Dipartimento di Matematica, Università di Pisa, Pisa, Italy, ✉ enrico.sorbera@hotmail.it

Many of today's real-life applications in computer networks require stringent Quality of Service (QoS) scheduling guarantees in terms of controlled "end-to-end" delay, a.k.a., *worst-case delay* (WCD); that is, the delay of any IP packet in a flow must be no larger than a given threshold. According to the traffic engineering literature, the WCD of a flow of IP packets can be defined as a function of (*i*) the (simple) path on the network on which the flow is routed, and (*ii*) of the bandwidth rate allocated to the flow along the path. This leads to the definition of the *Delay Constrained Routing Problem* (DCR), which requires computing the paths and reserving the resources along them on the IP network, so that the WCD constraints are respected for all the flows, while some objective function, usually the total resources utilization, is optimised.

In [1] it is shown that DCR can be formulated as a Mixed-Integer Second-Order Cone Program (MISOCP). Thus, in the single-flow single-path case, it can be solved efficiently enough with off-the-shelf, general-purpose solvers for small- to mid-size networks, but may struggle on larger ones. In this study we present a bespoke, "solver-free" method, based on a nested Benders-Lagrange approach, that provides both upper and lower bounds of very good quality in extremely short computing times. We start by describing a model for the DCR problem, which is similar to the one proposed in [1], but with some specific changes useful for our algorithmic purposes.

$$\min \textstyle\sum_{(i,j)\in A} f_{ij} r_{ij} \tag{1}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} - \sum_{(i,j)\in\delta^+(i)} x_{ij} = \begin{cases} -1 & \text{if } i = s \\ 1 & \text{if } i = d \\ 0 & \text{otherwise} \end{cases} \qquad i \in N \tag{2}$$

$$\frac{\sigma}{z} + \sum_{(i,j)\in A} \left(\frac{L {x_{ij}}^2}{r_{ij}} + \bar{l}_{ij} x_{ij} \right) \leq \delta \tag{3}$$

$$z\, x_{ij} \leq r_{ij} \leq c_{ij} x_{ij} \qquad (i,j) \in A \tag{4}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \tag{5}$$

$$z \in [\rho, c_{max}] \tag{6}$$

The arc-flow binary variables x_{ij} indicate whether arc (i,j) belongs to the chosen path, and (2) are the standard flow conservation constraints. Reserved rate variables r_{ij} represent the amount of bandwidth capacity reserved on arc (i,j), and z captures the minimum reserved rate along the path. Constraint (3) is the WCD for the common Strictly Rate-Proportional packet schedulers, while constraints (4) ensure that $r_{ij} = 0$ if $x_{ij} = 0$, and $\rho \leq z \leq r_{ij}$ if $x_{ij} = 1$.

The key observation in our approach is that one can view z as a "complicating" variable, in the sense that, when $\bar{z}$ is fixed, the corresponding model (1)–(5), to which we refer as $DCR(\bar{z})$, is a much easier optimization problem in the variables x and r. This suggests to use the *Generalized Benders Decomposition* (GBD) framework [3], but, as we shall see, we will employ it in a nonstandard way. The first step consists in projecting (1)–(6) onto z, i.e., solving

$$\min\{\, v(z) \,:\, z \in [\rho, c_{max}] \,\}$$

where $v(z)$ represents the *value function*, i.e., the optimal value of the Benders' *subproblem* $DCR(z)$. The second step uses the dual representation of $v(z)$

$$v(z) = \max \left\{ \, \min \left\{ \, f^T r + u^T G(x,r,z) \,:\, (x,r) \in X \, \right\} \right\} \tag{7}$$

where $G(x, r, z)$ denotes the set of constraints (linking variables x and r with the complicating variable z) that are dualized with multipliers u, and X represents the remaining set of constraints on x and r. Denoting by

$$\psi(z, u) = \min \left\{ f^T r + u^T G(x, r, z) \ : \ (x, r) \in X \right\}$$

this manipulation yields the equivalent Benders' *master problem* for DCR

$$\min \left\{ r_0 \ : \ r_0 \geq \psi(z, u^j) \quad j = 1 \dots p \ , \ z \in [\rho, c_{max}] \right\}$$

In order to apply the GBD method, the functions $\psi(z, u^j)$ should be convex in z and satisfy the *P-property* [3]; that is, for every u^j, $j = 1 \dots p$, the minimization problem involved in $\psi(z, u^j)$ should be solved independently of z. Unfortunately, the DCR model does not satisfy any of these conditions.
Therefore, we set out to construct proper approximations of the functions $\psi(z, u^j)$. The first step is to dualize constraints (2) and (3) with multipliers π_i, $i \in N$ and $\lambda \geq 0$, respectively. Denoting by $u = (\pi, \lambda)$ the vector of dual multipliers, and defining

$$q(u, x, r) = \sum_{(i,j) \in A} \left(f_{ij} r_{ij} + \lambda L \frac{x_{ij}^2}{r_{ij}} + \lambda \bar{l}_{ij} x_{ij} - \pi_i x_{ij} + \pi_j x_{ij} \right)$$

we obtain

$$\psi(z, u) = -\pi^T b + \frac{\lambda \sigma}{z} - \lambda \delta + \min_{(x,r)} \left\{ q(u, x, r) \ : \ (4) \ , \ (5) \right\} \tag{8}$$

Despite still lacking convexity and the *P-property*, the corresponding master problem presents three advantages for the construction of approximations. First, for a given value of $\bar{z}$, the Lagrangian dual can be efficiently solved (see [2]), therefore we can easily compute the corresponding optimal dual vector $\bar{u}$. Second, the minimization problem in (8) is separable on the arcs $(i, j) \in A$. Third, by analyzing the behaviour of $\psi(z, u)$ as a function of z, one can prove the following (we omit details for brevity):

- $\psi(z, u)$ is non-convex and non-differentiable for $z \in V_1 = [\rho, \bar{z}]$
- $\psi(z, u)$ is convex for $z \in V_2 = [\bar{z}, c_{max}]$

In order to apply the GBD framework, we construct convex approximations of $\psi(z, u^j)$ in the interval V_1, to obtain a global two-piece convex approximation for the Benders' master problem. This allows us to perform iteratively a double line search in the sub-intervals V_1 and V_2. Computational results are very encouraging and show that our method consistently provides upper and lower bounds of very good quality in extremely short computing times on realistic instances even of large size.

References

[1] A. Frangioni, L. Galli, M.G. Scutellà "Delay-Constrained Shortest Paths: Approximation Algorithms and Second-Order Cone Models" Journal of Optimization Theory and Applications, 164, 1051–1077 (2015)

[2] A. Frangioni, L. Galli, E. Sorbera "Lagrangian approaches for QoS scheduling in computer networks" in M. Bruglieri, P. Festa, G. Macrina, O. Pisacane (eds.) Optimization in Green Sustainability and Ecological Transition - ODS, Ischia, Italy, September 4–7 2023, AIRO Springer Series, to appear (2024)

[3] A.M. Geoffrion "Generalized Benders Decomposition. Journal of Optimization Theory and Applications" 10, 237–260 (1972)

Addressing demand uncertainty in the pickup and delivery problem with time windows via robust optimisation

Alex Abreu[1], Maria Battarra[2], and Pedro Munari[1]

[1]Federal University of São Carlos, Brazil, ✉ {abreualexp@gmail.com, munari@dep.ufscar.br}
[2]University of Bath, United Kingdom, ✉ m.battarra@bath.ac.uk

We consider the one-to-one pickup and delivery problem with time windows (PDPTW), a routing problem in which a set of capacitated vehicles is available to serve customer requests consisting of paired pickup and delivery locations [1]. Each used vehicle must start its route from the central depot, collect items from pickup locations, transport them to the corresponding delivery locations, and return to the depot. All visits must respect customer time windows and vehicle capacity constraints.

This variant has several applications in meal delivery services, oil transportation between terminals and offshore platforms, and disaster relief in humanitarian logistics [2]. Given that real-world data is often estimated or uncertain, incorporating uncertainties into the decision-making process is fundamental. Resilient routes, which have a higher chance of being feasible despite the data variability, can be obtained using fuzzy programming [3], stochastic programming [4], and robust optimisation [5].

We develop robust optimisation techniques to incorporate demand uncertainty into the PDPTW formulations. This ensures that optimal routes remain feasible for any demand realisation within an uncertainty set that suits the decision maker's risk aversion, without the necessity of choosing a probability distribution. We propose robust counterparts in which the demands of a given number of requests can reach their worst-case value based on a parameter known as the budget of uncertainty [6], which is set by the decision maker. Prior papers introducing uncertainty in the pickup and delivery problem mostly considered scenario-based uncertainty sets that typically lead to either overconservative or optimistic solutions [7]. Robust optimisation has so far being applied to the one-to-one variant by [8, 9].

We show how to obtain the robust counterparts in two different ways: (i) through the dualisation scheme, a commonly employed method in existing literature; and (ii) through the linearisation of dynamic programming equations, which was recently introduced and has shown superior performance compared to dualisation in various VRP variants. As we present, the application of these techniques in the PDPTW is not straightforward, as the pairing of customers impose additional challenges. We assess and compare the effectiveness of these formulations through computational experiments using benchmark instances and different levels of budget of uncertainty and demand deviations. The results indicate that the robust routes are indeed more resilient and can benefit real-world routing applications.

References

[1] M. Battarra, J.-F. Cordeau, and M. Iori, "Chapter 6: Pickup-and-Delivery Problems for Goods Transportation," in *MOS-SIAM Series on Optimization* (P. Toth and D. Vigo, eds.), pp. 161–191, Society for Industrial and Applied Mathematics, Nov. 2014.

[2] M. G. S. Furtado, P. Munari, and R. Morabito, "Pickup and delivery problem with time windows: A new compact two-index formulation," *Operations Research Letters*, vol. 45, pp. 334–341, July 2017.

[3] V. P. Singh, K. Sharma, and D. Chakraborty, "Solving capacitated vehicle routing problem with demands as fuzzy random variable," *Soft Computing*, vol. 27, pp. 16019–16039, Nov. 2023.

[4] J. De La Vega, M. Gendreau, R. Morabito, P. Munari, and F. Ordóñez, "An integer L-shaped algorithm for the vehicle routing problem with time windows and stochastic demands," *European Journal of Operational Research*, vol. 308, pp. 676–695, July 2023.

[5] P. Munari, A. Moreno, J. De La Vega, D. Alem, J. Gondzio, and R. Morabito, "The Robust Vehicle Routing Problem with Time Windows: Compact Formulation and Branch-Price-and-Cut Method," *Transportation Science*, vol. 53, pp. 1043–1066, July 2019.

[6] D. Bertsimas and M. Sim, "The Price of Robustness," *Operations Research*, vol. 52, pp. 35–53, 2004.

[7] S. Allahyari, S. Yaghoubi, and T. Van Woensel, "The secure time-dependent vehicle routing problem with uncertain demands," *Computers & Operations Research*, vol. 131, p. 105253, July 2021.

[8] Z. Al Chami, B. Bechara, H. Manier, M.-A. Manier, and M. Sleiman, "A GRASP-ALNS combination for robust pickup and delivery problem," *International Journal of Production Research*, vol. 60, pp. 3809–3828, June 2022.

[9] X. Liu, D. Wang, Y. Yin, and T. C. Cheng, "Robust optimization for the electric vehicle pickup and delivery problem with time windows and uncertain demands," *Computers & Operations Research*, vol. 151, p. 106119, Mar. 2023.

Addressing nurse preference in nurse assignment and routing problem in dynamic environment

Md Samiullah Ansari and Avijit Khanra

Department of Management Sciences, Indian Institute of Technology Kanpur, Kanpur, U.P., India, 208016
✉ msansari@iitk.ac.in, kavijit@iitk.ac.in

Abstract

With technological advancements, it is now possible to deliver non-clinical and certain clinical services at patients' homes. The provision of home care services can reduce care costs, improve patients conveniences, and assist conventional hospitals in alleviating overcrowding in general wards, emergency departments, etc. The efficient utilization of limited resources meeting all the requirements of service provider, patients, and nurses such as continuity of care, balanced workload among nurses, minimum operating costs, etc. is essential. In this study, we develop a multi-period integrated nurse assignment and routing problem as a mixed integer linear programming problem that considers patient arrivals over time and flexibility in nurses' working windows. First, we employ a CPLEX solver to solve the instances and observe that it only solves a few day-wise sub-instances. To overcome the limitations of the solver, we propose a subtour elimination based valid inequalities approach. Computational experiments on randomly generated instances demonstrate that the proposed approach not only improves computational efficiency but also solves more sub-instances compared to CPLEX.

1 Problem description and Methodology

The home healthcare system consists of care requests, medical staff, and care provider. A detailed literature review on home healthcare can be found in [1] and [2]. Most home healthcare studies on nurse assignment and routing problem address the static and single objective problem where all parameters are known with certainty and the objective is to minimize total travel cost or travel time. However, in order to develop a practical HHC model, the considerations that address real-life aspects such as dynamic patient arrivals, nurses' preferences, changes in appointment times, visit cancellations, etc. are also important. We explore the multi-period integrated nurse assignment and routing problem where patients are arriving over time. We also consider nurses' preferences for flexible work windows, where each nurse specifies their preferred working window. Each new patient has medical or social service needs. Thus, nurses have two skill levels and are hierarchical. It implies that nurses with higher skills can serve patients with lower skill requirements, but otherway is not true. The manager knows all patient related information, including service duration, frequency, episode of care, etc. The objective of the problem is to obtain best service delivery plan while minimizing the travel time of all nurses over the planning horizon. The mathematical model of the studied problem is developed as a mixed integer linear programming problem.

This problem is a *NP-hard* problem because it combines staff rostering and vehicle routing problem and both are well-known *NP-hard*. Therefore, using an optimization solver to solve practical size instances is not sufficient and it will give either poor quality solutions or no solutions within the given time limit. Initially, the studied model is coded in Python language (3.8) and solved by employing a standard IBM ILOG CPLEX 12.20 solver. Limitations on CPLEX are identified and then we propose a subtour elimination based valid inequalities approach with the hope that it will improve the bound of the LP relaxation. Hence, it will overcome the limitations of CPLEX by improving computational efficiency. Before starting the optimization of the original MILP model we have done preprocessing based on the structural properties of the problem.

2 Results and Discussion

The applicability and validity of the model and the suggested solution algorithm have been investigated using computational experiments on randomly generated instances. For this study, the test instances have been generated with three social workers, two medical nurses, and a maximum of five new patients arriving per day. We have experimented on two instances ($2 \times 8 = 16$ day-wise sub-instances). We set a computational time limit of 1800 seconds as the stopping criteria for solving each sub-instance with CPLEX. All computational experiments are conducted on a desktop equipped with a 12th Gen Intel(R) Core(TM) i7-12700 2.10 GHz and 64.0 GB RAM. Table 1 shows the results for instance-1, while Table 2 shows the results for instance-2. The results for each instance capture details such as accepted patients, patient set, objective value, computational time, and objective value for both CPLEX and CPLEX with valid inequalities cases. Additionally, a few more information such as the number of cuts added, and LP relaxation (LPR) bound improvement in the case of CPLEX with valid inequalities are recorded. Finally, the improvement in the computational time is presented in the last columns of the Tables.

The results for instance 1 show that CPLEX solves sub-instances till $day - 5$ and it does not produce any feasible solutions within the specified time limit for sub-instance corresponding to $day - 6$. On the other hand, CPLEX with valid inequalities solves sub-instances till day $day - 6$ (one extra sub-instance compared to CPLEX). The proposed algorithm improves the computational time for all sub-instances of instance 1. We can also observe that the proposed algorithm also improves the linear programming relaxation bound up to 101%. The computational results for instance 2 are shown in Table-2. CPLEX is able to solve four sub-instances till $day - 4$, however, it could not solve the $day - 5$ sub-instance for the given time limit. Nevertheless, the proposed algorithm solves all sub-instances within the specified time limit. It also improves computational efficiency except for the sub-instance corresponding to $day-4$ which takes about 3 seconds more time than CPLEX. The linear programming relaxation bound improvement is achieved up to 85%. The details can be found in Table-2. Considering the results, we can conclude that to get consistency in solving all sub-instances of any reasonable size instance of the studied problem, there is a need to develop better solution techniques.

Table 1: Results for Instance-1

Day	Accp_pat	Pat_set	Obj_val	CPLEX		CPLEX+valid inequalities				CTI
				Comp_time	Opt_gap	Comp_time	Opt_gap	nb_cuts	LPR_b	
1	5	5	10.71	0.203	0	0.078	0	14	60.129	61.576
2	5	10	18.11	0.36	0	0.266	0	23	55.725	26.111
3	5	15	28.08	0.875	0	0.75	0	34	63.271	14.286
4	5	20	38.39	10.328	0	8.422	0	45	73.944	18.455
5	5	25	44.47	47.719	0	42	0	61	47.109	11.985
6	0	25	39.53	–	–	48.343	0	55	101.914	–

Table 2: Results for Instance-2

Day	Accp_pat	Pat_set	Obj_val	CPLEX		CPLEX+valid inequalities				CTI
				Comp_time	Opt_gap	Comp_time	Opt_gap	nb_cuts	LPR_b	
1	5	5	17.62	0.094	0	0.109	0	11	15.719	0
2	4	9	19.22	0.25	0	0.25	0	23	27.075	6
3	5	14	29.35	4.828	0	3.047	0	29	24.818	36.889
4	5	19	34.64	28.156	0	32.562	0	45	37.112	-8.54
5	2	21	35.24	–	–	1800	2.2	48	67.412	–
6	2	23	30.11	–	–	1800	4.2	46	85.734	–
7	1	24	25.97	–	–	1800	1.8	43	80.989	–
8	5	26	25.35	–	–	1800	2.1	47	74.189	–

References

[1] Christian Fikar and Patrick Hirsch. Home health care routing and scheduling: A review. *Computers & Operations Research*, 77:86–95, 2017.

[2] Luca Grieco, Martin Utley, and Sonya Crowe. Operational research applied to decisions in home health care: A systematic literature review. *Journal of the Operational Research Society*, 72(9):1960–1991, 2021.

INOC - 2024

Session Session 5A: Network Interdiction

Wednesday 13 March 2024, 11:00-12:30

Q011

Adaptive Partition-based Methods in an Asymmetric Shortest-path Network Interdiction Problem

Di H. Nguyen[1] and Yongjia Song[2]

[1]School of Mechanical & Materials Engineering, University College Dublin, Dublin, Ireland, ✉ di.nguyen@ucd.ie
[2]Department of Industrial Engineering, Clemson University, Clemson, USA, ✉ yongjis@clemson.edu

In this study, we investigate adaptive partition-based methods (APMs) in the context of a shortest-path network interdiction problem involving a leader (defender) and a follower (attacker). The problem was first introduced by Nguyen and Smith [2], in which the leader aims to maximize the expected shortest-path cost by raising the cost of arcs in the network, subject to a budget constraint. After observing the leader's decision, the follower selects a shortest source-sink path. The asymmetry in the game arises from to the leader having to make a decision (i.e., raising arc costs) in advance and thus, only knowing the distribution of arc costs (e.g., uniform distribution). In contrast, the follower observes the exact arc costs when selecting a source-sink path.

Nguyen and Smith proposes an algorithmic solution approach that involves adaptive partitioning, which iteratively divides the uncertainty region into smaller non-overlapping sub-regions and examines the bounds on the attacker's shortest-path cost. Optimality is concluded once this cost is within a predetermined tolerance. The studied approach shows to be computationally expensive and highly dependent on the chosen value of the optimality tolerance.

In our study, we aim to examine APMs' efficacy in solving the introduced shortest-path network interdiction problem in both the support space and the scenario space. Alternative approaches leveraging APMs which can permit other cost distributions besides uniform distribution are also investigated. (APMs have previously been explored in the context of two-stage stochasic linear programs which involve finite and non-finite distributions [1, 3, 4].) Computational experiments are conducted to provide insights into the methods' effectiveness.

Problem Formulation. In the problem introduced by Nguyen and Smith [2], the game takes place on a directed network $G = (N, A)$, where N is the set of nodes and A is the set of arcs. Let the cost of each arc $(i, j) \in$ be denoted by random variable C_{ij}. Arc cost C_{ij} is uniformly distributed in $[c^L_{ij}, c^U_{ij}]$, where $0 \leq c^L_{ij} \leq c^U_{ij} < \infty$. Let binary variables x denote the leader's decision. Define d_{ij} as the additional cost to use arc $(i, j) \in A$ if the leader decides to raise arc (i, j)'s cost (i.e., $x_{ij} = 1$). The leader can raise a maximum number of b arcs. The follower's set of feasible solutions Y that corresponds to the set of source-sink paths is given by the set of solutions y satisfying the following constraints:

$$\sum_{j \in N:(s,j) \in A} y_{sj} = 1 \tag{1a}$$

$$\sum_{i \in N:(i,t) \in A} y_{it} = 1 \tag{1b}$$

$$\sum_{j \in N:(k,j) \in A} y_{kj} - \sum_{i \in N:(i,k) \in A} y_{ik} = 0 \qquad \forall k \in N \setminus \{s, t\} \tag{1c}$$

$$y_{ij} \geq 0 \qquad \forall (i, j) \in A. \tag{1d}$$

Let $\bar{c}_{ij}$ denote the expected cost of each arc $(i, j) \in A$. For a given defense decision $\hat{x}$, the shortest-path cost observed by the attacker is bounded above and below by (2) and (3), respectively.

$$g(\hat{x}, \bar{c}^k) = \min \left\{ \sum_{(i,j) \in A} (\bar{c}^k_{ij} + d_{ij}\hat{x}_{ij}) y_{ij} \mid y \in Y \right\} \tag{2}$$

$$h(\hat{x}, \bar{c}^k) = \min \sum_{(i,j)\in A} \Big(q_{ij} + (c_{ij}^{L,k} + d_{ij}\hat{x}_{ij}) y_{ij} \Big) \tag{3a}$$

$$\text{s.t. } q_{ij} \geq (\bar{c}_{ij}^k - c_{ij}^{L,k}) - M_{ij}(1 - y_{ij}) \qquad \forall (i,j) \in A \tag{3b}$$

$$q_{ij} \geq 0 \qquad \forall (i,j) \in A \tag{3c}$$

$$y \in Y. \tag{3d}$$

Jensen's inequality implies that $g(\hat{x}, \bar{c}) \geq \mathbb{E}[g(\hat{x}, C)] \geq \mathbb{E}[h(\hat{x}, C) \geq h(\hat{x}, \bar{c})]$. Denoting the leader's set of feasible solution $X = \{x : \sum_{(i,j)\in A} x_{ij} \leq b, x \in \{0,1\}^{|A|}\}$, the leader's problem is given by:

$$\text{MP}(\mathcal{P}) : \max_{x\in X} \sum_{k\in K} p^k z^k \tag{4a}$$

$$\text{s.t. } z^k \leq \sum_{(i,j)\in Y^P} (\bar{c}_{ij}^k + d_{ij} x_{ij}) \qquad \forall P \in \mathcal{P}, k \in K, \tag{4b}$$

where $\mathcal{P}$ is the set of all source-sink paths in the network, Y^P is the set of arcs in path $P \in \mathcal{P}$, and K is a partition of the uncertainty cost region consisting of non-overlapping sub-regions k. Alternative to model (4), we propose the following model which permits the use of lazy constraints:

$$\text{MP-Lazy}(\mathcal{P}) : \max_{x\in X} z \tag{5a}$$

$$\text{s.t. } z \leq \sum_{k\in K} p^k \sum_{(i,j)\in \mathbf{P}^k} (\bar{c}_{ij}^k + d_{ij} x_{ij}), \qquad \forall \mathbf{P} \in \mathcal{P} \times \mathcal{P} \times \cdots \mathcal{P} \ (|K| \text{ times}). \tag{5b}$$

Considerations in the Development of APMs. The following considerations are evaluated to determine an effective partition approach in obtaining an optimal defense decision x. Note that the algorithmic solution approach involves refining the uncertainty region to obtain tighter bounds on the shortest-path cost. The refinement decisions involve **arc selection** (e.g., selecting an arc based on the magnitude of its uncertainty range versus its likelihood to influence the follower's shortest path) and **dividing an arc's cost range** (e.g., at the mean base cost or at a base cost determined via sensitivity analysis). In addition, we also investigate the impact of an aggressive versus a delayed **partitioning approach**, and if a combination of these approaches can accelerate the process of obtaining an optimal solution. (A heavily refined uncertainty region may require less exploration of the solution space but can increase the computational effort required to solve the leader's problem due to a larger number of variables and constraints.) Computational experiments on the **model implementation**, i.e., a standard optimization model (4) versus a model utilizing lazy-constraints (5), are also conducted. An integrated branch-and-cut framework is employed with APMs. Finally, support-space APMs are evaluated against scenario-space APMs (e.g., sample average approximation). All computational experiments are conducted on randomly generated networks.

References

[1] M. Forcier and V. Leclère. Generalized adaptive partition-based method for two-stage stochastic linear programs: Geometric oracle and analysis. *Operations Research Letters*, 50(5):452–457, 2022.

[2] D. H. Nguyen and J. C. Smith. Network interdiction with asymmetric cost uncertainty. *European Journal of Operational Research*, 297(1):239–251, 2022.

[3] C. Ramirez-Pico and E. Moreno. Generalized adaptive partition-based method for two-stage stochastic linear programs with fixed recourse. *Mathematical Programming*, 196(1-2):755–774, 2022.

[4] Y. Song and J. Luedtke. An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse. *SIAM Journal on Optimization*, 25(3):1344–1367, 2015.

Assessing the Robustness of Projects via Longest-Path Network Interdiction with Failure Groups

Fei Wu[1], Jannik Matuschke[2], and Erik Demeulemeester[3]

[1,2,3]Operations Management Research Group, KU Leuven, Leuven, Belgium, ✉ fei.wu@kuleuven.be, jannik.matuschke@kuleuven.be, erik.demeulemeester@kuleuven.be

1 Introduction

Project scheduling is a part of project management that determines the sequence or scheduling plan for a series of related activities that are constituents of the project. Large-scale projects are often delayed by unforeseen disruptions that prolong some activities. These disruptions can arise from resource shortages, unexpected technical challenges, or external factors. To evaluate the robustness of a given project, one effective approach is to analyze the so-called *critical path*, i.e., the longest path in the *Project Evaluation and Review Technique* (PERT) network, a directed acyclic graph (DAG) consisting of nodes representing the activities and arcs representing the precedence constraints among them. In the absence of resource constraints, the critical path determines the makespan of the project [2]. Assessing how potential disruptions influence the longest-path distance in the PERT network thus helps to understand the robustness of the project and to identify potential bottlenecks or vulnerable activities. This leads to a longest-path interdiction problem in a DAG, where the goal of the interdictor is to increase the longest-path distance as much as possible.

This interdiction problem is well-understood for the case that the failure scenarios are described by a bound on the total number of disruptions or the cost of increasing individual task durations [1, 3, 4]. However, in practice, such disruptions usually do not happen independently from one another. For example, a technical issue with certain equipment usually affects all activities making use of this equipment. Similarly, a delay in the availability or a worse-than-expected performance of employees can cause correlated disruptions across different project activities. It is important to consider these interdependencies among various activities in such scenarios.

Therefore, we study a generalization of the aforementioned interdiction problem in which possible disruptions related to one another are grouped together in the same *failure group*. To avoid overly conservative estimates, the number of different failure groups that can be active simultaneously is bounded by a number given in the input. Within each such group, a polyhedral uncertainty set describes the possible delays of activities. We call this problem PSIP-FG (Project Scheduling Interdiction Problem with Failure Groups) for short. In order to assess the robustness of a project, we take the perspective of an interdictor who tries to prolong the completion/makespan of the project by as much as possible delaying some groups of activities. To the best of our knowledge, we are the first to model the coupling of uncertainties that appear among activities within each failure group as well as across multiple groups, and thus we consider the disruption caused by failure groups altogether. In the following sections, we will provide a formal description of our model and discuss preliminary results in terms of complexity and approximability of the problem.

