CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Valid Inequalities and a Branch-and-Cut Algorithm for Asymmetric Multi-Depot Routing Problems

Michiel A.J. Uit het Broek
Albert H. Schrotenboer
Bolor Jargalsaikhan
Kees Jan Roodbergen
Leandro C. Coelho

January 2019

CIRRELT-2019-02

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2019-002.

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Valid Inequalities and a Branch-and-Cut Algorithm for Asymmetric Multi-Depot Routing Problems

Michiel A.J. Uit het Broek[1], Albert H. Schrotenboer[1], Bolor Jargalsaikhan[1], Kees Jan Roodbergen[1], Leandro C. Coelho[2,*]

1. Department of Operations, Faculty of Economics and Business, University of Groningen, Nettelbosje 2, 9747 AE Groningen, The Netherlands michieluithetbroek@gmail.com, a.h.schrotenboer@rug.nl, bolor.jargalsaikhan@gmail.com, k.j.roodbergen@rug.nl
2. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Operations and Decision Systems, 2325 de la Terrasse, Université Laval, Québec, Canada G1V 0A6

Abstract. We present a generic branch-and-cut framework for solving routing problems with multiple depots and asymmetric cost-structures, which consist in finding a set of cost minimizing (capacitated) vehicle tours in order to fulfill a set of customer demands. The backbone of the branch-and-cut framework is a series of valid inequalities, and accompanying separation algorithms, exploiting the asymmetric cost-structure in directed graphs. We derive three new classes of so-called D_k inequalities that can eliminate subtours, enforce tours to be linked to a single depot, and impose bounds on the number of allowed customers in a tour. In addition, other well-known valid inequalities for solving vehicle routing problems are generalized and adapted to be valid for routing problems with multiple depots and asymmetric cost-structures. The resulting branch-and-cut framework is tested on four specific problem variants, for which we develop a new set of large-scale benchmark instances. The new D_k inequalities are able to reduce root node optimality gaps by up to 67.2% compared to existing approaches in the literature. The overall branch-and-cut framework is effective as, e.g., Asymmetric Multi-Depot Traveling Salesman Problem instances containing up to 400 customers and 50 depots can be solved to optimality, for which only solutions of instances up to 300 customer nodes and 60 depots were reported in the literature before.

Keywords. Branch-and-Cut, valid inequalities, asymmetric, multi-depot, vehicle routing.

Acknowledgements. This work was partly supported by the Netherlands Organisation for Scientific Research (NWO) through grants no 438-13-216 (Michiel A. J. Uit het Broek and Albert H. Schrotenboer) and no 439-16-612 (Kees Jan Roodbergen and Bolor Jargalsaikhan), and by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant 2014-05764 (Leandro C. Coelho).

---

* Corresponding author: Leandro.Coelho@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2019

1 Introduction

We study routing problems with *asymmetric costs* and *multiple depots*. These problems are defined on a directed graph $G = (V', A)$, where the node set V' is partitioned into a depot set $D = \{1, \dots, r\}$ and a customer set $V = \{r + 1, \dots, r + n\}$. The arc set is defined as $A = \{(i, j) \mid i \neq j \in V \vee i \in D, j \in V \vee i \in V, j \in D\}$. An *asymmetric* cost $c_{ij} \geq 0$ is incurred when traveling along arc $(i, j) \in A$, i.e., c_{ij} need not equal c_{ji}. Each available vehicle makes a single tour, starting and ending at the same depot. The objective is to find a set of cost minimizing tours which fulfill all customer demands.

Specifically, we focus on the following four problem variants:

(i) The Asymmetric Multi-Depot Traveling Salesman Problem (A-MDTSP): A single vehicle with unlimited capacity is available at each depot.

(ii) The Asymmetric Multi-Depot multiple Traveling Salesman Problem (A-MDmTSP): There are m vehicles available at each depot. Vehicle capacity is unlimited, but the number of customers per tour is bounded by $[\ell^{\min}, \ell^{\max}]$.

(iii) The Asymmetric Multi-Depot Capacitated Vehicle Routing Problem (A-MDCVRP): There are m vehicles available at each depot. Each vehicle has a limited capacity Q, and each customer has a demand $q_i > 0$ $(i \in V)$.

(iv) The Asymmetric Capacitated Location Routing Problem (A-CLRP): This problem extends the A-MDCVRP by incorporating depot location decisions, i.e., using a depot comes at a fixed cost $\tilde{c} \geq 0$.

We present a branch-and-cut framework that can address each of the four problem variants above. Furthermore, the design of our branch-and-cut framework is generic in the sense that – with some problem-specific, non-structural adaptations – it may be applied to other routing problems with multiple depots and asymmetric cost structures. This is relevant since such routing problems are nowadays often encountered in practice [8, 19, 20]. To indicate that results apply generically to routing problems with multiple depots and asymmetric costs, we make use of the descriptor Asymmetric Multi-depot Vehicle Routing Problem (A-MDVRP). Some results are specific for one of the four problem variants, in which case this is indicated explicitly. Numerical experiments are limited to the four problem variants.

The first contribution of the paper is a series of novel valid inequalities tailored to asymmetric cost structures and multiple depots, with accompanying separation algorithms, which are generalized from the asymmetric traveling salesman problem [9, 10]. The new valid inequalities make explicit use of the directness of the underlying graph, resulting in strengthened valid inequalities, whereas applying exact methods for symmetric routing problems on undirected graphs to the A-MDVRP leave this unexploited. We propose three new classes of so-called D_k inequalities that 1) eliminate subtours, 2) impose bounds on the tour size, and 3) eliminate paths that begin and arrive at different depots, i.e., the newly proposed D_k inequalities are not only valid but also model describing. Furthermore, CAT-inequalities are adapted to a multi-depot setting. To the best of our knowledge, there has been no study on the integer polytope of *asymmetric* routing problems since a series of works on the Asymmetric Traveling Salesman Problem (ATSP) and its variations [1, 2, 3, 9, 11, 10].

Our second contribution is the development of a generic branch-and-cut algorithm for the A-MDVRP that integrates the results from capacitated vehicle routing problems [15, 16, 18], location routing problems [6, 14], and symmetric multi-depot traveling salesman problems [7]. Along with the newly proposed valid inequalities, our branch-and-cut algorithm includes the following adapted asymmetric valid inequalities: *subtour elimination constraints*, *classical path elimination constraints*, *CAT inequalities*, *D_k^+ and D_k^- constraints*, *strengthened comb inequalities*, *T- and $T1$-comb inequalities*, *framed capacity inequalities*, and *homogenous- and large- multistar inequalities.*

Third, complementary to the branch-and-cut algorithm, we use a compact formulation with so-called neighborhood-arc constraints (i.e., considering a subset of promising arcs only) in order to provide upper bounds in a computationally simple yet effective manner. Due to this upper bound procedure, the overall efficiency of the branch-and-cut algorithm improves considerably. We refer to the joint use of the branch-and-cut algorithm and the upper bound procedure as the *branch-and-cut framework.*

Fourth, we show the effectiveness of the branch-and-cut framework on the four problem variants as detailed above. For each problem variant, we provide insights into the effectiveness of the valid inequalities. Furthermore, to obtain a general understanding of how asymmetry in the cost structure relates to computational difficulty, we perform extensive experiments on three classes of asymmetric cost structures. These classes have arc costs ranging from completely random ($c_{ij} \sim U(0, 1000)$) to Euclidean with a small noise. The instances used are publicly available and can serve as benchmark instances for future research. Moreover, in order to foster research on asymmetric vehicle routing problems, we provide public access to our *C++* implementation of the branch-and-cut framework[1].

We show that our newly developed valid inequalities help to reduce root node optimality gaps by up to 67.2%. Regarding the A-MDTSP, we are able to solve all instances of Bektaş et al. [4] to optimality, of which two were not solved before, and find an improved solution for one of the instances. Furthermore, our branch-and-cut framework can solve to optimality our newly developed benchmark instances for the A-MDTSP containing up to 400 customers and 50 depots, which is significantly larger than the 300 customer and 60 depot instances of Bektaş et al. [4].

Regarding the other three problem variants, no benchmark instances exist for routing problems with asymmetric costs on directed graphs. Hence, we tested our branch-and-cut framework on newly proposed benchmark instances and it appears to be effective for all remaining problem variants. For example, A-MDmTSP instances with up to 100 customers and 20 depots can be solved to optimality. This size is comparable to state-of-the-art methods for the single-depot symmetric multiple traveling salesman problem, which has a significantly smaller solution space [5].

Furthermore, we compare the branch-and-cut framework to a basic branch-and-cut algorithm, where the latter is our implementation of what can be considered state-of-the art. The branch-and-cut framework that exploits our new valid inequalities shows considerable computational improvements over the basic algorithm. Specifically, the branch-and-cut framework can solve considerably more instances to optimality than the basic algorithm, is on average faster for the instances that are solved to optimality by both algorithms, has smaller optimality gaps for instances that cannot be solved to optimality, and provides higher lower bounds for the instances for which no upper bound is found.

The remainder of the paper is organized as follows. In Section 2, we provide a compact problem formulation which is used in our upper bound procedure. In addition, we present the basic formulation which is used in the branch-and-cut algorithm. In Section 3, we provide our series of valid inequalities which are included in our branch-and-cut algorithm, and in Section 4, we discuss their corresponding separation algorithms. We continue in Section 5 with the outline of the branch-and-cut framework, consisting of the branch-and-cut algorithm and the upper bound procedure. In Section 6, we present the computational experiments with the branch-and-cut framework. We conclude and provide directions for further research in Section 7.

[1]When published: URL of public Github repository here

2 Problem formulation

We present two Mixed Integer Programming (MIP) formulations for the A-MDVRP. The first formulation is a compact formulation used in our upper bound procedure and consists of a *polynomial* number of variables and constraints. The second formulation, which we call the basic formulation, consists of exponentially many constraints. This formulation serves as starting point for our branch-and-cut algorithm and the derivation of our new valid inequalities.

2.1 Compact formulation

Let $x_{ij} \in \{0,1\}$ be the binary variable indicating whether or not arc $(i,j) \in A$ is traversed, and let $z_d \in \{0,1\}$ be the binary variable denoting whether or not depot $d \in D$ is opened. In addition, continuous variables y_{ij} (where $i,j \in V$) and u_i (where $i \in V$) are used to ensure that the capacity of a vehicle is not exceeded and to eliminate subtours, and the tours start and end at the same depot, respectively. We present the following compact MIP formulation (P_c) for the A-MDVRP.

$$\min \quad \sum_{(i,j)\in A} c_{ij}x_{ij} + \tilde{c}\sum_{d\in D} z_d \tag{1a}$$

$$\text{s.t.} \quad \sum_{j\in V'} x_{ij} = \sum_{j\in V'} x_{ji} = 1 \qquad \forall\, i \in V, \tag{1b}$$

$$\sum_{j\in V} x_{dj} = \sum_{j\in V} x_{jd} \leq m \qquad \forall\, d \in D, \tag{1c}$$

$$x_{di} \leq z_d \qquad \forall\, i \in V, d \in D, \tag{1d}$$

$$\sum_{i\in V'} (y_{ij} - y_{ji}) = q_j \qquad \forall j \in V, \tag{1e}$$

$$\sum_{d\in D}\sum_{j\in V} y_{dj} - \sum_{j\in V} q_j = 0, \tag{1f}$$

$$y_{ij} - (Q - q_i)x_{ij} \leq 0 \qquad \forall\, i \in V', j \in V, \tag{1g}$$

$$\sum_{d\in D} d(x_{di} + x_{id}) - u_i \leq 0 \qquad \forall\, i \in V, \tag{1h}$$

$$u_i - |D| - \sum_{d\in D}(d - |D|)(x_{di} + x_{id}) \leq 0 \qquad \forall\, i \in V, \tag{1i}$$

$$u_i - u_j - M(1 - x_{ij} - x_{ji}) \leq 0 \qquad \forall\, i,j \in V, i \neq j, \tag{1j}$$

$$y_{dj} + M(1 - x_{dj}) \geq \ell^{\min} \qquad \forall\, d \in D, j \in V, \tag{1k}$$

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

$$z_d \in \{0,1\} \qquad \forall\, d \in D, \tag{1m}$$

$$u_i \geq 0 \qquad \forall\, i \in V, \tag{1n}$$

$$0 \leq y_{ij} \leq Q \qquad \forall\, i,j \in V. \tag{1o}$$

Objective (1a) minimizes the sum of travel costs and depot opening costs. Constraints (1b) ensure that every customer is visited exactly once. Constraints (1c) limit the number of vehicles (or tours) at the depot, if necessary. Constraints (1d) link the z_d and x_{ij} variables. Constraints (1e)–(1g) ensure that loads do not exceed the vehicle capacity and exclude subtours [13]. Constraints (1h)–(1j) enforce that paths cannot start and end at different depots. Constraints (1k) ensure that the lower bound on the tour size is respected. Constraints (1l)–(1o) define the domain of the variables.

