Submitted to *Transportation Science*
manuscript TS-2018-0377.R1

The Team Orienteering Problem with Overlaps: an Application in Cash Logistics

Christos Orlis
Department of Supply Chain Analytics, Vrije Universiteit Amsterdam, The Netherlands
c.orlis@vu.nl

Nicola Bianchessi
Department of Supply Chain Analytics, Vrije Universiteit Amsterdam, The Netherlands
Dipartimento di Informatica, Università degli Studi di Milano, Via Celoria 18, I-20122 Milan, Italy
n.bianchessi@vu.nl,bianchessi@di.unimi.it

Roberto Roberti, Wout Dullaert
Department of Supply Chain Analytics, Vrije Universiteit Amsterdam, The Netherlands
r.roberti@vu.nl,w.e.h.dullaert@vu.nl

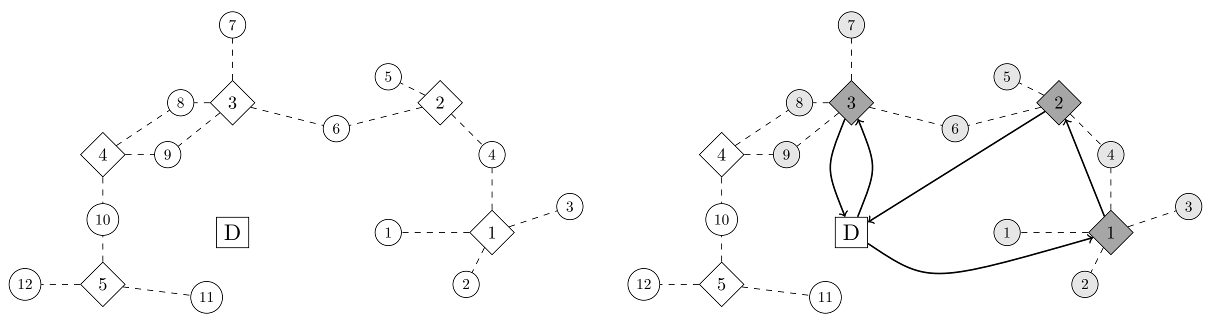
The *Team Orienteering Problem* (TOP) aims at finding a set of routes subject to maximum route duration constraints that maximize the total collected profit from a set of customers. Motivated by a real-life *Automated Teller Machine* (ATM) cash replenishment problem that seeks for routes maximizing the number of bank account holders having access to cash withdrawal, we investigate a generalization of the TOP that we call the *Team Orienteering Problem with Overlaps* (TOPO). For this problem, the sum of individual profits may overestimate the real profit. We present exact solution methods based on column generation and a metaheuristic based on large neighborhood search to solve the TOPO. An extensive computational analysis shows that the proposed solution methods can efficiently solve synthetic and real-life TOPO instances. Moreover, the proposed methods are competitive with the best algorithms from the literature for the TOP. In particular, the exact methods can find the optimal solution of 371 of the 387 benchmark TOP instances, 33 of which are closed for the first time.

Key words: team orienteering; cash distribution; routing with profits; column generation; metaheuristic
History: Submitted on November 8, 2018. First revision submitted on May 17, 2019.

1. Introduction

The *Team Orienteering Problem* (TOP) is a well-known decision-making problem of the class of Vehicle Routing Problem with Profits (see Archetti, Speranza, and Vigo (2014)). Given a set of vehicles and a set of customers, each one with an associated profit, the goal of the TOP is to design a set of routes (one for each vehicle) that maximizes the total profit collected by visiting (some of) the customers without exceeding a maximum duration constraint for each route.

The first paper on the TOP is owed to Butt and Cavalier (1994). Since then, many exact and heuristic approaches to solve the TOP and many of its variants have appeared in the literature



(a) (b)

Figure 1 Example of a TOPO instance (a) and a corresponding feasible solution (b)

(see Vansteenwegen, Souffriau, and Van Oudheusden (2011), Gunawan, Lau, and Vansteenwegen (2016) for recent surveys on the topic). The increasing interest in efficient solution methods to solve the TOP and related problems is due to the variety of real-life applications that can be modeled as TOPs, e.g., athlete recruiting (Chao, Golden, and Wasil (1996)), technician routing (Tang and Miller-Hooks (2005)), design of tourist trips (Vansteenwegen and Van Oudheusden (2007)), and customers selection in less-than-truckload transportation (Archetti et al. (2009)).

In this paper, we introduce and solve a new generalization of the TOP that we call the *Team Orienteering Problem with Overlaps* (TOPO). The TOPO is inspired by a real-life decision-making problem faced by Geldmaat, a joint venture of the three largest Dutch commercial banks (ABN AMRO, ING, and Rabobank) in charge of providing logistical services such as cash collection, counting, and distribution in the Netherlands. *Automated Teller Machines* (ATMs) are replenished by using a set of armored vehicles, whose routes are subject to maximum duration constraints deriving from legal restrictions that limit the time each driver can work every day. One of the main challenges faced by Geldmaat is to decide which ATMs to replenish to maximize the number of bank account holders that have access to ATMs. This is particularly challenging on days with peak demand, when most ATMs are empty and vehicles are in short supply. In our real-life setting, bank account holders (identified by the postal code of residence) are considered to have access to cash if there exists a replenished ATM within five kilometers from the residence postal code.

Figure 1(a) shows an example of a TOPO instance with five ATMs and 12 bank account holders. The depot is represented by the rectangle (D), the ATMs by diamonds (1-5), and the bank account holders by circles (numbered from 1 to 12). A dashed line between an ATM and a bank account holder indicates that the ATM can serve the bank account holder. Some bank account holders (e.g., 4, 6, 8, 9, and 10) can be served by two ATMs. A feasible solution for the instance is depicted in Figure 1(b). Two vehicles are used. The first serves ATMs 1 and 2, whereas the other serves just ATM 3. This solution allows to serve nine bank account holders (i.e., the grey circles). Notice that bank account holders 4 and 6 can withdraw from both ATMs, but they do not count twice in the total number of bank account holders served.

The TOPO also arises in other real-life distribution problems. A first application is the decision on which stores to replenish given that consumers can be served from one or several nearby stores. Another application is in humanitarian logistics, where first-aid resources or services have to be provided in large geographical areas and beneficiaries can be served at different service points. As the TOPO does not model just the problem faced by Geldmaat but also other real-life applications, we will refer to the ATMs as *service points* and the bank account holders as *consumers* in the rest of the paper.

The main contributions of this paper are the following: (*i*) we introduce the TOPO, a new generalization of the TOP that arises in real-life applications, together with (*ii*) exact branch-and-cut-and-price (BCP) algorithms that can solve to optimality instances with up to 100 service points; (*iii*) we show how the *ng*-path relaxation technique, introduced by Baldacci, Mingozzi, and Roberti (2011) to price out columns associated with possibly non-elementary paths, can be exploited not only to prevent some cycles but also to improve the linear relaxation bound by lifting some of the coefficients of the columns priced out; (*iv*) we propose a *Large Neighborhood Search* (LNS) metaheuristic for the TOPO that is able to find high-quality solutions of instances with up to 100 service points within short computation times; (*v*) we report on experimental results obtained by applying the best performing of our BCP algorithms to solve the TOP benchmark instances by Chao, Golden, and Wasil (1996), showing that it improves upon state-of-the-art exact methods by solving to optimality about 96% of the instances (371 out of 387) and closing 33 open instances; (*vi*) we show that both the best performing BCP algorithm and the LNS metaheuristic can provide high-quality solutions for real-life instances provided by Geldmaat with up to 100 service points and 203,717 consumers; (*vii*) from a managerial point of view, we report a quantitative analysis that sheds light on the potential losses that arise when the underlying real-life problem should be modeled as a TOPO but is addressed as a TOP.

The rest of the paper is organized as follows. Section 2 reviews the main contributions from the literature on exact and heuristic algorithms for the TOP. The TOPO is formally introduced in Section 3, where a compact formulation for the problem is provided. Section 4 describes the BCP algorithms. Section 5 illustrates the LNS metaheuristic. Section 6 discusses the computational results obtained by applying the devised algorithms on both synthetic and real-life instances. Finally, conclusions and future research directions are outlined in Section 7.

2. Literature Review

As the TOPO generalizes the well-known TOP (see Section 3), this section reviews the main exact and heuristic methods presented in the literature to solve the TOP. For an exhaustive overview of the literature on the TOP and related problems, the reader is referred to the surveys of Vansteenwegen, Souffriau, and Van Oudheusden (2011) and Gunawan, Lau, and Vansteenwegen (2016).

2.1. Exact Methods

The first exact method for the TOP is owed to Boussier, Feillet, and Gendreau (2007), who describe a branch-and-price algorithm based on a set packing formulation where each variable represents an elementary route. The pricing problem corresponds to an Elementary Shortest Path Problem with Resource Constraints (ESPPRC). Different acceleration techniques are proposed to solve the ESPPRC. The proposed branch-and-price is tested on the standard benchmark instances introduced by Chao, Golden, and Wasil (1996), hereafter referred to simply as *Chao instances*. The computational results show that 270 of the 387 Chao instances can be solved to optimality within two hours of computational time.

Poggi, Viana, and Uchoa (2010) describe three mathematical formulations and a robust branch-and-cut-and-price algorithm, where the pricing problem is solved by dynamic programming and two families of robust cuts (i.e., min cuts and triangle clique cuts) are separated. A partial implementation of the branching schemes is adopted, so only partial preliminary computational results on the Chao instances are reported.

Dang, El-Hajj, and Moukrim (2013) propose a branch-and-cut method based on a three-index formulation with a polynomial number of binary variables. The linear relaxation is tightened by adding different sets of valid inequalities and dominance properties, such as boundaries on profits, symmetry breaking, generalized subtour elimination, and clique constraints. The proposed branch-and-cut can solve 278 Chao instances within two hours of computational time.

Keshtkaran et al. (2016) build upon the exact method of Boussier, Feillet, and Gendreau (2007) to propose an enhanced branch-and-cut-and-price algorithm, where the pricing problem is solved by bounded bidirectional dynamic programming with decremental state-space relaxation and two-phase dominance rule relaxation. The Subset-Row inequalities introduced by Jepsen et al. (2008) are also separated to strengthen the linear relaxation of the Set Packing formulation. The proposed branch-and-cut-and-price can solve 301 Chao instances within two hours of computational time.

El-Hajj, Dang, and Moukrim (2016) describe a branch-and-cut algorithm building upon the three-index formulation and the cuts used by Dang, El-Hajj, and Moukrim (2013), but based on additional sets of valid inequalities. The resulting branch-and-cut approach can solve 300 Chao instances within two hours of computational time.

The most recent exact method is owed to Bianchessi, Mansini, and Speranza (2018), who introduce a branch-and-cut method based on a new two-index formulation with a polynomial number of variables and constraints. The linear relaxation is enforced by separating an exponential number of connectivity constraints with an exact algorithm for max-flow/min-cut problems. The proposed branch-and-cut algorithm can solve 327 of the 387 Chao instances within two hours of computational time. Nevertheless, 49 of the 387 Chao instances have not been solved to optimality by any of the exact methods presented so far in the literature.

2.2. Heuristic Methods

A variety of heuristics and metaheuristics have been proposed for the TOP since the early 2000s. In general, neighborhood-based approaches are more common than population-based approaches.

2.2.1. Neighborhood-Based Approaches. Tang and Miller-Hooks (2005) introduce a tabu search heuristic embedded in an adaptive memory procedure that explores both feasible and infeasible solutions, alternates between small and large neighborhoods in the solution improvement phase, and uses greedy and random components for generating neighborhood solutions.

Archetti, Hertz, and Speranza (2007) describe three metaheuristics, i.e., two generalized tabu search algorithms and a variable neighborhood search algorithm. The algorithms explore feasible and infeasible solutions. The impact of different strategies to jump between solutions, penalize infeasibility, and restore feasibility is assessed.

Vansteenwegen et al. (2009) describe an algorithm that combines different local search heuristics and uses Guided Local Search to improve the quality of the solutions achieved.

Two versions of a metaheuristic based on a pure path relinking approach combined with a greedy randomized adaptive search procedure are presented in Souffriau et al. (2010).

Lin (2013) proposes a multi-start simulated annealing algorithm that combines simulated annealing with multi-start hill climbing to minimize the chances of being trapped in local minima.

Kim, Li, and Johnson (2013) propose a large neighborhood search embedding three improvement algorithms (i.e., a local search, a shift-and-insertion, and a replacement improvement) and manage to compute all 387 best-known solutions of the Chao Instances.

Vidal et al. (2015) introduce a heuristic framework for solving the TOP and two other vehicle routing problems with profits and manage to compute the best-known solutions for all but one of the Chao instances. The heuristic is based on an exhaustive solution representation where first all customers are assigned to the vehicles and in a sequence that possibly violates the maximum-route duration constraints. Then, the final set of customers to be served is selected by exploring, in pseudo-polynomial time, an exponential neighborhood of solutions. Notice that Vidal et al. (2015) could also be classified as a population-based approach.

2.2.2. Population-Based Approaches. An ant colony optimization approach is proposed by Ke, Archetti, and Feng (2008). Four alternative methods (i.e., sequential, deterministic-concurrent, random-concurrent, and simultaneous) are applied to construct candidate solutions within the ant colony framework.

Bouly, Dang, and Moukrim (2010) present a memetic algorithm that combines genetic algorithms with local search techniques to improve the mutation phase. The encoding of a solution is based on a giant-tour representation, and an optimal split procedure is applied to evaluate the chromosomes. The algorithm of Bouly, Dang, and Moukrim (2010) is later hybridized with particle-swarm optimization in Dang, Guibadj, and Moukrim (2011) and further extended in Dang, Guibadj, and Moukrim (2013). The latter manages to compute all 387 best-known solutions of the Chao instances.

Ke et al. (2016) present a metaheuristic using a Pareto-dominance criterion to control the similarity between the generated and the incumbent solutions. Pareto-dominance is also used to update the population of incumbent solutions. This metaheuristic, similarly to Kim, Li, and Johnson (2013) and Dang, Guibadj, and Moukrim (2013) manages to compute all 387 best-known solutions for the Chao instances.

3. Problem Description

The TOPO can be formally described as follows. A set of consumers $\mathcal{C}$ is given. Consumers are served via a set of service points $\mathcal{S} = \{1, 2, \ldots, n\}$; in particular, each consumer $c \in \mathcal{C}$ can be served by a subset of service points $\mathcal{S}_c \subseteq \mathcal{S}$. Similarly, the subset of consumers that can be served by service point $i \in \mathcal{S}$ is indicated by $\mathcal{C}_i \subseteq \mathcal{C}$. Consumers can be served via the service points by routing a set of homogeneous vehicles $\mathcal{K}$ located at a depot, indicated by 0. Each vehicle can perform a route that starts from the depot, visits some service points, and returns to the depot. Each route cannot exceed a maximum route duration denoted by T. The travel time between each pair of depot/service point locations i and j ($i, j \in \mathcal{V} = \mathcal{S} \cup \{0\}$) is indicated by t_{ij}; travel times can be asymmetric and, without loss of generality, are assumed to be strictly positive and to satisfy the triangle inequality (i.e., $t_{ij} \leq t_{ik} + t_{kj}$ for each $i, j, k \in \mathcal{V}$). The TOPO aims at finding a set of routes, each one not exceeding the maximum route duration, that visit each service point at most once and maximize the number of consumers served.

In the TOP, a set of customers $\mathcal{U}$ is given. A profit r_u is associated with each customer $u \in \mathcal{U}$, and a set of m vehicles are located at the depot and can be used to serve the customers. The goal of the TOP is to find a set of at most m routes, each one not exceeding a maximum route duration T, such that each customer is served at most once and the total collected profit is maximized. We can observe that the TOP is a special case of the TOPO. Indeed, any TOP instance can be mapped into a TOPO instance as follows. The set $\mathcal{K}$ comprises m homogeneous vehicles, i.e., $|\mathcal{K}| = m$. Each customer of the TOP instance corresponds to a service point in the TOPO instance, and the

TOPO instance contains $|\mathcal{C}| = \sum_{u\in\mathcal{U}} r_u$ consumers. The sets $\mathcal{C}_u$ are defined in such a way that $\mathcal{C}_u$, $u \in \mathcal{U}$, contains r_u consumers (i.e., $|\mathcal{C}_u| = r_u$) and the sets $\mathcal{C}_u$ are pairwise disjoint (i.e., $\mathcal{C}_u \cap \mathcal{C}_{u'} = \varnothing$, $u, u' \in \mathcal{U} : u \neq u'$). Moreover, the set $\mathcal{S}_c$, $c \in \mathcal{C}$, contains a single element that corresponds to the only service point u such that $c \in \mathcal{C}_u$. It is easy to observe that any solution of the resulting TOPO instance corresponds to a solution of the original TOP instance.

The TOPO can be defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where the arc set is defined as $\mathcal{A} = \{(i,j) \,|\, i,j \in \mathcal{V} : i \neq j\}$. Let us define the following three sets of variables: $x_{ij} \in \{0,1\}$, binary variable equal to 1 if arc $(i,j) \in \mathcal{A}$ is traversed by one of the vehicles (0 otherwise); $y_c \in \{0,1\}$, binary variable equal to 1 if consumer $c \in \mathcal{C}$ is served (0 otherwise); and $z_i \in \mathbb{R}_+$, continuous variable indicating the arrival time at service point $i \in \mathcal{S}$. Then, the TOPO can be formulated as follows:

$$z^* = \max \sum_{c\in\mathcal{C}} y_c \tag{1a}$$

$$\text{s.t.} \sum_{(0,j)\in\mathcal{A}} x_{0j} \leq |\mathcal{K}| \tag{1b}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij} \leq 1 \qquad i \in \mathcal{S} \tag{1c}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij} = \sum_{(j,i)\in\mathcal{A}} x_{ji} \qquad i \in \mathcal{S} \tag{1d}$$

$$z_i + (T + t_{ij}) x_{ij} \leq z_j + T \qquad (i,j) \in \mathcal{A} : i,j \in \mathcal{S} \tag{1e}$$

$$t_{0i} \sum_{(i,j)\in\mathcal{A}} x_{ij} \leq z_i \leq \big(T - t_{i0}\big) \sum_{(i,j)\in\mathcal{A}} x_{ij} \qquad i \in \mathcal{S} \tag{1f}$$

$$\sum_{(i,j)\in\mathcal{A} : i\in\mathcal{S}_c} x_{ij} \geq y_c \qquad c \in \mathcal{C} \tag{1g}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in \mathcal{A} \tag{1h}$$

$$y_c \in \{0,1\} \qquad c \in \mathcal{C} \tag{1i}$$

The objective function (1a) asks for maximizing the number of consumers served. Constraint (1b) ensures that no more than $|\mathcal{K}|$ routes are designed. Constraints (1c) guarantee that each service point is visited at most once. Constraints (1d) are flow conservation constraints for the service points. Constraints (1e) link $\boldsymbol{x}$ and $\boldsymbol{z}$ variables to set the arrival time at each service point based on the traversed arcs and also prevent subtours in the designed routes. Constraints (1f) guarantee that if service point $i \in \mathcal{S}$ is visited, the arrival time of the vehicle visiting it is not less than the travel time from the depot to i and not greater than $T - t_{i0}$. Constraints (1g) ensure that each consumer $c \in \mathcal{C}$ is served only if at least one of the service points of the set $\mathcal{S}_c$ is visited. Constraints (1h)-(1i) define the range of the decision variables.

4. Exact Methods

In this section, we discuss the exact branch-and-cut-and-price (BCP) algorithms devised to address the TOPO. In Section 4.1, we introduce a first BCP algorithm. Then, starting from the BCP algorithm just presented, two alternative, potentially improving, versions are derived and discussed in Section 4.2.

4.1. Branch-and-Cut-and-Price Algorithm

In the following, we present the main features and components of the BCP algorithm.

4.1.1. Route-Based Model Let $\mathbb{H} = \{\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_{|\mathbb{H}|}\}$ be a partition of the consumers such that $\mathcal{S}_c = \mathcal{S}_{c'}$ for each pair of consumers $c, c' \in \mathcal{C}$, $c \neq c'$, belonging to the same subset $\mathcal{H}_h$, i.e., the two consumers can be served by the same subset of service points. In the following, each subset of the partition $\mathbb{H}$ is called *class of consumers* (or simply *class*) and we refer to each class corresponding to subset $\mathcal{H}_h$ by its index h. Moreover, let $\mathcal{S}(\mathcal{H}_h) \subseteq \mathcal{S}$ be the subset of service points that serve the consumers of the subset $\mathcal{H}_h$ - i.e., $\mathcal{S}(\mathcal{H}_h) = \mathcal{S}_c$ for each $c \in \mathcal{H}_h$ - and let $\mathcal{H}(i)$ be the subset of classes served by service point $i \in \mathcal{S}$ - i.e., $\mathcal{H}(i) = \{h \mid h = 1, \ldots, |\mathbb{H}|, i \in \mathcal{S}(\mathcal{H}_h)\}$. Let $\mathcal{R}$ be the set of all feasible routes in graph $\mathcal{G}$, and let a_{ir} be a integer coefficient indicating the number of times route $r \in \mathcal{R}$ visits service point $i \in \mathcal{S}$.

Let $\delta_h \in \mathbb{R}_+$ be a non-negative continuous variable (with a binary meaning) equal to 1 if the class of consumers $h = 1, \ldots, |\mathbb{H}|$ is not served (0 otherwise), and let $\xi_r \in \{0, 1\}$ be a binary variable equal to 1 if route $r \in \mathcal{R}$ is selected (0 otherwise).

$$[\overline{\mathrm{F}}] \quad \delta^* = \min \sum_{h=1}^{|\mathbb{H}|} |\mathcal{H}_h| \delta_h \tag{2a}$$

$$\text{s.t.} \sum_{r \in \mathcal{R}} a_{ir} \xi_r \leq 1 \qquad i \in \mathcal{S} \tag{2b}$$

$$\sum_{r \in \mathcal{R}} \xi_r \leq |\mathcal{K}| \tag{2c}$$

$$\sum_{r \in \mathcal{R}} \Big(\sum_{i \in \mathcal{S}(\mathcal{H}_h)} a_{ir} \Big) \xi_r + \delta_h \geq 1 \qquad h = 1, \ldots, |\mathbb{H}| \tag{2d}$$

$$\xi_r \in \{0, 1\} \qquad r \in \mathcal{R} \tag{2e}$$

$$\delta_h \in \mathbb{R}_+ \qquad h = 1, \ldots, |\mathbb{H}| \tag{2f}$$

The objective function (2a) aims at minimizing the number of consumers not served by any route; therefore the number of consumers served y^* is given by $|\mathcal{C}| - \delta^*$. Constraints (2b) ensure that each service point is visited at most once by the selected routes. Constraint (2c) guarantees that at most $|\mathcal{K}|$ routes are selected. Constraints (2d) model that each class of consumers h is either visited (in this case $\sum_{r\in\mathcal{R}}(\sum_{i\in\mathcal{S}(\mathcal{H}_h)} a_{ir})\xi_r \geq 1$ and $\delta_h = 0$) or not (in this case $\sum_{r\in\mathcal{R}}(\sum_{i\in\mathcal{S}(\mathcal{H}_h)} a_{ir})\xi_r = 0$ and $\delta_h = 1$). Constraints (2e)-(2f) define the range of the decision variables.

Let us call $\overline{\text{LF}}$ the linear relaxation of $\overline{\text{F}}$, and let $z(\overline{\text{LF}})$ be its optimal solution cost.

4.1.2. Pricing problem. Let $u_i \in \mathbb{R}_-$ be the dual variable associated with constraint (2b) of service point $i \in \mathcal{S}$ of $\overline{\text{LF}}$, $u_0 \in \mathbb{R}_-$ the dual variable associated with constraint (2c), and $v_h \in \mathbb{R}_+$ the dual variable of class $h = 1,\ldots,|\mathbb{H}|$ of constraint (2d). The dual of $\overline{\text{LF}}$ is

$$
\begin{aligned}
z(\text{D}) = \max & \sum_{i\in\mathcal{S}} u_i + |\mathcal{K}|u_0 + \sum_{h=1}^{|\mathbb{H}|} v_h \\
\text{s.t.} & \sum_{i\in\mathcal{S}} a_{ir}u_i + u_0 + \sum_{h=1}^{|\mathbb{H}|}\Big(\sum_{i\in\mathcal{S}(\mathcal{H}_h)} a_{ir}\Big)v_h \leq 0 && r \in \mathcal{R} \\
& 0 \leq v_h \leq |\mathcal{H}_h| && h = 1,\ldots,|\mathbb{H}| \\
& u_i \in \mathbb{R}_- && i \in \mathcal{V}
\end{aligned}
$$

Given the dual variables $(\boldsymbol{u},\boldsymbol{v})$, the pricing problem corresponds to finding the minimum reduced cost route, where the reduced cost $c_r(\boldsymbol{u},\boldsymbol{v})$ of route $r \in \mathcal{R}$ is defined as $c_r(\boldsymbol{u},\boldsymbol{v}) = -\sum_{i\in\mathcal{S}} a_{ir}u_i - u_0 - \sum_{h=1}^{|\mathbb{H}|}\Big(\sum_{i\in\mathcal{S}(\mathcal{H}_h)} a_{ir}\Big)v_h$.