2 Model description

A project can be represented by a directed acyclic graph $D = (V, E)$. Here, each node represents an activity, with two dummy activities s and t representing the start and the end of the project. Each arc represents a precedence constraint, i.e., for each $(i, j) \in E$, activity j can only start once activity i is completed. Given a vector of activity durations $d \in \mathbb{R}^V$, the makespan of the project is given by the

length of a longest s-t-path in D (where the length of a path P is given by the sum of its node weights $\sum_{v \in V(P)} d_v$).

We assume that the set of activities is classified into the union of m groups, that is $V = V_1 \cup V_2 \cup \ldots \cup V_m$. Every activity i has a nominal duration d_i. For every failure group $V_r, r \in [m]$, there is a polytope

$$Q_r = \{\boldsymbol{x} \in \mathbb{R}^V : \ A\boldsymbol{x} \leq \boldsymbol{b}, \boldsymbol{x} \geq 0 \text{ and } x_i = 0 \text{ for } i \notin V_r\}$$

describing possible disruptions. We assume that the delay of different failure groups is additive, so given $x_r \in Q_r, r \in [m]$, the prolonged activity durations are defined as $d_x = d + \sum_{r=1}^{m} x_r$. The task of the interdictor is to find a subset $S \subseteq [r]$ with $|S| \leq k$ and vectors $x_r \in \mathbb{R}^V$ for $r \in [m]$ with $x_r \in Q_r$ for $r \in S$ and $x_r = 0$ for $r \in [m] \setminus S$, so that the length of a longest s-t-path with respect to the activity durations d_x is maximized.

3 Preliminary results and future research

Our ongoing research has derived some hardness results for the PSIP-FG problem and yielded insights into the aspect of the approximation algorithm. An interesting special case arises when the number of delayed activities in each group V_r is bounded by a number ℓ_r, and the alternative processing time of any activity $i \in V_r$ is fixed. Using two distinct reductions from Set Cover and 3-SAT problem, we show the NP-hardness of PSIP-FG even in the following two special cases: (i) all the activities in the group V_r can be delayed, (ii) exactly one activity is allowed to be delayed in each group. We further investigate the approximability of general PSIP-FG and obtain an inapproximability bound

Theorem 1. *PSIP-FG does not admit* $(1 - 1/\mathrm{e} + \epsilon)$*-approximation for some* $\epsilon > 0$ *unless* $P = NP$.

Proof Sketch: Our proof relies on the reduction from Max k-cover: Given an instance of Max k-cover with a ground set and a collection of subsets of the elements, we construct an instance of PSIP-FP, where the set of activities corresponds to the ground set and failure groups are associated with subsets of the Max k-cover instance. The interdiction budget is set to k. Activities are arranged in series with default durations of 0. The uncertainty set for each group allows increasing the duration of each activity to 1 within the group. We show that the maximum union of k subsets is the ground set if and only if the distance of the interdicted longest s-t-path in the project network equals the cardinality of the ground set. Theorem 1 follows from the fact that it is NP-hard to approximate Max k-cover within $(1-1/\mathrm{e}+\epsilon)$. □

This inapproximability result even holds for (i) and even when D is a series-parallel graph. We also devised a $\frac{1}{k}$-approximation for the general version of the problem with arbitrary polyhedral uncertainty sets in each failure group.

Our next step will focus on identifying a harder inapproximabilty bound for general PSIP-FG from some special project network structure or on exploiting an approximation algorithm with a better guarantee. It is also of interest to identify some tractable or better approximable cases of PSIP-FG, for example, when the failure groups are restricted to node sets inducing a connected component in the graph. Additionally, we will also work on developing heuristic algorithms for solving realistic instances of the problem in practice.

References

[1] Gerald G Brown, W Matthew Carlyle, Robert C Harney, Eric M Skroch, and R Kevin Wood. Interdicting a nuclear-weapons project. *Operations Research*, 57(4):866–877, 2009.

[2] Eli Goldratt. The critic chain. great barrington, 1997.

[3] Eli Gutin, Daniel Kuhn, and Wolfram Wiesemann. Interdiction games on markovian pert networks. *Management Science*, 61(5):999–1017, 2015.

[4] R Kevin Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18, 1993.

An all-pairs shortest path coloring model to optimize network intrusion detection systems

Edoardo Scalzo[1], Floriano De Rango[2,3], Francesca Guerriero[1], Antonio Iera[2,3], and Mattia Giovanni Spina[2,3]

[1]Department of Mechanical, Energy and Management Engineering, University of Calabria, Rende (CS) 87036, Italy, ✉ {edoardo.scalzo, francesca.guerriero}@unical.it
[2]Department of Computer Engineering, Modeling, Electronics and Systems, University of Calabria, Rende (CS) 87036, Italy, ✉ {f.derango, antonio.iera, mattiagiovanni.spina}@dimes.unical.it
[3]CNIT - National Inter-University Consortium for Telecommunications, Parma, (PR) 43124, Italy

The colorability problem represents a stimulating challenge from both a theoretical and practical standpoint. Colorability is a concept that can be extended and integrated into a variety of combinatorial problems, leading to the creation of new models and variants that combine the main characteristics of classical problem with the unique challenges of coloring functions ([2], [4]). These new models can mathematically represent several problems that arise in real-world applications ([7], [6], [8]).

This work introduces a new variant of the all-pairs shortest path problem, which integrates a specific coloring constraint into the traditional framework. In this variant, the challenge is not only in solving the all-pairs shortest path, but also in strategically coloring a subset of vertices within a given edge-weighted, undirected, connected graph. With a set of colors at our disposal, the objective is to color certain vertices in such a way as to minimize the total cost of the shortest paths between all pairs of nodes, while satisfying a colorability constraint. This constraint requires that every shortest path must pass through at least one colored vertex for each color. Additionally, coloring a vertex incurs a cost, which is factored into the overall objective function to be minimized. This formulation adds a significant layer of complexity to the well-studied all-pairs shortest path problem.

We focus on the mathematical modeling of this problem, defining it as a non-linear integer programming model. In the proposed formulation, binary variables are used to represent both the traversing of edges and the coloring of vertices. The non-linearity of the model arises from the constraint ensuring that each shortest path contains at least one node of each color. We explore linearization techniques to effectively manage the combinatorial explosion, caused by the integration of coloring constraints into the shortest path problem. Additionally, the nature of the problem implies the potential for cycles induced by the coloring constraint. Therefore, our model requires the inclusion of subtour elimination constraints. We address this issue by considering a separation procedure, thereby dynamically processing these constraints to reduce the solution time ([3]).

To solve the problem on large-size instances, the present study introduces the use of the Biased Random-Key Genetic Algorithm (BRKGA) as an alternative solution approach to the exact model. The BRKGA is known for its effectiveness in solving complex problems, thanks to its ability to efficiently and quickly explore the solution space. In this context, the BRKGA is applied to generate solutions in scenarios where the exact approach proves to be too costly in terms of computational time ([1], [5]).

The proposed variant of the all-pairs shortest path problem can offer interesting applications in the context of next-generation networks (e.g., 5G and 6G networks) to optimally deploy, over programmable data plane, appliances Virtual Network Functions (VNFs). In our study, this variant is used to model a machine learning assisted network anomaly-based intrusion detection system ([9]). Specifically, the proposed optimization model is used to find the optimal deployment of such VNFs in terms of network security coverage, guaranteeing pervasive and ubiquitous network protection, and making the network itself the first line of defense against cyber-attacks. In our model, the graph nodes represent network nodes, whereas the edges between these nodes denote the physical and virtual connections among the network elements. The coloring of the nodes is employed to indicate the implementation of specific VNFs. Each distinct color represents a different type of VNF, reflecting the variety of functions necessary for a complete network security coverage. The cost associated with each color represents the implementation cost of a specific VNF. Meanwhile, the weight associated with each edge can represent various aspects,

essentially quantifying the characteristics or performance of the network connections, such as latency or bandwidth.

A comprehensive sensitivity analysis has been conducted, by varying the number of vertices, the number of available colors, and the density of the edges. This main aim was to evaluate the complexity of the exact model, solved using the general purpose software CPLEX, and to assess the performance and efficiency of the developed metaheuristic. Furthermore, to validate the proposed mathematical model and show the effectiveness of the BRKGA, a series of computational tests were carried out on various synthetic datasets. These datasets were specifically generated to reflect the particular conditions or limit cases identified in the sensitivity analysis, as well as choosing a median range of values to ensure an acceptable execution time.

References

[1] Carlos E. Andrade, Rodrigo F. Toso, José F. Gonçalves, and Mauricio G.C. Resende. The multi-parent biased random-key genetic algorithm with implicit path-relinking and its real-world applications. *European Journal of Operational Research*, 289(1):17–30, 2021.

[2] Francesco Carrabs, Carmine Cerrone, Raffaele Cerulli, and Selene Silvestri. On the complexity of rainbow spanning forest problem. *Optimization Letters*, 12:443–454, 2018.

[3] Raffaele Cerulli, Francesca Guerriero, Edoardo Scalzo, and Carmine Sorgente. Shortest paths with exclusive-disjunction arc pairs conflicts. *Computers & Operations Research*, 152:106158, 2023.

[4] Daniele Ferone, Paola Festa, and Francesca Guerriero. The rainbow steiner tree problem. *Computers & Operations Research*, 139:105621, 2022.

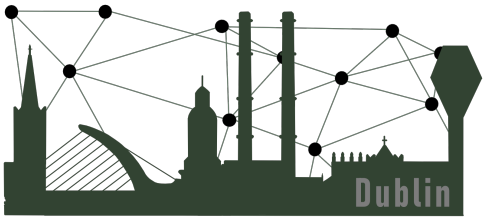
[5] Paola Festa, Francesca Guerriero, Mauricio GC Resende, and Edoardo Scalzo. A brkga with implicit path-relinking for the vehicle routing problem with occasional drivers and time windows. In *Metaheuristics International Conference*, pages 17–29. Springer, 2022.

[6] Philippe Galinier, Jean-Philippe Hamiez, Jin-Kao Hao, and Daniel Porumbel. *Recent Advances in Graph Vertex Coloring*, pages 505–528. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[7] Magnús M. Halldórsson and Guy Kortsarz. Tools for multicoloring with applications to planar graphs and partial k-trees. *Journal of Algorithms*, 42(2):334–366, 2002.

[8] Predrag Jovanović, Norbert Pavlović, Ivan Belošević, and Sanjin Milinković. Graph coloring-based approach for railway station design analysis and capacity determination. *European Journal of Operational Research*, 287(1):348–360, 2020.

[9] Daniele Moro, Giacomo Verticale, and Antonio Capone. A framework for network function decomposition and deployment. In *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020*, pages 1–6. IEEE, 2020.



INOC - 2024

Session Session 5B: Heuristics

Wednesday 13 March 2024, 11:00-12:30

Q012

Solving the Team Orienteering Arc Routing Problem: A Biased-Randomised Iterated Local Search Approach

Xabier A. Martin[1], Peter Keenan[2], Javier Panadero[3], Sean McGarraghy[2], and Angel A. Juan[1]

[1]Research Center on Production Management and Engineering, Universitat Politècnica de València, Alcoy, Spain, ✉ xamarsol@upv.es, ajuanp@upv.es
[2]Quinn School of Business, University College Dublin, Belfield, Ireland, ✉ peter.keenan@ucd.ie, sean.mcgarraghy@ucd.ie
[2]Department of Computer Architecture and Operating Systems, Universitat Autònoma de Barcelona, Bellaterra, Spain, ✉ javier.panadero@uab.cat

1 Introduction

Both the team orienteering problem and the arc routing problem have received an increasing amount of attention from the scientific community during the last decade. In this paper, we consider a hybridisation of both problems, the so-called team orienteering arc routing problem (TOARP). This problem has attracted considerable interest among researchers and practitioners due to its ability to effectively model routing scenarios involving unmanned aerial vehicles (UAVs) or other types of electric vehicles. This paper makes a significant contribution by addressing an extended and more realistic variant of the TOARP, which involves considering a destination depot that might be different from the origin one (Figure 1).

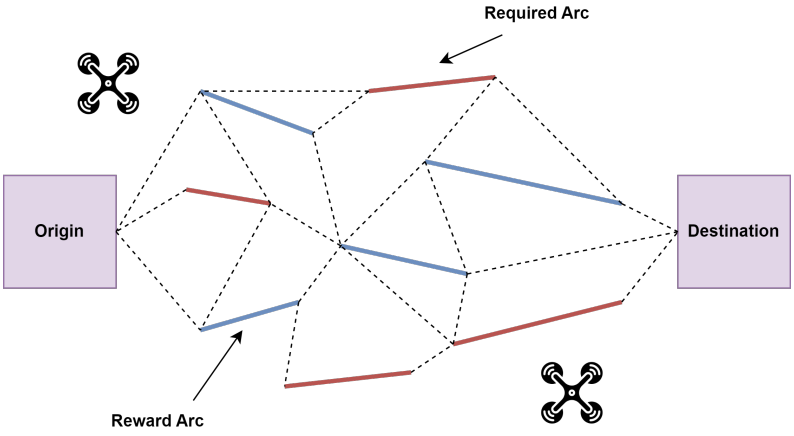


Figure 1: Illustrative example of a TOARP with two vehicles.

2 Problem Description

The problem formulation extends the one provided by Archetti et al. [2014] for the TOARP with a single depot. The objective is to maximise the total collected reward while ensuring that the constraints on the maximum number of vehicles available and the maximum time each vehicle can employ from its origin to its destination are not violated. Let $V = \{0, 1, 2, \ldots, n, n+1\}$ be the set of nodes connected by arcs, where node 0 is the origin depot and node $n+1$ is the destination one. Consider a set of arcs $A = \{(i, j)|i \in V, j \in V, i \neq j\}$ connecting some of the nodes in V (the network does not have to be complete). The arcs can be divided into two types: required arcs and reward arcs. The set of required arcs, $A_r \subseteq A$, need to be visited in any case, while reward arcs are optional and provide a reward the first time they are visited (they can be visited more than once, either by the same or by another vehicle,

but no additional reward can be collected from these additional visits). Let us denote by $r_{ij} \geq 0$ the reward associated with each arc $(i, j) \in A$. We have a fleet $K = \{1, 2, \ldots, |K|\}$ of vehicles available. Each vehicle will depart from the origin depot and needs to reach the destination depot within a maximum time $T_{max} > 0$.

3 Solving Methodology

The solving methodology introduces a biased-randomised iterated local search (BR-ILS) algorithm to solve the TOARP. As highlighted in Estrada-Moreno et al. [2019], these algorithms are efficient and relatively easy to implement, offering a good trade-off between simplicity and performance. Our algorithm consists of two stages, where we first generate a feasible initial solution using a novel heuristic inspired by the savings-based heuristic proposed by Panadero et al. [2020] for solving the classical TOP. During the second phase of our approach, we employ an ILS metaheuristic to enhance the initial solution that consists of a perturbation phase followed by a local search, repeated until a stopping criterion is met. Specifically, the perturbation phase is based on a destruction/reconstruction procedure, where the base solution undergoes a partial destruction and subsequent reconstruction using two perturbation operators. Subsequently, the ILS employs a local search operator to explore the vicinity of the perturbed solution and search for improving solutions by employing five distinct local search operators.

4 Computational Experiments

The set of published TOARP benchmark instances proposed by Archetti et al. [2014] has been used to assess the performance of the proposed BR-ILS algorithm. The benchmark instances consist of four sets of instances (R, G, D, and T50) considering 2, 3 and 4 vehicles. Computational time limits were set at 60s for class R instances and extended to 300s for classes D and G. Our solving methodology reaches the optimal solutions for class R. Moreover, it obtains percentage gaps of about 1.41% for class D, and 2.47% for class G with respect to the upper bound reported in the literature. In addition, the proposed algorithm obtains percentage gaps of about 0.73% for class D; and 1.46% for class G with respect to the optimal solutions. This results establish the competitiveness of the BR-ILS algorithm, particularly for instances where the optimal solution is known.

Moreover, the classical set of TOP instances has been modified to validate the capability of solving scenarios where the start and end depots are distinct. These modifications allow us to compare the performance of our BR-ILS algorithm with the best-known solutions of the adapted TOP instances. Thus, we can assess the effectiveness of our algorithm in solving problems with different start and end depots, while considering the unique characteristics of the TOARP.

5 Conclusions

This study proposes a novel BR-ILS algorithm for solving the TOARP, and an extended version considering different origin and destination depots for the vehicles. Our methodology combines a biased-randomisation strategy and an ILS framework to efficiently search for high-quality solutions. The BR-ILS algorithm has been tested on a set of previously published TOARP instances, and the results demonstrate the effectiveness of our algorithm in finding competitive solutions within short computational times.

References

Claudia Archetti, M. Grazia Speranza, Ángel Corberán, José M. Sanchis, and Isaac Plana. The team orienteering arc routing problem. *Transportation Science*, 48(3):442–457, 2014. ISSN 00411655, 15265447. URL `http://www.jstor.org/stable/43666696`.

Alejandro Estrada-Moreno, Martin Savelsbergh, Angel A Juan, and Javier Panadero. Biased-randomized iterated local search for a multiperiod vehicle routing problem with price discounts for delivery flexibility. *International Transactions in Operational Research*, 26(4):1293–1314, 2019.

Javier Panadero, Angel A Juan, Christopher Bayliss, and Christine Currie. Maximising reward from a team of surveillance drones: a simheuristic approach to the stochastic team orienteering problem. *European Journal of Industrial Engineering*, 14(4):485–516, 2020.

A Multi-Swap Heuristic for Rolling Stock Rotation Planning with Predictive Maintenance

Felix Prause
Zuse Institute Berlin
Berlin, Germany
prause@zib.de

ABSTRACT

We present a heuristic solution approach for the rolling stock rotation problem with predictive maintenance (RSRP-PdM). The task of this problem is to assign a sequence of trips to each of the vehicles and to schedule their maintenance such that all trips can be operated. Here, the health states of the vehicles are considered to be random variables distributed by a family of probability distribution functions, and the maintenance services should be scheduled based on the failure probability of the vehicles. The proposed algorithm first generates a solution by solving an integer linear program and then heuristically improves this solution by applying a local search procedure. For this purpose, the trips assigned to the vehicles are split up and recombined, whereby additional deadhead trips can be inserted between the partial assignments. Subsequently, the maintenance is scheduled by solving a shortest path problem in a state-expanded version of a space-time graph restricted to the trips of the individual vehicles. The solution approach is tested and evaluated on a set of test instances based on real-world timetables.

KEYWORDS

Rolling stock rotation planning, Predictive maintenance, Heuristic, State-expanded graph model, Integer linear program

1 INTRODUCTION

Planning rolling stock rotations is essential for the operation of rail transportation and has been studied in the literature for quite some time. However, against the backdrop of climate change and the associated decarbonization of the transport sector, rail transportation represents a possible solution. It can therefore be assumed that the volume of freight and passengers transported by rail will continue to increase, which will also increase the complexity of the train scheduling. In addition, the availability of sensors and the analysis of the data they provide by the application of machine learning or traditional data mining methods enables a predictive scheduling of the vehicle maintenance. There is therefore a need to develop solution approaches for the automated dispatching of vehicles that are capable of integrating predictive maintenance strategies.

1.1 Related Work

The rolling stock rotation problem (RSRP) has already been investigated by a great variety of authors. For an introduction we refer to [14]. On the one hand, the contributions can be distinguished by the applied maintenance regime: Either preventive time- or distance-based maintenance regulation are employed, see for example [14], a predictive maintenance strategy is used, e.g., [4, 10, 15, 18], or no maintenance is considered at all. For an overview on the literature concerning RSRP we refer to [17]. On the other hand, the presented approaches can be categorized by the employed solution methods. The commonly utilized approaches are the direct application of integer linear programs (ILP), column generation, or the usage of heuristics, see [16]. In the following, we restrict ourselves to scenarios where maintenance is considered and focus on articles that apply heuristics to RSRP.

There exists a variety of heuristics that already have been applied to RSRP, but the most common ones are local search algorithms. Here, already determined solutions are modified to make them feasible or to improve their objective value. In [5], the RSRP is modeled by a sequence graph in which the trips correspond to nodes and the arcs indicate if two trips can be performed consecutively. They try to obtain a feasible solution by solving an ILP and employ a local search algorithm to include unassigned trips into the vehicle schedules if only a partial solution could be obtained. This is done by shifting trips between vehicles. This approach was further developed by [6], where an initial solution is derived from a stable set of trip nodes. Another local neighborhood search for the rescheduling variant of the RSRP was presented by [11]. Their approach is based on a space-time graph in which the nodes correspond to departure or arrival events of the trips, while the trips, waiting periods and deadhead trips are represented by the arcs. They apply a 2-opt heuristic to vehicles that meet at a station and interchange their subsequent trips.

Local search approaches are also applied to hypergraph formulations of the RSRP for maintenance scheduling. In [3], the authors state that the non-maintenance relaxation of their model is not that hard to solve. Therefore, a local neighborhood search is used to construct feasible solutions out of rotations that violate the maintenance constraints. This hypergraph model was subsequently used by other authors. In [1], a backtrack heuristic is given for the insertion of long-term maintenance services into predetermined rotations, while [8] present a heuristic relying on the observation that in real-life instances maintenance services can usually be performed during over-night stops. This yields a local neighborhood search for generating feasible solutions from maintenance-infeasible ones.

But also other types of heuristics have been applied to RSRP. In [17], the problem is formulated as a resource-constrained shortest path instance with side constraints in a space-time graph. The authors consider a scenario with short-term maintenance and solve the problem by applying a resource-constrained shortest path algorithm within a hill climbing heuristic. Finally, [4] present a variety of heuristics for RSRP with predictive maintenance. These include

a genetic algorithm and three greedy algorithms that take the remaining useful life (RUL) of the considered vehicles into account.

1.2 Predictive Maintenance

The underlying idea of predictive maintenance in this article is the assumption that maintenance decisions should rely on the predicted health states of the considered vehicles, which cannot be measured directly. If we consider for example the doors of the vehicles, then their conditions must be approximated by either observing the number of occurring opening-closing cycles or by deriving them from sensor measurements like the voltage applied to the actuators or the vibration of the bearings. Since measurement errors occur here and further uncertainties arise when determining the health states from these values, e.g., by applying machine learning methods, the health states must be regarded as uncertain. In addition, the future load and operating conditions of the doors have to be predicted, which further increases the arising uncertainty. We therefore assume that the health states must be treated as random variables. The maintenance decisions are thus based on the probability that these random variables exceed some predefined threshold indicating a failure. We will denote this probability as the failure probability.

1.3 Contribution

In this article, we introduce the rolling stock rotation problem with predictive maintenance (RSRP-PdM), describe a graph model that approximates this problem, and present a local search heuristic to solve it. The proposed solution approach is able to handle predictive maintenance scenarios, where the health states of the vehicles are considered to be random variables and the maintenance is scheduled based on the failure probability of the vehicles. However, it can easily be adapted to handle distance- or time-based maintenance strategies. It also allows non-linear functions for modeling the degradation of the health states. Finally, the algorithm is tested and evaluated on a set of instances based on real-world timetables.

2 PROBLEM FORMULATION

In RSRP-PdM, we are given a set of vehicles $\mathcal{V}$, where each vehicle $v \in \mathcal{V}$ possesses a health state $H_v \in \mathbb{R}$. Here, $H_v = 1$ indicates that the vehicle is as good as new, while $H_v \leq 0$ corresponds to a breakdown of the vehicle. These health states are random variables to reflect their uncertainty and are distributed by a family of probability distribution functions Π with parameter space $\Theta \subseteq \mathbb{R}^n$, i.e., we have $H_v \sim \Pi_\theta \in \Pi$ for some $\theta \in \Theta$. H_v represents all possible health states of v with their respective probability of occurrence. Next, each $v \in \mathcal{V}$ has an initial state $H_v^0 \sim \Pi_{\theta_0}$, described by a $\theta_0 \in \Theta$. During the operation of trips and other services, e.g., deadhead trips, the conditions of the vehicles deteriorate, which is expressed by updating the parameters of their health states.

Furthermore, $\mathcal{L}$ is the set of all considered locations and $\mathcal{K}$ is a finite time horizon. Moreover, we are given a timetable $\mathcal{T}$ consisting of individual trips that need to be operated. To each trip $t \in \mathcal{T}$ we associate a departure location $l_t^d \in \mathcal{L}$ and a departure time $k_t^d \in \mathcal{K}$, as well as an arrival location $l_t^a \in \mathcal{L}$ and an arrival time $k_t^a \in \mathcal{K}$. Additionally, each trip possesses a degradation function $\Delta_t : \Theta \to \Theta$ altering the parameters of the health state of the vehicle operating t. We assume Δ_t to be continuous and monotonically increasing, i.e., $\nabla_{e_i}\Delta_t(\theta) \geq 0$ for all $i \in \{1, \dots, n\}$ and $\theta \in \Theta$, but we do not require it to be linear. Note that we associate similar degradation functions with the other activities of the vehicles, i.e., with waiting at stations, deadhead trips, and maintenance services. Finally, $n_t \in \mathbb{N}$ determines how many vehicles are required to operate t.

We associate costs with each of the possible operations, i.e., trips, waiting, deadhead trips, and maintenance services, and assume that breakdown costs arise when a vehicle failure occurs.

Next, we call a vehicle rotation balanced if the number of vehicles at each location $l \in \mathcal{L}$ is equal at the beginning and at the end of the considered time horizon. This balancedness is important as it gives rise to schedules that can be repeated on a weekly basis.

The task of RSRP-PdM is then to assign a sequence of trips, deadhead trips, and maintenance operations to each vehicle such that all given trips are operated and the resulting rotations are balanced. Here, the objective is to find a solution with minimum total costs, taking into account the operating costs, maintenance costs, and the expected costs of vehicle failures.

3 UTILIZED GRAPH MODELS

The proposed algorithm relies on two different graph models, that are presented in this section. The first is the *space-time graph*, which is widely used in the literature to model the RSRP. We present it briefly in the following and refer to [7] for a more detailed description. Afterwards, we introduce the *state-expanded event-graph*, which is a parameter-expanded version of the space-time graph and has been utilized in [13].

3.1 The Space-Time Graph

Given a RSRP-PdM instance as described in Section 2, the nodes of the space-time graph represent the departure and arrival events of the trips contained in $\mathcal{T}$. Each trip $t \in \mathcal{T}$ therefore induces two nodes $v_t^d = (l_t^d, k_t^d)$ and $v_t^a = (l_t^a, k_t^a)$, and corresponds to the arc $a_t = (v_t^d, v_t^a)$. By iterating over all trips and collecting the resulting nodes and arcs, we obtain the set of departure nodes V_+, the set of arrival nodes V_-, and the set of trip arcs $A_\mathcal{T}$. Afterwards, we add artificial start and end nodes for each location, i.e., $v_l^0 = (l, 0)$ and $v_l^\infty = (l, k_{\max})$ for each $l \in \mathcal{L}$, where $k_{\max} := \max\{\mathcal{K}\}$. These nodes form the sets of the start nodes V_0 and the end nodes V_∞. Thus, we define the node set to be $V := V_0 \cup V_+ \cup V_- \cup V_\infty$.

Next, we construct the arcs of the graph. First, we consider arcs representing that a vehicle waits at its current location. For this purpose, the nodes of each location $l \in \mathcal{L}$ are sorted in time-ascending order and an arc is added between each pair of time-consecutive nodes. This yields A_W. Finally, we construct the deadhead arcs A_D. Therefore, we iterate over all nodes $v_1 = (l_1, k_1) \in V_0 \cup V_-$ and add an arc to each $v_2 = (l_2, k_2) \in V_+ \cup V_\infty$ with $l_1 \neq l_2$, which has the smallest k_2 among the nodes at l_2 such that $k_1 + k(l_1, l_2) \leq k_2$, where $k(l_1, l_2)$ is the time required to travel from l_1 to l_2.

Combining these arc sets yields $A := A_\mathcal{T} \cup A_W \cup A_D$ and $G_{ST} = (V, A)$ is the resulting space-time graph. Note that we assign each arc the costs associated with its corresponding operation.

3.2 The State-Expanded Event-Graph

The space-time graph just described in Section 3.1 is well suited to determine assignments of trips to vehicles, however it is not

trivial to incorporate maintenance constraints into it. Usually, these constraints are modeled by considering the vehicle rotations as resource-constrained paths, where a maintenance service is required when a certain resource threshold is exceeded and a resource consumption is associated with each of the arcs. The arising problem becomes even more complicated if we consider non-linear degradation functions, i.e., non-linear resource consumption. However, this should not be excluded, as mechanical components generally exhibit non-linear deterioration behavior. Therefore, we utilize the state-expanded event graph G_{SE}, which provides a linear approximation to this non-linear problem.