The above compact MIP formulation (P_c) covers all four problem variants introduced in Section 1. Table 1 provides for each of the four problem variants the appropriate parameter values to be used in this MIP formulation. An '$*$' indicates that the particular parameter has no model-specific restrictions.

Table 1: Parameter restrictions of the various problem variants.

Problem variant	Q	q_i	$\ell^{\min}$	$\ell^{\max}$	$\tilde{c}$	m
A-MDTSP	n	1	1	n	0	1
A-MDmTSP	$\ell^{\max}$	1	$*$	$*$	0	$*$
A-MDCVRP	$*$	$*$	1	n	0	$*$
A-CLRP	$*$	$*$	1	n	$*$	$*$

The compact MIP formulation could be used to solve any A-MDVRP with off-the-shelf MIP solvers such as Gurobi and CPLEX. However, we will employ it for our upper bound procedure, for which purpose we consider a restricted version, using a subset of the arcs. The subset consists of 1) all arcs between depot nodes and customer nodes, and 2) for every customer node i, the δ cheapest outgoing arcs connected to other customer nodes. The resulting formulation is called the δ-compact formulation, which we refer to as P_δ. The upper bound procedure is predominantly based on iteratively solving δ-compact formulations and runs parallel to the branch-and-cut algorithm to provide upper bounds in a fast manner. The details of the upper bound procedure are provided in Section 5.1.

2.2 Basic formulation

In the following, we present a basic formulation that we use as the starting point of our branch-and-cut algorithm. Consider the following formulation (P):

$$\min \quad \sum_{(i,j)\in A} c_{ij}x_{ij} + \tilde{c}\sum_{d\in D} z_d \tag{2a}$$

$$\text{s.t.} \quad \sum_{j\in V'} x_{ij} = \sum_{j\in V'} x_{ji} = 1 \qquad \forall\, i \in V, \tag{2b}$$

$$x_{di} \leq z_d \qquad \forall\, i \in V, d \in D, \tag{2c}$$

$$\text{Depot Fixing Constraints,} \tag{2d}$$

$$\text{Capacity / Tour Size Constraints,} \tag{2e}$$

$$x_{ij} \in \{0,1\}, z_d \in \{0,1\} \qquad \forall i,j \in V, d \in D. \tag{2f}$$

Except for (2d) and (2e), the objective and the constraints are equal to the compact formulation (P_c). Note that the above formulation consists of x_{ij} and z_d variables only, whereas the compact formulation also uses the continuous variables y_{ij} and u_i.

In the remainder of this study, we use the following shorthand notation. We define $x(T) := \sum_{(i,j)\in T} x_{ij}$ for $T \subseteq A$, and $x(S_1,S_2) = \sum_{i\in S_1,\, j\in S_2,\, (i,j)\in A} x_{ij}$ for $S_1, S_2 \subseteq V'$. In addition, we denote $x(\gamma(S)) := x(S,S)$, $x(\delta^+(S)) := x(S, V'\setminus S)$, and $x(\delta^-(S)) := x(V'\setminus S, S)$ for $S \subset V'$. Thus $\delta^+(S)$ and $\delta^-(S)$ denote all outgoing and incoming arcs of the node set S, respectively, and $x(\delta^+(S))$ and $x(\delta^-(S))$ denote their corresponding x_{ij} variables. For brevity, we write $\delta^+(i) := \delta^+(\{i\})$ and $\delta^-(i) := \delta^-(\{i\})$ for $i \in V'$.

Depot Fixing constraints (2d), which ensure that each tour starts and ends at the same depot, are present in all problem variants. The following Depot Fixing Constraints are commonly used (see, e.g., Belenguer et al. [6], Laporte et al. [14]), and are referred to as path elimination constraints:

$$\sum_{o\in I} x_{oi} + \sum_{o\in D\setminus I} x_{jo} + x(\gamma(S\cup\{i,j\})) \;\leq\; |S| + 2 \quad \text{for all } S \subset V,\, I \subset D,\, i,j \in V\setminus S. \tag{3}$$

However, these constraints do not exclude single customer tours starting and ending at different depots. In Section 3, we present our new class of Depot Fixing Constraints based on D_k inequalities that can replace the above set of constraints. As will be explained later, our proposed constraints do exclude single customer tours starting and ending at different depots.

Capacity Constraints (2e) are present in the A-MDVRP variants to eliminate subtours and are particularly useful when vehicle capacity restrictions are considered (i.e., for the A-MDCVRP and the A-CLRP). In general, they have the following form:

$$x(\gamma(S)) \leq |S| - k(S) \quad \text{for all } S \subset V, \tag{4}$$

where $k(S)$ is the number of vehicles required to serve all customers in S. The number of vehicles $k(S)$ is the solution of a bin-packing problem and is commonly replaced by $\lceil \sum_{i \in S} q_i / Q \rceil$ [18]. Refinements to these capacity constraints are proposed by Letchford and Salazar-González [15, 16]. We remark that the capacity constraints from the literature are mostly evaluated in the light of symmetric single depot vehicle routing problem instances. We, on the other hand, adapt these capacity constraints to asymmetric multi-depot settings in Section 3.2.3.

Finally, Tour Size Constraints are only required for the A-MDmTSP variant where they impose that each tour must include at least $\ell^{\min}$ and at most $\ell^{\max}$ customers. In general, the upper bound constraints can be imposed by using the above capacity constraints by setting $Q = \ell^{\max}$ and $q_i = 1$ for all $i \in V$. Tour Size Constraints have been studied for single depot symmetric vehicle routing problems (see, e.g., Bektaş et al. [5], Gouveia et al. [12]), but not yet for the asymmetric multi-depot settings such as ours. In Section 3.3, we present a class of D_k inequalities that can be used to enforce Tour Size Constraints in the asymmetric multi-depot setting.

3 Model constraints and valid inequalities

This section introduces the model constraints (Section 3.1) and the valid inequalities (Section 3.2) that are used in the branch-and-cut algorithm. Further refinements for the A-MDmTSP are discussed in Section 3.3. The corresponding separation algorithms are then discussed in Section 4.

In the remainder of this paper we refer to inequalities that exhibit model defining characteristics as *constraints* and to inequalities that strengthen the linear hull but do not exhibit problem defining characteristics as *valid inequalities*. For readability purposes, we only specify actual problem variants if the results presented below are not generally applicable to all variants, and we write "A-MDVRPs" if the results hold generally.

3.1 Model constraints

We continue with describing two families of constraints that are valid for A-MDVRPs. We first discuss a generalization of the D_k^+ and D_k^- constraints to the multi-depot case, which can eliminate subtours. Next, we extend those constraints to eliminate paths starting and ending at different depots, referred to as the D_k^+ depot and D_k^- depot constraints.

3.1.1 D_k^+ and D_k^- constraints

We first adapt D_k inequalities to the multi-depot setting and show that the resulting constraints eliminate subtours.

Theorem 1. *Let $\{i_1, i_2, \ldots, i_k\} \subset V$ be a sequence of distinct customer nodes. Then the following inequalities*

$$D_k^+ : \quad x_{i_1 i_k} + \sum_{h=2}^{k} x_{i_h i_{h-1}} + 2\sum_{h=2}^{k-1} x_{i_1 i_h} + \sum_{h=3}^{k-1} x(\{i_2, \ldots, i_{h-1}\}, i_h) \leq k-1, \tag{5}$$

$$D_k^- : \quad x_{i_k i_1} + \sum_{h=2}^{k} x_{i_{h-1} i_h} + 2\sum_{h=2}^{k-1} x_{i_h i_1} + \sum_{h=3}^{k-1} x(i_h, \{i_2, \ldots, i_{h-1}\}) \leq k-1, \tag{6}$$

are valid cuts for the integer polytope of A-MDVRPs. Moreover, subtour elimination constraints are correctly formulated by (5) *or* (6)*.*

Proof. First consider the D_k^+ inequality as given in (5). There are only three scenarios for which the left hand side summation may become strictly larger than $k-1$. For each scenario we show that the corresponding solution is not a feasible solution for the A-MDVRP. Thereby it directly follows that (5) is a valid cut for the integer polytope of the A-MDVRP. The same reasoning holds for the validity of (6).

Firstly, suppose $\sum_{h=2}^{k-1} x_{i_1 i_h} = 0$. Then k nodes $\{i_1, \ldots, i_k\}$ can have at most k traversed arcs in (5) only if a feasible solution contains a traversed cycle $\{i_k, i_{k-1}, \ldots, i_1\}$ by construction. However, a cycle of only customer nodes is infeasible.

Secondly, suppose that $x_{i_1 i_h} = 1$ for some $h \in \{2, \ldots, (k-1)\}$ and note that arc (i_1, i_h) has weight 2. There is exactly one incoming arc (i_1, i_k) and one outgoing arc (i_k, i_{k-1}) considered in (5) for i_k. Since $x_{i_1 i_h} = 1$, we have $x_{i_1 i_k} = 0$. If $x_{i_k i_{k-1}} = 0$, then node i_k is excluded in the traversed arcs of (5). Thus, a cycle of remaining $(k-1)$ nodes must be involved so that the left-hand side summation equals k. However, a cycle of customer nodes is infeasible.

Lastly, let us consider the case $x_{i_1 i_h} = 1$ for some $h \in \{2, \ldots, (k-1)\}$ and $x_{i_k i_{k-1}} = 1$. Since traversed cycles must be excluded as before, the only option is to have a traversed chain consisting of all nodes $\{i_1, \ldots, i_k\}$. As the only possible incoming arc for i_k is not traversed ($x_{i_1 i_k} = 0$), node i_k must be the start of the chain. Note that there is only one outgoing arc from nodes $\{i_{k-1}, \ldots, i_2\}$. Following the possible arcs in (5) to traverse, we obtain a contradiction to $x_{i_1 i_h} = 1$. Therefore, the left-hand side of (5) is less than or equal to $(k-1)$, i.e., (5) is a valid inequality for the integer polytope of A-MDVRPs.

It is left to show that the constraints correctly eliminate subtours. If there is a subtour $\{j_1, \ldots, j_k\}$, then consider sequence $\{j_k, \ldots, j_1\}$ in (5) and we obtain a contradiction. Thus, subtours are eliminated by (5). □ □

3.1.2 D_k^+ and D_k^- Depot Fixing Constraints

We further exploit the structure of (5) and (6) to derive the D_k^+ and D_k^- Depot Fixing Constraints that eliminate paths starting and ending at different depots, and can thus model the Depot Fixing constraints (2d) instead of (3).

Theorem 2. *Let $\mathcal{I} = \{i_1, i_2, \ldots, i_k\} \subseteq V$ be a sequence of distinct customer nodes and let $O \subset D$ be a subset of depots. The following inequalities*

$$D_k^+ \text{ depot}: \sum_{s \in O} x_{i_1 s} + \sum_{s \in D \setminus O} x_{s i_k} + \sum_{h=2}^{k} x_{i_h i_{h-1}} + 2\sum_{h=2}^{k} x_{i_1 i_h} + \sum_{h=3}^{k} x(\{i_2, \ldots, i_{h-1}\}, i_h) \leq k \tag{7}$$

$$D_k^- \text{ depot}: \sum_{s \in O} x_{s i_1} + \sum_{s \in D \setminus O} x_{i_k s} + \sum_{h=2}^{k} x_{i_{h-1} i_h} + 2\sum_{h=2}^{k} x_{i_h i_1} + \sum_{h=3}^{k} x(i_h, \{i_2, \ldots, i_{h-1}\}) \leq k, \tag{8}$$

are valid cuts for the integer polytope of A-MDVRPs. Moreover, depot-fixing constraints of A-MDVRPs are correctly formulated by (7) or (8).

