The Multi-Stop Station Location Problem: Exact Approaches

Erik Mühmer*[1], Miriam Ganz[1], Marco E. Lübbecke[1], and Felix J. L. Willamowski[1]

[1]RWTH Aachen University, Chair of Operations Research, Kackertstr. 7, D-52072 Aachen, Germany

Abstract

The multi-stop station location problem (MSLP) aims to place stations such that a set of trips is feasible with respect to length bounds while minimizing cost. Each trip consists of a sequence of stops that must be visited in a given order, and a length bound that controls the maximum length that is possible without visiting a station. Installing stations and detours cause costs that are to be minimized. The MSLP relates to problems in transportation and telecommunications where strategic decisions (such as the placement of charging stations) depend on operational considerations (such as offering a certain set of planned trips). In this paper, we introduce exact approaches to solve the MSLP to optimality. First, we introduce an arc-based mixed-integer program (MIP) that captures the problem and can be solved with any MIP solver. In addition, we propose pattern-based formulations that we solve by branch-price-and-cut. We conduct experiments on randomly generated instances to evaluate the performance of our approaches and show that the pattern-based formulations outperform the compact MIP formulation. In addition, we show that a nested branch-price-and-cut approach is able to solve a practically relevant instance in the context of siting charging stations for an intercity bus service.

Keywords: Multi-Stop Station Location · Charging Station Placement · Branch-Price-and-Cut

1 Introduction

In this paper, we present exact approaches to the *(directed) multi-stop station location* problem (MSLP) [62]. The MSLP can be seen as an optimization problem on a graph. It consists of a location part and a pathfinding (or routing) part. The task is to enable *trips*, given by sequences of nodes (*stops*), with respect to length bounds while minimizing cost. For each trip the maximum distance that can be traveled through the graph without visiting special nodes (*stations*) is bounded. The feasibility of trips can be ensured by visiting stations between stops of a trip, where the traveled distances are reset. This means that we assume full charging in the context of electromobility and charging stations. In addition to the feasibility aspect, the cost of traversing edges and selecting stations must be minimized. Potential stations have costs that must be paid once if they are selected in a solution.

The MSLP is primarily motivated by the need to install charging stations for electric vehicles (EVs). Greenhouse gas emissions from the mobility sector alone have more than doubled since 1970, with 80 percent of that coming from road vehicles alone [16]. One way to reduce greenhouse gas emissions is to switch from combustion engine vehicles to zero-emission alternatives, such as battery-powered vehicles. However, switching to alternative fuels like electricity requires an appropriate refueling infrastructure such as charging stations. While the range of EVs without recharging may be sufficient for intracity travel, long-distance travel is more challenging. For example, an intercity bus service that wants to electrify its fleet needs to ensure that all of the

*Corresponding author; e-mail: muehmer@or.rwth-aachen.de

trips it offers can be realized with respect to the ranges of the EVs. In particular, it is desirable to install charging infrastructure in a cost-effective manner while minimizing detours for charging. The MSLP captures this problem in an abstract and formal way. Locating charging or alternative fuel stations has become a hot topic in recent years. Much of the research typically considers origin-destination flows. In contrast, the MSLP focuses on (but is not limited to) long distance trips consisting of multiple fixed stops.

In this paper, we propose several approaches to solve the MSLP. First, we introduce an arc-based mixed-integer program (MIP) that mathematically captures the MSLP and can be solved by any standard MIP solver. However, even small instances result in large MIPs, so solvers struggle to solve larger instances. Therefore, we propose pattern-based formulations. At first, we identify a pattern-based formulation that assigns a complete path to each trip. Secondly, we describe an alternative formulation that assigns paths to parts (*segments*) of a trip. The latter formulation leads to efficiently solvable shortest path pricing problems. Finally, we combine both pattern-based formulations to a nested approach. The pattern-based formulations are solved using branch-price-and-cut. To achieve integrality, we rely on traditional and problem specific branching strategies. In addition, we develop cuts that significantly speed up the solving process of the second pattern-based formulation. The implementation relies on SCIP [4]. We conduct experiments on randomly generated instances to evaluate the performance of all proposed formulations. We also show that we are able to solve a large real-world instance to optimality.

The remainder of this paper is organized as follows. In Section 1.1, we discuss the existing literature that is related to this work. In Section 2, we formally define the MSLP and introduce an arc-based formulation (Section 2.1) as well as pattern-based formulations (Section 2.2). The algorithmic components we use to solve the pattern-based formulations with branch-price-and-cut are described in Section 3. In Section 4, we present our experiments and discuss their results. Finally, in Section 5, we conclude our work and discuss possible extensions and research directions.

1.1 Related Work

Problems related to electromobility or other alternative fuels have received more and more attention in recent years. On the one hand, the limited ranges and the scarcity of the necessary refueling infrastructure, such as charging stations, have been incorporated into existing and well-studied problems (e.g., vehicle routing). On the other hand, new problems and models have emerged and been studied. In the following, we focus on related work that locates stations in a network to ensure feasibility with respect to length, distance, or similar constraints.

Kuby and Lim [32] introduce the Flow-refueling Location Problem (FRLP). For given origin-destination pairs with assigned flow volumes they want to locate refueling stations on a network such that the total (refueled) flow volume is maximized. Each flow is assumed to follow a fixed shortest path from origin to destination. This problem received much attention in the literature and several extensions and adaptations have been studied [7, 31, 33, 38, 40, 48, 49, 57, 58, 59]. Kim and Kuby [29] propose an extension that allows flows to deviate from the assigned shortest path to refuel. The authors propose a mixed-integer program that requires precomputation of possible detour paths. Since this precomputation step for these detour paths is very expensive, Kim and Kuby [28] also propose heuristic approaches. Again, many alternative approaches as well as extensions were studied [19, 24, 25, 39, 44, 66]. Yıldız, Arslan, and Karaşan [66] propose a branch-and-price approach to an extension that respects a distance threshold that limits the length of a deviation path. Subsequently, Göpfert and Bock [19] present a branch-and-cut approach to the problem and compare the performance with the results of Yıldız, Arslan, and Karaşan [66]. Most works in this area have in common that they consider origin-destination pairs without any intermediate (fixed) stops, and that a fixed number of stations have to be located while maximizing flow. In contrast, the MSLP considers given trips with multiple stops and costs of stations that are part of the objective function.

Moreover, other papers study the problem of charging station placement and are not based on the FRLP. Many of them consider some kind of coverage-based approach [18, 30, 54, 60, 61, 68]. For a given network, stations are placed such that all nodes of certain node sets are reachable from

each other with respect to the limited range. The objective function varies, but typically takes into account the infrastructure cost. Kınay, Gzara, and Alumur [30] consider origin-destination round trips and minimize the total required charging in addition to the infrastructure cost. This objective function is similar to the MSLP objective function, but does not consider intermediate stops. Kunith, Mendelevitch, and Goehlich [34] aim to electrify a city bus network. Among other things, they locate charging infrastructure for the electric buses. Charging stations can be located at bus stops to ensure that all fixed routes (i.e., no detours are allowed) are feasible and infrastructure costs are minimized. A similar problem is studied by Tzamakos, Iliopoulou, and Kepaptsoglou [56]. Other approaches focus on minimizing vehicle-related objective functions. For example, Hess et al. [22] minimize the average travel time of EVs (including charging and queuing times) given a limited budget for station placement. Furthermore, some authors include temporal considerations such as charging times or time-dependent demand [14, 27, 55]. Such papers typically focus on smaller areas, such as cities and urban areas. Kang and Recker [27] aim to locate hydrogen refueling stations with respect to activities assigned to households. They minimize infrastructure costs while keeping travel time below a threshold.

Routing problems with intermediate stops, for instance vehicle routing or location routing related problems, are also related to the MSLP. For an extensive overview we refer to the work of Schiffer et al. [51]. Vehicle routing problems (VRPs) with intermediate stops are related to the MSLP as the stations are inserted into trips as intermediate stops. Related problems were studied, e.g., by Conrad and Figliozzi [11], Crevier, Cordeau, and Laporte [12], Desaulniers et al. [15], Erdoğan and Miller-Hooks [17], Muter, Cordeau, and Laporte [42], and Schneider, Stenger, and Goeke [52]. However, those problems consider only the routing part. Location routing problems (LRP) combine the routing of vehicles and the placement of stations (for intermediate stops). In the recent years, many works dealing with location routing problems and solution approaches were published [2, 6, 10, 23, 50, 64, 65]. In contrast to those problems, the MSLP does not consider depots or vehicles that have to be assigned to customers, i.e., in the context of the MSLP customers are already assigned to vehicles (in a fixed order). Hence, the routing part of the MSLP differs as it allows only routing to stations between two stops.

Another class of related problems, not motivated by alternative fuel infrastructure design, are network design problems that aim to locate special vertices or arcs that, for example, refresh a signal. Cabral et al. [5] introduce the network design problem with relays. Given an undirected graph with edge costs and lengths, one must select edges and locate relays at vertices such that given origin-destination pairs are connected by a path that does not violate a length constraint on the travel distance of a signal without passing through a relay. The sum of the edge and relay costs is minimized, where each selected edge and relay just have to be payed once. Many variations of the problem have been studied, e.g., in the context of optical networks [8, 9, 21], or in the presence of multiple commodities and other additional constraints [26, 37]. While many heuristic approaches have been proposed, recent work also presents exact approaches based on branch-and-cut and branch-and-price [35, 36, 37, 43, 67]. In addition, Zheng et al. [69] studied an extension of the network design problem with relays in the context of placing capacitated charging stations. In contrast to such network design problems, the MSLP considers trips defined by multiple stops. Furthermore, the distance traveled is considered in the objective function for each trip.

Willamowski, Ganz, and Mühmer [62] introduce the MSLP, which, in contrast to the previously discussed problems, aims to plan charging stations for trips with fixed stops. The authors discuss theoretical results and propose an approximation algorithm to solve the MSLP. In this paper, we focus on exact approaches that use mixed-integer programming and branch-price-and-cut. However, we integrate the idea of their approximation algorithm into our approaches.

2 Formulations

Before introducing mathematical formulations to capture the MSLP, we first formally define the problem [62]. The directed multi-stop station location problem is given by a directed graph $G = (V, E)$, edge costs $c^E : E \to \mathbb{Q}_{\geq 0}$, edge lengths $\ell : E \to \mathbb{Q}_{\geq 0}$, a set $F \subseteq V$ of nodes at which stations can be placed, station costs $c^F : F \to \mathbb{Q}_{\geq 0}$, and a set of trips T. Each trip $t \in T$ is given

by a sequence of stops $\mathcal{S}_t = (v_1, \ldots, v_{m_t}) \in V^{m_t}$ for $m_t \in \mathbb{N}_{\geq 2}$ and a length bound $b_t \in \mathbb{Q}_{\geq 0}$. Let $\mathcal{S}_t[i]$ with $i \in [m_t] := \{1, \ldots, m_t\}$ denote the i-th element of $\mathcal{S}_t$. For convenience, we use the abbreviation $t[i] := \mathcal{S}_t[i]$. Each trip t consists of segments $S_t := [m_t - 1]$, which are implicitly given by the stops, i.e., the start and the end of a segment correspond to two consecutive stops of the trip. The goal is to select stations $F^* \subseteq F$ and for each trip $t \in T$ a path

$$\mathcal{P}_t = (v_1, f_1^1, \ldots, f_1^{k_1}, v_2, f_2^1, \ldots, f_2^{k_2}, \ldots, v_{m_t-1}, f_{m_t-1}^1, \ldots, f_{m_t-1}^{k_{m_t-1}}, v_{m_t})$$

with $v_i = t[i]$ and $f_i^{k_s} \in F^*$ for all $i \in [m_t]$ and $k_s \in \{0, \ldots, |F^*|\}$ with $s \in S_t$. The path $\mathcal{P}_t$ contains exactly the ordered stops of t and additional stops at stations. Between two consecutive stops of t the path $\mathcal{P}_t$ must only visit stations $f \in F^*$ that are used to reset the remaining range to b_t (i.e., in the context of electromobility, we assume full charging). Since each $\mathcal{P}_t$ has to obey the length bound, the lengths of the paths between the first stop and the first station stop, any two consecutive station stops (there could be planned stops of t in between), the last station stop and the last stop, or, if $\mathcal{P}_t = \mathcal{S}_t$, between the first stop and the last stop, must be within b_t. We assume that G is complete and that the triangle inequality holds for c^E, which ensures that it is not advantageous to visit a station to reduce the path cost. The objective is to minimize the sum of the station costs and the path costs

$$\sum_{f \in F^*} c^F(f) + \sum_{t \in T} c^E(\mathcal{P}_t) \quad .$$

Additionally, we introduce useful definitions for the multi-stop station location problem. Often we need to refer to specific edges depending of the trip and its segments. Therefore, we distinguish different types of edges. We can enter a segment, i.e., we move from a stop of a trip (the start of the segment) to a station node. If we are in a segment, we can move to other stations or leave the segment again by moving to the next stop of the trip. Moreover, we can omit any additional stops in a segment by moving directly from the start to the end of the segment (i.e., we do not visit a station between two consecutive stops). Thus, for a trip $t \in T$ and a segment $s \in S_t$, we define

- $\hat{\ell}_{t,s,f} := \ell(t[s], f)\ \forall f \in F$ (the length of the entering edge (v, f) connecting the start node of s and station f),
- $\bar{\ell}_{t,s} := \ell(t[s], t[s+1])$ (the length of the traversing edge (v_1, v_2) connecting the start node and the end node of s), and
- $\tilde{\ell}_{t,s,f} := \ell(f, t[s+1])\ \forall f \in F$ (the length of the leaving edge (f, v) connecting station f and the end node of s).

Analogously, concerning the edge costs, we define for a trip $t \in T$ and a segment $s \in S_t$

- $\hat{c}_{t,s,f} := c^E(t[s], f)\ \forall f \in F$,
- $\bar{c}_{t,s} := c^E(t[s], t[s+1])$, and
- $\tilde{c}_{t,s,f} := c^E(f, t[s+1])\ \forall f \in F$.

2.1 Arc-based Formulation

The MSLP can be described mathematically by an arc-based compact formulation, which allows us to solve a MSLP instance as a mixed-integer program (MIP) using a standard MIP solver. We call this approach $\mathcal{A}_\mathrm{A}$. Let $F_{t,v} := \{f \mid f \in F, v \neq f, \ell(v, f) \leq b_t\}$ be the set of all stations that are directly reachable from $v \in V$ with respect to the length bound b_t of trip $t \in T$. We use different types of x-variables to model the possible arcs of each trip. The binary variable $x_{t,s,i,j}$ indicates whether station j is visited directly after station i within segment s of trip t. Similarly, the binary variables $\hat{x}_{t,s,i}$ and $\tilde{x}_{t,s,i}$ indicate whether a station i is visited as first or last station in a segment s of trip t, respectively. The binary variable $\bar{x}_{t,s}$ is used to decide whether no station is used at all (set to 1) for a segment s of trip t. Additionally, the binary variable y_f indicates whether a station should be installed and the continuous variable $r_{t,s}$ is used to keep track of the

remaining distance when leaving segment s of trip t. We model the directed multi-stop station location problem as follows:

$$\min \quad \sum_{f \in F} c^F(f) y_f + \sum_{\substack{t \in T, \\ s \in S_t}} \left(\bar{c}_{t,s} \bar{x}_{t,s} + \sum_{i \in F} \left(\hat{c}_{t,s,i} \hat{x}_{t,s,i} + \tilde{c}_{t,s,i} \tilde{x}_{t,s,i} + \sum_{j \in F_{t,i}} c^E(i,j) x_{t,s,i,j} \right) \right) \tag{1}$$

$$\text{s.t.} \quad \hat{x}_{t,s,j} + \sum_{\substack{i \in F: \\ j \in F_{t,i}}} x_{t,s,i,j} = \tilde{x}_{t,s,j} + \sum_{i \in F_{t,j}} x_{t,s,j,i} \qquad \forall t \in T,\ \forall s \in S_t,\ \forall j \in F \tag{2}$$

$$\bar{x}_{t,s} + \sum_{f \in F} \hat{x}_{t,s,f} = 1 \qquad \forall t \in T,\ \forall s \in S_t \tag{3}$$

$$\hat{x}_{t,s,f} + \sum_{i \in F_{t,f}} x_{t,s,i,f} \leq y_f \qquad \forall t \in T,\ \forall s \in S_t,\ \forall f \in F \tag{4}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{f \in F} \hat{\ell}_{t,s,f} \hat{x}_{t,s,f} \leq \begin{cases} b_t, & \text{if } s = 1 \\ r_{t,s-1}, & \text{else} \end{cases} \qquad \forall t \in T,\ \forall s \in S_t \tag{5}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{f \in F} \tilde{\ell}_{t,s,f} \tilde{x}_{t,s,f} \leq b_t - r_{t,s} \qquad \forall t \in T,\ \forall s \in S_t \tag{6}$$

$$r_{t,s} + (b_t + \bar{\ell}_{t,s}) \bar{x}_{t,s} \leq \begin{cases} 2b_t, & \text{if } s = 1 \\ b_t + r_{t,s-1}, & \text{else} \end{cases} \qquad \forall t \in T,\ \forall s \in S_t \tag{7}$$

$$x_{t,s,i,j} \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t, i \in F, j \in F_{t,i} \tag{8}$$

$$\bar{x}_{t,s} \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t \tag{9}$$

$$\hat{x}_{t,s,i} \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t,\ \forall i \in F \tag{10}$$

$$\tilde{x}_{t,s,i} \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t,\ \forall i \in F \tag{11}$$

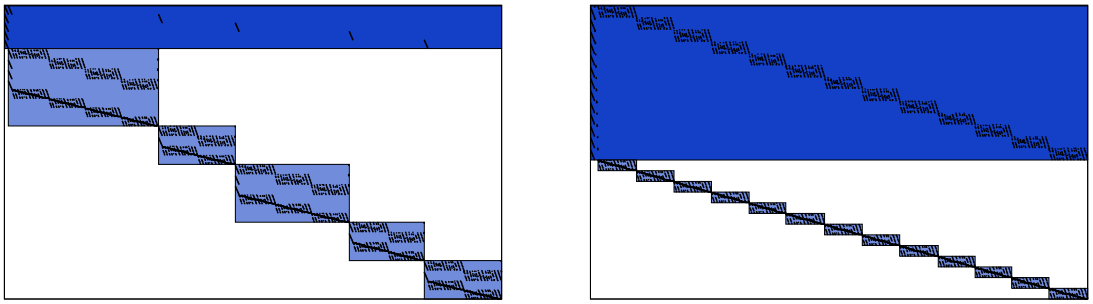
$$y_f \in \{0,1\} \qquad \forall f \in F \tag{12}$$

$$r_{t,s} \geq 0 \qquad \forall t \in T,\ \forall s \in S_t \tag{13}$$

The objective function minimizes the cost of installing new stations and the path cost of all trips. For each segment of a trip, the flow constraint (2) ensures that a node of a station is entered and left if visited. In addition, the constraint (3) ensures that each trip is completely planned and that no segment is omitted. The constraint (4) ensures that a station must be opened if it is visited in at least one segment of a trip. Finally, the following constraints implement the range restrictions. In a segment we only have the option of either going directly from the start node of the segment to the end node of the segment, or visiting one or more stations from the start node before reaching the end node of the segment. The x-variables are defined only for the edges for which $\ell(i,j) \leq b_t$ where $i, j \in F$. This ensures that the range restrictions for edges connecting two stations are not violated. The constraints (5) and (6) limit the available range at the end of each segment of a trip. Constraint (5) corresponds to the case where the remaining range after the previous segment must be large enough to enter the next segment. Supplementing this, constraint (6) ensures that the remaining range at the end of a segment takes into account the last step to reach the end of the segment. Constraint (7) links the r-variable of a segment to the r-variable of the previous segment if no station is visited. Finally, the constraint (13) ensures that the range is not exceeded for any trip.

2.2 Pattern-based Formulations

The arc-based formulation allows us to solve instances of the multi-stop station location problem using a MIP solver. However, for large instances, the size of the formulation grows rapidly because it uses $O(N^2M)$ variables and $O(NM)$ constraints with $N = |V|$ and $M = \sum_{t \in T} |S_t|$. Thus, we propose multiple decompositions that result in pattern-based formulations. These are then solved with branch-price-and-cut.



(a) trip patterns (b) segment patterns

Figure 1: The two exploited structures of a MSLP instance are visualized by the rearranged coefficient matrices of the arc-based formulation. Black dots correspond to non-zero entries. The dark blue boxes represent the master problems and the light blue boxes represent the pricing problems.

2.2.1 Trip Patterns

The first formulation (denoted by $\mathcal{A}_{\mathrm{T}}$) decomposes the problem by its trips, as visualized in Figure 1a. It is similar to the reformulation obtained by performing a Dantzig-Wolfe decomposition [13] of the arc-based formulation by keeping only constraint (3) in the master problem. Let P_t be the set of all possible (and valid) patterns for trip $t \in T$. A pattern $p \in P_t$ corresponds to a valid path for trip t, i.e., all stops are visited in the correct order and the length bounds are not violated by visiting stations if necessary. The cost c_p of p is the sum of the costs of all edges used by the pattern. This formulation assigns such a pattern $p \in P_t$ to each trip t by using pattern variables. Since this would result in too many variables, we use column generation to solve the linear relaxation of the master problem. Its restricted version contains only a (small) subset of these pattern variables. New pattern variables are generated by several pricing problems in an iterative process until the master problem is solved. While the master problem consists of only two different types of constraints, the pricing problems are much more complex.

Master Problem The master problem of the trip-pattern-based formulation has to select a pattern for each trip and compute the total costs. This is realized with pattern variables $\lambda_p \in \{0,1\}$ with $p \in P_t$ and station variables $y_f \in \{0,1\}$ with $f \in F$, which we already know from the arc-based formulation. We denote the set of stations visited by a pattern p by F_p.

$$\min \quad \sum_{f \in F} c^F(f) y_f + \sum_{\substack{t \in T,\\ p \in P_t}} c_p \lambda_p \tag{14}$$

$$\text{s.t.} \quad \sum_{p \in P_t} \lambda_p \geq 1 \qquad \forall t \in T \qquad [\pi_t \geq 0] \tag{15}$$

$$\sum_{\substack{p \in P_t:\\ f \in F_p}} \lambda_p \leq y_f \qquad \forall t \in T,\ \forall f \in F \qquad [\pi_{t,f} \leq 0] \tag{16}$$

$$\lambda_p \in \{0,1\} \qquad \forall t \in T,\ \forall p \in P_t \tag{17}$$

$$y_f \in \{0,1\} \qquad \forall f \in F \tag{18}$$

$$\tag{19}$$

The objective function minimizes the costs of all selected patterns and the costs of the selected stations. The constraint (15) ensures that a pattern is selected for each trip and the constraint (16) pushes the station variables to 1 if the stations are visited by any trip. For constraints containing pattern variables, the corresponding dual variables are given (in brackets) as we need them to formulate the pricing problems.