Given a discretization $\mathcal{D}$ of the parameter space Θ, i.e., a finite set $\mathcal{D} \subseteq \Theta$, we construct the nodes of G_{SE} by creating multiple copies of the nodes in G_{ST} for each $\theta \in \mathcal{D}$. This yields the node set $V' := \{(l, k, \theta) \mid (l, k) \in V(G_{ST}), \theta \in \mathcal{D}\}$ of G_{SE}.

Next, we generate the arcs. As in the construction of G_{ST}, each arc corresponds to a particular operation and the idea is that the arcs implicitly model the degradation, i.e., the resource consumption, of this operation. A vehicle traversing arc $a = (v_1, v_2)$ from $v_1 = (l_1, k_1, \theta_1)$ to $v_2 = (l_2, k_2, \theta_2)$ has a health state distributed by Π_{θ_1} before performing the task corresponding to a and a health state distributed by Π_{θ_2} afterwards. Here, the update of the health state parameters is given by the degradation function Δ_a of the arc, which is the degradation function of the associated operation.

But we do not necessarily have $\Delta_a(\theta_1) \in \mathcal{D}$, so there does not have to exist a head for a in V'. To resolve this problem, we define the following function that rounds to the nearest element of $\mathcal{D}$:

$$\lfloor \cdot \rceil_{\mathcal{D}} : \Theta \to \mathcal{D}, \quad \lfloor \theta \rceil_{\mathcal{D}} := \operatorname{argmin}_{\varphi \in \mathcal{D}} \{\|\theta - \varphi\|_2\}$$

Using this function, we construct the arcs of G_{SE} by iterating over its nodes and copying the outgoing arcs of their counterparts in G_{ST}. Let therefore $v_1 = (l_1, k_1, \theta_1)$ be a node in G_{SE}, and let $u_1 = (l_1, k_1)$ be the corresponding node in G_{ST}. Furthermore, consider any arc $a = (u_1, u_2) \in \delta^+(u_1)$, for some $u_2 = (l_2, k_2)$. Then, we determine $\theta_2 := \lfloor \Delta_a(\theta_1) \rceil_{\mathcal{D}}$ and add an arc from v_1 to $v_2 = (l_2, k_2, \theta_2)$. We repeat this procedure for all arcs originating from u_1 and subsequently for all nodes of G_{SE}. This results in the arc set A' of G_{SE} consisting of trip arcs, waiting arcs, and deadhead arcs. Note that this construction leads to multiple arcs corresponding to each of the trips.

Next, we introduce maintenance arcs. Their construction is similar to the construction of the deadhead arcs for the space-time graph and they are generated for every arrival node in G_{SE}. The difference in the construction is that instead of the time required to travel from location l_1 to l_2, the sum of the times necessary to travel from l_1 to the workshop, carry out the maintenance service there and then travel to l_2 is now considered. Moreover, as in the generation of the other arcs of G_{SE}, we apply $\lfloor \Delta_M(\cdot) \rceil_{\mathcal{D}}$ to the parameters of the tail node of each maintenance arc, where Δ_M is the degradation function associated with the maintenance activities. This resets the parameters to values that are as good as new.

Finally, the costs of the arcs in the state-expanded event-graph are equal to the costs of their corresponding arcs in the space-time graph, but we add the expected failure costs to the trip arcs. Therefore, we need to determine the failure probability of the vehicles during the operation of the trips. Consider any arc $a = (v_1, v_2) \in A'$ corresponding to some trip $t \in \mathcal{T}$. Let θ_1, θ_2 be the parameters of

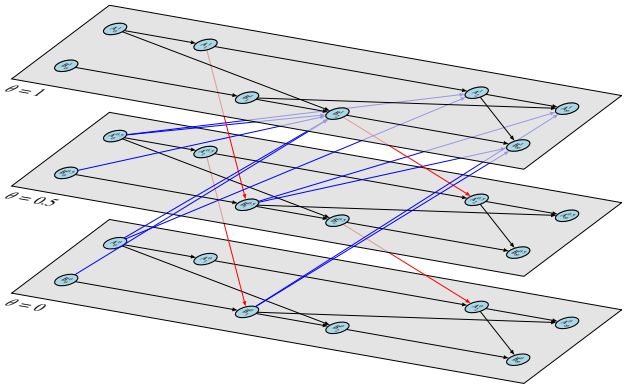


Figure 1: Example of a state-expanded event-graph adapted from [13].

v_1, v_2 respectively, then traversing a depicts that a vehicle v with health state $H_v \sim \Pi_{\theta_1}$ is operating t and its health state gets updated to $H_v \sim \Pi_{\theta_2}$ due to the occurring degradation. Thus, the failure probability of the vehicle is given by the probability that H_v exceeds the given failure threshold. Assuming that a breakdown occurs when H_v falls below zero, we need to determine $\mathbb{P}[H_v \leq 0] = \int_{-\infty}^{0} \Pi_{\theta_2}(x)\, dx$. The expected failure costs are then calculated by multiplying this value with the breakdown costs.

An example of a state-expanded event-graph is given in Figure 1, where the layers of nodes having the same parameter values are shaded in gray. The waiting and deadhead arcs are depicted in black, while the trip arcs are colored red. Traversing these arcs decreases the parameter by 0.5, while the maintenance arcs (blue) reset it to one. Note that a projection onto the space-time plane, i.e., ignoring the parameters, yields the underlying space-time graph.

4 ALGORITHM

Now, we present our solution approach for RSRP-PdM. This is based on the same observation as made in [3], where the authors state that the non-maintenance relaxation of RSRP is not that hard to solve. Thus, we first solve the RSRP ignoring the maintenance constraints and postpone the service scheduling to a subsequent step.

4.1 Generating Initial Solutions

To solve the non-maintenance relaxation of RSRP-PdM, we need to assign the trips to the vehicles in a cost-minimal way such that each trip is operated by the required number of vehicles and the vehicle balance of each location is even. Such an assignment corresponds to a flow in the space-time graph, which sufficiently covers the trip arcs and can be determined by solving ILP formulation (NMF).

In this formulation, $c_a \in \mathbb{R}_{\geq 0}$ are the costs associated with the arcs of the underlying space-time graph, and the objective function (1) aims at minimizing the total costs of all contained operations. Constraints (2) ensure the flow conservation and constraints (3) guarantee the balancedness of the resulting vehicle rotations. Constraints (4) depict the initial positioning of the vehicles in the beginning of the scenario, where $n_l \in \mathbb{N}_0$ is the number of vehicles located at $l \in \mathcal{L}$. Trip coverage is enforced by constraints (5), since

the required number of vehicles is assigned to each trip. Finally, the domains of the variables are defined in (6).

$$\text{(NMF)} \quad \min \sum_{a \in A} c_a x_a \tag{1}$$

$$\text{s.t.} \sum_{a \in \delta^+(v)} x_a = \sum_{a \in \delta^-(v)} x_a \quad \forall v \in V \setminus \{V_0 \cup V_\infty\} \tag{2}$$

$$\sum_{a \in \delta^+(v_l^0)} x_a = \sum_{a \in \delta^-(v_l^\infty)} x_a \quad \forall l \in \mathcal{L} \tag{3}$$

$$\sum_{a \in \delta^+(v_l^0)} x_a \leq n_l \quad \forall l \in \mathcal{L} \tag{4}$$

$$x_{a_t} = n_t \quad \forall t \in \mathcal{T} \tag{5}$$

$$x_a \in \mathbb{N}_0 \quad \forall a \in A \tag{6}$$

The solution of (NMF) represents a flow in the space-time graph given by a set of paths. Each path corresponds to the tasks assigned to one of the vehicles and consists of a sequence of trips connected by waiting or deadhead arcs. Since maintenance is not considered here, the path of each vehicle is determined by the assigned trips together with its origin and destination, as these have to be connected by the most cost-effective paths consisting of waiting and deadhead arcs.

Note that, the value of an optimal solution to (NMF) is a lower bound to the objective value of RSRP-PdM, since the costs can only increase if the deterioration of the vehicles, the associated expected failure costs and their maintenance are taken into account.

4.2 Scheduling Maintenance

After solving the non-maintenance relaxation of RSRP-PdM, we need to incorporate the maintenance services into the schedules of the individual vehicles. Therefore, we present a method that approximates the parameters of the health states of the vehicles and schedules the maintenance accordingly.

Recall that given a subset of trips $S \subseteq \mathcal{T}$ together with an origin $l_0 \in \mathcal{L}$ and a destination $l_\infty \in \mathcal{L}$, we can reconstruct the corresponding path in G_{ST} by sorting the trips in time-ascending order and connect the respective trip arcs by paths consisting of waiting and deadhead arcs. These paths will be termed *idle paths* in the following. We now transfer this idea to the state-expanded event-graph. Due to the construction of G_{SE}, based on a discretization $\mathcal{D}$, we may have multiple arcs corresponding to each of the trips. Consider a trip $t \in \mathcal{T}$ that starts at (l_t^d, k_t^d), then we added an outgoing arc corresponding to t to each node in $\{(l_t^d, k_t^d, \theta) \in V(G_{SE}) \mid \theta \in \mathcal{D}\}$. In the following, we will denote the set of arcs corresponding to a trip $t \in \mathcal{T}$ by $A(t)$. Thus, given a subset of trips $S \subseteq \mathcal{T}$ and $l_0, l_\infty \in \mathcal{L}$, we can determine a feasible schedule by selecting one arc from each $A(t)$, for all $t \in S$, and connecting them by idle paths. Note that the idle paths in G_{SE} can also contain maintenance arcs. In addition, the desired path must take into account the parameters of the initial health state of the assigned vehicle, i.e., $\theta_0 \in \Theta$, therefore it has to start at $v_0 = (l_0, 0, \lfloor \theta_0 \rceil_{\mathcal{D}}) \in V(G_{SE})$.

To find such a path, we first restrict G_{SE} to the arcs that are necessary for a vehicle that starts with parameters θ_0 at l_0, operates the trips in S and arrives at l_∞. We then determine a shortest path in this restricted graph. This graph will be referred to as $G_{SE}\big|_r$, for $r = (S, l_0, l_\infty, \theta_0)$, and can be obtained from G_{SE} as follows: First, we delete all trip arcs that belong to any $A(t)$, for $t \notin S$. Then, we remove all arcs that are not contained in an idle path that connects two of the remaining trip arcs. Next, all arcs that have a time overlap with one of the trip arcs are deleted. Finally, we add a sink node v_s to $G_{SE}\big|_r$ and add artificial arcs with costs equal to zero from all nodes in $\{v = (l_\infty, k_{\max}, \theta) \mid \theta \in \mathcal{D}\}$ to v_s.

S_1:	t_1	t_3, t_4, t_6	t_8	t_9, t_{11}		t_{15}, t_{17}
S_2:		t_2, t_5, t_7		t_{10}, t_{12}	t_{13}, t_{14}	t_{16}

Figure 2: Example of the exchangeable parts of two schedules.

Due to the construction of $G_{SE}\big|_r$, a shortest v_0-v_s-path thus corresponds to a minimum-cost schedule for $r = (S, l_0, l_\infty, \theta_0)$ w.r.t. the applied discretization $\mathcal{D}$. Note that the approximation quality depends on the granularity of $\mathcal{D}$. In the following, we assume that a solution x to the RSRP without maintenance is given by a set of schedules, where the schedule of each vehicle v_i can be represented as $s_i = (S_i, l_{0,i}, l_{\infty,i})$, for $i \in \{1, \ldots, |\mathcal{V}|\}$. Then, we can derive an approximate solution to RSRP-PdM from x by scheduling the maintenance of each vehicle, i.e., by determining a shortest path in each $G_{SE}\big|_{r_i}$, for $r_i = (s_i, \theta_{0,i})$. We will refer to this procedure by `scheduleMaintenance`(x, G_{SE}).

4.3 Improving Schedules by Swapping Trips

After presenting an approach to determine a solution for the non-maintenance relaxation of RSRP-PdM and a procedure for incorporating maintenance into the resulting vehicle schedules, we describe a local search algorithm that aims to improve a given solution by swapping parts of the vehicles' schedules.

Suppose we are given a non-maintenance solution x consisting of vehicle schedules $s_i = (S_i, l_{0,i}, l_{\infty,i})$, for $i \in \{1, \ldots, |\mathcal{V}|\}$. Then, we randomly select two vehicles $v_1, v_2 \in \mathcal{V}$ and consider their corresponding trip sets $S_1, S_2 \subseteq \mathcal{T}$. First, we sort S_1 and S_2 in ascending order of their departure times. Afterwards, we iterate over time-consecutive pairs $(t_{1,i}, t_{1,i+1})$ of S_1 and $(t_{2,j}, t_{2,j+1})$ of S_2 and check if it is possible to operate $t_{2,j+1}$ after $t_{1,i}$ and $t_{1,i+1}$ after $t_{2,j}$. If this is the case, (i, j) is a possible swap position. To find all swap positions, we next examine whether a swap is possible at the beginning or after the end of the schedules. Therefore, we check whether there are trips of S_1 that can be operated before the first trip of S_2 and if it is possible to reach them from $l_{0,2}$, i.e., the origin of v_2. The same is then repeated for S_2. Analogously, we check whether it is possible to swap trips after the last trip of S_1 or S_2.

These swap positions separate S_1 and S_2 into two ordered collections of equal cardinality. Their contained subsets of trips having the same indices can be exchanged without violating the feasibility of the resulting schedules. This procedure will be referred to as `getSwappingParts`(S_1, S_2) and an example is illustrated in Figure 2. Here, the trips contained in blocks standing underneath each other can be exchanged arbitrarily and both resulting schedules would be feasible. For example, it would be possible to swap $\{t_9, t_{11}\}$ and $\{t_{10}, t_{12}\}$, or to shift t_8 to S_2.

Since a swap can occur before the first trip of a schedule, it is possible that the initial departure location of a schedule is changed.

It could therefore be advisable to assign this trip sequence to another vehicle at another origin, which can reach the new departure location with a less expensive deadhead trip. We thus define `matchVehiclePositions`(x), which determines a matching between the vehicles and the schedules of x. For this purpose, we calculate the minimum costs of the idle paths that connect the origins of the vehicles with the initial departure location of the trip sequences, provided they can be reached in time. Then, we assign the vehicles to the sequences according to the solution of the minimum-cost matching. This yields the updated origins $l_{0,i}$ of the schedules. To ensure the balancedness, we then solve another minimum-cost matching, which assigns the sequences to the destinations of the vehicles such that each location occurs as a destination exactly as often as it was employed as an origin. This results in the updated destinations $l_{\infty,i}$.

Thus, given a non-maintenance solution x, we define the procedure `multiSwap`(x) as follows: First, we randomly select two schedules S_1 and S_2 contained in x and apply `getSwappingParts`(S_1, S_2) to determine the subsets of trips that can potentially be swapped between them. We then decide at random for each of the corresponding parts which part is assigned to S_1 and S_2, respectively. This results in a modified solution y. Subsequently, we reassign the vehicle origins and destinations to the trip sequences by applying `matchVehiclePositions`(y) and obtain the solution z, which is the result of the multi-swap procedure.

Note that `multiSwap`(x) is a generalization of 2-opt, since it is obtained by selecting a certain swap position and interchanging all subsequent trip parts, while leaving the parts before unchanged.

4.4 The Resulting Algorithm

Combining the procedures presented throughout this section, yields the multi-swap heuristic for RSRP-PdM, see Algorithm 1. First, formulation (NMF) is solved to generate an initial solution. If this formulation is infeasible, there cannot be a solution to RSRP-PdM since the ILP solves the non-maintenance relaxation of the problem. Subsequently, `multiSwap`(x) is utilized to modify the current best solution and thus to explore the solution space, while `scheduleMaintenance`(x, G_{SE}) is employed to schedule the maintenance based on the approximated health states of the vehicles. The algorithm is stopped when the given time limit is reached.

Algorithm 1: Multi-swap heuristic for RSRP-PdM

Data: RSRP-PdM instance $\mathcal{I}$, discretization $\mathcal{D}$
Result: Solution to $\mathcal{I}$ or infeasible

```
x ← solution to (NMF)
if x is infeasible then
    return infeasible
end
G_SE ← state-expanded event-graph based on D
sol ← scheduleMaintenance(x, G_SE)
repeat
    y ← multiSwap(x)
    z ← scheduleMaintenance(y, G_SE)
    if v(z) < v(sol) then
        x ← y
        sol ← z
    end
until time limit is reached
return sol
```

5 COMPUTATIONAL RESULTS

In this section, we present the results of the proposed solution approach to RSRP-PdM and compare them to the LP-based lower bound given in [13]. This lower bound is based on a relaxation of an ILP in which the vehicle rotations are represented by paths in G_{SE} and collectively cover each trip n_t times. The algorithm was tested on the data set provided in [12], which originates from genuine timetables. The characteristics of the individual instances are listed in the first four columns of Table 1. All instances are based on a rail network with a length of 4,221 km and three maintenance facilities. The health states are assumed to be distributed by two-parameter normal distributions and the trips possess non-linear degradation functions. The considered components are the doors of the vehicles, which are not safety-relevant. Thus, the vehicles can continue to be operated even after a failure, but this causes additional costs. It is assumed that the doors perform 1,500 opening-closing cycles before failing and undergo zero to four cycles at each stop of a trip.

5.1 Computational Setup

We performed all computations on a machine with Intel(R) Xeon(R) Gold 6342 @ 2.80GHz CPUs, eight cores and 64GB of RAM. The algorithm was implemented in Julia v1.9.1 [2] and Gurobi v10.0.2 [9] was used to solve (NMF) and the LPs for the lower bounds. All computations had a time limit of one hour.

5.2 Results

The results of the conducted computational experiments are given in the last five columns of Table 1. These contain the value of the best solution and the best lower bound for each instance, the resulting gap between these values, the gap after 180 seconds, and the time when the best solution was found. The obtained gaps show the effectiveness of the proposed algorithm, as they vary between 0 and 3.5% and are less than 4.3% after just 180 seconds.

The progression of the gap between the best heuristic result and the lower bound over time is quite similar across all instances and depicted in Figure 3. The majority and most significant improvements were achieved during the first 400 seconds. Afterwards, the solution value could still be enhanced, but the gained improvements decreased, and after 30 minutes almost no further progress could be recorded. An exception to this behavior is instance T5, where the best solution was found after just 14 seconds with a gap of 0.01%. These outcomes emphasize that the presented heuristic not only generates high-quality solutions, but is also capable of finding good results in a short time.

6 CONCLUSION

In this article we presented a heuristic solution approach to RSRP-PdM. We first defined the problem and then introduced two graph models: The first is used to model the non-maintenance relaxation of RSRP-PdM, while the other provides an approximation to the

Table 1: Characteristics and results for the test instances.

Instance	Trips	Destinations	Vehicles	Solution Value	Lower Bound	Gap in %	Gap after 180 s in %	Running Time in s
T1	566	8	6	269,728.67	261,432.23	3.08	3.08	3,517
T2	608	10	7	436,955.86	428,348.63	1.97	2.73	2,988
T3	636	15	16	1,427,088.22	1,380,028.25	3.30	4.28	3,597
T4	679	9	8	196,410.58	189,576.54	3.48	3.48	3,387
T5	813	16	14	327,804.96	327,770.13	0.01	0.01	14
T6	919	17	29	2,355,022.71	2,290,595.54	2.74	3.45	3,545

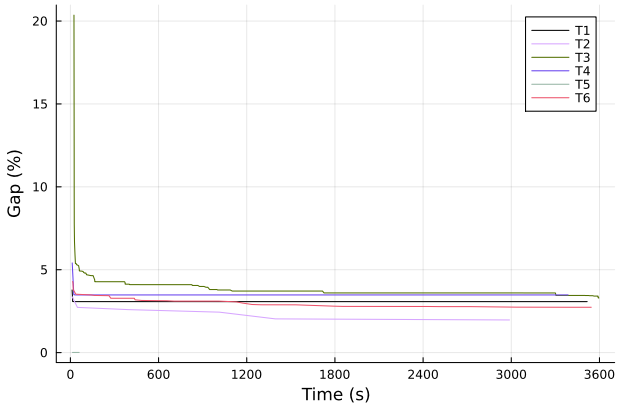


Figure 3: Gaps of all instances over time.

problem itself. Then, we discussed the individual steps of the proposed algorithm. These consist of an ILP to find an initial solution to the non-maintenance relaxation, a method for approximate maintenance scheduling and finally a local search procedure based on the multiple random swapping of trips. The effectiveness of the proposed algorithm is then demonstrated by conducting computational experiments on a set of test instances.

Possible next steps for future research are a combination of the presented approach with simulated annealing as well as the investigation of other meta-heuristics. Furthermore, it would be interesting to examine whether it is more effective to employ these heuristics on the non-maintenance relaxation and postpone the maintenance scheduling to a later step, or whether it is advantageous to apply these approaches directly to the state-expanded event-graph.

ACKNOWLEDGMENTS

The author would like to thank Boris Grimm for his comments and interesting discussions related to this work. This research was supported by the Pro FIT innovation funding program (grant no. 10174564) of the State of Berlin and co-financed by the European Union.

REFERENCES

[1] Timo Berthold, Boris Grimm, Markus Reuther, Stanley Schade, and Thomas Schlechte. 2019. Strategic planning of rolling stock rotations for public tenders. In *RailNorrköping 2019. 8th International Conference On Railway Operations Modelling And Analysis, Norrköping, Sweden, June 17th–20th, 2019.* Linköping University Electronic Press, 128–139.

[2] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. Julia: A fresh approach to numerical computing. https://julialang.org. *SIAM review* 59, 1 (2017), 65–98. https://doi.org/10.1137/141000671

[3] Ralf Borndörfer, Markus Reuther, Thomas Schlechte, Kerstin Waas, and Steffen Weider. 2016. Integrated optimization of rolling stock rotations for intercity railways. *Transportation Science* 50, 3 (2016), 863–877. https://doi.org/10.1287/trsc.2015.0633

[4] Omar Bougacha, Christophe Varnier, and Noureddine Zerhouni. 2022. Impact of decision horizon on post-prognostics maintenance and missions scheduling: a railways case study. *International Journal of Rail Transportation* 10, 4 (2022), 516–546. https://doi.org/10.1080/23248378.2021.1940329

[5] Valentina Cacchiani, Alberto Caprara, and Paolo Toth. 2012. A fast heuristic algorithm for the train unit assignment problem. In *12th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems.* Schloss Dagstuhl-Leibniz-Zentrum für Informatik. https://doi.org/10.4230/OASIcs.ATMOS.2012.1

[6] Valentina Cacchiani, Alberto Caprara, and Paolo Toth. 2019. An effective peak period heuristic for railway rolling stock planning. *Transportation Science* 53, 3 (2019), 746–762. https://doi.org/10.1287/trsc.2018.0858

[7] Jean-François Cordeau, François Soumis, and Jacques Desrosiers. 2001. Simultaneous assignment of locomotives and cars to passenger trains. *Operations research* 49, 4 (2001), 531–548. https://doi.org/10.1287/opre.49.4.531.11226

[8] Boris Grimm, Ralf Borndörfer, Markus Reuther, and Thomas Schlechte. 2019. A Cut Separation Approach for the Rolling Stock Rotation Problem with Vehicle Maintenance. In *19th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2019).* Schloss Dagstuhl-Leibniz-Zentrum für Informatik. https://doi.org/10.4230/OASIcs.ATMOS.2019.1

[9] Gurobi Optimization, LLC. 2023. Gurobi Optimizer Reference Manual, Version 10.0.2. http://www.gurobi.com.

[10] Nathalie Herr, Jean-Marc Nicod, Christophe Varnier, Noureddine Zerhouni, and Pierre Dersin. 2017. Predictive maintenance of moving systems. In *2017 Prognostics and System Health Management Conference (PHM-Harbin).* IEEE, 1–6. https://doi.org/10.1109/PHM.2017.8079111

[11] Rowan Hoogervorst, Twan Dollevoet, Gábor Maróti, and Dennis Huisman. 2021. A Variable Neighborhood Search heuristic for rolling stock rescheduling. *EURO Journal on transportation and logistics* 10 (2021), 100032. https://doi.org/10.1016/j.ejtl.2021.100032

[12] Felix Prause and Ralf Borndörfer. 2023. *Construction of a Test Library for the Rolling Stock Rotation Problem with Predictive Maintenance.* Technical Report 23-20. ZIB, Takustr. 7, 14195 Berlin.

[13] Felix Prause, Ralf Borndörfer, Boris Grimm, and Alexander Tesch. 2023. *Approximating the RSRP with Predictive Maintenance.* Technical Report 23-04. ZIB, Takustr. 7, 14195 Berlin.

[14] Markus Reuther. 2017. *Mathematical optimization of rolling stock rotations.* Ph.D. Dissertation. Technische Universität Berlin (Germany). https://doi.org/10.14279/depositonce-5865

[15] Pegah Rokhforoz and Olga Fink. 2021. Hierarchical multi-agent predictive maintenance scheduling for trains using price-based approach. *Computers & Industrial Engineering* 159 (2021), 107475. https://doi.org/10.1016/j.cie.2021.107475

[16] Thomas Schlechte, Christian Blome, Stefan Gerber, Stefan Hauser, Jens Kasten, Gilbert Müller, Christof Schulz, Michel Thüring, and Steffen Weider. 2023. *The Bouquet of Features in Rolling Stock Rotation Planning.* Technical Report. EasyChair.

[17] Per Thorlacius, Jesper Larsen, and Marco Laumanns. 2015. An integrated rolling stock planning model for the Copenhagen suburban passenger railway. *Journal of Rail Transport Planning & Management* 5, 4 (2015), 240–262. https://doi.org/10.1016/j.jrtpm.2015.11.001

[18] Meng-Ju Wu and Yung-Cheng Lai. 2019. Train-set Assignment Optimization with Predictive Maintenance. In *RailNorrköping 2019. 8th International Conference on Railway Operations Modelling and Analysis (ICROMA), Norrköping, Sweden, June 17th–20th, 2019.* Linköping University Electronic Press, 1131–1139.

An improved variant of the Iterated Inside Out algorithm for solving the optimal transport DOTmark Instances

Roberto Bargetto[1], Federico Della Croce[2], and Rosario Scatamacchia[3]

[1]DIGEP, Politecnico di Torino, Torino, Italy, ✉ roberto.bargetto@polito.it
[2]DIGEP, Politecnico di Torino, Torino, Italy, ✉ federico.dellacroce@polito.it
[3]DIGEP, Politecnico di Torino, Torino, Italy, ✉ rosario.scatamacchia@polito.it

The optimal transport (OT) is one the most paradigmatic among network optimization problems. Solving to optimality OT implies finding the minimum cost transportation plan that moves quantities of a single item from a set of sources M to a set of destinations N. A connection among any pair of a source i and a destination j exists and is characterized by a unitary transportation cost c_{ij}. Given respectively a_i and b_j the item quantities available at the sources and requested by destinations, the problem can be formulated by means of the following LP model [3].

$$\min \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j \in N} x_{ij} = a_i \quad \forall\, i \in M \tag{2}$$

$$\sum_{i \in M} x_{ij} = b_j \quad \forall\, j \in N \tag{3}$$

$$x_{ij} \geq 0 \quad \forall\, i \in M,\, j \in N. \tag{4}$$

Since [3], a large literature florished on OT and primal network simplex (NS) algorithms, *e.g.*, [5], for the minimum cost flow problem (that generalizes OT) were denoted as the best performing approaches [4] for OT (performing also much better than LP-based algorithms). The interest in the OT has been renewed recently because of machine learning and artificial intelligence applications requiring the computation of the distance between pairs of images or probability distributions with source and destination quantities being pixel intensities or probability measures respectively. A set of image processing instances, called DOTmark, is nowadays the benchmark for OT and the NS of [2] was indicated in [7] as the best performing algorithm for those instances. Interior point based approaches, still less efficient than [2], were also proposed, see, e.g., [8].