Proof. In the following, we only prove the validity of (7), since the validity of (8) can be shown along the same line of reasoning.

Suppose $x_{i_1 i_h} = 0$ for all $h \in \{2, \ldots, k\}$. There is at most one incoming arc to O and at most one outgoing arc from $D \setminus O$ traversed in the left-hand side of (7) as there is a single depot assigned to nodes i_1 and i_k. In total, k customer nodes $\{i_1, \ldots, i_k\}$ can have at most $k+1$ traversed arcs in the left-hand side of (7) only if i_1 and i_k are connected to O and $D \setminus O$ respectively, and the chain $\{i_k, \dots, i_1\}$ is traversed due to the construction of the left-hand side of (7). However, this is infeasible as a tour must start and end at the same depot.

Next let $x_{i_1 i_h} = 1$ for some $h \in \{2, \dots, k\}$. Consider the following sequence $\{i_1, i_2, \dots, i_k, i_{k+1}\}$ where $x_{i_1 i_{k+1}} := \sum_{s \in O} x_{i_1 s}$ and $x_{i_{k+1} i_k} := \sum_{s \in D \setminus O} x_{s i_k}$ to arrive at an inequality as denoted in Theorem 1. By following the same argument as in the proof of Theorem 1 for the case $x_{i_1 i_h} = 1$ we obtain that (7) is a valid inequality for the integer polytope of A-MDVRPs.

Now let us show that path elimination constraints are correctly formulated by (7). By contradiction, assume that there is a tour $\{s_1, i_1, \dots, i_k, s_2\}$ with $s_1, s_2 \in D, s_1 \neq s_2$ and $i_1, \dots, i_k \in V$. Consider sequence $\{i_k, \dots, i_1\}$ and $O := s_2$ in (7) and we obtain a contradiction. Thus, there is no tour that starts and ends at different depots. □ □

Note that when $k = 1$ in Theorem 2, i.e., $\mathcal{I} = \{i_1\}$, we have

$$\sum_{s \in O} x_{s i_1} + \sum_{s \in D \setminus O} x_{i_1 s} \leq 1$$

which is a depot fixing constraint involving a single node $i_1 \in V$. Note that this is different from the classical path-eliminating constraints (3) as these are typically used when tours of a single size are forbidden. As it can be seen from Theorems 1 and 2, the modeling capabilities of those D_k type of constraints are rather extensive. As a direct consequence, we obtain the following corollary.

Corollary 2.1. *A-MDTSPs are correctly formulated as* min (2a) *subject to* (1c), (2b), (2c), *and completed with:*

(i) at least one of (5) *and* (6) *to eliminate subtours,*

(ii) at least one of (7) *and* (8) *for depot fixing constraints.*

3.2 Valid inequalities

In the following, we present several strengthening valid inequalities. We first discuss CAT-type inequalities, introduced by Balas [2] for the ATSP, and generalize these to the multi-depot case. Thereafter, we provide several comb-type inequalities and capacity inequalities, both generalized and adapted from the CVRP case.

3.2.1 CAT-type inequalities

We modify the Closed Alternating Trail (CAT) inequalities derived by Balas [2], such that they are valid for the A-MDVRP. Before discussing the CAT inequalities, we need to introduce the notion of incompatible arcs. Notice that this is different from Balas [2]. An arc $(i, j) \in A$ is incompatible in the following circumstances:

1. If $i, j \in V$, then (i, j) is incompatible with (i, u), (u, j), and (j, i), for any $u \in V'$.
2. If $i \in D$ and $j \in V$, then (i, j) is incompatible with: (i) (u, j) for any $u \in V'$, and (ii) (j, o) for any $o \in D \setminus \{i\}$.
3. If $i \in V$ and $j \in D$, then (i, j) is incompatible with: (i) (i, u) for any $u \in V'$, and (ii) (o, i) for any $o \in D \setminus \{j\}$.

A CAT inequality is described by a sequence of arcs $T = \{a_1, \ldots, a_t\}$ such that $a_1, \ldots, a_t \in A$, $t \geq 5$ and odd, and any arc in T is incompatible with its neighbor arcs in T and compatible with all other arcs in T. The neighbor arcs of a_1 are considered to be a_2 and a_t. In other words, we consider that arcs $\{a_1, \ldots, a_t\}$ are placed on a circle maintaining their order.

Furthermore, node i is called a source if $\delta^+(i) \cap T = 2$, and a sink if $\delta^-(i) \cap T = 2$. In addition, an arc $(i, j) \in A \setminus T$ is called a chord of type 1 if $i \in V$ is a source and $j \in V$ is a sink. We let R be the collection of all chords of type 1 as induced by the set T and the graph G.

Theorem 3. *Let $T = \{a_1, \ldots, a_t\} \subset A$ with an odd $t \geq 5$ such that any arc in T is incompatible with its neighbor arcs and compatible with the other arcs in T. Let R be the set of chords of type 1. Then, the CAT inequality*

$$x(T \cup R) \leq \frac{t-1}{2} \tag{9}$$

is valid for the integer polytope of A-MDVRPs.

Proof. As $T \cap R = \emptyset$, we have $x(T \cup R) = x(T) + x(R)$. By the definition of "incompatibility", any two neighboring arcs are mutually exclusive. In other words, the corresponding values of x of any two consecutive arcs cannot be 1 at the same time. Since t is odd, we obtain

$$x(T) \leq \frac{t-1}{2}. \tag{10}$$

Next let us examine set R. Consider $(i,j) \in R$ then we have $\delta^+(i) \cap T = 2$ and $\delta^-(j) \cap T = 2$ by definition. Thus, there are incompatible neighbor arcs (i,u) and (i,v) in T for some $u,v \in V'$. Arcs (i,u) and (i,v) have to be a neighbor, as all non-neighbor arcs are compatible by the construction of CAT. The same argument holds for arcs $(u',j) \in T$ and $(v',j) \in T$ for some $u',v' \in V'$.

By contradiction, suppose that $x_{ij} = 1$ and the equality holds for (10). If $x_{ij} = 1$, then we must have $x_{iu} = x_{iv} = x_{u'j} = x_{v'j} = 0$. Since $(i,j) \in R \subset A \setminus T$, these two neighbors do not overlap. So the corresponding x value of set T must be of form $\{0, 0, \ldots, 0, 0, \ldots\}$. On the other hand, using $x(T) = (t-1)/2$, there cannot be three consecutive zeros. Thus, the arcs in T must have value $\{0, 0, 1, 0, 1, 0, 1, \ldots, 0, 1\}$, which is a contradiction to the pattern $\{0, 0, \ldots, 0, 0, \ldots\}$. Therefore, when $x(T) = (t-1)/2$, we have $x(R) = 0$ and thus (9) holds.

Suppose now $x(R) = q$ with $q > 0$. As before, whenever $x_{ij} = 1$ for $(i,j) \in R$, we have $x_{iu} = x_{iv} = x_{u'j} = x_{v'j} = 0$. Excluding all these zero valued arcs, there are $t - 4q$ remaining arcs of T which are separated into at most $2q$ parts. As $t - 4q$ is odd, there are at most $(2q-1)$ parts with an odd number of arcs and at least one part with an even number of arcs. The value of a part with odd $\tilde{t}$ arcs is at most $(\tilde{t}+1)/2$ and a part with even $\widehat{t}$ arcs is at most $(\widehat{t}/2)$. By summing all these separated parts, we obtain

$$x(T) + x(R) \leq \frac{(t-4q) + (2q-1)}{2} + q = \frac{t-1}{2},$$

and we have shown that (9) holds for A-MDVRP. □ □

3.2.2 Comb type inequalities

Asymmetric comb inequalities for the ATSP are defined in [9] on a complete digraph (V, A), where V is a node set and A is the arc set with $n(n-1)$ arcs. Define a set handle $H \in V$ and an odd number of subsets called teeth $H_i \subset V$ with $i = 1, \ldots, t$, and $t \geq 3$ such that

$$H \cap T_i \neq \emptyset, \quad T_i \setminus H \neq \emptyset \quad \text{and} \quad T_i \cap T_j = \emptyset \quad \text{for all} \quad i,j = 1, \ldots, t. \tag{11}$$

Then it is shown by Fischetti [9] that the comb inequality

$$x(\delta^+(H)) + \sum_{j=1}^{t} x(\delta^+(T_j)) \geq \frac{3t+1}{2} \tag{12}$$

describes a facet of the ATSP polytope on a directed graph for $t \geq 7$.

The comb inequalities have been generalized to the multi-depot multiple traveling salesman problem with symmetric costs in [7] and are shown to be facets for $t \geq 3$. Similarly, we can reformulate the ATSP combs (12) such that they are valid for the A-MDVRP by imposing the additional constraint that all the depot nodes should be in the same element of the comb.

Proposition 1. *Let $G_d \subseteq G$ be a graph with only a single depot node $d \in D$. In graph G_d we can derive the comb inequalities* (12)*. Consider such an inequality and add all remaining depot nodes $d' \in D \setminus \{d\}$ to all the sets (i.e., handle and teeth) that contain d. This modified inequality* (12) *with $t \geq 3$ is valid for the A-MDVRP and is referred to as an ATSP-comb.*

Proof. Every edge in the formulation of [7] is defined by x_{ij} and x_{ji} with $x_{ij} = x_{ji}$. In other words, incoming and outgoing arcs are both included in their formulation if the corresponding asymmetric case is considered. Reformulating the H-comb inequality of [7] for the asymmetric case, we derive

$$x(\delta^+(H)) + x(\delta^-(H)) + \sum_{j=1}^{t} x(\delta^+(T_j)) + \sum_{j=1}^{t} x(\delta^-(T_j)) \geq 3t + 1.$$

As the number of incoming arcs and the outgoing arcs are the same, we directly obtain (12). □ □

In general, the symmetric case is translated to the asymmetric case by simply replacing every edge (symmetric case) by their two corresponding arcs (asymmetric case). However, the above argument does not work for their result as the edge is defined by both variables x_{ij} and x_{ji} with $x_{ij} = x_{ji}$ for any $i, j \in V$. For completeness, using the same argument as above, we reformulate the T-, and H-comb inequalities proposed in Benavent and Martínez [7] such that they are valid for the A-MDVRP.

Proposition 2. *Let the handle $H \in V \cup D$ and an odd number of teeth $T_i \subset V$ with $i = 1, \ldots, t$ and $t \geq 3$ be defined such that* (11) *is satisfied, and that $H \cap D \neq \emptyset$ and $D \backslash H \neq \emptyset$. The following H-comb inequalities are valid for the A-MDVRP.*

$$x(\delta^+(H)) + \sum_{j=1}^{t} x(\delta^+(T_j)) \geq \frac{3t+1}{2}. \tag{13}$$

Proposition 3. *Let the handle $H \in V \cup D$ and the teeth $T_i \subset V \cup I$ with $i = 1, \ldots, t$ and $t \geq 1$ be defined such that* (11) *is satisfied. In addition, assume that (i) $T_i \cap I \neq \emptyset$ for $i = 1, \ldots, t$, (ii) $H \backslash \cup_{i=1}^{t} T_i \neq \emptyset$, and (iii) $I \backslash \cup_{i=1}^{t} T_i \neq \emptyset$. Then the following T-comb inequalities are valid for the A-MDVRP*

$$x(\delta^+(H)) + \sum_{j=1}^{t} x(\delta^+(T_j)) \geq 2t + 2. \tag{14}$$

3.2.3 Capacity inequalities

Capacity inequalities such as the generalized large multistar inequalities and the knapsack large multistar inequalities are defined for symmetric single depot capacitated vehicle routing problem. However, the inequalities only involve customer nodes, which allows us to apply the transformation from edge variables (for the symmetric case) to arc variables (for the asymmetric case) in the following way: we substitute every edge variable x_e with the arc variables $x_{ij} + x_{ji}$, where i and j are the endpoints of e. Thus, the valid inequalities for single depot capacitated vehicle routing problems from Letchford and Salazar-González [16] are directly applicable for multi-depot settings.