Pricing Problems To generate new pattern variables, we define pricing problems that are solved during the pricing phase. As mentioned earlier, each pattern variable corresponds to a specific path for a trip. We use one pricing problem for each trip $t \in T$:

$$\min \quad -\pi_t - \sum_{f \in F} \pi_{t,f} y_f + \sum_{\substack{t \in T, \\ s \in S_t}} \left(\bar{c}_{t,s} \bar{x}_{t,s} + \sum_{f \in F} \left(\hat{c}_{t,s,f} \hat{x}_{t,s,f} + \tilde{c}_{t,s,f} \tilde{x}_{t,s,f} + \sum_{f' \in F_{t,f}} c^E(f,f') x_{t,s,f,f'} \right) \right) \tag{20}$$

$$\text{s.t.} \quad \hat{x}_{t,s,f} + \sum_{f' \in F} x_{t,s,f',f} = \tilde{x}_{t,s,f} + \sum_{f' \in F} x_{t,s,f,f'} \qquad \forall s \in S_t,\ \forall f \in F \tag{21}$$

$$\bar{x}_{t,s} + \sum_{f \in F} \hat{x}_{t,s,f} \geq 1 \qquad \forall s \in S_t \tag{22}$$

$$\hat{x}_{t,s,f} + \sum_{f' \in F_{t,f}} x_{t,s,f',f} \leq y_f \qquad \forall s \in S_t,\ \forall f \in F \tag{23}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{f \in F} \hat{\ell}_{t,s,f} \hat{x}_{t,s,f} \leq \begin{cases} b_t, & \text{if } s = 1 \\ r_{t,s-1}, & \text{else} \end{cases} \qquad \forall s \in S_t \tag{24}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{f \in F} \tilde{\ell}_{t,s,f} \tilde{x}_{t,s,f} \leq b_t - r_{t,s} \qquad \forall s \in S_t \tag{25}$$

$$r_{t,s} + (b_t + \bar{\ell}_{t,s}) \bar{x}_{t,s} \leq \begin{cases} 2b_t, & \text{if } s = 1 \\ b_t + r_{t,s-1}, & \text{else} \end{cases} \qquad \forall s \in S_t \tag{26}$$

$$x_{t,s,i,j} \in \{0,1\} \qquad \forall i \in F, j \in F_{t,i} \tag{27}$$

$$\bar{x}_{t,s} \in \{0,1\} \qquad \forall s \in S_t \tag{28}$$

$$\hat{x}_{t,s,i} \in \{0,1\} \qquad \forall i \in F \tag{29}$$

$$\tilde{x}_{t,s,i} \in \{0,1\} \qquad \forall i \in F \tag{30}$$

$$y_f \in \{0,1\} \qquad \forall f \in F \tag{31}$$

$$r_{t,s} \geq 0 \qquad \forall s \in S_t \tag{32}$$

The objective function minimizes the reduced cost of a pattern variable that belongs to the pattern described by the variables of the pricing problem. The constraints of the pricing problem are the same as the constraints of the arc-based formulation, but limited to a single trip.

2.2.2 Segment Patterns

The second pattern-based formulation (referred to as $\mathcal{A}_S$) decomposes the problem by its segments. It is basically the result of performing a Dantzig-Wolfe decomposition [13] of the arc-based formulation by moving the constraints (2) and (3) into the pricing problems (one problem per segment), except that we explicitly keep the $\bar{x}$-variables (9) in the master problem. Figure 1b visualizes the exploited structure. The segment-pattern-based formulation uses binary pattern variables that represent paths through a specific segment using at least one station. As with the previous formulation, this would result in too many variables, and we use column generation. We chose this formulation because the resulting pricing problems can be solved very efficiently.

Master Problem The master problem of the segment-pattern-based formulation is more complex, since it must ensure that the patterns selected for the segments of a trip are compatible with respect to the length bounds. The main difference is that the segment-pattern-based formulation uses patterns $p \in P_{t,s}$ that describe a (valid) path through a segment $s \in S_t$ of a trip $t \in T$ with cost c_p. For a given pattern p, the new parameters $\hat{\ell}_p \in \mathbb{Q}_{\geq 0}$ and $\tilde{\ell}_p \in \mathbb{Q}_{\geq 0}$ specify the length of the edge from the start of the segment to the first station and from the last station to the end of the segment, respectively. That is, for a pattern p belonging to a segment s of trip t, we have $\hat{\ell}_p = \hat{\ell}_{t,s,i}$ and $\tilde{\ell}_p = \tilde{\ell}_{t,s,j}$ where $i \in F$ is the first station visited and $j \in F$ is the last station visited according to pattern p. Moreover, variables and constraints responsible for finding a path through a segment in the arc-based formulation are removed.

$$\min \quad \sum_{f \in F} c^F(f) y_f + \sum_{\substack{t \in T, \\ s \in S_t}} \left(\bar{c}_{t,s} \bar{x}_{t,s} + \sum_{p \in P_{t,s}} c_p \lambda_p \right) \tag{33}$$

$$\text{s.t.} \quad \bar{x}_{t,s} + \sum_{p \in P_{t,s}} \lambda_p \geq 1 \qquad \forall t \in T,\ \forall s \in S_t \qquad [\pi_{t,s} \geq 0] \tag{34}$$

$$\sum_{\substack{p \in P_{t,s}: \\ f \in F_p}} \lambda_p \leq y_f \qquad \forall t \in T,\ \forall s \in S_t,\ \forall f \in F \qquad [\pi_{t,s,f} \leq 0] \tag{35}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{p \in P_{t,s}} \hat{\ell}_p \lambda_p \leq \begin{cases} b_t, & \text{if } s = 1 \\ r_{t,s-1}, & \text{else} \end{cases} \qquad \forall t \in T,\ \forall s \in S_t \qquad [\hat{\pi}_{t,s} \leq 0] \tag{36}$$

$$\bar{\ell}_{t,s} \bar{x}_{t,s} + \sum_{p \in P_{t,s}} \tilde{\ell}_p \lambda_p \leq b_t - r_{t,s} \qquad \forall t \in T,\ \forall s \in S_t \qquad [\tilde{\pi}_{t,s} \leq 0] \tag{37}$$

$$r_{t,s} + (b_t + \bar{\ell}_{t,s}) \bar{x}_{t,s} \leq \begin{cases} 2b_t, & \text{if } s = 1 \\ b_t + r_{t,s-1}, & \text{else} \end{cases} \qquad \forall t \in T,\ \forall s \in S_t \tag{38}$$

$$\lambda_p \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t,\ \forall p \in P_{t,s} \tag{39}$$

$$\bar{x}_{t,s} \in \{0,1\} \qquad \forall t \in T,\ \forall s \in S_t \tag{40}$$

$$y_f \in \{0,1\} \qquad \forall f \in F \tag{41}$$

$$r_{t,s} \geq 0 \qquad \forall t \in T,\ \forall s \in S_t \tag{42}$$

The constraint (34) corresponds to the constraint (3), (35) to (4), (36) to (5), (37) to (6), and (38) to (7) of the arc-based formulation. For constraints containing pattern variables the corresponding dual variables are given in brackets.

Pricing Problems Each pattern variable corresponds to a particular path through a segment of a trip that visits at least one station. We use a pricing problem for each segment $s \in S_t$ and each trip $t \in T$:

$$\min \quad -\pi_{t,s} + \sum_{i \in F} (\hat{c}_{t,s,i} - \pi_{t,s,i} - \hat{\pi}_{t,s} \hat{\ell}_{t,s}) \hat{x}_{t,s,i} + (\tilde{c}_{t,s,i} - \tilde{\pi}_{t,s} \tilde{\ell}_{t,s}) \tilde{x}_{t,s,i} \tag{43}$$

$$+ \sum_{i \in F} \sum_{j \in F_{t,i}} (c^E(i,j) - \pi_{t,s,j}) x_{t,s,i,j} \tag{44}$$

$$\text{s.t.} \quad \hat{x}_{t,s,j} + \sum_{\substack{i \in F: \\ j \in F_{t,i}}} x_{t,s,i,j} = \tilde{x}_{t,s,j} + \sum_{i \in F_{t,j}} x_{t,s,j,i} \qquad \forall j \in F \tag{45}$$

$$\sum_{f \in F} \hat{x}_{t,s,f} = 1 \tag{46}$$

$$x_{t,s,i,j} \in \{0,1\} \qquad \forall i \in F, j \in F_{t,i} \tag{47}$$

$$\hat{x}_{t,s,i} \in \{0,1\} \qquad \forall i \in F \tag{48}$$

$$\tilde{x}_{t,s,i} \in \{0,1\} \qquad \forall i \in F \tag{49}$$

As before, the objective function minimizes the reduced cost of a pattern variable that belongs to the pattern described by the variables of the pricing problem. The constraint (45) corresponds to the constraint (2) of the arc-based formulation. In addition, the constraint (46) (corresponding to the constraint (3)) ensures that a path is computed by forcing to leave the first node of the segment. Note that every pricing problem is a shortest path problem and can be solved efficiently. Therefore, we create a graph $G_s(V_s, E_s)$ to solve the pricing problem of segment s. The node set V_s contains all possible stations F, an artificial source node, and an artificial sink node. The source node corresponds to the start node of the segment and the sink node corresponds to the end node of the segment. The source has only outgoing edges to station nodes and the sink has

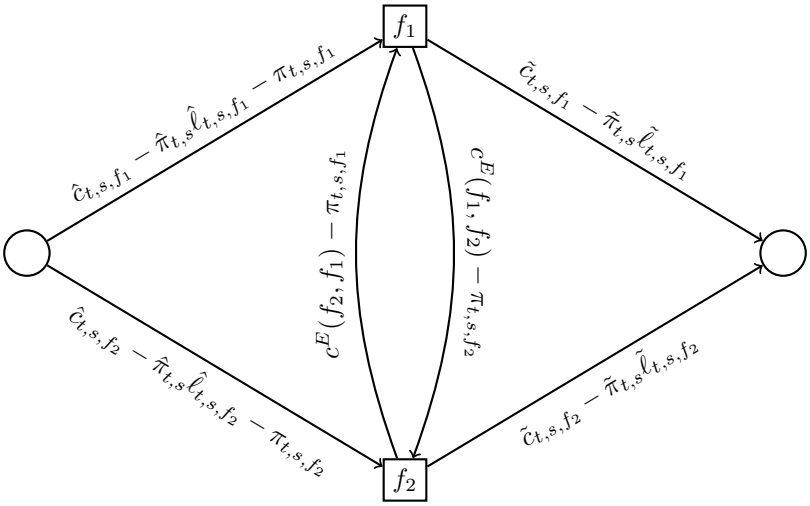


Figure 2: A pricing problem modeled as a shortest path problem. The left circle represents the start node and the right circle represents the end node of the considered paths.

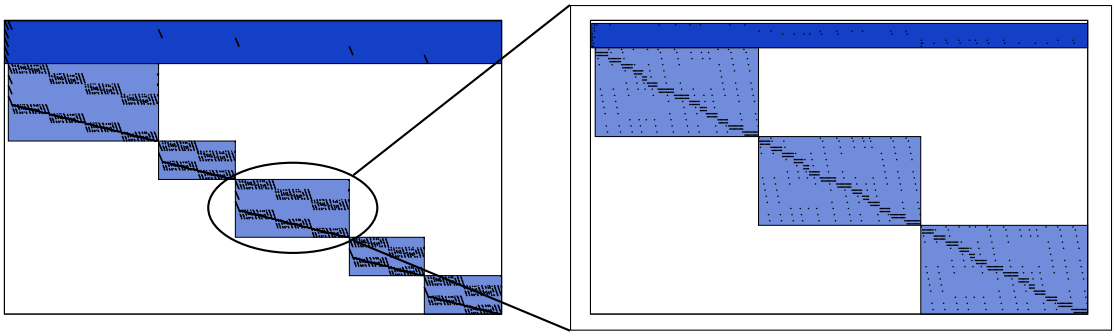


Figure 3: The nested structure of a MSLP instance is visualized using rearranged coefficient matrices. Black dots correspond to non-zero entries, the dark blue boxes represent the master problems, and the light blue boxes represent the pricing problems. On the left is the structure exploited by the outer trip-pattern-based formulation, and on the right is the structure of a single pricing problem exploited by the inner segment-pattern-based formulation.

only incoming edges from station nodes. Furthermore, $E_s \subseteq E$ contains all edges between possible stations that do not violate the length bound. As a result, all paths between two stations in G_s are valid with respect to the length bound. A feasible solution to a pricing problem is a path through the modified graph starting at the source and ending at the sink. The coefficients of the objective function are used to calculate the edge lengths such that the length of a path from the source to the sink is equal to the second part (the sum) of the objective function. Since all coefficients of the variables in the objective function are greater than or equal to zero, we do not have to worry about edges with negative lengths or negative cycles. We can calculate the reduced cost of a variable represented by a path through the graph by subtracting $\pi_{t,s}$ from the length of the path. Figure 2 depicts a small example.

2.2.3 Nested Decomposition Approach

As mentioned in Section 2.2.1, the trip-pattern-based formulation $\mathcal{A}_{\mathrm{T}}$ uses pricing problems that solve the MSLP with respect to a trip and a modified objective function. Hence, we can again exploit the structure of these pricing problems by decomposing them into segments. Figure 3 visualizes how a pricing problem is decomposed. This is equivalent to applying the segment-pattern-based formulation $\mathcal{A}_{\mathrm{S}}$ to a single trip instance. Instead of the actual cost value of a charging station $f \in F$, we use the negative value of the dual variable $\pi_{t,f}$ (constraint (16)) as

the cost value for f. We obtain the reduced cost value by subtracting π_t (constraint (15)) from the objective value. This approach is called $\mathcal{A}_{\mathrm{N}}$.

3 Algorithmic Components

In the previous sections, we introduced several formulations that allow us to solve MSLP instances. The arc-based formulation (Section 2.1) can be solved with a standard MIP solver, while the pattern-based formulations presented in Section 2.2.1, Section 2.2.2, and Section 2.2.3 require branch-and-price algorithms. Our approaches are based on SCIP [4], an open-source solver. In the following sections, we introduce all important algorithmic parts of our branch-and-price and branch-price-and-cut approaches.

3.1 Primal Heuristic

We use a problem specific primal heuristic that works on the original problem. The main idea is to transform the problem into a shortest path problem for each trip by constructing a new graph using a given set of allowed stations. If the constructed graphs are connected, a shortest feasible path is computed for each trip. We have adapted the construction used by the approximation algorithm presented by Willamowski, Ganz, and Mühmer [62]. Instead of using the actual station costs, we set all station costs to 0. To prevent the heuristic from computing the same solution every time, we restrict the set of stations based on the current fractional solution to the restricted master problem. Note that for a given fixed set of stations (i.e., F^* is known in advance), the primal heuristic computes an optimal solution if a feasible solution exists. This is easy to see because the given stations can be used at no additional cost, and by construction a shortest path is computed for each trip such that the length bound is not violated. Thus, the path costs of the trips are minimal with respect to the provided stations. After the primal heuristic is finished, a solution to the master problem is created. Since the heuristic works on the original problem, it may find solutions that cannot be represented by the current variables of the restricted master problem. In this case, we generate them and add them to the restricted master problem. Additionally, SCIP's (default) primal heuristics are enabled and work on the restricted master problem.

3.2 Pricing

The following sections describe how new variables are created by the pricing process. First, Farkas pricing [1] is used whenever our current LP is infeasible. This can happen at the root node (if there is no trivial solution) or after branching. Then, if necessary, the current node is solved by iteratively solving the restricted master LP and pricing new variables with standard (reduced) cost pricing [1].

3.2.1 Farkas Pricing

Farkas pricing allows new variables to be priced if the restricted master problem is currently infeasible. Instead of using the textbook Farkas pricing procedure, we use the algorithm described in Section 3.1 since it produces a feasible solution if and only if the problem instance is feasible. We distinguish two cases. First, if the restricted master problem does not contain any priced variables at the beginning of the solving process. In this case, we run the algorithm with multiple configurations to generate a set of initial patterns. We use static station costs (set to 0 and to the actual costs) as well as dynamic station costs, which are iteratively set to 0 (see Willamowski, Ganz, and Mühmer [62]). For each subsequent call to the Farkas pricing procedure, the primal heuristic is run with all stations that are not disabled (e.g., by branching) and costs set to 0. After that, we know either that the current node is infeasible, or which patterns we can add to make the LP feasible.

3.2.2 Reduced Cost Pricing

Depending on the used pattern-based formulation, the reduced cost pricing procedure differs in whether the primal heuristic (see Section 3.1) is called. When we solve a problem using $\mathcal{A}_{\mathrm{T}}$ or $\mathcal{A}_{\mathrm{N}}$, the primal heuristic is called at each pricing iteration at the root node. In contrast, when we use $\mathcal{A}_{\mathrm{S}}$, we run the primal heuristic once at each node of the branch-and-price tree. We decided to distinguish these cases because preliminary experiments show that using $\mathcal{A}_{\mathrm{S}}$ typically results in larger trees and was able to benefit from solutions found by the heuristic during branching. In contrast, $\mathcal{A}_{\mathrm{T}}$ and $\mathcal{A}_{\mathrm{N}}$ result in smaller trees and much higher single node processing times. Running the primal heuristic later in the tree did not improve performance when using these formulations. For all formulations, we select the allowed stations provided to the heuristic by evaluating the current solution to the restricted master LP. All stations whose variables have a value of at least $\tau \in \mathbb{Q}_{\geq 0}$ in the current LP solution can be selected by the heuristic. Preliminary tests have shown that invoking the heuristic with the threshold $\tau = 0.5$ yields comparable results to other thresholds or even multiple invocations with different thresholds. If the heuristic is successful (i.e., the heuristic found a valid solution) and we need new pattern variables, they are added to the restricted master problem and the primal bound is updated if necessary. Moreover, if all station variables are fixed, we do not need to price new variables since the heuristic computes an optimal solution for the specific set of stations. Otherwise, we price new variables by solving the pricing problems using the current dual information. Then, the priced variables are added to the restricted master problem and we update the current lower bound by computing the Lagrangian lower bound using the results of the pricing problems and the current objective value of the LP relaxation.

Outline of the main pricing procedure:

1. Run the heuristic with the allowed edges and all stations that currently have a solution value of at least τ. Do this
   - once at every tree node for $\mathcal{A}_{\mathrm{S}}$ or
   - only at the root node for $\mathcal{A}_{\mathrm{T}}$ and $\mathcal{A}_{\mathrm{N}}$.
2. Check whether all stations are fixed (e.g., due to branching).
   - If so, no pricing is required. The result of the heuristic is processed and the node is marked as solved.
3. Solve the pricing problems parameterized with the current dual information.
4. Update the current lower bound based on this iteration.

3.3 Branching

In addition to the column generation approach, we have implemented multiple branching strategies so that branch-and-price can be applied. At every tree node we decide which branching strategy should be used. This is done by static rules and a pseudo cost branching scheme. First, we introduce all types of branching candidates. Then, we explain how a branching candidate is selected.

3.3.1 Branching on Station Variables

The first type of branching candidates is used by $\mathcal{A}_{\mathrm{T}}$, $\mathcal{A}_{\mathrm{S}}$, and $\mathcal{A}_{\mathrm{N}}$. All fractional station variables y_f with $f \in F$ are branching candidates. When such a candidate is selected, we branch directly on the corresponding variable. Since these are binary variables, the branching decisions correspond to selecting or forbidding the station f, i.e., the variable is fixed to 1 or 0 in the master problem. In addition, the pricing problems can no longer use this station if it is forbidden. This can be achieved by fixing the corresponding variables (of the pricing problems) to 0 or by removing the corresponding node from the graph so that it cannot be selected by the shortest path algorithm. For $\mathcal{A}_{\mathrm{T}}$ and $\mathcal{A}_{\mathrm{N}}$ it is sufficient to branch on station variables. We can verify this claim by considering the case where we have a fractional solution in which all station variables are integer.

This means that at least for one trip $t \in T$ we have a set of fractional pattern variables Λ_t^{frac}. Furthermore, because of (15), it holds that $|\Lambda_t^{\text{frac}}| > 1$. The objective coefficient of all pattern variables in Λ_t^{frac} have to be the same as otherwise we would get a better solution by choosing the pattern variable with the smallest cost coefficient and setting it to 1 and the others to 0. This is possible without losing feasibility or increasing other costs because the station variables are already equal to 1 for all used stations. Hence, we can construct an alternative feasible solution that has the same cost by setting one variable of Λ_t^{frac} to 1 and all the others to 0. We can repeat this procedure for all affected trips to get an integer solution with the same objective value. However, this does not hold for $\mathcal{A}_{\text{S}}$. Branching on station variables would be sufficient even for $\mathcal{A}_{\text{S}}$ if we also branch on integer but unfixed variables because the primal heuristic computes an optimal solution for the active branch when all station variables are fixed (as stated in Section 3.1). However, branching on variables that are already integer is very inefficient. Only if no other branching candidate is available, we branch on unfixed (integer) station variables, which we call *fallback branching*.

3.3.2 Branching on Shortcut Variables

In addition to fractional station variables, $\mathcal{A}_{\text{S}}$ may encounter fractional shortcut variables $\bar{x}_{t,s}$ with $t \in T$, $s \in S_t$. Therefore, we also use these variables as branching candidates. When such a candidate is selected, we branch directly on the corresponding variable. Since these are binary variables, the branching decisions correspond to forbidding any station or enforcing at least one station in the segment s, i.e., the variable is fixed to 1 or 0 in the master problem. The pricing problem of the corresponding segment does not need to be called anymore if the shortcut variable is fixed to 1 (and all pattern variables of this segment can be fixed to 0).

3.3.3 Branching on Distances

The last branching candidates, used exclusively by $\mathcal{A}_{\text{S}}$, are related to fractional pattern variables λ_p with $p \in P_{t,s}$, $t \in T$, and $s \in S_t$. We have found that fractional master variables often help to satisfy the range constraints. So, we look for a fractional path (for a fixed trip t) that would violate the range in the integer case. Such a path starts with leaving a segment $s_1 \in S_t$ and ends with entering a segment $s_2 \in S_t$ with $s_1 < s_2$. All segments between s_1 and s_2 must not visit any station, i.e., for all segments $s' \in S_t$ with $s_1 < s' < s_2$ the variable $\bar{x}_{t,s}$ must be fixed to 1. Figure 4 shows an example of such a situation and visualizes the idea of these branching candidates. We assume that we have found such a path starting with pattern $p_1 \in P_{t,s_1}$ and ending with pattern $p_2 \in P_{t,s_2}$. Let

$$\bar{\ell}_{t,s_1,s_2} = \sum_{\substack{s' \in S_t: \\ s_1 < s' < s_2}} \bar{\ell}_{t,s'}$$

be the length of the segments between s_1 and s_2. The branching is realized by creating two child nodes and each node forbids a set of pattern variables. We calculate two length bounds (50) and (51) to define the two sets $B_1 = \{\lambda_p \mid p \in P_{t,s_1} \wedge \tilde{\ell}_p \leq b_1\}$ and $B_2 = \{\lambda_p \mid p \in P_{t,s_2} \wedge \hat{\ell}_p \leq b_2\}$ such that $\lambda_{p_1} \in B_1$, $\lambda_{p_2} \in B_2$, and $\lambda_{p_1}, \lambda_{p_2} \notin B_1 \cap B_2$.

$$b_1 = \tilde{\ell}_{p_1} - \frac{\tilde{\ell}_{p_1}}{\tilde{\ell}_{p_1} + \hat{\ell}_{p_2}} \left(\tilde{\ell}_{p_1} + \hat{\ell}_{p_2} + \bar{\ell}_{t,s_1,s_2} - b_t \right) \tag{50}$$

$$b_2 = b_t - b_1 - \bar{\ell}_{t,s_1,s_2} \tag{51}$$

Figure 5 depicts the branching decision (in the master problem). No feasible solution is excluded because we have $b_t = b_1 + b_2 + \bar{\ell}_{t,s_1,s_2}$ and, thus, solutions that do not belong to any branch would violate the length bound. In the pricing problem, we need to ensure that we do not generate patterns that violate these bounds. Therefore, we remove all edges from the graph that do not satisfy the branching decision, i.e., all starting or ending edges that are too long. As a result, we can still use our shortest path approach to generate new patterns.