Very recently, a novel exact algorithm for solving OT, called iterated Inside Out (IIO), has been proposed [1]. The strength of this new method relies on the fact that potentially many pivoting operations are performed for each computation of dual multipliers and reduced costs. IIO requires in input a basic feasible solution and is composed by two phases that are iterated until an optimal basic feasible solution is found. The first 'inside' phase progressively improves the current basic solution by increasing the value of several non-basic variables with negative reduced cost and typically outputs a non-basic feasible solution corresponding to an interior point of the constraints' set polytope. The second 'out' phase operates in the opposite direction by iteratively setting to zero several variables until a new improved basic feasible solution is reached. The basic version of IIO combined with the shielding neighborhood [6] shows up to be approximately twice faster than [2] on the DOTmark instances.

Here, we propose a variant of IIO devised for solving much more efficiently the DOTmark instances w.r.t. [1]. The new version of IIO solves the DOTmark instances exploiting the characterizing structure of the transportation costs: these costs depend on the couple of indexes (i, j) of the related problem variables and present strong regularity in the way they increase or decrease according to a change of index i or j. Given the dual multipliers of the current basic solution, the described structure of costs enables the algorithm to predict from scratch the positivity of a large set of reduced costs correspondingly

avoiding their computation. The new version of IIO computes, at each iteration, an *ad-hoc* subset of *neighbor* reduced costs that guarantees the optimality of the final solution but strongly reduces the total number of reduced costs to be computed. Let denote this neighbor subset as the DOTmark neighborhood (DM-NGH).

Computational experiments revealed DM-NGH to be very effective but characterized by a long-tail effect, namely a large subset of iterations (the final ones) showing marginal improvement of the objective function value. To overcome this phenomenon, we designed a second new neighborhood of a current basic solution as follows. For each variable $x_{i,j}$ of a DOTmark instance with image size L, the closest variables to $x_{i,j}$ in terms of transportation cost are variables $x_{i+1,j}$, $x_{i-1,j}$, $x_{i,j+1}$, $x_{i,j-1}$, $x_{i+L,j}$, $x_{i-L,j}$, $x_{i,j+L}$ and $x_{i,j-L}$. We denote by $8-set(x_{i,j})$ the corresponding subset of variables, and by 8-vars neighborhood (8V-NGH) the union of all $8-set(x_{i,j})$ subsets over all basic variables $x_{i,j}$. Note that 8V-NGH nearly always contains at least one variable with negative reduced cost. The new IIO for DOTmark instances first applies DM-NGH until the improvement of the objective function is regularly above a fixed threshold, then applies 8V-NGH until negative reduced costs are found, finally completes the optimization reapplying DM-NGH (often just to prove optimality).

In Table 1, we report a computation time comparison of three versions of IIO plus Bonneel's NS [2] on the DOTmark instances considered in [7]. All experiments ran as single thread processes on a laptop with a *11th Gen Intel Core i7-1165G7 2.80GHz* $\times$ *8* processor and 16GB of RAM, and running Ubuntu 20.04.5 LTS. Table 1 shows that the best version of IIO algorithm for the DOTmark instances strongly outperforms the current state-of-the-art approaches.

	IIO with shielding	IIO with DM-NGH	IIO with DM-NGH + 8V-NGH	Bonneel's NS
image size	[sec.]	[sec.]	[sec.]	[sec.]
32x32	0.13	0.10	**0.09**	0.22
64x64	2.41	1.45	**1.18**	5.79
128x128	175.74	41.67	**23.87**	335.24

Table 1: Algorithms performances on DOTmark instances

References

[1] R. Bargetto, F. Della Croce, and R. Scatamacchia. Iterated inside out: a new exact algorithm for the transportation problem. *https://arxiv.org/abs/2302.10826*, (available online), 2023.

[2] N. Bonneel, M. Van De Panne, S. Paris, and W. Heidrich. Displacement interpolation using lagrangian mass transport. *Proceedings of the 2011 SIGGRAPH Asia Conference*, pages 1–12, 2011.

[3] F.L. Hithcock. The distribution of a product from several sources to numerous localities. *Journal of Mathematics and Physics*, (20):224–230, 1941.

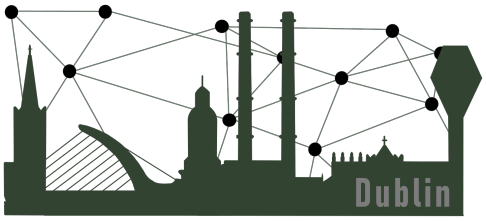
[4] P. Kovács. Minimum-cost flow algorithms: an experimental evaluation. *Optimization Methods and Software*, (30):94–127, 2015.

[5] J.B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows", mathematical programming. *Mathematical Programming*, 78(2):109–129, 1996.

[6] B. Schmitzer. A sparse multiscale algorithm for dense optimal transport. *Journal of Mathematical Imaging and Vision*, (56):238–259, 2016.

[7] J. Schrieber, D. Schuhmacher, and C. Gottschlich. Dotmark - a benchmark for discrete optimal transport. *IEEE Access*, (5):271–282, 2016.

[8] F. Zanetti and J. Gondzio. An interior point–inspired algorithm for linear programs arising in discrete optimal transport. *INFORMS Journal on Computing*, articles in press, 2023.



INOC - 2024

Session Session 5C: Graph Theory
Wednesday 13 March 2024, 11:00-12:30
Q013

On the k-slow Burning Conjecture

Arie M.C.A. Koster[1], Michaela Hiller[1], Jonas Kreyer[1], and Philipp Pabst[2]

[1]Discrete Optimization, RWTH Aachen University, Aachen, Germany, ✉ {koster,hiller}@math2.rwth-aachen.de
[2]Chair of Management Science, RWTH Aachen University, Aachen, Germany, ✉ philipp.pabst@oms.rwth-aachen.de

The graph burning problem studies the speed at which information can spread in graphs across their edges. Graph burning is carried out as a step-wise process on an undirected graph $G = (V, E)$, $|V| = n$, where in every step first, every burning vertex spreads the fire to its entire neighbourhood, before second, a new source of fire is ignited. If $(v_1, \ldots, v_n)$ is a sequence of vertices, such that, when choosing v_i as the i-th source of fire, G can be burned within t time steps, it is called a burning sequence for G. The aim is to choose the new sources of fire in a way that minimises the length of a burning sequence. The minimum number of steps necessary to ignite every vertex in G is denoted by the *burning number*, $b(G)$. For paths P_n and cycles C_n it is known that $b(G) = \lceil \sqrt{n} \rceil$. The *burning number conjecture* sattes that this value is an upper bound for all graphs.

In this talk, we discuss a recently introduced variant of the problem, k-slow burning, in which every burning vertex can only ignite up to k of its neighbours in each step of the burning process. We consider the complexity of computing the corresponding graph parameter, the k-slow burning number $b_s(k, G)$.

We prove NP-hardness on multiple graph classes, most notably the class of graphs of radius 1, where normal graph burning is solvable in polynomial time. Furthermore, we show that among all connected graphs on n vertices, the k-slow burning number of the star graph S_{n-1} is maximal for $k \in \{1, 2\}$ and asymptotically maximal for fixed $k \geq 3$. This observation leads to a generalisation of the burning number conjecture in regard to k-slow burning:

$$b_s(k, G) \leq \max\{b_s(k, P_n), b_s(k, S_{n-1})\}$$

for all connected graphs. By solving a mixed integer program for all relevant cases, we could confirm this conjecture for all graphs with at most 20 vertices. By the same approach the burning number conjecture could be verified for all graphs with at most 25 vertices.

Solving the multi-color Travelling Salesman Problem

Juan-José Salazar-González[1] and Roberto Wolfler-Calvo[2]

[1]IMAULL, University of La Laguna, Tenerife, Spain, ✉ jjsalaza@ull.es
[2]LIPN, Université Paris 13, Villetaneuse, France, ✉ rwolfler@lipn.univ-paris13.fr

We introduce a new variant of the well-known Travelling Salesman Problem (TSP) where each node is associated to a *color*, or cluster as in the Generalized Travelling Salesman Problem (GTSP). For that reason, the new problem is called *Multi-Color Travelling Salesman Problem* (MCTSP). We assume to work on a directed graph with costs associated with arcs. There are n nodes. Node 1 has its own color (say white), representing home and being the start and end of the route for a vehicle. For each color k, we are given with two positive numbers α_k and β_k such that $0 \leq \alpha_k \leq \beta_k \leq n$. The aim of the MCTSP is to find a minimum-cost Hamiltonian cycle visiting each node once, as in the classical TSP, with the additional requirement that the number of nodes between two nodes of a color k must be at least α_k and at most β_k.

The MCTSP is motivated by a more-complex and real-world problem called the Overnight Security Service Problem (OSSP) and heuristically addressed in [6]. The OSSP consists in optimizing the routes of a fleet of guards that should check buildings (customers) in an urban area for security reasons. The guards have to take care of three different types of services which are radically different. The first one is called *ticket* and is the simplest and cheapest service, as well as the most required since one (and only one) passage is necessary during the night. The second one is called *watch* and is a more careful check, requiring access to the building. It is repeated as many times as the customer requires, and the inspection time is recorded by a mechanical control system. The third type of service is called *alarm* and is disregarded in this paper since it rarely occurs and each security company manages it in a different way. This paper addresses the design of the a priori routes for the guards of the company to serve the tickets and the watches. In order to reduce the costs of the solution and to guarantee a better service to customers, the routes should be planned in advance.

The MCTSP is the particular case of the OSSP where we consider only one guard (vehicle) and the time requirement between two consecutive visits to a customer (building) is relaxed to only consider the number of other visits. It is related to the Black-and-While Travelling Salesman Problem introduced in [2], where a Hamiltonian tour is feasible when the number of white visits between two consecutive black visits and the time distance are upper bounded. See e.g. [4] for other variants of the well-known Travelling Salesman Problem (TSP). To the best of our knowledge the MCTSP is not mathematically modelled, nor solved, in the literature.

The MCTSP is closely related to the Capacitated Vehicle Routing Problem (CVRP) with Unit-Demand customers, where one limits the number of customers between two consecutive visits to the depot. It is also related to the CVRP with lower bound capacities [3] where one is also interested in visiting a minimum number of customers along each route. This relation inspires a formulation for the MCTSP based on multi-star like inequalities, suggesting a branch-and-cut approach for solving MCTSP to optimality.

Another related problem is the one-commodity pickup-and-delivery TSP with split demands, where a customer may need to be visited several times. This other relation motivates another formulation for the MCTSP in the line of the branch-and-cut approach in [1, 5].

In this paper we propose several mixed integer linear programming models and analyze computational results to solve instances to optimality.

References

[1] Güneş Erdoğan, Maria Battarra, and Roberto Wolfler Calvo. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *European Journal of Operational Research*, 245(3):667–679, 2015.

[2] Gianpaolo Ghiani, Gilbert Laporte, and Frédéric Semet. The black and white traveling salesman problem. *Operations Research*, 54(2):366–378, 2006.

[3] Luis Gouveia, Jorge Riera-Ledesma, and Juan-José Salazar-González. Reverse multistar inequalities and vehicle routing problems with a lower bound on the number of customers per route. *Networks*, 61(4):309–321, 2013.

[4] G. Gutin and A. P. Punnen, editors. *The Traveling Salesman Problem and Its Variations*. Springer, Boston, MA, 2002.

[5] Hipólito Hernández-Pérez and Juan-José Salazar-González. A branch-and-cut algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, 297(2):467–483, 2022.

[6] Roberto Wolfler Calvo and Roberto Cordone. A heuristic approach to the overnight security service problem. *Comput. Oper. Res.*, 30(9):1269–1287, 2003.

Compact and non-compact formulations for the Dominated Coloring Problem

Dilson Lucas Pereira
Departamento de Computação Aplicada, Universidade Federal de Lavras
Lavras, Minas Gerais, Brazil
dilson.pereira@ufla.br

Abilio Lucena
Programa de Engenharia de Sistemas e Computação, Universidade Federal de Minas Gerais
Rio de Janeiro, Rio de Janeiro
Brazil
abiliolucena@pesc.ufrj.br

Alexandre Salles da Cunha
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais
Belo Horizonte, Minas Gerais
Brazil
acunha@dcc.ufmg.br

ABSTRACT

Assume that a proper coloring (PC) is available for a given undirected graph $G = (V, E)$. Assume as well that all vertices in any of the color classes at hand are simultaneously dominated by a same vertex. Such a PC is then called a dominated coloring (DC) of G and the least number of color classes a DC might have, $\chi_{dom}(G)$, is called the dominated chromatic number of G. In turn, the problem of finding a DC of cardinality exactly $\chi_{dom}(G)$ is called the Dominated Coloring Problem (DCP). In this paper, we investigate two Integer Programming formulations for DCP and accompanying Branch-and-bound algorithms. One formulation relies on the concept of representatives and is strengthened with a set of valid inequalities that substantially improves its Linear Programming Relaxation bounds for sparse graph instances. The other is a set covering (SC) formulation that assigns binary variables to the maximal cliques in the open neighborhoods of every individual vertex of G. Our preliminary numerical results suggest that the clique formulation is, on average, 47% stronger than the formulation by representatives. Additionally, its corresponding Branch-and-bound algorithm also provides substantially better results, despite the fact that, at least for the moment, we explicitly enumerate and keep all necessary cliques, as opposed to resorting to a properly defined restricted master problem, in a column generation scheme.

1 INTRODUCTION

Let $G = (V, E)$ be a undirected graph with $n = |V|$ vertices and $m = |E|$ edges. The (open) neighborhood of $i \in V$, $N(i) = \{j \in V : \{i, j\} \in E\}$, corresponds to the set of vertices that share an edge of E with i. Vertex $i \in V$ dominates a set $S \subset V$ if and only if $S \subseteq N(i)$ applies. A proper coloring of G, or simply a coloring, is a function $c : V \to \{1, 2, \ldots, n\}$ such that no pair of adjacent vertices are colored with the same color. A *color class* $C_i = \{j \in V : c(j) = i\}$ corresponds to all vertices of G which are assigned a same color and consequently defines a stable set of G. A k-coloring of G is a partitioning of V into k color classes. Additionally, a k-coloring of G is *dominated* if and only if $C_i \subseteq N(u)$ holds for some $u \in V$, for every color class C_i in the partitioning. Assume, from now on, that when we say a vertex of V *dominates a color class* it implies that such a vertex dominates, i.e., is a neighbor to, all vertices in that class. The Dominated Coloring Problem (DCP) then asks a dominated k-coloring of G with k as small as possible, i.e., one for which $k = \chi_{dom}(G)$ applies. Note that according to such a definition, color classes may eventually contain a single vertex, provided G has no leaves.

Vertex coloring problems [8] are intensively investigated in the literature. This applies mostly due to their widespread theoretical and practical applicability and also to the fact that they are generally NP-complete. More recently, problems combining coloring and domination demands started to be investigated, DCP among them. In particular, DCP was introduced in [10] where its decision version was proven NP-Complete for arbitrary graphs with $\chi_{dom}(G) \geq 4$. Additionally, a polynomial time algorithm for recognizing graphs with $\chi_{dom}(G) \leq 3$ is also proposed in [10]. Besides its intrinsic theoretical importance, DCP finds applications in social networks [5], in genetic networks [6] for finding minimum groups of proteins satisfying some given types of interactions and in the interconnection of computer networks [7].

A problem that is closely related to DCP is the *Dominator Coloring Problem* (DtorCP) [4, 6]. A coloring is said to be feasible for it if every vertex of G dominates at least one color class, possibly its own color class (i.e., dominates all vertices in that class). Accordingly, among other differences, dominance requirements differ from DCP to DtorCP.

To the best of our knowledge, no Integer Programming (IP) formulations or IP based exact solution approach appears to exist for DCP. In this paper, we introduce two IP formulations, valid inequalities and Branch-and-bound algorithms for the problem. Apart from this introduction, the paper contains four additional sections. In Section 2, we present the formulations and in Section 3 we give some implementation details of our Branch-and-bound algorithms for solving them. Some preliminary computational results are reported in Section 4. We conclude the paper in Section 5, where we highlight our main findings and offer some directions for future research.

2 INTEGER PROGRAMMING FORMULATIONS

The presentation that follows relies on some standard notation in Graph Theory [2], summarized next. The closed neighborhood of i is $N[i] := N(i) \cup \{i\}$. The anti-neighborhoods of i are $\overline{N}(i) = V \setminus (N(i) \cup \{i\})$ and $\overline{N}[i] = V \setminus N(i)$. Pairs of vertices $\{i, j\}$ that are not neighbors in G are identified by the end points to the edge $e = \{i, j\} \in \overline{E}$, where $\overline{E}$ is the complement of E. Accordingly, $\overline{G}$ is the graph $(V, \overline{E})$, that complements G. We assume that G is connected since otherwise DCP would then decompose into various, smaller, DCPs, one for every connected component of the graph. We also assume that G has no leaves, since otherwise any leaf $i \in V$ might be colored with the same color of a vertex j adjacent to the only neighbor of i, p ($p \in N(i) \cap N(j)$), without increasing the chromatic number. Given a set $S \subseteq V$, the subgraph induced by S is $G[S] = (S, E(S))$, where $E(S)$ denotes the set of edges of E with both endpoints in S. Likewise, $\overline{G}[S] = (S, \overline{E}(S))$ denotes a subgraph of $\overline{G}$, where $\overline{E}(S) = \{\{i, j\} \in S : \{i, j\} \in \overline{E}\}$. A clique of G is a set $S \subseteq V$ such that $G[S]$ is complete, i.e., all vertices in S are pairwise neighbors in G. Thus, a stable set of G corresponds to a clique of $\overline{G}$, after we extend the clique definition to subsets of vertices of size 2 and 3, i.e., respectively edges and triangles. As usual, we assume that $\mathbb{B}$ is the set $\{0, 1\}$.

2.1 A formulation by representatives

The formulation uses two sets of decision variables, namely:

- $\mathbf{x} = \{x_{ij} \in \mathbb{B} : i \in V, j \in \overline{N}[i], j \geq i\}$. If $x_{jj} = 1$ applies, vertex j is colored with color $c(j) = j$, and is said to represent C_j, the color class containing it. Conversely, if $x_{jj} = 0$ holds, vertex j must belong to a color class represented by one of its anti-neighbors. In more detail, variable x_{ij} for $i < j$ is used to indicate whether or not vertex j belongs to the color class C_i. In case it does, $x_{ij} = 1$ must hold and both vertices are colored identically with color $c(i) = c(j) = i$ and the two vertices therefore belong to C_i. Otherwise, $x_{ij} = 0$ applies. Note that, in order to break formulation symmetries, variables x_{ij} are not assigned to anti-neighbors i, j such that $i > j$.
- $\mathbf{z} = \{z_u^p \in \mathbb{B} : u \in V, p \in N(u)\}$. Variable z_u^p is used to indicate whether or not $u \in V$ is the vertex chosen to dominate C_p, an eventual color class represented by p. In case it is the chosen vertex, all vertices colored with color $c(p) = p$ must be neighbors of u. Thus, if $z_u^p = 1$, $C_p \subseteq N(u)$ must hold.

DCP can be formulated as the following IP

$$\chi_{dom}(G) = \min \left\{ \sum_{i \in V} x_{ii} : (\mathbf{x}, \mathbf{z}) \in \mathcal{P}_r \cap (\mathbb{B}^{d_x}, \mathbb{B}^{d_z}) \right\}, \quad (1)$$

where the polyhedral set $\mathcal{P}_r$ is defined by constraints (2)-(7) and d_x and d_z respectively denote the dimension of $\mathbf{x}$ and $\mathbf{z}$. Accordingly, $\mathcal{P}_r$ is thus formulated as

$$\sum_{v \in \overline{N}[u], v \leq u} x_{vu} = 1 \qquad u \in V \quad (2)$$

$$x_{pi} + x_{pj} \leq x_{pp} \qquad \{i, j\} \in E, p \in \overline{N}(i) \cap \overline{N}(j), \; p < i, p < j \quad (3)$$

$$\sum_{u \in N(v)} z_u^v = x_{vv} \qquad v \in V \quad (4)$$

$$z_u^v + x_{vt} \leq x_{vv} \qquad u \in V, v \in N(u), \; t \in \overline{N}(u) \cap \overline{N}(v), v < t \quad (5)$$

$$x_{ij} \in [0, 1] \qquad i \in V, j \in \overline{N}[i], j \geq i \quad (6)$$

$$z_u^p \in [0, 1] \qquad u \in V, p \in N(u) \quad (7)$$

Constraints (2) enforce that all vertices of G are assigned to a color class and the number of classes is minimized by the objective function in (1). In turn, constraints (3) imply that a vertex p cannot represent the color of a anti-neighbor, say i, unless p is the representative of its own color class. They also enforce that no pair of neighbors i and j can be represented by a common anti-neighbor p.

The fact that every color class must be dominated by a vertex is ensured by constraints (4) and (5). Notice that constraints (4) imply that if v represents a color class, there must be another vertex u, in the neighborhood of v, such that $z_u^v = 1$ applies. Now, under the assumption that $z_u^v = 1$ holds, constraints (5) forbid vertex v to represent a vertex t outside the neighborhood of u, the vertex chosen to dominate the color class C_v. Note that a color class C_v may well be dominated by more than two neighbors of v. The formulation, however, requires that only one variable, say $z_u^v : u \in N(v)$, assumes value one in such cases.

Formulation $\mathcal{P}_r$ could be reinforced, for instance, by replacing inequalities (3) by its stronger *clique* form:

$$\sum_{i \in Q, i > p} x_{pi} \leq x_{pp}, p \in V, Q \text{ a maximal clique of } G[\overline{N}(p)]. \quad (8)$$

It can also be strengthened with the following set of valid inequalities:

$$x_{kt} \leq \sum_{i \in N(t) \cap N(k)} z_i^k, k \in V, t \in \overline{N}(k), t > k. \quad (9)$$

If $x_{kt} = 0$ applies, inequality (9) is trivially valid. Otherwise, if $x_{kt} = 1$ holds, color k must be dominated by a vertex in the open neighborhood of both k and t. Hence, inequalities (9) are valid for DCP.

For the moment, we do not use the stronger set (8) to enforce proper colorings of G. Thus, denote by $\mathcal{P}_r^+$ the polyhedral region $\mathcal{P}_r$ reinforced with constraints (9) only, i.e., $\mathcal{P}_r^+$ is defined by constraints (2)-(7) and (9).

2.2 A formulation based on cliques of $\overline{G}[N(u)]$

From the seminal work of Mehrotra and Trick [9] onwards, it became a common practice to use maximal cliques to formulate the vertex coloring problem and variants of it.

As a result, column generation based Branch-and-bound algorithms became the standard approach for solving coloring problems. Our additional DCP formulation complies with this standard. It is a set covering one that assigns binary variables to a subset of the cliques of $\overline{G}$. In doing so, the formulation makes sure that every vertex of G must be part of at least one clique. Given that color classes must be dominated by at least one vertex, one must only consider cliques contained in $\{\overline{G}[N(u)] : u \in V\}$.

Our formulation makes a clear distinction between cliques of sizes 3 or greater and cliques of size 2. As we shall discuss later on, among all cliques of size at least 3 for the graphs $\{\overline{G}[N(u)] : u \in V\}$, we can restrict ourselves to maximal ones.

Prior to introducing the formulation, additional notation is required. Firstly, let $\mathcal{Q}_u$ denote the set of all maximal cliques of $\overline{G}[N(u)]$ with at least 3 vertices. Accordingly, denote by $\mathcal{Q} = \bigcup_{u \in V} \mathcal{Q}_u$ the set of all these cliques in $\overline{G}[N(u)]$. Additionally, define the set $\overline{\delta}(u)$ as the subset of edges of the complement graph $\overline{G}$ that are incident to u. Furthermore, define $\overline{\delta}_R(u) = \{\{u, p\} \in \overline{\delta}(u) : u, p \in N(v) \text{ for some } v \in V\}$ as the subset of edges of $\overline{\delta}(u)$ whose endpoints share a common neighbor in E. Finally, define $\overline{E}_R = \bigcup_{u \in V} \overline{\delta}_R(u)$.

We now discuss the decision variables required by the formulation. Recall that DCP allows for color classes composed by singleton vertices. Since we assume that G is connected, any color class C_u composed by just a single vertex u can be dominated by a neighbor of u. In order to model the case where a vertex alone defines a color class, the formulation makes use of variables $\mathbf{w} = \{w_u \in \mathbb{B} : u \in V\}$. If $w_u = 1$, $C_u = \{u\}$, u represents itself and no other vertex in V. Otherwise, if $w_u = 0$ holds, vertex u must be part of a color class containing at least two vertices.

Note that if a color class is composed by just two vertices, say two anti-neighbors u and p, then these two vertices must define an edge of $\overline{\delta}_R(u)$ (and $\overline{\delta}_R(p)$) as they must have a common neighbor v that dominates them. To model these cliques, the formulation uses binary variables $\mathbf{y} = \{y_{up} \in \mathbb{B} : \{u, p\} \in \overline{E}_R\}$. Since two anti-neighbors u and p that do not share a common neighbor cannot define a color class, the formulation needs not to assign variables to edges in $\overline{E} \setminus \overline{E}_R$.

The formulation also uses binary decision variables $\lambda = \{\lambda^Q \in \mathbb{B} : Q \in \mathcal{Q}\}$ associated to the maximal cliques in $\mathcal{Q}$. If $\lambda^Q = 1$, all the vertices in Q define a color class. Otherwise, if $\lambda^Q = 0$ holds, at least one vertex in $\mathcal{Q}$ is colored differently from the others.

Our set covering based formulation is defined as

$$\chi_{dom}(G) = \min \left\{ \sum_{Q \in \mathcal{Q}} \lambda^Q + \sum_{u \in V} w_u + \sum_{\{p,q\} \in \overline{E}_R} y_{pq} : (\mathbf{w}, \mathbf{y}, \lambda) \in \mathcal{P}_c \cap (\mathbb{B}^n, \mathbb{B}^{|\overline{E}_R|}, \mathbb{B}^{|\mathcal{Q}|}) \right\}, \quad (10)$$

where $\mathcal{P}_c$ is the defined by constraints (11)-(14).

$$\sum_{Q \in \mathcal{Q}: u \in Q} \lambda^Q + \sum_{\{u,q\} \in \overline{\delta}_R(u)} y_{uq} + w_u \geq 1 \qquad u \in V \quad (11)$$

$$\lambda^Q \in [0,1] \quad Q \in \mathcal{Q} \quad (12)$$

$$w_u \in [0,1] \quad u \in V \quad (13)$$

$$y_{pq} \in [0,1] \quad \{p,q\} \in \overline{E}_R \quad (14)$$

In order to attest its validity, initially note that the definition of the decision variables enforces that all color classes are dominated by some vertex of V. Therefore the dominance property of our coloring is naturally enforced by set covering constraints (11), which also ensure that every vertex belongs to at least one color class. Since decision variables λ are assigned to maximal cliques of $\{\overline{G}[N(u)] : u \in V\}$ only, and not to general cliques of these graphs, the formulation must impose a covering of the vertices of V and not a partitioning of them. Additionally, in order to certify that assigning a same vertex to two distinct color classes does not represent a problem, let $(\hat{\lambda}, \hat{\mathbf{w}}, \hat{\mathbf{y}})$ be an optimal solution to (10) with $\hat{\chi}_{dom}(G)$ color classes. Suppose as well that u belongs to two or more color classes in such a solution. A dominated coloring of G containing no more than $\hat{\chi}_{dom}(G)$ color classes and such that every vertex of G belongs to a single color class, may be obtained directly from $(\hat{\lambda}, \hat{\mathbf{w}}, \hat{\mathbf{y}})$, as follows:

- If $\hat{w}_u = 1$ and either $\hat{y}_{up} = 1$ or $\hat{\lambda}^Q = 1, u \in Q$, applies, one may safely set $\hat{w}_u = 0$, and obtain a solution with $\hat{\chi}_{dom}(G) - 1$ color classes, contradicting the optimality of $(\hat{\lambda}, \hat{\mathbf{w}}, \hat{\mathbf{y}})$.
- If two or more cliques with at least two vertices contain the same u, we can remove u from all but one of them and an alternative optimal solution to (10), with $\hat{\chi}_{dom}(G)$ color classes is thus obtained. In particular, note that the resulting solution remains a proper coloring of G.