Solving the pricing problem corresponds to finding a route of minimum reduced cost, i.e., solving the problem $\min_{r\in\mathcal{R}} c_r(\boldsymbol{u},\boldsymbol{v})$. This problem corresponds to the well-known *Elementary Shortest Path Problem with Resource Constraints* (ESPPRC), which is known to be NP-hard in the strong sense (Dror 1994). To simplify the pricing problem, it is common in the literature to price out *ng*-routes, thus allowing routes to visit some of the service points more than once and allow some sub-tours. The *ng*-path relaxation has been introduced in Baldacci, Mingozzi, and Roberti (2011) and applied to the TOPO as follows.

Let $\mathcal{N}_i \subseteq \mathcal{S}$ be a priori-defined set of selected service points associated with service point $i \in \mathcal{N}$ such that $i \in \mathcal{N}_i$. With each path $P = (0, i_1, i_2, \ldots, i_{\ell(P)})$ that starts from the depot, visits a set of $\ell(P)$ service points, and ends at service point $i_{\ell(P)}$, we associate a set $NG(P) \subseteq \mathcal{N}_{i_{\ell(P)}}$ defined as follows

$$NG(P) = \{i_k, k = 1,\ldots,\ell(P)-1 \mid i_k \in \cap_{j=k+1}^{\ell(P)} \mathcal{N}_{i_j}\} \cup \{i_{\ell(P)}\} \tag{4}$$

that is, the set $NG(P)$ contains the final vertex $i_{\ell(P)}$ visited by path P and any other vertex i_k, $k = 1,\ldots,\ell(P)-1$, that belongs to all sets $\mathcal{N}_{i_{k+1}}, \mathcal{N}_{i_{k+2}}, \ldots, \mathcal{N}_{i_{\ell(P)}}$ associated with the vertices visited by path P after visiting i_k. Note that, without forcing i to be included in $\mathcal{N}_i$ it would not have been possible to define $NG(P)$ as a subset of $\mathcal{N}_{i_{\ell(P)}}$.

An ng-path (NG, t, i) is a non-necessarily elementary path $P = (0, i_1, i_2, \ldots, i_{\ell(P)})$ that starts from the depot, visits a set of service points such that $NG(P) = NG$, has a total duration of t, and ends at vertex $i \in \mathcal{N}$ such that $i \notin NG(P')$, where $P' = (0, i_1, i_2, \ldots, i_{\ell(P)-1})$ is an ng-path. Any $(NG, t, 0)$-path ending at the depot at time $t > 0$ is called ng-route.

Let $f(NG, t, i)$ be the cost of a least cost ng-path (NG, t, i). Functions $f(NG, t, i)$ can be computed by using dynamic programming as follows. Define the state set $\mathbb{S} = \{(NG, t, i) : i \in \mathcal{V},\ t \in [0, T],\ NG \subseteq \mathcal{N}_i \text{ s.t. } i \in NG\}$. For each state $(NG, t, i) \in \mathbb{S}$, function $f(NG, t, i)$ is computed as

$$f(NG, t, i) = \min_{(NG', t', j) \in \Gamma(NG, t, i)} \{f(NG', t', j) + c_{ji}(\boldsymbol{u}, \boldsymbol{v})\}, \tag{5}$$

where the set $\Gamma(NG, t, i)$ contains the subset of predecessor states of (NG, t, i) defined as $\Gamma(NG, t, i) = \{(NG', t', j) \in \mathbb{S} \,|\, t' = t - t_{ji},\ j \in \mathcal{V} \setminus \{i\}, NG' \subseteq \mathcal{N}_j : j \in NG' \text{ and } NG' \cap \mathcal{N}_i = NG \setminus \{i\}\}$, and $c_{ji}(\boldsymbol{u}, \boldsymbol{v})$ is the reduced cost of arc $(j, i) \in \mathcal{A}$ with respect to the dual solution $(\boldsymbol{u}, \boldsymbol{v})$ defined as

$$c_{ji}(\boldsymbol{u}, \boldsymbol{v}) = -u_i - \sum_{h \in \mathcal{H}(i)} v_h \tag{6}$$

To compute functions $f(NG, t, i)$, the following initialization is required: $f(\varnothing, 0, 0) = 0$ and $f(\varnothing, t, 0) = \infty,\ t \in (0, T]$.

The number of functions $f(NG, t, i)$ to compute can be reduced by applying the following dominance rule.

Dominance 1 *Let $(NG, t, i), (NG', t', i) \in \mathbb{S}$ be two states such that (1) $f(NG, t, i) \le f(NG', t', i)$, (2) $t \le t'$, and (3) $NG \subseteq NG'$ (and such that one of the three conditions is strictly satisfied), then state (NG, t, i) dominates (NG', t', i).*

4.1.3. Speeding up the pricing problem solution. We use three acceleration techniques to speed up the pricing problem solution algorithm:

1. *Bi-directional ng-path (Righini and Salani 2006)*: ng-routes are priced out by generating ng-paths up to an half-way point and paths are combined to generate routes. We can use the time to set the half-way point and stop propagating a path as soon as the arrival time $t(P)$ at the last customer i of an ng-path (NG, t, i) exceeds $\lceil \frac{T}{2} \rceil$. A forward ng-path (NG_1, t_1, i_1) can be combined with a backward ng-path (NG_2, t_2, i_2) if either (NG_1, t_1, i_1) ends at the depot or $t_1 > \lceil \frac{T}{2} \rceil$, $i_1 = i_2$, $NG_1 \cap NG_2 = \{i_1\}$, and $t_1 + t_2 \le T$; moreover, the reduced cost $c_r(\boldsymbol{u}, \boldsymbol{v})$ of the route obtained by combining paths P_1 and P_2 $(r = P_1 \oplus P_2)$ is $c_r(\boldsymbol{u}, \boldsymbol{v}) = f(NG_1, t_1, i_1) + f(NG_2, t_2, i_2)$.

2. *Heuristic pricing*: Before solving the pricing problem to optimality, two heuristics are applied in sequence in order to eventually generate negative reduced cost columns. Both the heuristics apply the Dominance Rule 1 by ignoring the condition $NG_1 \subseteq NG_2$, relaxing then the criteria for the dominance. In addition, the first heuristic propagates each state (NG, t, i) towards the service points corresponding to the η_A arcs outgoing from i and having the lowest reduced costs $c_{ij}(\boldsymbol{u}, \boldsymbol{v})$. The second heuristic is run only if the first does not succeed in finding negative reduced cost columns. When both heuristics fail, the pricing problem is solved to optimality.
3. *Dynamic ng-path (Roberti and Mingozzi 2014)*: The idea is to compute increasingly better lower bounds by starting from small sets $\mathcal{N}_i$ and iteratively adding service points to the sets $\mathcal{N}_i$ based on the optimal fractional solution computed for $\overline{\text{LF}}$ with the current sets $\mathcal{N}_i$. In particular, the initial sets $\mathcal{N}_i$ contain the $\underline{\eta}_{N_i}$ service points closest to i and, at each iteration, the sets $\mathcal{N}_i$ are updated to eliminate the first shortest cycle contained in each route of the optimal fractional solution. The process iterates until either all the routes of the optimal fractional solution are cycle-free or all the possible updates would increase the cardinality of a set $\mathcal{N}_i$ beyond a given maximum value $\overline{\eta}_{N_i}$.

4.1.4. Valid inequalities. The linear relaxation of (2a) – (2f) can be strengthened by adding the well-known *Subset-Row* (SR) inequalities introduced by Jepsen et al. (2008) for the VRPTW. As commonly done, we consider SRs defined over triples of service points only, which is

$$\sum_{r \in \mathcal{R}} \left\lfloor \frac{a_{ir} + a_{jr} + a_{kr}}{2} \right\rfloor \xi_r \leq 1 \qquad \{i, j, k\} \subseteq \mathcal{S} \tag{7}$$

that can easily be separated by pure enumeration. In order to handle these non-robust cuts, the pricing problem solution algorithm is modified as illustrated in Jepsen et al. (2008).

4.1.5. Restricted master heuristic. To speed up the branch-and-price algorithm, we embed a restricted master heuristic (Joncour et al. 2010) in the solution framework. The basic idea behind restricted master heuristics is to solve, by means of a general mixed integer linear programming (MILP) solver, the master problem ($\overline{\text{F}}$) restricted to a subset of the generated columns. By removing constraints (2b) from model $\overline{\text{F}}$, any subset of generated columns (even including columns corresponding to routes with cycles) can be used to compute an infeasible solution (optimal w.r.t. the considered columns) that can be easily made feasible (without worsening its value) by removing all but one visit for each service point visited multiple times. Thus, each time we compute an optimal fractional solution for $\overline{\text{LF}}$, the columns defining the solution are used as they are to initialize the restricted master problem (from which constraints (2b) are removed) which is then solved by means of a general MILP solver. The heuristic is run up to a given level of the branch-and-bound tree as specified in Section 6.

4.1.6. Branching. We perform a traditional binary branching according to two hierarchical rules. The first rule is on whether a service point is visited or not. The second rule is on the arcs to traverse.

4.2. Improved Branch-and-Cut-and-Price Algorithms

The integrality gap of route-based model $\overline{\text{F}}$ presented in Section 4.1.1 is strongly affected by the coefficients of variables ξ_r in constraints (2d). Indeed, in constraint (2d) of a given class h ($h = 1, \ldots, |\mathbb{H}|$), the variable ξ_r associated with route $r \in \mathcal{R}$ has a coefficient $\overline{a}_{hr}$ equal to the number of times the service points that can serve the class h (i.e., $\overline{a}_{hr} = \sum_{i \in \mathcal{S}(\mathcal{H}_h)} a_{ir}$) are visited in the route.

Let us associate with each route r and each class h a binary coefficient $\underline{a}_{hr}$ defined as

$$\underline{a}_{hr} = \begin{cases} 1 & \text{if } \overline{a}_{hr} \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

that is, $\underline{a}_{hr}$ is equal to 1 if at least one of the service points that can serve the class h is visited by route r (0 otherwise). An alternative formulation for the TOPO is obtained from $\overline{\text{F}}$ by replacing in constraints (2d) the coefficients $\overline{a}_{hr} = \sum_{i \in \mathcal{S}(\mathcal{H}_h)} a_{ir}$ with $\underline{a}_{hr}$. The resulting formulation $\underline{\text{F}}$ is

$$[\underline{\text{F}}] \quad \delta^* = \min \sum_{h=1}^{|\mathbb{H}|} |\mathcal{H}_h| \delta_h \tag{8a}$$

$$\text{s.t.} \sum_{r \in \mathcal{R}} a_{ir} \xi_r \leq 1 \qquad i \in \mathcal{S} \tag{8b}$$

$$\sum_{r \in \mathcal{R}} \xi_r \leq |\mathcal{K}| \tag{8c}$$

$$\sum_{r \in \mathcal{R}} \underline{a}_{hr} \xi_r + \delta_h \geq 1 \qquad h = 1, \ldots, |\mathbb{H}| \tag{8d}$$

$$\xi_r \in \{0, 1\} \qquad r \in \mathcal{R} \tag{8e}$$

$$\delta_h \in \mathbb{R}_+ \qquad h = 1, \ldots, |\mathbb{H}| \tag{8f}$$

It is easy to observe that the linear relaxation of $\underline{\text{F}}$ (hereafter $\underline{\text{LF}}$) provides a lower bound $z(\underline{\text{LF}})$ on the TOPO that is greater than or equal to $z(\overline{\text{LF}})$. Nevertheless, it is computationally challenging to solve the pricing problem of $\underline{\text{F}}$ because the definition of coefficients $\underline{a}_{hr}$ requires to keep track of all service points visited by a route in order to properly take into account the dual variables $\boldsymbol{v}$. Therefore, even computing lower bounds to $z(\underline{\text{LF}})$ by pricing out non-elementary routes (such as ng-routes) may be computationally prohibitive.

In Sections 4.2.1 and 4.2.2, we propose two alternative ways of defining coefficients to replace $\overline{a}_{hr}$ in constraints (2d) that are computationally tractable when pricing out ng-routes and provide lower bounds that can be worse than $z(\underline{\text{LF}})$ but can be (significantly) better than $z(\overline{\text{LF}})$. Each of the definitions leads to an improved route-based model that can be used in place of model $\overline{\text{F}}$, and to an alternative, potentially improving, version of the BCP algorithm discussed in Section 4.1.1.

4.2.1. First Improved BCP Algorithm. In the first computationally-tractable improved route-based model (hereafter called F′), for each route $r \in \mathcal{R}$, we count the number of times a class h can be served by a service point j (i.e., $j \in \mathcal{S}(\mathcal{H}_h)$) visited in the route and the service point i visited right before j in the route cannot serve class h (i.e., $i \notin \mathcal{S}(\mathcal{H}_h)$). Therefore, given route $R = (0, i_1, i_2, \ldots, i_{\ell(R)}, 0)$, the number of times a'_{hr} class $h = 1, \ldots, |\mathbb{H}|$ is served by route R is computed as $a'_{hr} = \big|\{i_k, k = 1, \ldots, \ell(R) \,|\, i_k \in \mathcal{S}(\mathcal{H}_h),\, i_{k-1} \notin \mathcal{S}(\mathcal{H}_h)\}\big|$.

We can observe that $\underline{a}_{hr} \le a'_{hr} \le \overline{a}_{hr}$ for each class $h = 1, \ldots, |\mathbb{H}|$ and each route $r \in \mathcal{R}$. Let us call F′ the formulation obtained from $\underline{\mathrm{F}}$ by replacing coefficients $\underline{a}_{hr}$ with coefficients a'_{hr} in constraints (8d). Moreover, let LF′ be the linear relaxation of F′, and let $z(\mathrm{LF}')$ be its optimal solution cost. We can observe that $z(\overline{\mathrm{LF}}) \le z(\mathrm{LF}') \le z(\underline{\mathrm{LF}})$.

The pricing problem to compute $z(\mathrm{LF}')$ can be solved by using the same methods used to compute $z(\overline{\mathrm{LF}})$ by redefining the arc reduced costs (6) as

$$c'_{ji}(\boldsymbol{u}, \boldsymbol{v}) = -u_i - \sum_{h \in \mathcal{H}(i) \setminus \mathcal{H}(j)} v_h \tag{9}$$

4.2.2. Second Improved BCP Algorithm. In the second computationally-tractable improved route-based model (hereafter called F″), the number of times a route $r \in \mathcal{R}$ serves class $h = 1, \ldots, |\mathbb{H}|$ depends on the sets $\mathcal{N}_i \subseteq \mathcal{S}$ of selected service points associated with each service point $i \in \mathcal{S}$. As in definition (4), for each service point i_q visited along a route $r = (0, i_1, i_2, \ldots, i_{\ell(r)}, 0)$, let us define a set of service points $NG_q(r)$ associated with i_q as $NG_q(r) = \{i_k, k = 1, \ldots, q-1 \,|\, i_k \in \cap_{j=k+1}^{q} \mathcal{N}_{i_j}, \} \cup \{i_q\}$. The number of times a class h is served by route r is then counted as the number of visits to a service point i_q that can serve class h (i.e., $i_q \in \mathcal{S}(\mathcal{H}_h)$) and such that none of the service points in the set $NG_{q-1}(r)$ can serve class h (i.e., $NG_{q-1}(r) \cap \mathcal{S}(\mathcal{H}_h) = \varnothing$), which is $a''_{hr} = \big|\{i_q, q = 1, \ldots, \ell(r) \,|\, i_q \in \mathcal{S}(\mathcal{H}_h),\, NG_{q-1}(r) \cap \mathcal{S}(\mathcal{H}_h) = \varnothing\}\big|$. We can observe that $\underline{a}_{hr} \le a''_{hr} \le a'_{hr} \le \overline{a}_{hr}$. In particular, $\underline{a}_{hr} = a''_{hr}$ for each $h = 1, \ldots, |\mathbb{H}|$ and for each $r \in \mathcal{R}$ if $\mathcal{N}_i = \mathcal{S}$ for each $i \in \mathcal{S}$, and $a''_{hr} = a'_{hr}$ for each $h = 1, \ldots, |\mathbb{H}|$ and for each $r \in \mathcal{R}$ if $\mathcal{N}_i = \{i\}$ for each $i \in \mathcal{S}$.

Let us call F″ the formulation obtained from $\underline{\mathrm{F}}$ by replacing coefficients $\underline{a}_{hr}$ with coefficients a''_{hr} in constraints (8d). Moreover, let LF″ be the linear relaxation of F″, and let $z(\mathrm{LF}'')$ be its optimal solution cost. We can observe that $z(\mathrm{LF}') \le z(\mathrm{LF}'') \le z(\underline{\mathrm{LF}})$. Moreover, $z(\mathrm{LF}') = z(\mathrm{LF}'')$ if $\mathcal{N}_i = \{i\}$ for each $i \in \mathcal{S}$, and $z(\mathrm{LF}'') = z(\underline{\mathrm{LF}})$ if $\mathcal{N}_i = \mathcal{S}$ for each $i \in \mathcal{S}$.

The pricing problem to compute $z(\mathrm{LF}'')$ can be solved by using the same methods used to compute $z(\overline{\mathrm{LF}})$ by redefining recursion (5) as follows: $f(NG, t, i) = \min_{(NG', t', j) \in \Gamma(NG, t, i)} \{f(NG', t', j) + c''_{ji}(NG', \boldsymbol{u}, \boldsymbol{v})\}$, where $c''_{ji}(NG', \boldsymbol{u}, \boldsymbol{v})$ is the cost of a propagation along arc $(j, i) \in \mathcal{A}$ when $NG_j(r) = NG'$ defined as

$$c''_{ji}(NG', \boldsymbol{u}, \boldsymbol{v}) = -u_i - \sum_{h \in \mathcal{H}(i) : h \notin \bigcup_{k \in NG'} \mathcal{H}(k)} v_h \tag{10}$$

The use of sets $NG_q(r)$ to compute coefficients a''_{hr} implies a mono-directional propagation of the states. Actually, by propagating states backward along a route, it is possible to define column coefficients that are different from those computed by using forward propagation. Consider the example of a route $r = (0, i_1, i_2, i_3, 0)$, with $i_1 \in \mathcal{N}_{i_2}$, $i_3 \notin \mathcal{N}_{i_2}$, and $\mathcal{H}(i_1) \cap \mathcal{H}(i_2) = \mathcal{H}(i_2) \cap \mathcal{H}(i_3) = \varnothing$, $\mathcal{H}(i_1) \cap \mathcal{H}(i_3) = \{\bar{h}\}$. The forward propagation of the states along the route implies $a''_{\bar{h}r} = 1$, whereas $a''_{\bar{h}r} = 2$ when states are propagated backward. Given this asymmetry in the solution spaces implied by the forward and backward propagation of the states, the bi-directional acceleration technique is not applied while solving the pricing problem arising for F$''$.

To our knowledge, this is the first time that ng-paths are used not just to eliminate cycles but also to improve the linear relaxation bound by lifting some of the column coefficients. This benefit comes at the expense of an increase in the computational time needed to manage the propagation of the states and the definition of the column coefficients. Such an idea may be successfully applied to other routing problems where vehicles can fulfill (multiple) tasks by visiting locations and each task can be fulfilled by visiting a subset of all locations; this happens, for example, in applications of the Generalized Rural Postman Problem and its generalizations (see Drexl (2007)).

5. A Large Neighborhood Search Metaheuristic

In this section, we present a *Large Neighborhood Search* (LNS) metaheuristic for the TOPO. Since its introduction in Shaw (1998), LNS has proved to be an efficient tool for solving many vehicle routing problems inspired by real-life applications (e.g., Ropke and Pisinger (2006), Adulyasak, Cordeau, and Jans (2012), Masson, Lehuédé, and Péton (2013), Emde and Schneider (2018), and Hübner and Ostermeier (2018)). According to Pisinger and Ropke (2019), the main idea of LNS is to start from an initial solution and gradually improve it by alternately applying a destroy method and a repair method to the incumbent solution. The destroy method significantly changes the incumbent solution to guarantee diversification. The repair method intensifies the search in the neighborhood of the incumbent solution. An exhaustive review of LNS and related metaheuristics is provided by Pisinger and Ropke (2019).

5.1. Overview of the Proposed LNS

The `LNS` (Algorithm 1) starts with generating an initial solution $\mathcal{X}$ (Line 1) by using an adaptation of the well-known nearest-neighbor approach of Solomon (1987). In particular, a route is created for each vehicle by iteratively inserting the unvisited service point that minimizes the extra-mileage until no more service points can be added. The initial solution obtained is then added to the pool of best solutions found Ω and also represents the best-known solution $\mathcal{X}^*$ (Line 2).

The core of the `LNS` is represented by the main loop (Lines 3-11), which is iterated $\overline{\eta}_1$ times, where $\overline{\eta}_1$ is a parameter. At each iteration, the first step is to randomly choose a solution $\mathcal{X}$ from the pool (Line 4) according to a uniform probability distribution. The second step is to destroy solution $\mathcal{X}$ (Line 5) by randomly removing a percentage of the visited service points. This percentage is chosen according to a uniform distribution within two intervals $[\alpha_1, \beta_1]$ and $[\alpha_2, \beta_2]$, where $\alpha_1 < \beta_1 \ll \alpha_2 < \beta_2$. A percentage in the interval $[\alpha_1, \beta_1]$ intensifies the search around the incumbent solution whereas a percentage in the interval $[\alpha_2, \beta_2]$ diversifies the search. The service points to be removed are randomly selected with an equal probability. Then, the solution returned by the destroy method is improved by applying a repair method consisting of a local search procedure (Line 6), a random replacement of some of the visited service points (Line 7), and again the local search procedure (Line 8). More details on these procedures are provided in the next sections and in the pseudo-code descriptions of Algorithms 2 and 3. At the end of each iteration, the best solution found $\mathcal{X}^*$ is updated (Line 9) along with the pool of solutions Ω (Line 10), to ensure that it contains the ω best solutions found by `LNS`, ω being a parameter.

Algorithm 1 Overview of `LNS`

Input: TOPO input data

Parameters: max iterations, $\overline{\eta}_1$

Output: TOPO solution $\mathcal{X}^*$

1: $\mathcal{X} \leftarrow$ *generateInitialSolution*
2: $\mathcal{X}^* \leftarrow \mathcal{X}$, $\Omega \leftarrow \{\mathcal{X}\}$
3: **for** $\eta_1 = 1, \ldots, \overline{\eta}_1$ **do**
4: $\quad \mathcal{X} \leftarrow \mathit{randomlySelectFromPool}(\Omega)$
5: $\quad \mathcal{X} \leftarrow \mathit{destroySolution}(\mathcal{X})$
6: $\quad \mathcal{X} \leftarrow \mathit{applyLocalSearch}(\mathcal{X})$ ▷ See Algorithm 2
7: $\quad \mathcal{X} \leftarrow \mathit{replaceRandom}(\mathcal{X})$ ▷ See Algorithm 3
8: $\quad \mathcal{X} \leftarrow \mathit{applyLocalSearch}(\mathcal{X})$
9: $\quad \mathcal{X}^* \leftarrow \mathit{updateBest}(\mathcal{X}^*, \mathcal{X})$
10: $\quad \Omega \leftarrow \mathit{updatePool}(\Omega, \mathcal{X})$
11: **end for**
12: **return** $\mathcal{X}^*$

5.2. Local Search

The local search procedure applied at each iteration of the LNS is detailed in Algorithm 2. The input is a solution $\mathcal{X}$, and the output is the solution obtained by iteratively applying a sequence of local search operators. The local search procedure is repeated as long as the incumbent solution is improved by any of the operators.