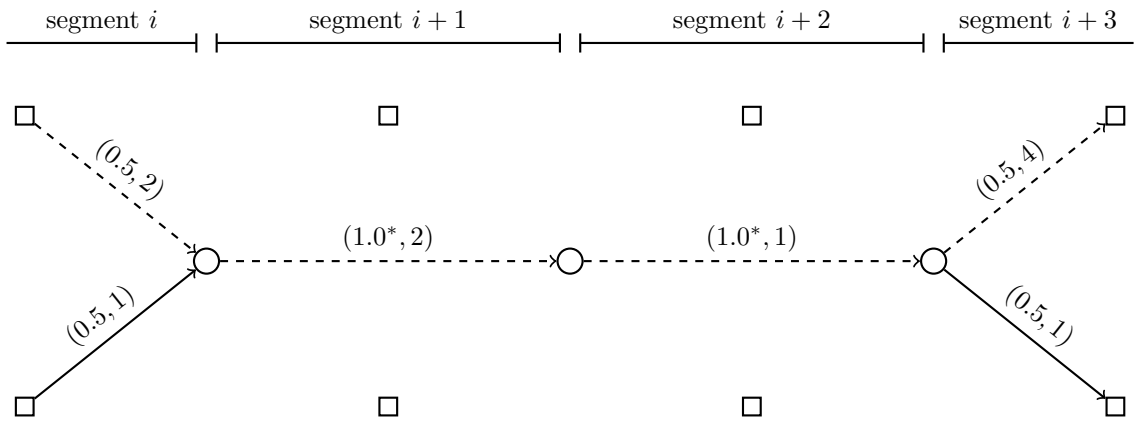


Figure 4: A part of a fractional solution is visualized. The graph shown is not the underlying instance graph. The circles represent stops of a trip (two consecutive stops belong to a segment as start and end). The rectangles are copies of station nodes. For each segment all possible stations are available (here we have 2 stations). The edges show the (fractional) path used by the current solution. A tuple (a, b) corresponds to the (fractional) value a of the solution variable and to the length b of the edge. Solution values marked with a $*$ are fixed to this value. The dashed edges belong to a path that would be invalid in the integer case. In this example the length bound b_t is 7.

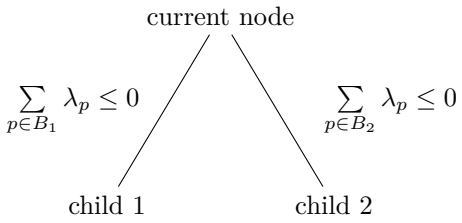


Figure 5: The nodes created with respect to a distance branching candidate are shown. The constraint of each branch can be enforced by fixing the affected pattern variables to 0.

3.3.4 Selection of a Branching Candidate

At each node that requires branching, we must decide which branching candidate should be used. All approaches rely on a branching score calculated using strong branching [1] and pseudo cost [1]. We use a hierarchical selection process that works as follows:

1. Select a best station variable branching candidate according to the calculated branching score.
2. If no fractional station variable is available, select a best available branching candidate based on the branching score.
3. If no branching candidate is available, perform fallback branching.

The approaches $\mathcal{A}_{\mathrm{T}}$, $\mathcal{A}_{\mathrm{S}}$, and $\mathcal{A}_{\mathrm{N}}$ only need to perform the first step because a fractional solution always contains a fractional station variable. In contrast, $\mathcal{A}_{\mathrm{S}}$ may need to perform the other steps as well. We prefer branching on station variables because we observed that fixing station variables helps the primal heuristic (see Section 3.1). This is plausible since the algorithm relies heavily on the provided set of stations. As a result, the overall solving process can benefit from a strong primal bound during branching. If no fractional station variable is available, we consider the other branching candidates.

To select a branching candidate, we rely on pseudo cost values [1]. For all branching candidates, pseudo cost values are maintained that are used to predict the objective gains if we branch on

a candidate. Let $j \in \{1, 2\}$ be a branch's direction (first or second branch). Then, the pseudo cost value $\psi_{h,j}$ is the normalized objective gain we expect when we branch on candidate h. For a branch, we know the fractional part $\delta_{h,j}$ of the solution values restricted by that branch. For example, if we branch on a station variable with value 0.4, the fractional parts are 0.4 and 0.6 (first branch and second branch). Using these values, we can calculate the predicted objective gain (52) of a branch.

$$g_{h,j} = \psi_{h,j} \cdot \delta_{h,j} \tag{52}$$

Whenever we have to decide between branching candidates, we calculate the branching score

$$\sigma_h = g_{h,1} \cdot g_{h,2} \tag{53}$$

for each branching candidate h. We choose the candidate with the highest branching score. After branching we need to update the pseudo cost values. The experienced objective value gain is used to update the pseudo cost values by calculating

$$\psi_{h,j}^{\text{new}} = \psi_{h,j} + \frac{1}{k}\left(\frac{z_{\text{lp}}^{\text{child}} - z_{\text{lp}}^{\text{parent}}}{\delta_{h,j}} - \psi_{h,j}\right), \tag{54}$$

where k is the number of branchings performed on branching candidate h and $z_{\text{lp}}^{\text{child}}$ and $z_{\text{lp}}^{\text{parent}}$ are the objective values of the master problem's relaxation of the child and parent nodes, respectively. If we have branched only a few times (or not at all) on a branching candidate, we have no reliable pseudo cost values available (all are initially zero). Then, strong branching is used to compute the objective gains. If we have branched at least five times on that candidate, we use the pseudo cost values to predict the objective gains.

It may happen that no branching candidate (of the previously introduced types) is available at all. In this case, we fix an unfixed station variable to 0 and 1. This is always possible since we can compute the optimal objective value for a fixed set of (selected) stations (see Section 3.1), i.e., if all station variables are fixed we do not need to branch.

3.4 Cutting Planes

We use cuts in the master problem of $\mathcal{A}_{\text{S}}$ that follow the idea described in Section 3.3.3 and are similar to knapsack cover cuts [1]. The cuts exploit (fractional) paths that would be infeasible in the integer case. We assume that (for a fixed trip t) we have found a fractional path starting at segment s_1 using variable λ_{p_1} and ending at segment s_2 using variable λ_{p_2} with

$$\bar{\ell}_{t,s_1,s_2} = \sum_{\substack{s' \in S_t: \\ s_1 < s' < s_2}} \bar{\ell}_{t,s'}.$$

If

$$\tilde{\ell}_{p_1} + \hat{\ell}_{p_2} + \bar{\ell}_{t,s_1,s_2} > b_t$$

and

$$\lambda_{p_1} + \lambda_{p_2} + \sum_{\substack{s' \in S_t: \\ s_1 < s' < s_2}} \bar{x}_{t,s'} > |\{s' \in S_t \mid s_1 \le s' \le s_2\}| - 1,$$

we can add the valid cut (55).

$$\lambda_{p_1} + \lambda_{p_2} + \sum_{\substack{s' \in S_t: \\ s_1 < s' < s_2}} \bar{x}_{t,s'} \le |\{s' \in S_t \mid s_1 \le s' \le s_2\}| - 1 \tag{55}$$

In addition to λ_{p_1} and λ_{p_2}, we can include other pattern variables belonging to s_1 and s_2 in the cut if the corresponding patterns would also violate the length bound. Hence, we generalize this cut to (56) by choosing the bounds b_1 and b_2 such that $b_1 + b_2 + \bar{\ell}_{t,s_1,s_2} \ge b_t$.

$$\sum_{\substack{p \in P_{t,s_1}: \\ \tilde{\ell}_p > b_1}} \lambda_p + \sum_{\substack{p \in P_{t,s_2}: \\ \hat{\ell}_p > b_2}} \lambda_p + \sum_{\substack{s' \in S_t: \\ s_1 < s' < s_2}} \bar{x}_{t,s'} \le |\{s' \in S_t \mid s_1 \le s' \le s_2\}| - 1 \tag{56}$$

Finally, a path can start or end with a shortcut instead of a pattern. Let

$$X^{b_1,b_2}_{t,s_1,s_2} = \{\bar{x}_{t,s'} \mid s' \in S_t,\ s_1 < s' < s_2\} \cup \{\bar{x}_{t,s_1} \mid \bar{\ell}_{t,s_1} > b_1\} \cup \{\bar{x}_{t,s_2} \mid \bar{\ell}_{t,s_2} > b_2\}$$

be the set of all shortcut variables relevant for this extension. We can then generalize (56) to (57).

$$\sum_{\substack{p \in P_{t,s_1}: \\ \tilde{\ell}_p > b_1}} \lambda_p + \sum_{\substack{p \in P_{t,s_2}: \\ \hat{\ell}_p > b_2}} \lambda_p + \sum_{\bar{x} \in X^{b_1,b_2}_{t,s_1,s_2}} \bar{x} \leq |\{s' \in S_t \mid s_1 \leq s' \leq s_2\}| - 1 \tag{57}$$

These cuts are valid and do not cut any feasible integer solution because for every feasible integer solution there exists a segment $s \in S_t$ with $s_1 \leq s \leq s_2$ for which all related variables in the cut are zero since otherwise the range bound would be violated. Thus, the sum of the variable values of any cut is at most $|\{s' \in S_t \mid s_1 \leq s' \leq s_2\}| - 1$. Since these cuts are added to the master problem, the pricing problems must handle the new dual values corresponding to the added master cuts. Let π_r be the associated dual value of cut r, which is added as a new row to the master problem. We need to add

$$-\sum_{\substack{i \in F: \\ \tilde{\ell}_{t,s_1,i} > b_1}} \pi_r \tilde{x}_{t,s_1,i}$$

to the objective function of the pricing problem of s_1 and

$$-\sum_{\substack{i \in F: \\ \hat{\ell}_{t,s_2,i} > b_2}} \pi_r \hat{x}_{t,s_2,i}$$

to the objective function of the pricing problem of s_2 (see Section 2.2.2). Therefore, in the graph used to solve a pricing problem, we must adapt the lengths of the edges entering the sink or leaving the source if they violate the length bounds b_1 or b_2, respectively. This modification cannot lead to negative distances in the graph because $\pi_r \leq 0$. Thus, we do not need to worry about negative lengths when solving a pricing problem even in the presence of these cuts.

Note that these cuts can also be obtained by combining constraints of the types (36), (37), and (38) and using an idea described by Wolsey [63]. For a trip $t \in T$ and two segments $s_1,\ s_2 \in S_t$ with $s_1 \leq s_2$, we obtain a knapsack constraint by combining the constraint of type (37) of segment s_1, the constraints of type (38) of all segments $s' \in S_t$ with $s_1 < s' < s_2$, and the constraint of type (36) of segment s_2. Due to the constraints (34), any sum of pattern and shortcut variables of a segment is bounded from above by 1. Thus, we can apply the idea described by Wolsey [63] to the obtained knapsack constraint with respect to the upper bounds implied by (34).

Furthermore, the cuts can also be translated to the arc-based formulation. By replacing the pattern variables in (57), we get (58).

$$\sum_{\substack{f \in F: \\ \tilde{\ell}_{t,s_1,f} > b_1}} \tilde{x}_{t,s_1,f} + \sum_{\substack{f \in F: \\ \hat{\ell}_{t,s_2,f} > b_2}} \hat{x}_{t,s_2,f} + \sum_{\bar{x} \in X^{b_1,b_2}_{t,s_1,s_2}} \bar{x} \leq |\{s' \in S_t \mid s_1 \leq s' \leq s_2\}| - 1 \tag{58}$$

Thus, we can separate the cuts while solving the arc-based formulation as well as while solving the pricing problems of $\mathcal{A}_{\mathrm{T}}$.

The cuts are separated at each tree node and the separation is done heuristically for each trip. To find a violated cut, we move a start and end segment of a potential cut through the trip such that the path between them violates the length bound of the trip. The total length of the path between the start and end segment is computed using the (start and end) patterns that add the largest edge lengths and belong to a variable that is assigned a nonzero value in the current solution. For each potential cut found, we check if the cut is actually violated and add it if so. Note that this procedure may miss violated cuts.

3.5 Preprocessing

Finally, we added a preprocessing step to reduce the size of the original problem instance by removing constraints of types (4), (35), and (16) that will not be active in an optimal solution. I.e., if the trip does not use the station in any optimal solution, we can forbid this assignment. This preprocessing exploits the triangle inequality with respect to c^E. We identify such trip-station pairs by using a solution provided by the heuristic, which is run at the beginning of the solving process. The preprocessing considers each trip separately. The idea is that assigning a station to a distant segment can cause such high detour costs so that the path of a known solution for that trip dominates all possible paths that use this assignment, even if no other trip uses any of the selected stations. I.e., if the minimum cost of a path visiting a station in a particular segment is greater than the sum of the installation costs of the used stations and the cost of a path used in another (valid) solution (with respect to a single trip), we can forbid that station for this segment. If this is true for all segments of a trip, we can forbid the station for the entire trip.

3.6 Implementation

We solve the arc-based formulation using Gurobi 9.5.2 [20]. Gurobi's C++ interface allows us to create and solve models without any further technical hurdles. For our branch-price-and-cut implementation we heavily rely on SCIP 8.0.2 [4], which provides a C/C++ API. SCIP is an open-source solver for mixed-integer linear and nonlinear programming, among others. Our solver was built on Debian 11 using g++ 10.2.1.

For all implemented approaches, we exclude all variables belonging to edges that violate the length bound of the trip. In addition, we rely on presolving provided by Gurobi and SCIP. Thus, the formulations are strengthened by the specific solver or framework before the main solving process starts. For example, redundant variables, such as the range variables assigned to the last segment of a trip, are removed.

We added our primal heuristic and our problem specific cuts to Gurobi. Therefore, we implemented a callback and registered it with Gurobi. The heuristic is run once at every tree node and the cut separation is run whenever Gurobi has solved the LP relaxation to optimality.

For our branch-and-price implementation, we use SCIP because it provides a branch-price-and-cut framework that allows us to focus on the important and problem specific parts. SCIP consists of plugins that add functionality to the solver. For instance, we can implement our own pricer plugin and add it to SCIP. Moreover, we use the Boost Graph Library [53] (version 1.75) to implement the shortest path pricing problems and the heuristic. Parts of our implementation are parallelized using OpenMP [45]. In the following, we describe the main parts of our implementation.

Our implementation works with a SCIP object that manages the (restricted) master problem, i.e., we do not maintain any SCIP structures for the original problem. We include all default plugins but disable SCIP's separation functionality. Additionally, we use several plugins to inject our implementation into SCIP's solving process. The main component is our pricer, which performs Farkas and reduced cost pricing. It manages the pricing of new variables by invoking the pricing problem. During the solving process, the pricer plugin is called by SCIP via designated callbacks.

As described before, we do not use SCIP's default branching strategy. Therefore, we register a branching rule with SCIP that is called before SCIP's default branching rules are executed. Our implementation uses SCIP's strong branching and pseudo cost features, i.e., we evaluate multiple branching candidates by using SCIP's branching score, which is computed based on previous branching decisions and strong branching evaluations. After selecting a branching candidate, the branching rule creates new tree nodes accordingly and attaches branching constraints to each new node. SCIP cannot handle these branching constraints by itself. Therefore, we have implemented a constraint handler that is responsible for enforcing the constraints. SCIP tells the constraint handler which constraints are activated or deactivated. This allows us to fix variables and change the pricing problems accordingly.

SCIP uses so-called separators to generate cutting planes to strengthen the LP relaxation. As

mentioned before, we have disabled all default separators and only use our own cuts. These cuts are separated by our separator, which searches for possible cuts. The separator works on multiple trips in parallel and is called at each node of the tree.

The implementation of $\mathcal{A}_{\mathrm{S}}$ maintains data structures for all pricing problems. Each pricing problem is represented by a graph and a shortest-path algorithm is used to compute the objective of the pricing problem (using the Boost Graph Library). By default, each graph is cached and only modified for each pricing iteration. For large problems caching can be disabled to lower memory requirements. Since the shortest-path algorithm computes only one path, every pricing problem generates at most one new variable. In addition, since the pricing problems are solved in parallel, we ensure that new variables are added to the master problem in a deterministic way so that runtime variations do not affect the solving process. For the implementation of the pricing problems used by $\mathcal{A}_{\mathrm{T}}$, we use the implementation of $\mathcal{A}_{\mathrm{A}}$, i.e., the pricing problems are solved by Gurobi using the arc-based formulation, the primal heuristic, and the cuts. Similarly, the implementation of $\mathcal{A}_{\mathrm{N}}$ uses $\mathcal{A}_{\mathrm{S}}$ as solver for the pricing problems.

The primal heuristic we use is registered as a plugin in SCIP and implements the corresponding callbacks. Since the heuristic can find solutions that cannot be represented with the current set of variables of the restricted master problem, it is not run when SCIP calls it, but only makes previously found solutions available to SCIP. Instead, our pricer calls the heuristic during the pricing so that we can add missing variables. The heuristic processes multiple trips in parallel if possible (i.e., if static station costs are used).

4 Experiments

We conducted experiments to compare the performance of the arc-based formulation and the extended formulation solved with branch-price-and-cut. In addition, we investigate to what extent the problem specific cuts of the extended formulation affect the performance. First, we present the instance sets and the platforms used. We then evaluate and discuss the results.

4.1 Platform and Instances

For our experiments, we generated random instances based on instances of the TSPLIB [47], excluding instances with more than 500 nodes. We processed the TSP instances using the Python library tsplib95 (version 0.7.1). Each instance inherits the graph of the corresponding TSP instance and the cost of an edge is equal to its length, so $\ell = c^E$. A station can be placed at any node (i.e., $F = V$). Let $c_{\text{trips}} = \sum_{t \in T} \sum_{s \in S_t} \bar{c}_{t,s}$ be the total cost of all trips without detours. Then, we balance the trip and station costs such that the cost of building 20% of the stations is equal to c_{trips} (i.e., 20% of the stations are worth the same as a detour that doubles the trip cost). Hence, we set the installation cost of all stations $f \in F$ to

$$c^F(f) = 5\frac{c_{\text{trips}}}{|F|}.$$

We have created two instance sets, I_1 and I_2. Both instance sets contain the same number of instances with the same sizes (in terms of number of nodes and number of trips): for each TSP instance we created instances with 10, 20, 30, and 40 trips. In addition, for both instance sets and for each trip, the number of stops was chosen uniformly at random from $[2, \sqrt{|V|}]$. For instances of I_1, a stop of a trip was chosen uniformly at random from the set of all nodes without its direct predecessor. In contrast, for instances of I_2, a (new) stop of a trip was chosen based on a normal distribution and based on the distances to the last three (already chosen) stops. Let ℓ_{trips} be the total length of all trips, i.e., $\ell_{\text{trips}} = c_{\text{trips}}$. We use the same length bound

$$b_t = \left\lfloor \frac{2}{3} \frac{\ell_{\text{trips}}}{|T|} \right\rfloor$$

for all trips $t \in T$, which ensures that a trip of average length cannot be completed without visiting at least one station. We ran these experiments on Debian 11 computing nodes equipped with two

Intel Xeon L5630 processors (providing 16 logical processors in total) and 128 GB of DDR3 RAM. We enforced a time limit of one hour.

Moreover, we created a large instance based on real-world data. For this purpose, we used public GTFS data [41], containing (public) transportation schedules (including stops, routes, etc.). We decided to use data of an intercity bus service. The data contain trips offered in Europe from January to March 2020. Since the raw data are not suitable for our purpose, we preprocessed the data. First, we clustered the stops based on their zip code and distance to each other, so that we ended up with 1644 nodes. We also removed trips that are completely contained within other trips and trips that are shorter than the length bound. After preprocessing, we were left with 2408 trips and 17405 segments in total. Furthermore, all nodes are possible locations for stations. We computed shortest paths between all nodes in terms of travel time. The lengths ℓ of the edges correspond to the length (in kilometers) of such a shortest path and the costs c^E to the travel time in minutes. Furthermore, we balanced station cost against trip cost by setting the cost of one station to 15% of the average length (in terms of travel time) of all trips, i.e.,

$$c^F(f) = \left\lfloor 0.15 \frac{c_{\text{trips}}}{|T|} \right\rfloor = 65$$

for all stations $f \in F$. Thus, if the travel time of a trip would increase higher than this value, it is beneficial to install a proper station (which does not imply a detour). For all pairs of nodes, we computed a route that connects these nodes and minimizes the travel time. We set the length of an edge between two nodes to the length of the route and the cost of the edge to the travel time. The distances between the nodes correspond to their actual distances in kilometers. Since the range of electric buses is roughly between 90 and 550 kilometers [3, 46], we set the length bound to 250 kilometers. We ran this instance (without a time limit) on a Debian 11 workstation equipped with an Intel i7-8700 processor (providing 12 logical processors in total) and 32 GB of DDR4 RAM.

4.2 Results and Discussion

In the following, we present and discuss the results of the conducted experiments. The raw data and detailed figures for each instance set and size category can be found in Appendix A. In addition to the presented approaches, we evaluated the performance of $\mathcal{A}_{\text{S}}$ without separating the cuts presented in Section 3.4. By doing so, we want to investigate the effect of our cuts on the performance. We refer to $\mathcal{A}_{\text{S}}$ without cut separation as $\mathcal{A}'_{\text{S}}$. Furthermore, we checked whether our primal heuristic and cuts help Gurobi by disabling both features. We call this modification $\mathcal{A}'_{\text{A}}$. Since our experiments show that Gurobi benefits from our heuristic and cuts, we do not include the results of $\mathcal{A}'_{\text{A}}$ in this section, but they are included in Appendix A. In the following, all box plots show the first, second, and third quartiles of the data. Moreover, the whiskers in a box plot visualize the range of all data points that fall within the range of the first quartile minus 1.5 times the interquartile range and the third quartile plus 1.5 times the interquartile range. Outliers are not shown unless otherwise stated. Figure 6 shows the performance profiles with respect to the solving time and Figure 7 shows the aggregated achieved relative gaps (after an approach terminated). The relative gap is defined as

$$gap = \frac{\bar{z} - \underline{z}}{\underline{z}},$$

where $\bar{z}$ is the best found primal bound and $\underline{z}$ is the best computed dual bound. If instances are infeasible and the solver detected this within the time limit, the gap is recorded in the data as 0%.