Our numerical results show that optimal solutions to (10) typically involve non-disjoint color classes. Accordingly, a fast post-processing procedure based on the second observation above suffices to obtain an alternative optimal solution, with every vertex of G belonging to exactly one color class.

3 ALGORITHMS FOR SOLVING THE FORMULATIONS

We implemented two Branch-and-bound algorithms for solving DCP formulations $\mathcal{P}_r^+$ and $\mathcal{P}_c$, `BBR` and `BBCLK`, respectively. They are implemented in `Python 3.8.5` and rely on the `XPRESS` Mixed Integer Programming (MIP) suite [12], release 39.01.04, for carrying out Branch-and-bound demands. `XPRESS` thus takes care of solving Linear Programming Relaxations (LPRs) and managing the search tree. The solver separates general purpose cutting planes, implements branching and searches the Branch-and-bound tree according to its default policies. Aside from forbidding multi-threading, we changed no other default `XPRESS` parameters.

As formulation $\mathcal{P}_c$ involves exponentially many decision variables, a natural solution approach for solving it would resort to column generation, i.e., generating maximal cliques on-the-fly. Instead of that, we enumerate all cliques of $\mathcal{Q}$ and use them directly in the formulation. Accordingly, differently from most graph coloring algorithms introduced in the past twenty years, ours is a Branch-and-bound algorithm based on the full blown formulation $\mathcal{P}_c$ and not a Branch-and-price one.

The enumeration of maximal cliques of $\{\overline{G}[N(u)] : u \in V\}$ is carried out by our `C` implementation of the algorithm in [11], a variant of the widely known Bron-Kerbosh algorithm [3].

The most time consuming operation for loading formulation $\mathcal{P}_c$ into the MIP solver was not the enumeration of all required maximal cliques, but checking for duplicates among them. Since the same set Q may define a maximal clique for different graphs $\overline{G}[N(u)]$ and $\overline{G}[N(v)]$, we used a hash table to identify duplicate cliques. This hash table was implemented using standard `Python` data structures. As we shall see next, the number of variables appearing in our formulation tends to be relatively small. Additionally, the total CPU times taken to enumerate maximal cliques we require, to check for duplicate ones among them, and to get the algorithm up and running for solving our initial LPRs were not an issue.

4 PRELIMINARY NUMERICAL EXPERIMENTS

This section presents numerical results obtained with formulations $\mathcal{P}_r$, $\mathcal{P}_r^+$ and $\mathcal{P}_c$. We first compare the quality of their LPR bounds and then compare algorithms `BBR` and `BBCLK`, respectively based on $\mathcal{P}_r^+$ and $\mathcal{P}_c$. From now on, assume that $w(\mathcal{P}_r)$, $w(\mathcal{P}_r^+)$ and $w(\mathcal{P}_c)$ denote the LPR bounds associated with formulations $\mathcal{P}_r$, $\mathcal{P}_r^+$ and $\mathcal{P}_c$, respectively.

Our numerical investigation was conducted with two sets of test instances. One of them comprising 29 graphs frequently used to test exact solution algorithms for the Minimum Connected Dominating Set (MCDS) [1]. These are randomly generated instances with $n \in [30, 120]$ vertices and different graph densities, in the range $[5\%, 70\%]$. Instances are identified as v_n_den, where n gives the number of vertices for the corresponding connected input graph and den is the instance density. Additional details on how they were generated can be found in [1]. The other set comprises 8 benchmark instances for the Maximal Clique Problem, introduced in the 2nd DIMACS Challenge, being available at the web repository `https://iridia.ulb.ac.be/fmascia/maximum_clique`. Among the instances available there, we collected 8, with $n \leq 120$ vertices, namely: myciel4, myciel5, myciel6, hamming6-2, hamming6-4, johnson8-2-4, johnson8-4-4 and MANN_a9.

Table 1 presents some numerical results. Its first four columns provide the instance name, followed by n, m and graph density (den), $\frac{2m}{n(n-1)}$, in percentage values. The three subsequent columns indicate the dual bounds $w(\mathcal{P}_r)$, $w(\mathcal{P}_r^+)$ and $w(\mathcal{P}_c)$. Additionally, the table provides numerical results for the two algorithms under comparison, `BBR` and `BBCLK`. Each algorithm was allowed to run for 1800 seconds, for every instance involved. The results displayed for algorithm BBR are: the best lower (BLB) and upper (BUB) bounds attained during the search, the CPU time (t, in seconds) taken to solve the instance and the number of nodes investigated in the search. For algorithm `BBCLK`, the table provides an additional information, nv, the total number of decision variables needed to formulate the problem. If the best DCP lower and upper bounds found after hitting the imposed time limit do not match, a label "tl" indicates that the instance remained unsolved after 1800 CPU seconds. Numerical experiments were conducted with a 12 core Intel i7-5820K machine, running at 3.30GHz with 32Gbytes of shared RAM memory. Our clique enumeration algorithm was implemented in `C` and compiled with `gcc`, with optimization flag `-O3` turned on.

We first evaluate the impact of strengthening formulation $\mathcal{P}_r$ with valid inequalities (9). To that aim, some additional results are depicted in Figure 1. For each instance in our test set, we plot the LPR ratio $\frac{w(\mathcal{P}_r^+)}{w(\mathcal{P}_r)}$ in the horizontal axis and the graph density, den, in the vertical axis. Our results indicate that the inclusion of inequalities (9) impacts positively for sparse instances. For such cases, bounds $w(\mathcal{P}_r)$ frequently more than doubled, without significantly increasing the CPU time demands for their evaluation. In contrast, these inequalities brought no strengthening benefits for instances with input graph densities in the $[50\%, 70\%]$ range.

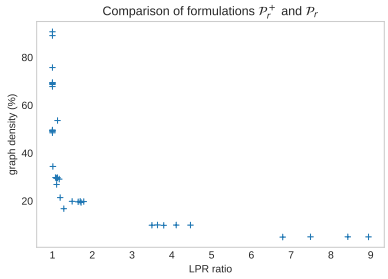


Figure 1: Comparison of formulations $\mathcal{P}_r^+$ and $\mathcal{P}_r$. The horizontal axis gives the ratio $\frac{w(\mathcal{P}_r^+)}{w(\mathcal{P}_r)}$ whereas the vertical axis gives the graph density.

In order to compare formulations $\mathcal{P}_r^+$ and $\mathcal{P}_c$, Figure 2 plots, for each instance, the LPR bound ratio $\frac{w(\mathcal{P}_c)}{w(\mathcal{P}_r^+)}$ (indicated in the horizontal axis) and the ratio between the CPU times taken to compute $w(\mathcal{P}_c)$ and $w(\mathcal{P}_r^+)$ (in the vertical axis). The CPU times we recorded for the computation of $w(\mathcal{P}_c)$ account for the time taken to enumerate cliques, for the checking of duplicate ones, as well as for solving the LPR itself. Measured by the gap $\frac{w(\mathcal{P}_c)-w(\mathcal{P}_r^+)}{w(\mathcal{P}_r^+)}$, formulation $w(\mathcal{P}_c)$ is about 47% stronger, on average, than $w(\mathcal{P}_r^+)$. Compared to $\mathcal{P}_r^+$, formulation $\mathcal{P}_c$ becomes stronger as the graph density increases. Notice that Figure 2 shows that the CPU times taken to compute $w(\mathcal{P}_c)$ are frequently below 50% of the

CPU times needed to compute $w(\mathcal{P}_r^+)$. That happens despite the fact that formulation $\mathcal{P}_c$ involves, at times, more than 150 thousand variables, all of them being explicitly used in the linear programming master program. Our numerical results thus lean in favor of formulation $\mathcal{P}_c$ not only in terms of bound quality but also in terms of the computational effort taken to upload and solve it.

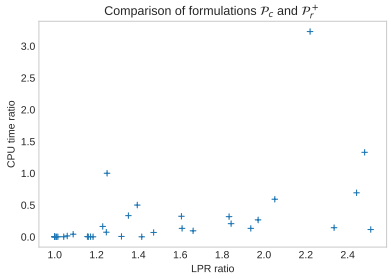


Figure 2: Comparison of formulations $\mathcal{P}_c$ and $\mathcal{P}_r^+$, in terms of LPR relaxation bounds and LPR CPU times. The horizontal axis gives the ratio $\frac{w(\mathcal{P}_c)}{w(\mathcal{P}_r^+)}$ whereas the vertical axis gives the ratio between the CPU time taken to compute bounds $w(\mathcal{P}_c)$ and the time needed to compute $w(\mathcal{P}_r^+)$.

We now discuss numerical results attained by Branch-and-bound algorithms `BBCLK` and `BBR`. Out of the 37 instances tested here, `BBCLK` and `BBR` respectively solved 30 and 24 instances to proven optimality, within the 1800 seconds time limit. All instances solved by `BBR` were also solved by `BBCLK`. While `BBCLK` takes less than 35 seconds to solve all these 24 instances, `BBR` takes more than 2744 seconds to accomplish that. `BBCLK` solved all instances with up to 70 vertices coming from the MCDS literature and failed to solve larger instances with densities in the intermediate range for our test bed. `BBCLK` solved all 8 maximum clique instances, whereas `BBR` solved 6 of them.

Considering now the 7 instances both algorithms left unsolved, `BBCLK` also has the edge for them. Best lower bounds provided by `BBCLK` are always stronger than `BBR`'s counterparts when the time limit is hit. Similarly, the BUB values attained by `BBCLK` are strictly smaller than `BBR`'s in 6 out of 7 cases. For just one case, both algorithms attained feasible solutions of the same value. At termination, `BBR` attains an average duality gap of 38.9% for these 7 instances, while the corresponding figure for `BBCLK` is just 15.8%.

Another interesting result is that `BBR` spent the entire CPU time at the root node when solving instance v120_d20. Notice that for this instance, the best lower bound attained by `BBR` (9.41) is strictly larger than the $w(\mathcal{P}_r^+)$ value (5.94), but smaller than the $w(\mathcal{P}_c)$ counterpart (10.89). `BBR` root node lower bounds are stronger than the $w(\mathcal{P}_r^+)$ values since `BBR` (as well as `BBCLK`) benefits from the general purpose cutting plane algorithm implemented by `XPRESS`, in the sense that after LPR bounds $w(\mathcal{P}_r^+)$ (and $w(\mathcal{P}_c)$) are computed, `XPRESS` adds some additional valid inequalities to the formulation at hand. However, for 8 out of the 13 instances not solved by `BBR`, the best `BBR` lower bounds, after the addition of these `XPRESS` cuts throughout the search tree and the enumeration of hundreds of `BBR` nodes, are weaker than LPR bounds $w(\mathcal{P}_c)$.

5 CONCLUSIONS

We investigated formulations, valid inequalities and Branch-and-bound algorithms for the Dominated Coloring Problem. Two Integer Programming formulations were proposed here. One is based on a model by representatives and the other makes use of exponentially many maximal cliques of the complement graphs induced by the open neighborhood of the vertices of G. Two Branch-and-bound algorithms based on these formulations were also numerically tested here. Our so far limited numerical experience suggests a clear advantage of the formulation based on cliques over the representatives. Linear Programming relaxation bounds for the clique formulation are about 47% stronger than those attained by the formulation by representatives, even after strengthening the latter with a set of valid inequalities introduced here. Better results were also provided by the Branch-and-bound algorithm based on the clique formulation, despite the fact that the algorithm enumerates and explicitly uses all decision variables in the model, without resorting to column generation.

The formulation by representatives could be further strengthened by separating the stronger form (8) of inequalities (3). In doing so, the resulting Branch-and-cut algorithm might become more competitive with the algorithm that relies on maximal cliques. The implementation of a Branch-and-price algorithm that prices cliques instead of explicitly using them in the model should also be an interesting research direction, that might improve the preliminary numerical results provided here.

REFERENCES

[1] Bernard Gendron and Abilio Lucena and Alexandre Salles da Cunha and Luidi Simonetti. 2014. Benders Decomposition, Branch-and-Cut, and Hybrid Algorithms for the Minimum Connected Dominating Set Problem. *INFORMS Journal on Computing* 26(4) (2014), 645–657.
[2] Adrian Bondy and U.S.R. Murty. 2008. *Graph Theory*. Springer.
[3] Coen Bron and Joep Kerbosh. 1973. Finding all cliques of an undirected graph. *Commun. ACM* 16 (1973), 575–577. Issue 9.
[4] Mustapha Chellali and Frédéric Maffray. 2012. Dominator Colorings in Some Classes of Graphs. *Graphs and Combinatorics* 28 (2012), 97–107.
[5] Yen Hung Chen. 2014. The Dominated Coloring Problem and Its Application. In *Computational Science and Its Applications – ICCSA 2014*, Beniamino Murgante et al. (Ed.). Springer International Publishing, 132–145.
[6] Sandi Klavzar and Mostafa Tavakoli. 2021. Dominated and dominator colorings over (edge) corona and hierarchical products. *Appl. Math. Comput.* 390 (2021). https://doi.org/10.1016/j.amc.2020.125647
[7] Minhui Li and Shumin Zhang Chengfu Ye. 2023. Dominated coloring in product graphs. *Journal of Combinatorial Optimization* 46, 4 (2023).
[8] Enrico Malaguti and Paolo Toth. [n.d.]. A survey on vertex coloring problems. *International Transactions in Operational Research* 17, 1 ([n. d.]), 1–34.

Table 1: Linear programming relaxation bounds and numerical results obtained by algorithms BBR and BBCLK.

Instance data				LPR bounds			BBR results				BBCLK results				
Inst	n	m	den	$w(\mathcal{P}\_r)$	$w(\mathcal{P}\_r^+)$	$w(\mathcal{P}\_c)$	BLB	BUB	t	nodes	nv	BLB	BUB	t	nodes
v30_d10	30	44	9.78	3.16	12.00	12.00	12.00	12.00	0.01	1	128	12.00	12.00	0.01	1
v30_d20	30	87	19.33	3.61	6.20	7.20	8.00	8.00	0.08	1	363	8.00	8.00	0.01	1
v30_d30	30	131	29.11	3.50	4.09	5.80	7.00	7.00	1.83	11	589	7.00	7.00	0.15	1
v30_d50	30	218	48.44	5.33	5.33	6.67	7.00	7.00	0.1	1	695	7.00	7.00	0.01	1
v30_d70	30	305	67.78	7.89	7.89	9.20	10.00	10.00	0.01	1	284	10.00	10.00	0.01	1
v50_d10	50	123	9.84	3.00	10.50	11.43	12.00	12.00	1.1	11	545	12.00	12.00	0.01	1
v50_d20	50	245	19.60	3.52	5.79	7.22	8.00	8.00	0.84	1	1443	8.00	8.00	0.02	1
v50_d30	50	368	29.44	4.05	4.46	7.17	8.00	8.00	66.6	1019	2767	8.00	8.00	0.3	1
v50_d50	50	613	49.04	5.02	5.02	9.24	10.00	10.00	6.96	3	3654	10.00	10.00	0.14	1
v50_d5	50	61	4.88	3.17	21.50	21.50	22.00	22.00	0.01	1	160	22.00	22.00	0.01	1
v50_d70	50	858	68.64	9.51	9.51	13.26	14.00	14.00	0.14	1	1094	14.00	14.00	0.02	1
v70_d5	70	121	4.94	2.63	23.50	23.71	24.00	24.00	0.05	1	439	24.00	24.00	0.01	1
v70_d10	70	242	9.88	2.55	11.41	12.09	14.00	14.00	14.35	89	1416	14.00	14.00	0.67	11
v70_d20	70	483	19.71	3.38	6.03	8.88	10.00	10.00	1572.98	19826	4400	10.00	10.00	21.73	1516
v70_d30	70	725	29.59	3.85	4.34	8.56	7.62	12.00	tl	6828	9969	10.00	10.00	21.57	1317
v70_d50	70	1208	49.31	5.32	5.32	10.91	10.55	13.00	tl	6393	13636	12.00	12.00	2.19	1
v70_d70	70	1691	69.02	10.42	10.42	16.73	18.00	18.00	9.16	255	2766	18.00	18.00	0.85	7
v100_d5	100	248	4.96	2.50	21.07	21.98	24.00	24.00	11.12	197	1201	24.00	24.00	0.36	11
v100_d10	100	495	9.90	2.67	10.96	12.67	15.00	15.00	829.04	4006	3717	15.00	15.00	9.45	1793
v100_d20	100	990	19.80	4.00	5.98	9.93	9.35	13.00	tl	261	17324	10.36	12.00	tl	88008
v100_d30	100	1485	29.70	4.00	4.47	10.43	8.05	16.00	tl	95	52674	10.81	12.00	tl	23237
v100_d50	100	2475	49.50	5.78	5.78	14.12	12.42	18.00	tl	543	62598	14.48	16.00	tl	25322
v100_d70	100	3465	69.30	10.91	10.91	21.13	21.29	23.00	tl	24682	7919	22.00	22.00	35.73	6957
v120_d5	120	357	4.96	3.00	22.46	22.80	25.00	25.00	24.43	101	2005	25.00	25.00	0.36	11
v120_d10	120	714	9.92	3.00	10.92	12.79	13.12	16.00	tl	1981	6235	14.22	16.00	tl	279751
v120_d20	120	1428	19.83	3.50	5.94	10.89	9.41	18.00	tl	0	45607	11.09	15.00	tl	13941
v120_d30	120	2142	29.75	4.40	4.73	11.74	8.35	20.00	tl	4	144023	11.83	15.00	tl	1937
v120_d50	120	3570	49.58	6.86	6.86	15.24	12.84	22.00	tl	104	150147	15.35	18.00	tl	2356
v120_d70	120	4998	69.42	9.44	9.44	23.70	23.24	26.00	tl	6090	14401	25.00	25.00	102.98	10632
myciel4	23	71	26.84	2.94	3.24	3.24	5.00	5.00	0.55	25	228	5.00	5.00	0.04	1
myciel5	47	236	21.37	2.98	3.55	3.55	6.00	6.00	203.19	23061	939	6.00	6.00	0.12	11
myciel6	95	755	16.73	2.98	3.83	3.83	4.72	7.00	tl	10582	3900	7.00	7.00	0.32	55
hamming6-2	64	1824	89.06	23.67	23.67	32.00	32.00	32.00	0.01	1	256	32.00	32.00	0.01	1
hamming6-4	64	704	34.38	4.00	4.04	5.33	5.66	7.00	tl	7325	2848	7.00	7.00	0.72	5
johnson8-2-4	28	210	53.57	4.20	4.73	5.60	6.00	6.00	1.16	87	420	6.00	6.00	0.14	1
johnson8-4-4	70	1855	75.71	11.39	11.39	14.00	14.00	14.00	0.36	1	1862	14.00	14.00	0.08	1
MANN_a9	45	918	90.67	15.00	15.00	18.00	18.00	18.00	0.0	1	129	18.00	18.00	0.01	1

[9] Anuj Mehrotra and Michael A Trick. 1996. A Column Generation Approach for Graph Coloring. *INFORMS journal on computing* 8, 4 (1996), 344–354.

[10] Houcine Boumediene Merouane, Mohammed Haddad, Mustapha Chellali, and Hamamache Kheddouci. 2015. Dominated Colorings of Graphs. *Graphs and Combinatorics* 31 (2015), 713–727.

[11] Pablo San Segundo, Jorge Artieda, and Darren Strash. 2018. Efficiently enumerating all maximal cliques with bit-parallelism. *Computers and Operations Research* 92 (2018), 37–46.

[12] FICO XPRESS. 2023. XPRESS mixed integer optimization package, release 8.13, Optimizer 39.01.04.

ACKNOWLEDGMENTS

Alexandre Cunha was supported by CNPq grant 305357/2021-2 and FAPEMIG grants CEX - PPM-00164/17 and RED-00119-21. Abilio Lucena was supported by CNPq grant 310185/2021-1.

When will the first collision occur?

Walid Ben-Ameur[1] and Alessandro Maddaloni[1]

[1]Samovar, Télécom SudParis, Institut Polytechnique de Paris, Palaiseau, France, ✉
walid.benameur,alessandro.maddaloni@telecom-sudparis.eu

Consider a set of robots simultaneously walking through a directed graph. The robots have the same speed and need one time step to go from a vertex to one of its neighbors. A robot can stay at a vertex only if there is a loop on that vertex.

A first natural question would be the following: can the robots walk forever without meeting (i.e., having collisions)? In this talk, a collision occurs if two robots are simultaneously on the same vertex. The answer will obviously depend on the structure of the graph and the number of robots. An immediate second question would be related to collision time: if we know that a collision will necessarily occur, then is there a time step for which we are sure that there should be at least one collision no later than that time?

The first question was answered in [1]. The second question will be answered here. Consider a directed graph $D = (V, A)$ where the minimum outdegree is at least one. A subset of vertices $S \subseteq V$ is called an independent set if one can build $|S|$ infinitely long walks (one starting from each vertex) that never meet. Let $\mathcal{I}$ be the set of all independent sets. It is shown in [1] that $(V, \mathcal{I})$ is a matroid (*the no-meet matroid*), denoted by $N(D)$. The set of no-meet matroids contains transversal matroids and is a strict subset of gammoids. A polynomial-time algorithm, based on flow techniques, is provided in [1] to check the independence of any subset S. The dimension of this matroid (i.e., the size of a largest independent set), denoted by $d(N(D))$ is shown to be equal to the maximum number of vertices belonging to a collection of vertex disjoint cycles. Dimension can also be computed in polynomial time using a matching algorithm.

Consider a number w of robots. Each robot will take a walk so w represents a number of walks. If $w \leq d(N(D))$, then one can build infinitely long walks, starting at some independent set, that never meet. On the other hand, if $w > d(N(D))$, then a collision will necessarily occur.

Let t_w be the first time step such that, given any set of w walks, at least two of them must meet no later than t_w. The main result presented in this talk is that:

$$t_w \leq |V| - w + 2 \tag{1}$$

for $d(N(D)) < w \leq |V|$. Notice that t_w is not defined if $w \leq d(N(D))$.

A connection is established between the meeting time t_w of walks and a cops and robber game on directed graphs with helicopter cops and an invisible slow robber. Let us first describe the game.

A set of cops wants to capture a robber moving on D with the following rules:

1. At step 1 the cops pick $W_1 \subseteq V$, then the robber picks a vertex $r_i \in V$.
2. At step $i + 1$ (for any $i \geq 1$) the cops pick $W_{i+1} \subseteq V$ and the robber picks $r_{i+1} \in N^+(r_i)$.

In other words, at each step the cops can pick any vertex, while the robber must pick a vertex adjacent from his current position (we use $N^+(v)$ to denote the set of vertices y such that (v, y) is an arc of D. If S is a set, $N^+(S) := \{y \in V(D) \mid \exists x \in S, \text{ with } (x, y) \in A(D)\}$.
We say that the cops capture the robber if $r_i \in W_i$ for some $i \geq 1$. The capture is at time t, if t is the minimum index such that $r_t \in W_t$. The cops do not know the vertex picked by the robber; their strategy is defined by the sequence $(W_i)_{i\geq 1}$, while the robber strategy is defined by $(r_i)_{i\geq 1}$. A cop strategy is winning if, by playing that strategy, they can capture the robber regardless of his strategy. The capture time of a cop winning strategy is the maximum time step T, over all possible robber's strategies, such that the cops capture the robber at time T. A cop winning strategy uses c attempts if $c = \sum_{i=1}^{T} |W_i|$, where T is the capture time of the cops' strategy. Notice that we can have $|W_i| = 0$ for some $i \in \{1, ..., T\}$.

Let R_i be the set of vertices where the robber can be at step i. Observe that if the robber was not yet captured at time step $i - 1$, then we have $R_i = N^+(R_{i-1}) \setminus W_i$.

The capture attempts number of D, denoted by $ca(D)$, is the minimum c such that there exists a cop winning strategy using c attempts. One can prove that:

$$ca(D) = d(N(D)). \tag{2}$$

The capture time using $c \geq ca(D)$ attempts, denoted by $ct(D, c)$, is the minimum capture time of a cop winning strategy using c attempts. When fixing a time limit l, we can define $ca(D, l)$ as the minimum number of capture attempts needed for the cops to capture the robber no later than time step l. Observe that $ca(D, l) \leq c$ is equivalent to $ct(D, c) \leq l$. One can prove that the $ca(D, l)$ and $ct(D, c)$ parameters can be computed in polynomial time.

Let us now go back to meeting time. We can show that the meeting time of w walks equals the capture time in this game, when at most $w - 1$ capture attempts are allowed. In other words, for $d(N(D)) < w \leq |V|$ and $c = w - 1$, we have:

$$t_w = ct(D, c = w - 1). \tag{3}$$

To illustrate the definitions and some of the results, consider the graph of Figure 1. The dimension of the no-meet matroid $d(N(D))$ is equal to 3. Using (2), we deduce that only 3 capture attempts are required to guarantee the capture of the robber. More precisely, we have $ca(D, 1) = 5$, $ca(D, 2) = 4$ and $ca(D, 3) = 3$ implying that 4 capture attemps are needed to guarantee a capture no later than time 2. This also implies that $ct(D, 3) = 3$, $ct(D, 4) = 2$ and $ct(D, 5) = 1$. From (3), the meeting time t_w is defined for $w > 3$ and is given by: $t_4 = ct(D, 3) = 3$ and $t_5 = ct(D, 4) = 2$.

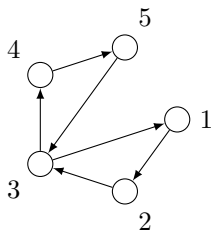
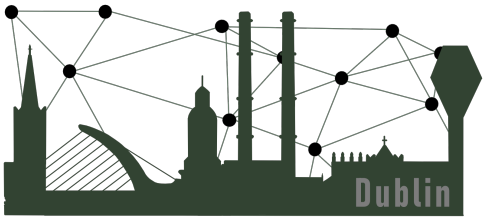


Figure 1: Example of a digraph D with two intersecting 3-cycles.

Other results related to the cops and robber game will be presented. For example, let us focus on the number of cops that are needed to capture the robber. Let $cn(D)$ be such number. Observe that we are not interested in the number of capture attempts but only in the minimum number of cops to capture the robber. Notice that the number of cops is simply given by $\max_{i=1}^{T} |W_i|$ if the robber is captured no later than time T. While $ca(D)$ can be computed in polynomial time, we show that $cn(D)$ is NP-hard to compute. It is also not difficult to prove that $cn(D)$ is smaller than the 1 + the pathwidth of D. One can also impose a time limit for capture and define $cn(D, l)$ as the minimum number of cops needed to guarantee capture no later than time l. Unfortunately, $cn(D, l)$ is also difficult to compute. If we consider again the graph of Figure 1, one can show that $cn(D, 1) = 5$, $cn(D, 2) = 2$ and $cn(D, 3) = 1$ implying that $cn(D) = 1$.

References

[1] Walid Ben-Ameur, Natalia Kushik, Alessandro Maddaloni, José Neto, and Dimitri Watel. The no-meet matroid. *Discrete Applied Mathematics*, 2022.



INOC - 2024

Session Session 6A: Integer Programming

Wednesday 13 March 2024, 14:00-15:00

Q011

Multi-depot split delivery of batches

Críston Pereira de Souza
Federal University of Ceará
Quixadá, Ceará, Brazil
criston@ufc.br

Andréa Cynthia Santos
LITIS, ISEL, Université Le Havre Normandie
Le Havre, France

ABSTRACT

This study focuses on a vehicle routing problem variant involving multiple depots and split deliveries with discrete deliveries and small integer vehicle capacities and demands. This can be interpreted as the batching of items (k batches per vehicle), and the problem is referred to as k-MD-DSDVRP. An Integer Programming (IP) formulation is proposed, as well as cuts to reduce symmetries. In addition, we discuss the existence of an optimal solution without split cycles and establish bounds for the ratio between the optimal values of k-MD-DSDVRP and MD-DSDVRP. Furthermore, a reformulation of k-MD-DSDVRP as an MDCVRP is presented, followed by a solution approach through RCSP-based map decomposition. Experiments using instances of MDSDVRP and SDVRP from the literature were conducted to evaluate the proposed method, with an analysis of the impact of using batches and a comparison of bounds of k-MD-DSDVRP and MDSDVRP.

KEYWORDS

Vehicle routing, multi-depot, split delivery, logistics, column generation.