Algorithm 2 *applyLocalSearch*($\mathcal{X}$)

Input: solution $\mathcal{X} = (R_1, R_2, \ldots, R_{|\mathcal{K}|})$

Output: new solution $\mathcal{X}$ improved with local search

do
  $\mathcal{X}' \leftarrow \mathcal{X}$
  do ▷ Routing improvement
    $\mathcal{X}'' \leftarrow \mathcal{X}$
    $\mathcal{X} \leftarrow$ *2-opt-intra*($\mathcal{X}$)
    $\mathcal{X} \leftarrow$ *OR-opt2-intra*($\mathcal{X}$)
    $\mathcal{X} \leftarrow$ *swap-inter*($\mathcal{X}$)
    $\mathcal{X} \leftarrow$ *relocate-inter*($\mathcal{X}$)
    $\mathcal{X} \leftarrow$ *2-1-exchange-inter*($\mathcal{X}$)
  while $\mathcal{X} \neq \mathcal{X}''$
  do ▷ Profit improvement
    $\mathcal{X}'' \leftarrow \mathcal{X}$
    $\mathcal{X} \leftarrow$ *replace*($\mathcal{X}$)
    $\mathcal{X} \leftarrow$ *add*($\mathcal{X}$)
  while $\mathcal{X} \neq \mathcal{X}''$
while $\mathcal{X} \neq \mathcal{X}'$
return $\mathcal{X}$

The local search procedure consists of two main phases (i.e., a *routing improvement* phase and a *profit improvement* phase), each one iterated until a local minimum is reached. The routing improvement phase does not change the set of visited service points (and therefore does not change the profit of the solution) and aims at minimizing the length of the shortest route in the incumbent solution. To this end, five local search operators are applied: (1) *2-opt-intra* (i.e., the well-known 2-opt operator applied within a route), (2) *OR-opt2-intra* (i.e., shifting sequences of two consecutive service points forward and backward in the corresponding route), (3) *swap-inter* (i.e., exchanging two service points of two different routes), (4) *relocate-inter* (i.e., moving a service point from its route to any position in any other route), and (5) *2-1-exchange-inter* (i.e., exchanging a service point of a route with two service points of another route). The profit improvement phase aims at increasing the total profit of the visited service points by applying two operators that change the set of service points: (1) *replace*, which replaces a visited service point with an unvisited service point, and (2) *add*, which inserts unvisited service points in any of the routes.

Notice that, computing the profit for a solution of the two profit improving operators can be time-consuming. Let $f_{\text{profit}}(S)$ be the total profit of the service points of the set $S \subseteq \mathcal{S}$, and let us call S the set of service points visited by the incumbent solution $\mathcal{X}$. A move of the *replace* operator tries to replace a service point $i \in S$ with a service point $j \in \mathcal{S} \setminus S$. Unfortunately, we can observe that the profit of the set of service points $(S \setminus \{i\}) \cup \{j\}$ may not be $f_{\text{profit}}(S) - f_{\text{profit}}(\{i\}) + f_{\text{profit}}(\{j\})$, indeed $f_{\text{profit}}((S \setminus \{i\}) \cup \{j\}) \leq f_{\text{profit}}(S) - f_{\text{profit}}(\{i\}) + f_{\text{profit}}(\{j\})$. Similarly, a generic move of the *add* operator tries to add a service point $j \in \mathcal{S} \setminus S$ to the routes of $\mathcal{X}$, but we can observe that $f_{\text{profit}}(S \cup \{j\})$ may be less than $f_{\text{profit}}(S) + f_{\text{profit}}(\{j\})$ (i.e., $f_{\text{profit}}(S \cup \{j\}) \leq f_{\text{profit}}(S) + f_{\text{profit}}(\{j\})$).

Therefore, to limit the computation time for evaluating the *replace* and *add* operators, we apply a memory-intensive pre-processing step that pre-computes the effect of adding/removing a service point to/from a set of visited service points on the total profit. In particular, for each service point $i \in \mathcal{S}$, let $\mathcal{O}(i) \subseteq \mathcal{S} \setminus \{i\}$ be the set of service points that serve at least one of the consumers served by service point i, i.e., $\mathcal{O}(i) = \{j \in \mathcal{S} \setminus \{i\} \,|\, \mathcal{C}_i \cap \mathcal{C}_j \neq \varnothing\}$. Let $\mathscr{P}(\mathcal{O}(i))$ be the powerset of $\mathcal{O}(i)$. For each subset of service points $O \in \mathscr{P}(\mathcal{O}(i))$, we pre-compute the difference $\Delta(O, i)$ between the profit of the set of service points $O \cup \{i\}$ and O, i.e., $\Delta(O, i) = f_{\text{profit}}(O \cup \{i\}) - f_{\text{profit}}(O)$. If it is possible to compute the values $\Delta(O, i)$ for each $i \in \mathcal{S}$ and each $O \in \mathscr{P}(\mathcal{O}(i))$, then the profit of any *replace* and *add* move during the `LNS` can easily be computed in constant time: any *replace* move that replaces the visited service point $i \in S$ with the unvisited service points $j \in \mathcal{S} \setminus S$ achieves a solution with profit equal to $f_{\text{profit}}(S) - \Delta(S \cap \mathcal{O}(i), i) + \Delta((S \setminus \{i\}) \cap \mathcal{O}(j), j)$, and any *add* move that adds unvisited service point $j \in \mathcal{S} \setminus S$ to the visited service points S achieves a solution with profit equal to $f_{\text{profit}}(S) + \Delta(S \cap \mathcal{O}(j), j)$. Clearly, if the cardinality of the sets $\mathcal{O}(i)$ is high, pre-computing all values $\Delta(O, i)$ may become prohibitive. In such a case, other strategies to estimate the increase (decrease) in the profit of the incumbent solution derived from adding (removing) a service point need to be devised. For example, one could create a subset of the service points $\hat{\mathcal{O}}(i) \subset \mathcal{O}(i)$ such that it is possible to generate the powerset $\mathscr{P}(\hat{\mathcal{O}}(i))$. In our computational experiments, we do not investigate the case where all the values $\Delta(O, i)$ cannot be computed a-priori due to memory limitations because such an issue does not appear in the real-life cases we consider.

5.3. Random Replacement

The *replaceRandom* procedure is performed once at each iteration of the `LNS` in order to increase the profit of the incumbent solution. The input of the procedure is a solution $\mathcal{X} = (R_1, R_2, \ldots, R_{|\mathcal{K}|})$, and the output is another solution with possibly a higher profit. The procedure has two parameters $\overline{\eta}_2$ and $\overline{\eta}_3$, which control the number of times a route can be changed to improve its profit and the maximum number of changes that can be performed every time, respectively. A detailed pseudo-code of the *replaceRandom* procedure is provided in Algorithm 3.

Algorithm 3 *replaceRandom*($\mathcal{X}$)

Input: solution $\mathcal{X} = (R_1, R_2, \ldots, R_{|\mathcal{K}|})$

Parameters: max iterations of changes per route, $\overline{\eta}_2$; max changes per iteration, $\overline{\eta}_3$

Output: new solution $\mathcal{X}$

```
1:  for r = 1, ..., |K| do
2:      for η2 = 1, ..., η̄2 do
3:          Randomly select a set S̃ of η3 service points from route R_r, η3 ~ U[1, min{η̄3, ℓ(R_r)}]
4:          if f_profit(S̃) < f_profit(S \ S) then
5:              Remove the service points S̃ from R_r
6:              Randomly insert as many unvisited service points as possible into R_r
7:              if the removals and insertions of service points decreased the profit of R_r then
8:                  Undo the changes of R_r
9:              end if
10:         end if
11:     end for
12: end for
13: return X
```

The *replaceRandom* procedure focuses on one route at a time (Line 1). For each route r, $\overline{\eta}_2$ attempts to improve the total profit are made (Line 2). In each attempt, a set $\tilde{\mathcal{S}}$ of η_3 service points visited by route R_r are randomly selected with a uniform distribution (Line 3); η_3 is a random number between 1 and $\min\{\overline{\eta}_3, \ell(R_r)\}$, where $\overline{\eta}_3$ is a parameter and $\ell(R_r)$ represents the number of service points visited by route R_r. The service points $\tilde{\mathcal{S}}$ are candidates to leave route R_r. If the total profit of the service points $\tilde{\mathcal{S}}$ is smaller than the profit of the unvisited ($\mathcal{S} \setminus S$) service points (Line 4), then first the service points $\tilde{\mathcal{S}}$ are removed from route R_r and marked as unvisited (Line 5), and secondly the unvisited service points are iteratively inserted into route R_r (in a random order) in the position that minimizes the distance to visit them (Line 6). If the removal and insertion operations on route R_r increase its profit (Line 6), then the changes are maintained; otherwise, the changed are undone (Line 7).

6. Computational results

This section presents a computational analysis of the branch-and-cut-and-price algorithms described in Section 4 and of the LNS metaheuristic presented in Section 5. For the sake of brevity, only aggregated results are presented and discussed in this section. Detailed results can be found in the e-companion of the paper.

All proposed algorithms have been implemented in C++, compiled with Visual Studio 2017 64-bit and tested on a single-core of an Intel Core i7-6700U running at 4.00 GHz, equipped with 24 GB of memory. Moreover, for the BCP algorithms, at each column generation iteration, the linear relaxation of model $\overline{\text{F}}$, F', or F'' is solved with CPLEX 12.7, which is also used as general MILP solver in the restricted master heuristic (see Section 4.1.5).

All the BCP algorithms share the following setting. The first heuristic to solve the pricing problem limits the number of propagations of a state to $\eta_A = 5$ service points. The dynamic ng-path technique is applied up to the third level of the branch-and-bound tree, with $\underline{\eta}_{N_i} = 3$ and $\overline{\eta}_{N_i} = 11$. Once the dynamic update of the sets $\mathcal{N}_i$ fails while processing a node, sets $\mathcal{N}_i$ cannot be further updated for that node. At most one SR inequality (7) at a time can be added to the considered model (i.e., the inequality associated with the greatest violation value, with a minimum violation threshold of 0.01), and at most two inequalities per branch-and-bound node. The separation algorithm checking for violated SR inequalities is run up to the tenth level of the branch-and-bound tree, for a maximum of 20 SR inequalities per sub-tree. In particular, the separation algorithm can be run only if the dynamic update of the sets $\mathcal{N}_i$ fails (or when the update of the sets is no more allowed). The restricted master heuristic is applied up to the tenth level of the branch-and-bound tree. Finally, branching is done on the variable associated with the most fractional value.

For the LNS, unless stated otherwise, the following parameter settings are used: $[\alpha_1, \beta_1] = [20\%, 30\%]$, $[\alpha_2, \beta_2] = [80\%, 90\%]$, $\omega = 50$, $\overline{\eta}_1 = 10000 - \lfloor \frac{\max\{0, 100-|\mathcal{S}|\}}{10} \rfloor \cdot 500$, $\overline{\eta}_2 = 30$, $\overline{\eta}_3 = 3$.

Hereafter, we will refer to the BCP algorithms based on models $\overline{\text{F}}$, F', or F'' as to `Baseline`, `BCP1`, `BCP2`, respectively. Computational times are reported in seconds throughout the section. Furthermore, to simplify the comparison with the literature, all primal and dual bounds are reported as if the objective function of the TOPO is to maximize the number of consumers served (as defined in (1a)) instead of minimize the number of consumers that are not served (as defined in (2a) and (8a)). Indeed, as observed in Section 4, objective function (2a) and (8a) can also be equivalently formulated as $\min\{|\mathcal{C}| - \delta^*\}$.

The computational analysis is conducted on three sets of test instances. The first set (discussed in Section 6.1) consists of 215 synthetic TOPO instances derived from the instances by Chao, Golden, and Wasil (1996) commonly used to assess the performance of solution methods for the TOP. The second set (discussed in Section 6.2) consists of the 387 Chao TOP instances, which, as explained in Section 3, is a special case of the TOPO. The third set (discussed in Section 6.3) consists of 10 real-life instances of the TOPO that are faced by Geldmaat in their cash supply chain, which motivated the research on the TOPO. For all instances, travel times are rounded to the nearest value with a precision of 1.0e-12, and post-processing is applied to enforce the triangular inequality. The first two sets of instances are available upon request; for confidentiality reasons, we cannot disclose the third set of instances.

6.1. Computational Results on Synthetic TOPO Instances

6.1.1. Description of the Instances The first set of instances consists of 215 synthetic TOPO instances we derived from the well-known TOP instances introduced by Chao, Golden, and Wasil (1996). The Chao instances include seven families of instances (numbered from 1 to 7), each with a fixed number of service points, overall ranging from 19 to 100. Each family of instances consists of three groups of instances, where each group contains the same number of instances and is characterized by a different number of vehicles (two, three, or four). Each instance of the resulting 21 groups is further characterized by a different maximum route duration T. It should be noted that in instances with small values of T, some of the service points may not be reachable. Table 1 summarizes the main features of each family, namely, number of service points ($|\mathcal{S}|$), number of instances (nInst), minimum and maximum route duration of the instances in each group ($\mathsf{T}_{|\mathcal{K}|=2}$, $\mathsf{T}_{|\mathcal{K}|=3}$, $\mathsf{T}_{|\mathcal{K}|=4}$).

Table 1 Features of the Chao instances

Family	$\lvert\mathcal{S}\rvert$	nInst	$\mathsf{T}_{\lvert\mathcal{K}\rvert=2}$	$\mathsf{T}_{\lvert\mathcal{K}\rvert=3}$	$\mathsf{T}_{\lvert\mathcal{K}\rvert=4}$
1	30	54	2.5-42.5	1.7-28.3	1.2-21.2
2	19	33	7.5-22.5	5.0-15.0	3.8-11.2
3	31	60	7.5-55.0	5.0-36.7	3.8-27.5
4	98	60	25.0-120.0	16.7-80.0	12.5-60.0
5	64	78	2.5-65.0	1.7-43.3	1.2-32.5
6	62	42	7.5-40.0	5.0-26.7	3.8-20.0
7	100	60	10.0-200.0	6.7-133.3	5.0-100.0

To generate the 215 synthetic TOPO instances, we selected for each group of the Chao instances up to three TOP instances for which all service points are reachable. In particular, we selected those instances with the smallest, largest and median maximum route duration in the group, resulting in 43 instances as outlined in Table 2, where $\mathsf{nInst}_{|\mathcal{K}|=2}$, $\mathsf{nInst}_{|\mathcal{K}|=3}$, and $\mathsf{nInst}_{|\mathcal{K}|=4}$ indicate the number of instances per group with two, three, and four vehicles.

Then, for each of these 43 instances, five TOPO instances are derived by applying a three-step procedure. In the first step, a service radius ρ defining the maximum distance between a service

Table 2 Features of the selected Chao instances

Family	$\|\mathcal{S}\|$	nInst	$\text{nInst}_{\|\mathcal{K}\|=2}$	$\text{nInst}_{\|\mathcal{K}\|=3}$	$\text{nInst}_{\|\mathcal{K}\|=4}$
1	30	6	3	3	0
3	31	5	3	2	0
4	98	9	3	3	3
5	64	9	3	3	3
6	62	7	3	3	1
7	100	7	3	3	1

point and the consumers it can serve is computed as $\rho = 0.5 * \min\{t_{ij} : i, j \in \mathcal{S} : i \neq j\}$. The service radius ρ subsequently is used to define non-overlapping circular service regions centered around each service point. In the second step, for every service point (within its service region), as many consumers as the profit of the associated service point in the original TOP instance are first randomly generated and then allocated. In the third step, we compute the smallest value of ρ such that $\sum_{i \in \mathcal{S}} |\mathcal{C}_i| \geq |\mathcal{C}| * (1+\gamma)$, where $\gamma = 0.1$, 0.2, 0.3, 0.4, and 0.5. This results into a total of 215 TOPO instances with varying degrees of overlap. This procedure guarantees that the optimal solution value of the original TOP instance is a valid lower bound to the optimal solution value of any of the resulting five TOPO instances, thus allowing a direct assessment of the impact of increasing overlaps among service regions. Further details on the instance generator are available upon request.

6.1.2. Preliminary Computational Results of the Exact Methods In order to compare the performance of the three branch-and-cut-and-price algorithms, we first conducted preliminary experiments on all 215 TOPO instances with a short time limit of 900 seconds. In particular, to assess the impact of the implemented dynamic ng-path acceleration technique, the three algorithms were tested under two settings: static and dynamic depending on whether the sets $\mathcal{N}_i$ are defined in a static or dynamic way. The dynamic setting is the default one. The algorithms under the static setting are obtained by imposing $\underline{\eta}_{N_i} = 11$.

Table 3 Preliminary results, grouped by degree of overlap, of `Baseline`, `BCP1`, and `BCP2` (static and dynamic)

		Static						Dynamic					
		`Baseline`		`BCP1`		`BCP2`		`Baseline`		`BCP1`		`BCP2`	
Overlap	nInst	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}
10%	43	26	0.59	32	0.35	33	0.47	31	0.52	35	0.25	30	0.92
20%	43	26	1.23	28	0.47	28	1.02	25	1.03	30	0.34	29	0.86
30%	43	21	1.78	30	0.55	27	1.09	22	1.42	30	0.39	29	1.08
40%	43	16	2.84	27	0.72	26	0.87	19	2.70	29	0.59	25	1.41
50%	43	15	3.35	26	0.72	27	0.66	16	3.13	29	0.46	27	1.19
	215	104	1.96	143	0.56	141	0.82	113	1.76	153	0.41	140	1.09

Table 4 Preliminary Results, grouped by number of vehicles, of Baseline, BCP1, and BCP2 (static and dynamic)

		Static						Dynamic					
		Baseline		BCP1		BCP2		Baseline		BCP1		BCP2	
$\|\mathcal{K}\|$	nInst	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}	opt	gap_{Op}
2	90	49	1.96	62	0.62	58	1.24	51	1.81	68	0.38	55	1.94
3	85	38	1.99	55	0.58	56	0.55	44	1.72	58	0.44	58	0.54
4	40	17	1.87	26	0.39	27	0.47	18	1.73	27	0.38	27	0.34
	215	104	1.96	143	0.56	141	0.82	113	1.76	153	0.41	140	1.09

Tables 3 and 4 summarize these preliminary experiments by grouping them by degree of overlap and number of vehicles, respectively. For each group of instances, each table reports the number of instances (nInst), the number of instances solved to optimality (opt) and the average final percentage gap over the open instances (gap_{Op}) for each of the three algorithms under the two settings. The average percentage gap for each instance is computed as $(\frac{\text{ub}_\text{F}}{\text{blb}} - 1) * 100$, where ub_F is the final best upper bound and blb is the best lower bound computed.

Table 3 shows that, under the static setting, BCP1 solves more instances (i.e., 143) than Baseline and BCP2, which solve 104 and 141 instances, respectively, and the final gap is on average smaller (i.e., 0.56% vs 1.96% and 0.82%, respectively). Similar results are achieved under the dynamic setting. The table also indicates that Baseline and BCP1 solve more instances and provide lower gaps under the dynamic setting than the static setting, whereas BCP2 has better performance under the static setting. Nevertheless, BCP1 under both settings performs better than the other two algorithms. In particular, for BCP2 the increased computational overhead seems to have a substantial impact on its performance, nullifying the eventual improvement in the linear relaxation bound (see Section 4.2.2). We can also see that the performance of all six algorithms tends to be better when the overlaps are smaller (i.e., Overlap 10% and 20%) and deteriorates when the overlaps are larger (i.e., Overlap 40% and 50%). This indicates that instances with larger overlaps among service regions are more challenging. Table 4 indicates that BCP1 is significantly better than Baseline and BCP2 on instances with two vehicles.

A more detailed comparison of the performance of BCP1 in the static (BCP1-Static) and the dynamic (BCP1-Dynamic) setting is provided in Tables 5 and 6 on the basis of the 138 instances solved to optimality under both settings. Each row of these two table reports the number of instances commonly solved to optimality (opt) in the corresponding group, and, w.r.t. the instances solved, the average upper bound at the root node (ub_R), the average root solution time (cpu_R), and the average total solution time per instance (cpu_F). Average values for BCP1-Dynamic are reported as geometric mean of the ratios with respect to the corresponding values of the static version. For BCP1-Dynamic, we also report the average cardinality of the sets $\mathcal{N}_i$. Tables 5 and 6 show

Table 5 Full comparison between BCP1-Static and BCP1-Dynamic: results grouped by degree of overlap

		BCP1-Static			BCP1-Dynamic			
Overlap	opt	$\mathsf{ub_R}$	$\mathsf{cpu_R}$	$\mathsf{cpu_F}$	$\lvert\mathcal{N}_i\rvert$	$\mathsf{ub_R}$	$\mathsf{cpu_R}$	$\mathsf{cpu_F}$
10%	32	312.2	6.0	123.7	4.2	1.00	1.50	0.85
20%	27	311.5	3.0	62.7	4.0	1.00	1.55	1.00
30%	28	289.7	2.8	101.2	4.1	1.01	1.94	0.82
40%	26	297.2	2.3	119.9	4.0	1.00	1.84	0.94
50%	25	252.4	2.4	114.3	4.1	1.00	1.50	0.75
	138	293.8	3.4	104.8	4.1	1.00	1.65	0.87

Table 6 Full comparison between BCP1-Static and BCP1-Dynamic: results grouped by number of vehicles

		BCP1-Static			BCP1-Dynamic			
$\lvert\mathcal{K}\rvert$	opt	$\mathsf{ub_R}$	$\mathsf{cpu_R}$	$\mathsf{cpu_F}$	$\lvert\mathcal{N}_i\rvert$	$\mathsf{ub_R}$	$\mathsf{cpu_R}$	$\mathsf{cpu_F}$
2	59	320.7	3.3	123.7	4.2	1.00	1.41	0.93
3	53	247.3	2.9	86.3	4.2	1.00	2.28	0.97
4	26	327.7	4.6	99.4.9	3.6	1.00	1.22	0.60
	138	293.8	3.4	104.8	4.1	1.00	1.65	0.87

that the versions of BCP1 provide similar average upper bounds at the root node, even though the cardinality of the sets $\mathcal{N}_i$ is on average just 4.1 for BCP1-Dynamic. Albeit BCP1-Dynamic takes on average 65% more time than BCP1-Static to terminate the root node, it takes on average 13% less time to close the instances. Moreover, it seems that the number of vehicles does not affect the performance of the two algorithms.

6.1.3. Results of the Exact Method As the preliminary results for a time limit of 900 seconds suggested that BCP1 (in its dynamic setting, i.e., the default setting) is on average superior to the other algorithms, we extensively tested it on all 215 instances for a time limit of one hour to solve each instance. Table 7 summarizes the results achieved grouped by degree of overlap and by number of vehicles. The following information is reported: number of instances (nInst), average gap at the root node before changing the $\mathcal{N}_i$ sets and before adding SR inequalities ($\mathsf{gap_0}$), average final gap at the root node ($\mathsf{gap_R}$), number of instances solved to optimality (opt), average solution time ($\mathsf{cpu_F}$), number of instances open (nOpen), and average gap between the highest upper bound of the unexplored nodes left in the search tree and the best lower bound computed ($\mathsf{gap_{Op}}$).

Table 7 shows that BCP1 can solve 179 of the 215 instances and the average gap left for the 36 open instances is quite small (i.e., 0.44%). The average solution time is five minutes. When comparing the results on instances with different overlaps, it is clear that the higher the overlap, the more difficult the instances are: this is due to the fact that the root node provides worse upper bounds when the overlaps increase.