In our branch-and-cut algorithm, we use framed capacity inequalities, homogeneous multistar inequalities, and large multistar inequalities. To keep our exposition concise, we restate the inequalities and we refer to Letchford et al. [17] and Lysgaard et al. [18] for detailed information.

Homogeneous multi-star inequalities are represented by nucleus $S \subset V$, satellites $T \subset V \backslash S$, connectors $C \subset S$, and three integers A, B, L, and are given by

$$Bx(\delta(S)) - Ax(C, T) \geq L. \tag{15}$$

Generalized large multi-star inequalities are represented by a nucleus $S \subset V$ and the remaining node set $\bar{S} := V \backslash N$, and are given by

$$Qx(N, N) + \sum_{j \in \bar{S}} q_j x(N, \{j\}) \leq Q|S| - \sum_{i \in S} q_i. \tag{16}$$

Framed capacity inequalities are defined by a frame $S \subseteq V$ and a partition $\Omega = \{S_1, \ldots, S_p\}$ of S, and are given by

$$x(\delta(S)) + \sum_{i=1}^{p} x(\delta(S_i)) \geq 2k(S, \Omega) + 2 \sum_{i=1}^{p} \Big\lceil \sum_{i \in S_i} q_i / Q \Big\rceil. \tag{17}$$

3.3 Specialized model constraints for the A-MDmTSP

In this section, we study tour size constraints and strengthened path elimination constraints for the A-MDmTSP.

3.3.1 Tour size constraints

As previously mentioned, we can model the upper limit $\ell^{\max}$ on the tour size (i.e., the number of customers in a tour) by setting the customer demands equal to 1 and the vehicle capacity equal to $\ell^{\max}$. Furthermore, we propose the following simple constraints to impose the lower bound restriction for multi-depot routing problems.

$$x(\delta^+(S)) - \sum_{d\in D}\sum_{s\in S}(x_{ds} + x_{sd}) \geq 0 \quad \text{for all } S \subset V,\ |S| \leq \ell^{\min}. \tag{18}$$

On the other hand, D_k-type inequalities can also model the tour size lower and upper limits. Consider a sequence $\{i_1, i_2, \ldots, i_k, i_{k+1}\}$ in Theorem 1. By construction of the left-hand side of (5), observe that the last node i_{k+1} can be either a depot or a customer node in a multi-depot setting, since there is exactly one incoming and one outgoing arc considered in (5) for i_{k+1}. Thus, we may replace i_{k+1} as the set of depots in the multi-depot setting.

Theorem 4. *Let $\{i_1, i_2, \ldots, i_k\} \subseteq V$ be a sequence of distinct customer nodes. Then the following inequalities*

$$D_k^+\text{-}lim: \quad \sum_{s\in D} x_{i_1 s} + \sum_{s\in D} x_{s i_k} + \sum_{h=2}^{k} x_{i_h i_{h-1}} + 2\sum_{h=2}^{k} x_{i_1 i_h} + \sum_{h=3}^{k} x(\{i_2, \ldots, i_{h-1}\}, i_h) \leq k \tag{19}$$

$$D_k^-\text{-}lim: \quad \sum_{s\in D} x_{s i_1} + \sum_{s\in D} x_{i_k s} + \sum_{h=2}^{k} x_{i_{h-1} i_h} + 2\sum_{h=2}^{k} x_{i_h i_1} + \sum_{h=3}^{k} x(i_h, \{i_2, \ldots, i_{h-1}\}) \leq k \tag{20}$$

are valid cuts for the integer polytope of the A-MDmTSP for $k < \ell^{\min}$ or $k > \ell^{\max}$. Moreover, the constraints enforcing the lower and upper bounds on the tour size of the A-MDmTSP are correctly formulated by (19) *or* (20).

Proof. We prove that the proposed cuts are valid for the integer polytope of the A-MDmTSP by showing that the left-hand side of (19) and (20) cannot exceed k for feasible solutions.

First suppose $x_{i_1 i_h} = 0$ for all $h \in \{2, \ldots, k\}$. As mentioned before, at most one incoming arc from depots and at most one outgoing arc to depots are possible to be traversed in (19). In total, k customer nodes $\{i_1, \ldots, i_k\}$ can have at most $k + 1$ traversed arcs in the left-hand side of (19) only if i_1 and i_k are connected to depots due to the construction of the left-hand side of (7). If i_1 and i_k are connected to different depots, then the tour is infeasible since tours must start and end at the same depot. Furthermore, if i_1 and i_k are connected to the same depot, the tour is still infeasible because this implies that the tour has size k with $k < \ell^{\min}$ or $k > \ell^{\max}$.

The remaining case where $x_{i_1 i_h} = 1$ for some $h \in \{2, \ldots, k\}$ is shown as in the proof of Theorem 1 by considering i_{k+1} as the depot set D. We conclude that (19) is a valid inequality for the integer polytope of the A-MDmTSP.

Now let us show that the tour size constraints are correctly formulated by (19). By contradiction, assume that there is a tour $\{s, i_1, \ldots, i_k, s\}$ with $s \in D$ and $i_1, \ldots, i_k \in V$ such that $k < \ell^{\min}$ or $k > \ell^{\max}$. Consider sequence $\{i_k, \ldots, i_1\}$ in (19) and we obtain a contradiction. Thus, there is no tour with size $k < \ell^{\min}$ or $k > \ell^{\max}$. □ □

Based on (19) and (20), we derive a novel formulation for A-MDmTSP:

Corollary 4.1. *The A-MDmTSP is correctly formulated as* min (2a) *subject to* (2b), (2c), *and completed with:*

(i) at least one of (5) *and* (6) *to eliminate subtours*

(ii) at least one of (7) *and* (8) *for depot fixing constraints,*

(iii) at least one of (19) *and* (20) *for tour size constraints.*

Observe that the path elimination constraints (7) and (8) are redundant for the cases $k < \ell^{\min}$ and $k > \ell^{\max}$ if (19) and (20) are present, respectively. Thus, for example, when (19) and (7) are used together, it is sufficient to consider (7) only for the case $\ell^{\min} \leq k \leq \ell^{\max}$.

3.3.2 Path elimination constraints

Without loss of generality, we assume that $\ell^{\min} \geq 2$ for the A-MDmTSP. Under this assumption, we can further refine path elimination constraints (3). This is summarized in the following proposition.

Proposition 4. *Consider the A-MDmTSP with* $\ell^{\min} \geq 2$. *Then path elimination constraints* (3) *can be strengthened as follows*

$$\sum_{o \in I} x_{io} + \sum_{s \in D \setminus I} x_{sj} + \sum_{o \in I} x_{oi} + \sum_{s \in D \setminus I} x_{js} + x(\gamma(S \cup \{i,j\})) \leq |S| + 2$$

$$\text{for all } I \subset D, i, j \in V, S \subset V \setminus \{i,j\}. \quad (21)$$

In other words, a feasible solution of the A-MDmTSP satisfies the above inequality.

Proof. Note that $\sum_{o \in I} x_{io} + \sum_{o \in I} x_{oi} \leq 1$ and $\sum_{s \in D \setminus I} x_{sj} + \sum_{s \in D \setminus I} x_{js} \leq 1$ hold since $\ell^{\min} \geq 2$. Moreover, $x(\gamma(S \cup \{i,j\})) \leq |S| + 1$ is a valid subtour elimination constraint. The equalities are attained for the above three inequalities only if the feasible solution contains a tour consisting of all nodes of $S \cup \{i, j\}$ and i, j are connected to different depots, which is infeasible. Thus, by combining the above three inequalities, we obtain (21). □ □

Note that the above path elimination constraints do not remove tours with a single customer connected to different depots. Thus, we need to add the following inequality

$$\sum_{o \in I} x_{oh} + \sum_{o \in D \setminus I} x_{ho} \leq 1 \text{ for all } h \in V,\ I \subset D. \tag{22}$$

Now let us formulate the A-MDmTSP without using any D_k-type inequalities by including the above constraints.

Corollary 4.2. *The A-MDmTSP is correctly formulated as* min (2a) *subject to* (2b), (2c) *and completed with*

(i) tour size constraints (4) *and* (18) *(subtours are also eliminated),*

(ii) path elimination constraints (21) *and* (22).

4 Separation algorithms

Efficient separation procedures for the model constraints and valid inequalities introduced in Section 3 are crucial for the computational performance of the overall branch-and-cut framework. Failing to find violated inequalities in a quick manner results in increased run-times. In the following, we discuss the heuristic and exact procedures used for separating the valid inequalities.

Let x^* be the LP solution for which the inequalities are required to be separated. We call x^*_a the capacity/weight of arc $a \in A$. Define $\mathcal{C}_\mathrm{s}(x^*)$ as the set of *strongly* connected components of customer nodes in x^*. For each strongly connected component $C \in \mathcal{C}_\mathrm{s}(x^*)$, there can be sent a positive flow between all nodes $i, j \in C$. The set of *weakly* connected components $\mathcal{C}_\mathrm{w}(x^*)$ of customer nodes equals the set of strongly connected components on a modified graph $G' = (V, A')$, where each arc $(i, j) \in A'$ has capacity $x^*_{ij} := x^*_{ij} + x^*_{ji}$. For convenience, we refer to $\mathcal{C}_\mathrm{s}(x^*)$ and $\mathcal{C}_\mathrm{w}(x^*)$ by $\mathcal{C}_\mathrm{s}$ and $\mathcal{C}_\mathrm{w}$, respectively. Note that finding connected components is done by standard depth-first search methods.

Before discussing our new separation algorithms, we briefly summarize the separation algorithms described in the literature that are exploited in our branch-and-cut algorithm. The seperation algorightms for the CAT inequalities (9) and the D_k^+ and D_k^- inequalities (5) and (6) are adapted versions of the separation procedures described by Fischetti and Toth [10]. For both the CAT inequalities and the D_k^+ and D_k^- inequalities, we first identify the set of weakly connected components $\mathcal{C}_{\mathrm{w}}$. Then, separation is done for each component separately. For the CAT inequalities, we have to impose our definition of arc incompatibility (see Section 3.2.1) in order to remain valid for A-MDVRPs. Regarding the D_k^+ and D_k^- inequalities, we refrain ourselves from using the heuristic procedure described by Fischetti and Toth [10], and instead use the exact separation procedure for separating these inequalities as its run-time is negligible compared to the overall run-time of the branch-and-cut algorithm.

The separation of D_k^+ and D_k^- tour size constraints for the A-MDmTSP is done by the same procedure as for the regular D_k^+ and D_k^- inequalities. However, we simply prune a partial tour as soon as the depth-first search method considers tour larger or equal to $\ell^{\min}$, and we automatically have a separation algorithm for the D_k^+ and D_k^- tour size constraints. Note that $\ell^{\max}$ is enforced by setting the vehicle capacity equal to $\ell^{\max}$ and all customer demands q_i $(i \in V)$ equal to 1.

Moreover, we separate the subtour elimination constraints (4), the large- and homogeneous multistar inequalities, and the framed capacity inequalities (see Section 3.2.3) with the help of the procedures given by Lysgaard et al. [18]. However, these separation algorithms are tailored for capacitated routing problems with a single depot and a symmetric edge set. In order to still be able to use those separation algorithms, we modify the arc capacities to $\hat{x}_{ij}^* := x_{ij}^* + x_{ji}^*$. We ensure that $x_{ij} + x_{ji} < 1$ for all $i, j \in V$ by means of explicitly including all the subtour elimination constraints of size 2.

4.1 D_k^+ and D_k^- depot-fixing constraints

We now introduce the separation algorithm for the D_k depot constraints (7) as described by Theorem 2. We only discuss the separation of the D_k^+ depot constraint since the separation algorithm for the D_k^- depot constraint (8) can be obtained by swapping the indices. The separation algorithm is based on a depth-first search with sophisticated pruning rules. This may sound similar to the separation of D_k^+ and D_k^- inequalities as described by Fischetti and Toth [10], however, the separation algorithm differs structurally as will be made clear in the following.