The results show that $\mathcal{A}_{\text{N}}$ and $\mathcal{A}_{\text{S}}$ outperform the other approaches, as they typically terminated earlier with an optimal solution or achieved a smaller gap when the time limit was reached. $\mathcal{A}_{\text{S}}$ was the fastest approach on the largest number of instances (45.5%), followed by $\mathcal{A}_{\text{N}}$ (40.0%). However, $\mathcal{A}_{\text{N}}$ outperforms $\mathcal{A}_{\text{S}}$ in terms of increasing performance ratios. In particular, it solved most of the instances to optimality within the time limit (80.5%). This indicates that $\mathcal{A}_{\text{N}}$ behaves

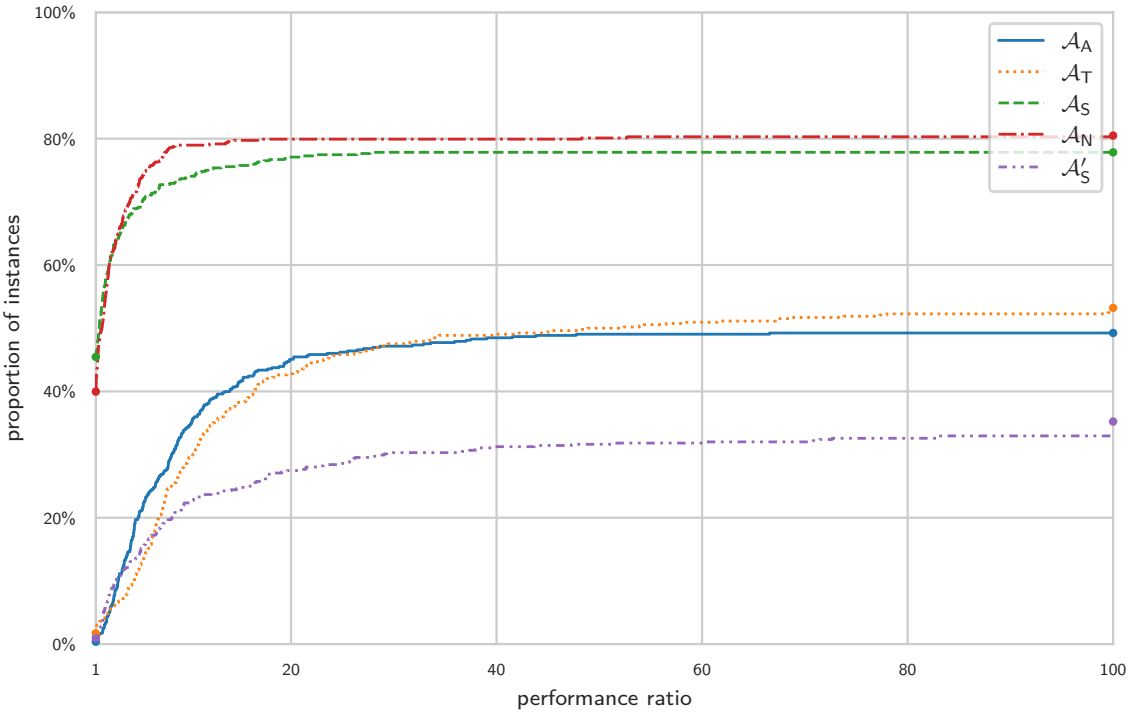


Figure 6: Performance profile with respect to the solving times. The x-axis is truncated for better readability. The right dot of each line corresponds to the proportion of instances solved to optimality within the time limit.

more "stable," i.e., its performance does not vary as much across instances of the same size. In addition, the results show that the cuts contribute greatly to the performance of $\mathcal{A}_S$. Without cut separation, $\mathcal{A}_T$ and $\mathcal{A}_A$ outperform $\mathcal{A}'_S$ (except for small performance ratios). Note that $\mathcal{A}'_A$ (arc-based formulation without our primal heuristic and cuts) also outperforms $\mathcal{A}'_S$, but the difference is smaller. The picture is similar if we look at the remaining gaps when the approach terminated (either optimal or when the time limit was reached). Figure 8 shows the distribution of the remaining gaps for all approaches. Both approaches, $\mathcal{A}_N$ and $\mathcal{A}_S$, achieve better results even when the time limit was reached. Moreover, $\mathcal{A}_T$ typically terminated with a smaller remaining relative gap compared to $\mathcal{A}_A$ and $\mathcal{A}'_S$. This is consistent with the fact that $\mathcal{A}_T$ managed to solve more instances to optimality within the time limit than $\mathcal{A}_A$ and $\mathcal{A}'_S$. Note that the maximum remaining gap achieved by $\mathcal{A}_A$ is infinite, since it was unable to compute the required bounds for some instances.

In the following, we compare the qualities of the primal and dual bounds. Since cutting planes cut off fractional points that do not belong to the integer solution space, it is reasonable that the dual bounds computed by $\mathcal{A}'_S$ are worse than those computed by $\mathcal{A}_S$, while the primal bounds computed by $\mathcal{A}'_S$ may still be of high quality. To verify this intuition, we computed (for all instances for which we know the optimal objective value) the fraction of (what we call) the primal part of the total gap

$$pp = \frac{\bar{z} - z^*}{\bar{z} - \underline{z}},$$

where z^* denotes the optimal objective value. Thus, a small value ($pp < 0.5$) indicates that the primal bound is of better quality than the computed dual bound. Figure 9 shows box plots summarizing the achieved fractions. Note that the purpose of this visualization is to compare the qualities of the primal and dual bounds of a single approach, not to compare two approaches. All approaches tend to compute better primal bounds than dual bounds. In particular, the fractions achieved by $\mathcal{A}_S$ and $\mathcal{A}'_S$ are very small and the computed primal bounds are typically very close to the optimal objective value. When we relate this to the achieved gaps, we see that $\mathcal{A}'_S$ finds very

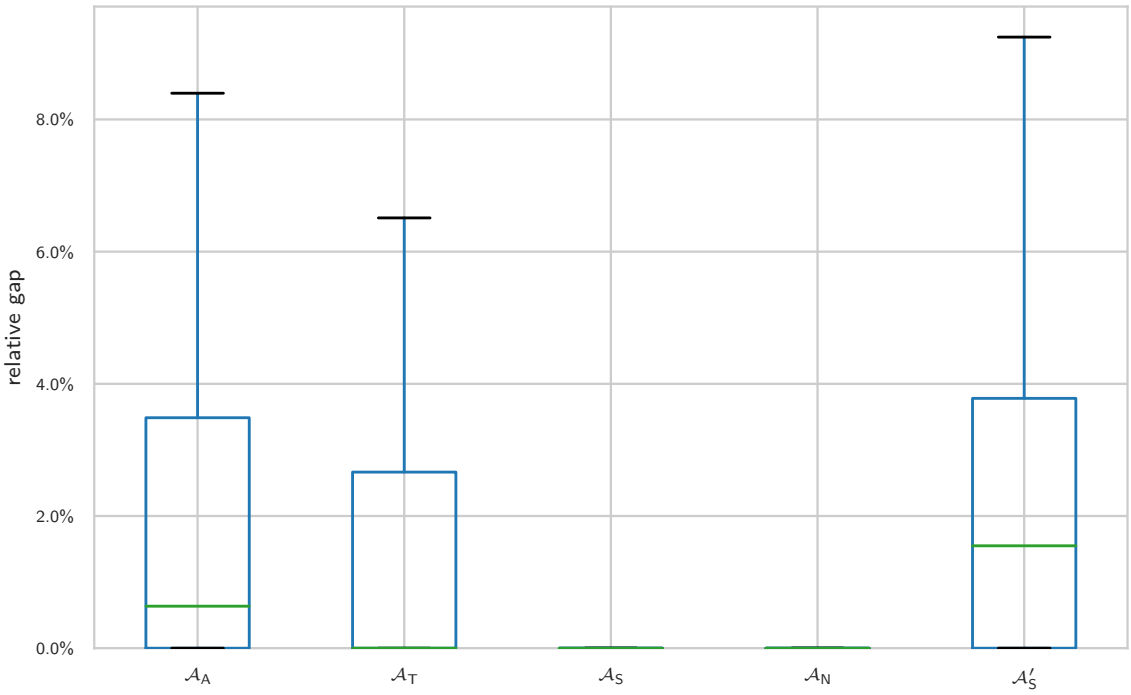


Figure 7: The gaps achieved after the approaches terminated are summarized by a box plot for each approach.

good or close to optimal solutions, but fails to compute good dual bounds to prove optimality.

To take a closer look at this phenomenon, Figure 10 shows the distributions of the applied branching strategies and Figure 11 shows the total number of tree nodes (for the approaches $\mathcal{A}_S$ and $\mathcal{A}'_S$). The visualized data shows that $\mathcal{A}_S$ created most of the tree nodes by branching on fractional station variables. Much smaller proportions of nodes were created by branching on shortcut variables, followed by branching on distances. The proportions shift a bit to branching on shortcut variables and distances when the number of trips increases. In contrast, when no cuts were separated, shortcut variable branching, distance branching, and even fallback branching were needed more often. While $\mathcal{A}'_S$ branched on shortcut variables or distances more often than $\mathcal{A}_S$ for the majority of instances, fallback branching was only applied for a few instances. However, when fallback branching was used, it was used for a significant fraction of branching decisions. For more details, see Figure 17 and Figure 18, which show the results for each instance. In summary, the presented results show that the cuts introduced in Section 3.4 help to drastically reduce the size of the branch-and-price tree and lead to faster solving times.

Finally, we evaluated our approaches on a real-world instance. Preliminary experiments with large real-world instances showed that $\mathcal{A}_N$ performs best among all our presented approaches, which is consistent with the results presented earlier. In particular, the memory requirements of $\mathcal{A}_A$ are so large that we were not able to run it on large real-world instances. On such instances, even the restricted master problem of $\mathcal{A}_S$ dominates the runtime, so that the advantage of the fast pricing problems vanishes. Therefore, we only evaluated $\mathcal{A}_N$ on the real-world instance. The solving process terminated after 62.9 hours with an optimal and feasible solution. The instance and the found optimal solution are visualized in Figure 12. A total of 297 stations are selected in the solution. Figure 13 shows two box plots summarizing detour statistics over all trips. The first box plot summarizes the detour cost factors, i.e., the factors by which the cost of a trip (i.e., the travel time) in the solution increased relative to the cost without visiting a station. The second box plot refers to the total values by which the cost of a trip increased. On average, the cost of a trip increased by 1.0% (median 1.0%), for a total cost increase of 6.0 minutes (median 1.0 minutes). Furthermore, decision makers may want to investigate which trips deviate significantly from their cost without visiting stations. As the figures show, the majority of trips have only

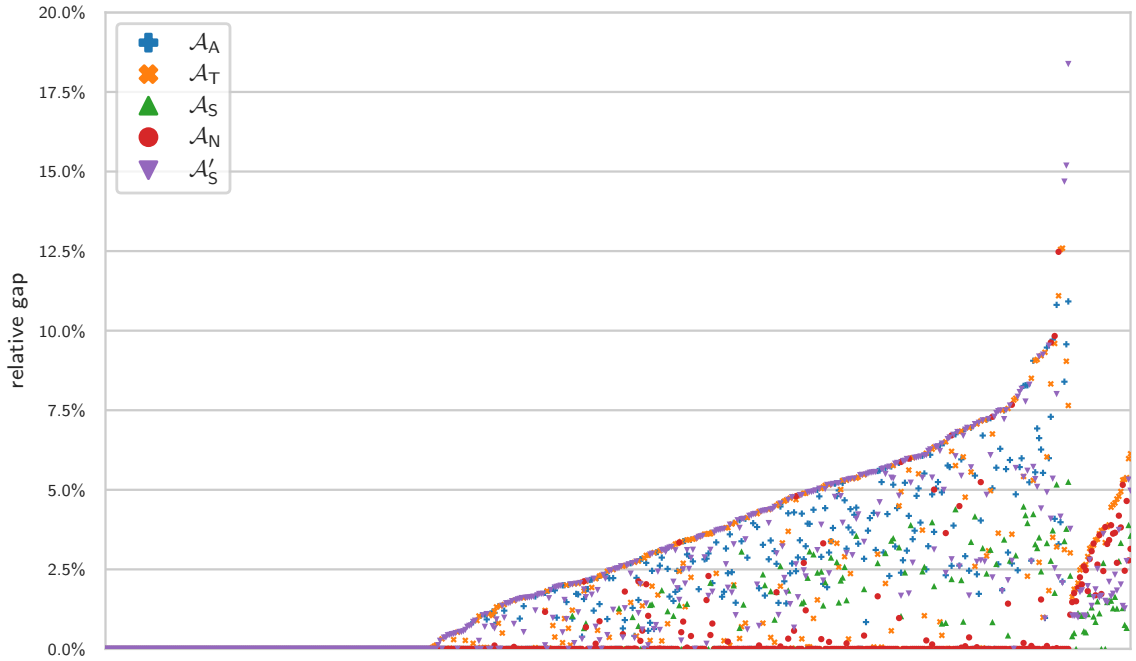


Figure 8: Achieved gaps after the approaches terminated. Each column corresponds to a single instance. The columns are sorted according to the gaps. For readability, we excluded entries with a gap larger than 20%.

a small increase in travel time. However, there are outliers with high detour cost factors. For example, the travel time of one trip increased to 200% (a total of 331 minutes) of the original travel time. This is caused by missing opportunities to install stations. As a result, the trip has to be rerouted between stations, which means a large detour. Therefore, the decision maker should consider adding candidate stations in this area so that the trip can be completed with less detour costs.

5 Conclusion

In this paper, we study exact approaches to the (directed) multi-stop location problem. First, we present an arc-based mixed-integer formulation that can be solved with any MIP solver. Then, we propose several pattern-based formulations solved by branch-price-and-cut. On the one hand, a pattern-based formulation that assigns a complete path to each trip, and a formulation that assigns paths to the segments of a trip. Our approaches use a problem-specific primal heuristic and branching strategies, as well as problem-specific cuts. We conduct experiments using randomly generated instances based on instances of the TSPLIB. Our experiments show that the segment-pattern-based formulation significantly outperforms the trip-pattern-based formulation and the arc-based formulation. In particular, the cuts help to speed up the solving process by keeping the branch-and-price tree small. Furthermore, combining the two pattern-based formulations by solving the pricing problems of the trip-pattern-based formulation with the segment-pattern-based formulation significantly improves the performance, so that this nested approach outperforms all other proposed ones. Moreover, we create a large instance using real-world data of an intercity bus service to demonstrate the practical usefulness of our approaches. In fact, the nested branch-price-and-cut approach is able to solve this instance to optimality in less than three days.

Future work could look at extensions and how they affect performance. For example, one might want to weight trips differently. In terms of implementation, such an extension should be easy to incorporate into our approach. Other extensions require more work, such as an upper bound on the extra costs caused by detours. Nevertheless, our approaches can serve as a promising foundation for further work on this problem.

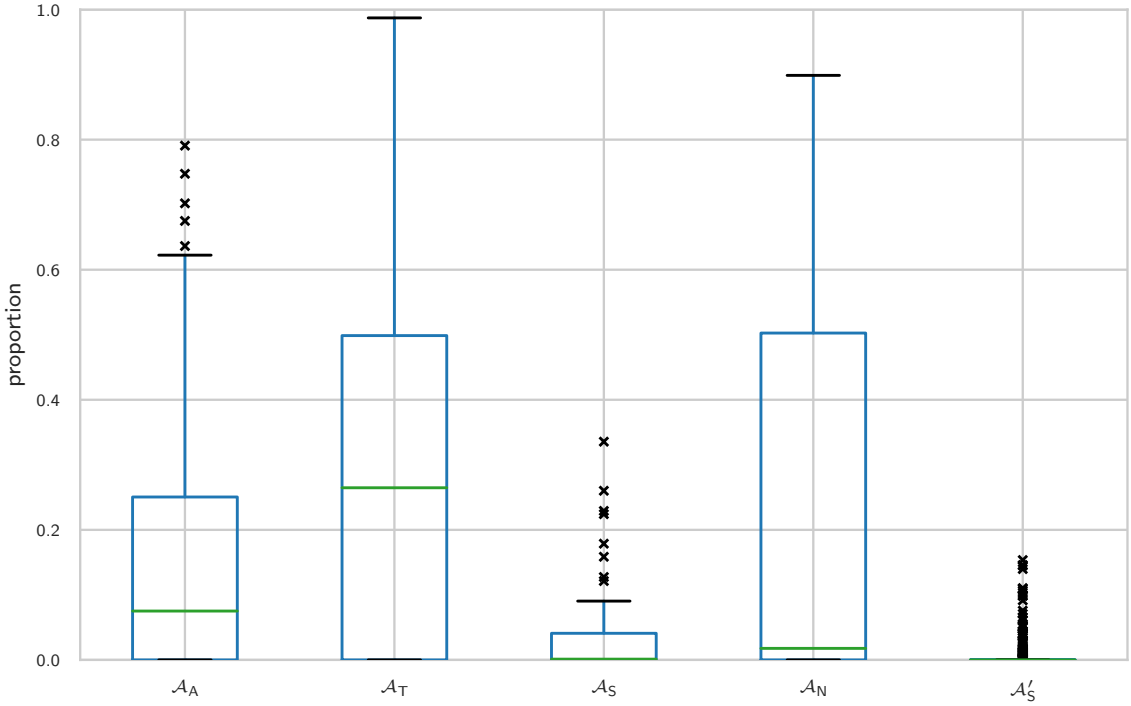


Figure 9: The proportions of the primal parts of the overall gaps are summarized by a box plot. Outliers are represented by X's.

References

[1] Tobias Achterberg. "Constraint Integer Programming". PhD thesis. Technical University of Berlin, 2007.

[2] Abdullah Almouhanna et al. "The location routing problem using electric vehicles with constrained distance". In: *Computers & Operations Research* 115 (Mar. 2020), p. 104864. ISSN: 03050548. DOI: 10.1016/j.cor.2019.104864.

[3] Automotive World Ltd. *MAN sets the standard for range: Fully electric bus breaks the 550 kilometre barrier*. 2021. URL: https://www.automotiveworld.com/news-releases/man-sets-the-standard-for-range-fully-electric-bus-breaks-the-550-kilometre-barrier/ (visited on 04/03/2023).

[4] Ksenia Bestuzheva et al. *The SCIP Optimization Suite 8.0*. 2021. DOI: 10.48550/ARXIV.2112.08872.

[5] Edgar Alberto Cabral et al. "The network design problem with relays". In: *European Journal of Operational Research* 180.2 (2007), pp. 834–844.

[6] Hatice Calik et al. "The electric location-routing problem: Formulation and Benders decomposition approach". In: (2018).

[7] Ismail Capar and Michael Kuby. "An efficient formulation of the flow refueling location model for alternative-fuel stations". In: *IIE Transactions* 44.8 (Aug. 1, 2012), pp. 622–636. ISSN: 0740-817X. DOI: 10.1080/0740817X.2011.635175.

[8] Si Chen, Ivana Ljubić, and S. Raghavan. "The Generalized Regenerator Location Problem". In: *INFORMS Journal on Computing* 27.2 (Apr. 2015), pp. 204–220. ISSN: 1091-9856, 1526-5528. DOI: 10.1287/ijoc.2014.0621.

[9] Si Chen, Ivana Ljubić, and S. Raghavan. "The regenerator location problem". In: *Networks* 55.3 (2010), pp. 205–220.

[10] Yanru Chen et al. "Solving the battery swap station location-routing problem with a mixed fleet of electric and conventional vehicles using a heuristic branch-and-price algorithm with an adaptive selection scheme". In: *Expert Systems with Applications* 186 (Dec. 30, 2021), p. 115683. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.115683.

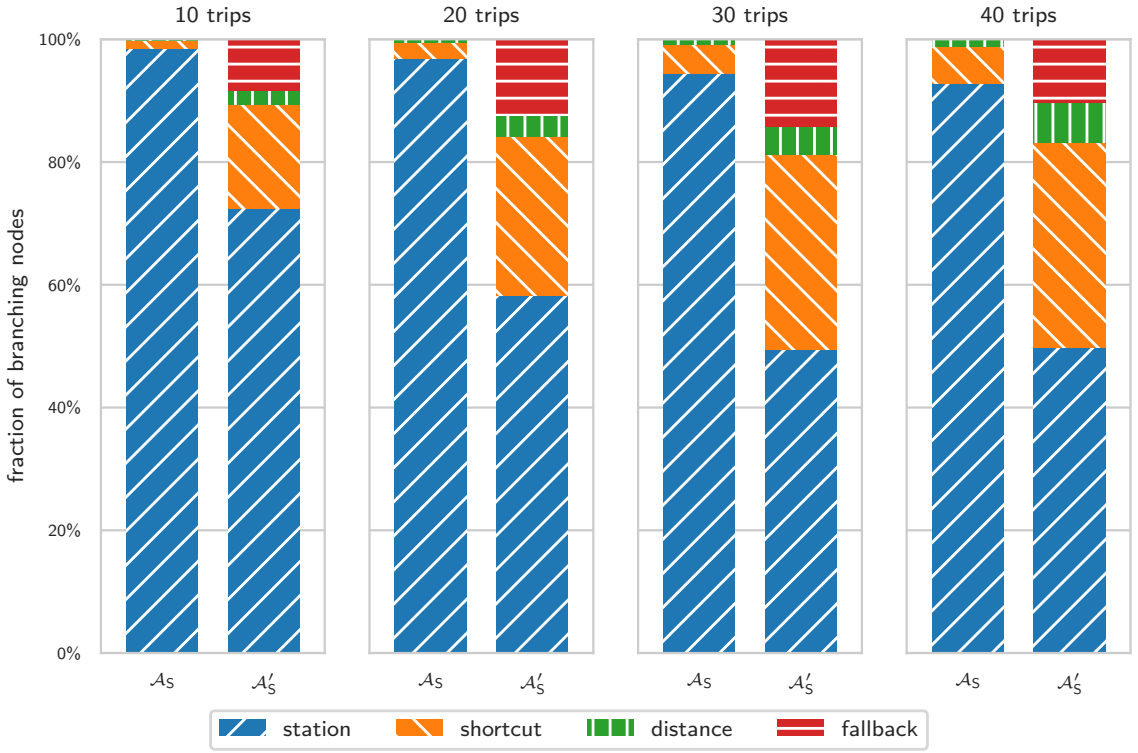


Figure 10: The fractions of tree node types created by $\mathcal{A}'_S$ and $\mathcal{A}_S$.

[11] Ryan G Conrad and Miguel Andres Figliozzi. "The Recharging Vehicle Routing Problem". In: *Proceedings of the 2011 Industrial Engineering Research Conference* (2011).

[12] Benoit Crevier, Jean-François Cordeau, and Gilbert Laporte. "The multi-depot vehicle routing problem with inter-depot routes". In: *European Journal of Operational Research* 176.2 (Jan. 2007), pp. 756–773. ISSN: 03772217. DOI: 10.1016/j.ejor.2005.08.015.

[13] George B. Dantzig and Philip Wolfe. "Decomposition Principle for Linear Programs". In: *Operations Research* 8.1 (Feb. 1960), pp. 101–111. ISSN: 0030-364X, 1526-5463. DOI: 10.1287/opre.8.1.101.

[14] Amarjit Datta, Brian K Ledbetter, and M. Ashiqur Rahman. "Optimal Deployment of Charging Stations for Electric Vehicles: A Formal Approach". In: *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). June 2017, pp. 83–90. DOI: 10.1109/ICDCSW.2017.26.

[15] Guy Desaulniers et al. "Exact Algorithms for Electric Vehicle-Routing Problems with Time Windows". In: *Operations Research* 64.6 (Dec. 2016), pp. 1388–1405. ISSN: 0030-364X, 1526-5463. DOI: 10.1287/opre.2016.1535.

[16] O. Edenhofer et al. *Climate Change 2014: Mitigation of Climate Change. Contribution of Working Group III to the Fifth Assessment Report of the Intergovern- mental Panel on Climate Change [Edenhofer, O., R. Pichs-Madruga, Y. Sokona, E. Farahani, S. Kadner, K. Seyboth, A. Adler, I. Baum, S. Brunner, P. Eickemeier, B. Kriemann, J. Savolainen, S. Schlömer, C. von Stechow, T. Zwickel and J.C. Minx (eds.)]*. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 2014. Chap. Transport.

[17] Sevgi Erdoğan and Elise Miller-Hooks. "A Green Vehicle Routing Problem". In: *Transportation Research Part E: Logistics and Transportation Review* 48.1 (Jan. 2012), pp. 100–114. ISSN: 13665545. DOI: 10.1016/j.tre.2011.08.001.