1 INTRODUCTION

In this study, we focus on a vehicle routing problem that involves multiple depots and split deliveries. Split deliveries refer to a situation where a customer's demand can be delivered by more than one vehicle. Hence, "delivery" in this context refers to providing a part of a customer's demand supplied by a vehicle to a customer. Compared to the MDSDVRP [11] (a standard definition in the literature for this problem), we assume that the vehicle capacity is a small integer k and that customer demands are at most $k + 1$. These additional assumptions can be interpreted as grouping items into "batches." This variant is referred to here as k-MD-DSDVRP.

Motivations for investigating the k-MD-DSDVRP are listed below. Further discussion on the benefits of batching items can be found in [8].

- A MDSDVRP solution can have many tiny deliveries, which can be inconvenient for customers, causing interruptions to receive an insignificant portion of a demand.
- Fractional delivery can also be inconvenient to measure and control its amount on the fly. Using "batches" simplifies the process since supplies rely on integer values.
- From a logistics point of view, it is definitely interesting to simplify the preparation of deliveries through batches with many items.
- In a theoretical perspective, an algorithm for k-MD-DSDVRP can provide upper bounds for MD-DSDVRP. This is shown in Section 4.3.
- When k is small, it is possible to efficiently model the k-MD-DSDVRP as an MDCVRP [12] by creating k replicas of each client. Thus, allowing the use of successful algorithms for the MDCVRP available in the literature.

Contributions. This study brings the following contributions: (i) an IP formulation for the MD-DSDVRP with cuts to reduce symmetries; (ii) a discussion of the existence of optimal solution without split cycles and the relation between the number of splits and the number of routes; (iii) a transformation of instances and solutions between k-MD-DSDVRP and MD-DSDVRP, allowing to establish bounds for the ratio between the optimal values of these problems; (iv) a k-MD-DSDVRP model as an MDCVRP, with and a solution approach through RCSP-based map decomposition; (v) numerical experiments to evaluate the RCSP-based map decomposition and the inclusion of cuts to remove split cycles, using instances of MDSDVRP and SDVRP available in the literature.

The remaining of the manuscript is as follows. Section 2 presents an overview of the CVRP variants closely related to this study. Then, Section 3 defines the k-MD-DSDVRP and describes a proposed IP formulation. Section 4 discusses some properties of optimal k-MD-DSDVRP solutions and a comparison with optimal MD-DSDVRP solutions. Section 5 details an RCSP-based map decomposition approach for solving the k-MD-DSDVRP. Finally, in Section 6, numerical experiments and remarks are presented.

2 LITERATURE REVIEW

The goal of this Section is to provide entry points for articles defining closely related problems. The well-known *Capacitated Vehicle Routing Problem* (CVRP) is defined on an undirected and complete graph $G = (V, E)$ with a set of vertices $V = \{0, 1, \ldots, n\}$, where 0 represents the depot, the others vertices represent clients, and E is the set of edges. An unlimited number of homogeneous vehicles is available, each with a capacity $Q > 0$. A demand $0 < q_i \leq Q$ is associated with each customer i in $V \setminus \{0\}$. Each customer is visited exactly once, and their demand is fully supplied. Each edge $e \in E$ has an associated cost $c_e \geq 0$, satisfying the triangular inequality. The goal is to build a set of minimum-cost routes for the vehicles that meet all customers' demands and respect the vehicles' capacities.

The *Split Delivery Vehicle Routing Problem* (SDVRP) was formally defined by Dror and Trudeau [6, 7]. Unlike the CVRP, the SDVRP allows fractions of a customer's demand to be delivered by different

vehicles, such that the sum of the fractions equals the total demand of the customer. In the problems mentioned in this article, we only consider the case in which customer demands are at most the vehicle's capacity, even though some works in the literature investigate the more general case without this constraint.

Gulczynski et al. [10] investigated the SDVRP variant with the additional constraint that each delivery to a customer has a size of at least a certain fraction of its demand and called this problem the *Split Delivery Vehicle Routing Problem with Minimum Delivery Amounts* (SDVRP-MDA). The motivation is to avoid too small deliveries, which generate the inconvenience of managing the receipt of a quantity with little impact on the total expected delivery.

The *Multi-Depot Split Delivery Vehicle Routing Problem* (MDSDVRP) is a generalization of the SDVRP that allows more than one depot, as defined in [11]. Therefore, the set of vertices V is partitioned into a set of depots $D \subset V$ and a set of customers $C = V \setminus D$. In this case, we must decide the routes and which depot each vehicle will depart from. It is also necessary to ensure that each vehicle returns to the depot from which it left. When we add the restriction that exactly one route passes through each customer, we have the so-called *Capacitated Multi-Depot Vehicle Routing Problem* (MDCVRP) [12].

Discrete Split Delivery Vehicle Routing Problem (DSDVRP), proposed by Nakao and Nagamochi [14], considers demands and delivery sizes restricted to positive integers. To clarify the difference, we call this variant of the *Multi-Depot Discrete Split Delivery Vehicle Routing Problem* (MD-DSDVRP).

The use of integer demands and small integer vehicle capacity was investigated by Archetti et al. [2] for the SDVRP. They showed that the problem can be solved in polynomial time using a matching algorithm when the vehicle capacity is 2 and is NP-hard when the vehicle capacity is 3. Furthermore, they showed that in the case of vehicle capacity 3, the optimal value of CVRP (without allowing split deliveries) is at most 3/2 of the optimal value of SDVRP.

3 k-MD-DSDVRP

This study investigates the *Multi-Depot Discrete Split Delivery Vehicle Routing with Small Vehicle Capacity k* (k-MD-DSDVRP). Given a small positive integer k, the k-MD-DSDVRP is a variant of MD-DSDVRP, where vehicle capacity and demands are at most k and $k+1$, respectively. As the deliveries are integers, the vehicle capacity and customer demands are also considered integers. We denote by k-DSDVRP the problem k-MD-DSDVRP with a single depot.

Vehicle capacity, demands, and delivery sizes are integers in k-MD-DSDVRP. Thus, k is a "batch" where several items are grouped into a single one. In this way, the k-MD-DSDVRP can be interpreted as a "discretization" of the MDSDVRP when the batches are constructed respecting the vehicles' capacities and the customers' demands.

3.1 IP formulation

An IP formulation for the MD-DSDVRP was proposed by [16], inspired by the formulation of the split deliveries found in [4], and using the so-called MTZ subtour elimination constraints of [13].

Here, a multiflow formulation is proposed, where subtour elimination constraints found in [5] are used. The problem is defined in a complete directed graph $G = (V, A)$ with a cost $c_{ij} \geq 0$ assigned to each edge $(i, j) \in A$. This cost function satisfies the triangular inequality. G has a set of vertices $V = D \cup C$ divided into two disjoint sets : D (depots) and C (customers). Each customer $i \in C$ has a non-negative integer demand q_i. Moreover, there is a set of homogeneous vehicles R with capacity $Q = k$ to deliver all customer demands. Note that the expression $\sum_{i \in C} \lceil q_i / k \rceil$ is an upper bound on the number of vehicles. The objective is to minimize the total travel cost while fulfilling all customer demands, respecting vehicle capacities, and returning each vehicle to its initial depot. Additionally, the solution must determine the depot of each vehicle.

Equations (1a)–(1i) present an IP formulation for MD-DSDVRP. The binary variable x_{ij}^r is equal to 1 if and only if vehicle $r \in R$ uses arc $(i, j) \in A$. On the other hand, the integer variable y_i^r represents the amount that vehicle $r \in R$ delivers to customer $i \in C$. The objective is to minimize the total costs of the selected arcs, as shown in Equation (1a). Equation (1b) establishes that all vehicles entering a node must also leave it (flow conservation). Equations (1c) and (1d) ensure that each vehicle's route passes through exactly one depot and that there is no cycle on this route containing only customers. The delivery of the entire demand for each customer is guaranteed by Equation (1e). Equation (1f) only allows a vehicle to deliver to a customer if its route passes through it. Finally, the vehicle capacities are guarantee by Equation (1g).

$$\min \sum_{(i,j) \in A} \sum_{r \in R} c_{ij} \cdot x_{ij}^r \tag{1a}$$

$$\sum_{i:(i,v) \in A} x_{ij}^r - \sum_{j:(v,j) \in A} x_{ji}^r = 0 \qquad \forall v \in V,\ \forall r \in R \tag{1b}$$

$$\sum_{d \in D} \sum_{j:(d,j) \in A} x_{dj}^r = 1 \qquad \forall r \in R \tag{1c}$$

$$\sum_{\substack{(i,j) \in A \\ i,j \in S}} x_{ij}^r \leq |S| - 1 \qquad \forall S \subseteq C, r \in R \tag{1d}$$

$$\sum_{r \in R} y_i^r = q_i \qquad \forall i \in C \tag{1e}$$

$$q_j \cdot \sum_{i:(i,j) \in A} x_{ij}^r \geq y_j^r \qquad \forall j \in C,\ \forall r \in R \tag{1f}$$

$$\sum_{i \in C} y_i^r \leq Q \qquad \forall r \in R \tag{1g}$$

$$y_i^r \in \{0, 1, 2, \ldots, q_i\} \qquad \forall i \in C, \forall r \in R \tag{1h}$$

$$x_{ij}^r \in \{0, 1\} \qquad \forall (i, j) \in A, \forall r \in R \tag{1i}$$

3.1.1 Reduction of symmetries. All permutations of the vehicles produce equivalent solutions, thus generating many symmetries. However, it is possible to reduce this problem by ensuring that the order of vehicle indexes corresponds to the order of depot indexes. More precisely, if vehicle r passes through depot d and vehicle r' passes through depot d', with $d < d'$, then $r < r'$. The constraint is provided in Equation (2).

$$\sum_{j \in C} x_{dj}^{r'} + \sum_{j \in C} x_{d'j}^{r} \leq 1, \qquad \begin{matrix} \forall (d, d') \in D^2 : d < d', \\ \forall (r, r') \in R^2 : r < r'. \end{matrix} \tag{2}$$

4 OPTIMAL SOLUTIONS PROPERTIES

In this section, some properties of optimal k-MD-DSDVRP solutions are derived from existing studies. For the sake of clarity, *split cycle* is formally defined in Definition 1. First, we show that there is an optimal solution for the k-MD-DSDVRP without a *split cycle* and define the relation between the number of splits and the number of routes. Dror and Trudeau [7] proved this property for the problem with one depot and fractional deliveries, while Gouveia et al. [9] extended it for multiple depots. The difference from here is that the k-MD-DSDVRP deals with integer deliveries and multiple depots.

In the following, a transformation from MD-DSDVRP instances to k-MD- DSDVRP instances and transformations between solutions of these problems are presented, together with the optimal values correspondence obtained through these transformations. This allows the use of MD-DSDVRP instances from the literature by grouping the items into batches.

4.1 Existence of split cycles

Definition 1. Let R be a set of routes, and a support graph $H = (V, E)$ be an undirected graph where V is the set of customers and an edge $e = (u, v)$ belongs to E iff there exists a route r in R such that r passes through u and v. If C is a cycle of G, the nodes of C form a *split cycle* of R.

Property 2 (Dror and Trudeau, 1990 [7]). *If the edge costs satisfy the triangular inequality, then an optimal solution without split cycles exists for every feasible SDVRP instance.*

The authors in [7] proved this property for the SDVRP, and Gouveia et al. [9] have extended that for the MD-SDVRP. We claim that this property also applies to k-MD-DSDVRP since the exchange argument in the demonstration of [7] can also be extended for discrete deliveries. Note that the limit of k on the vehicle capacity is a characteristic of k-MD-DSDVRP inputs, not of its solutions.

Property 3. *If the edge costs satisfy the triangular inequality, then an optimal solution without split cycles exists for every feasible k-MD-DSDVRP instance.*

The existence of an optimal solution without split cycles does not apply to all CVRP variants. Indeed, Gulczynski et al. [10] showed that SDVRP-MDA instances exist for which all optimal solutions have a split cycle.

4.2 Number of splits and number of routes

The relation between the number of splits and the number of routes (employed vehicles) results from the existence of optimal solution without *split cycles*. This result can be applied to define cuts to remove solutions with split cycles, see Section 5.2.1.

Definition 4. The *number of deliveries n_i of customer i* is the number of routes that deliver a positive amount to i. The *number of splits for customer i* is defined as $n_i - 1$, and the *number of splits of a solution* is the sum of the number of splits among all customers.

Property 5 (Archetti et al., 2006 [3]). *If the edge costs satisfy the triangular inequality, then there is an optimal SDVRP solution where the number of splits is smaller than the number of routes.*

Indeed, Property 5 could be generalized to all sets of routes without a *split cycle*.

Property 6. *If the number of splits is at least the number of routes for some set of routes S, then S has a split cycle.*

Proof. Let $G(S \cup C, E)$ be a bipartite undirected graph where C is the set of customers with at least two deliveries in S, and we have an edge $(r, c) \in E$ if and only if the route $r \in S$ delivers to customer $c \in C$. By contrapositive, assume that S does not have a *split cycle*, and therefore G is acyclic. Thus, the number of edges $|E|$ is less than the number of vertices $|S| + |C|$. Since the number of splits m in S is equal to $|E| - |C|$, we conclude that $m = |E| - |C| < (|S| + |C|) - |C| = |S|$. □

4.3 MD-DSDVRP and k-MD-DSDVRP

In Section 4.3.1, a transformation from MD-DSDVRP instances to k-MD-DSDVRP instances are provided. Then, in Sections 4.3.2 and 4.3.3, the transformations between the resulting solutions are given. These transformations allow us to establish Theorem 7, which allows a correspondance between optimal values for the MD-DSDVRP and k-MD-DSDVRP.

4.3.1 From MD-DSDVRP instances to k-MD-DSDVRP instances. Let I be an instance of MD-DSDVRP with vehicle capacity Q and demand q_i for each customer i. For transforming I into a k-MD-DSDVRP instance I', items of I are grouped into batches of $B = \lfloor Q/k \rfloor$ items. Thus, the vehicle capacity of I' becomes $Q' = k$, and each customer's demand q_i becomes $q'_i = \lceil q_i/B \rceil$. Note that this transformation may produce some demands greater than the vehicle's capacity k (but not exceeding $k+1$), requiring at least two deliveries to these customers. For example, when $k = 3$, $Q = 4$ and $q_i = 4$, in instance I' the batch size is $B = 1$ and customer i has demand $q'_i = 4$, which is greater than the vehicle capacity $Q' = 3$.

4.3.2 From k-MD-DSDVRP solutions to MD-DSDVRP solutions. For every feasible solution S' for k-MD-DSDVRP, q'_i batches are delivered to each customer i. These q'_i batches can transport up to $B \cdot q'_i \geq q_i$ items, enough to satisfy the demand q_i of each customer i. As the sum of deliveries of each vehicle in S' is at most k, the total number of items transported by each vehicle is at most $k \cdot B \leq Q$. Therefore, the routes used in S' are feasible for MD-DSDVRP.

4.3.3 From MD-DSDVRP solutions to k-MD-DSDVRP solutions. The solution transformation preserves MD-DSDVRP routes, in spite of an increase on the number of vehicles. If y_{jr} denotes the amount delivered to customer j by route r of MD-DSDVRP, then this delivery can be made using $\lceil y_{jr}/B \rceil$ batches in k-MD-DSDVRP. Thus, the total number of batches required to make the deliveries of route r is $y'_r = \sum_{j \in C} \lceil y_{jr}/B \rceil$. As each vehicle in k-MD-DSDVRP delivers at most k batches, $\lceil y'_r/k \rceil$ vehicles in k-MD-DSDVRP are required to deliver the demands of each route r.

4.3.4 Comparing optimal values. When comparing the optimal value of two problems, say P_1 and P_2, we denote by $z(P)$ the optimal value of the problem P, and the comparison $z(P_1) \leq z(P_2)$ indicates that the optimal value of P_1 is less than or equal to the optimal value of P_2 whenever both problems have the same instance after the required adjustments.

Theorem 7. *Applying the instance and solution transformations described in sections 4.3.1, 4.3.2 and 4.3.3, we have that*

$$z(\text{MD-DSDVRP}) \leq z(k\text{-MD-DSDVRP}) \leq \left\lceil \frac{\min\{Q, n\}}{k} \right\rceil \cdot z(\text{MD-DSDVRP}), \tag{3}$$

where Q is the vehicle capacity of the MD-DSDVRP instance, and n is the number of customers. Besides, these bounds are tight.

Proof. The first inequality arises from the fact that the optimal k-MD-DSDVRP solution can be transformed into a feasible solution for the corresponding MD-DSDVRP instance without additional cost, as discussed in Section 4.3.2. The second inequality arises from a worst case for the transformation described in Section 4.3.3 when MD-DSDVRP demands are unitary, and each vehicle delivers to $\min\{Q, n\}$ customers. In this case, as the vehicles in k-MD-DSDVRP can deliver batches to a maximum of k vehicles, we need $\lceil \min\{Q, n\}/k \rceil$ vehicles for each route in the solution of MD-DSDVRP. According to the triangular inequality and the solution's optimality, the cost of delivering to a subset of customers on route r is not greater than the cost of r. Therefore, the solution for k-MD-DSDVRP can cost at most $\lceil \min\{Q, n\}/k \rceil$ times the cost of the optimal solution for MD-DSDVRP.

To show that the bounds are tight, consider an instance of MD-DSDVRP where all edges between a client and the single depot has cost one, and the cost of edges between customers is a tiny value ϵ. Therefore, the cost of any possible route converges to 2, remaining to count the number of routes to determine the cost of the solution. A tight example for the first inequality occurs when Q is a multiple of k, and all customers have demand Q. In this case, an optimal solution for both MD-DSDVRP and k-MD-DSDVRP consists of making exclusive deliveries to all customers; that is, each route delivers to only one customer. Note that when Q is a multiple of k, it is not possible to have a customer with demand greater than the vehicle's capacity in the transformation in Section 4.3.1. On the other hand, if the vehicle capacity in MD-DSDVRP is the number of customers and each customer has demand equal to one, then a single vehicle would be able to make all deliveries in MD-DSDVRP. However, it would require $\lceil n/k \rceil = \lceil Q/k \rceil = \lceil \min\{Q, n\}/k \rceil$ vehicles to carry out these deliveries in the k-MD-DSDVRP, from which we conclude that this is a tight example for the second inequality. □

5 RCSP-BASED MAP DECOMPOSITION

This section describes an RCSP-based map decomposition for k-MD-DSDVRP, that is, a column generation where the pricing problem is the *Resource Constrained Shortest Path* (RCSP), and sets of arcs in RCSP are mapped to integer variables of an IP model so that in the final solution each variable has value equals to the number of used mapped arcs in the RCSP solution.

The RCSP is defined over a directed graph $G(V, A)$ where V is the set of nodes, and A is the set of arcs. The set V contains two special nodes, s and d, representing the source and destination nodes, respectively. Each arc $a \in A$ has a cost and a consumption. The cost of a path p is the sum of the costs of its arcs, and the consumption of p is the sum of the consumption of its arcs. Each node $v \in V$ has a lower and upper bound for the available resource when the path enters the v. The cost of a set of paths is the sum of the costs of its paths. The objective of RCSP is to find a set with K paths from s to d of minimum cost such that each path starts with Q units of resource, and the available resource of each path passing through a node $v \in V$ respects the lower and upper bounds of v.

In the following, we define the input graphs for the pricing problem (RCSP) and then present a formulation for the k-MD-DSDVRP using the variables mapped on the arcs of these graphs. For clarity, the notation of Section 3.1 is adopted in the following sections.

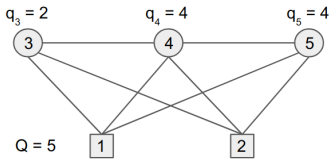
5.1 RCSP Graphs

Let $G^d = (V^d, A^d)$ be a directed graph for each depot $d \in D$. For each customer $i \in C$, we create a set $V_i^d = \{v_{i,1}^d, v_{i,2}^d, \ldots, v_{i,q_i}^d\}$ containing q_i replicas of customer i. Recall that q_i denotes the (integer) demand of customer i. Then, we set $V^d = D \cup \bigcup_{i \in C} V_i^d$, where D is the set of depots. The source and destination nodes are the depot $d \in D$.

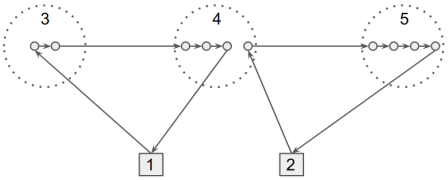
To reduce symmetries, two replicas of the same customer are connected by an arc if and only if they have consecutive indexes. That is, for each replica $v_{i,j}^d$, with $j < q_i$, the only arc going from $v_{i,j}^d$ to another replica of i is the arc $(v_{i,j}^d, v_{i,j+1}^d)$. All other possible arcs between nodes of V^d are included in A^d. The cost between replicas of the same client is zero, and each arc starting from a replica of client i to a replica of a different client j has a cost equal to $c_{i,j}$ (cost of the arc between customers i and j in the input graph). Similarly, the cost of an arc between a depot and a replica of client i is equal to the cost of the corresponding arc between this depot and client i in the input graph. Thus, when a route reaches a given customer, the number of consecutive replicas of this customer served by the route does not affect the route's cost.

A single resource is consumed on each route, representing the use of vehicle capacity. There is a unitary resource consumption whenever the route enters a node other than the depot, as each replica represents a unit of demand delivered to the customer. There is no resource consumption in the depot. We set zero and Q at each node as the accumulated resource consumption limits.

A constraint is added to the IP model to ensure that each client has exactly one delivery. This can be done through the map decomposition described in Section 5.2. Figure 1 provides an example of an k-MD-DSDVRP instance, and an RCSP optimal solution considering the additional constraint of exactly one route passing through each node. In Figure 1-(a), the k-MD-DSDVRP input graph has two depots represented by squares (nodes 1 and 2) and three customers represented by circles (nodes 3, 4, and 5). Each q_i represents the demand of customer i. Edge costs are the Euclidean distances between nodes, and the vehicle capacity Q is 5. An optimal solution with two routes is provided in Figure 1-(b). Assuming additional constraint of exactly one route passing through each node. Each customer i is represented by a dotted circle and is transformed into q_i replicas. Only arcs between consecutive replicas are allowed among replicas of the same customer, all with zero cost. The costs of the remaining arcs come from the input graph. The arcs returning to a depot have a consumption zero, and all the remaining arcs have a consumption one.



(a) k-MD-DSDVRP input graph.



(b) An optimal solution.

Figure 1: Example of k-MD-DSDVRP input graph and corresponding map decomposition optimal solution.

Complexity of transformation. Each client $i \in C$ has q_i replicas in each graph G^d, $d \in D$. Therefore, the resulting total number of nodes is $|D| \cdot \sum_{i \in C} q_i$. Among the replicas of the same customer, there are arcs only between consecutive replicas, but all possible arcs between replicas of different customers are created. Thus, the total number of arcs is $|D| \cdot (\sum_{i \in C}(q_i - 1) + \sum_{i \in C} q_i \sum_{j \in C:\, i \neq j} q_j)$. As $q_i \leq k + 1$ for all $i \in C$, we have that the total number of nodes is at most $(k + 1) \cdot |D| \cdot |C|$, and the total number of arcs is at most $|D| \cdot |C| \cdot (k + (k + 1)^2 \cdot (|C| - 1)) \in \Theta(k^2 \cdot |D| \cdot |C|^2)$.

5.2 Formulation and mapping of variables

Let C' be the set of replicas, and let $G'(C' \cup D, E')$ be an undirected graph where the vertices are all replicas and depots, and an edge $\{u, v\} \in E'$ if and only if $(u, v) \in A^d$ or $(v, u) \in A^d$ for any of the graphs $G^d(V^d, A^d)$, $d \in D$. In Equations (4a)–(4c), a formulation of k-MD-DSDVRP is given, where the decision variables x_e, $e \in E'$, are mapped into arcs of the RCSP graphs G^d, $d \in D$. That is, $x_{\{u,v\}}$ represents the number of routes passing through $\{u, v\}$ in the RCSP solution, taking into account all graphs G^d, $d \in D$. Thus, the formulation minimizes the sum of the costs of the edges used, subject to the condition that exactly one route passes through each replica. Let $\delta(v)$ be the set of edges in E' incident to the replica $v \in C'$, and thus, Constraint (4b) indicates that exactly two edges in $\delta(v)$ are used for each replica $v \in C'$. Note that this constraint affects all graphs G^d, $d \in D$, so that each replica $v \in C'$ is served by route in exactly one of these graphs. As k-MD-DSDVRP does not restrict the number of vehicles, the number of vehicles considered in the RCSP of each graph G^d, $d \in D$, may range from zero to the number of replicas.

$$\min \sum_{e \in E'} c_e \cdot x_e \tag{4a}$$

$$\sum_{e \in \delta(v)} x_e = 2 \qquad \forall v \in C' \tag{4b}$$

$$x_e \in \{0, 1, 2, \ldots\} \qquad \forall e \in E' \tag{4c}$$

Table 1: VRPSolver's non-default parameter values.

parameter	value
RCSPhardTimeThresholdInPricing	25
RCSPmaxNumOfLabelsInEnumeration	500000
RCSPmaxNumOfEnumeratedSolutions	5000000
RCSPrankOneCutsMemoryType	0
CutTailingOffThreshold	0.015
StrongBranchingPhaseOneCandidatesNumber	100
StrongBranchingPhaseOneTreeSizeEstimRatio	0.2
StrongBranchingPhaseTwoCandidatesNumber	3
StrongBranchingPhaseTwoTreeSizeEstimRatio	0.02
GlobalTimeLimit	3600

5.2.1 Valid inequality: split cycle removal. Equation (5) excludes solutions where the number of splits is greater than or equal to the number of routes. Then, by Property 6, only solutions with a split cycle are excluded. Besides, by Property 3, at least one optimal solution is preserved.

Let E_r be the set of all edges $\{u, v\} \in E'$ such that u and v are replicas of the same customer. The number of unused edges in E_r (i.e., with $x_e = 0$) provides the number of splits in the solution, as each unused edge implies an additional route delivering to the customer. Thus, the number of splits can be obtained through the expression $\sum_{e \in E_r}(1 - x_e)$. For example, Figure 1b depicts a solution with one split and one unused arc between the replicas of customer 4. If E_d is the set of arcs incident on any deposit, then the number of routes in the solution can be obtained by the expression $\sum_{e \in E_d} x_e/2$, as each route leaves and enters a single deposit just once.

$$2 \cdot \sum_{e \in E_r} (1 - x_e) \leq \sum_{e \in E_d} x_e - 2 \tag{5}$$

6 COMPUTATIONAL EXPERIMENTS

The experiments were conducted on an Ubuntu 16.04.7 LTS virtual machine with 8 Xeon 2.13GHz cores and 4MB cache, along with 24GB RAM. The VRPSolver 0.4.1 [15] was used for the implementation, with non-default parameters listed in Table 1, and IBM ILOG CPLEX 12.10 as the employed solver. Each replica node of graph G^d, $d \in D$ is defined as a vertex packing set and added a capacity cut separator with a limit of Q. We also assigned branching priority to the x variables.

The RCSP-based map decomposition method presented in Section 5 is tested using two instance sets available in the literature for MDSDVRP and SDVRP. The goal is to evaluate its effectiveness for $k \in \{3, 4, 5\}$. In addition, the impact of adding the cuts suggested in Section 5.2.1 to eliminate split cycles is analyzed.

Instances. Using the procedure described in Section 4.3.1, we adapted 42 instances from [9] to k-MD-DSDVRP and 89 instances from [1] to k-DSDVRP. None of the transformed instances had a demand that exceeded the vehicle's capacity.