Let $\mathcal{I} = \{i_1, i_2, \dots, i_k\} \subseteq V$ be a sequence of distinct customer nodes. To be valid for the integer polytope of the A-MDVRP, it is sufficient to only consider sequences $\mathcal{I}$ for which $\sum_{s \in D} x_{i_1,s}^* +$

$\sum_{s \in D} x^*_{s,i_k} > 1$. Furthermore, we only consider sequences $\mathcal{I}$ for which each customer is an element of the same connected component.

Observe that partitioning the set of depots into O and $O_2 := D \setminus O$ only affects the first two terms of (7) and is independent of the other terms. For a given LP solution x^*, we find the partitioning corresponding to the highest violation by greedily assigning depots into a set such that it contributes the most to the violation, that is, we assign a depot $s \in D$ to O if $x^*_{i_1,s} \geq x^*_{s,i_k}$ and to O_2 otherwise. For the optimal partitioning we have

$$\sum_{s \in O} x_{i_1,s} + \sum_{s \in D \setminus O} x_{s,i_k} = \sum_{s \in D} \max\left\{x^*_{i_1,s},\ x^*_{s,i_k}\right\}.$$

To find the sequence $\mathcal{I}$ with the largest violation $\phi(\mathcal{I})$, we use a depth-first search approach similar as for the D_k inequalities without depots. For each weakly connected component $C \in \mathcal{C}_{\mathrm{w}}$, we consider all pairs of customers nodes $i, j \in C$ for which $\sum_{s \in D} x^*_{i_1,s} + \sum_{s \in D} x^*_{s,i_k} > 1$. For any pair, we initialize the sequence $\mathcal{I} = \{i, j\}$ and iteratively explore all options under the restriction that node i remains first and node j remains last in the sequence. At each iteration, we calculate the degree of violation and a valid inequality is returned when a positive violation is observed.

A pruning rule is used to avoid a complete enumeration of all sequences. Let $\mathcal{B} = \{b_1, \ldots, b_\eta\} \subseteq V \setminus \mathcal{I}$ be the sequence of η customer nodes that is inserted into $\mathcal{I}$ such that we obtain $\mathcal{I}^+ = \{i_1, \ldots, i_{k-1}, b_1, \ldots, b_\eta, i_k\}$. We let $\Delta(\mathcal{I} \mid \mathcal{B})$ denote the violation increase when $\mathcal{B}$ is inserted into $\mathcal{I}$. By explicitly deriving $\Delta(\mathcal{I} \mid \mathcal{B}) = \phi(\mathcal{I}^+) - \phi(\mathcal{I})$ we get

$$\begin{aligned}\Delta(\mathcal{I} \mid \mathcal{B}) = {} & x^*_{i_k,b_\eta} + x^*_{b_1,i_{k-1}} - x^*_{i_k,i_{k-1}} - \eta \\ & + \sum_{j=1}^{\eta} x^*_{i_1,b_j} + \sum_{h=1}^{k-1}\sum_{j=1}^{\eta} x^*_{i_h,b_j} + \sum_{h=1}^{\eta-1}\sum_{j=h+1}^{\eta} x^*_{b_h,b_j} \\ & + \sum_{h=2}^{\eta} x^*_{b_h,b_{h-1}} + \sum_{h=1}^{\eta} x^*_{b_h,i_k}.\end{aligned} \tag{23}$$

Observe that the total weight of the incoming arcs into $\mathcal{B}$ is at most $x^*(V', \mathcal{B}) = \eta$ since $|\mathcal{B}| = \eta$. Furthermore, $x^*(V', \mathcal{B}) = x^*(D, \mathcal{B}) + x^*(V, \mathcal{B})$ and thus $x^*(V, \mathcal{B}) = \eta - x^*(D, \mathcal{B})$. It follows that $x^*(V, \mathcal{B}) \leq \eta - \sum_{s \in D} x^*_{s,b_1}$. Consequently,

$$x^*_{i_k,b_\eta} + \sum_{h=1}^{k-1}\sum_{j=1}^{\eta} x^*_{i_h,b_j} + \sum_{h=1}^{\eta-1}\sum_{j=h+1}^{\eta} x^*_{b_h,b_j} + \sum_{h=2}^{\eta} x^*_{b_h,b_{h-1}} \leq x^*(V, \mathcal{B}) \leq \eta - \sum_{s \in D} x^*_{s,b_1}. \tag{24}$$

Substituting (24) into (23) gives

$$\Delta(\mathcal{I} \mid \mathcal{B}) \leq x^*_{b_1,i_{k-1}} - \sum_{s \in D} x^*_{s,b_1} - x^*_{i_k,i_{k-1}} + \sum_{j=1}^{\eta} x^*_{i_1,b_j} + \sum_{h=1}^{\eta} x^*_{b_h,i_k}. \tag{25}$$

The third term of the right-hand side of (25) is a known constant for any sequence $\mathcal{I}$. Furthermore, the fourth term (i.e., the sum of weights from node i_1 towards a node of the new inserted sequence $\mathcal{B}$) is at most $1-\alpha$ where α is the current outgoing weight from node i_1. The fifth term (i.e., the sum of weights from all nodes of $\mathcal{B}$ towards node i_k) is at most $1-\beta$ where β is the current incoming weight into node i_k. For a given sequence $\mathcal{I}$ we have,

$$\alpha(\mathcal{I}) = \sum_{s\in D} x^*_{i_1,s} + \sum_{h=2}^{k} x^*_{i_1,i_h},$$

$$\beta(\mathcal{I}) = \sum_{s\in D} x^*_{s,i_k} + \sum_{h=1}^{k-1} x^*_{i_h,i_k}.$$

Substituting the upper bounds $1-\alpha(\mathcal{I})$ and $1-\beta(\mathcal{I})$ into (25) gives

$$\Delta(\mathcal{I} \mid \mathcal{B}) \leq x^*_{b_1,i_{k-1}} - \sum_{s\in D} x^*_{s,b_1} - x^*_{i_k,i_{k-1}} + 2 - \alpha(\mathcal{I}) - \beta(\mathcal{I}). \tag{26}$$

Upper bound (26) only depends on the known sequence $\mathcal{I}$ and on the first node of the sequence $\mathcal{B}$. Hence, the upper bound can be exploited during the depth-first search to prune parts of the enumeration tree.

For a particular $\mathcal{I}$, let $\phi(\mathcal{I})$ be the current violation of (7). Moreover, let $\phi_{\max}$ be the largest violation of (7) found so far during the depth-first search. We know that if $\phi(\mathcal{I})+\Delta(\mathcal{I} \mid \mathcal{B}) \leq \phi_{\max}$, the customer sequence $\mathcal{I}$ and all its further extensions in the enumeration tree can be pruned. Using the upper bound (26) on $\Delta(\mathcal{I} \mid \mathcal{B})$, we obtain the following pruning rule,

$$x^*_{b_1,i_{k-1}} - \sum_{s\in D} x^*_{s,b_1} \leq \phi_{\max} - \phi(\mathcal{I}) + \alpha(\mathcal{I}) + \beta(\mathcal{I}) + x^*_{i_k,i_{k-1}} - 2. \tag{27}$$

This rule is easily incorporated in the depth-first search by maintaining the values of $\alpha(\mathcal{I})$, $\beta(\mathcal{I})$, and $\phi(\mathcal{I})$ in the following way:

$$\begin{aligned}
\alpha(\mathcal{I}^+) &= \alpha(\mathcal{I}) + x^*_{i_1,b_1},\\
\beta(\mathcal{I}^+) &= \beta(\mathcal{I}) + x^*_{b_1,i_k},\\
\phi(\mathcal{I}^+) &= \phi(\mathcal{I}) + x^*_{i_k,b_1} + x^*_{b_1,i_{k-1}} - x^*_{i_k,i_{k-1}} + x^*_{i_1,b_1}\\
&\qquad + x^*(\{i_1,\ldots,i_{k-1}\},b_1) + x^*_{b_1,i_k} - 1.
\end{aligned}$$

4.2 Separating path-elimination constraints

Separating the depot fixing constraints (3) and (21) has been done for symmetric edge sets by Belenguer et al. [6]. However, for asymmetric arc sets some structurally different steps need to be taken. For completeness, we fully describe our separation procedure.

We first discuss the separation of inequalities (3), and then we discuss the changes required for separating inequalities (21). Recall that a violated inequality (3) is determined by two nodes $i, j \in V$, a subset $S \subset V \backslash \{i, j\}$, and a subset $I \subset D$.

For each $C \in \mathcal{C}_w$, we consider all pairs of customer nodes $i, j \in C$. Given i and j, we can determine $I \subseteq D$ such that $\sum_{o \in I} x^*_{oi} + \sum_{s \in D \setminus I} x^*_{js}$ is maximized. Two things are noticed: (i) finding I is independent of finding $S \subset V$ such that $x(\gamma(S \cup \{i, j\}))$ is maximized, and (ii) if that particular sum is smaller than one, there will be no violated inequality (3) since we assume that the subtour elimination constraints are already satisfied.

After having determined candidate nodes i and j, we need to find a subset $S \subset V$ in order to maximize $x(\gamma(S \cup \{i, j\}))$. We consider a modified graph $\tilde{G} = (\tilde{N}, \tilde{A})$, with $\tilde{N} = D \cup C_w \cup \{s^+\} \cup \{s^-\}$, where s^+ is an artificial source node and s^- an artificial sink node. Let $\tilde{A} := \{(i, j) : i, j \in \tilde{N}\}$. Let c_{uv} be the arc capacity of any $(u, v) \in \tilde{A}$, where $c_{uv} := x^*_{uv} + x^*_{vu}$. Moreover, we create arcs from s^+ to both i and j, and we create arcs from each $o \in D$ to s^+, all with a large enough capacity making them not being selected in a minimum cut.

We then determine the minimum weighted (s^+, s^-)-cut in graph $\tilde{G}$. All the nodes on the s^+-side of the minimum weighted cut will be part of the set S. If the found I, S, i, and j lead to a violated inequality (3), we add it to the model.

Summarizing, this method is exact and finds for every i and j a depot-fixing inequality with the largest violation in the form of inequality (3). The method can be easily adapted to separate constraints (21), since only calculating the sums corresponding to the connection of i and j to the depots is different. This remains independent from the remaining separation procedure, and hence, no structural changes are required except calculating the depot sums differently.

4.3 Comb inequalities

Separation of ATSP comb inequalities (11) is done by means of the procedure of Lysgaard et al. [18] for finding strengthened comb inequalities. The procedure returns a handle $H \subset V'$ and a set of teeth $T_1, \dots, T_\pi \subset V'$. To check the returned inequalities for validity for the A-MDVRP, we verify whether $T_\ell \cup T_k = \emptyset$ for all $\ell, k \in \{1, \dots, \pi\}$ with $\ell \neq k$. As is described in Proposition 1, we check which element contains the depot, and insert all other depots in the same element. If (11) is violated, we add the valid inequality to the model.

In order to separate T-comb inequalities, we consider two different separation algorithms. First, we focus on T-comb inequalities with a single tooth T, and refer to those inequalities as $T1$-comb inequalities. The second separation algorithm finds general T-comb inequalities (with multiple teeth).

Both separation procedures are based on a greedy search for finding violated $T(1)$-comb inequalities, similar as in Benavent and Martínez [7] but with some minor adaptations. For completeness, we fully describe our heuristic separation procedure.

We first get all weakly connected components $\mathcal{C}_{\mathrm{w}}$ and in each of those connected components we perform the following greedy search. For the $T1$-comb inequalities, we search a tooth T that contains a single depot, and we iteratively add customers so that $\delta^+(x(T))$ is as small as possible. We stop when adding a customer results in $x(\delta^+(T)) < 2$. We initialize the handle $H = V \cap T$, and iteratively add customers so that $x(\delta^+(H))$ is as small as possible. Then, if $x(\delta^+(T))+x(\delta^+(T)) < 4$, a violated $T1$-comb inequality is found. Notice that we repeat this greedy search for each depot in each weakly connected component.