[18] Stefan Funke, Andre Nusser, and Sabine Storandt. "Placement of Loading Stations for Electric Vehicles: No Detours Necessary!" In: *Journal of Artificial Intelligence Research* 53 (Aug. 12, 2015), pp. 633–658. ISSN: 1076-9757. DOI: 10.1613/jair.4688.

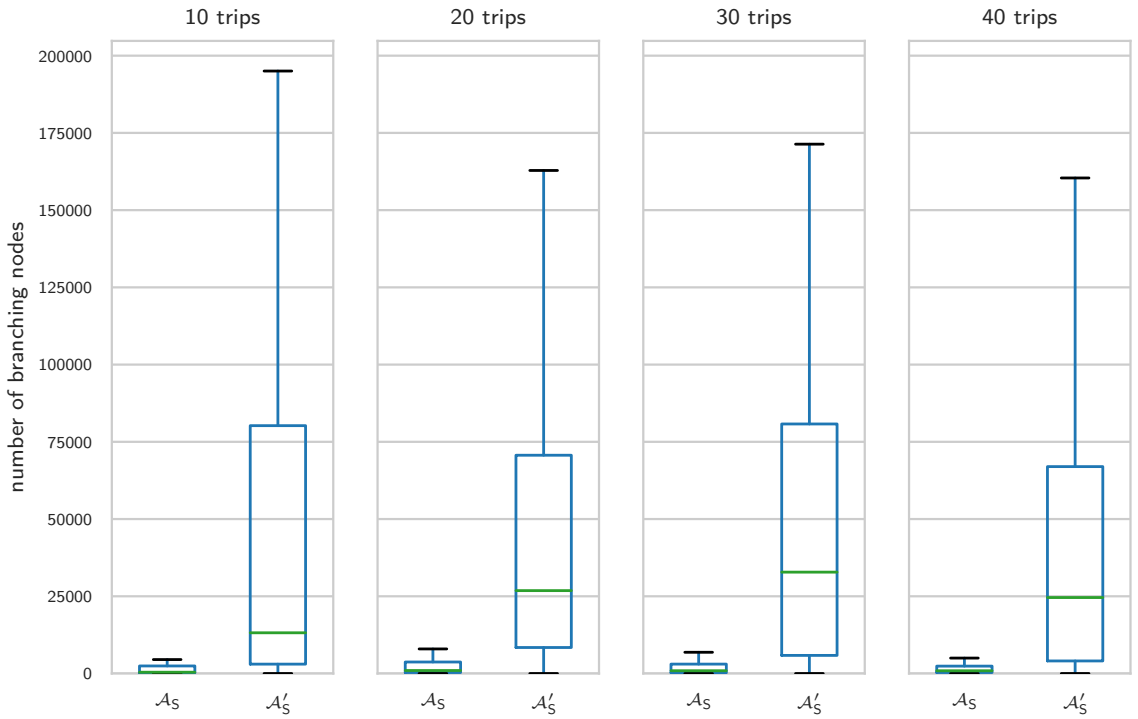


Figure 11: The total numbers of tree nodes created by $\mathcal{A}'_S$ and $\mathcal{A}_S$.

[19] Paul Göpfert and Stefan Bock. "A Branch&Cut approach to recharging and refueling infrastructure planning". In: *European Journal of Operational Research* 279.3 (Dec. 16, 2019), pp. 808–823. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2019.06.031.

[20] Gurobi Optimization, LLC. *Gurobi Optimizer*. 2023. URL: https://www.gurobi.com/solutions/gurobi-optimizer/ (visited on 04/03/2023).

[21] Itamar Hartstein, Mordechai Shalom, and Shmuel Zaks. "On the complexity of the regenerator location problem treewidth and other parameters". In: *Discrete Applied Mathematics* 199 (Jan. 2016), pp. 199–225. ISSN: 0166218X. DOI: 10.1016/j.dam.2015.01.036.

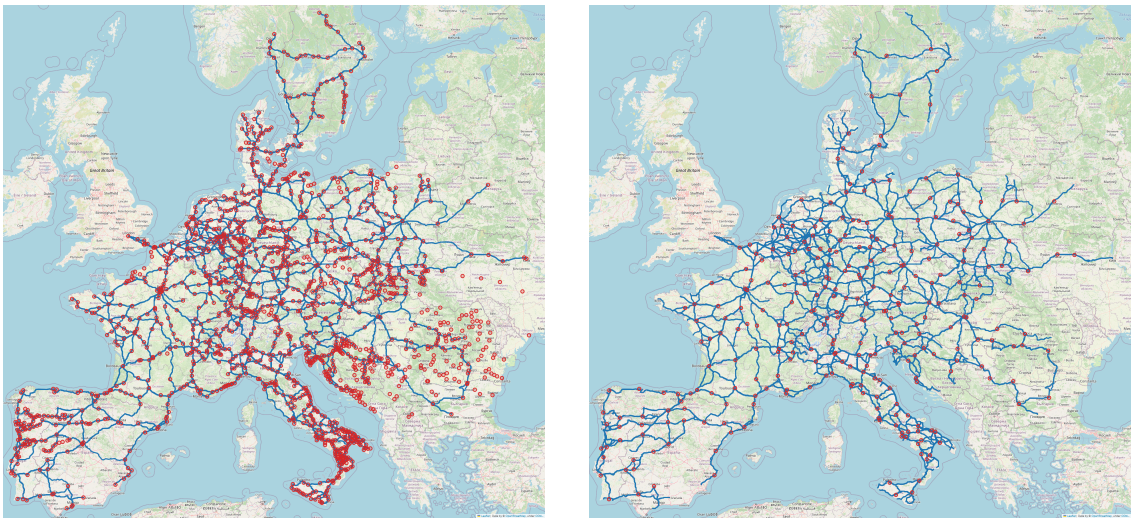
[22] Andrea Hess et al. "Optimal deployment of charging stations for electric vehicular networks". In: *Proceedings of the first workshop on Urban networking - UrbaNe '12*. the first workshop. Nice, France: ACM Press, 2012, p. 1. ISBN: 978-1-4503-1781-8. DOI: 10.1145/2413236.2413238.

[23] Julian Hof, Michael Schneider, and Dominik Goeke. "Solving the battery swap station location-routing problem with capacitated electric vehicles using an AVNS algorithm for vehicle-routing problems with intermediate stops". In: *Transportation Research Part B: Methodological* 97 (Mar. 1, 2017), pp. 102–112. ISSN: 0191-2615. DOI: 10.1016/j.trb.2016.11.009.

[24] M. Hosseini, S.A. MirHassani, and F. Hooshmand. "Deviation-flow refueling location problem with capacitated facilities: Model and algorithm". In: *Transportation Research Part D: Transport and Environment* 54 (July 2017), pp. 269–281. ISSN: 13619209. DOI: 10.1016/j.trd.2017.05.015.

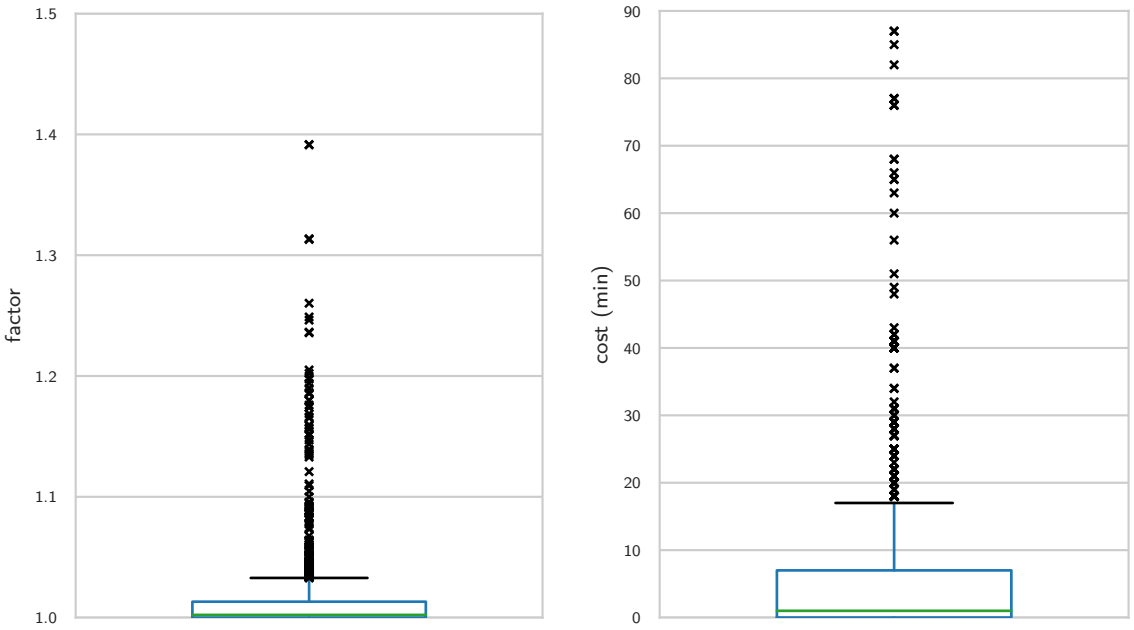
[25] Yongxi Huang, Shengyin Li, and Zhen Sean Qian. "Optimal Deployment of Alternative Fueling Stations on Transportation Networks Considering Deviation Paths". In: *Networks and Spatial Economics* 15.1 (Mar. 1, 2015), pp. 183–204. ISSN: 1572-9427. DOI: 10.1007/s11067-014-9275-1.

[26] Ozgur Kabadurmus and Alice E. Smith. "Multi-commodity k-splittable survivable network design problems with relays". In: *Telecommunication Systems* 62.1 (May 2016), pp. 123–133. ISSN: 1018-4864. DOI: 10.1007/s11235-015-0067-9.



(a) instance (b) solution

Figure 12: The real-world instance (left) and the best found solution (right) are visualized. Trips are represented by blue lines and stations by red dots. The visualization relies on data licensed under the Open Data Commons Open Database License (ODbL) by the OpenStreetMap Foundation.



(a) cost factors (b) total detour costs

Figure 13: The detour cost factors (left side) and the total detour costs (right side, additional travel time in minutes) are summarized over all trips by box plots. Outliers are represented by X's. For readability, we exclude outliers corresponding to cost factors larger than 1.5 and total detour costs larger than 90 minutes in the left and right plot, respectively.

[27] Jee Eun Kang and Will Recker. "Strategic Hydrogen Refueling Station Locations with Scheduling and Routing Considerations of Individual Vehicles". In: *Transportation Science* 49.4 (Nov. 2015), pp. 767–783. ISSN: 0041-1655, 1526-5447. DOI: 10.1287/trsc.2014.0519.

[28] Jong-Geun Kim and Michael Kuby. "A network transformation heuristic approach for the deviation flow refueling location model". In: *Computers & Operations Research* 40.4 (Apr. 2013), pp. 1122–1131. ISSN: 03050548. DOI: 10.1016/j.cor.2012.10.021.

[29] Jong-Geun Kim and Michael Kuby. "The deviation-flow refueling location model for optimizing a network of refueling stations". In: *International Journal of Hydrogen Energy* 37.6 (Mar. 2012), pp. 5406–5420. ISSN: 03603199. DOI: 10.1016/j.ijhydene.2011.08.108.

[30] Ömer Burak Kınay, Fatma Gzara, and Sibel A Alumur. "Full cover charging station location problem with routing". In: *Transportation Research Part B: Methodological* 144 (Feb. 1, 2021), pp. 1–22. ISSN: 0191-2615. DOI: 10.1016/j.trb.2020.12.001.

[31] Michael Kuby and Seow Lim. "Location of Alternative-Fuel Stations Using the Flow-Refueling Location Model and Dispersion of Candidate Sites on Arcs". In: *Networks and Spatial Economics* 7.2 (Apr. 3, 2007), pp. 129–152. ISSN: 1566-113X, 1572-9427. DOI: 10.1007/s11067-006-9003-6.

[32] Michael Kuby and Seow Lim. "The flow-refueling location problem for alternative-fuel vehicles". In: *Socio-Economic Planning Sciences* 39.2 (June 2005), pp. 125–145. ISSN: 00380121. DOI: 10.1016/j.seps.2004.03.001.

[33] Michael Kuby et al. "Optimization of hydrogen stations in Florida using the Flow-Refueling Location Model". In: *International Journal of Hydrogen Energy* 34.15 (Aug. 2009), pp. 6045–6064. ISSN: 03603199. DOI: 10.1016/j.ijhydene.2009.05.050.

[34] Alexander Kunith, Roman Mendelevitch, and Dietmar Goehlich. "Electrification of a city bus network—An optimization model for cost-effective placing of charging infrastructure and battery sizing of fast-charging electric bus systems". In: *International Journal of Sustainable Transportation* 11.10 (Nov. 26, 2017), pp. 707–720. ISSN: 1556-8318, 1556-8334. DOI: 10.1080/15568318.2017.1310962.

[35] Markus Leitner et al. "Exact approaches for network design problems with relays". In: *INFORMS Journal on Computing* 31.1 (2019), pp. 171–192. ISSN: 15265528. DOI: 10.1287/ijoc.2018.0820.

[36] Markus Leitner et al. "Exact approaches for the directed network design problem with relays". In: *Omega* 91 (2020), p. 102005.

[37] Xiangyong Li et al. "Models and column generation approach for the resource-constrained minimum cost path problem with relays". In: *Omega* 66 (Jan. 2017), pp. 79–90. ISSN: 03050483. DOI: 10.1016/j.omega.2016.01.012.

[38] Seow Lim and Michael Kuby. "Heuristic algorithms for siting alternative-fuel stations using the Flow-Refueling Location Model". In: *European Journal of Operational Research* 204.1 (July 2010), pp. 51–61. ISSN: 03772217. DOI: 10.1016/j.ejor.2009.09.032.

[39] Cheng-Chang Lin and Chuan-Chih Lin. "The p-center flow-refueling facility location problem". In: *Transportation Research Part B: Methodological* 118 (Dec. 2018), pp. 124–142. ISSN: 01912615. DOI: 10.1016/j.trb.2018.10.008.

[40] S. A. MirHassani and R. Ebrazi. "A Flexible Reformulation of the Refueling Station Location Problem". In: *Transportation Science* 47.4 (Nov. 2013), pp. 617–628. ISSN: 0041-1655, 1526-5447. DOI: 10.1287/trsc.1120.0430.

[41] MobilityData. *General Transit Feed Specification Reference*. 2019. URL: https://gtfs.org/reference/static (visited on 04/03/2023).

[42] Ibrahim Muter, Jean-François Cordeau, and Gilbert Laporte. "A Branch-and-Price Algorithm for the Multidepot Vehicle Routing Problem with Interdepot Routes". In: *Transportation Science* 48.3 (Aug. 2014), pp. 425–441. ISSN: 0041-1655, 1526-5447. DOI: 10.1287/trsc.2013.0489.

[43] Ashutosh Nigam and Yogesh K. Agarwal. "Optimal relay node placement in delay constrained wireless sensor network design". In: *European Journal of Operational Research* 233.1 (Feb. 2014), pp. 220–233. ISSN: 03772217. DOI: 10.1016/j.ejor.2013.08.031.

[44] Nicholas Nordlund, Leandros Tassiulas, and Jan-Hendrik Lange. *Optimization methods for the capacitated refueling station location problem with routing*. en. arXiv:2310.05569 [math]. Oct. 2023. URL: http://arxiv.org/abs/2310.05569 (visited on 10/12/2023).

[45] OpenMP Architecture Review Board. *OpenMP Application Programming Interface*. 2018. URL: https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5.0.pdf (visited on 04/03/2023).

[46] Shyam S.G. Perumal, Richard M. Lusby, and Jesper Larsen. "Electric bus planning & scheduling: A review of related problems and methodologies". In: *European Journal of Operational Research* 301.2 (Sept. 2022), pp. 395–413. ISSN: 03772217. DOI: 10.1016/j.ejor.2021.10.058.

[47] Gerhard Reinelt. "TSPLIB—A Traveling Salesman Problem Library". In: *ORSA Journal on Computing* 3.4 (Nov. 1991), pp. 376–384. ISSN: 0899-1499. DOI: 10.1287/ijoc.3.4.376.

[48] Philipp K. Rose et al. "Optimal development of alternative fuel station networks considering node capacity restrictions". In: *Transportation Research Part D: Transport and Environment* 78 (Jan. 1, 2020), p. 102189. ISSN: 1361-9209. DOI: 10.1016/j.trd.2019.11.018.

[49] Barbara Scheiper, Maximilian Schiffer, and Grit Walther. "The flow refueling location problem with load flow control". In: *Omega* 83 (Mar. 2019), pp. 50–69. ISSN: 03050483. DOI: 10.1016/j.omega.2018.02.003.

[50] Maximilian Schiffer and Grit Walther. "The electric location routing problem with time windows and partial recharging". In: *European Journal of Operational Research* 260.3 (Aug. 2017), pp. 995–1013. ISSN: 03772217. DOI: 10.1016/j.ejor.2017.01.011.

[51] Maximilian Schiffer et al. "Vehicle Routing and Location Routing with Intermediate Stops: A Review". In: *Transportation Science* 53.2 (Mar. 2019), pp. 319–343. ISSN: 0041-1655. DOI: 10.1287/trsc.2018.0836.

[52] Michael Schneider, Andreas Stenger, and Dominik Goeke. "The Electric Vehicle-Routing Problem with Time Windows and Recharging Stations". In: *Transportation Science* 48.4 (Nov. 2014), pp. 500–520. ISSN: 0041-1655, 1526-5447. DOI: 10.1287/trsc.2013.0490.

[53] Jeremy Siek, Lie-Quan Lee, and Andrew Lumsdaine. *The Boost Graph Library (BGL)*. 2001. URL: https://www.boost.org/doc/libs/1_75_0/libs/graph/doc/index.html (visited on 04/03/2023).

[54] Sabine Storandt and Stefab Funke. "Enabling E-Mobility: Facility Location for Battery Loading Stations". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 27.1 (June 29, 2013), pp. 1341–1347. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v27i1.8478.

[55] Wei Tu et al. "Optimizing the locations of electric taxi charging stations: A spatial–temporal demand coverage approach". In: *Transportation Research Part C: Emerging Technologies* 65 (Apr. 2016), pp. 172–189. ISSN: 0968090X. DOI: 10.1016/j.trc.2015.10.004.

[56] Dionysios Tzamakos, Christina Iliopoulou, and Konstantinos Kepaptsoglou. "Electric bus charging station location optimization considering queues". en. In: *International Journal of Transportation Science and Technology* 12.1 (Mar. 2023), pp. 291–300. ISSN: 20460430. DOI: 10.1016/j.ijtst.2022.02.007. URL: https://linkinghub.elsevier.com/retrieve/pii/S2046043022000259 (visited on 12/04/2023).

[57] Christopher Upchurch and Michael Kuby. "Comparing the p-median and flow-refueling models for locating alternative-fuel stations". In: *Journal of Transport Geography* 18.6 (Nov. 2010), pp. 750–758. ISSN: 09666923. DOI: 10.1016/j.jtrangeo.2010.06.015.

[58] Christopher Upchurch, Michael Kuby, and Seow Lim. "A Model for Location of Capacitated Alternative-Fuel Stations". In: *Geographical Analysis* 41.1 (Jan. 2009), pp. 85–106. ISSN: 00167363. DOI: 10.1111/j.1538-4632.2009.00744.x.

[59] Harwin de Vries and Evelot Duijzer. "Incorporating driving range variability in network design for refueling facilities". In: *Omega* 69 (June 2017), pp. 102–114. ISSN: 03050483. DOI: 10.1016/j.omega.2016.08.005.

[60] Ying-Wei Wang and Chuah-Chih Lin. "Locating road-vehicle refueling stations". In: *Transportation Research Part E: Logistics and Transportation Review* 45.5 (Sept. 1, 2009), pp. 821–829. ISSN: 1366-5545. DOI: 10.1016/j.tre.2009.03.002.

[61] Ying-Wei Wang and Chuan-Ren Wang. "Locating passenger vehicle refueling stations". In: *Transportation Research Part E: Logistics and Transportation Review* 46.5 (Sept. 1, 2010), pp. 791–801. ISSN: 1366-5545. DOI: 10.1016/j.tre.2009.12.001.

[62] Felix J L Willamowski, Miriam Ganz, and Erik Mühmer. *The Multi-Stop Station Location Problem*. repORt 2020–60. Lehrstuhl für Operations Research, RWTH Aachen University,

Apr. 25, 2020. URL: https://or.rwth-aachen.de/files/research/repORt/mslp_theory_v2.pdf.

[63] Laurence A. Wolsey. "Valid inequalities for 0–1 knapsacks and mips with generalised upper bound constraints". In: *Discrete Applied Mathematics* 29.2 (Dec. 1990), pp. 251–261. ISSN: 0166218X. DOI: 10.1016/0166-218X(90)90148-6.

[64] Owen Worley, Diego Klabjan, and Timothy M. Sweda. "Simultaneous vehicle routing and charging station siting for commercial electric vehicles". In: *2012 IEEE International Electric Vehicle Conference*. IEEE. 2012, pp. 1–3.

[65] Jun Yang and Hao Sun. "Battery swap station location-routing problem with capacitated electric vehicles". In: *Computers & Operations Research* 55 (Mar. 2015), pp. 217–232. ISSN: 03050548. DOI: 10.1016/j.cor.2014.07.003.

[66] Barış Yıldız, Okan Arslan, and Oya Ekin Karaşan. "A branch and price approach for routing and refueling station location model". In: *European Journal of Operational Research* 248.3 (Feb. 2016), pp. 815–826. ISSN: 03772217. DOI: 10.1016/j.ejor.2015.05.021.

[67] Barış Yıldız, Oya Ekin Karaşan, and Hande Yaman. "Branch-and-price approaches for the network design problem with relays". In: *Computers and Operations Research* 92 (2018), pp. 155–169. ISSN: 03050548. DOI: 10.1016/j.cor.2018.01.004.

[68] Peng-Sheng You and Yi-Chih Hsieh. "A hybrid heuristic approach to the problem of the location of vehicle charging stations". In: *Computers & Industrial Engineering* 70 (Apr. 1, 2014), pp. 195–204. ISSN: 0360-8352. DOI: 10.1016/j.cie.2014.02.001.

[69] Hong Zheng et al. "Traffic Equilibrium and Charging Facility Locations for Electric Vehicles". In: *Networks and Spatial Economics* 17.2 (June 2017), pp. 435–457. ISSN: 1566-113X. DOI: 10.1007/s11067-016-9332-z.