Table 7 Summary of the results of BCP1 with one hour of time limit

Overlap	$\|\mathcal{K}\|$	nInst	gap_0	gap_R	opt	cpu_F	nOpen	gap_{Op}
10%		43	1.26	0.48	38	214.2	5	0.23
	2	18	1.17	0.42	17	233.3	1	0.00
	3	17	1.50	0.48	15	262.4	2	0.27
	4	8	0.96	0.62	6	39.6	2	0.32
20%		43	1.34	0.52	38	353.2	5	0.39
	2	18	1.29	0.34	18	461.5	0	-
	3	17	1.52	0.64	14	312.6	3	0.33
	4	8	1.07	0.70	6	122.8	2	0.48
30%		43	1.48	0.68	36	303.1	7	0.41
	2	18	1.53	0.65	15	327.2	3	0.32
	3	17	1.67	0.74	15	368.7	2	0.45
	4	8	0.93	0.62	6	78.6	2	0.52
40%		43	1.58	0.80	34	309.4	9	0.41
	2	18	1.57	0.69	15	283.4	3	0.39
	3	17	1.94	1.03	13	363.8	4	0.48
	4	8	0.82	0.57	6	256.6	2	0.31
50%		43	1.66	0.84	33	319.9	10	0.59
	2	18	1.67	0.74	16	270.6	2	0.84
	3	17	2.12	1.16	11	252.8	6	0.58
	4	8	0.68	0.42	6	574.3	2	0.39
		215	1.46	0.67	179	299.1	36	0.44

6.1.4. Results of the LNS Table 8 summarizes the computational results of LNS on the 215 synthetic TOPO instances. Results are grouped by degree of overlap and by number of vehicles. On each instance, LNS was run ten times. Columns report the following information: number of instances (nInst), the number of instances on which LNS found the best-known solution in any of the runs even if it was not proved to be optimal by BCP1 (bk), the number of instances on which LNS found an optimal solution in any of the runs (opt), the percentage gap between the best solution found by the LNS and the best primal bound found by BCP1 (gap_{BPB} - notice that negative gaps mean that the best solution found by LNS is better than the best solution found by BCP1), the percentage gap between the best solution found by LNS and the best dual bound provided by the BCP1 (gap_{BDB}), the percentage gap between the average best solution found by LNS over all runs and the best dual bound provided by BCP1 (gap_{ADB}), and the average solution time (cpu_F). Table 8 shows that LNS could find the best-known solutions of 183 out of 215 instances. Moreover, the average gaps between the best and the average lower bounds provided by LNS and the upper bounds provided by BCP1 are small (i.e., 0.13% and 0.23%, respectively), yet these values may be overestimated because not all upper bounds provided by BCP1 were proved to be optimal. We can also observe that, similar to BCP1, LNS performs better on instances with smaller overlaps.

Table 8 Summary of the results of LNS on all 215 TOPO instances

Overlap	$\|\mathcal{K}\|$	nInst	bk	opt	gap_{BPB}	gap_{BDB}	gap_{ADB}	cpu_F
10%		43	36	33	-0.05	0.07	0.17	200.9
	2	18	14	14	-0.02	0.07	0.17	134.0
	3	17	15	14	-0.10	0.05	0.14	208.5
	4	8	7	5	-0.02	0.11	0.20	335.0
20%		43	39	35	-0.04	0.07	0.15	205.9
	2	18	16	16	0.01	0.01	0.12	140.8
	3	17	16	14	-0.08	0.07	0.14	207.8
	4	8	7	5	-0.04	0.19	0.25	348.5
30%		43	36	30	-0.04	0.15	0.23	206.2
	2	18	14	12	-0.06	0.12	0.18	145.2
	3	17	15	13	-0.07	0.12	0.21	199.0
	4	8	7	5	0.07	0.27	0.37	358.5
40%		43	36	31	-0.03	0.15	0.28	205.2
	2	18	13	13	0.07	0.15	0.25	147.0
	3	17	16	13	-0.15	0.15	0.29	195.9
	4	8	7	5	0.01	0.17	0.33	355.9
50%		43	36	29	-0.07	0.20	0.34	211.5
	2	18	14	13	0.00	0.18	0.33	151.1
	3	17	15	12	-0.12	0.26	0.40	199.1
	4	8	7	5	-0.16	0.12	0.27	374.0
		215	183	158	-0.04	0.13	0.23	205.9

We have conducted further experiments, summarized in Table 9, to shed light on the effectiveness of some key components of the LNS presented in Section 5. In particular, we tested the impact of (*i*) the pre-processing step, (*ii*) the acceptance criterion of the intra-route operators, (*iii*) the *replaceRandom* procedure, and (*iv*) the selection criterion of the solution from the pool Ω. Therefore, we tested four additional versions of the LNS on all 215 synthetic TOPO instances. In LNS2, the pre-processing step is not performed, so the *replace* and *add* operators need to compute the profit of the solutions in the neighborhoods in a more computationally intensive way; in this version, a time limit of 720.0 seconds was imposed on each experiment - this time limit exceeds the maximum computation time (i.e., 658.1 seconds) observed when LNS is used. In LNS3, the acceptance criterion of the intra-route operators is not the minimization of the length of the shortest route but rather the minimization of the length of all routes in the incumbent solution. In LNS4, the *replaceRandom* procedure is removed. Finally, whereas in LNS in each iteration a solution is randomly selected from the pool Ω according to a uniform probability distribution, in LNS5 the probability is proportional to the solution quality.

Table 9 shows that versions LNS2 and LNS4 have a significantly worse performance than LNS, thus proving that the pre-processing step and the *replaceRandom* procedure are crucial. The computational results for LNS3 illustrate that the alternative acceptance criterion of minimizing the length of all routes prevents finding two optimal and three best-known solutions found by LNS and results in an increased computing time. Finally, the results of LNS5 and LNS are nearly identical.

Table 9 **Evaluation of the different functionality modules of LNS**

Version	nInst	bk	opt	gap_{BPB}	gap_{BDB}	gap_{ADB}	cpu_F
LNS	215	183	158	-0.04	0.13	0.23	205.9
LNS2	215	123	110	0.32	0.43	0.86	572.7
LNS3	215	180	156	-0.05	0.12	0.22	241.2
LNS4	215	139	115	0.15	0.32	0.71	113.6
LNS5	215	183	158	-0.04	0.13	0.23	205.2

6.2. Computational Results on Benchmark TOP Instances

As the TOPO generalizes the TOP, which has been extensively studied in the literature, we tested BCP1 and LNS also on the Chao TOP instances to compare our solution method against the state-of-the-art solution methods from the literature. The results are summarized in this section.

6.2.1. Results of the Exact Method Table 10 summarizes the computational results achieved by BCP1 on the 387 TOP instances. Results are grouped by family of instances and reported in the same columns of Table 7. Table 10 shows that BCP1 can solve all but 16 instances to optimality with an average solution time of 122.7 seconds. The success of BCP1 is clearly due to the quality of the upper bounds provided by the root node (i.e., 1.26% and 0.44% before and after adding SR inequalities and changing the sets $\mathcal{N}_i$). The gap left on the 16 open instances is also quite small (i.e., 0.30%). It is worth noting that, considering the best lower bounds available from the literature, BCP1 could prove the optimality of three additional instances, namely p4.2.s, p4.3.t, and p4.4.r, in addition to the 371 instances solved.

In Table 11, we compare the results achieved by BCP1 with those achieved by the state-of-the-art solution methods from the literature, in particular with the branch-and-price algorithm of Boussier, Feillet, and Gendreau (2007) (BFG07), the branch-and-cut algorithm of Dang, El-Hajj, and Moukrim (2013) (DEM13), the branch-and-cut-and-price algorithm of Keshtkaran et al. (2016) (KZBV16), the branch-and-cut plane algorithm of El-Hajj, Dang, and Moukrim (2016) (EDM16), and the branch-and-cut algorithm of Bianchessi, Mansini, and Speranza (2018) (BMS18). For the sake of completeness, we also report the processor details of the computing environments used by

Table 10 Summary of the results of BCP1 on the TOP instances

Family	nInst	gap_0	gap_R	opt	cpu_F	nOpen	gap_{Op}
1	54	0.65	0.39	54	0.1	0	-
2	33	0.30	0.13	33	0.0	0	-
3	60	1.91	0.96	60	0.7	0	-
4	60	1.23	0.56	49	509.0	11	0.26
5	78	1.00	0.34	75	137.0	3	0.47
6	42	0.34	0.31	42	12.0	0	-
7	60	2.71	0.26	58	168.3	2	0.19
	387	1.26	0.44	371	122.7	16	0.30

the six methods. Results are summarized by family of instances. For each method, we report the number of instances solved to optimality (opt) and the average solution time (cpu_F) - we report na when the average solution time is not available.

Table 11 Comparison between BCP1 and the state-of-the-art exact methods on TOP instances

		BFG07 Intel Pentium IV (3.20 GHz)		DEM13 AMD Opteron (2.60 GHz)		KZBV16 Intel Core i7 (3.60 GHz)		EDM16 AMD Opteron (2.60 GHz)		BMS18 Intel Xeon (W3680)		BCP1 Intel Core (i7-6700U)	
Family	nInst	opt	cpu_F	opt	cpu_F	opt	cpu_F	opt	cpu_F	opt	cpu_F	opt	cpu_F
1	54	51	38.2	54	na	54	12.9	54	na	54	1.1	54	0.1
2	33	33	0.1	33	na	33	0.1	33	na	33	0.2	33	0.0
3	60	50	103.8	60	na	60	258.3	60	na	60	184.9	60	0.7
4	60	25	459.9	22	na	20	120.4	30	na	39	870.4	49	509.0
5	78	48	200.6	44	na	60	252.2	54	na	60	517.9	75	137.0
6	42	36	286.1	42	na	36	203.2	42	na	36	22.1	42	12.0
7	60	27	203.2	23	na	38	768.6	27	na	45	992.8	58	168.3
	387	270		278		301		300		327		371	

Table 11 shows that BCP1 can solve more instances than all the methods available from the literature. In particular, BCP1 can solve 44 instances more than the recent branch-and-cut algorithm of Bianchessi, Mansini, and Speranza (2018) and 71 instances more than the most recent column-generation-based algorithm of Keshtkaran et al. (2016). We can also mention that BCP1 solves 33 out of the 49 instances that were still open (see Section 2.1 and the e-companion). Finally, we note that for instance p4.4.n, we computed an optimal value of 976, which is inconsistent with the primal bound of 977 reported in Tang and Miller-Hooks (2005) but consistent with all the other primal bounds reported in the literature so far.

6.2.2. Results of LNS Table 12 summarizes the computational results of LNS on the TOP instances and compares them with the results of Dang, Guibadj, and Moukrim (2013) (DGM13), Vidal et al. (2015) (VOP15), and Ke et al. (2016) (KLC16) - the reader is referred to Section 2 for a discussion of these methods. For the sake of completeness, we also report the processor details of the computing environments used to test the four solution methods. For each method, the table reports the number of instances on which the best-known solution was found (bk), the average gap between the best solution found by the heuristic and the best-known solution available (gap_{BPB}), and the average solution time (cpu_{F}). The method VOP15 was tested on a subset of instances only, so the last two rows of the table summarize the results over all seven families for DCM13, KLC16, and `LNS` and over families 4-7 for VOP15 and `LNS`.

Table 12 Comparison between `LNS` and the state-of-the-art heuristic methods on TOP instances

		DGM13 AMD Opteron (2.60 GHz)			VOP15 Intel Xeon (3.07 GHz)			KLC16 Intel Core i5 (3.2 GHz)			`LNS` Intel Core i7-6700U		
Family	nInst	bk	gap_{BPB}	cpu_{F}	bk	gap_{BPB}	cpu_{F}	bk	gap_{BPB}	cpu_{F}	bk	gap_{BPB}	cpu_{F}
1	54	54	0.0	2.1				54	0.0	6.7	54	0.0	3.2
2	33	33	0.0	0.4				33	0.0	1.4	33	0.0	0.3
3	60	60	0.0	3.2				60	0.0	9.6	60	0.0	4.2
4	60	60	0.0	214.1	58	0.0038	224.2	60	0.0	108.5	60	0.0	209.2
5	78	78	0.0	49.3	78	0.0	110.6	78	0.0	22.9	77	0.0043	45.3
6	42	42	0.0	43.5	42	0.0	54.2	42	0.0	26.9	42	0.0	44.7
7	60	60	0.0	96.8	60	0.0	166.1	60	0.0	54.3	60	0.0	114.1
1-7	387	387	0.0	58.5				387	0.0	32.9	386	0.0	60.1
4-7	240				238	0.0	138.8				239	0.0	103.3

Table 12 shows that `LNS` is able to find the best-known solution of all but one TOP instance in a few minutes of computing time. Moreover, `LNS` is competitive with VOP15 on instances of families 4-7 but cannot find one of the best-known solutions achieved by DGM13 and KLC16.

6.3. Computational Results on Real-Life Instances

To illustrate the managerial relevance of the TOPO and the performance of the proposed solution approaches, real-life instances were provided by Geldmaat representing the cash replenishment problems they face in four major cities in the Netherlands (i.e., Almere, Amsterdam, Arnhem, and Tilburg). The service points represent ATMs that require replenishment and consumers represent the bank account holders that have to be served.

The instances of the four cities have the following number of ATMs/service points ($|\mathcal{S}|$) and consumers/bank account holders ($|\mathcal{C}|$): Almere $|\mathcal{S}| = 32$ and $|\mathcal{C}| = 112,803$; Amsterdam $|\mathcal{S}| = 100$ and $|\mathcal{C}| = 203,717$; Arnhem $|\mathcal{S}| = 33$ and $|\mathcal{C}| = 74,805$; Tilburg $|\mathcal{S}| = 35$ and $|\mathcal{C}| = 92,688$. For the cities of Almere, Arnhem and Tilburg, we generated two instances: the first with one vehicle and the second with two vehicles. For the city of Amsterdam, we generated four instances with one, two, three, and four vehicles. In total, we considered ten instances. Each instance features a maximum route duration equal to 480 minutes, which corresponds to the typical working day of eight hours. Bank account holders (identified by the postal code of residence) are considered served if there exists a replenished ATM within five kilometers from the residence postal code.

6.3.1. Results of the Exact Method Table 13 reports the results of BCP1 on the ten real-life instances. A time limit of one hour was used for each instance. For each instance (identified by the city and the number of vehicles), the table reports the best lower bound (blb) found by either BCP1 or LNS the upper bound at the root node ($\mathsf{ub_R}$) and the corresponding gap ($\mathsf{gap_R}$), the final upper bound ($\mathsf{ub_F}$) and the corresponding gap ($\mathsf{gap_F}$), the final lower bound ($\mathsf{lb_F}$) and the corresponding gap ($\mathsf{gap_{Op}}$) if the instance was not solved to optimality, the number of nodes of the search tree (nds), and the total solution time ($\mathsf{cpu_F}$).

Table 13 Results of BCP1 on real-life instances

City	$\lvert\mathcal{K}\rvert$	blb	$\mathsf{ub_R}$	$\mathsf{gap_R}$	$\mathsf{ub_F}$	$\mathsf{gap_F}$	$\mathsf{lb_F}$	$\mathsf{gap_{Op}}$	nds	$\mathsf{cpu_F}$
Almere	1	103,173	103,271	0.09	103,173	0.00	103,173	-	22	190.3
Almere	2	112,803	112,803	0.00	112,803	0.00	112,803	-	4	0.6
Amsterdam	1	116,611	116,611	0.00	116,611	0.00	116,611	-	1	146.0
Amsterdam	2	182,779	183,648	0.48	183,593	0.45	172,457	5.65	30	tl
Amsterdam	3	202,175	202,606	0.21	202,605	0.21	199,345	1.40	2	tl
Amsterdam	4	203,717	203,717	0.00	203,717	0.00	203,717	-	39	231.5
Arnhem	1	74,643	74,643	0.00	74,643	0.00	74,643	-	2	35.2
Arnhem	2	74,805	74,805	0.00	74,805	0.00	74,805	-	10	2.2
Tilburg	1	73,522	73,522	0.00	73,522	0.00	73,522	-	1	2.0
Tilburg	2	92,688	92,688	0.00	92,688	0.00	92,688	-	1	0.2

Table 13 shows that the six instances for the cities of Almere, Arnhem, and Tilburg and the two instances of Amsterdam with one and four vehicles can easily be solved to optimality. BCP1 cannot solve the Amsterdam instances with two and three vehicles, but managed to find upper bounds that are 0.45% and 0.21% from the best-known lower bound. We can also notice that the gap at the root node is never larger than 0.48%. Notice that all bank account holders can be served in Almere, Arnhem, and Tilburg with two vehicles and in Amsterdam with four vehicles.

6.3.2. Results of the LNS Table 14 summarizes the computational results of LNS on the ten real-life instances. The metaheuristic used the same parameter setting as in the experiments for the synthetic instances. Similarly, ten runs per instance were performed. For each instance, the table reports the worst ($\mathsf{lb_W}$), average ($\mathsf{lb_A}$), and best ($\mathsf{lb_B}$) lower bounds found along with their corresponding gaps computed with respect to the best-known upper bound available ($\mathsf{gap_{WDB}}$, $\mathsf{gap_{ADB}}$, $\mathsf{gap_{BDB}}$), the gap between $\mathsf{lb_B}$ and the best lower bound $\mathsf{lb_F}$ found by BCP1 ($\mathsf{gap_{BPB}}$), the best lower bound computed (blb), and the average solution time ($\mathsf{cpu_F}$) over the ten runs.

We can observe that LNS manages to find solutions that are within 0.74% from the best-known upper bound provided by BCP1 on all instances. Moreover, LNS can improve the final lower bound provided by BCP1 on the Amsterdam instances with two and three vehicles.

Table 14 Results of LNS on real-life instances

City	$\|\mathcal{K}\|$	$\mathsf{lb_W}$	$\mathsf{gap_{WDB}}$	$\mathsf{lb_A}$	$\mathsf{gap_{ADB}}$	$\mathsf{lb_B}$	$\mathsf{gap_{BDB}}$	$\mathsf{gap_{BPB}}$	blb	$\mathsf{cpu_F}$
Almere	1	101.919	1.33	102,785.9	0.47	103,097	0.17	0.07	103,173	26.8
Almere	2	112,803	0.00	112,803.0	0.00	112,803	0.00	0.00	112,803	0.0
Amsterdam	1	113,795	2.47	113,994.3	2.30	115,788	0.71	0.71	116,611	155.5
Amsterdam	2	181,431	1.22	181,800.7	1.02	182,779	0.48	-5.99	182,779	613.7
Amsterdam	3	201,947	0.33	202,068.5	0.27	202,175	0.21	-1.42	202,175	1435.1
Amsterdam	4	202,469	0.62	202,812.8	0.45	203,182	0.26	0.26	203,717	1207.1
Arnhem	1	74,480	0.22	74,563.3	0.11	74,635	0.01	0.01	74,643	15.2
Arnhem	2	74,805	0.00	74,805.0	0.00	74,805	0.00	0.00	74,805	0.0
Tilburg	1	71,146	3.34	72,172.2	1.87	72,980	0.74	0.74	73,522	10.0
Tilburg	2	91,220	1.61	92,033.1	0.71	92,625	0.07	0.07	92,688	24.2

6.3.3. Managerial Implications To illustrate the managerial relevance of our research, we conducted a further computational study to assess the impact of solving the ten real-life TOPO instances as TOP instances. We consider two intuitive approaches of solving TOPO instances as TOP instances. The first approach (hereafter TOP1) solves a TOP instance derived from the original TOPO instance where the profit of each service point/ATM is equal to the number of consumers/bank account holders that are within its service region; customers in overlapping zones can therefore be assigned to multiple service points. The second approach (hereafter TOP2) solves a TOP instance where each consumer/bank account holder is a-priori assigned to its closest ATM, and the profit of each ATM is then the number of bank account holders for which that ATM is the closest one. Therefore, TOP1 overestimates the profit of each ATM, and TOP2 underestimates it. Both for the solution of TOP1 and TOP2 the profit that will be collected in reality is obtained by a post-processing procedure.

Table 15 reports, for each of the ten real-life instances and each of the approaches (TOP1/ TOP2), the real value of the best solution found (blb) and its percentage deviation (Δ) w.r.t. the value of the best-known solution found solving directly the instance as TOPO instance (TOPO blb). For all cases, solutions are computed using `LNS`.

The most striking observation from Table 15 is that, on average, the solutions obtained by TOP1 and TOP2 serve respectively 11.8% and 16.8% less bank account holders than the solutions found by considering the problem as a TOPO. In other words, wrongfully assuming a TOPO problem as a TOP can lead to significantly lower profits or service levels in real-life applications.

Table 15 Comparison of the performance of TOPO, TOP1, and TOP2 on the real-life instances

		TOPO	TOP1		TOP2	
City	$\lvert\mathcal{K}\rvert$	blb	blb	Δ	blb	Δ
Almere	1	103,097	78,900	-23.5	84,735	-17.8
Almere	2	112,803	112,803	0.0	112,803	0.0
Amsterdam	1	115,788	90,110	-22.2	64,421	-44.4
Amsterdam	2	182,779	135,014	-26.1	109,805	-39.9
Amsterdam	3	202,175	173,114	-14.4	147,575	-27.0
Amsterdam	4	203,182	192,970	-5.0	183,633	-9.6
Arnhem	1	74,635	65,374	-12.4	64,846	-13.1
Arnhem	2	74,805	74,805	0.0	74,805	0.0
Tilburg	1	72,980	63,061	-13.6	62,469	-14.4
Tilburg	2	92,625	91,787	-0.9	91,175	-1.6
Avg				-11.8		-16.8

7. Conclusions and Future Research

Motivated by a real-life ATM cash replenishment problem encountered in the Netherlands, we have investigated a new generalization of the *Team Orienteering Problem* (TOP), called *TOP with Overlaps* (TOPO). We have proposed exact methods based on column generation for the exact solution of the problem, where our main contribution has been in exploiting the ng-path relaxation to obtain high-quality dual bounds for the problem. We have also proposed a metaheuristic based on Large Neighborhood Search. The performance of the proposed solution methods has been assessed by an extensive computational study on synthetic and real-life instances. The computational results have shown that the proposed methods can find high-quality bounds of all considered instances. We have also shown that the proposed solution methods are competitive with the state-of-the-art solution methods for the TOP. In particular, we could find the optimal solution of 96% of the well-known Chao instances and close 33 open instances. From a managerial standpoint, we have also shown that modeling a real-life cash replenishment problem as a TOPO provides significantly better solutions than solving it as a TOP and can help to strongly improve service levels.

References

Adulyasak Y, Cordeau JF, Jans R, 2012 *Optimization-based adaptive large neighborhood search for the production routing problem. Transportation Science* 48(1):20–45.

Archetti C, Feillet D, Hertz A, Speranza MG, 2009 *The capacitated team orienteering and profitable tour problems. Journal of the Operational Research Society* 60(6):831–842.

Archetti C, Hertz A, Speranza MG, 2007 *Metaheuristics for the team orienteering problem. Journal of Heuristics* 13(1):49–76.

Archetti C, Speranza MG, Vigo D, 2014 *Chapter 10: Vehicle routing problems with profits. Vehicle Routing: Problems, Methods, and Applications, Second Edition*, 273–297 (SIAM).

Baldacci R, Mingozzi A, Roberti R, 2011 *New route relaxation and pricing strategies for the vehicle routing problem. Operations Research* 59(5):1269–1283.

Bianchessi N, Mansini R, Speranza MG, 2018 *A branch-and-cut algorithm for the team orienteering problem. International Transactions in Operational Research* 25(2):627–635.

Bouly H, Dang DC, Moukrim A, 2010 *A memetic algorithm for the team orienteering problem. 4OR* 8(1):49–70.

Boussier S, Feillet D, Gendreau M, 2007 *An exact algorithm for team orienteering problems. 4OR* 5(3):211–230.

Butt SE, Cavalier TM, 1994 *A heuristic for the multiple tour maximum collection problem. Computers & Operations Research* 21(1):101–111.

Chao IM, Golden BL, Wasil EA, 1996 *The team orienteering problem. European journal of operational research* 88(3):464–474.

Dang DC, El-Hajj R, Moukrim A, 2013 *A branch-and-cut algorithm for solving the team orienteering problem. International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, 332–339 (Springer).

Dang DC, Guibadj RN, Moukrim A, 2011 *A PSO-based memetic algorithm for the team orienteering problem. Applications of Evolutionary Computation*, 471–480 (Berlin, Heidelberg: Springer Berlin Heidelberg).

Dang DC, Guibadj RN, Moukrim A, 2013 *An effective pso-inspired algorithm for the team orienteering problem. European Journal of Operational Research* 229(2):332–344.

Drexl M, 2007 *On some generalized routing problems*. Technical report, Deutsche Post Lehrstuhl für Optimierung von Distributionsnetzwerken (NN).

Dror M, 1994 *Note on the complexity of the shortest path models for column generation in vrptw. Operations Research* 42:977–978.

El-Hajj R, Dang DC, Moukrim A, 2016 *Solving the team orienteering problem with cutting planes. Computers & Operations Research* 74:21–30.

Emde S, Schneider M, 2018 *Just-in-time vehicle routing for in-house part feeding to assembly lines. Transportation Science* 52(3):657–672.

Gunawan A, Lau HC, Vansteenwegen P, 2016 *Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research* 255(2):315–332.

Hübner A, Ostermeier M, 2018 *A multi-compartment vehicle routing problem with loading and unloading costs. Transportation Science* .