6.1 Results

The results of the experiment are presented in Table 2, where the column "problem" is the instance type (k-DSDVRP for instances

Table 2: Summary of results.

problem	SCR	solved	gap	gap_MDSD	time
3-DSDVRP	N	97.7%	0.00%	54.6%	323
3-DSDVRP	Y	96.6%	0.00%	55.1%	288
3-MD-DSDVRP	N	81.0%	0.00%	16.2%	828
3-MD-DSDVRP	Y	81.0%	0.00%	16.2%	850
4-DSDVRP	N	86.4%	0.01%	43.3%	768
4-DSDVRP	Y	85.2%	0.00%	43.5%	771
4-MD-DSDVRP	N	54.8%	0.02%	14.8%	2187
4-MD-DSDVRP	Y	54.8%	0.04%	14.8%	2189
5-DSDVRP	N	72.1%	0.03%	39.3%	1351
5-DSDVRP	Y	70.9%	0.01%	39.4%	1345
5-MD-DSDVRP	N	32.5%	0.03%	12.0%	2667
5-MD-DSDVRP	Y	28.2%	0.07%	11.2%	2806

adapted from SDVRP and k-MD-DSDVRP for instances adapted from MDSDVRP), column "SCR" indicates the inclusion of split cycle removal constraints (with value 'Y' for yes and 'N' for no), column "solved" is the proportion of solved instances, column "gap" is the mean relative difference between the best lower and upper bounds, column "gap_MDSD" is the mean relative difference between the best upper bound and the best reported lower bound for the MDSDVRP, and column "time" is the mean execution time in seconds. A time limit of 1 hour was set for each instance. The following observations can be made:

- The number of solved instances drops quickly for k-MD-DSDVRP as k increases. However, it drops more slowly for k-DSDVRP. This suggests that the search for feasible solutions becomes more difficult as the number of depots increases.
- In all cases where a feasible solution was found, the gap was very small (below 0.1%). This indicates that the solutions produced are very close to being optimal.
- When the quality of the solutions was compared with the lower bounds provided by [9] for MDSDVRP, the k-MD-DSDVRP solution value was, on average, about 14.3% above. This indicates that integer deliveries and the batching of items into few batches per vehicle (3, 4 or 5) did not strongly affect the quality of the solution. However, this average difference was greater for k-DSDVRP, reaching 54.6% for $k = 3$, but this difference decreases whenever k increases.
- The execution time increases with the number of depots and the value of k, as they reflect the number of graphs and the number of replicas per customer, respectively.
- The inclusion of split cycle constraints worsened the average of all metrics, with a relative difference of around 30% for the gap, and less than 2% for the other metrics.

7 CONCLUDING REMARKS

This study investigates the k-MD-DSDVRP, which is convenient for daily logistics by organizing deliveries in batches. Properties are derived for this problem, and a formulation and an RCSP-based map decomposition are proposed. The experimental results show that the proposed map decomposition finds a near-optimal feasible solution for more than 80% of the instances for $k = 3$, but this percentage drops as k increases. It is also possible to conclude that the effect of grouping into batches and discrete deliveries is small (average gap of 14.3%) for instances with multiple deposits.

Several avenues of research are opened, such as investigating the impact of parameter k on the effectiveness of the dynamic programming method proposed in [8] for DSDVRP and the approach presented in [9] for MDSDVRP. Moreover, alternative solution transformations may provide tighter bounds than Theorem 7 for the ratio of $z(k\text{-MD-DSDVRP})$ and $z(\text{MDSDVRP})$. It is also worth investigating whether the demonstration of [2] for the NP-hardness of 3-SDVRP can be adapted to multi-depot and integer deliveries.

REFERENCES

[1] Claudia Archetti, Nicola Bianchessi, and Maria Grazia Speranza. 2011. A column generation approach for the split delivery vehicle routing problem. *Networks* 58, 4 (2011), 241–254.
[2] Claudia Archetti, Renata Mansini, and M. Grazia Speranza. 2005. Complexity and reducibility of the skip delivery problem. *Transportation Science* 39, 2 (2005), 182–187.
[3] Claudia Archetti, Martin W. P. Savelsbergh, and M. Grazia Speranza. 2006. Worst-Case Analysis for Split Delivery Vehicle Routing Problems. *Transportation Science* 40, 2 (May 2006), 226–234. https://doi.org/10.1287/trsc.1050.0117
[4] Claudia Archetti and Maria Grazia Speranza. 2008. The split delivery vehicle routing problem: A survey. *The vehicle routing problem: Latest advances and new challenges* (2008), 103–122.
[5] George Dantzig, Ray Fulkerson, and Selmer Johnson. 1954. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America* 2, 4 (1954), 393–410.
[6] Moshe Dror and Pierre Trudeau. 1989. Savings by Split Delivery Routing. *Transportation Science* 23, 2 (May 1989), 141–145. https://doi.org/10.1287/trsc.23.2.141
[7] Moshe Dror and Pierre Trudeau. 1990. Split delivery routing. *Naval Research Logistics (NRL)* 37, 3 (1990), 383–402.
[8] Weikang Fang, Zailin Guan, Peiyue Su, Dan Luo, Linshan Ding, and Lei Yue. 2022. Multi-Objective Material Logistics Planning with Discrete Split Deliveries Using a Hybrid NSGA-II Algorithm. *Mathematics* 10, 16 (2022), 2871.
[9] Luis Gouveia, Markus Leitner, and Mario Ruthmair. 2023. Multi-Depot Routing with Split Deliveries: Models and a Branch-and-Cut Algorithm. *Transportation Science* 57, 2 (Mar 2023), 512–530. https://doi.org/10.1287/trsc.2022.1179
[10] Damon Gulczynski, Bruce Golden, and Edward Wasil. 2010. The split delivery vehicle routing problem with minimum delivery amounts. *Transportation Research Part E: Logistics and Transportation Review* 46, 5 (Sep 2010), 612–626. https://doi.org/10.1016/j.tre.2009.12.007
[11] Damon Gulczynski, Bruce Golden, and Edward Wasil. 2011. The multi-depot split delivery vehicle routing problem: An integer programming-based heuristic, new test problems, and computational results. *Computers & Industrial Engineering* 61, 3 (2011), 794–804.
[12] Gilbert Laporte. 1984. Optimal solutions to capacitated multidepot vehicle routing problems. *Congressus Nemerantium* 4 (1984), 283–292.
[13] Clair E Miller, Albert W Tucker, and Richard A Zemlin. 1960. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)* 7, 4 (1960), 326–329.
[14] Yoshitaka Nakao and Hiroshi Nagamochi. 2007. A DP-based Heuristic Algorithm for the Discrete Split Delivery Vehicle Routing Problem. *Journal of Advanced Mechanical Design, Systems, and Manufacturing* 1, 2 (2007), 217–226. https://doi.org/10.1299/jamdsm.1.217
[15] Artur Pessoa, Ruslan Sadykov, Eduardo Uchoa, and François Vanderbeck. 2020. A generic exact solver for vehicle routing and related problems. *Mathematical Programming* 183 (2020), 483–523.
[16] Andréa Cynthia Santos. 2016. Shared transportation for macro-distribution in post-disaster relief. In *Proceedings of the 9th Triennial Symposium on Transportation Analysis (TRISTAN 2016).*

Integer Linear Programming for energy-efficient scheduling with time-dependent consumption functions

Mirko Mucciarini[1], Giulia Caselli[1], Daniele De Santis[3], Manuel Iori[3], and Juan José Miranda Bront[2]

[1]Department of Economics "Marco Biagi", University of Modena and Reggio Emilia, Modena, Italy, ✉ mirko.mucciarini@unimore.it, giulia.caselli@unimore.it
[2]School of Business, Universidad Torcuato Di Tella, Buenos Aires, Argentina, ✉ jmiranda@utdt.edu
[3]Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy, ✉ manuel.iori@unimore.it, 240265@studenti.unimore.it

1 Introduction

The field of energy-efficient scheduling has recently witnessed considerable attention and research contributions (see, e.g., [1], [3]). In this paper, we consider a real-world energy-efficient machine scheduling problem with time-dependent resource consumption functions which, to the best of our knowledge, has never been considered previously in the related literature. The problem is inspired by an Italian company that conducts performance tests on electric components for the automotive industry. Briefly, a collection of tests (named *jobs* hereafter) must be scheduled on a set of heterogeneous machines within a given planning horizon, discretized into time slots. Each job has a known processing time, expressed in time slots, in which energy is consumed. The energy consumption is not assumed to be constant during the execution of the job, but rather time-dependent and variable within its execution. In addition to this time dependency, this consumption is also assumed to be machine-dependent. The energy required to execute the schedule can be obtained from two different sources: (i) from photovoltaic panels available, or (ii) buying energy from the market. If convenient, the company can also sell energy to the market. The amount of energy captured through the panels as well as the energy prices are also considered time-dependent, modeled as a step function over the planning horizon. The objective is to allocate a set of jobs to a set of machines within the time horizon in such a way that the total energy cost is minimized. We show the effectiveness of our model on a set of random instances.

2 Problem definition and mathematical model

Let J be the set of jobs, I the set of machines, and $T = 0, ..., t_{max}$ the set of time slots. Every job $j \in J$ has a processing time p_j and an energy consumption function $u_{ji\tau}$ that depends on machine $i \in I$ and time slot $\tau = 0, ..., p_j$. For $t \in T$, let e_t denote the amount of energy obtained from the photovoltaic panels, c_t the buying cost, and s_t the selling price, per unit. We assume that the total net amount of energy that the facility can process in every time slot is limited by E, this value depends on the average consumption of the machines. For $j \in J$ and $t \in T$, let $T_{jt} = \{\max\{0, t - p_j + 1\}, ..., t\}$ be the set of feasible starting times for job j such that its execution spans until t, independently of the machine assigned. We define binary variables X_{jit} taking value one iff job j is assigned to machine i starting in time slot t. Integer variables U_t and V_t indicate the quantity of energy bought and sold in every time slot $t \in T$. Finally, integer variables W_{jt} define the quantity of energy consumed by job j in time slot t.

The ILP model reads:

$$\min \quad \sum_{t \in T} (c_t U_t - s_t V_t) \tag{1}$$

$$\text{s.t.} \quad \sum_{i \in I} \sum_{t=0}^{t_{\max} - p_j} X_{jit} = 1 \qquad j \in J \tag{2}$$

$$\sum_{j\in J}\sum_{\tau\in T_{jt}} X_{ji\tau} \le 1 \qquad i\in I, t\in T \tag{3}$$

$$W_{jt} = \sum_{i\in I}\sum_{\tau\in T_{jt}} u_{ji(t-\tau)} X_{ji\tau} \qquad j\in J, t\in T \tag{4}$$

$$\sum_{j\in J} W_{jt} = e_t + U_t - V_t \qquad t\in T \tag{5}$$

$$e_t + U_t - V_t \le E \qquad t\in T \tag{6}$$

$$X_{jit} = 0 \qquad j\in J, i\in I, t\in T\setminus\{0,\dots,t_{\max}-p_j\} \tag{7}$$

$$X_{jit} \in \{0,1\} \qquad j\in J, i\in I, t\in T \tag{8}$$

$$W_{jt}, U_t, V_t \in \mathbb{Z}^+ \qquad j\in J, t\in T \tag{9}$$

The objective function (1) minimizes the total energy cost (i.e., the difference between the total cost of buying and selling energy). Constraints (2) impose that each job is scheduled exactly once. Constraints (3) prevent processing more than one job in the same time slot on the same machine. Constraints (4) define the energy consumption variables W_{jt} based on the time-dependent consumption function of each job. Constraints (5) match the total energy consumed by the schedule with the net amount of energy processed in each time slot $t \in T$. Constraints (6) impose the upper limit of energy in the facility in every time slot. Constraints (7) guarantee that every job is scheduled in such a way that it will be completed by the last time slot $t = t_{max}$, which is considered as the overall scheduling deadline. Finally, constraints (8)-(9) define the variables domain.

3 Preliminary computational results and future research

We solve the proposed ILP model on a dataset of 30 randomly generated instances, and we compare its performance with a Constraint Programming (CP) model. Both models are coded in Python and the experiments are executed on a Windows 10 system equipped with an Intel Xeon Gold 6252N 2.30GHz with 32 GB of RAM. We use Gurobi 9.5.1 for solving ILP and CPLEX 22.1.1.0 for CP. A time limit of 3600 seconds is imposed for each execution of the models. In Table 1, we report the average results on 3 groups of 10 instances of larger size. The main message from the table is that the ILP is able to find optimal solutions for some instances up to 30 jobs, while CP struggles even for small values of $|J|$. Future research aims at tackling larger instances. Noting that the ILP degrades its performance when $|J| \ge 30$, we are working on a promising matheuristic algorithm inspired by the machine-assignment fixing strategy proposed by [2].

Table 1: Comparison of average results on ILP and CP models

Parameters		ILP		CP		
Size	#inst	#opt	time[s](*)	#opt	time[s](**)	%gap(*)
$\|J\| \le 10$	10	10	0.78	7	1087.22	1.40%
$15 \le \|J\| \le 20$	10	9	407.28	0	T.L.	30.17%
$\|J\| \ge 30$	10	7	616.95	0	T.L.	28.06%

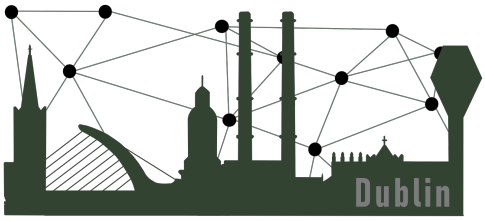
* computed only on instances solved to optimality by ILP
** counting 3600s for instances optimal for ILP and non optimal for CP

References

[1] D. Catanzaro, R. Pesenti, and R. Ronco. Job scheduling under time-of-use energy tariffs for sustainable manufacturing: a survey. *European Journal of Operational Research*, 308(3):1091–1109, 2023.

[2] L. Fanjul-Peyro, F. Perea, and R. Ruiz. Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources. *European Journal of Operational Research*, 260(2):482–493, 2017.

[3] C. Gahm, F. Denz, M. Dirr, and A. Tuma. Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, 248(3):744–757, 2016.



INOC - 2024

Session Session 6B: Flow Applications

Wednesday 13 March 2024, 14:00-15:00

Q012

Multi-Objective Multi-Commodity Flow Optimization for Wartime Planning with Cyber-Effects

Full Paper

Alex Hoffendahl
United States Air Force
alex.hoffendahl.1@us.af.mil

Chancellor Johnstone
Air Force Institute of Technology
chancellor.johnstone.1@us.af.mil

Alex Stephens
United States Air Force
alexander.stephens.7@us.af.mil

Richard Dill
Air Force Institute of Technology
richard.dill@us.af.mil

Lance Champagne
Air Force Institute of Technology
lance.champagne.1@us.af.mil

ABSTRACT

In this work, we enhance the analytical capabilities and overall usability of cyber-wargames by providing a quantitative approach for generating optimal cyber-effect courses of action in conjunction with other kinetic courses of action. Specifically, we introduce the Cyber-Wargame Commodity Course of Action Automated Analysis Method, which balances risk with cost. We utilize a multi-commodity flow (MCF) formulation within a multi-objective mixed-integer program (MO-MIP) to determine optimal courses of action in a wargame scenario. We also assess the robustness of our optimal course of action through sensitivity analysis.

1 INTRODUCTION

The focus of this research is to explore the implementation of cyber-effects in wargaming. To motivate this discussion, we recall a real-world example concerning Iran. Since 2009, Iran has executed a continuous stream of cyber attacks targeting the United States (US) government and private sector systems, costing western firms millions of dollars in lost business and creating a substantial financial burden to local residents. Beginning in 2020, conflicts between the US and Iran have consistently taken place in cyberspace. Although the breadth remains unclear, cyberspace has become a primary battleground, providing an alternative to kinetic military action [12]. The prevalence of cyber capabilities has increased the technical complexity of modern warfare; today's warfare is more technologically advanced than ever and those using cyber capabilities gain an operational advantage [11].

In previous research, [8] introduced the Wargame Commodity Course of Action Automated Analysis Method (WCCAAM) as a systematic procedure to aid in the course of action (COA) development, analysis and comparison phases of the military decision-making process (MDMP). Assuming enemy behavior is known, along with high-reliability on information sources, WCCAAM generates an optimal COA, minimizing engagement risk subject to successfully achieving various objectives, e.g., nullifying enemy targets. We introduce an extended version of WCCAAM, named Cyber-WCCAAM (C-WCCAAM), which delivers an optimal friendly COA considering two objectives:

(1) minimize engagement risk
(2) minimize cyber-effect cost

To consider decisions related to the employment of cyber-effects within a wargame, we utilize a mixed-integer program (MIP). C-WCCAAM encodes cyber-effects as binary decision variables; a one represents the use of a particular cyber-effect and a zero represents anything otherwise. Thus, C-WCCAAM enables decisions related to the employment of friendly forces and cyber-decisions, simultaneously. We note that while our application utilizes a multi-objective formulation with two objectives, i.e., a bi-objective formulation, there is no reason one could not include additional objectives. The inclusion of additional objectives would be left to the decision-maker.

We compare results generated with WCCAAM to those generated with C-WCCAAM on a fictitious, yet plausible, scenario, adding friendly cyber-effects into the decision space. With C-WCCAAM, we provide a trade-off between engagement risk and cost for a cyber-effect. Ultimately, we extend WCCAAM to incorporate multiple cyber-effects, formulating a new modeling approach that advances the state-of-the-art in cyber wargaming and COA development.

Section 2 of this paper provides relevant background used in the research. Section 3 provides a detailed methodology specific to constructing C-WCCAAM. Section 4 is dedicated to analysis results and discussion. Section 5 concludes the paper.

2 BACKGROUND

This section provides the necessary background and foundational concepts for C-WCCAAM, to include a brief discussion of cyber-wargaming and WCCAAM.

2.1 Wargaming in the Cyber Realm

Historically, the US military has relied on wargaming to achieve both short-term and long-term objectives, examples of which include naval warfare during World War II [25], the US response to the Iraqi invasion of Kuwait during the Gulf War [4] and counterinsurgency tactics during the Vietnam War [24]. Wargaming has a long history as a tool for decision-makers to improve their critical thinking and inventiveness [20]. Wargaming is also a crucial step

in the military decision-making process [10], which enables the construction and selection of effective COAs to achieve strategic, operational and tactical goals [8].

Today's warfare is marked by technological advances in information, communication, and artificial intelligence [5]. Thus, there is a growing demand for wargames that effectively incorporate modern elements, i.e., specific effects, that can directly impact one's operational advantage. One such set of effects includes *cyber-effects*. While cyber-effects might be "invisible," they are as vital to victory in armed conflict as kinetic effects [17]. A cyber-effect denotes an attempt to breach the information systems of another person or organization to gain some advantage by executing unauthorized activities to disrupt, manipulate, or destroy opposition electronic systems, networks or data [1]. These effects come in many forms, e.g., phishing scams, ransomware or denial-of-service attacks, and can cause significant financial damage, as well as damage to reputation [18].

The complexity of cyber-wargames cannot be overstated. While the behavior of the air, sea, land, and space assets is well-known in most games, cyber-effects themselves are often abstract or misunderstood [15]. As a result, many decision-makers undermine rules to implement cyber-effects in a wargame [23]. For instance, players may use a cyber-effect at any time during a cyber-wargaming session, regardless of practical implementation. As a result, the gap between the actual operations and simulated games often diminishes the results generated by cyber-wargaming.

Examples of effective implementation of cyber-effects in wargaming include [14] and [2].

2.2 Wargaming Commodity Course of Action Automated Analysis Method

To reduce the time required to develop, analyze and compare COAs, [8] developed the Wargaming Commodity Course of Action Automated Analysis Method (WCCAAM). As our foundational model, WCCAAM sets the groundwork for the extensions included in this paper. In WCCAAM, a collection of different friendly units, identified as *commodities*, are utilized to confront enemy COAs. Various commodities are dispatched from multiple locations to nullify enemy targets, as set out by tactical and strategic objectives, while minimizing engagement risk to friendly forces.

WCCAAM relies on a multi-commodity flow algorithm (MCFA) to process the directed network, made up of nodes, i.e., bases and targets, and engagement paths; this network is derived from the mission analysis phase of the MDMP. The MCFA outputs the optimal flow of each commodity along each engagement path in the network. This output translates to an optimal COA for a commander to allocate resources to accomplish different objectives while minimizing operational risk. These objectives can be tactical, operational or strategic in nature, e.g., achieve air superiority or eliminate all enemy armor assets.

Formally, the mathematical formulation utilized by WCCAAM is

$$\min_{x} \sum_{(i,j)} \sum_{t} R_{tij} x_{tij} \tag{1a}$$

$$\text{subject to: } \sum_{j} x_{tij} \leq S_{it} \qquad \forall\, i \in N \tag{1b}$$

$$\sum_{i} x_{tij} \geq D_{jt} \qquad \forall\, j \in N \tag{1c}$$

$$x_{tij} \geq 0 \qquad \forall\, t \in T, i, j \in N \tag{1d}$$

where R_{tij} is the engagement risk associated with commodity t on engagement path (i, j), x_{ijt} is a decision variable related to the number of commodity t sent along path (i, j), S_{it} is the *supply* of commodity t available at node i, and D_{jt} is the *demand* for commodity t at node j. Demand, in this case, refers to enemy threats that must be nullified or objectives that require certain friendly assets to be achieved.

3 METHODOLOGY

In this section, we introduce C-WCCAAM, which includes decisions related to the implementation of cyber-effects. We first introduce relevant notation, then extend the formulation previously introduced in Section 2 to include cyber-effect decisions.

3.1 Notation

Relevant model decision variables and parameters are shown below.

Sets:

- T: set of friendly commodities with index $t \in T$
- N: set of nodes
- K: set $\{0, 1\}$ denoting non-use (0) and use (1) of cyber-effect.
- E: set of engagement paths (edges) from node i to node j with index set $(i, j) \in E$ where $i, j \in N$

Parameters:

- R_{tij}: engagement risk of commodity t sent from node i to satisfy demand at node j
- S_{ti}: number of commodity t that can be sent from node i
- D_{tj}: demand for commodity t at node j
- P : cyber budget
- C_{tij}: cost of using cyber-effects for commodity t when sent along engagement path (i, j)
- ϵ_{tij}: engagement risk-reduction factor for commodity t along engagement path (i, j); value between 0 to 1

Decision Variables:

- x_{tij0}: number of commodity t from friendly base i sent to nullify target j without the use of cyber-effects
- x_{tij1}: number of commodity t from friendly base i sent to nullify target j with the use of cyber-effects
- y_{tij}: equal to 1 when cyber-effect activated for commodity t when engaging target j from base i, 0 otherwise

Engagement risks are used to determine effective engagement paths for moving commodities to nullify targets. The higher R_{tij} for a given commodity-engagement path pair, the higher the potential operational risk.

The risk-reduction factor, ϵ_{tij} for commodity t on engagement path (i, j), defines the percentage of engagement-risk that can

be eliminated through the use of a specific cyber-effect. The risk-reduction-factor, for example, can be used to measure the disruption, e.g., reduction in accuracy, of adversarial surface-to-air missile (SAM) systems, as a result of some offensive cyber action. The greater the risk-reduction factor, the greater the disruption to these systems.

3.2 Cyber-WCCAAM Formulation

To consider cyber decisions when determining the optimal friendly COA, we introduce a binary decision variable y_{tij} to the original WCCAAM formulation in (1). The activation of y_{tij} comes with a reduction in engagement risk, ϵ_{tij}, along engagement path (i, j) for commodity t. We note that domains of indices are defined in Section 3.1.

In essence, we augment the original WCCAAM formulation in (1) with a slightly different objective function

$$\underbrace{\sum_t \sum_{(i,j)} R_{tij} x_{tij} (1 - \epsilon_{tij} y_{tij})}_{f_1} + \underbrace{\sum_t \sum_{(i,j)} C_{tij} y_{tij}}_{f_2}, \tag{2}$$

including an engagement risk term, f_1, and a cyber-effect cost term, f_2, which we aim to minimize, turning our problem into a multi-objective one. To reduce the problem back to a single objective, we can forgo the inclusion of f_2 in the objective function and instead add an additional cyber-budget constraint

$$\sum_t \sum_{(i,j)} C_{tij} y_{tij} \leq P, \tag{3}$$

thus adding a knapsack problem [19] to WCCAAM, with the objective to minimize engagement risk without exceeding a *cyber budget*, denoted as P. We note that in the context of multi-objective optimization, the addition of constraint (3) is called the ϵ-constrained approach [16]. This approach allows for the translation of a multi-objective function to a single-objective function. The ϵ-constrained approach is often used to determine Pareto-optimal solutions; this is in contrast to, say, a weighted multi-objective function.

Unfortunately, the introduction of cyber-effect decision variables into the objective function shown in (2) transforms the linear WCCAAM formulation into a nonlinear one, which can be time-consuming and impractical to solve within a high-dimension problem [9]. Thus, we linearize the formulation by constructing cyber decisions as alternating engagement paths, one associated with cyber reinforcement, engagement path $(i, j, 1)$, and one without reinforcement, engagement path $(i, j, 0)$. The decision variables x_{tij0} and x_{tij1} are then constrained to ensure that all of commodity t are sent across engagement path $(i, j, 1)$ if y_{tij} is equal to 1; otherwise, all of commodity t utilizing path (i, j) must be sent across $(i, j, 0)$.

With the inclusion of f_1 and (3), as well as the subsequent changes to make the formulation linear, the formulation for C-WCCAMM is

$$\min_{x,y} \sum_t \sum_{(i,j)} \sum_k R_{tijk} x_{tijk} \tag{4a}$$

$$\text{subject to: } \sum_j (x_{tij0} + x_{tij1}) \leq S_{ti} \quad \forall\, t \in T, i \in N \tag{4b}$$

$$\sum_i (x_{tij0} + x_{tij1}) \geq D_{tj} \quad \forall\, t \in T, j \in N \tag{4c}$$

$$\sum_t \sum_{(i,j)} C_{tij} y_{tij} \leq P \tag{4d}$$

$$x_{tij0} \leq M(1 - y_{tij}) \quad \forall\, t \in T, (i,j) \in E \tag{4e}$$

$$x_{tij1} \leq M y_{tij} \quad \forall\, t \in T, (i,j) \in E \tag{4f}$$

$$x_{tijk} \geq 0 \quad \forall\, t \in T, (i,j) \in E, k \in K \tag{4g}$$

$$y_{tij} \in \{0, 1\} \quad \forall\, t \in T, (i,j) \in E \tag{4h}$$

where R_{tij0} is the engagement risk without the use of cyber-effects across engagement path (i, j) and $R_{tij1} = R_{tij0}(1 - \epsilon_{tij})$ is the engagement risk with the use of cyber-effects across engagement path (i, j), all for commodity t.

Additionally, we constrain the total cost of cyber-effects with some cyber budget P using constraint (4d). An alternate constraint might instead constrain the number of cyber-effects used. For a simplified decision space, we can utilize the constraint

$$y_{tij} = y_{t'ij} \;\forall\, t, t' \in T, (i,j) \in E, \tag{5}$$

which ensures that any cyber-effect activated along path (i, j) is activated for all commodities. This constraint can be added for cyber-effects shared across commodities. We explore the addition of constraint (5) in later sections. Constraints (4e) and (4f) enforce the cyber-path restrictions, with M defined as a value large enough to prevent breaking said constraints.

3.3 Assumptions

In a real-world scenario, the success of a cyber-effect is often random [7]. The probability of a successful cyber-effect may be dependent on multiple factors, e.g., enemy cyber defenses. In many cases there is also a *probability of detection*, whereby to achieve a particular cyber-effect, a cyber-attack must also go undetected by enemy forces. For our purposes, we ignore this potential unpredictability. Examples of cyber-games utilizing these probabilistic approaches include [6] and [21], among others.

[22] extends WCCAAM itself to deal with uncertainties related to enemy force size. Weaknesses of disregarding the probabilistic aspect of cyber-effects in wargames in the context of denial and deception are described in [13]. We also require precise and reliable information related to engagement risk, enemy force structure and enemy action, as well as risk-reduction factors for cyber-effects. More specifically, we require that these model inputs be *known*.

Table 1: Cyber-Effect Risk-Reduction Factors

		Mountain AB	Plains AB	Capital AB
Armor	Striker AB	0.60	0.33	0.40
	Camp Kipling	0.30	0.70	0.40
	Pendem IAP	0.40	0.20	0.40
Fighters	Striker AB	0.30	0.50	0.40
	Camp Kipling	0.30	1.00	0.40
	Pendem IAP	0.00	0.50	0.40
Infantry	Striker AB	0.60	0.20	0.83
	Camp Kipling	0.50	0.40	0.50
	Pendem IAP	0.67	0.20	0.20

4 APPLICATION TO OPERATIONAL SCENARIO

In this section we introduce a modified operational scenario. We then compare optimal COAs generated with WCCAAM and C-WCCAAM. To further explore these optimal COAs, we also provide sensitivity analysis for the C-WCCAAM results.