A Appendix

Table A1: Achieved times (in seconds) for each instance of I_1 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$
a280	2446	3601	**301**	1872	3600	3602	3601	3602	**1229**	3616	3600	3607	3602	3603	**1953**	3613	3600	3603	3604	3600	**2956**	3602	3600	3629
att48	7	66	**1**	9	2	11	80	173	**36**	39	690	101	211	259	41	**33**	3600	140	492	653	**157**	183	3600	683
bayg29	**1**	23	**1**	5	**1**	2	17	61	10	**4**	81	21	19	24	15	**5**	106	17	35	45	29	**22**	1525	100
bays29	6	9	**1**	2	2	6	5	8	**2**	**2**	3	6	32	15	19	**3**	730	22	85	14	40	**2**	1605	28
berlin52	62	88	10	**7**	57	73	361	186	57	**25**	2086	251	278	1336	**74**	98	3600	776	959	525	154	**134**	3600	736
bier127	2439	3601	**126**	1750	431	2886	3604	3601	**783**	1739	3600	3609	3603	3601	**685**	1591	3600	3600	3601	3601	**1534**	2091	3600	3604
brazil58	60	127	**6**	10	11	41	285	1210	**42**	79	1075	738	605	694	184	**108**	3600	1312	465	1822	**105**	173	3600	1087
brg180	152	798	**10**	498	11	142	347	1100	**42**	344	52	335	531	3301	93	559	**73**	495	534	3601	66	782	**57**	493
burma14	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**	1	**0**
ch130	628	846	**61**	446	619	813	1277	2235	**127**	451	3600	2970	1941	3601	**183**	957	3600	3606	3604	3601	**487**	1130	3600	3601
ch150	966	3306	**69**	252	370	2623	3600	3601	**888**	2548	3600	3605	3601	3601	**426**	1476	3600	3602	3602	3601	**1968**	2319	3600	3601
d198	3604	3601	**311**	1766	3600	3610	3656	3601	3361	**3259**	3600	3623	3602	3602	**3219**	3605	3600	3601	3603	3602	**1905**	3606	3600	3603
d493	3602	3609	**1467**	3619	3600	3606	3626	3605	**3600**	3625	**3600**	3619	**3600**	3604	**3600**	3605	**3600**	**3600**	**3600**	3609	**3600**	3605	3601	**3600**
dantzig42	15	41	**3**	6	6	14	35	119	13	**11**	39	28	135	191	28	**18**	3257	85	80	197	77	**48**	3600	153
eil101	295	1302	**13**	107	70	297	1286	2468	**84**	152	595	3002	3601	3600	**142**	617	3600	3601	3601	3601	**203**	409	3600	3601
eil51	36	418	**15**	20	42	35	123	380	**23**	49	257	153	140	330	**29**	52	200	583	282	684	**65**	76	3600	912
eil76	116	752	**14**	45	72	200	349	641	**41**	73	287	400	996	2563	**119**	148	3600	3600	2087	3473	207	**105**	3600	3601
fl417	3605	3602	**3600**	3612	**3600**	3606	3620	3603	**3600**	3603	**3600**	3621	3674	3609	**3600**	3612	**3600**	3658	**3600**	3606	**3600**	3609	3601	**3600**
fri26	8	2	**1**	**1**	2	7	11	31	**3**	4	17	10	13	25	12	**4**	482	21	32	36	**6**	7	1379	27
gil262	3604	3602	**1110**	2680	3600	3601	3602	3602	**1393**	3602	3600	3604	3605	3601	**3600**	3614	**3600**	3602	3611	**3600**	**3600**	**3600**	**3600**	3602
gr120	393	1597	**24**	91	206	1088	1844	3600	**164**	283	3600	2819	3601	3601	**275**	888	3600	3603	2646	3600	**1433**	2750	3600	3601
gr137	104	1142	**43**	223	55	306	3601	3601	**279**	881	3600	3601	2351	3601	**347**	2348	3600	3602	3604	3601	**1841**	3600	3600	3601
gr17	**0**	**0**	**0**	**0**	**0**	**0**	1	1	**0**	**0**	1	2	1	2	**0**	1	1	1	4	5	**1**	**1**	32	5
gr202	2626	3602	**240**	1795	1035	2788	3602	3601	**744**	3601	3600	3604	3601	3602	**3600**	3603	**3600**	3602	3603	3603	**3600**	3612	**3600**	3604
gr21	**0**	1	**0**	**0**	**0**	**0**	2	6	**1**	**1**	3	2	8	5	7	**1**	12	8	8	9	8	**3**	10	8
gr229	2387	3601	**209**	855	1002	2640	3603	3601	**2292**	3604	3600	3631	3602	3602	**3600**	3605	**3600**	3601	3602	3601	**3600**	3604	**3600**	3602
gr24	2	2	**1**	**1**	2	2	11	8	5	**2**	16	8	12	5	6	**1**	14	8	14	19	9	**5**	26	14
gr431	3604	3602	**2403**	3619	3600	3605	3615	3609	**3600**	3602	**3600**	3628	3611	3609	**3600**	3603	3601	3613	**3600**	3608	**3600**	3605	**3600**	**3600**
gr48	18	99	**2**	8	4	21	62	430	**8**	51	43	73	95	79	**12**	15	1259	106	641	290	41	**32**	3600	555
gr96	87	1620	**14**	116	28	136	596	577	**74**	86	313	3407	3600	1579	**147**	298	2592	3600	3602	3600	3342	**595**	3600	3601
hk48	29	70	**3**	6	12	27	53	156	33	**19**	102	58	118	322	**57**	69	3600	118	165	193	130	**89**	3600	505
kroA100	143	1069	**14**	71	72	259	985	649	**45**	94	1982	3548	3600	3600	**182**	761	3600	3600	3604	3601	**276**	578	3600	3600
kroA150	319	3601	**45**	2366	217	228	3604	3601	**729**	1203	3600	3600	3601	3601	**343**	930	3600	3602	3601	3601	**691**	1595	3600	3601
kroA200	1188	2847	**133**	733	1392	2268	2075	3601	**203**	1777	2490	3601	3601	3601	**698**	3603	3600	3602	3601	3602	**1900**	3601	3600	3775
kroB100	218	1053	**14**	71	62	227	615	1161	**71**	192	304	3472	3601	3600	**242**	546	3600	3600	3600	3600	**325**	548	3600	3624
kroB150	3601	3592	**79**	418	1397	3604	1483	3601	**180**	658	1676	3602	3603	3601	**436**	3132	3600	3604	3606	3601	**959**	1744	3600	3604
kroB200	1801	3601	**134**	2240	3600	2587	3602	3602	**422**	3218	3600	3643	3601	3602	**882**	3602	3600	3602	3603	3600	**3398**	3608	3600	3601
kroC100	341	1018	**36**	104	193	597	1958	3600	**79**	464	540	3609	2405	3381	**144**	312	3600	3611	3606	3601	**514**	1556	3600	3600
kroD100	173	330	**11**	45	55	395	686	1283	**57**	205	1592	1124	978	1878	**159**	254	3600	3602	3601	3601	**449**	1103	3600	3602
kroE100	225	586	**27**	97	105	353	2055	2273	**198**	450	1586	3601	759	3600	**84**	189	3600	3600	3604	3540	**316**	623	3600	3608
lin105	65	2050	**12**	1233	81	159	1794	1580	**165**	308	3600	3606	3600	3600	**159**	463	3600	3604	3601	3600	**271**	1375	3600	3601
lin318	3604	3602	**949**	3601	3600	3601	3605	3605	**3347**	3603	3600	3602	3618	3603	**3600**	**3600**	**3600**	3605	3635	3603	**3600**	3603	**3600**	3621
linhp318	3609	3602	**696**	2426	3600	3605	3613	3604	**3600**	3601	**3600**	3604	3611	3602	**2610**	3605	3600	3629	3626	**3600**	**3600**	3602	**3600**	3612
pcb442	3604	3606	**1136**	3601	3600	3604	3675	3604	**3600**	3614	**3600**	3616	**3600**	3610	**3600**	3614	**3600**	**3600**	**3600**	3610	**3600**	3606	**3600**	**3600**

Continued on next page

Table A1: Achieved times (in seconds) for each instance of I_1 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$
pr107	801	3600	**101**	794	965	1954	3601	3601	**360**	571	3600	3600	3605	3601	**489**	3601	3600	3604	3605	3601	**1224**	1324	3600	3604
pr124	347	1262	**23**	282	262	591	3602	3601	**472**	2469	3600	3600	3604	3601	1357	**658**	3600	3601	3603	3601	**559**	907	3600	3605
pr136	522	2879	**47**	214	285	516	3602	3601	**410**	3601	3600	3602	3608	3601	**696**	879	3601	3603	3602	3601	**484**	1388	3600	3602
pr144	2767	3033	**216**	410	3060	3601	3604	3601	**748**	3609	3600	3602	3601	3601	**360**	792	3600	3604	3603	3601	**1866**	3614	3600	3603
pr152	259	1013	**45**	255	260	343	1204	3601	**126**	637	2057	3606	3602	3601	**791**	1761	3600	3604	3601	3601	**617**	1282	3600	3601
pr226	3604	3602	**2637**	3603	3600	3603	3630	3601	**2363**	3611	3600	3631	3605	3602	**1965**	3604	3600	3601	3602	3602	**3600**	3605	**3600**	3602
pr264	3608	3602	**821**	1935	3600	3601	3602	3603	**3600**	3605	**3600**	3602	3602	3604	**3600**	3604	**3600**	3602	3603	3603	**3600**	3603	**3600**	3604
pr299	1758	3601	**239**	3604	3600	3602	3606	3602	**807**	3618	3600	3601	3602	3601	**2246**	3604	3600	3603	3605	3604	**2913**	3603	3600	3602
pr439	3056	3602	**836**	3603	3600	3606	3604	3602	**3600**	3601	**3600**	3626	3637	3604	**3600**	3603	**3600**	3615	3634	**3600**	3601	3611	**3600**	3619
pr76	60	1275	**13**	100	206	47	610	827	70	**53**	3600	1785	3601	2256	182	**131**	3600	3600	3473	3117	298	**191**	3600	3601
rat195	358	3601	**81**	606	118	543	3614	3601	**1242**	2208	3600	3638	3603	3601	**460**	3616	3600	3618	3601	3602	**1474**	3605	3600	3602
rat99	154	624	**12**	59	74	182	1144	3242	**116**	163	852	3618	1044	3600	**112**	215	3600	3600	3601	3600	**327**	751	3600	3627
rd100	876	3600	**59**	273	345	963	3600	3600	**133**	480	1267	3600	3600	3601	**138**	592	2076	3601	3602	3601	**179**	454	3600	3603
rd400	3601	3601	**555**	3605	3600	3607	3617	3603	**2545**	3617	3600	3654	3652	3606	**3600**	3622	**3600**	3670	**3600**	3611	**3600**	3610	**3600**	**3600**
si175	2570	3601	**231**	1514	3600	3601	3607	3602	**635**	3190	3600	3601	3615	3602	**965**	3600	3600	3606	3607	3602	**2106**	3601	3600	3604
st70	81	309	**9**	30	57	88	280	1853	**55**	172	286	1160	1443	3600	**75**	174	538	804	3600	2509	**209**	226	3600	3600
swiss42	20	88	**7**	13	25	18	68	106	43	**16**	134	128	101	234	63	**27**	2234	112	129	276	128	**48**	3600	431
ts225	3643	3601	**338**	880	3600	3605	3602	3602	**569**	1586	3600	3603	3603	3600	**1119**	3612	3600	3603	3602	3602	**1468**	3601	3600	3602
tsp225	975	3601	**113**	464	1986	1650	3603	3602	**504**	3127	3600	3608	3639	3602	**3200**	3603	3600	3633	3602	3601	**3162**	3602	3600	3604
u159	641	966	**39**	171	398	857	3601	3601	**198**	419	2602	3601	3601	3601	**346**	1629	3600	3601	3603	3601	**479**	2666	3600	3601
ulysses16	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ulysses22	3	10	**1**	2	**1**	3	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table A2: Achieved gaps (as percentages) for each instance of I_1 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_A$	$\mathcal{A}_T$	$\mathcal{A}_S$	$\mathcal{A}_N$	$\mathcal{A}'_S$	$\mathcal{A}'_A$	$\mathcal{A}_A$	$\mathcal{A}_T$	$\mathcal{A}_S$	$\mathcal{A}_N$	$\mathcal{A}'_S$	$\mathcal{A}'_A$	$\mathcal{A}_A$	$\mathcal{A}_T$	$\mathcal{A}_S$	$\mathcal{A}_N$	$\mathcal{A}'_S$	$\mathcal{A}'_A$	$\mathcal{A}_A$	$\mathcal{A}_T$	$\mathcal{A}_S$	$\mathcal{A}_N$	$\mathcal{A}'_S$	$\mathcal{A}'_A$
a280	**0.0**	2.0	**0.0**	**0.0**	0.2	1.2	2.1	3.7	**0.0**	0.8	1.8	2.6	3.0	3.6	**0.0**	2.3	1.7	3.0	-	3.7	**0.0**	1.7	1.7	-
att48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.6	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.4	**0.0**
bayg29	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
bays29	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
berlin52	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.7	**0.0**
bier127	**0.0**	1.8	**0.0**	**0.0**	**0.0**	**0.0**	4.6	3.3	**0.0**	**0.0**	3.2	5.9	2.1	12.6	**0.0**	**0.0**	2.1	5.5	3.4	6.0	**0.0**	**0.0**	2.7	7.2
brazil58	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**
brg180	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.4	**0.0**	**0.0**	**0.0**	**0.0**
burma14	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
ch130	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	**0.0**	**0.0**	0.5	**0.0**	**0.0**	1.4	4.1	2.2	0.6	**0.0**	**0.0**	1.8	3.7
ch150	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.8	3.6	**0.0**	**0.0**	2.9	3.8	2.4	4.7	**0.0**	**0.0**	1.4	3.8	3.0	1.7	**0.0**	**0.0**	2.4	4.0
d198	0.9	2.4	**0.0**	**0.0**	0.4	2.4	2.1	4.6	**0.0**	**0.0**	2.2	2.7	2.5	6.9	**0.0**	0.4	1.7	3.0	3.2	4.9	**0.0**	1.2	2.3	3.7
d493	1.8	1.2	**0.0**	1.2	0.1	1.8	-	2.1	1.1	1.9	**1.0**	-	-	4.5	**1.5**	3.9	1.6	-	-	3.6	**1.1**	3.6	1.6	-
dantzig42	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.5	**0.0**
eil101	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.5	4.0	**0.0**	**0.0**	1.1	2.2	1.8	0.4	**0.0**	**0.0**	1.9	4.4
eil51	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.7	**0.0**
eil76	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.4	4.8	**0.0**	**0.0**	**0.0**	**0.0**	1.7	4.8
fl417	1.5	2.8	**0.7**	2.1	1.1	2.0	-	5.3	**0.8**	2.5	1.3	-	-	2.7	**0.5**	2.5	1.0	-	-	2.6	**0.6**	2.6	1.1	-
fri26	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gil262	2.2	2.5	**0.0**	**0.0**	1.2	2.7	1.9	4.3	**0.0**	0.1	0.7	2.2	2.0	5.1	**1.4**	3.3	2.1	2.8	3.0	7.6	**0.4**	1.4	2.2	3.1
gr120	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.4	**0.0**	2.0	1.0	**0.0**	**0.0**	1.5	5.0	**0.0**	7.2	**0.0**	**0.0**	2.7	2.3
gr137	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.1	**0.0**	**0.0**	**0.0**	0.7	1.6	**0.0**	0.3	**0.0**	**0.0**	0.5	0.6	3.3	5.4	**0.0**	**0.0**	4.2	6.7
gr17	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr202	**0.0**	2.3	**0.0**	**0.0**	**0.0**	**0.0**	3.3	11.1	**0.0**	12.5	2.3	4.1	2.3	6.5	**0.8**	3.6	2.4	3.8	3.7	6.3	**2.0**	5.0	3.1	4.5
gr21	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr229	**0.0**	0.9	**0.0**	**0.0**	**0.0**	**0.0**	2.4	4.7	**0.0**	4.8	2.7	2.9	2.8	4.9	**1.5**	2.7	2.5	5.6	7.3	8.3	**2.8**	9.6	4.1	6.4
gr24	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr431	0.6	2.9	**0.0**	1.0	0.1	1.0	-	1.9	**0.5**	1.8	1.0	-	-	4.9	3.3	3.8	**2.0**	-	-	3.3	**1.0**	3.3	1.6	-
gr48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.1	**0.0**
gr96	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.2	**0.0**	**0.0**	**0.0**	**0.0**	4.3	4.4	2.4	**0.0**	**0.0**	5.3	8.2
hk48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.0	**0.0**
kroA100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.0	**0.0**	**0.0**	**0.0**	0.7	3.7	2.2	0.4	**0.0**	**0.0**	2.5	5.0
kroA150	**0.0**	0.1	**0.0**	**0.0**	**0.0**	**0.0**	3.2	3.8	**0.0**	**0.0**	2.7	4.0	1.0	9.3	**0.0**	**0.0**	1.0	4.3	1.3	1.6	**0.0**	**0.0**	0.5	2.8
kroA200	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.1	**0.0**	**0.0**	**0.0**	0.7	2.9	4.1	**0.0**	**0.0**	0.6	3.5	2.7	3.6	**0.0**	7.7	2.8	3.5
kroB100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.2	0.4	**0.0**	**0.0**	2.7	4.6	0.9	**0.0**	**0.0**	**0.0**	1.1	1.6
kroB150	1.1	**0.0**	**0.0**	**0.0**	**0.0**	2.9	**0.0**	0.5	**0.0**	**0.0**	**0.0**	0.6	2.2	1.9	**0.0**	**0.0**	1.7	3.9	1.4	4.6	**0.0**	**0.0**	1.1	3.6
kroB200	**0.0**	1.0	**0.0**	**0.0**	0.3	**0.0**	0.4	2.1	**0.0**	**0.0**	0.3	1.5	2.5	3.4	**0.0**	0.5	0.3	3.7	2.8	8.5	**0.0**	0.1	2.7	3.8
kroC100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	**0.0**	**0.0**	**0.0**	1.8	**0.0**	**0.0**	**0.0**	**0.0**	0.5	1.1	4.1	0.8	**0.0**	**0.0**	3.4	5.4
kroD100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.9	1.7	2.5	3.3	**0.0**	**0.0**	1.9	5.9
kroE100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.9	**0.0**	**0.0**	**0.0**	**0.0**	0.4	0.8	1.9	**0.0**	**0.0**	**0.0**	2.1	5.0
lin105	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.1	2.5	1.0	**0.0**	**0.0**	**0.0**	1.6	1.6	0.8	1.3	**0.0**	**0.0**	0.9	4.5
lin318	1.1	1.2	**0.0**	0.1	0.6	1.8	2.7	2.9	**0.0**	2.0	1.4	3.0	-	3.8	**1.5**	3.3	**1.5**	-	-	4.7	**1.3**	2.7	1.8	-
linhp318	1.5	2.0	**0.0**	**0.0**	0.9	2.0	1.9	2.4	**1.4**	3.3	1.5	2.5	-	3.3	**0.0**	3.8	1.5	248.8	-	3.4	**0.9**	1.7	1.9	-
pcb442	1.4	2.3	**0.0**	0.9	0.3	1.7	-	1.6	**0.4**	1.5	1.0	-	-	3.2	**1.3**	3.1	1.4	-	-	5.3	**1.3**	5.2	**1.3**	-

Continued on next page

Table A2: Achieved gaps (as percentages) for each instance of I_1 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{A}}$	$\mathcal{A}_{\mathrm{T}}$	$\mathcal{A}_{\mathrm{S}}$	$\mathcal{A}_{\mathrm{N}}$	$\mathcal{A}'_{\mathrm{S}}$	$\mathcal{A}'_{\mathrm{A}}$
pr107	**0.0**	3.5	**0.0**	**0.0**	**0.0**	**0.0**	2.9	3.4	**0.0**	**0.0**	2.5	5.3	3.3	2.4	**0.0**	**0.0**	3.7	5.1	4.0	12.6	**0.0**	**0.0**	4.2	5.4
pr124	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.7	5.3	**0.0**	**0.0**	2.9	5.1	3.4	1.3	**0.0**	**0.0**	2.1	5.1	2.4	3.9	**0.0**	**0.0**	1.8	5.1
pr136	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.8	1.4	**0.0**	0.6	0.6	3.5	2.7	1.5	**0.0**	**0.0**	2.4	4.9	1.8	7.5	**0.0**	**0.0**	1.8	3.2
pr144	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.1	3.4	2.2	**0.0**	0.4	2.9	4.0	1.6	**0.0**	**0.0**	**0.0**	1.2	3.3	3.4	2.7	**0.0**	0.9	2.5	4.1
pr152	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.5	1.8	3.3	**0.0**	**0.0**	3.1	3.5	1.7	2.1	**0.0**	**0.0**	1.1	3.6
pr226	2.8	2.1	**0.0**	0.3	2.0	2.8	2.4	3.8	**0.0**	0.2	1.9	2.7	4.0	7.2	**0.0**	5.2	2.8	3.1	2.6	6.2	**2.2**	6.7	2.8	3.4
pr264	1.9	0.7	**0.0**	**0.0**	1.0	1.8	2.4	4.9	**1.0**	5.9	1.7	4.0	4.1	9.6	**1.8**	9.8	2.5	4.7	5.0	6.8	**0.8**	7.3	2.2	5.0
pr299	**0.0**	1.5	**0.0**	0.1	0.8	0.8	0.9	2.6	**0.0**	1.8	1.3	1.4	2.8	3.6	**0.0**	1.5	0.2	2.8	4.2	5.6	**0.0**	6.0	1.7	4.2
pr439	**0.0**	2.2	**0.0**	0.2	0.7	1.1	-	2.5	**1.0**	2.0	1.8	-	-	3.9	**1.5**	3.4	2.1	-	-	4.6	**1.6**	3.7	2.3	-
pr76	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.7	**0.0**	1.0	**0.0**	**0.0**	**0.0**	2.0	2.2	**0.0**	**0.0**	**0.0**	**0.0**	2.6	7.4
rat195	**0.0**	2.9	**0.0**	**0.0**	**0.0**	**0.0**	2.9	4.8	**0.0**	**0.0**	1.6	3.5	0.4	2.7	**0.0**	**0.0**	0.3	0.6	3.0	5.2	**0.0**	0.2	2.4	2.3
rat99	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.8	**0.0**	**0.0**	**0.0**	**0.0**	0.5	1.4	3.1	**0.0**	**0.0**	**0.0**	3.0	4.0
rd100	**0.0**	7.8	**0.0**	**0.0**	**0.0**	**0.0**	2.0	0.6	**0.0**	**0.0**	**0.0**	3.4	1.1	**0.0**	**0.0**	**0.0**	**0.0**	1.6	1.8	**0.0**	**0.0**	**0.0**	1.6	4.1
rd400	0.6	2.6	**0.0**	0.5	0.4	1.8	-	1.9	**0.0**	1.5	1.1	-	-	4.5	**1.3**	3.6	1.7	-	-	2.5	**1.1**	2.3	1.8	-
si175	**0.0**	3.1	**0.0**	**0.0**	7.2	3.9	2.7	3.6	**0.0**	**0.0**	7.5	5.8	2.9	4.8	**0.0**	0.1	6.9	5.6	2.5	6.1	**0.0**	0.3	6.1	5.6
st70	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.1	**0.0**	**0.0**	**0.0**	**0.0**	1.9	**0.0**	**0.0**	**0.0**	5.2	3.0
swiss42	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**
ts225	2.0	0.2	**0.0**	**0.0**	0.5	2.7	1.2	1.0	**0.0**	**0.0**	0.7	2.0	2.6	3.1	**0.0**	**0.0**	0.6	3.4	4.3	4.8	**0.0**	0.6	0.7	3.2
tsp225	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.6	3.0	**0.0**	**0.0**	0.6	2.7	2.3	7.1	**0.0**	0.1	2.7	2.8	2.6	6.8	**0.0**	4.5	2.6	3.3
u159	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.3	0.8	**0.0**	**0.0**	**0.0**	2.5	1.4	4.7	**0.0**	**0.0**	1.8	4.1	1.2	1.4	**0.0**	**0.0**	1.0	2.9
ulysses16	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
ulysses22	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**