Jepsen M, Petersen B, Spoorendonk S, Pisinger D, 2008 *Subset-row inequalities applied to the vehicle-routing problem with time windows. Operations Research* 56(2):497–511.

Joncour C, Michel S, Sadykov R, Sverdlov D, Vanderbeck F, 2010 *Column generation based primal heuristics. Electronic Notes in Discrete Mathematics* 36:695–702.

Ke L, Archetti C, Feng Z, 2008 *Ants can solve the team orienteering problem. Computers & Industrial Engineering* 54(3):648–665.

Ke L, Zhai L, Li J, Chan FT, 2016 *Pareto mimic algorithm: An approach to the team orienteering problem. Omega* 61:155–166.

Keshtkaran M, Ziarati K, Bettinelli A, Vigo D, 2016 *Enhanced exact solution methods for the team orienteering problem. International Journal of Production Research* 54(2):591–601.

Kim BI, Li H, Johnson AL, 2013 *An augmented large neighborhood search method for solving the team orienteering problem. Expert Systems with applications* 40(8):3065–3072.

Lin SW, 2013 *Solving the team orienteering problem using effective multi-start simulated annealing. Applied Soft Computing Journal* 13:1064–1073.

Masson R, Lehuédé F, Péton O, 2013 *An adaptive large neighborhood search for the pickup and delivery problem with transfers. Transportation Science* 47(3):344–355.

Pisinger D, Ropke S, 2019 *Large Neighborhood Search*, 99–127 (Cham: Springer International Publishing).

Poggi M, Viana H, Uchoa E, 2010 *The team orienteering problem: Formulations and branch-cut and price. OASIcs-OpenAccess Series in Informatics*, volume 14.

Righini G, Salani M, 2006 *Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optimization* 3(3):255–273.

Roberti R, Mingozzi A, 2014 *Dynamic ng-path relaxation for the delivery man problem. Transportation Science* 48(3):413–424.

Ropke S, Pisinger D, 2006 *An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation science* 40(4):455–472.

Shaw P, 1998 *Using constraint programming and local search methods to solve vehicle routing problems. International conference on principles and practice of constraint programming*, 417–431 (Springer).

Solomon MM, 1987 *Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations research* 35(2):254–265.

Souffriau W, Vansteenwegen P, Berghe GV, Van Oudheusden D, 2010 *A path relinking approach for the team orienteering problem. Computers & operations research* 37(11):1853–1859.

Tang H, Miller-Hooks E, 2005 *A tabu search heuristic for the team orienteering problem. Computers & Operations Research* 32(6):1379–1407.

Vansteenwegen P, Souffriau W, Berghe GV, Van Oudheusden D, 2009 *A guided local search metaheuristic for the team orienteering problem. European journal of operational research* 196(1):118–127.

Vansteenwegen P, Souffriau W, Van Oudheusden D, 2011 *The orienteering problem: A survey. European Journal of Operational Research* 209(1):1–10.

Vansteenwegen P, Van Oudheusden D, 2007 *The mobile tourist guide: An or opportunity. OR Insight* 20(3):21–27.

Vidal T, Maculan N, Ochi LS, Vaz Penna PH, 2015 *Large neighborhoods with implicit customer selection for vehicle routing problems with profits. Transportation Science* 50(2):720–734.

This page is intentionally blank. Proper e-companion title page, with INFORMS branding and exact metadata of the main paper, will be produced by the INFORMS office when the issue is being assembled.

Detailed Computational Results

Tables EC.1-EC.5 report the detailed computational results of `BCP1` on all 215 TOPO instances. Each table refers to the 43 instances with the same overlap, ranging from 10% to 50% as explained in Section 6. In each of these five tables, the columns report the instance name (Inst), the number of available vehicles ($|\mathcal{K}|$), the number of service points ($|\mathcal{S}|$), the total number of consumers ($|\mathcal{C}|$), the best lower bound (blb) found by either `BCP1` or `LNS`, the upper bound at the root node (ub_R), the final upper bound (ub_F), the final lower bound (lb_F) with the corresponding gaps (gap_R, gap_F, and gap, respectively), the number of nodes of the search tree (nds), and the total solution time (cpu_F). A time limit of one hour was imposed on each run. The name of each instance, $a.b.c.d$, indicates the family (a), the number of vehicles (b), the original TOP instance from the Chao set (c), and the overlap (d).

Tables EC.6-EC.10 report the detailed computational results of `LNS` on all 215 TOPO instances. In each of these five tables, the columns report the instance name (Inst), the number of available vehicles ($|\mathcal{K}|$), the number of service points ($|\mathcal{S}|$), the total number of consumers ($|\mathcal{C}|$), the worst (lb_W), average (lb_A), and best (lb_B) lower bounds found along with their corresponding gaps computed with respect to the best-known upper bound available ($\mathsf{gap}_\mathsf{WDB}$, $\mathsf{gap}_\mathsf{ADB}$, $\mathsf{gap}_\mathsf{BDB}$), the gap between lb_B and the best lower bound lb_F found by `BCP1` ($\mathsf{gap}_\mathsf{BPB}$), the best-known lower bound (blb), and the average solution time (cpu_F) over the ten runs. When the percentage gap between the best solution found by the `LNS` and the best primal bound found by `BCP1` ($\mathsf{gap}_\mathsf{BPB}$) is negative this mean that the best solution found by `LNS` is better than the best solution found by `BCP1`).

Tables EC.11-EC.14 report the detailed computational results of `BCP1` on the 240 Chao instances of families four to seven. In each of these four tables, the columns report the best known lower bound (blb) found by `BCP1` or already available in the literature, the upper bound at the root node (ub_R) and the corresponding gap (gap_R), the final upper bound (ub_F) and the corresponding gap (gap_F), the final lower bound (lb_F) and the corresponding gap (gap), the number of nodes of the search tree (nds), and the total solution time (cpu_F). A time limit of one hour was imposed on each run.

Tables EC.15-EC.18 report the detailed computational results of `LNS` on 240 Chao instances of families four to seven. The same columns used in Tables EC.6-EC.10 are displayed.

Table EC.1 Detailed computational results of BCP1 on TOPO instances with overlap 10%

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
1.2.j.10	2	30	285	141	141	0.00	141	0.00	141	0.00	1	0.1
1.2.n.10	2	30	285	240	240	0.00	240	0.00	240	0.00	1	0.3
1.2.r.10	2	30	285	280	280	0.00	280	0.00	280	0.00	1	0.6
1.3.p.10	3	30	285	221	221	0.00	221	0.00	221	0.00	1	0.1
1.3.q.10	3	30	285	234	234	0.00	234	0.00	234	0.00	1	0.3
1.3.r.10	3	30	285	251	251	0.00	251	0.00	251	0.00	1	0.3
3.2.l.10	2	31	800	599	605	1.00	599	0.00	599	0.00	5	1.3
3.2.p.10	2	31	800	721	733	1.66	721	0.00	721	0.00	109	27.1
3.2.t.10	2	31	800	800	800	0.00	800	0.00	800	0.00	3	1.0
3.3.s.10	3	31	800	720	738	2.50	720	0.00	720	0.00	9	5.8
3.3.t.10	3	31	800	760	762	0.26	760	0.00	760	0.00	3	5.4
4.2.f.10	2	98	1306	699	720	3.00	699	0.00	699	0.00	2257	1684.3
4.2.m.10	2	98	1306	1144	1145	0.09	1144	0.00	1144	0.00	7	455.1
4.2.t.10	2	98	1306	1306	1306	0.00	1306	0.00	1286	1.53	55	TL
4.3.k.10	3	98	1306	934	948	1.50	934	0.00	934	0.00	61	249.5
4.3.p.10	3	98	1306	1230	1230	0.00	1230	0.00	1230	0.00	5	187.8
4.3.t.10	3	98	1306	1306	1306	0.00	1306	0.00	1301	0.38	12	TL
4.4.o.10	4	98	1306	1067	1077	0.94	1067	0.00	1067	0.00	51	131.0
4.4.r.10	4	98	1306	1222	1240	1.47	1227	0.41	1222	0.00	210	TL
4.4.t.10	4	98	1306	1291	1298	0.54	1294	0.23	1288	0.23	65	TL
5.2.h.10	2	64	1680	446	446	0.00	446	0.00	446	0.00	1	0.1
5.2.q.10	2	64	1680	1240	1258	1.45	1240	0.00	1240	0.00	445	217.5
5.2.z.10	2	64	1680	1680	1680	0.00	1680	0.00	1680	0.00	90	208.2
5.3.l.10	3	64	1680	637	642	0.78	637	0.00	637	0.00	5	1.1
5.3.s.10	3	64	1680	1250	1259	0.72	1250	0.00	1250	0.00	35	31.1
5.3.z.10	3	64	1680	1651	1651	0.00	1651	0.00	1651	0.00	93	370.3
5.4.p.10	4	64	1680	820	835	1.83	820	0.00	820	0.00	15	3.5
5.4.u.10	4	64	1680	1337	1337	0.00	1337	0.00	1337	0.00	1	0.4
5.4.z.10	4	64	1680	1639	1639	0.00	1639	0.00	1639	0.00	2	6.6
6.2.f.10	2	62	1344	603	603	0.00	603	0.00	603	0.00	1	0.1
6.2.j.10	2	62	1344	973	973	0.00	973	0.00	973	0.00	1	0.4
6.2.n.10	2	62	1344	1266	1266	0.00	1266	0.00	1266	0.00	76	75.6
6.3.j.10	3	62	1344	849	849	0.00	849	0.00	849	0.00	1	0.1
6.3.l.10	3	62	1344	1027	1027	0.00	1027	0.00	1027	0.00	1	1.2
6.3.n.10	3	62	1344	1191	1192	0.08	1191	0.00	1191	0.00	9	11.8
6.4.n.10	4	62	1344	1090	1090	0.00	1090	0.00	1090	0.00	1	0.1
7.2.j.10	2	100	1458	667	667	0.00	667	0.00	667	0.00	1	14.3
7.2.o.10	2	100	1458	956	958	0.21	956	0.00	956	0.00	11	336.3
7.2.t.10	2	100	1458	1185	1186	0.08	1185	0.00	1185	0.00	6	944.2
7.3.o.10	3	100	1458	886	892	0.68	886	0.00	886	0.00	15	346.7
7.3.r.10	3	100	1458	1039	1047	0.77	1039	0.00	1039	0.00	19	2724.2
7.3.t.10	3	100	1458	1128	1138	0.89	1134	0.53	1111	1.51	4	TL
7.4.t.10	4	100	1458	1083	1085	0.18	1083	0.00	1083	0.00	5	96.2

Table EC.2 Detailed computational results of BCP1 on TOPO instances with overlap 20%

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
1.2.j.20	2	30	285	147	147	0.00	147	0.00	147	0.00	1	0.2
1.2.n.20	2	30	285	244	244	0.00	244	0.00	244	0.00	1	0.3
1.2.r.20	2	30	285	280	280	0.00	280	0.00	280	0.00	1	1.4
1.3.p.20	3	30	285	226	226	0.00	226	0.00	226	0.00	1	0.1
1.3.q.20	3	30	285	239	239	0.00	239	0.00	239	0.00	1	0.1
1.3.r.20	3	30	285	254	254	0.00	254	0.00	254	0.00	1	0.4
3.2.l.20	2	31	800	603	610	1.16	603	0.00	603	0.00	9	2.1
3.2.p.20	2	31	800	740	742	0.27	740	0.00	740	0.00	5	3.0
3.2.t.20	2	31	800	800	800	0.00	800	0.00	800	0.00	1	0.5
3.3.s.20	3	31	800	720	740	2.78	720	0.00	720	0.00	51	16.0
3.3.t.20	3	31	800	760	766	0.79	760	0.00	760	0.00	21	9.4
4.2.f.20	2	98	1306	709	727	2.54	709	0.00	709	0.00	2159	1328.8
4.2.m.20	2	98	1306	1147	1150	0.26	1147	0.00	1147	0.00	15	683.5
4.2.t.20	2	98	1306	1306	1306	0.00	1306	0.00	1306	0.00	71	585.8
4.3.k.20	3	98	1306	939	955	1.70	939	0.00	939	0.00	1157	3166.3
4.3.p.20	3	98	1306	1233	1237	0.32	1233	0.00	1233	0.00	9	276.5
4.3.t.20	3	98	1306	1306	1306	0.00	1306	0.00	1302	0.31	24	TL
4.4.o.20	4	98	1306	1071	1083	1.12	1071	0.00	1071	0.00	321	601.7
4.4.r.20	4	98	1306	1228	1246	1.47	1235	0.57	1228	0.00	205	TL
4.4.t.20	4	98	1306	1292	1300	0.62	1297	0.39	1285	0.54	75	TL
5.2.h.20	2	64	1680	467	467	0.00	467	0.00	467	0.00	1	0.1
5.2.q.20	2	64	1680	1266	1283	1.34	1266	0.00	1266	0.00	375	207.2
5.2.z.20	2	64	1680	1680	1680	0.00	1680	0.00	1680	0.00	733	2732.4
5.3.l.20	3	64	1680	665	671	0.90	665	0.00	665	0.00	5	1.3
5.3.s.20	3	64	1680	1277	1294	1.33	1277	0.00	1277	0.00	185	202.7
5.3.z.20	3	64	1680	1658	1658	0.00	1658	0.00	1658	0.00	124	641.1
5.4.p.20	4	64	1680	846	865	2.25	846	0.00	846	0.00	95	16.0
5.4.u.20	4	64	1680	1368	1368	0.00	1368	0.00	1368	0.00	1	0.5
5.4.z.20	4	64	1680	1644	1644	0.00	1644	0.00	1644	0.00	3	7.6
6.2.f.20	2	62	1344	619	619	0.00	619	0.00	619	0.00	1	0.1
6.2.j.20	2	62	1344	997	997	0.00	997	0.00	997	0.00	1	2.1
6.2.n.20	2	62	1344	1274	1275	0.08	1274	0.00	1274	0.00	88	125.6
6.3.j.20	3	62	1344	870	870	0.00	870	0.00	870	0.00	1	0.3
6.3.l.20	3	62	1344	1044	1049	0.48	1044	0.00	1044	0.00	17	6.4
6.3.n.20	3	62	1344	1210	1213	0.25	1210	0.00	1210	0.00	19	36.7
6.4.n.20	4	62	1344	1110	1110	0.00	1110	0.00	1110	0.00	1	0.1
7.2.j.20	2	100	1458	668	668	0.00	668	0.00	668	0.00	1	32.5
7.2.o.20	2	100	1458	960	961	0.10	960	0.00	960	0.00	5	261.8
7.2.t.20	2	100	1458	1187	1191	0.34	1187	0.00	1187	0.00	69	2340.2
7.3.o.20	3	100	1458	901	901	0.00	901	0.00	901	0.00	1	18.5
7.3.r.20	3	100	1458	1045	1054	0.86	1047	0.19	1044	0.10	9	TL
7.3.t.20	3	100	1458	1129	1145	1.42	1138	0.80	1117	1.06	14	TL
7.4.t.20	4	100	1458	1085	1087	0.18	1085	0.00	1085	0.00	5	110.8

Table EC.3 Detailed computational results of BCP1 on TOPO instances with overlap 30%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
1.2.j.30	2	30	285	156	156	0.00	156	0.00	156	0.00	1	0.2
1.2.n.30	2	30	285	252	252	0.00	252	0.00	252	0.00	1	0.2
1.2.r.30	2	30	285	282	282	0.00	282	0.00	282	0.00	1	2.6
1.3.p.30	3	30	285	233	233	0.00	233	0.00	233	0.00	1	0.1
1.3.q.30	3	30	285	247	247	0.00	247	0.00	247	0.00	1	0.1
1.3.r.30	3	30	285	258	258	0.00	258	0.00	258	0.00	1	0.3
3.2.l.30	2	31	800	609	619	1.64	609	0.00	609	0.00	53	4.7
3.2.p.30	2	31	800	748	753	0.67	748	0.00	748	0.00	3	1.6
3.2.t.30	2	31	800	800	800	0.00	800	0.00	800	0.00	1	0.4
3.3.s.30	3	31	800	720	741	2.92	720	0.00	720	0.00	149	43.4
3.3.t.30	3	31	800	760	771	1.45	760	0.00	760	0.00	59	19.8
4.2.f.30	2	98	1306	712	741	4.07	717	0.70	710	0.28	3187	TL
4.2.m.30	2	98	1306	1157	1162	0.43	1157	0.00	1157	0.00	60	1013.8
4.2.t.30	2	98	1306	1306	1306	0.00	1306	0.00	1306	0.00	118	1231.3
4.3.k.30	3	98	1306	949	965	1.69	949	0.00	949	0.00	537	1605.2
4.3.p.30	3	98	1306	1238	1245	0.57	1238	0.00	1238	0.00	81	2517.5
4.3.t.30	3	98	1306	1306	1306	0.00	1306	0.00	1306	0.00	19	327.7
4.4.o.30	4	98	1306	1087	1097	0.92	1087	0.00	1087	0.00	143	337.8
4.4.r.30	4	98	1306	1235	1254	1.54	1245	0.81	1234	0.08	267	TL
4.4.t.30	4	98	1306	1296	1302	0.46	1299	0.23	1291	0.39	82	TL
5.2.h.30	2	64	1680	479	479	0.00	479	0.00	479	0.00	1	0.1
5.2.q.30	2	64	1680	1283	1300	1.33	1283	0.00	1283	0.00	435	245.4
5.2.z.30	2	64	1680	1680	1680	0.00	1680	0.00	1660	1.19	560	TL
5.3.l.30	3	64	1680	684	688	0.58	684	0.00	684	0.00	3	0.8
5.3.s.30	3	64	1680	1292	1316	1.86	1292	0.00	1292	0.00	439	540.6
5.3.z.30	3	64	1680	1660	1660	0.00	1660	0.00	1660	0.00	19	42.9
5.4.p.30	4	64	1680	868	885	1.96	868	0.00	868	0.00	35	7.2
5.4.u.30	4	64	1680	1387	1387	0.00	1387	0.00	1387	0.00	1	0.5
5.4.z.30	4	64	1680	1646	1646	0.00	1646	0.00	1646	0.00	1	5.1
6.2.f.30	2	62	1344	641	641	0.00	641	0.00	641	0.00	1	0.1
6.2.j.30	2	62	1344	1019	1022	0.29	1019	0.00	1019	0.00	11	11.1
6.2.n.30	2	62	1344	1282	1282	0.00	1282	0.00	1282	0.00	26	48.0
6.3.j.30	3	62	1344	891	893	0.22	891	0.00	891	0.00	5	0.8
6.3.l.30	3	62	1344	1064	1071	0.66	1064	0.00	1064	0.00	21	8.6
6.3.n.30	3	62	1344	1222	1229	0.57	1222	0.00	1222	0.00	217	379.9
6.4.n.30	4	62	1344	1123	1123	0.00	1123	0.00	1123	0.00	1	0.4
7.2.j.30	2	100	1458	673	687	2.08	673	0.00	673	0.00	45	199.0
7.2.o.30	2	100	1458	965	972	0.73	965	0.00	965	0.00	247	2149.9
7.2.t.30	2	100	1458	1189	1194	0.42	1192	0.25	1188	0.08	23	TL
7.3.o.30	3	100	1458	911	911	0.00	911	0.00	911	0.00	1	43.2
7.3.r.30	3	100	1458	1050	1062	1.14	1053	0.29	1045	0.48	7	TL
7.3.t.30	3	100	1458	1131	1141	0.88	1138	0.62	1113	1.59	3	TL
7.4.t.30	4	100	1458	1093	1094	0.09	1093	0.00	1093	0.00	7	120.9

Table EC.4 Detailed computational results of BCP1 on TOPO instances with overlap 40%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
1.2.j.40	2	30	285	162	162	0.00	162	0.00	162	0.00	1	0.2
1.2.n.40	2	30	285	254	254	0.00	254	0.00	254	0.00	1	0.2
1.2.r.40	2	30	285	284	284	0.00	284	0.00	284	0.00	1	1.1
1.3.p.40	3	30	285	235	237	0.85	235	0.00	235	0.00	3	0.3
1.3.q.40	3	30	285	248	250	0.81	248	0.00	248	0.00	3	0.3
1.3.r.40	3	30	285	263	263	0.00	263	0.00	263	0.00	1	0.2
3.2.l.40	2	31	800	616	620	0.65	616	0.00	616	0.00	27	2.6
3.2.p.40	2	31	800	750	756	0.80	750	0.00	750	0.00	9	2.4
3.2.t.40	2	31	800	800	800	0.00	800	0.00	800	0.00	1	0.3
3.3.s.40	3	31	800	720	742	3.06	720	0.00	720	0.00	735	198.3
3.3.t.40	3	31	800	760	772	1.58	760	0.00	760	0.00	43	13.4
4.2.f.40	2	98	1306	732	749	2.32	732	0.00	732	0.00	407	282.3
4.2.m.40	2	98	1306	1164	1181	1.46	1173	0.77	1164	0.00	157	TL
4.2.t.40	2	98	1306	1306	1306	0.00	1306	0.00	1306	0.00	113	812.0
4.3.k.40	3	98	1306	970	989	1.96	970	0.00	970	0.00	628	2100.2
4.3.p.40	3	98	1306	1241	1255	1.13	1248	0.56	1236	0.40	50	TL
4.3.t.40	3	98	1306	1306	1306	0.00	1306	0.00	1305	0.08	49	TL
4.4.o.40	4	98	1306	1110	1126	1.44	1110	0.00	1110	0.00	801	1444.1
4.4.r.40	4	98	1306	1260	1271	0.87	1263	0.24	1258	0.16	181	TL
4.4.t.40	4	98	1306	1298	1305	0.54	1303	0.39	1293	0.39	76	TL
5.2.h.40	2	64	1680	500	500	0.00	500	0.00	500	0.00	1	0.1
5.2.q.40	2	64	1680	1313	1332	1.45	1313	0.00	1313	0.00	615	444.3
5.2.z.40	2	64	1680	1680	1680	0.00	1680	0.00	1680	0.00	45	81.6
5.3.l.40	3	64	1680	713	716	0.42	713	0.00	713	0.00	3	1.0
5.3.s.40	3	64	1680	1325	1356	2.34	1325	0.00	1325	0.00	1263	1613.4
5.3.z.40	3	64	1680	1662	1662	0.00	1662	0.00	1662	0.00	38	157.5
5.4.p.40	4	64	1680	909	919	1.10	909	0.00	909	0.00	11	3.0
5.4.u.40	4	64	1680	1416	1416	0.00	1416	0.00	1416	0.00	1	0.8
5.4.z.40	4	64	1680	1652	1652	0.00	1652	0.00	1652	0.00	1	5.1
6.2.f.40	2	62	1344	667	667	0.00	667	0.00	667	0.00	1	0.1
6.2.j.40	2	62	1344	1051	1064	1.24	1051	0.00	1051	0.00	229	96.2
6.2.n.40	2	62	1344	1288	1293	0.39	1290	0.16	1288	0.00	1947	TL
6.3.j.40	3	62	1344	923	928	0.54	923	0.00	923	0.00	13	2.0
6.3.l.40	3	62	1344	1096	1111	1.37	1096	0.00	1096	0.00	197	88.6
6.3.n.40	3	62	1344	1243	1252	0.72	1243	0.00	1243	0.00	331	465.6
6.4.n.40	4	62	1344	1153	1158	0.43	1153	0.00	1153	0.00	9	2.7
7.2.j.40	2	100	1458	691	711	2.89	691	0.00	691	0.00	101	311.2
7.2.o.40	2	100	1458	980	988	0.82	980	0.00	980	0.00	209	2216.8
7.2.t.40	2	100	1458	1198	1203	0.42	1201	0.25	1198	0.00	51	TL
7.3.o.40	3	100	1458	927	929	0.22	927	0.00	927	0.00	5	88.7
7.3.r.40	3	100	1458	1059	1072	1.23	1066	0.66	1053	0.57	7	TL
7.3.t.40	3	100	1458	1139	1153	1.23	1147	0.70	1122	1.49	8	TL
7.4.t.40	4	100	1458	1111	1113	0.18	1111	0.00	1111	0.00	3	83.6