T-comb inequalities are found by a similar procedure, except that we need to find multiple teeth instead of a single tooth. For each connected component, we create for each depot a tooth as for the $T1$-comb inequalities. This results in a collection of teeth of size d, the number of depots.

Then the following procedure is repeated n_1 times. We take an arbitrary subset of teeth, and check whether the teeth are not pairwise disjoint, and check whether the teeth spans the complete connected component. In either case, no violated T-comb inequality can be found and we continue with a new random selection of teeth.

What remains is to search for handle H. We initialize H with $\mathcal{C}_{\mathrm{w}} \cap V$ and iteratively remove customers from H while checking for a violated T-comb inequality with the current set H and the randomly selected teeth. If no customers remain in H, the search is terminated and we start over with a new random selection of teeth.

5 Branch-and-cut algorithm

Our branch-and-cut framework for solving A-MDVRPs consists of two simultaneously running algorithms. First, we have the branch-and-cut algorithm including the valid inequalities and accompanying separation algorithms that are discussed throughout Sections 3 and 4. Second, we have designed a simple yet effective upper bound procedure based on the δ-compact formulation (P_δ).

We note that the following chosen parameter values and the order of calling valid inequalities are the result of an extensive preliminary computational campaign. These values are kept fixed for all the different problem variants solved with the branch-and-cut framework. One might argue that different parameter values for each problem variant would perform better, however, we chose not to do so since using parameter values that work well on all problem types enhances the general applicability of the branch-and-cut framework.

5.1 A novel and easy to implement upper bound procedure

The upper bound procedure is based on iteratively solving the δ-compact formulation (P_δ) as introduced in Section 2.1. Recall that this formulation is obtained from (P_c) by removing the $(n - \delta)$ most expensive outgoing arcs to customers at each customer node.

For a given δ, we solve the δ-compact formulation with standard off-the-shelf commercial MIP solvers. We use Gurobi 8.0, since preliminary experiments have shown that it provides high quality upper bounds relatively fast. When a new solution is found, we store the corresponding solution into a global variable that is accessible by the branch-and-cut algorithm (implemented in CPLEX 12.8). The branch-and-cut algorithm checks during its search for improved upper bounds, and whenever available, it uses the solution provided by (P_δ) as new incumbent solution.

The overall working of the upper bound procedure is as follows. We iteratively solve δ-compact formulations. We differentiate three different phases in our upper bound procedure, where each phase is characterized by a series of values for δ and a maximum run-time. In the first phase, we allow for a short run-time for small values of δ in order to find an upper bound relatively quick. The second phase allows for more time solving (P_δ), starting at the δ that has provided the best upper bound from the first phase. If the second stage is finished, we turn to a third phase that is characterized by long run-times and large deltas.

When we increase δ or change from phase, the new model is initialized with the best solution so far. Based on initial experiments, for the first phase we range δ from 5 to $\min\{10, |V|\}$ with a maximum run-time of 20 seconds for each δ. In the second phase, we set the maximum run-time to 300 seconds and let δ range up to $\min\{25, |V|\}$. Finally, the run-time is increased to 600 seconds and we let δ range up to $\min\{50, |V|\}$.

Using the upper bound procedure has three clear advantages. First, if the upper bound becomes close enough to the lower bound, variable fixing (i.e., pricing out variables by their reduced costs) will eliminate a significant number of variables thereby decreasing the problem size. Second, the branch-and-bound tree will have a smaller size, thereby being easier and quicker to maintain. Third, CPLEX will normally dive into the branch-and-bound tree in order to find some incumbent solution. We noticed that this behaviour of CPLEX is less expensive in terms of run-time when the upper bound from the matheuristic (using Gurobi) is used to provide new incumbent solutions.

5.2 Branch-and-cut implementation

The separation algorithms are coded in the user- and lazy callback routines of CPLEX 12.8. The user callback is called for every fractional LP solution, whereas the lazy callback is called for every integer LP solution. As the user callback is rather expensive in terms of run-time, it is only called for the first N^{user} nodes of the branch and bound tree, and once these are processed it is called once every F^{user} branch-and-bound nodes. We examine the branch-and-bound nodes in a worst-bound first manner, which implies that we call our separation algorithms on branch-and-bound nodes that determine the current global lower bound. Initial experiments have shown that $N^{\text{user}} = 200$ and $F^{\text{user}} = 200$ are suitable values for all problem types.

The root-node of the branch-and-bound tree is additionally controlled by the parameter N^{root} which specifies the maximum number of so-called cut loops (i.e., the number of times iterating between solving the LP relaxation and adding valid inequalities). Although adding more valid inequalities to the root node will increase the LP relaxation, it will also result in more complex rows of the simplex tableau that are notoriously more difficult to process efficiently, leading to increased run-times for finding LP relaxations in branch-and-bound nodes. Initial experiments have shown that $N^{\text{root}} = 250$ is a suitable value for all problem types.

An overview of all the valid inequalities included in the branch-and-cut algorithm is provided in Table 2. It specifies the order in which all separation routines are executed, which is kept fixed throughout all the numerical experiments as presented in this paper. We employ a few common acceleration strategies to speed-up the branch-and-cut algorithm. First, while calling the valid inequalities in the order as specified in Table 2, we terminate the callback procedure as soon as one of the valid inequalities has found an inequality violating the current LP solution. Second, a valid inequality is only added if it is violated by more than a predetermined threshold. Third, we limit the number of valid inequalities that we add during each call of their corresponding separation procedures.

Table 2: Branch-and-cut details regarding the order of separation procedures being called, their violation threshold, and the maximum number of added inequalities per call of the separation procedure. A * indicates that there is no limit on the number of added inequalities

Order	Valid inequality	Threshold	Max number of cuts
1	Subtour Elimination constraints (4)	0.1	100
2	Path-eliminating constraints (3) or (21)	0	*
3	D_k^+- and D_k^- depot constraints (7) and (8)	0	*
4	D_k^+- and D_k^- tour size constraints (19) and (20)	0	*
5	CAT inequalities (9)	0	*
6	D_k^+- and D_k^+ inequalities (5) and (6)	0	*
7	Framed capacity inequalities (17)	0.1	10
8	Homogeneous multi-star inequalities (15)	0.1	10
9	Generalized large multi-star inequalities (16)	0.1	10
11	ATSP-comb inequalities (12)	0.1	5
11	T1-comb inequalities (14)	0.1	5
12	T-comb inequalities (14)	0.1	5

In addition, we include all subtour elimination constraints of size 2 directly into our model formulation. We also consider subtour elimination constraints of size 3 and size 4, but including all of them is computationally intractable. Hence, we include, for each customer node, the 5 vehicle subtours of shortest length. We add those to a pool of lazy constraints which are then automatically checked for violation at each integer LP solution during the branch-and-bound. Moreover, for instances with more than 100 customers we do not consider vehicle subtours of size 4, and for instances with more than 200 customers we do not consider subtours of size 3. Finally, when solving A-MDTSP instances, we limit the maximum number of outgoing arcs of any depot to 1.

6 Numerical experiments

In this section, we assess the numerical performance of the branch-and-cut framework. We start by comparing the performance of our branch-and-cut framework with the (only yet existing) set of benchmark instances for A-MDVRPs by Bektaş et al. [4]. Thereafter we propose a set of newly developed benchmark instances. These instances are then used to study the impact of the valid inequalities on the root node optimality gaps, and to examine the numerical performance of our complete branch-and-cut-framework.

To study the effectiveness of the proposed branch-and-cut framework (i.e., the valid inequalities and the upper bound procedure), we compare the framework with a standard branch-and-cut algorithm consisting of subtour elimination constraints and the depot-fixing constraints in the fashion of Laporte et al. [14], which are commonly used in the literature (see, e.g., Belenguer et al. [6]) and can be considered state-of-the-art. We refer to this standard branch-and-cut algorithm without the upper bound procedure and without the new valid inequalities as the *basic algorithm*.

Our branch-and-cut framework is implemented in *C++17* and uses CPLEX 12.8 for the branch-and-cut algorithm and Gurobi 8.0 for the upper bound procedure. All experiments are performed on an Intel Xeon E5 2680v3 CPU (2.5GHz) with 24GB memory. Our framework uses six parallel running threads (which are commonly available on new desktop machines), from which five are assigned to the branch-and-cut algorithm and one to the upper bound procedure. Besides the 24GB memory, we allow the algorithms to store nodes on a hard drive with 50GB storage. We note in advance that the additional storage is particularly used by the basic algorithm and is typically not required for our branch-and-cut framework. In order to foster future research, we provide free access to our implementation of the algorithms and to our benchmark instances[2].

6.1 Comparison to Bektaş et al. [4]

To the best of our knowledge, there is only a single set of benchmark instances available that coincides with one of the four problem variants we consider in this study. Namely, the A-MDTSP instances of Bektaş et al. [4].

The performance of our branch-and-cut framework on the instances of Bektaş et al. [4] is shown in Table 3. For each instance, we present the root node relaxation, the best upper bound, and

[2]When published: URL of public Github repository here

the run-time in seconds. Column ΔGap (%) presents the relative root node gap reduction of our framework compared to the algorithm of Bektaş et al. [4], where we defined the gap as the difference between the optimal objective value and the root node relaxation.

Table 3: Results on the benchmark instances of Bektaş et al. [4]. The lower bounds of the two unsolved instances (marked with an asterisk symbol) are 1624 and 1614. The column ΔGap (%) shows the relative gap reduction between the root node relaxation and the optimal objective value of our framework compared to that of Bektaş et al. [4]

	Results of Bektaş et al. [4]			Branch-and-cut framework			
	Root	obj.	time (s)	Root	obj.	time (s)	ΔGap (%)
bgs-100-05-1a	865.00	879	5	879.00	879	3	100.0
bgs-100-10-1a	856.25	873	20	871.04	873	5	88.3
bgs-100-20-1a	848.85	861	25	857.72	861	5	73.0
bgs-100-05-2a	948.54	954	5	954.00	954	2	100.0
bgs-100-10-2a	945.37	952	5	952.00	952	2	100.0
bgs-100-20-2a	931.00	933	4	933.00	933	1	100.0
bgs-100-05-3a	893.41	904	13	904.00	904	4	100.0
bgs-100-10-3a	890.92	904	28	901.46	904	7	80.6
bgs-100-20-3a	871.07	879	30	877.02	879	5	75.0
bgs-200-10a	1291.88	1296	5	1296.00	1296	5	100.0
bgs-200-20a	1290.14	1295	5	1295.00	1295	4	100.0
bgs-200-40a	1288.33	1293	12	1293.00	1293	18	100.0
bgs-300-10a	1609.54	* 1627	10800	1621.12	**<u>1626</u>**	292	70.4
bgs-300-20a	1604.09	1622	10387	1615.56	1622	220	64.0
bgs-300-30a	1599.22	* 1617	10800	1610.64	1617	173	64.2
bgs-300-40a	1593.11	1606	799	1602.08	1606	62	69.6
bgs-300-60a	1583.37	1592	533	1590.72	1592	166	85.2
Average	1171.18	-	1969	1179.61	-	57	86.5

Our framework solves all instances within five minutes, thereby solving two instances that were not solved before (namely bgs-300-10a and bgs-300-30a). We also improve the best known upper bound for instance bgs-300-30a, which is underlined and bold. The run-times for small instances (up to 200 customers) are comparable, and for the larger instances we clearly outperform the algorithm of Bektaş et al. [4]. For instance, our framework solves instance bgs-300-20a around fifty times faster than the algorithm of Bektaş et al. [4]. Furthermore, the developed valid inequalities are able to considerably close the root node gaps for all instances (on average by 86.5% compared to Bektaş et al. [4]), and eight instances are solved in the root node.

6.2 New benchmark instances

We consider three classes of asymmetric cost-structures (called I, II, and III), which are in line with the ATSP instances of Fischetti and Toth [10]. Recall that c_{ij} is the traveling cost between nodes $i, j \in V'$, and let $U\{a, b\}$ denote the uniform integer distribution with support $\{a, a+1, \ldots, b-1, b\}$. The three asymmetric cost-structure classes are generated as follows.