Table A3: Achieved times (in seconds) for each instance of I_2 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$
a280	1015	2984	546	**232**	3600	3219	3601	3602	3600	**741**	3600	3601	3601	3603	3600	**2818**	3600	3602	3602	3602	**3600**	3616	**3600**	3603
att48	13	59	**4**	**4**	27	19	25	103	**13**	16	46	65	226	88	65	**8**	3600	716	246	109	127	**20**	3600	561
bayg29	3	4	**1**	**1**	5	4	8	5	5	**3**	11	8	33	43	41	**7**	1868	28	34	26	16	**9**	3571	43
bays29	4	2	**1**	**1**	2	4	13	19	8	**3**	26	13	21	27	13	**7**	109	19	25	16	17	**2**	115	31
berlin52	9	27	**2**	6	**2**	10	595	167	30	**15**	606	440	359	264	232	**40**	3600	636	768	157	252	**29**	3600	682
bier127	443	3600	**55**	347	894	694	1635	3601	**147**	485	3600	3601	3604	3601	3600	**838**	3600	3603	3601	3601	**316**	334	3600	3601
brazil58	29	205	**5**	10	37	23	93	138	49	**20**	2862	114	497	297	84	**31**	3600	1140	967	439	165	**49**	3600	1332
brg180	551	2251	**34**	129	3600	946	3606	1685	**101**	195	3600	3607	3614	3601	1109	**501**	3600	3637	3604	3601	2183	**538**	3600	3622
burma14	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ch130	186	302	55	**46**	3600	304	3603	2271	370	**228**	3600	3603	3617	3601	3600	**349**	3600	3603	3601	3541	864	**212**	3600	3602
ch150	218	284	66	**49**	3600	327	3602	1485	3600	**156**	3600	3607	3602	3601	3320	**222**	3600	3601	3602	3201	3600	**504**	3600	3602
d198	1446	3234	**149**	441	1396	3601	3606	3601	3600	**1038**	3600	3607	3603	3601	**2256**	3605	3600	3601	3603	3601	3600	**1360**	3600	3601
d493	3601	3602	**3600**	3605	**3600**	3609	3627	3602	**3600**	3614	**3600**	3618	3654	3608	**3600**	3601	**3600**	3610	**3600**	3616	**3600**	3603	3601	**3600**
dantzig42	2	19	**0**	1	1	3	36	109	31	**11**	85	150	118	103	49	**13**	3011	112	144	168	85	**12**	3600	283
eil101	477	726	104	**70**	1817	1572	3600	1181	329	**71**	3600	3622	3601	1944	397	**182**	3600	3600	3601	3601	3600	**247**	3600	3601
eil51	30	198	**5**	18	43	30	151	170	44	**24**	1746	472	424	198	85	**22**	3600	810	1045	287	309	**32**	3600	1253
eil76	61	30	7	**6**	137	68	463	568	54	**36**	1374	648	3601	1059	281	**148**	3600	3600	3601	1774	2311	**143**	3600	3600
fl417	2386	3600	3600	**729**	3600	3322	3613	3601	3600	**1306**	3600	3608	3604	3602	**3600**	3601	**3600**	3601	3603	3601	**3600**	3605	**3600**	3601
fri26	5	16	**1**	2	2	5	9	17	3	**2**	17	12	15	4	5	**2**	28	16	26	31	23	**5**	2716	33
gil262	3619	2071	3600	**294**	3600	3600	3624	3602	3600	**1626**	3600	3607	3602	**3600**	**3600**	3606	**3600**	3605	3647	3602	3600	**1637**	3600	3604
gr120	130	380	**11**	45	192	162	1770	436	196	**56**	3600	3618	3602	1617	350	**103**	3600	3624	3601	3601	1582	**667**	3600	3602
gr137	570	1939	**84**	322	497	842	3309	1316	**271**	319	3600	3602	3603	1825	891	**245**	3600	3601	3601	3601	**3600**	3601	**3600**	3602
gr17	**0**	1	**0**	**0**	**0**	**0**	1	7	**0**	1	1	1	**0**	1	**0**	**0**	1	**0**	1	2	**0**	1	5	1
gr202	943	3601	**78**	424	3600	745	3604	3601	**1107**	2407	3600	3605	3602	3601	**3600**	3609	**3600**	3603	3603	3602	**3600**	3601	**3600**	3603
gr21	**0**	1	**0**	**0**	**0**	**0**	3	3	**1**	**1**	2	3	4	10	3	**1**	19	4	14	12	14	**2**	26	11
gr229	3605	3602	2210	**1639**	3600	3602	3641	3602	3600	**2590**	3600	3616	3602	3602	3600	**1922**	3600	3602	3604	3602	3600	**3432**	3600	3603
gr24	3	6	**1**	**1**	2	3	1	1	**0**	**0**	2	2	3	2	3	**0**	18	3	15	13	7	**2**	50	13
gr431	3601	3601	**3600**	3615	**3600**	3606	3607	3603	**3600**	3615	**3600**	3604	3609	3606	3601	3603	**3600**	3619	3666	3607	**3600**	3602	**3600**	3634
gr48	27	78	**5**	12	50	33	95	151	**27**	28	284	111	193	113	100	**10**	3600	489	738	212	462	**21**	3600	763
gr96	81	175	**20**	24	1200	137	3600	2373	**307**	320	3600	3601	3601	2299	489	**183**	3600	3601	3604	2873	2544	**121**	3600	3617
hk48	19	61	**3**	5	62	14	57	278	**17**	**17**	135	67	390	398	131	**82**	3600	477	338	153	282	**23**	3600	337
kroA100	257	1062	**54**	100	3600	970	653	2046	**85**	387	2444	3324	3600	1327	491	**111**	3600	3616	3600	1374	2769	**246**	3600	3602
kroA150	125	252	112	**46**	3600	322	3603	1748	709	**271**	3600	3605	3600	3601	1655	**514**	3600	3604	3603	3095	3600	**403**	3600	3651
kroA200	2677	766	1028	**96**	3600	3422	3607	2758	**367**	381	3600	3626	3602	3601	3600	**1365**	3600	3630	3603	3602	3600	**1129**	3600	3603
kroB100	184	236	**21**	47	534	224	1581	852	151	**42**	3600	3601	3600	1256	1309	**108**	3600	3601	3601	1007	463	**83**	3600	3600
kroB150	2791	545	1391	**83**	3600	3601	3603	3601	**719**	853	3600	3602	3601	1922	2832	**243**	3600	3604	3609	3601	3600	**601**	3600	3605
kroB200	2236	3600	**157**	454	3600	1284	3603	3601	3600	**416**	3600	3600	3604	3601	3600	**608**	3600	3601	3602	3602	3600	**1064**	3600	3601
kroC100	86	286	**9**	19	133	149	952	320	59	**47**	3600	3601	3600	602	325	**70**	3600	3601	3600	3140	1090	**215**	3600	3609
kroD100	93	354	**13**	42	328	135	3600	1808	901	**124**	3600	3601	3602	669	478	**148**	3600	3600	3605	3550	1566	**170**	3600	3601
kroE100	174	543	151	**84**	3600	883	3602	1000	856	**122**	3600	3600	3600	1071	500	**94**	3600	3600	3603	1321	3600	**184**	3600	3606
lin105	161	332	**37**	42	1901	153	3603	1343	760	**221**	3600	3602	3600	1245	633	**183**	3600	3612	3601	3601	862	**534**	3600	3601
lin318	2741	2982	2857	**397**	3600	3600	3643	3603	3600	**886**	3600	3640	3602	3605	3600	**2914**	3600	3603	3609	3607	3600	**2785**	3600	3604
linhp318	3601	1355	2837	**678**	3600	3606	3601	3602	**3600**	**3600**	**3600**	3663	3602	3605	**3600**	3603	**3600**	3602	3607	3603	**3600**	3615	**3600**	3610
pcb442	3613	3168	3600	**830**	3600	3626	3610	3603	3600	**2685**	3600	3601	3608	3610	**3600**	3603	**3600**	3609	3667	3612	**3600**	3605	**3600**	3639

Continued on next page

Table A3: Achieved times (in seconds) for each instance of I_2 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$
pr107	8	**0**	**0**	**0**	**0**	8	180	618	**38**	**38**	668	484	26	**1**	**1**	**1**	**1**	26	44	**1**	**1**	**1**	**1**	41
pr124	1241	879	409	**218**	3600	1176	3602	1420	3190	**193**	3600	3600	3604	3600	3600	**797**	3600	3608	3602	2336	3600	**241**	3600	3603
pr136	377	289	**50**	58	1814	451	1398	544	**134**	550	3600	2795	3601	2311	1482	**232**	3600	3603	3602	1235	2410	**383**	3600	3602
pr144	3601	470	1934	**73**	3600	3601	3602	1315	1244	**72**	3600	3601	3601	3601	2853	**312**	3600	3648	3604	2308	3600	**294**	3600	3601
pr152	80	585	1012	**37**	3600	181	3600	3600	610	**156**	3600	3600	3604	3601	3600	**1624**	3600	3601	3603	3601	**3600**	3605	**3600**	3601
pr226	288	610	3600	**62**	3600	898	3601	3600	3600	**229**	3600	3603	3610	3601	3600	**1408**	3600	3602	3601	3602	3600	**1061**	3600	3601
pr264	501	1949	546	**365**	3600	419	3602	3600	3600	**521**	3600	3606	3605	3601	**3600**	3604	**3600**	3601	3601	3601	3600	**1942**	3600	3712
pr299	1049	3601	2088	**542**	3600	3601	3610	3603	3600	**2530**	3600	3601	3604	3603	3600	**3145**	3600	3604	3617	3603	**3600**	3604	**3600**	3608
pr439	3603	3602	3600	**1709**	3600	3602	3604	3604	**3600**	3613	**3600**	3611	3618	3605	**3600**	3610	**3600**	3609	3629	3602	**3600**	3602	**3600**	3629
pr76	73	2679	**27**	388	578	283	1853	825	**105**	106	3600	3084	948	377	263	**48**	1403	3600	3601	3600	**517**	595	3600	3600
rat195	682	547	650	**121**	3600	3019	3610	1193	274	**202**	3600	3602	3606	3601	3600	**2425**	3600	3603	3605	3601	3600	**1469**	3600	3601
rat99	189	256	32	**21**	816	247	3601	756	236	**56**	3600	3601	3601	1545	822	**114**	3600	3601	3601	3600	**338**	410	3600	3604
rd100	490	236	82	**19**	3567	1820	1359	563	139	**63**	3600	3600	3615	1097	996	**75**	3600	3610	3602	1482	2710	**136**	3600	3600
rd400	3606	3601	3600	**1872**	3600	3600	3618	3608	3600	**2560**	3600	3627	3619	3608	**3600**	3601	**3600**	3633	3641	3610	**3600**	3603	3601	3634
si175	902	1189	249	**157**	3600	1200	3607	3601	3600	**466**	3600	3605	3617	3603	3600	**639**	3600	3606	3601	3603	3600	**799**	3600	3626
st70	102	525	**28**	52	552	90	1082	669	**72**	110	3600	1577	919	585	115	**114**	3600	3600	3099	633	889	**84**	3600	3600
swiss42	32	67	**7**	9	52	31	44	44	20	**8**	242	129	65	78	33	**16**	612	193	550	90	148	**8**	3600	498
ts225	777	1479	**140**	351	3600	1136	3601	3601	1209	**928**	3600	3606	3603	3602	3600	**959**	3600	3609	3603	3602	**3600**	3601	**3600**	3603
tsp225	827	3280	798	**356**	3600	1134	3601	3602	3600	**1882**	3600	3612	3602	3602	3600	**2164**	3600	3627	3602	3602	3600	**3005**	3600	3602
u159	612	537	130	**48**	3600	3097	3601	3601	1768	**388**	3600	3601	3602	3600	3600	**423**	3600	3617	3603	3601	3600	**643**	3600	3602
ulysses16	**0**	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ulysses22	1	2	**0**	**0**	1	3	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

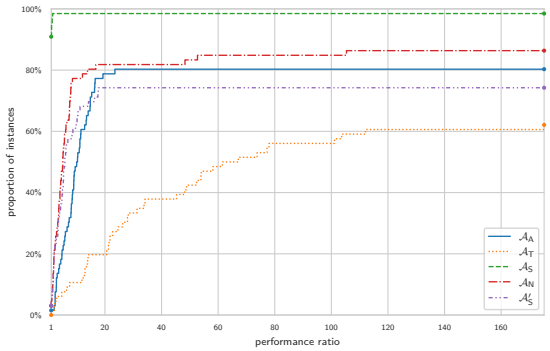
Table A4: Achieved gaps (as percentages) for each instance of I_2 and approach.

Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$
a280	**0.0**	**0.0**	**0.0**	**0.0**	2.6	**0.0**	3.4	0.4	1.8	**0.0**	4.4	5.2	5.2	5.8	3.5	**0.0**	5.4	5.9	9.1	5.3	4.2	**0.0**	5.7	9.1
att48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.7	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.1	**0.0**
bayg29	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
bays29	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
berlin52	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	5.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	5.5	**0.0**
bier127	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	**0.0**	**0.0**	0.6	2.2	6.8	6.0	0.9	**0.0**	6.7	9.2	0.8	0.2	**0.0**	**0.0**	0.8	3.3
brazil58	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	7.0	**0.0**
brg180	**0.0**	**0.0**	**0.0**	**0.0**	4.3	**0.0**	1.7	**0.0**	**0.0**	**0.0**	7.1	4.3	6.0	1.8	**0.0**	**0.0**	9.5	9.3	5.5	1.0	**0.0**	**0.0**	9.2	8.3
burma14	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
ch130	**0.0**	**0.0**	**0.0**	**0.0**	2.5	**0.0**	4.4	**0.0**	**0.0**	**0.0**	5.8	4.7	4.9	0.4	0.8	**0.0**	6.5	7.7	4.1	**0.0**	**0.0**	**0.0**	5.5	6.6
ch150	**0.0**	**0.0**	**0.0**	**0.0**	3.6	**0.0**	5.2	**0.0**	2.4	**0.0**	5.9	6.3	6.4	0.9	**0.0**	**0.0**	6.2	8.5	5.7	**0.0**	2.7	**0.0**	7.4	7.4
d198	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.3	3.4	0.7	3.1	**0.0**	4.1	5.6	3.8	4.3	**0.0**	**0.0**	2.0	2.2	4.4	5.4	3.6	**0.0**	4.1	5.2
d493	1.6	1.8	**0.7**	2.1	1.9	1.8	-	2.9	**1.1**	1.8	2.0	-	-	3.4	**1.5**	2.7	2.5	-	-	3.2	**1.7**	2.8	2.8	-
dantzig42	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.5	**0.0**
eil101	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	6.5	**0.0**	**0.0**	**0.0**	5.0	7.7	3.1	**0.0**	**0.0**	**0.0**	6.0	5.7	6.6	2.7	3.5	**0.0**	9.2	8.8
eil51	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	6.8	**0.0**
eil76	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.2	**0.0**	**0.0**	**0.0**	5.8	4.9	6.3	**0.0**	**0.0**	**0.0**	7.5	7.0
fl417	**0.0**	1.2	1.3	**0.0**	2.5	**0.0**	2.5	2.0	1.4	**0.0**	3.0	2.3	2.9	5.0	2.4	**0.3**	3.9	3.1	4.9	6.3	4.0	**0.1**	5.4	5.3
fri26	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gil262	1.7	**0.0**	2.0	**0.0**	4.3	3.8	5.3	5.0	2.2	**0.0**	4.8	6.6	4.3	3.7	1.8	**0.3**	4.7	4.3	7.0	5.6	3.4	**0.0**	5.7	9.9
gr120	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	4.4	4.2	2.8	**0.0**	**0.0**	**0.0**	5.9	4.7	5.0	5.5	**0.0**	**0.0**	6.0	6.0
gr137	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.7	4.7	3.4	**0.0**	**0.0**	**0.0**	4.8	6.0	3.1	0.2	0.9	**0.0**	4.5	6.1
gr17	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr202	**0.0**	**0.0**	**0.0**	**0.0**	0.7	**0.0**	1.9	3.5	**0.0**	**0.0**	3.5	2.8	3.0	3.4	1.4	**0.0**	3.1	4.6	2.8	5.5	1.9	**0.4**	3.4	5.5
gr21	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr229	2.8	3.4	**0.0**	**0.0**	3.1	3.1	5.0	1.5	2.4	**0.0**	3.1	4.2	1.5	3.2	1.0	**0.0**	2.1	2.8	4.7	2.9	2.6	**0.0**	5.3	7.6
gr24	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
gr431	1.9	2.1	0.8	**0.7**	1.6	2.4	2.7	4.5	**1.8**	**1.8**	3.0	3.6	-	5.4	**0.7**	4.6	2.8	-	-	4.8	**2.8**	4.2	3.7	-
gr48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.9	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	5.5	**0.0**
gr96	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.1	**0.0**	**0.0**	**0.0**	5.6	5.5	3.9	**0.0**	**0.0**	**0.0**	6.7	5.3	5.7	**0.0**	**0.0**	**0.0**	6.6	9.6
hk48	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.7	**0.0**
kroA100	**0.0**	**0.0**	**0.0**	**0.0**	1.7	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.0	**0.0**	**0.0**	**0.0**	5.7	8.5	4.8	**0.0**	**0.0**	**0.0**	6.0	9.7
kroA150	**0.0**	**0.0**	**0.0**	**0.0**	4.2	**0.0**	4.0	**0.0**	**0.0**	**0.0**	4.2	5.7	4.3	0.1	**0.0**	**0.0**	5.6	6.9	6.0	**0.0**	3.9	**0.0**	5.2	8.6
kroA200	**0.0**	**0.0**	**0.0**	**0.0**	3.1	**0.0**	2.3	**0.0**	**0.0**	**0.0**	2.2	2.4	7.2	2.9	2.4	**0.0**	4.8	6.6	9.7	3.5	3.7	**0.0**	5.3	11.4
kroB100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.7	5.7	5.7	**0.0**	**0.0**	**0.0**	7.5	8.2	5.6	**0.0**	**0.0**	**0.0**	4.6	7.7
kroB150	**0.0**	**0.0**	**0.0**	**0.0**	5.5	1.3	3.6	0.1	**0.0**	**0.0**	3.9	5.6	5.2	**0.0**	**0.0**	**0.0**	6.4	7.2	5.5	1.0	2.6	**0.0**	6.1	8.0
kroB200	**0.0**	0.1	**0.0**	**0.0**	2.0	**0.0**	3.9	0.1	3.1	**0.0**	4.9	6.8	4.0	0.3	0.7	**0.0**	3.8	7.0	5.7	0.6	3.4	**0.0**	5.4	8.4
kroC100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	0.8	3.2	**0.0**	**0.0**	**0.0**	6.2	8.2	6.1	**0.0**	**0.0**	**0.0**	6.3	10.0
kroD100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.8	**0.0**	**0.0**	**0.0**	5.2	5.7	3.7	**0.0**	**0.0**	**0.0**	6.0	6.2	5.2	**0.0**	**0.0**	**0.0**	8.3	7.1
kroE100	**0.0**	**0.0**	**0.0**	**0.0**	4.0	**0.0**	2.5	**0.0**	**0.0**	**0.0**	7.9	5.9	3.7	**0.0**	**0.0**	**0.0**	5.5	6.5	5.9	**0.0**	2.8	**0.0**	7.6	7.4
lin105	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**	**0.0**	**0.0**	3.2	3.6	4.2	**0.0**	**0.0**	**0.0**	5.4	5.2	3.8	0.3	**0.0**	**0.0**	5.3	6.3
lin318	**0.0**	**0.0**	**0.0**	**0.0**	3.2	1.3	5.7	2.1	3.0	**0.0**	5.5	5.9	8.3	2.3	4.0	**0.0**	5.6	8.3	7.3	5.0	3.1	**0.0**	6.1	8.1
linhp318	1.2	**0.0**	**0.0**	**0.0**	2.2	2.2	4.5	2.0	2.6	**0.1**	4.6	4.0	8.3	5.4	4.5	**0.2**	5.4	8.3	9.5	6.0	3.3	**0.1**	4.9	9.5
pcb442	1.3	**0.0**	1.0	**0.0**	2.7	2.1	3.4	3.4	2.9	**0.0**	5.2	4.1	5.8	4.5	3.2	**1.0**	4.8	5.8	-	6.1	3.6	**3.1**	5.0	-

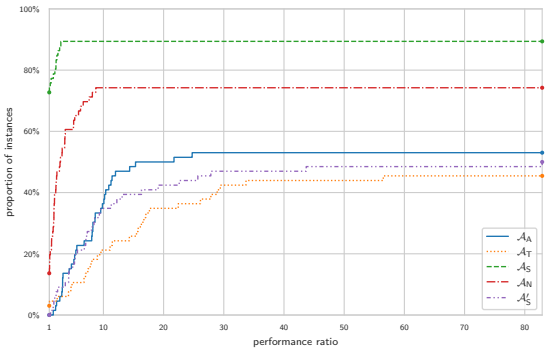
Continued on next page

Table A4: Achieved gaps (as percentages) for each instance of I_2 and approach.