Table EC.5 Detailed computational results of BCP1 on TOPO instances with overlap 50%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
1.2.j.50	2	30	285	168	168	0.00	168	0.00	168	0.00	1	0.1
1.2.n.50	2	30	285	255	255	0.00	255	0.00	255	0.00	1	0.5
1.2.r.50	2	30	285	285	285	0.00	285	0.00	285	0.00	2	0.7
1.3.p.50	3	30	285	237	240	1.27	237	0.00	237	0.00	9	0.6
1.3.q.50	3	30	285	254	255	0.39	254	0.00	254	0.00	3	0.3
1.3.r.50	3	30	285	265	265	0.00	265	0.00	265	0.00	1	0.2
3.2.l.50	2	31	800	621	625	0.64	621	0.00	621	0.00	9	1.4
3.2.p.50	2	31	800	752	759	0.93	752	0.00	752	0.00	5	1.8
3.2.t.50	2	31	800	800	800	0.00	800	0.00	800	0.00	8	1.0
3.3.s.50	3	31	800	730	744	1.92	730	0.00	730	0.00	777	206.4
3.3.t.50	3	31	800	760	775	1.97	760	0.00	760	0.00	125	32.6
4.2.f.50	2	98	1306	743	766	3.10	744	0.13	743	0.00	4325	TL
4.2.m.50	2	98	1306	1172	1202	2.56	1190	1.54	1168	0.34	141	TL
4.2.t.50	2	98	1306	1306	1306	0.00	1306	0.00	1306	0.00	112	537.1
4.3.k.50	3	98	1306	984	1012	2.85	990	0.61	983	0.10	651	TL
4.3.p.50	3	98	1306	1247	1267	1.60	1262	1.20	1239	0.64	59	TL
4.3.t.50	3	98	1306	1306	1306	0.00	1306	0.00	1303	0.23	42	TL
4.4.o.50	4	98	1306	1128	1140	1.06	1128	0.00	1128	0.00	3699	3397.7
4.4.r.50	4	98	1306	1268	1280	0.95	1274	0.47	1253	1.18	201	TL
4.4.t.50	4	98	1306	1302	1306	0.31	1306	0.31	1300	0.15	284	TL
5.2.h.50	2	64	1680	537	537	0.00	537	0.00	537	0.00	1	0.2
5.2.q.50	2	64	1680	1382	1397	1.09	1382	0.00	1382	0.00	97	150.1
5.2.z.50	2	64	1680	1680	1680	0.00	1680	0.00	1680	0.00	234	623.1
5.3.l.50	3	64	1680	763	765	0.26	763	0.00	763	0.00	3	0.7
5.3.s.50	3	64	1680	1376	1427	3.71	1388	0.87	1376	0.00	1023	TL
5.3.z.50	3	64	1680	1670	1670	0.00	1670	0.00	1670	0.00	93	856.2
5.4.p.50	4	64	1680	992	992	0.00	992	0.00	992	0.00	1	0.2
5.4.u.50	4	64	1680	1472	1473	0.07	1472	0.00	1472	0.00	3	3.2
5.4.z.50	4	64	1680	1661	1661	0.00	1661	0.00	1661	0.00	2	10.1
6.2.f.50	2	62	1344	683	683	0.00	683	0.00	683	0.00	1	0.1
6.2.j.50	2	62	1344	1081	1097	1.48	1081	0.00	1081	0.00	459	218.5
6.2.n.50	2	62	1344	1302	1304	0.15	1302	0.00	1302	0.00	278	397.3
6.3.j.50	3	62	1344	948	954	0.63	948	0.00	948	0.00	16	3.5
6.3.l.50	3	62	1344	1130	1146	1.42	1130	0.00	1130	0.00	143	70.9
6.3.n.50	3	62	1344	1265	1275	0.79	1265	0.00	1265	0.00	611	1344.0
6.4.n.50	4	62	1344	1179	1189	0.85	1179	0.00	1179	0.00	31	7.6
7.2.j.50	2	100	1458	701	722	3.00	701	0.00	701	0.00	167	680.6
7.2.o.50	2	100	1458	998	1001	0.30	998	0.00	998	0.00	29	1062.1
7.2.t.50	2	100	1458	1212	1212	0.00	1212	0.00	1212	0.00	2	654.4
7.3.o.50	3	100	1458	934	943	0.96	934	0.00	934	0.00	21	265.0
7.3.r.50	3	100	1458	1072	1085	1.21	1076	0.37	1072	0.00	16	TL
7.3.t.50	3	100	1458	1152	1160	0.69	1157	0.43	1134	1.56	7	TL
7.4.t.50	4	100	1458	1123	1124	0.09	1123	0.00	1123	0.00	3	27.2

Table EC.6 Detailed computational results of `LNS` on TOPO instances with overlap 10%

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	lb_W	gap_{WDB}	lb_A	gap_{ADB}	lb_B	gap_{BDB}	gap_{BPB}	blb	cpu_F
1.2.j.10	2	30	285	141	0.00	141.0	0.00	141	0.00	0.00	141	5.6
1.2.n.10	2	30	285	240	0.00	240.0	0.00	240	0.00	0.00	240	6.5
1.2.r.10	2	30	285	280	0.00	280.0	0.00	280	0.00	0.00	280	7.9
1.3.p.10	3	30	285	221	0.00	221.0	0.00	221	0.00	0.00	221	7.9
1.3.q.10	3	30	285	234	0.00	234.0	0.00	234	0.00	0.00	234	8.4
1.3.r.10	3	30	285	251	0.00	251.0	0.00	251	0.00	0.00	251	8.2
3.2.l.10	2	31	800	599	0.00	599.0	0.00	599	0.00	0.00	599	10.3
3.2.p.10	2	31	800	721	0.00	721.0	0.00	721	0.00	0.00	721	10.4
3.2.t.10	2	31	800	800	0.00	800.0	0.00	800	0.00	0.00	800	0.0
3.3.s.10	3	31	800	720	0.00	720.0	0.00	720	0.00	0.00	720	12.3
3.3.t.10	3	31	800	760	0.00	760.0	0.00	760	0.00	0.00	760	11.7
4.2.f.10	2	98	1306	690	1.30	694.5	0.65	698	0.14	0.14	699	214.8
4.2.m.10	2	98	1306	1137	0.62	1140.7	0.29	1143	0.09	0.09	1144	424.9
4.2.t.10	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	-1.53	1306	14.7
4.3.k.10	3	98	1306	927	0.76	932.6	0.15	934	0.00	0.00	934	358.8
4.3.p.10	3	98	1306	1224	0.49	1225.2	0.39	1228	0.16	0.16	1230	496.3
4.3.t.10	3	98	1306	1303	0.23	1304.9	0.08	1306	0.00	-0.38	1306	600.5
4.4.o.10	4	98	1306	1061	0.57	1064.0	0.28	1066	0.09	0.09	1067	445.6
4.4.r.10	4	98	1306	1214	1.07	1217.7	0.76	1222	0.41	0.00	1222	522.8
4.4.t.10	4	98	1306	1286	0.70	1288.5	0.50	1291	0.31	-0.23	1291	581.8
5.2.h.10	2	64	1680	446	0.00	446.0	0.00	446	0.00	0.00	446	68.0
5.2.q.10	2	64	1680	1231	0.73	1238.0	0.16	1240	0.00	0.00	1240	170.8
5.2.z.10	2	64	1680	1672	0.48	1672.9	0.42	1673	0.42	0.42	1680	198.0
5.3.l.10	3	64	1680	634	0.47	635.2	0.28	637	0.00	0.00	637	103.5
5.3.s.10	3	64	1680	1250	0.00	1250.0	0.00	1250	0.00	0.00	1250	164.9
5.3.z.10	3	64	1680	1651	0.00	1651.0	0.00	1651	0.00	0.00	1651	209.0
5.4.p.10	4	64	1680	820	0.00	820.0	0.00	820	0.00	0.00	820	140.0
5.4.u.10	4	64	1680	1337	0.00	1337.0	0.00	1337	0.00	0.00	1337	174.2
5.4.z.10	4	64	1680	1639	0.00	1639.0	0.00	1639	0.00	0.00	1639	201.6
6.2.f.10	2	62	1344	603	0.00	603.0	0.00	603	0.00	0.00	603	89.7
6.2.j.10	2	62	1344	970	0.31	971.5	0.15	973	0.00	0.00	973	136.4
6.2.n.10	2	62	1344	1241	2.01	1249.6	1.31	1257	0.72	0.72	1266	161.8
6.3.j.10	3	62	1344	849	0.00	849.0	0.00	849	0.00	0.00	849	140.3
6.3.l.10	3	62	1344	1027	0.00	1027.0	0.00	1027	0.00	0.00	1027	147.7
6.3.n.10	3	62	1344	1186	0.42	1190.0	0.08	1191	0.00	0.00	1191	160.6
6.4.n.10	4	62	1344	1090	0.00	1090.0	0.00	1090	0.00	0.00	1090	174.9
7.2.j.10	2	100	1458	667	0.00	667.0	0.00	667	0.00	0.00	667	186.8
7.2.o.10	2	100	1458	950	0.63	955.4	0.06	956	0.00	0.00	956	290.6
7.2.t.10	2	100	1458	1182	0.25	1183.8	0.10	1185	0.00	0.00	1185	415.2
7.3.o.10	3	100	1458	874	1.37	884.8	0.14	886	0.00	0.00	886	294.7
7.3.r.10	3	100	1458	1030	0.87	1035.2	0.37	1038	0.10	0.10	1039	386.9
7.3.t.10	3	100	1458	1120	1.25	1124.5	0.84	1128	0.53	-1.51	1128	433.2
7.4.t.10	4	100	1458	1082	0.09	1082.9	0.01	1083	0.00	0.00	1083	439.3

Table EC.7 Detailed computational results of LNS on TOPO instances with overlap 20%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	lb_W	gap_{WDB}	lb_A	gap_{ADB}	lb_B	gap_{BDB}	gap_{BPB}	blb	cpu_F
1.2.j.20	2	30	285	147	0.00	147.0	0.00	147	0.00	0.00	147	6.0
1.2.n.20	2	30	285	244	0.00	244.0	0.00	244	0.00	0.00	244	6.9
1.2.r.20	2	30	285	280	0.00	280.0	0.00	280	0.00	0.00	280	8.4
1.3.p.20	3	30	285	226	0.00	226.0	0.00	226	0.00	0.00	226	8.0
1.3.q.20	3	30	285	239	0.00	239.0	0.00	239	0.00	0.00	239	8.3
1.3.r.20	3	30	285	254	0.00	254.0	0.00	254	0.00	0.00	254	8.2
3.2.l.20	2	31	800	603	0.00	603.0	0.00	603	0.00	0.00	603	9.9
3.2.p.20	2	31	800	730	1.37	739.0	0.14	740	0.00	0.00	740	10.4
3.2.t.20	2	31	800	800	0.00	800.0	0.00	800	0.00	0.00	800	0.0
3.3.s.20	3	31	800	720	0.00	720.0	0.00	720	0.00	0.00	720	12.6
3.3.t.20	3	31	800	760	0.00	760.0	0.00	760	0.00	0.00	760	12.1
4.2.f.20	2	98	1306	705	0.57	706.3	0.38	708	0.14	0.14	709	222.4
4.2.m.20	2	98	1306	1141	0.53	1144.1	0.25	1146	0.09	0.09	1147	431.9
4.2.t.20	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	0.00	1306	11.3
4.3.k.20	3	98	1306	934	0.54	936.1	0.31	939	0.00	0.00	939	361.6
4.3.p.20	3	98	1306	1228	0.41	1229.2	0.31	1233	0.00	0.00	1233	507.8
4.3.t.20	3	98	1306	1305	0.08	1305.7	0.02	1306	0.00	-0.31	1306	414.1
4.4.o.20	4	98	1306	1067	0.37	1067.7	0.31	1068	0.28	0.28	1071	450.4
4.4.r.20	4	98	1306	1219	1.31	1224.0	0.90	1226	0.73	-0.08	1226	533.5
4.4.t.20	4	98	1306	1288	0.70	1290.1	0.53	1292	0.39	-0.54	1292	601.8
5.2.h.20	2	64	1680	467	0.00	467.0	0.00	467	0.00	0.00	467	67.5
5.2.q.20	2	64	1680	1259	0.56	1264.6	0.11	1266	0.00	0.00	1266	168.0
5.2.z.20	2	64	1680	1675	0.30	1676.3	0.22	1680	0.00	0.00	1680	195.3
5.3.l.20	3	64	1680	665	0.00	665.0	0.00	665	0.00	0.00	665	101.7
5.3.s.20	3	64	1680	1277	0.00	1277.0	0.00	1277	0.00	0.00	1277	168.6
5.3.z.20	3	64	1680	1657	0.06	1657.6	0.02	1658	0.00	0.00	1658	215.7
5.4.p.20	4	64	1680	846	0.00	846.0	0.00	846	0.00	0.00	846	139.0
5.4.u.20	4	64	1680	1368	0.00	1368.0	0.00	1368	0.00	0.00	1368	183.0
5.4.z.20	4	64	1680	1644	0.00	1644.0	0.00	1644	0.00	0.00	1644	208.1
6.2.f.20	2	62	1344	619	0.00	619.0	0.00	619	0.00	0.00	619	94.2
6.2.j.20	2	62	1344	992	0.50	995.0	0.20	997	0.00	0.00	997	147.3
6.2.n.20	2	62	1344	1258	1.27	1266.0	0.63	1274	0.00	0.00	1274	177.4
6.3.j.20	3	62	1344	870	0.00	870.0	0.00	870	0.00	0.00	870	148.8
6.3.l.20	3	62	1344	1044	0.00	1044.0	0.00	1044	0.00	0.00	1044	157.4
6.3.n.20	3	62	1344	1203	0.58	1206.2	0.32	1208	0.17	0.17	1210	174.2
6.4.n.20	4	62	1344	1110	0.00	1110.0	0.00	1110	0.00	0.00	1110	187.8
7.2.j.20	2	100	1458	668	0.00	668.0	0.00	668	0.00	0.00	668	202.6
7.2.o.20	2	100	1458	958	0.21	959.8	0.02	960	0.00	0.00	960	318.5
7.2.t.20	2	100	1458	1182	0.42	1185.3	0.14	1187	0.00	0.00	1187	456.1
7.3.o.20	3	100	1458	901	0.00	901.0	0.00	901	0.00	0.00	901	341.9
7.3.r.20	3	100	1458	1041	0.58	1041.9	0.49	1045	0.19	-0.10	1045	412.5
7.3.t.20	3	100	1458	1125	1.16	1127.8	0.90	1129	0.80	-1.06	1129	479.3
7.4.t.20	4	100	1458	1082	0.28	1082.7	0.21	1085	0.00	0.00	1085	484.7

Table EC.8 Detailed computational results of LNS on TOPO instances with overlap 30%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	lb_W	gap_{WDB}	lb_A	gap_{ADB}	lb_B	gap_{BDB}	gap_{BPB}	blb	cpu_F
1.2.j.30	2	30	285	156	0.00	156.0	0.00	156	0.00	0.00	156	6.0
1.2.n.30	2	30	285	252	0.00	252.0	0.00	252	0.00	0.00	252	6.9
1.2.r.30	2	30	285	282	0.00	282.0	0.00	282	0.00	0.00	282	8.7
1.3.p.30	3	30	285	233	0.00	233.0	0.00	233	0.00	0.00	233	8.5
1.3.q.30	3	30	285	247	0.00	247.0	0.00	247	0.00	0.00	247	8.8
1.3.r.30	3	30	285	258	0.00	258.0	0.00	258	0.00	0.00	258	8.8
3.2.l.30	2	31	800	609	0.00	609.0	0.00	609	0.00	0.00	609	10.4
3.2.p.30	2	31	800	748	0.00	748.0	0.00	748	0.00	0.00	748	10.8
3.2.t.30	2	31	800	800	0.00	800.0	0.00	800	0.00	0.00	800	0.0
3.3.s.30	3	31	800	720	0.00	720.0	0.00	720	0.00	0.00	720	13.0
3.3.t.30	3	31	800	760	0.00	760.0	0.00	760	0.00	0.00	760	12.5
4.2.f.30	2	98	1306	709	1.27	711.2	0.96	712	0.84	-0.70	712	233.0
4.2.m.30	2	98	1306	1152	0.43	1153.9	0.27	1155	0.17	0.17	1157	435.0
4.2.t.30	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	0.00	1306	8.4
4.3.k.30	3	98	1306	935	1.50	937.7	1.21	944	0.53	0.53	949	377.7
4.3.p.30	3	98	1306	1236	0.16	1237.0	0.08	1238	0.00	0.00	1238	511.4
4.3.t.30	3	98	1306	1306	0.00	1306.0	0.00	1306	0.00	0.00	1306	197.7
4.4.o.30	4	98	1306	1073	1.30	1075.0	1.12	1076	1.02	1.02	1087	462.4
4.4.r.30	4	98	1306	1230	1.22	1232.1	1.05	1235	0.81	-0.08	1235	561.8
4.4.t.30	4	98	1306	1292	0.54	1293.9	0.39	1296	0.23	-0.39	1296	620.7
5.2.h.30	2	64	1680	479	0.00	479.0	0.00	479	0.00	0.00	479	67.6
5.2.q.30	2	64	1680	1278	0.39	1281.5	0.12	1283	0.00	0.00	1283	166.7
5.2.z.30	2	64	1680	1678	0.12	1678.2	0.11	1680	0.00	-1.19	1680	192.5
5.3.l.30	3	64	1680	684	0.00	684.0	0.00	684	0.00	0.00	684	99.3
5.3.s.30	3	64	1680	1292	0.00	1292.0	0.00	1292	0.00	0.00	1292	168.8
5.3.z.30	3	64	1680	1657	0.18	1659.0	0.06	1660	0.00	0.00	1660	216.9
5.4.p.30	4	64	1680	860	0.93	866.2	0.21	868	0.00	0.00	868	137.8
5.4.u.30	4	64	1680	1387	0.00	1387.0	0.00	1387	0.00	0.00	1387	187.0
5.4.z.30	4	64	1680	1646	0.00	1646.0	0.00	1646	0.00	0.00	1646	213.3
6.2.f.30	2	62	1344	641	0.00	641.0	0.00	641	0.00	0.00	641	93.2
6.2.j.30	2	62	1344	1014	0.49	1015.4	0.35	1017	0.20	0.20	1019	144.7
6.2.n.30	2	62	1344	1268	1.10	1272.7	0.73	1277	0.39	0.39	1282	180.0
6.3.j.30	3	62	1344	891	0.00	891.0	0.00	891	0.00	0.00	891	148.7
6.3.l.30	3	62	1344	1061	0.28	1063.4	0.06	1064	0.00	0.00	1064	153.8
6.3.n.30	3	62	1344	1213	0.74	1215.4	0.54	1218	0.33	0.33	1222	178.5
6.4.n.30	4	62	1344	1123	0.00	1123.0	0.00	1123	0.00	0.00	1123	184.1
7.2.j.30	2	100	1458	673	0.00	673.0	0.00	673	0.00	0.00	673	219.2
7.2.o.30	2	100	1458	961	0.42	962.3	0.28	963	0.21	0.21	965	346.7
7.2.t.30	2	100	1458	1186	0.51	1187.7	0.36	1189	0.25	-0.08	1189	484.6
7.3.o.30	3	100	1458	911	0.00	911.0	0.00	911	0.00	0.00	911	349.6
7.3.r.30	3	100	1458	1045	0.96	1047.2	0.74	1050	0.48	-0.48	1050	436.7
7.3.t.30	3	100	1458	1127	1.06	1129.6	0.83	1131	0.71	-1.59	1131	492.7
7.4.t.30	4	100	1458	1090	0.28	1091.6	0.13	1093	0.00	0.00	1093	501.0

Table EC.9 Detailed computational results of LNS on TOPO instances with overlap 40%

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	lb_W	gap_{WDB}	lb_A	gap_{ADB}	lb_B	gap_{BDB}	gap_{BPB}	blb	cpu_F
1.2.j.40	2	30	285	162	0.00	162.0	0.00	162	0.00	0.00	162	6.3
1.2.n.40	2	30	285	254	0.00	254.0	0.00	254	0.00	0.00	254	7.6
1.2.r.40	2	30	285	284	0.00	284.0	0.00	284	0.00	0.00	284	9.3
1.3.p.40	3	30	285	235	0.00	235.0	0.00	235	0.00	0.00	235	9.0
1.3.q.40	3	30	285	248	0.00	248.0	0.00	248	0.00	0.00	248	9.5
1.3.r.40	3	30	285	263	0.00	263.0	0.00	263	0.00	0.00	263	9.4
3.2.l.40	2	31	800	615	0.16	615.6	0.06	616	0.00	0.00	616	11.0
3.2.p.40	2	31	800	750	0.00	750.0	0.00	750	0.00	0.00	750	12.1
3.2.t.40	2	31	800	800	0.00	800.0	0.00	800	0.00	0.00	800	0.0
3.3.s.40	3	31	800	720	0.00	720.0	0.00	720	0.00	0.00	720	14.2
3.3.t.40	3	31	800	760	0.00	760.0	0.00	760	0.00	0.00	760	13.7
4.2.f.40	2	98	1306	718	1.95	727.7	0.59	732	0.00	0.00	732	214.8
4.2.m.40	2	98	1306	1154	1.65	1156.2	1.45	1157	1.38	0.61	1164	451.9
4.2.t.40	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	0.00	1306	3.6
4.3.k.40	3	98	1306	945	2.65	961.6	0.87	970	0.00	0.00	970	359.5
4.3.p.40	3	98	1306	1238	0.89	1239.5	0.77	1241	0.64	-0.40	1241	528.8
4.3.t.40	3	98	1306	1306	0.00	1306.0	0.00	1306	0.00	-0.46	1306	129.7
4.4.o.40	4	98	1306	1098	1.09	1101.7	0.75	1103	0.63	0.63	1110	458.2
4.4.r.40	4	98	1306	1246	1.36	1252.7	0.82	1260	0.24	-0.16	1260	552.6
4.4.t.40	4	98	1306	1294	0.70	1296.0	0.54	1298	0.39	-0.39	1298	627.5
5.2.h.40	2	64	1680	500	0.00	500.0	0.00	500	0.00	0.00	500	68.9
5.2.q.40	2	64	1680	1302	0.84	1310.9	0.16	1313	0.00	0.00	1313	162.5
5.2.z.40	2	64	1680	1678	0.12	1678.0	0.12	1678	0.12	0.12	1680	215.4
5.3.l.40	3	64	1680	713	0.00	713.0	0.00	713	0.00	0.00	713	105.6
5.3.s.40	3	64	1680	1319	0.45	1322.7	0.17	1325	0.00	0.00	1325	166.4
5.3.z.40	3	64	1680	1660	0.12	1660.8	0.07	1662	0.00	0.00	1662	218.0
5.4.p.40	4	64	1680	899	1.11	905.6	0.38	909	0.00	0.00	909	137.1
5.4.u.40	4	64	1680	1416	0.00	1416.0	0.00	1416	0.00	0.00	1416	187.3
5.4.z.40	4	64	1680	1652	0.00	1652.0	0.00	1652	0.00	0.00	1652	211.4
6.2.f.40	2	62	1344	667	0.00	667.0	0.00	667	0.00	0.00	667	93.1
6.2.j.40	2	62	1344	1043	0.77	1046.0	0.48	1049	0.19	0.19	1051	140.6
6.2.n.40	2	62	1344	1278	0.94	1282.1	0.62	1286	0.31	0.16	1288	182.0
6.3.j.40	3	62	1344	920	0.33	922.1	0.10	923	0.00	0.00	923	144.7
6.3.l.40	3	62	1344	1091	0.46	1095.5	0.05	1096	0.00	0.00	1096	153.0
6.3.n.40	3	62	1344	1234	0.73	1235.7	0.59	1238	0.40	0.40	1243	177.4
6.4.n.40	4	62	1344	1153	0.00	1153.0	0.00	1153	0.00	0.00	1153	184.6
7.2.j.40	2	100	1458	687	0.58	690.6	0.06	691	0.00	0.00	691	233.3
7.2.o.40	2	100	1458	970	1.03	977.1	0.30	980	0.00	0.00	980	347.4
7.2.t.40	2	100	1458	1192	0.76	1193.7	0.61	1195	0.50	0.25	1198	486.4
7.3.o.40	3	100	1458	924	0.32	926.7	0.03	927	0.00	0.00	927	356.2
7.3.r.40	3	100	1458	1052	1.43	1054.9	1.15	1059	0.76	-0.57	1059	438.4
7.3.t.40	3	100	1458	1132	1.33	1134.7	1.08	1139	0.70	-1.49	1139	497.4
7.4.t.40	4	100	1458	1108	0.27	1110.4	0.05	1111	0.00	0.00	1111	488.6

Table EC.10 Detailed computational results of LNS on TOPO instances with overlap 50%