I. $c_{ij} \sim U\{1, 1000\}$.

II. $c_{ij} := a_{ij} + b_{ij}$, where $a_{ij} = a_{ji} \sim U\{1, 1000\}$ and $b_{ij} \sim U\{1, 20\}$.

III. $c_{ij} := d_{ij} + b_{ij}$, where d_{ij} is the floored Euclidean distance between i and j, and b_{ij} is as defined above.

For the instances of Class III, we considered a grid $[0, 500] \times [0, 500]$ from which we draw the geographical locations of the nodes. For half of the nodes, we did this uniformly along the grid, while for the other half, we created clusters of random sizes.

In total, we consider twelve problem categories, namely four problem variants (A-MDTSP, A-MDmTSP, A-MDCVRP, A-CLRP) with each three asymmetric cost-structures (I, II, III). We varied the number of customers and depots for each of the problem variants. The A-MDTSP instances range from 200 – 400 customers with 30–50 depots, the A-MDmTSP instances contain between 50 – 100 customer with 5 – 20 depots, and the A-MDCVRP and A-CLRP instances consist of 40 – 85 customers and 5 – 20 depots. For each considered combination of the number of customers and the number of depots we created three replications. Thus we have 45×3, 69×3, 63×3, and 63×3 instances for the A-MDTSP, A-MDmTSP, A-MDCVRP, and A-CLRP, respectively, giving a total of 720 instances.

Each instance is named $XX\_n\_r\_x$, where XX is the problem variant considered (i.e., A-MDTSP, A-MDmTSP, A-MDCVRP, A-CLRP), n is the number of customers, r is the number of depots, and x the replication index.

The other parameter values are independently drawn from a uniform distribution for each instance. For the A-MDmTSP, we draw the minimum tour length uniformly between 2 and 6, and the maximum tour length uniformly between 15 and 25. For the A-MDCVRP, we let customers demand be random between 15 and 25, and the vehicles capacity is uniformly drawn between 150 and 300. For the A-CLRP, capacity characteristics are equal to the A-MDCVRP and the depot opening costs are generated uniformly between 500 and 3000 for each instance. An overview of all the instances and the detailed results of solving these with basic algorithm and the branch-and-cut framework are provided in the appendix in Tables 8–15.

6.3 Valid inequalities and effect on root node

To examine the effect of the proposed valid inequalities on the root node relaxation, we consider six combinations of the valid inequalities discussed in this study. To provide a fair comparison of the impact of our inequalities, we disabled all standard CPLEX cuts that are otherwise added by default. We remark that similar results were observed when these cuts are enabled.

Table 4 presents the average root node relaxation when only the path elimination constraints by Laporte et al. [14] are used (column *Path*), when only the D_k depot inequalities are used (column D_k *depot*), and when both are used (column *Both*). To show the effect of the other valid inequalities, we include the root node results with and without separating the other valid inequalities for each of the three different depot fixing methods. Next, we define the root node gap as the difference between the best known solution and the root node relaxation. The column Δ*Gap (%)* shows the average and maximum percentage of the root node optimality gap reduction when we use our complete branch-and-cut framework (i.e., with fract. cuts, Both) instead of the basic algorithm (i.e., without fract. cuts, Path).

Adding the proposed valid inequalities reduces root node optimality gaps with 16.5%, 8.3%, 7.8%, and 0.4% on average for the A-MDTSP, A-MDmTSP, A-MDCVRP, and A-CLRP instances, respectively. Furthermore, the asymmetry class considerably affects the effectiveness of the proposed valid inequalities, as the root node gap reduces on averages by 1.5%, 9.5%, and 13.8% for the asymmetry classes I, II, and III, respectively. Thus, for the first three problem types, the effect of the added valid inequalities is clearly visible in the root node optimality gaps, whereas for the last problem type the strength of the valid inequalities will become more clear in the complete run of the branch-and-cut framework (discussed in the next section). This is mainly related to the large root node optimality gaps of the A-CLRP instances (mostly between 20-30%) and for the A-MDTSP-I (mostly larger than 75% due to the lack of reasonable upper bounds), while the other problem types typically have a root node optimality gap smaller than 10%.

In addition, using the D_k depot inequalities instead of the path elimination constraints results in higher root node relaxations for 715 out of 720 instances. Including our fractional valid inequalities

Table 4: Average root node lower bounds for various combinations of valid inequalities

		without fract. cuts			with fract. cuts			ΔGap (%)	
Type	Asy.	Path	D_k depot	Both	Path	D_k depot	Both	Avg.	Max.
A-MDTSP	I	1395.0	1395.6	1395.6	1395.4	1396.0	1396.0	1.6	17.4
	II	4025.0	4027.3	4027.3	4026.7	4028.9	4028.9	20.7	67.2
	III	7715.8	7729.3	7729.3	7734.1	7747.1	7747.1	27.2	49.4
	Avg.	*4378.6*	*4384.1*	*4384.1*	*4385.4*	*4390.7*	*4390.7*	*16.5*	-
A-MDmTSP	I	1485.4	1485.8	1485.9	1486.1	1486.6	1486.6	1.4	14.5
	II	2530.9	2537.6	2537.7	2533.3	2540.0	2540.0	10.1	32.3
	III	3722.9	3738.5	3738.5	3732.6	3748.0	3748.0	13.5	35.8
	Avg.	*2579.7*	*2587.3*	*2587.4*	*2584.0*	*2591.5*	*2591.5*	*8.3*	-
A-MDCVRP	I	1486.7	1488.9	1489.0	1488.0	1490.2	1490.3	2.5	28.3
	II	2388.1	2396.1	2396.1	2392.7	2400.7	2400.8	6.7	30.0
	III	3482.1	3497.5	3497.5	3493.9	3507.0	3507.0	14.3	50.5
	Avg.	*2452.3*	*2460.8*	*2460.9*	*2458.2*	*2466.0*	*2466.0*	*7.8*	-
A-CLRP	I	2565.8	2566.4	2566.4	2566.9	2567.5	2567.5	0.4	7.0
	II	3495.3	3497.2	3497.2	3497.0	3498.6	3498.6	0.4	1.6
	III	4318.7	4319.9	4319.9	4324.9	4325.8	4325.8	0.5	2.0
	Avg.	*3459.9*	*3461.2*	*3461.2*	*3462.9*	*3464.0*	*3464.0*	*0.4*	-

does (almost) not affect the difference between the column *Path* and D_k depot . We conclude that the D_k depot inequalities are computationally stronger than the traditional path elimination constraints, however, the two inequalities are not redundant to each other. We therefore choose to use both the traditional path elimination constraints and the D_k depot constraints in the branch-and-cut framework.

To show the notable performance of the proposed valid inequalities for some particular instances, we zoom in on instance A-MDTSP-II_20_200_1 with optimal objective value 3240. Adding all proposed inequalities increases the root node relaxation from 3227.4 to 3235.9, i.e., the root node optimality gap is closed by 67.2%. We note that both algorithms solve the root node within a second.

6.4 Performance of the branch-and-cut framework

We solved all benchmark instances of the 12 problem categories with both the branch-and-cut framework and the basic algorithm. Recall that the basic algorithm can be considered as what is readily known in the literature for solving the A-MDVRP, namely, it is a branch-and-cut algorithm with subtour elimination constraints and the traditional path elimination constraints.

Tables 5–7 present various summary statistics of the basic algorithm, the branch-and-cut frame-

Table 5: Summary statistics of the performance of the basic algorithm. N_{sol} gives the number of instances for which a feasible solution is found, $\#_{\leq x\%}$ indicate the number of instance with an optimality gap smaller or equal to $x\%$, and *nodes* is the number of processed branch-and-bound nodes

Type	Asy.	#inst	N_{sol}	$\#_{\leq 0\%}$	$\#_{\leq 1\%}$	$\#_{\leq 5\%}$	LB	time (s)	nodes (10^3)
A-MDTSP	I	45	14	7	8	10	1403.80	15368*	107
	II	45	41	33	39	40	4041.02	5886*	46
	III	45	45	28	43	45	7819.31	7876*	129
A-MDmTSP	I	69	47	29	30	32	1535.74	10993	82
	II	69	65	46	49	52	2605.44	7628	80
	III	69	69	48	50	60	3883.13	6330	83
A-MDCVRP	I	63	51	31	31	35	1567.79	10620	105
	II	63	44	18	19	20	2470.18	13237	97
	III	63	62	42	44	56	3636.46	7314	144
A-CLRP	I	63	55	35	36	37	2986.91	9163	56
	II	63	49	29	31	32	4266.30	10256	56
	III	63	52	24	24	29	5383.89	12037	134

*We remark that 29, 9, and 19 out of 45 A-MDTSP instances stopped prematurely by reaching the memory limit.

Table 6: Summary statistics of the performance of the branch-and-cut framework. N_{sol} gives the number of instances for which a feasible solution is found, $\#_{\leq x\%}$ indicate the number of instance with an optimality gap smaller or equal to $x\%$, and *nodes* is the number of processed branch-and-bound nodes

Type	Asy.	# inst	N_{sol}	$\#_{\leq 0\%}$	$\#_{\leq 1\%}$	$\#_{\leq 5\%}$	LB	time (s)	nodes (10^3)
A-MDTSP	I	45	45	12	12	18	1412.43	13980*	127
	II	45	45	39	39	41	4043.86	4399	21
	III	45	45	45	45	45	7831.82	2400	14
A-MDmTSP	I	69	69	41	45	58	1557.13	7887	82
	II	69	69	59	59	65	2617.60	3967	65
	III	69	69	56	61	69	3897.58	4657	82
A-MDCVRP	I	63	63	37	38	48	1592.97	7894	107
	II	63	63	25	26	38	2516.33	11430	141
	III	63	63	44	51	63	3652.26	6693	139
A-CLRP	I	63	63	43	47	57	3018.25	7050	65
	II	63	62	35	36	51	4329.92	8517	80
	III	63	63	25	26	40	5590.65	11492	132

*We remark that 13 out of 45 A-MDTSP instances stopped prematurely by reaching the memory limit.

work, and a comparison of both, respectively. For detailed information on an individual instance level, we refer to Tables 8–15 in the Appendix. In Tables 5 and 6, the columns $\#_{\leq x\%}$ indicate the number of instances with an optimality gap smaller than $x\%$, *LB* is the average lower bound of all instances, *time* is the average run-time of all instances, and *nodes* is the number of processed branch-and-bound nodes. In Table 7, *time* denotes the average run-time for instances that are solved to optimality by both algorithms, *gap* presents the average gap for instances that are not solved to optimality by one of the algorithms, and *LB* is the average lower bound for the remaining

Table 7: Summary statistics comparing both algorithms. *Time* is the average time of the instances that are solved by both algorithms, *gap* the average gap of the instances where both algorithms found an upper bound but at least one did not prove optimality, and *LB* the average lower bound of the remaining instances, i.e., at least one algorithm did not find an upper bound.

		basic algorithm			branch-and-cut framework		
Type	Asy.	time (s)	gap (%)	LB	time (s)	gap (%)	LB
A-MDTSP	I	1080	21.3	1396.6	406	1.3	1407.5
	II	1481	2.7	4329.4	1051	9.3	4344.5
	III	1729	0.6	-	869	0.0	-
A-MDmTSP	I	1327	196.6	1590.7	104	0.9	1632.3
	II	2442	104.7	2659.4	362	1.1	2704.9
	III	1224	6.6	-	376	1.1	-
A-MDCVRP	I	3003	156.6	1545.3	344	2.8	1596.0
	II	1329	103.3	2634.4	164	3.4	2697.7
	III	1971	5.3	4253.8	1121	1.4	4341.2
A-CLRP	I	2094	33.8	3444.2	364	1.4	3534.6
	II	1177	45.6	4622.7	252	2.5	5009.7
	III	2348	28.1	5757.3	1519	5.0	6194.9
	Avg.	*1832*	*63.1*	*2656.3*	*586*	*2.4*	*2770.2*

instances (i.e., instances for which at least one algorithm could not find a feasible solution).