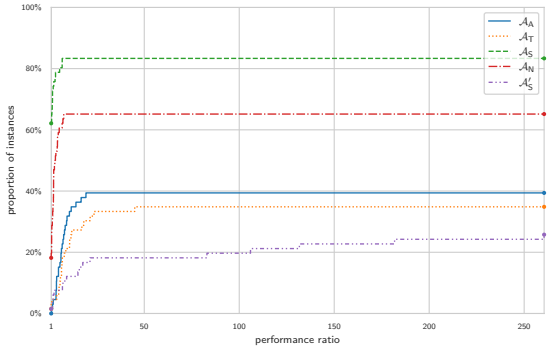
Instances	10 trips						20 trips						30 trips						40 trips					
	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$	$\mathcal{A}_{\text{A}}$	$\mathcal{A}_{\text{T}}$	$\mathcal{A}_{\text{S}}$	$\mathcal{A}_{\text{N}}$	$\mathcal{A}'_{\text{S}}$	$\mathcal{A}'_{\text{A}}$
pr107	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
pr124	**0.0**	**0.0**	**0.0**	**0.0**	4.9	**0.0**	4.0	**0.0**	**0.0**	**0.0**	6.0	3.9	6.7	**0.0**	3.2	**0.0**	7.2	6.0	8.3	**0.0**	4.2	**0.0**	7.8	8.8
pr136	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.5	**0.0**	4.8	**0.0**	**0.0**	**0.0**	5.0	6.4	5.0	**0.0**	**0.0**	**0.0**	6.4	6.8
pr144	0.8	**0.0**	**0.0**	**0.0**	5.5	0.8	5.2	**0.0**	**0.0**	**0.0**	4.8	4.9	5.8	**0.0**	**0.0**	**0.0**	6.6	6.5	7.5	**0.0**	3.4	**0.0**	7.2	9.3
pr152	**0.0**	**0.0**	**0.0**	**0.0**	4.2	**0.0**	1.6	**0.0**	**0.0**	**0.0**	3.8	4.6	6.1	3.7	2.4	**0.0**	5.4	7.2	10.8	3.2	5.2	**0.0**	8.0	11.5
pr226	**0.0**	**0.0**	1.4	**0.0**	4.1	**0.0**	5.4	2.6	4.2	**0.0**	7.1	6.2	4.4	1.0	3.0	**0.0**	5.0	5.6	6.7	5.8	4.4	**0.0**	6.1	8.9
pr264	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**	2.9	0.3	1.8	**0.0**	3.1	2.0	3.2	5.3	1.5	**0.2**	3.0	3.7	5.5	9.1	3.3	**0.0**	5.3	7.4
pr299	**0.0**	**0.0**	**0.0**	**0.0**	2.3	0.7	2.5	0.8	1.2	**0.0**	3.2	4.1	4.0	2.0	2.0	**0.0**	3.3	4.9	6.3	9.2	2.4	**1.6**	4.4	6.3
pr439	1.0	1.2	1.0	**0.0**	2.0	2.4	3.6	3.6	2.1	**0.4**	3.1	4.1	207.6	3.0	2.3	**1.1**	3.8	-	-	3.7	2.7	**2.4**	3.6	-
pr76	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.7	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	5.2	5.2	0.1	**0.0**	**0.0**	5.6	7.4
rat195	**0.0**	**0.0**	**0.0**	**0.0**	4.2	**0.0**	1.7	**0.0**	**0.0**	**0.0**	2.0	2.5	4.4	5.1	2.1	**0.0**	5.0	7.3	5.0	2.6	1.5	**0.0**	5.2	5.2
rat99	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	2.3	**0.0**	**0.0**	**0.0**	5.7	5.6	4.2	**0.0**	**0.0**	**0.0**	5.7	4.7	2.7	**0.0**	**0.0**	**0.0**	4.8	5.9
rd100	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	4.4	6.1	6.0	**0.0**	**0.0**	**0.0**	8.2	11.5	5.7	**0.0**	**0.0**	**0.0**	8.1	10.1
rd400	0.8	0.3	1.4	**0.0**	3.0	1.7	3.3	0.8	2.3	**0.0**	4.3	4.7	5.6	3.0	2.7	**1.7**	4.2	5.6	-	6.0	3.9	**2.8**	5.3	-
si175	**0.0**	**0.0**	**0.0**	**0.0**	6.9	**0.0**	8.4	3.1	3.4	**0.0**	14.7	12.8	10.9	7.6	5.3	**0.0**	18.4	16.4	9.6	9.0	3.8	**0.0**	15.2	14.6
st70	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	3.5	1.9	**0.0**	**0.0**	**0.0**	**0.0**	7.3	6.3
swiss42	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**
ts225	**0.0**	**0.0**	**0.0**	**0.0**	1.6	**0.0**	1.8	0.3	**0.0**	**0.0**	3.8	3.7	5.2	5.3	2.2	**0.0**	4.0	6.8	6.0	6.8	2.2	**0.0**	5.4	6.3
tsp225	**0.0**	**0.0**	**0.0**	**0.0**	2.5	**0.0**	4.8	3.3	2.7	**0.0**	5.2	5.7	4.9	7.9	2.9	**0.0**	5.7	6.5	6.9	9.1	3.2	**0.0**	5.1	8.5
u159	**0.0**	**0.0**	**0.0**	**0.0**	1.4	**0.0**	2.8	4.4	**0.0**	**0.0**	4.0	5.0	4.3	0.3	1.4	**0.0**	4.8	5.5	3.9	0.1	2.2	**0.0**	4.9	6.1
ulysses16	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
ulysses22	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**



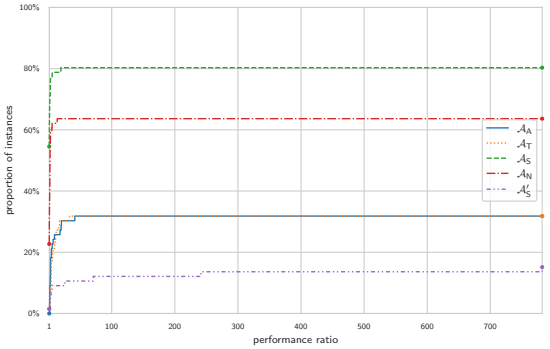
(a) I_1, 10 trips



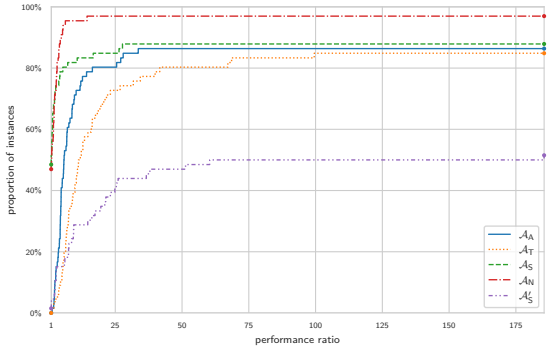
(b) I_1, 20 trips



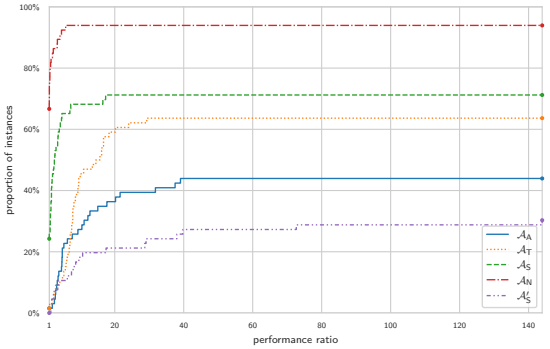
(c) I_1, 30 trips



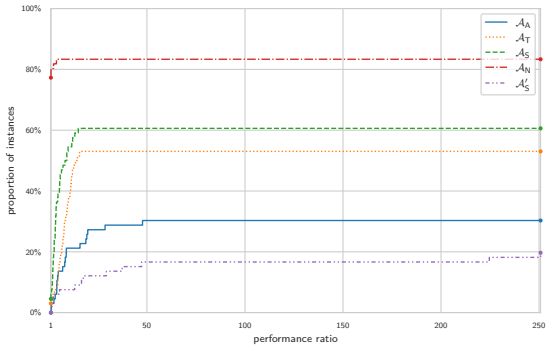
(d) I_1, 40 trips



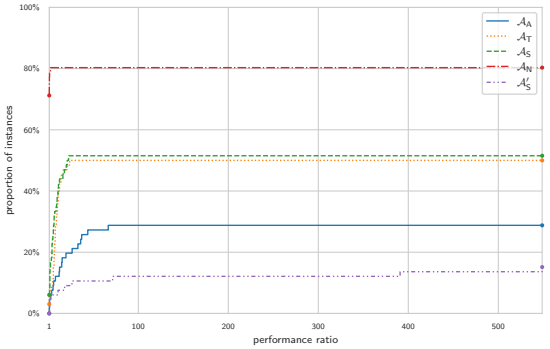
(e) I_2, 10 trips



(f) I_2, 20 trips

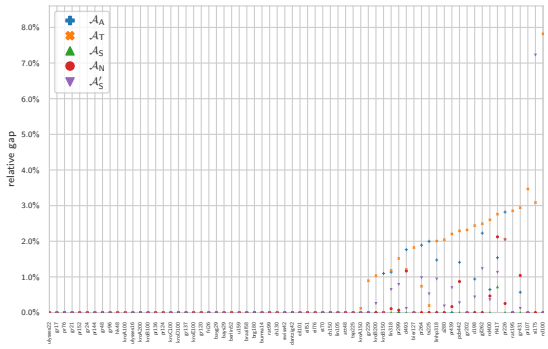


(g) I_2, 30 trips

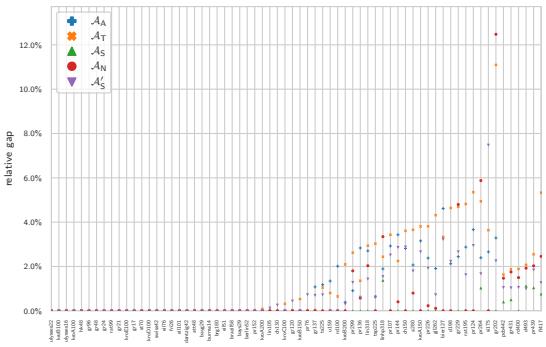


(h) I_2, 40 trips

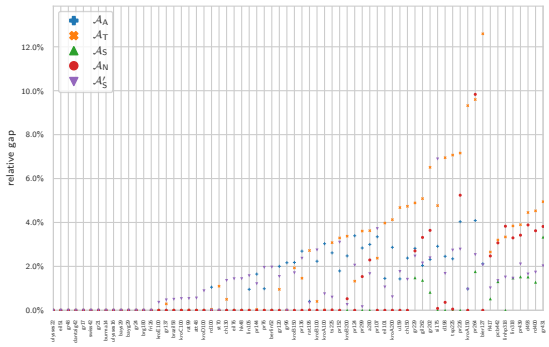
Figure 14: Performance profiles with respect to the solving times.



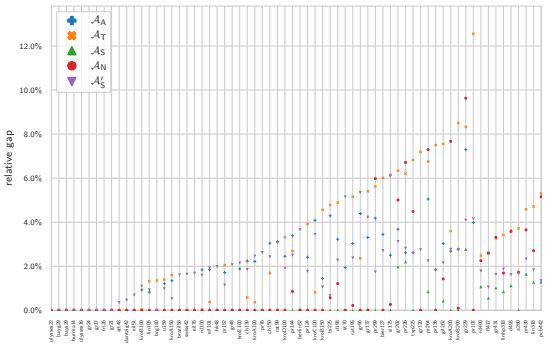
(a) I_1, 10 trips



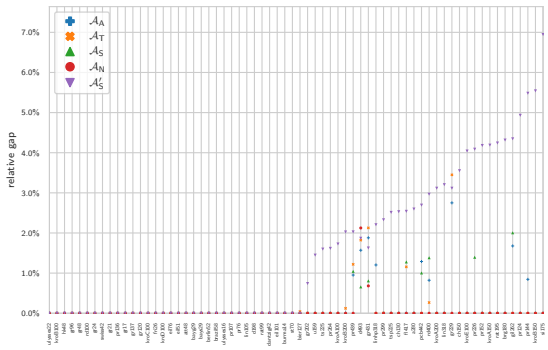
(b) I_1, 20 trips



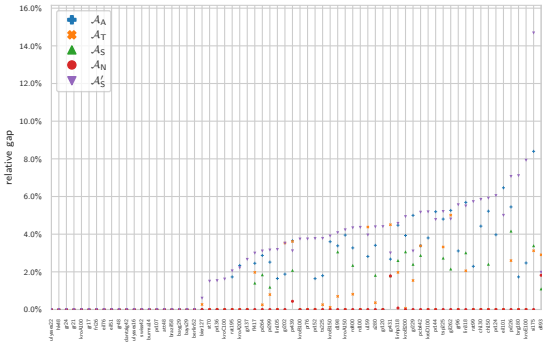
(c) I_1, 30 trips



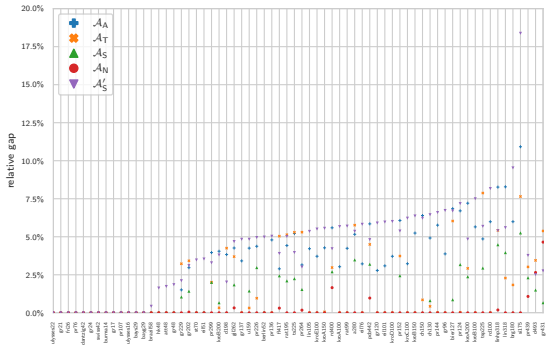
(d) I_1, 40 trips



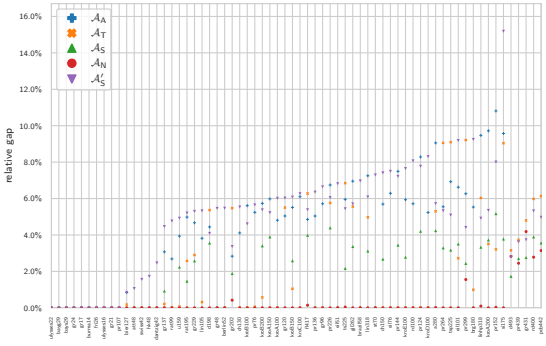
(e) I_2, 10 trips



(f) I_2, 20 trips

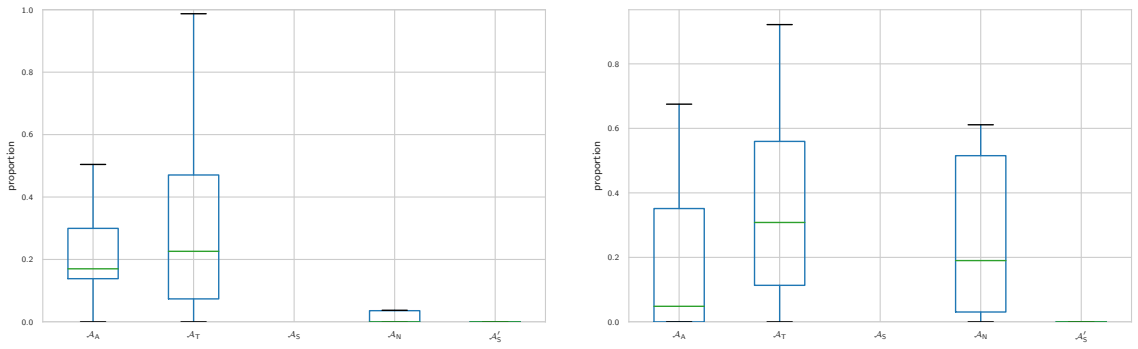


(g) I_2, 30 trips



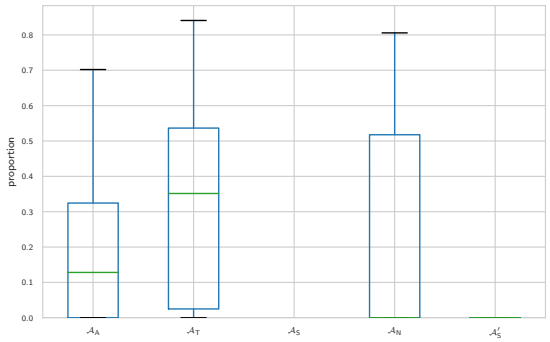
(h) I_2, 40 trips

Figure 15: Achieved gaps.

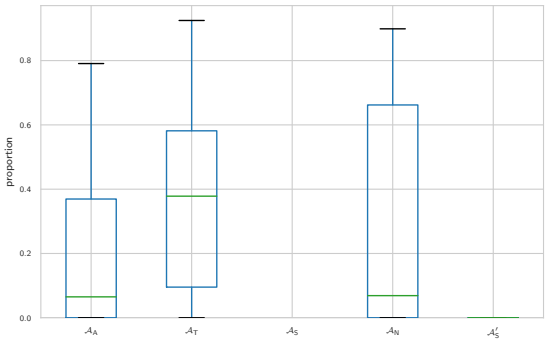


(a) I_1, 10 trips

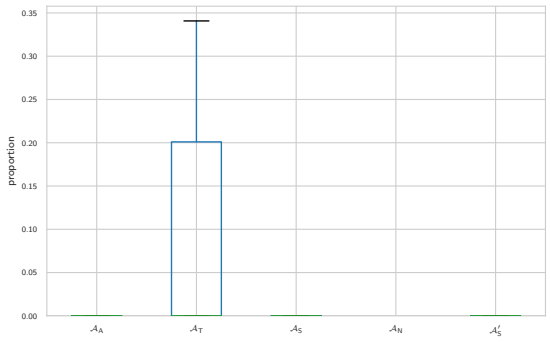
(b) I_1, 20 trips



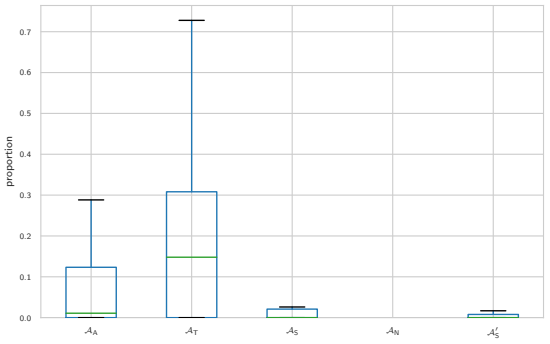
(c) I_1, 30 trips



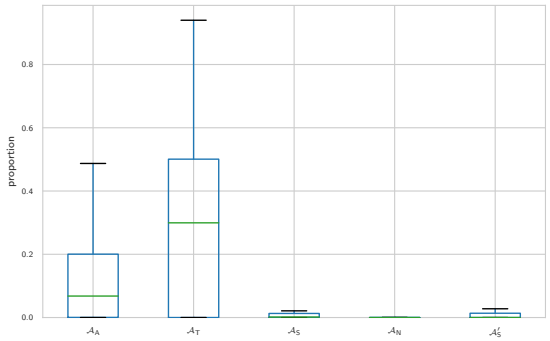
(d) I_1, 40 trips



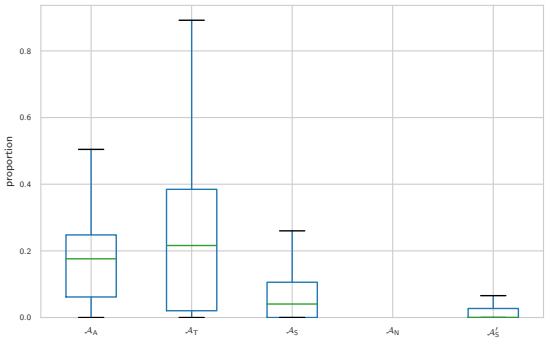
(e) I_2, 10 trips



(f) I_2, 20 trips

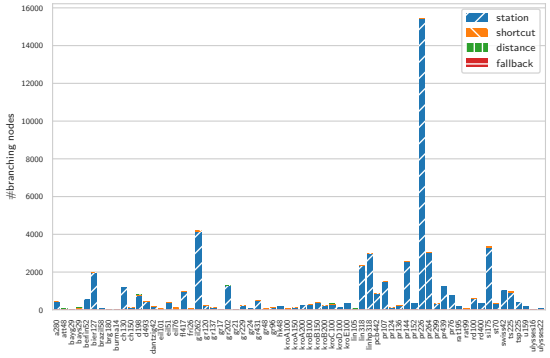


(g) I_2, 30 trips

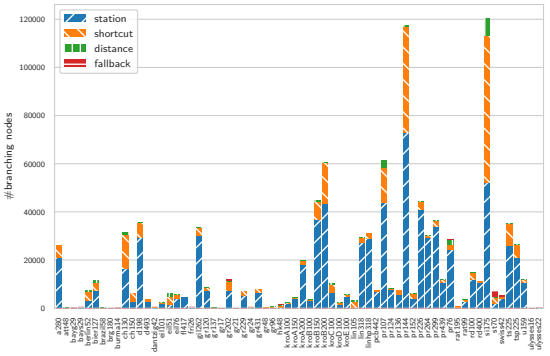


(h) I_2, 40 trips

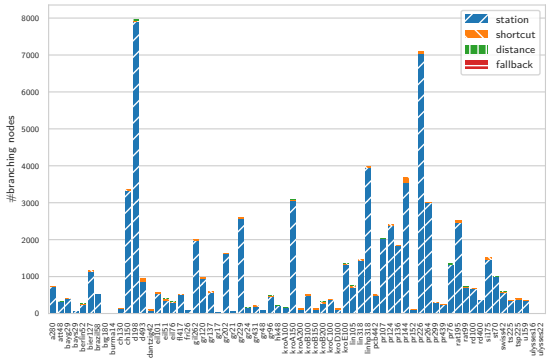
Figure 16: Primal parts of the achieved gaps.



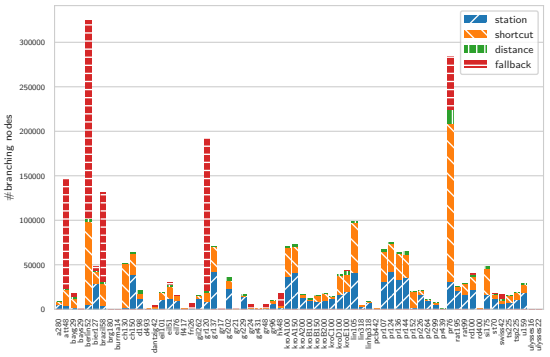
(a) $\mathcal{A}_S$, 10 trips



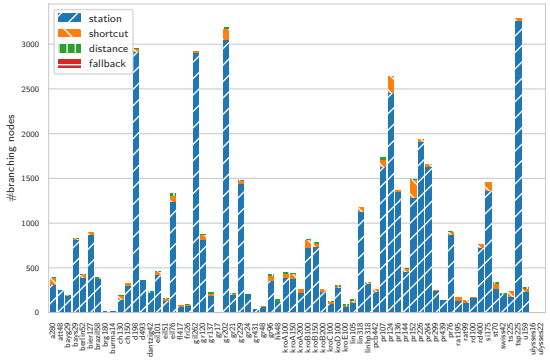
(b) $\mathcal{A}'_S$, 10 trips



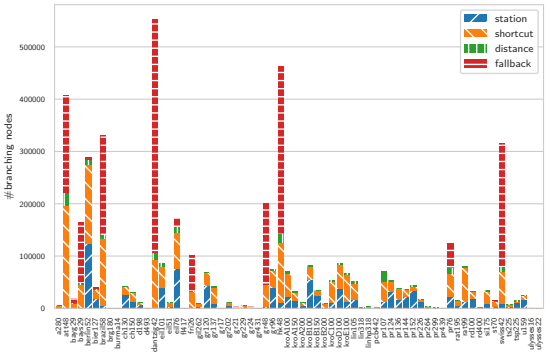
(c) $\mathcal{A}_S$, 20 trips



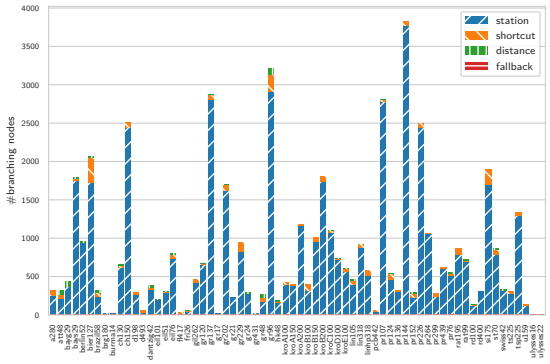
(d) $\mathcal{A}'_S$, 20 trips



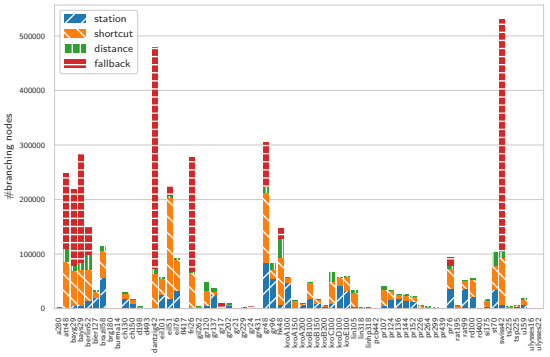
(e) $\mathcal{A}_S$, 30 trips



(f) $\mathcal{A}'_S$, 30 trips



(g) $\mathcal{A}_S$, 40 trips



(h) $\mathcal{A}'_S$, 40 trips

Figure 17: Number of tree nodes created by the branch-and-price approaches $\mathcal{A}_S$ and $\mathcal{A}'_S$ (instance set I_1).

(a) $\mathcal{A}_S$, 10 trips

(b) $\mathcal{A}'_S$, 10 trips

(c) $\mathcal{A}_S$, 20 trips

(d) $\mathcal{A}'_S$, 20 trips

(e) $\mathcal{A}_S$, 30 trips

(f) $\mathcal{A}'_S$, 30 trips

(g) $\mathcal{A}_S$, 40 trips

(h) $\mathcal{A}'_S$, 40 trips

Figure 18: Number of tree nodes created by the branch-and-price approaches $\mathcal{A}_S$ and $\mathcal{A}'_S$ (instance set I_2).