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	lb_W	$\mathsf{gap}_\mathsf{WDB}$	lb_A	$\mathsf{gap}_\mathsf{ADB}$	lb_B	$\mathsf{gap}_\mathsf{BDB}$	$\mathsf{gap}_\mathsf{BPB}$	blb	cpu_F
1.2.j.50	2	30	285	168	0.00	168.0	0.00	168	0.00	0.00	168	6.5
1.2.n.50	2	30	285	255	0.00	255.0	0.00	255	0.00	0.00	255	7.8
1.2.r.50	2	30	285	285	0.00	285.0	0.00	285	0.00	0.00	285	0.1
1.3.p.50	3	30	285	237	0.00	237.0	0.00	237	0.00	0.00	237	9.3
1.3.q.50	3	30	285	254	0.00	254.0	0.00	254	0.00	0.00	254	9.8
1.3.r.50	3	30	285	265	0.00	265.0	0.00	265	0.00	0.00	265	9.7
3.2.l.50	2	31	800	619	0.32	620.6	0.06	621	0.00	0.00	621	11.7
3.2.p.50	2	31	800	752	0.00	752.0	0.00	752	0.00	0.00	752	12.2
3.2.t.50	2	31	800	800	0.00	800.0	0.00	800	0.00	0.00	800	0.0
3.3.s.50	3	31	800	720	1.39	725.6	0.61	730	0.00	0.00	730	14.6
3.3.t.50	3	31	800	760	0.00	760.0	0.00	760	0.00	0.00	760	14.2
4.2.f.50	2	98	1306	736	1.22	737.5	1.02	739	0.81	0.54	743	220.5
4.2.m.50	2	98	1306	1166	2.14	1167.3	2.03	1172	1.62	-1.28	1172	462.1
4.2.t.50	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	0.00	1306	2.0
4.3.k.50	3	98	1306	969	2.27	978.6	1.27	984	0.71	-0.10	984	370.2
4.3.p.50	3	98	1306	1245	1.37	1245.9	1.29	1247	1.20	-0.64	1247	540.8
4.3.t.50	3	98	1306	1306	0.00	1306.0	0.00	1306	0.00	-0.23	1306	35.0
4.4.o.50	4	98	1306	1116	1.16	1120.8	0.73	1127	0.18	0.09	1128	468.3
4.4.r.50	4	98	1306	1254	1.59	1262.1	0.94	1268	0.47	-1.18	1268	566.2
4.4.t.50	4	98	1306	1300	0.46	1300.8	0.40	1302	0.31	-0.15	1302	658.1
5.2.h.50	2	64	1680	537	0.00	537.0	0.00	537	0.00	0.00	537	79.2
5.2.q.50	2	64	1680	1373	0.66	1377.8	0.30	1382	0.00	0.00	1382	191.3
5.2.z.50	2	64	1680	1679	0.06	1679.1	0.05	1680	0.00	0.00	1680	250.8
5.3.l.50	3	64	1680	759	0.53	760.8	0.29	763	0.00	0.00	763	118.0
5.3.s.50	3	64	1680	1374	1.09	1375.5	0.98	1376	0.94	0.00	1376	207.9
5.3.z.50	3	64	1680	1669	0.06	1669.7	0.02	1670	0.00	0.00	1670	262.7
5.4.p.50	4	64	1680	992	0.00	992.0	0.00	992	0.00	0.00	992	152.5
5.4.u.50	4	64	1680	1472	0.00	1472.0	0.00	1472	0.00	0.00	1472	234.7
5.4.z.50	4	64	1680	1661	0.00	1661.0	0.00	1661	0.00	0.00	1661	260.7
6.2.f.50	2	62	1344	683	0.00	683.0	0.00	683	0.00	0.00	683	94.3
6.2.j.50	2	62	1344	1072	0.84	1076.4	0.43	1080	0.09	0.09	1081	138.3
6.2.n.50	2	62	1344	1295	0.54	1298.3	0.28	1301	0.08	0.08	1302	185.1
6.3.j.50	3	62	1344	947	0.11	947.8	0.02	948	0.00	0.00	948	144.9
6.3.l.50	3	62	1344	1129	0.09	1129.9	0.01	1130	0.00	0.00	1130	156.4
6.3.n.50	3	62	1344	1259	0.48	1261.8	0.25	1264	0.08	0.08	1265	179.6
6.4.n.50	4	62	1344	1177	0.17	1178.5	0.04	1179	0.00	0.00	1179	187.5
7.2.j.50	2	100	1458	692	1.30	696.6	0.63	701	0.00	0.00	701	221.5
7.2.o.50	2	100	1458	996	0.20	997.1	0.09	998	0.00	0.00	998	343.7
7.2.t.50	2	100	1458	1197	1.25	1200.6	0.95	1204	0.66	0.66	1212	492.7
7.3.o.50	3	100	1458	932	0.21	933.4	0.06	934	0.00	0.00	934	357.3
7.3.r.50	3	100	1458	1065	1.13	1066.4	0.99	1067	0.94	0.47	1072	448.8
7.3.t.50	3	100	1458	1143	1.22	1147.7	0.81	1152	0.43	-1.56	1152	504.9
7.4.t.50	4	100	1458	1123	0.00	1123.0	0.00	1123	0.00	0.00	1123	464.0

Table EC.11 Detailed computational results of BCP1 on TOP instances of family 4

Inst	$\|\mathcal{K}\|$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
p4.2.a	2	206	206	0.00	206	0.00	206	0.00	1	0.1
p4.2.b	2	341	341	0.00	341	0.00	341	0.00	1	0.1
p4.2.c	2	452	458	1.33	452	0.00	452	0.00	5	1.0
p4.2.d	2	531	535	0.85	531	0.00	531	0.00	11	5.6
p4.2.e	2	618	623	0.93	618	0.00	618	0.00	11	13.3
p4.2.f	2	687	695	1.30	687	0.00	687	0.00	234	229.3
p4.2.g	2	757	770	1.81	757	0.00	757	0.00	524	787.8
p4.2.h	2	835	843	1.01	835	0.00	835	0.00	81	218.1
p4.2.i	2	918	918	0.00	918	0.00	918	0.00	1	16.2
p4.2.j	2	965	968	0.35	965	0.00	965	0.00	41	344.3
p4.2.k	2	1022	1028	0.62	1022	0.00	1022	0.00	155	1169.9
p4.2.l	2	1074	1080	0.56	1074	0.00	1074	0.00	47	1893.1
p4.2.m	2	1132	1132	0.00	1132	0.00	1132	0.00	1	331.8
p4.2.n	2	1174	1179	0.48	1174	0.00	1174	0.00	89	3584.4
p4.2.o	2	1218	1221	0.25	1218	0.00	1218	0.00	19	1592.0
p4.2.p	2	1242	1250	0.67	1248	0.48	1230	0.97	56	TL
p4.2.q	2	1268	1275	0.61	1273	0.39	1253	1.18	29	TL
p4.2.r	2	1292	1293	0.13	1293	0.08	1272	1.55	26	TL
p4.2.s	2	1304	1305	0.09	1304	0.00	1286	1.38	94	TL
p4.2.t	2	1306	1306	0.00	1306	0.00	1306	0.00	101	4569.8
p4.3.a	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p4.3.b	3	38	38	0.00	38	0.00	38	0.00	1	0.0
p4.3.c	3	193	193	0.00	193	0.00	193	0.00	1	0.1
p4.3.d	3	335	335	0.18	335	0.00	335	0.00	1	0.2
p4.3.e	3	468	468	0.00	468	0.00	468	0.00	1	0.2
p4.3.f	3	579	582	0.62	579	0.00	579	0.00	3	1.7
p4.3.g	3	653	654	0.15	653	0.00	653	0.00	3	2.5
p4.3.h	3	729	732	0.54	729	0.00	729	0.00	16	15.9
p4.3.i	3	809	810	0.23	809	0.00	809	0.00	3	5.4
p4.3.j	3	861	870	1.08	861	0.00	861	0.00	71	223.1
p4.3.k	3	919	928	1.04	919	0.00	919	0.00	19	91.2
p4.3.l	3	979	994	1.58	979	0.00	979	0.00	203	1918.6
p4.3.m	3	1063	1063	0.03	1063	0.00	1063	0.00	2	46.0
p4.3.n	3	1121	1124	0.33	1121	0.00	1121	0.00	7	200.0
p4.3.o	3	1172	1178	0.52	1172	0.00	1172	0.00	15	310.1
p4.3.p	3	1222	1223	0.10	1222	0.00	1222	0.00	3	154.2
p4.3.q	3	1253	1255	0.23	1253	0.00	1253	0.00	8	479.8
p4.3.r	3	1273	1281	0.65	1277	0.31	1273	0.00	39	TL
p4.3.s	3	1295	1298	0.26	1296	0.08	1294	0.08	24	TL
p4.3.t	3	1305	1306	0.08	1305	0.00	1302	0.23	15	TL
p4.4.a	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p4.4.b	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p4.4.c	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p4.4.d	4	38	38	0.00	38	0.00	38	0.00	1	0.0
p4.4.e	4	183	183	0.00	183	0.00	183	0.00	1	0.1
p4.4.f	4	324	324	0.00	324	0.00	324	0.00	1	0.1
p4.4.g	4	461	461	0.11	461	0.00	461	0.00	1	0.2
p4.4.h	4	571	571	0.00	571	0.00	571	0.00	1	0.2
p4.4.i	4	657	664	1.13	657	0.00	657	0.00	5	2.0
p4.4.j	4	732	739	0.98	732	0.00	732	0.00	16	8.7
p4.4.k	4	821	829	1.06	821	0.00	821	0.00	21	17.0
p4.4.l	4	880	891	1.30	880	0.00	880	0.00	43	58.2
p4.4.m	4	919	942	2.51	921	0.22	919	0.00	2824	TL
p4.4.n	4	976	997	2.22	976	0.00	976	0.00	1402	3284.8
p4.4.o	4	1061	1067	0.63	1061	0.00	1061	0.00	5	24.4
p4.4.p	4	1124	1128	0.38	1124	0.00	1124	0.00	13	76.2
p4.4.q	4	1161	1179	1.58	1161	0.00	1161	0.00	253	3263.0
p4.4.r	4	1216	1231	1.28	1216	0.00	1215	0.08	441	TL
p4.4.s	4	1260	1271	0.91	1262	0.16	1254	0.48	118	TL
p4.4.t	4	1285	1295	0.85	1290	0.39	1278	0.54	76	TL
Avg				0.56		0.04		0.11		509.0

Table EC.12 Detailed computational results of BCP1 on TOP instances of family 5

Inst	$\|\mathcal{K}\|$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
p5.2.a	2	0	0	0.00	0	0.00	0	0.00	1	0.0
p5.2.b	2	20	20	0.00	20	0.00	20	0.00	1	0.0
p5.2.c	2	50	50	0.00	50	0.00	50	0.00	1	0.0
p5.2.d	2	80	80	0.00	80	0.00	80	0.00	1	0.1
p5.2.e	2	180	180	0.00	180	0.00	180	0.00	1	0.0
p5.2.f	2	240	240	0.00	240	0.00	240	0.00	1	0.0
p5.2.g	2	320	320	0.00	320	0.00	320	0.00	1	0.1
p5.2.h	2	410	410	0.00	410	0.00	410	0.00	1	0.1
p5.2.i	2	480	480	0.00	480	0.00	480	0.00	1	0.5
p5.2.j	2	580	580	0.00	580	0.00	580	0.00	1	0.2
p5.2.k	2	670	670	0.00	670	0.00	670	0.00	1	0.3
p5.2.l	2	800	800	0.00	800	0.00	800	0.00	1	0.2
p5.2.m	2	860	860	0.00	860	0.00	860	0.00	1	0.8
p5.2.n	2	925	930	0.54	925	0.00	925	0.00	4	3.7
p5.2.o	2	1020	1030	0.98	1020	0.00	1020	0.00	7	2.5
p5.2.p	2	1150	1150	0.00	1150	0.00	1150	0.00	1	0.4
p5.2.q	2	1195	1210	1.26	1195	0.00	1195	0.00	345	146.4
p5.2.r	2	1260	1260	0.00	1260	0.00	1260	0.00	4	4.6
p5.2.s	2	1340	1340	0.00	1340	0.00	1340	0.00	1	4.7
p5.2.t	2	1400	1400	0.00	1400	0.00	1400	0.00	10	14.5
p5.2.u	2	1460	1460	0.00	1460	0.00	1460	0.00	3	12.6
p5.2.v	2	1505	1510	0.33	1505	0.00	1505	0.00	679	1382.2
p5.2.w	2	1565	1570	0.32	1565	0.00	1565	0.00	220	533.2
p5.2.x	2	1610	1610	0.00	1610	0.00	1610	0.00	26	43.9
p5.2.y	2	1645	1650	0.30	1650	0.30	1635	0.61	754	TL
p5.2.z	2	1680	1680	0.00	1680	0.00	1680	0.00	1085	2556.6
p5.3.a	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p5.3.b	3	15	15	0.00	15	0.00	15	0.00	1	0.0
p5.3.c	3	20	20	0.00	20	0.00	20	0.00	1	0.0
p5.3.d	3	60	60	0.00	60	0.00	60	0.00	1	0.0
p5.3.e	3	95	95	0.00	95	0.00	95	0.00	1	0.1
p5.3.f	3	110	110	0.00	110	0.00	110	0.00	1	0.0
p5.3.g	3	185	185	0.00	185	0.00	185	0.00	1	0.0
p5.3.h	3	260	260	0.00	260	0.00	260	0.00	1	0.1
p5.3.i	3	335	335	0.00	335	0.00	335	0.00	1	0.3
p5.3.j	3	470	470	0.00	470	0.00	470	0.00	1	0.0
p5.3.k	3	495	495	0.00	495	0.00	495	0.00	1	0.1
p5.3.l	3	595	604	1.54	595	0.00	595	0.00	5	0.6
p5.3.m	3	650	650	0.00	650	0.00	650	0.00	1	0.3
p5.3.n	3	755	755	0.00	755	0.00	755	0.00	1	0.7
p5.3.o	3	870	870	0.00	870	0.00	870	0.00	1	0.2
p5.3.p	3	990	990	0.00	990	0.00	990	0.00	1	0.2
p5.3.q	3	1070	1090	1.87	1070	0.00	1070	0.00	17	8.3
p5.3.r	3	1125	1155	2.67	1125	0.00	1125	0.00	139	117.1
p5.3.s	3	1190	1200	0.84	1190	0.00	1190	0.00	55	46.9
p5.3.t	3	1260	1270	0.79	1260	0.00	1260	0.00	53	63.4
p5.3.u	3	1345	1350	0.37	1345	0.00	1345	0.00	13	22.9
p5.3.v	3	1425	1430	0.35	1425	0.00	1425	0.00	35	54.3
p5.3.w	3	1485	1515	2.02	1493	0.54	1480	0.34	942	TL
p5.3.x	3	1555	1582	1.78	1555	0.00	1555	0.00	969	4416.3
p5.3.y	3	1595	1610	0.94	1604	0.56	1595	0.00	558	TL
p5.3.z	3	1635	1635	0.00	1635	0.00	1635	0.00	43	80.7
p5.4.a	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p5.4.b	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p5.4.c	4	20	20	0.00	20	0.00	20	0.00	1	0.0
p5.4.d	4	20	20	0.00	20	0.00	20	0.00	1	0.0
p5.4.e	4	20	20	0.00	20	0.00	20	0.00	1	0.0
p5.4.f	4	80	80	0.00	80	0.00	80	0.00	1	0.0
p5.4.g	4	140	140	0.00	140	0.00	140	0.00	1	0.0
p5.4.h	4	140	140	0.00	140	0.00	140	0.00	1	0.1
p5.4.i	4	240	240	0.00	240	0.00	240	0.00	1	0.0
p5.4.j	4	340	340	0.00	340	0.00	340	0.00	1	0.0
p5.4.k	4	340	340	0.00	340	0.00	340	0.00	1	0.1
p5.4.l	4	430	430	0.00	430	0.00	430	0.00	1	0.2
p5.4.m	4	555	555	0.00	555	0.00	555	0.00	1	0.1
p5.4.n	4	620	620	0.00	620	0.00	620	0.00	1	0.1
p5.4.o	4	690	690	0.00	690	0.00	690	0.00	1	0.3
p5.4.p	4	765	790	3.27	765	0.00	765	0.00	99	15.0
p5.4.q	4	860	860	0.00	860	0.00	860	0.00	1	0.4
p5.4.r	4	960	960	0.00	960	0.00	960	0.00	1	0.8
p5.4.s	4	1030	1050	1.94	1030	0.00	1030	0.00	127	34.7
p5.4.t	4	1160	1160	0.00	1160	0.00	1160	0.00	1	0.3
p5.4.u	4	1300	1300	0.00	1300	0.00	1300	0.00	1	0.2
p5.4.v	4	1320	1320	0.00	1320	0.00	1320	0.00	1	0.3
p5.4.w	4	1390	1415	1.80	1390	0.00	1390	0.00	51	34.6
p5.4.x	4	1450	1486	2.53	1450	0.00	1450	0.00	511	656.0
p5.4.y	4	1520	1520	0.00	1520	0.00	1520	0.00	14	12.8
p5.4.z	4	1620	1620	0.00	1620	0.00	1620	0.00	1	1.1
Avg				0.34		0.02		0.01		137.0

Table EC.13 Detailed computational results of BCP1 on TOP instances of family 6

Inst	$\lvert\mathcal{K}\rvert$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
p6.2.a	2	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.2.b	2	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.2.c	2	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.2.d	2	192	192	0.00	192	0.00	192	0.00	1	0.0
p6.2.e	2	360	360	0.00	360	0.00	360	0.00	1	0.0
p6.2.f	2	588	588	0.00	588	0.00	588	0.00	1	0.1
p6.2.g	2	660	660	0.00	660	0.00	660	0.00	1	0.1
p6.2.h	2	780	780	0.00	780	0.00	780	0.00	1	0.1
p6.2.i	2	888	888	0.00	888	0.00	888	0.00	1	1.6
p6.2.j	2	948	948	0.00	948	0.00	948	0.00	1	0.9
p6.2.k	2	1032	1032	0.00	1032	0.00	1032	0.00	2	2.6
p6.2.l	2	1116	1116	0.00	1116	0.00	1116	0.00	1	2.2
p6.2.m	2	1188	1188	0.00	1188	0.00	1188	0.00	134	104.4
p6.2.n	2	1260	1260	0.00	1260	0.00	1260	0.00	28	20.0
p6.3.a	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.b	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.c	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.d	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.e	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.f	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.3.g	3	282	282	0.00	282	0.00	282	0.00	1	0.0
p6.3.h	3	444	444	0.00	444	0.00	444	0.00	2	0.2
p6.3.i	3	642	642	0.00	642	0.00	642	0.00	1	0.1
p6.3.j	3	828	828	0.00	828	0.00	828	0.00	1	0.1
p6.3.k	3	894	936	4.70	894	0.00	894	0.00	615	102.7
p6.3.l	3	1002	1014	1.20	1002	0.00	1002	0.00	11	3.6
p6.3.m	3	1080	1104	2.22	1080	0.00	1080	0.00	487	252.4
p6.3.n	3	1170	1170	0.00	1170	0.00	1170	0.00	10	5.6
p6.4.a	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.b	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.c	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.d	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.e	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.f	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.g	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.h	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.i	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p6.4.j	4	366	366	0.00	366	0.00	366	0.00	1	0.0
p6.4.k	4	528	528	0.00	528	0.00	528	0.00	6	0.5
p6.4.l	4	696	708	1.72	696	0.00	696	0.00	49	4.0
p6.4.m	4	912	940	3.13	912	0.00	912	0.00	11	1.3
p6.4.n	4	1068	1068	0.00	1068	0.00	1068	0.00	1	0.1
Avg				0.31		0.00		0.00		12.0

Table EC.14 Detailed computational results of BCP1 on TOP instances of family 7

Inst	$\lvert\mathcal{K}\rvert$	blb	ub_R	gap_R	ub_F	gap_F	lb_F	gap	nds	cpu_F
p7.2.a	2	30	30	0.00	30	0.00	30	0.00	1	0.0
p7.2.b	2	64	64	0.00	64	0.00	64	0.00	1	0.0
p7.2.c	2	101	101	0.00	101	0.00	101	0.00	1	0.1
p7.2.d	2	190	190	0.00	190	0.00	190	0.00	1	0.1
p7.2.e	2	290	290	0.00	290	0.00	290	0.00	1	0.4
p7.2.f	2	387	387	0.00	387	0.00	387	0.00	1	0.8
p7.2.g	2	459	459	0.00	459	0.00	459	0.00	1	1.8
p7.2.h	2	521	523	0.51	521	0.00	521	0.00	54	64.1
p7.2.i	2	580	585	0.86	580	0.00	580	0.00	18	47.8
p7.2.j	2	646	648	0.33	646	0.00	646	0.00	9	43.5
p7.2.k	2	705	706	0.20	705	0.00	705	0.00	3	81.4
p7.2.l	2	767	767	0.06	767	0.00	767	0.00	1	81.6
p7.2.m	2	827	831	0.48	827	0.00	827	0.00	19	357.0
p7.2.n	2	888	892	0.49	888	0.00	888	0.00	3	206.6
p7.2.o	2	945	948	0.37	945	0.00	945	0.00	17	325.3
p7.2.p	2	1002	1002	0.00	1002	0.00	1002	0.00	1	130.8
p7.2.q	2	1044	1048	0.48	1044	0.00	1044	0.00	99	3198.6
p7.2.r	2	1094	1094	0.02	1094	0.00	1094	0.00	1	265.3
p7.2.s	2	1136	1138	0.18	1136	0.00	1136	0.00	12	976.9
p7.2.t	2	1179	1179	0.05	1179	0.00	1179	0.00	1	268.0
p7.3.a	3	0	0	0.00	0	0.00	0	0.00	1	0.0
p7.3.b	3	46	46	0.00	46	0.00	46	0.00	1	0.0
p7.3.c	3	79	79	0.00	79	0.00	79	0.00	1	0.0
p7.3.d	3	117	117	0.00	117	0.00	117	0.00	1	0.0
p7.3.e	3	175	175	0.00	175	0.00	175	0.00	1	0.1
p7.3.f	3	247	247	0.00	247	0.00	247	0.00	1	0.1
p7.3.g	3	344	344	0.00	344	0.00	344	0.00	1	0.2
p7.3.h	3	425	427	0.59	425	0.00	425	0.00	3	1.4
p7.3.i	3	487	494	1.62	487	0.00	487	0.00	17	12.9
p7.3.j	3	564	567	0.62	564	0.00	564	0.00	3	5.8
p7.3.k	3	633	633	0.00	633	0.00	633	0.00	1	10.7
p7.3.l	3	684	689	0.77	684	0.00	684	0.00	11	119.2
p7.3.m	3	762	762	0.08	762	0.00	762	0.00	1	11.9
p7.3.n	3	820	820	0.00	820	0.00	820	0.00	1	18.5
p7.3.o	3	874	875	0.21	874	0.00	874	0.00	3	82.1
p7.3.p	3	929	933	0.46	929	0.00	929	0.00	13	312.8
p7.3.q	3	987	987	0.00	987	0.00	987	0.00	1	93.7
p7.3.r	3	1026	1036	1.04	1028	0.19	1022	0.39	9	TL
p7.3.s	3	1081	1086	0.53	1081	0.00	1081	0.00	14	2466.6
p7.3.t	3	1120	1128	0.74	1122	0.18	1117	0.27	9	TL
p7.4.a	4	0	0	0.00	0	0.00	0	0.00	1	0.0
p7.4.b	4	30	30	0.00	30	0.00	30	0.00	1	0.0
p7.4.c	4	46	46	0.00	46	0.00	46	0.00	1	0.0
p7.4.d	4	79	79	0.00	79	0.00	79	0.00	1	0.0
p7.4.e	4	123	123	0.00	123	0.00	123	0.00	1	0.0
p7.4.f	4	164	164	0.00	164	0.00	164	0.00	1	0.0
p7.4.g	4	217	217	0.00	217	0.00	217	0.00	1	0.1
p7.4.h	4	285	285	0.00	285	0.00	285	0.00	1	0.2
p7.4.i	4	366	366	0.00	366	0.00	366	0.00	1	0.2
p7.4.j	4	462	462	0.03	462	0.00	462	0.00	1	0.4
p7.4.k	4	520	523	0.73	520	0.00	520	0.00	9	3.5
p7.4.l	4	590	591	0.27	590	0.00	590	0.00	3	3.1
p7.4.m	4	646	659	2.16	646	0.00	646	0.00	67	76.6
p7.4.n	4	730	730	0.11	730	0.00	730	0.00	1	3.2
p7.4.o	4	781	785	0.61	781	0.00	781	0.00	21	78.9
p7.4.p	4	846	847	0.12	846	0.00	846	0.00	3	24.7
p7.4.q	4	909	910	0.11	909	0.00	909	0.00	3	27.4
p7.4.r	4	970	973	0.34	970	0.00	970	0.00	5	253.5
p7.4.s	4	1022	1023	0.18	1022	0.00	1022	0.00	3	85.7
p7.4.t	4	1077	1077	0.08	1077	0.00	1077	0.00	1	16.9
Avg				0.26		0.01		0.01		168.3