4.1 Scenario

For this work, we adjust a scenario previously used in [8] and introduced in [3]. The scenario concerns two civilizations, Phoenicia and Sumer, fighting against each other in a multi-domain conflict. In this section, we use the terms *friendly* and *enemy* interchangeably with Phoenicia and Sumer, respectively. We adjust the original scenario to include additional fighter, armor, and infantry units for Phoenicia and Sumer.

While (4) is a general formulation, we simplify the scenario at hand to include supply nodes and demand nodes. Phoenician bases (Striker Air Base (AB)), Camp Kipling and Pendem International Airpot (IAP)) have a fixed supply of various commodities, while Sumerian targets require a certain number of dedicated Phoenician commodities to be nullified. This scenario aims to allocate and assign friendly commodities to eliminate all enemy targets while optimally employing cyber-effects to minimize overall engagement risk.

For initial exploration of the scenario of interest, we utilize the pseudo-data shown Table 1. Cyber-effect costs are \$2K, \$3K and \$9K for armor, fighters and infantry, respectively. We include force structure in Table 2 and Table 3 for Phoenician and Sumerian forces, respectively.

Computational experiments were implemented using Python within the base version of Google Colabratory. We utilized the default CBC solver included in PuLP version 2.7.0.

Table 2: Phoenician Forces

Blue	Striker AB	Camp Kipling	Pendem IAP
Armor	5	20	0
Fighters	4	2	0
Infantry	280	20	150

Table 3: Sumerian Forces

Red	Mountain AB	Plains AB	Capital AB
Armor	10	15	0
Fighters	2	1	3
Infantry	100	50	300

4.2 Results

With the parameters stated earlier, we generate two COAs: one constructed with WCCAAM and the other with C-WCCAAM using a cyber budget of \$50K. These COAs are shown in Table 4.

Table 4: WCCAAM and C-WCCAAM Optimal COAs

WCCAAM	C-WCCAAM
Optimal Flow for Armor:	Optimal Flow for Armor:
Striker AB → Mountain AB: 5	Striker AB → Mountain AB: 5*
Camp Kipling → Mountain AB: 5	Camp Kipling → Mountain AB: 5*
Camp Kipling → Plains AB: 15	Camp Kipling → Plains AB: 15*
Optimal Flow for Fighters:	Optimal Flow for Fighters:
Striker AB → Plains AB: 1	Striker AB → Plains AB: 1
Striker AB → Capital AB: 3	Striker AB → Capital AB: 3*
Camp Kipling → Mountain AB: 2	Camp Kipling → Mountain AB: 2*
Optimal Flow for Infantry:	Optimal Flow for Infantry:
Striker AB → Mountain AB: 100	Striker AB → Mountain AB: 100*
Striker AB → Plains AB: 50	Striker AB → Plains AB: 30*
Striker AB → Capital AB: 130	Striker AB → Capital AB: 150*
Camp Kipling → Capital AB: 20	Camp Kipling → Plains AB: 20
Pendem IAP → Capital AB: 150	Pendem IAP → Capital AB: 150*
	*denotes use of cyber-effect
	Cyber Budget: \$50K
Total Engagement Risk: 1125	Total Engagement Risk: 777

The optimal solution with C-WCCAAM decreased engagement risk from 1,125 (using WCCAAM) to 777. While this decrease can be attributed to cyber-effect decisions, it is interesting to explore where decisions in optimal commodity flows differ from WCCAAM, as opposed to decreases in engagement risk due to the risk-reduction factors associated with cyber-effects alone. We can see slight differences between the two COAs, specifically in the deployment of infantry to counteract enemies.

With WCCAAM, the optimal COA satisfies the demand required by the Plains AB Infantry forces completely through infantry supplied by Striker AB, while C-WCCAAM satisfies this demand through the use of infantry at Striker AB and Camp Kipling. WCCAAM also uses infantry at all three bases to satisfy Capital AB Infantry demand, while C-WCCAAM consolidates by using forces from Striker AB and Pendem IAP. The optimal COA for C-WCCAAM selects cyber-effects for forces moving from Striker AB to Capital AB Infantry, resulting in additional infantry sent along this engagement path compared to the optimal WCCAAM COA. Thus, fewer infantry are available to be sent to Capital AB infantry from Striker AB, requiring Camp Kipling to send infantry.

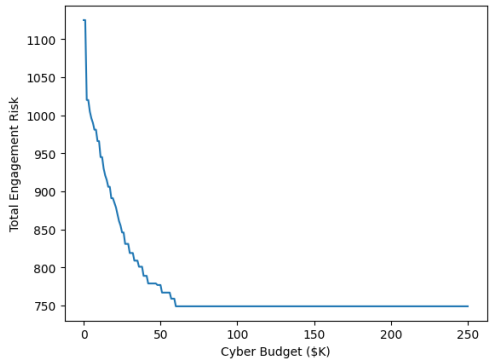


Figure 1: Total engagement risk with respect to cyber budget.

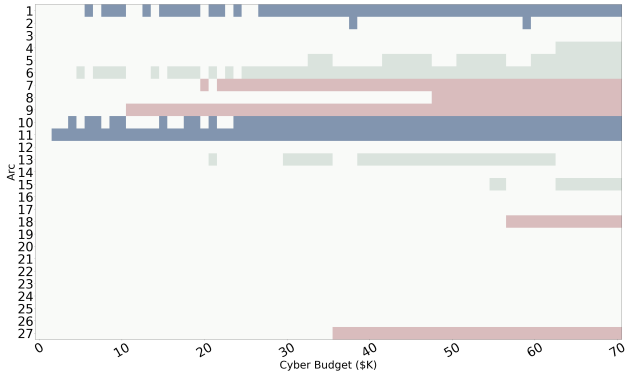


Figure 2: Location for cyber-effects with cyber budget of $1K-$70K for armor (blue), fighters (gray), and infantry (red).

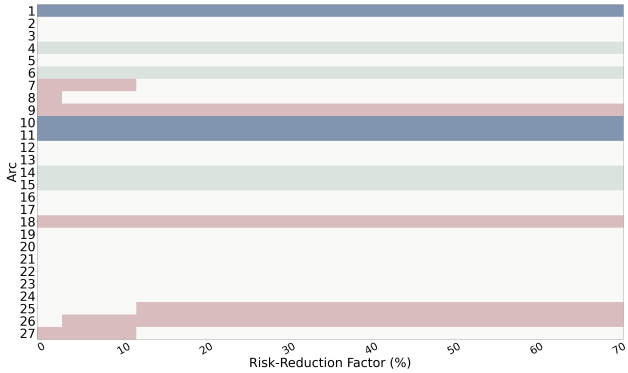


Figure 3: Location of cyber-effects vs. risk-reduction factor on paths 25, 26 and 27 simultaneously.

4.3 Sensitivity Analysis

In this section, we utilize sensitivity analysis to assess the robustness of the C-WCCAAM COA shown in Table 4. Specifically, we perturb cyber budget, cyber-effect costs and engagement risk. Labels for engagement paths are shown in Table 5. We note that these path labels are the same for all friendly commodities.

Table 5: Scenario Engagement Path Labels

Path	Armor			Fighters			Infantry		
	Mountain	Plains	Capital	Mountain	Plains	Capital	Mountain	Plains	Capital
Striker AB	1	2	3	4	5	6	7	8	9
Camp Kipling	10	11	12	13	14	15	16	17	18
Pendem IAP	19	20	21	22	23	24	25	26	27

We first explore trade-offs between total COA engagement risk and cyber budget by varying the cyber budget P, the results of which are shown in Figure 1. Under a cyber budget constraint formulation, C-WCCAAM achieves greater cyber-effect selection stability at a lower cost of approximately $65K.

Figure 1 shows the relationship between the overall engagement risk and cyber budget as P increases. The cyber budget intervals of constant engagement risk indicate regions that do not change the optimal objective function value; in rare cases, the same objective function value can result from different optimal solutions.

4.3.1 Location of Cyber-Effects vs. Cyber Budget. Cyber-effect locations may shift in response to a change in cost of executing a cyber-attack in another location. Figure 2 shows on which paths cyber-effects are used as the cyber budget increases. We note that, based on Figure 2, changes to the cyber budget greatly affect the overall optimal solution, and, unlike the reduction of engagement risk on a single arc, does not result in predictable changes to said solution.

4.3.2 Location of Cyber-Effects vs. Cyber Risk-Reduction Factor. Figure 3 illustrates the changes in cyber-effect deployment location when there is a simultaneous adjustment in the engagement risk-reduction factor for infantry moving from Pendam IAP to engage infantry stationed at Mountain AB, Plain AB and Capital AB. With risk-reduction ranging from 0 to 16%, preference shifts towards applying cyber-effects at path 7, where infantry from Striker AB engage those at Mountain AB, and at path 27, which supports the infantry moving from Pendam IAP to confront forces at Capital AB, in conjunction with infantry from Striker AFB engaging the Capital AB infantry. Additionally, with a risk-reduction from 0% to 2%, there is an initiation of cyber-effects at path 8, deploying infantry from Striker AB to face off against those at Plains AB, rather than deploying from Pendam IAP. However, once the risk-reduction reaches 16% or higher, cyber-effects at paths 7, 8, and 27 are ceased, and reliance is placed solely on cyber-effects at paths 25 and 26, which involve deploying the majority of infantry from Pendam IAP against forces at Plain AB and Capital AB.

4.3.3 Results Using Constraint (5). In this section, we explore results when we enforce shared cyber-effects using constraint (5). We note that the use of constraint (5) allows for cyber-effects to be utilized on paths where no commodities are sent. However, the inclusion or exclusion of this constraint can vary based on my factors, e.g., decision-maker opinion, operational relevance.

Figure 4 shows the decrease in overall engagement risk as cyber budget increases; we reach a minimum engagement risk of 749 at a cyber budget of $179. Cyber-effects are not utilized until the cyber budget reaches $5K. This is due to the cost associated with enabling cyber-effects for each of the three commodities to satisfy constraint (5); we see the largest decrease in engagement risk as the cyber

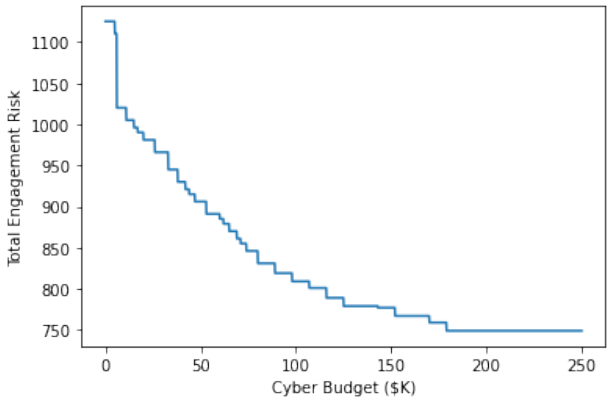


Figure 4: Total engagement risk with respect to cyber budget when utilizing constraint (5)**.**

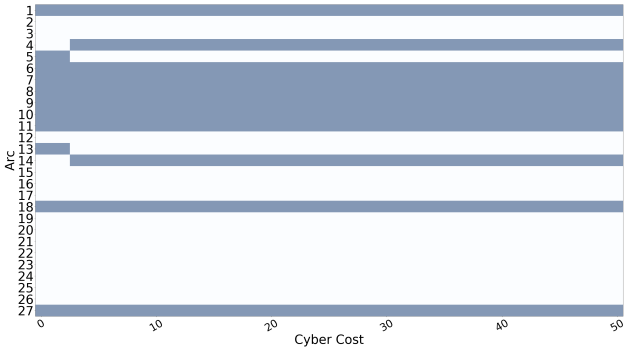


Figure 5: Location of cyber-effects vs. cyber-effect cost on path 13 when utilizing constraint (5)**.**

budget increases at $6K. A cyber-effect for Camp Kipling Armor to Plains AB Armor was utilized resulting in a 0.7 engagement risk-reduction factor along that engagement path.

Figure 5 shows how the costs of cyber-effects on blue fighters from Camp Kipling can affect where to implement other effects. When the cyber cost on path 13 is $3K or less, the cyber-effects are launched at paths 5 and 13. When the cost exceeds $3K, the cyber-effects utilized previously at paths 5 and 13 switch to paths 4 and 14.

5 CONCLUSION

With the inclusion of cyber-effects, C-WCCAAM generates a comprehensive, quantitative approach for wargaming scenarios to facilitate effective cyber-effect decision-making. Moreover, C-WCCAAM provides a course of action for implementing cyber capabilities into military operations, ensuring that the use of cyber assets is optimized and aligned with other mission objectives.

Future work could weaken assumptions related to the certainties associated with our model parameters, e.g., engagement risk and cyber-effectiveness. Additionally, in our work, we assume enemy action is known. In future work, we might instead consider a small set of enemy COAs, each with a specific probability of occurring. Additionally, we can use optimization techniques to react to possible enemy actions by using C-WCCAAM in a real-world wargame, making a C-WCCAAM application through open-source methods.

DISCLAIMER

The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, United States Department of Defense, or United States Government.

REFERENCES

[1] 2022. What is A Cyberattack? https://www.cisco.com/c/en/us/products/security/common-cyberattacks.html

[2] Kimo Bumanglag, David Law, Adam Welle, and Peter Barrett. 2019. Constructing large scale cyber wargames. In *International conference on cyber warfare and security*. Academic Conferences International Limited, 653–X.

[3] Matthew B Caffrey. 2019. *On wargaming: How wargames have shaped history and how they may shape the future.* Vol. 43. Naval War College Press.

[4] Matthew Caffrey Jr. 2000. Toward a history-based doctrine for wargaming. *Air & Space Power Journal* 14, 3 (2000), 33.

[5] Gary Chapman. 2003. An introduction to the revolution in military affairs. In *XV Amaldi Conference on Problems in Global Security*. Citeseer, 1–21.

[6] Edward J Colbert, Alexander Kott, Lawrence III Knachel, and Daniel T Sullivan. 2017. *Modeling Cyber Physical War Gaming.* Technical Report. US Army Research Laboratory Aberdeen Proving Ground United States.

[7] John Curry and Nick Drage. 2018. Developments in state level cyber wargaming. In *Proceedings of the 11th International Conference on Security of Information and Networks*. 1–6.

[8] William T DeBerry, Richard Dill, Kenneth Hopkinson, Douglas D Hodson, and Michael Grimaila. 2021. The wargame commodity course of action automated analysis method. *The Journal of Defense Modeling and Simulation* (2021).

[9] F Delbos, T Feng, J Ch Gilbert, and D Sinoquet. 2008. Nonlinear optimization for reservoir characterization. In *ENGOPT International conference on engineering optimization, Rio de Janeiro, Brazil.*

[10] Department of the Army. 2019. *ADP 5-0 The OPERATIONS PROCESS.*

[11] David B Fox, Catherine D McCollum, Eric I Arnoth, and Darrell J Mak. 2018. *Cyber wargaming: Framework for enhancing cyber wargaming with realistic business context.* Technical Report. Mitre Corporation Homeland Security Systems Engineering and Developme Institute.

[12] Andrew Hanna. 2019. The Invisible US-Iran Cyber War. *The Iran Primer* (2019).

[13] Kristin E Heckman and Frank J Stech. 2015. Cyber counterdeception: How to detect denial & deception (D&D). In *Cyber Warfare*. Springer, 103–140.

[14] Kristin E Heckman, Michael J Walsh, Frank J Stech, Todd A O'boyle, Stephen R DiCato, and Audra F Herber. 2013. Active cyber defense with denial and deception: A cyber-wargame experiment. *computers & security* 37 (2013), 72–77.

[15] Nina Kollars. 2021. Pathologies of Obfuscation.

[16] Kaisa Miettinen. 2012. *Nonlinear multiobjective optimization.* Vol. 12. Springer Science & Business Media.

[17] Vikram Mittal and Andrew Davidson. 2020. Combining wargaming with modeling and simulation to project future military technology requirements. *IEEE Transactions on Engineering Management* 68, 4 (2020), 1195–1207.

[18] Seattle Office of Emergency Management. 2023. Cyber Attack and Disruption. https://seattle.gov/documents/departments/emergency/plansoem/shiva/shivav7.0-cyber.pdf

[19] Harvey M Salkin and Cornelis A De Kluyver. 1975. The knapsack problem: a survey. *Naval Research Logistics Quarterly* 22, 1 (1975), 127–144.

[20] Jeremy Sepinsky, Eric Heubel, and Matthew Cumpian. 2019. Gaming Cyber in an Operational-Level Wargame: Merlin Cyber Wargame Module Rules for Adjudicators. (2019).

[21] Abderrahmane Sokri. [n.d.]. Cyber Deterrence: A Wargaming Approach. ([n. d.]).

[22] Alexander Stephens, Richard Dill, Chancellor Johnstone, and Douglas D. Hodson. 2023. The Wargame Commodity Course of Action Automated Analysis Method Under Uncertainty.

[23] Bibi van den Berg and Sanneke Kuipers. 2022. Vulnerabilities and cyberspace: A new kind of crises. *Oxford Research Encyclopedia of Politics* (2022). https://doi.org/10.1093/acrefore/9780190228637.013.1604

[24] Natalia Wojtowicz. 2019. From sandboxes to laboratories: evolution of wargaming into a method for experimental studies. *International Journal of Scientific and Research Publications* 9, 12 (2019), 399.

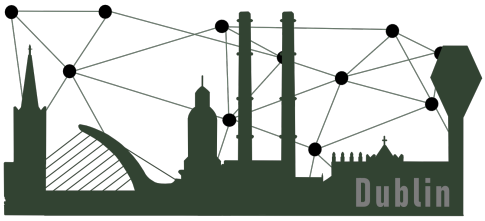
[25] Robert Work and Paul Selva. 2015. Revitalizing wargaming is necessary to be prepared for future wars. *War on the Rocks* 8 (2015).

A combinatorial flow-based formulation for temporal bin packing problems

John Martinovic[1], Nico Strasdat[2], José Valério de Carvalho[3], and Fábio Furini[4]

[1]Institute of Numerical Mathematics, Technische Universität Dresden, 01062 Dresden, Germany, ✉ john.martinovic@tu-dresden.de
[2]Institute of Numerical Mathematics, Technische Universität Dresden, 01062 Dresden, Germany, ✉ nico.strasdat@tu-dresden.de
[3]Departamento de Produção e Sistemas/Centro ALGORITMI, Universidade do Minho, 4710-057 Braga, Portugal, ✉ vc@dps.uminho.pt
[4]Department of Computer, Control and Management Engineering "Antonio Ruberti", Sapienza University of Rome, 00185 Roma, Italy, ✉ fabio.furini@uniroma1.it

We consider two neighboring generalizations of the classical bin packing problem: the temporal bin packing problem (TBPP) and the temporal bin packing problem with fire-ups (TBPP-FU). In both cases, the task is to arrange a set of given jobs, characterized by a resource consumption and an activity window, on ho- mogeneous servers of limited capacity. To keep operational costs but also energy consumption low, TBPP is concerned with minimizing the number of servers in use, whereas TBPP-FU additionally takes into ac- count the switch-on processes required for their operation. Either way, challenging integer optimization problems are obtained, which can differ significantly from each other despite the seemingly only marginal variation of the problems. In the literature, a branch-and-price method enriched with many preprocessing steps (for TBPP) and compact formulations (for TBPP-FU), benefiting from numerous reduction methods, have emerged as, currently, the most promising solution methods. In this paper, we introduce, in a sense, a unified solution framework for both problems (and, in fact, a wide variety of further interval scheduling applications) based on graph theory. Any scientific contributions in this direction failed so far because of the exponential size of the associated networks. The approach we present in this article does not change the theoretical exponentiality itself, but it can make it controllable by clever construction of the resulting graphs. In particular, for the first time all classical benchmark instances (and even larger ones) for the two problems can be solved –in times that significantly improve those of the previous approaches.



INOC - 2024

Session Session 6C: Network Applications
Wednesday 13 March 2024, 14:00-15:00
Q013

Instantaneous and limiting behavior of an n-node blockchain under cyber attacks from multiple hackers

Liang Hong[1] and Xiufeng Xu[2]

[1]Department of Mathematical Sciences, The University of Texas at Dallas, 800 West Campbell Road, Richardson, TX 75080, USA. , ✉ liang.hong@utdallas.edu
[2]Department of Mathematical Sciences, The University of Texas at Dallas, 800 West Campbell Road, Richardson, TX 75080, USA. , ✉ xiufeng.xu@utdallas.edu

We investigate the instantaneous and limiting behavior of an n-node blockchain which is under continuous monitoring of the IT department of a company but faces non-stop cyber attacks from multiple hackers. The blockchain is functional as far as no data stored on it has been changed, deleted, or locked. Once the IT department detects the attack from the hacker, it will immediately re-set the blockchain, rendering all previous efforts of the hacker in vain. The hacker will not stop until the blockchain is dysfunctional. When the hacking times and detecting times follow arbitrary distributions, we derive the limiting functional probability, instantaneous functional probability, and mean functional time of the blockchain. We also show that all these quantities are increasing functions of the number of nodes, substantiating the intuition that the more nodes a blockchain has, the harder it is for a hacker to succeed in a cyber attack. In particular, this result formalizes the intuition that "a blockchain is safer than a single computer" for the first time in the literature. In addition, we argue that this result does not mean a firm should design a blockchain with as many nodes as possible by taking account of the cost of operation. We demonstrate that there is an optimal number of nodes in a blockchain to minimize the cost of operation.

Identification of reaction chains in metabolic and genomic networks for species comparison

Cabret Florent[1], Bocquillon Ronan[1], and Néron Emmanuel[1]

[1]Laboratoire d'Informatique Fondamentale et Appliquée, Université de Tours, Tours, France, ✉
florent.cabret@etu.univ-tours.fr

1 Introduction

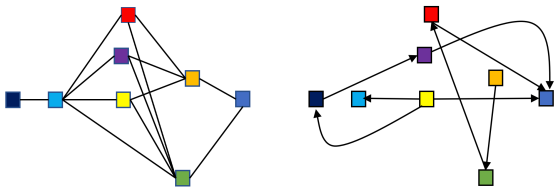
To understand the inner workings of living organisms, biologists have started to study their biological components separately. This approach no longer works for complex organisms and biologists are now looking at the relations between these components by comparing the networks they forms [2].

We are interested in the joint study of metabolic and genomic networks as a follow-up to previous work [3, 5].

We aim to identify species *markers* in order to compare them. The metabolic network is modelled by a directed graph D whose vertices represent reactions and whose arcs indicate that a reaction produces a metabolite that is the substrate of another reaction. Genomic proximity is represented by an undirected graph G, in which two reactions are linked by an edge if they are catalyzed by enzymes coded by "close" genes (separated by no more than δ_G genes apart). Note that both graphs are built on the same vertex set.

2 Problem description and references

Let D, be a directed acyclic graph and G an undirected graph, built on the same set of vertices $V = \{1, 2, \ldots, n\}$. A DG-consistent path is a path in D, such that the subgraph of G induced by the vertices of this path is connected. The problem *One-To-One SkewGraM* consists in calculating a longest DG-consistent path.



An instance of the *One-To-One SkewGraM* problem: on the left G; on the right D; nodes in orange, green, red and blue are a DG-consistant path of length 4.

Fertin and *al.* [3] introduced this problem, showed that it is NP-hard in the strong sense and proposed an exact *branch-and-bound* method.

More recently, the problem has been extended to the search of trails (simple but not necessarily elementary paths), i.e., paths that can pass several times through the same vertex, but not through the same arc, having maximum coverage [5]. Mathematical and constraint programming models [1] have been proposed for these two variants of the problem.

3 New binary integer programming formulation

In this section we present a new formulation for the trail variant. We start with a simple unitary flow problem (1-8) and each time the solver finds a new feasible or optimal solution we check its validity and generate constraints (9-10) to cut this solution if not.

$$
\begin{aligned}
\text{maximize} \quad & \textstyle\sum_{i \in V} z_i && && (1)\\
\text{subject to} \quad & \textstyle\sum_{j \in V} so_j = 1 && && (2)\\
& \textstyle\sum_{i \in V} si_i = 1 && && (3)\\
& so_v + \textstyle\sum_{(i,v) \in E(D)} x_{i,v} = si_v + \sum_{(v,j) \in E(D)} x_{v,j} && , \forall v \in V && (4)\\
& z_i \leq si_i + \textstyle\sum_{(i,j) \in E(D)} x_{i,j} && , \forall i \in V && (5)\\
& z_i \geq x_{i,j} && , \forall (i,j) \in E(D) && (6)\\
& z_i \geq si_i && , \forall i \in V && (7)\\
& z_i \leq \textstyle\sum_{(i,j) \in E(G)} z_j && , \forall i \in V && (8)\\
& z_i + z_j - \textstyle\sum_{v \in S} z_v \leq 1 && , \forall (i,j) \in \binom{V}{2}, S \subset V && (9)\\
& \textstyle\sum_{j \in V(C)} so_j + \sum_{\substack{i \in V \setminus V(C) \\ j \in V(C)}} x_{i,j} \geq \sum_{(i,j) \in E(C)} x_{i,j} - |E(C)| + 1 && , \forall C \subseteq D && (10)
\end{aligned}
$$

All the variables are binary and are defined as follow $z_i = 1$ if the node i is in the solution, $x_{i,j} = 1$ if the edge (i, j) is in the solution and $so_j / si_i = 1$ if the edge between the virtual source, resp. sink, and the node j, resp. i, is in the solution.

The goal is to maximize the number of nodes in the trail (1) that begins with the virtual source (2) and ends with the virtual sink (3) and where the intermediary nodes have the same number of in and out edges (4). A node is said to be in the solution if there exists an edge that begins with it (5-7).

With only these equations (1-7) we could have a trail and multiples disjoint cycles and there is no guarantee that the induced subgraph in G is connected.

In order to solve the first problem, we force each cycle C in the solution to have an edge that enters it (10).

To solve the second problem we force each pair of nodes that does not share an edge to have one of the node of theirs *node separator* in the solution (8-9). An (i, j)-vertex separator is the minimal set of nodes, denoted here as S , such that removing nodes from S from the graph G makes it so that there is no path between i and j. These constraints have been shown to perform really well for similar connectivity problems [4].

This new formulations called *branch-and-check* outperforms existing methods by two to three orders of magnitude in term of the runtime (from 100-1000 to 0.1-1 seconds).

4 Conclusion and future works

Building on the success of this new model, we plan to introduce new variants of the problem by incorporating comparisons between trails of different species directly into the objective function, and by defining a new notion of gene proximity.

References

[1] Mohamed Lemine Ahmed Sidi et al. “Improved Approaches to Solve the One-To-One SkewGraM Problem”. In: *Computers & Operations Research* 138 (Feb. 1, 2022), p. 105584. ISSN: 0305-0548. DOI: 10.1016/j.cor.2021.105584.

[2] Frank J. Bruggeman and Hans V. Westerhoff. “The Nature of Systems Biology”. In: *Trends in Microbiology* 15.1 (Jan. 1, 2007), pp. 45–50. ISSN: 0966-842X, 1878-4380. DOI: 10.1016/j.tim.2006.11.003. pmid: 17113776.

[3] Guillaume Fertin, Hafedh Mohamed Babou, and Irena Rusu. “Algorithms for Subnetwork Mining in Heterogeneous Networks”. In: *Experimental Algorithms*. Ed. by Ralf Klasing. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2012, pp. 184–194. ISBN: 978-3-642-30850-5. DOI: 10.1007/978-3-642-30850-5_17.

[4] Yiming Wang, Austin Buchanan, and Sergiy Butenko. “On Imposing Connectivity Constraints in Integer Programs”. In: *Mathematical Programming* 166.1 (Nov. 1, 2017), pp. 241–271. ISSN: 1436-4646. DOI: 10.1007/s10107-017-1117-8.

[5] Alexandra Zaharia et al. “CoMetGeNe: Mining Conserved Neighborhood Patterns in Metabolic and Genomic Contexts”. In: *BMC Bioinformatics* 20.1 (Jan. 10, 2019), p. 19. ISSN: 1471-2105. DOI: 10.1186/s12859-018-2542-2.