Table EC.15 Detailed computational results of LNS on TOP instances of family 4

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	lb_W	gap_{Wblb}	lb_A	gap_{Ablb}	lb_B	gap_{Bblb}	blb	cpu_F
p4.2.a	2	98	1306	206	0.00	206.0	0.00	206	0.00	206	5.6
p4.2.b	2	98	1306	341	0.00	341.0	0.00	341	0.00	341	25.3
p4.2.c	2	98	1306	452	0.00	452.0	0.00	452	0.00	452	56.1
p4.2.d	2	98	1306	531	0.00	531.0	0.00	531	0.00	531	93.2
p4.2.e	2	98	1306	618	0.00	618.0	0.00	618	0.00	618	130.0
p4.2.f	2	98	1306	677	1.48	684.3	0.39	687	0.00	687	150.3
p4.2.g	2	98	1306	750	0.93	755.0	0.26	757	0.00	757	159.5
p4.2.h	2	98	1306	835	1.71	835.0	0.00	835	0.00	835	175.3
p4.2.i	2	98	1306	918	0.00	912.2	0.64	918	0.00	918	191.7
p4.2.j	2	98	1306	962	0.31	962.6	0.25	965	0.00	965	218.5
p4.2.k	2	98	1306	1022	0.00	1022.0	0.00	1022	0.00	1022	237.2
p4.2.l	2	98	1306	1074	0.00	1073.8	0.02	1074	0.00	1074	263.9
p4.2.m	2	98	1306	1132	0.00	1132.0	0.00	1132	0.00	1132	286.8
p4.2.n	2	98	1306	1172	0.17	1173.8	0.02	1174	0.00	1174	328.2
p4.2.o	2	98	1306	1218	0.00	1218.0	0.00	1218	0.00	1218	347.9
p4.2.p	2	98	1306	1239	0.24	1240.9	0.09	1242	0.00	1242	410.3
p4.2.q	2	98	1306	1265	0.24	1267.2	0.06	1268	0.00	1268	454.3
p4.2.r	2	98	1306	1284	0.62	1290.0	0.16	1292	0.00	1292	485.0
p4.2.s	2	98	1306	1302	0.15	1303.4	0.05	1304	0.00	1304	535.8
p4.2.t	2	98	1306	1306	0.00	1306.0	0.00	1306	0.00	1306	19.7
p4.3.a	3	98	1306	0	0.00	0.0	0.00	0	0.00	0	0.0
p4.3.b	3	98	1306	38	0.00	38.0	0.00	38	0.00	38	0.0
p4.3.c	3	98	1306	193	0.00	193.0	0.00	193	0.00	193	1.9
p4.3.d	3	98	1306	335	0.00	335.0	0.00	335	0.00	335	18.1
p4.3.e	3	98	1306	468	0.00	468.0	0.00	468	0.00	468	35.4
p4.3.f	3	98	1306	579	0.00	579.0	0.00	579	0.00	579	70.5
p4.3.g	3	98	1306	653	0.00	653.0	0.00	653	0.00	653	96.9
p4.3.h	3	98	1306	728	0.14	728.9	0.01	729	0.00	729	150.0
p4.3.i	3	98	1306	799	1.25	808.0	0.12	809	0.00	809	191.5
p4.3.j	3	98	1306	860	0.12	860.9	0.01	861	0.00	861	202.2
p4.3.k	3	98	1306	915	0.44	917.2	0.20	919	0.00	919	231.3
p4.3.l	3	98	1306	975	0.41	978.5	0.05	979	0.00	979	239.7
p4.3.m	3	98	1306	1053	0.95	1060.0	0.28	1063	0.00	1063	255.8
p4.3.n	3	98	1306	1121	0.00	1121.0	0.00	1121	0.00	1121	270.0
p4.3.o	3	98	1306	1171	0.09	1171.7	0.03	1172	0.00	1172	314.5
p4.3.p	3	98	1306	1222	0.00	1222.0	0.00	1222	0.00	1222	335.6
p4.3.q	3	98	1306	1253	0.00	1253.0	0.00	1253	0.00	1253	359.5
p4.3.r	3	98	1306	1269	0.32	1271.3	0.13	1273	0.00	1273	368.0
p4.3.s	3	98	1306	1292	0.23	1294.5	0.04	1295	0.00	1295	409.6
p4.3.t	3	98	1306	1304	0.08	1304.1	0.07	1305	0.00	1305	422.8
p4.4.a	4	98	1306	0	0.00	0.0	0.00	0	0.00	0	0.0
p4.4.b	4	98	1306	0	0.00	0.0	0.00	0	0.00	0	0.0
p4.4.c	4	98	1306	0	0.00	0.0	0.00	0	0.00	0	0.0
p4.4.d	4	98	1306	38	0.00	38.0	0.00	38	0.00	38	0.0
p4.4.e	4	98	1306	183	0.00	183.0	0.00	183	0.00	183	0.0
p4.4.f	4	98	1306	324	0.00	324.0	0.00	324	0.00	324	9.4
p4.4.g	4	98	1306	461	0.00	461.0	0.00	461	0.00	461	29.1
p4.4.h	4	98	1306	571	0.00	571.0	0.00	571	0.00	571	42.4
p4.4.i	4	98	1306	657	0.00	657.0	0.00	657	0.00	657	81.1
p4.4.j	4	98	1306	732	0.00	732.0	0.00	732	0.00	732	112.8
p4.4.k	4	98	1306	820	0.12	820.8	0.02	821	0.00	821	158.0
p4.4.l	4	98	1306	879	0.11	879.9	0.01	880	0.00	880	231.3
p4.4.m	4	98	1306	916	0.33	918.5	0.05	919	0.00	919	260.4
p4.4.n	4	98	1306	972	0.41	975.3	0.07	976	0.00	976	262.8
p4.4.o	4	98	1306	1051	0.95	1058.4	0.25	1061	0.00	1061	301.8
p4.4.p	4	98	1306	1119	0.45	1122.1	0.17	1124	0.00	1124	321.9
p4.4.q	4	98	1306	1160	0.09	1160.7	0.03	1161	0.00	1161	334.5
p4.4.r	4	98	1306	1203	1.08	1209.0	0.58	1216	0.00	1216	342.1
p4.4.s	4	98	1306	1257	0.24	1258.7	0.10	1260	0.00	1260	361.5
p4.4.t	4	98	1306	1282	0.23	1283.9	0.09	1285	0.00	1285	375.9
Avg					0.2313		0.0708		0.0000		196.2

Table EC.16 Detailed computational results of `LNS` on TOP instances of family 5

Inst	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	lb_W	gap_{Wblb}	lb_A	gap_{Ablb}	lb_B	gap_{Bblb}	blb	cpu_F
p5.2.a	2	64	1680	0	0.00	0.0	0.00	0	0.00	0	0.0
p5.2.b	2	64	1680	20	0.00	20.0	0.00	20	0.00	20	0.1
p5.2.c	2	64	1680	50	0.00	50.0	0.00	50	0.00	50	0.7
p5.2.d	2	64	1680	80	0.00	80.0	0.00	80	0.00	80	1.3
p5.2.e	2	64	1680	180	0.00	180.0	0.00	180	0.00	180	6.6
p5.2.f	2	64	1680	240	0.00	240.0	0.00	240	0.00	240	13.3
p5.2.g	2	64	1680	320	0.00	320.0	0.00	320	0.00	320	24.7
p5.2.h	2	64	1680	410	0.00	410.0	0.00	410	0.00	410	38.1
p5.2.i	2	64	1680	480	0.00	480.0	0.00	480	0.00	480	33.9
p5.2.j	2	64	1680	580	0.00	580.0	0.00	580	0.00	580	38.8
p5.2.k	2	64	1680	670	0.00	670.0	0.00	670	0.00	670	43.2
p5.2.l	2	64	1680	800	0.00	800.0	0.00	800	0.00	800	46.0
p5.2.m	2	64	1680	860	0.00	860.0	0.00	860	0.00	860	49.2
p5.2.n	2	64	1680	925	0.00	925.0	0.00	925	0.00	925	50.9
p5.2.o	2	64	1680	1020	0.00	1020.0	0.00	1020	0.00	1020	54.8
p5.2.p	2	64	1680	1150	0.00	1150.0	0.00	1150	0.00	1150	59.9
p5.2.q	2	64	1680	1195	0.00	1195.0	0.00	1195	0.00	1195	64.0
p5.2.r	2	64	1680	1260	0.00	1260.0	0.00	1260	0.00	1260	66.6
p5.2.s	2	64	1680	1330	0.75	1339.0	0.07	1340	0.00	1340	71.7
p5.2.t	2	64	1680	1400	0.00	1400.0	0.00	1400	0.00	1400	73.9
p5.2.u	2	64	1680	1460	0.00	1460.0	0.00	1460	0.00	1460	77.8
p5.2.v	2	64	1680	1495	0.67	1501.0	0.27	1505	0.00	1505	82.1
p5.2.w	2	64	1680	1560	0.32	1560.0	0.32	1560	0.32	1565	85.8
p5.2.x	2	64	1680	1590	1.26	1597.5	0.78	1610	0.00	1610	88.9
p5.2.y	2	64	1680	1635	0.61	1639.0	0.37	1645	0.00	1645	91.3
p5.2.z	2	64	1680	1670	0.60	1672.0	0.48	1680	0.00	1680	84.5
p5.3.a	3	64	1680	0	0.00	0.0	0.00	0	0.00	0	0.0
p5.3.b	3	64	1680	15	0.00	15.0	0.00	15	0.00	15	0.2
p5.3.c	3	64	1680	20	0.00	20.0	0.00	20	0.00	20	0.2
p5.3.d	3	64	1680	60	0.00	60.0	0.00	60	0.00	60	0.9
p5.3.e	3	64	1680	95	0.00	95.0	0.00	95	0.00	95	0.8
p5.3.f	3	64	1680	110	0.00	110.0	0.00	110	0.00	110	1.5
p5.3.g	3	64	1680	185	0.00	185.0	0.00	185	0.00	185	7.9
p5.3.h	3	64	1680	260	0.00	260.0	0.00	260	0.00	260	8.0
p5.3.i	3	64	1680	335	0.00	335.0	0.00	335	0.00	335	17.9
p5.3.j	3	64	1680	470	0.00	470.0	0.00	470	0.00	470	30.4
p5.3.k	3	64	1680	495	0.00	495.0	0.00	495	0.00	495	43.0
p5.3.l	3	64	1680	595	0.00	595.0	0.00	595	0.00	595	52.6
p5.3.m	3	64	1680	650	0.00	650.0	0.00	650	0.00	650	52.4
p5.3.n	3	64	1680	755	0.00	755.0	0.00	755	0.00	755	54.8
p5.3.o	3	64	1680	870	0.00	870.0	0.00	870	0.00	870	53.7
p5.3.p	3	64	1680	990	0.00	990.0	0.00	990	0.00	990	59.1
p5.3.q	3	64	1680	1070	0.00	1070.0	0.00	1070	0.00	1070	63.0
p5.3.r	3	64	1680	1125	0.00	1125.0	0.00	1125	0.00	1125	64.9
p5.3.s	3	64	1680	1190	0.00	1190.0	0.00	1190	0.00	1190	63.8
p5.3.t	3	64	1680	1260	0.00	1260.0	0.00	1260	0.00	1260	67.6
p5.3.u	3	64	1680	1345	0.00	1345.0	0.00	1345	0.00	1345	69.2
p5.3.v	3	64	1680	1420	0.35	1424.0	0.07	1425	0.00	1425	72.1
p5.3.w	3	64	1680	1475	0.68	1481.5	0.24	1485	0.00	1485	73.7
p5.3.x	3	64	1680	1530	1.63	1544.5	0.68	1555	0.00	1555	76.4
p5.3.y	3	64	1680	1590	0.31	1592.0	0.19	1595	0.00	1595	79.8
p5.3.z	3	64	1680	1635	0.00	1635.0	0.00	1635	0.00	1635	82.7
p5.4.a	4	64	1680	0	0.00	0.0	0.00	0	0.00	0	0.0
p5.4.b	4	64	1680	0	0.00	0.0	0.00	0	0.00	0	0.0
p5.4.c	4	64	1680	20	0.00	20.0	0.00	20	0.00	20	0.0
p5.4.d	4	64	1680	20	0.00	20.0	0.00	20	0.00	20	0.0
p5.4.e	4	64	1680	20	0.00	20.0	0.00	20	0.00	20	0.0
p5.4.f	4	64	1680	80	0.00	80.0	0.00	80	0.00	80	1.1
p5.4.g	4	64	1680	140	0.00	140.0	0.00	140	0.00	140	1.8
p5.4.h	4	64	1680	140	0.00	140.0	0.00	140	0.00	140	1.9
p5.4.i	4	64	1680	240	0.00	240.0	0.00	240	0.00	240	4.4
p5.4.j	4	64	1680	340	0.00	340.0	0.00	340	0.00	340	10.1
p5.4.k	4	64	1680	340	0.00	340.0	0.00	340	0.00	340	9.6
p5.4.l	4	64	1680	430	0.00	430.0	0.00	430	0.00	430	21.6
p5.4.m	4	64	1680	555	0.00	555.0	0.00	555	0.00	555	36.5
p5.4.n	4	64	1680	620	0.00	620.0	0.00	620	0.00	620	55.2
p5.4.o	4	64	1680	690	0.00	690.0	0.00	690	0.00	690	55.1
p5.4.p	4	64	1680	765	0.00	765.0	0.00	765	0.00	765	68.3
p5.4.q	4	64	1680	860	0.00	860.0	0.00	860	0.00	860	68.4
p5.4.r	4	64	1680	960	0.00	960.0	0.00	960	0.00	960	66.9
p5.4.s	4	64	1680	1030	0.00	1030.0	0.00	1030	0.00	1030	68.9
p5.4.t	4	64	1680	1160	0.00	1160.0	0.00	1160	0.00	1160	66.1
p5.4.u	4	64	1680	1300	0.00	1300.0	0.00	1300	0.00	1300	69.1
p5.4.v	4	64	1680	1320	0.00	1320.0	0.00	1320	0.00	1320	74.8
p5.4.w	4	64	1680	1385	0.36	1386.0	0.29	1390	0.00	1390	78.8
p5.4.x	4	64	1680	1445	0.35	1449.5	0.03	1450	0.00	1450	82.2
p5.4.y	4	64	1680	1520	0.00	1520.0	0.00	1520	0.00	1520	81.8
p5.4.z	4	64	1680	1620	0.00	1620.0	0.00	1620	0.00	1620	85.9
Avg					0.1012		0.0485		0.0041		43.0

Table EC.17 Detailed computational results of LNS on TOP instances of family 6

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	lb_W	gap_{Wblb}	lb_A	gap_{Ablb}	lb_B	gap_{Bblb}	blb	cpu_F
p6.2.a	2	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.2.b	2	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.2.c	2	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.2.d	2	62	1344	192	0.00	192.0	0.00	192	0.00	192	6.2
p6.2.e	2	62	1344	360	0.00	360.0	0.00	360	0.00	360	27.2
p6.2.f	2	62	1344	588	0.00	588.0	0.00	588	0.00	588	41.7
p6.2.g	2	62	1344	660	0.00	660.0	0.00	660	0.00	660	40.5
p6.2.h	2	62	1344	780	0.00	780.0	0.00	780	0.00	780	44.9
p6.2.i	2	62	1344	888	0.00	888.0	0.00	888	0.00	888	50.4
p6.2.j	2	62	1344	948	0.00	948.0	0.00	948	0.00	948	52.1
p6.2.k	2	62	1344	1032	0.00	1032.0	0.00	1032	0.00	1032	57.2
p6.2.l	2	62	1344	1110	0.54	1111.8	0.38	1116	0.00	1116	62.8
p6.2.m	2	62	1344	1170	1.54	1184.4	0.30	1188	0.00	1188	65.9
p6.2.n	2	62	1344	1242	1.45	1252.2	0.62	1260	0.00	1260	69.7
p6.3.a	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.b	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.c	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.d	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.e	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.f	3	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.3.g	3	62	1344	282	0.00	282.0	0.00	282	0.00	282	7.7
p6.3.h	3	62	1344	444	0.00	444.0	0.00	444	0.00	444	24.9
p6.3.i	3	62	1344	642	0.00	642.0	0.00	642	0.00	642	39.8
p6.3.j	3	62	1344	828	0.00	828.0	0.00	828	0.00	828	56.8
p6.3.k	3	62	1344	894	0.00	894.0	0.00	894	0.00	894	55.2
p6.3.l	3	62	1344	1002	0.00	1002.0	0.00	1002	0.00	1002	55.2
p6.3.m	3	62	1344	1080	0.00	1080.0	0.00	1080	0.00	1080	58.0
p6.3.n	3	62	1344	1170	0.00	1170.0	0.00	1170	0.00	1170	62.4
p6.4.a	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.b	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.c	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.d	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.e	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.f	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.g	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.h	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.i	4	62	1344	0	0.00	0.0	0.00	0	0.00	0	0.0
p6.4.j	4	62	1344	366	0.00	366.0	0.00	366	0.00	366	9.6
p6.4.k	4	62	1344	528	0.00	528.0	0.00	528	0.00	528	26.3
p6.4.l	4	62	1344	696	0.00	696.0	0.00	696	0.00	696	46.1
p6.4.m	4	62	1344	912	0.00	912.0	0.00	912	0.00	912	58.3
p6.4.n	4	62	1344	1068	0.00	1068.0	0.00	1068	0.00	1068	69.9
					0.0840		0.0311		0.0000		25.9

Table EC.18 Detailed computational results of LNS on TOP instances of family 7

Inst	$\|\mathcal{K}\|$	$\|\mathcal{S}\|$	$\|\mathcal{C}\|$	lb_W	gap_{Wblb}	lb_A	gap_{Ablb}	lb_B	gap_{Bblb}	blb	cpu_F
p7.2.a	2	100	1458	30	0.00	30.0	0.00	30	0.00	30	0.0
p7.2.b	2	100	1458	64	0.00	64.0	0.00	64	0.00	64	0.2
p7.2.c	2	100	1458	101	0.00	101.0	0.00	101	0.00	101	0.9
p7.2.d	2	100	1458	190	0.00	190.0	0.00	190	0.00	190	4.4
p7.2.e	2	100	1458	290	0.00	290.0	0.00	290	0.00	290	18.7
p7.2.f	2	100	1458	387	0.00	387.0	0.00	387	0.00	387	42.0
p7.2.g	2	100	1458	459	0.00	459.0	0.00	459	0.00	459	75.0
p7.2.h	2	100	1458	521	0.00	521.0	0.00	521	0.00	521	92.8
p7.2.i	2	100	1458	578	0.35	579.6	0.07	580	0.00	580	131.4
p7.2.j	2	100	1458	646	0.00	646.0	0.00	646	0.00	646	137.7
p7.2.k	2	100	1458	704	0.14	704.3	0.10	705	0.00	705	151.4
p7.2.l	2	100	1458	767	0.00	767.0	0.00	767	0.00	767	155.1
p4.2.m	2	100	1458	821	0.73	825.2	0.22	827	0.00	827	175.9
p7.2.n	2	100	1458	888	0.00	888.0	0.00	888	0.00	888	181.4
p7.2.o	2	100	1458	945	0.00	945.0	0.00	945	0.00	945	203.1
p7.2.p	2	100	1458	1002	0.00	1002.0	0.00	1002	0.00	1002	208.7
p7.2.q	2	100	1458	1041	0.29	1043.2	0.08	1044	0.00	1044	233.8
p7.2.r	2	100	1458	1094	0.00	1094.0	0.00	1094	0.00	1094	247.2
p7.2.s	2	100	1458	1135	0.09	1135.3	0.06	1136	0.00	1136	266.9
p7.2.t	2	100	1458	1176	0.26	1177.4	0.14	1179	0.00	1179	283.5
p7.3.a	3	100	1458	0	0.00	0.0	0.00	0	0.00	0	0.0
p7.3.b	3	100	1458	46	0.00	46.0	0.00	46	0.00	46	0.0
p7.3.c	3	100	1458	79	0.00	79.0	0.00	79	0.00	79	0.2
p7.3.d	3	100	1458	117	0.00	117.0	0.00	117	0.00	117	0.8
p7.3.e	3	100	1458	175	0.00	175.0	0.00	175	0.00	175	1.8
p7.3.f	3	100	1458	247	0.00	247.0	0.00	247	0.00	247	5.6
p7.3.g	3	100	1458	344	0.00	344.0	0.00	344	0.00	344	14.7
p7.3.h	3	100	1458	424	0.24	424.8	0.05	425	0.00	425	38.6
p7.3.i	3	100	1458	487	0.00	487.0	0.00	487	0.00	487	59.9
p7.3.j	3	100	1458	564	0.00	564.0	0.00	564	0.00	564	86.7
p7.3.k	3	100	1458	633	0.00	633.0	0.00	633	0.00	633	121.7
p7.3.l	3	100	1458	681	0.44	683.7	0.04	684	0.00	684	132.9
p4.3.m	3	100	1458	762	0.00	762.0	0.00	762	0.00	762	160.8
p7.3.n	3	100	1458	820	0.00	820.0	0.00	820	0.00	820	185.0
p7.3.o	3	100	1458	860	1.63	871.2	0.32	874	0.00	874	202.1
p7.3.p	3	100	1458	920	0.98	926.2	0.30	929	0.00	929	231.2
p7.3.q	3	100	1458	987	0.00	987.0	0.00	987	0.00	987	235.1
p7.3.r	3	100	1458	1021	0.49	1022.8	0.31	1026	0.00	1026	260.9
p7.3.s	3	100	1458	1081	0.00	1081.0	0.00	1081	0.00	1081	261.9
p7.3.t	3	100	1458	1118	0.18	1118.3	0.15	1120	0.00	1120	275.7
p7.4.a	4	100	1458	0	0.00	0.0	0.00	0	0.00	0	0.0
p7.4.b	4	100	1458	30	0.00	30.0	0.00	30	0.00	30	0.0
p7.4.c	4	100	1458	46	0.00	46.0	0.00	46	0.00	46	0.0
p7.4.d	4	100	1458	79	0.00	79.0	0.00	79	0.00	79	0.3
p7.4.e	4	100	1458	123	0.00	123.0	0.00	123	0.00	123	1.1
p7.4.f	4	100	1458	164	0.00	164.0	0.00	164	0.00	164	1.4
p7.4.g	4	100	1458	217	0.00	217.0	0.00	217	0.00	217	3.7
p7.4.h	4	100	1458	285	0.00	285.0	0.00	285	0.00	285	6.9
p7.4.i	4	100	1458	366	0.00	366.0	0.00	366	0.00	366	16.4
p7.4.j	4	100	1458	462	0.00	462.0	0.00	462	0.00	462	32.9
p7.4.k	4	100	1458	518	0.39	519.8	0.04	520	0.00	520	51.5
p7.4.l	4	100	1458	590	0.00	590.0	0.00	590	0.00	590	73.0
p7.4.m	4	100	1458	646	0.00	646.0	0.00	646	0.00	646	109.8
p7.4.n	4	100	1458	730	0.00	730.0	0.00	730	0.00	730	141.2
p7.4.o	4	100	1458	781	0.00	781.0	0.00	781	0.00	781	165.3
p7.4.p	4	100	1458	846	0.00	846.0	0.00	846	0.00	846	183.9
p7.4.q	4	100	1458	909	0.00	909.0	0.00	909	0.00	909	216.6
p7.4.r	4	100	1458	970	0.00	970.0	0.00	970	0.00	970	228.2
p7.4.s	4	100	1458	1022	0.00	1022.0	0.00	1022	0.00	1022	241.3
p7.4.t	4	100	1458	1077	0.00	1077.0	0.00	1077	0.00	1077	277.6
Avg					0.1031		0.0313		0.0000		110.6