The key observation is that our branch-and-cut framework solves considerably more instances to optimality for each problem type. For instance, from the 69 A-MDmTSP-I instances, our framework solves 41 instances to optimality, while the basic algorithm only solves 29 instances. The main reason is that the branch-and-cut framework provides higher lower bounds than the basic algorithm, which is observed uniformly for all the 12 problem categories. For the A-MDmTSP-I instances, the average lower bound increases from 1535.74 to 1557.13.

Second, the basic algorithm has difficulty finding feasible solutions, which our framework effectively overcomes by using the upper bound procedure described in Section 5.1. The basic algorithm fails to find feasible solutions for 35, 26, 32, and 33 instances of the A-MDTSP, A-MDmTSP, A-MDCVRP and A-CLRP, respectively, whereas the branch-and-cut framework finds feasible solutions to all instances. This is as well exemplified by the number of instances with relatively small optimality gaps. Using the branch-and-cut framework instead of the basic algorithm increases the number of instances with optimality gaps smaller than 5% from 90 to 104, from 144 to 194, from 111 to 151, and from 100 to 150 for the four problem variants, respectively. The average optimality gap for the instances for which both algorithms found feasible solutions but did not solve to optimaly, equals 2.4% for the branch-and-cut framework and 63.1% for the basic algorithm, as is shown in Table 7.

Comparing the run-times of the basic algorithm and the branch-and-cut framework, it should be noted that the branch-and-cut framework is on average faster than the basic algorithm for each of the 12 problem categories. Significant run-time decreases can be found when we zoom in on individual instances. For instance, A-CLRP-II_75_10_2 is solved to optimality by both algorithms, however the basic algorithm needs 13549 seconds compared to 1422 seconds for the branch-and-cut framework, which is around ten times faster. On average, the run-time of the instances solved to optimality by both algorithms equals 586 seconds for the branch-and-cut framework and 1832 seconds for the basic algorithm, as is denoted in Table 7.

In addition, due to adding all valid inequalities, the branch-and-bound tree is searched more effectively. This has two major consequences. First, for some problem categories, this leads to a reduction in the number of processed branch-and-bound nodes. For instance, a reduction from $129 \cdot 10^3$ to $14 \cdot 10^3$ for problem category A-MDTSP-III. Second, searching the branch-and-bound tree more effectively may result in the pruning of more branch-and-bound nodes, and thereby the ability to process more nodes. In other words, memory is used more efficiently which is exemplified by the number of instances stopped before the time limit due to reaching the memory limit of 75GB (57 versus 13).

Finally, a difference in computational performance is noted among the different asymmetry classes. Comparing between the basic algorithm and the branch-and-cut framework, it is notable that especially for asymmetry class I the number of instances solved to optimality (or within small optimality gaps) increases when using the branch-and-cut framework instead of the basic algorithm. There is no clear ordering in the difficulty of the asymmetry classes over the all 12 problem categories. However, within each problem variant, a clear ordering in difficulty is observed. For the A-MDTSP and A-MDmTSP, it holds that when distances are closer to the Euclidian one, both algorithms are able to provide higher quality solutions. For the A-MDCVRP this effect is not as clear, as asymmetry class II seems to be most difficult there. Finally, for the A-CLRP class, the Euclidean arc costs with noise (i.e., class III) seems to be most difficult.

7 Conclusion

In this paper we have studied a class of routing problems that are characterized by an asymmetric cost structure and the presence of multiple depots. Based on the derivation of new valid inequalities with model-describing capabilities, we have developed a generic branch-and-cut framework that can solve multiple variants of the A-MDVRP efficiently.

Backbone of the branch-and-cut framework is a branch-and-cut algorithm that exploits, among others, a series of generalizations of so-called D_k inequalities that result in a set of constraints that eliminate subtours, ensure that tours start and end at the same depot, and impose limits on the maximum number of customers in a tour. Next to the branch-and-cut algorithm, the framework employs an upper bound procedure which is based on iteratively solving a compact mixed integer programming formulation of a restricted version of the A-MDVRP.

The computational efficiency of the branch-and-cut framework is assessed on known and new benchmark sets. On existing instances of the A-MDTSP our algorithm has proved optimality to all instances, significantly closing the gaps at the root node compared to a state-of-the-art approach. Our method uses, on average, only 4.3% of the run-time.

When solving a large set of newly developed benchmark instances for four problem variants of the A-MDVRP, the derived valid inequalities appear to be effective for three of the problem variants. For instance, root node optimality gaps are closed by up to 67.2% and on average by 10.2%. To further assess the performance of the branch-and-cut framework, we developed a basic branch-and-cut algorithm based on traditional methods to model and solve A-MDVRPs. Overall, the framework significantly outperforms the basic algorithm as it solves more instances to optimality due to improved lower bounds and upper bounds. In addition, the new framework is considerably faster and uses less memory. The performance is exemplified by the ability to solve Asymmetric Multi-Depot Traveling Salesman Problem instances up to 400 customers and 50 depots (before, only solutions to instances of 300 customers and 60 depots were available in the literature).

Our implementation of the branch-and-cut framework and the newly constructed benchmark instances are openly shared to stimulate future research in this field. Such future research might take our branch-and-cut framework as a starting point to solve new and practically inspired routing problems with asymmetric cost structures and multiple depots.

Acknowledgements

This work was partly supported by the Netherlands Organisation for Scientific Research (NWO) through grants no 438-13-216 (Michiel A. J. Uit het Broek and Albert H. Schrotenboer) and no 439-16-612 (Kees Jan Roodbergen and Bolor Jargalsaikhan), and by the Canadian Natural Sciences and Engineering Research Council (NSERC) under grant 2014-05764 (Leandro C. Coelho).

References

[1] Ascheuer N, Fischetti M, Grötschel M (2001) Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. Mathematical Programming 90(3):475–506

[2] Balas E (1989) The asymmetric assignment problem and some new facets of the traveling salesman polytope on a directed graph. SIAM Journal on Discrete Mathematics 2(4):425–451

[3] Balas E, Fischetti M (1993) A lifting procedure for the asymmetric traveling salesman polytope and a large new class of facets. Mathematical Programming 58(1-3):325–352

[4] Bektaş T, Gouveia L, Santos D (2017) New path elimination constraints for multi-depot routing problems. Networks 70(3):246–261

[5] Bektaş T, Gouveia L, Martínez-Sykora A, Salazar-González JJ (2019) Balanced vehicle routing: Polyhedral analysis and branch-and-cut algorithm. European Journal of Operational Research 273(2):452 – 463

[6] Belenguer JM, Benavent E, Prins C, Prodhon C, Calvo RW (2011) A branch-and-cut method for the capacitated location-routing problem. Computers & Operations Research 38(6):931–941

[7] Benavent E, Martínez A (2013) Multi-depot multiple tsp: a polyhedral study and computational results. Annals of Operations Research 207(1):7–25

[8] Fernández E, Laporte G, Rodríguez-Pereira J (2017) A branch-and-cut algorithm for the multidepot rural postman problem. Transportation Science 52(2):353–369

[9] Fischetti M (1991) Facets of the asymmetric traveling salesman polytope. Mathematics of Operations Research 16(1):42–56

[10] Fischetti M, Toth P (1997) A polyhedral approach to the asymmetric traveling salesman problem. Management Science 43(11):1520–1536

[11] Fischetti M, Toth P, Vigo D (1994) A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. Operations Research 42(5):846–859

[12] Gouveia L, Riera-Ledesma J, Salazar-González JJ (2013) Reverse multistar inequalities and vehicle routing problems with a lower bound on the number of customers per route. Networks 61(4):309–321

[13] Lahyani R, Coelho LC, Renaud J (2018) Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem. OR Spectrum 40(1):125–157

[14] Laporte G, Nobert Y, Taillefer S (1988) Solving a family of multi-depot vehicle routing and location-routing problems. Transportation Science 22(3):161–172

[15] Letchford AN, Salazar-González JJ (2006) Projection results for vehicle routing. Mathematical Programming 105(2-3):251–274

[16] Letchford AN, Salazar-González JJ (2015) Stronger multi-commodity flow formulations of the capacitated vehicle routing problem. European Journal of Operational Research 244(3):730–738

[17] Letchford AN, Eglese RW, Lysgaard J (2002) Multistars, partial multistars and the capacitated vehicle routing problem. Mathematical Programming 94:21–40

[18] Lysgaard J, Letchford AN, Eglese RW (2004) A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming 100(2):423–445

[19] Schrotenboer AH, Uit het Broek MAJ, Jargalsaikhan B, Roodbergen KJ (2018) Coordinating technician allocation and maintenance routing for offshore wind farms. Computers & Operations Research 98:185–197

[20] Schrotenboer AH, Ursavas E, Vis IFA (2019) A branch-and-price-and-cut algorithm for resource constrained pickup and delivery problems. Transportation Science (Forthcoming)

Appendix: Solutions to benchmark instances

In Tables 8 - 15 we provide the statistics of solving the benchmark instances with the branch-and-cut framework and with the basic algorithm. Each table is dedicated to a particular problem variant, with each four columns of a table indicating the asymmetry class.

Table 8: Branch and cut framework solutions to the A-MDTSP instances

Instance A-MDTSP	I					II					III				
	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)
_200_20_0	1395	1395	**0.00**	0	75	3094	3094	**0.00**	0	35	6320	6320	**0.00**	3	229
_200_20_1	1392	1392	**0.00**	3	113	3240	3240	**0.00**	0	22	5896	5896	**0.00**	0	392
_200_20_2	1574	1574	**0.00**	36	401	3567	3567	**0.00**	0	135	5748	5748	**0.00**	0	373
_200_30_0	1348	1348	**0.00**	309	6582	3313	3313	**0.00**	0	57	5836	5836	**0.00**	0	147
_200_30_1	1437	1437	**0.00**	16	341	3470	3470	**0.00**	0	118	6102	6102	**0.00**	0	166
_200_30_2	1357	1357	**0.00**	182	4682	3112	3112	**0.00**	0	69	5927	5927	**0.00**	3	198
_200_40_0	1465	1495	2.07	515	18000	3149	3149	**0.00**	4	203	5861	5861	**0.00**	0	75
_200_40_1	1472	1520	3.28	483	18000	3450	3450	**0.00**	5	294	6131	6131	**0.00**	3	190
_200_40_2	1421	1421	**0.00**	622	11604	3227	3227	**0.00**	0	60	6526	6526	**0.00**	0	93
_300_20_0	1542	1542	**0.00**	3	321	3676	3676	**0.00**	0	131	7660	7660	**0.00**	0	1462
_300_20_1	1474	1474	**0.00**	2	414	3747	3747	**0.00**	0	134	7704	7704	**0.00**	0	1634
_300_20_2	1401	1401	**0.00**	21	1175	3833	3833	**0.00**	0	372	7535	7535	**0.00**	1	468
_300_30_0	1491	1547	3.74	200	18000	3848	3848	**0.00**	1	404	7356	7356	**0.00**	0	517
_300_30_1	1524	1566	2.76	151	18000	4245	4245	**0.00**	3	681	7565	7565	**0.00**	3	553
_300_30_2	1516	1947	28.45	168	18000	3895	3895	**0.00**	3	513	7282	7282	**0.00**	1	548
_300_40_0	1466	7341	400.62	130	18000	4247	4247	**0.00**	15	3348	7277	7277	**0.00**	0	192
_300_40_1	1462	3259	122.86	141	16646	3885	3885	**0.00**	202	16371	7440	7440	**0.00**	0	738
_300_40_2	1364	1549	13.53	191	18000	3943	3943	**0.00**	13	2054	7323	7323	**0.00**	19	1947
_350_30_0	1410	1410	**0.00**	105	5720	4339	4339	**0.00**	24	7065	8047	8047	**0.00**	1	831
_350_30_1	1360	3579	163.18	111	14466	4152	4152	**0.00**	1	371	8169	8169	**0.00**	1	1293
_350_30_2	1328	1365	2.75	255	18000	4361	4361	**0.00**	6	1807	8373	8373	**0.00**	6	1521
_350_40_0	1295	4096	216.21	107	14471	4249	4461	5.00	65	18000	8253	8253	**0.00**	97	6767
_350_40_1	1374	4036	193.79	103	16034	4199	4199	**0.00**	25	4126	8642	8642	**0.00**	18	2048
_350_40_2	1357	13275	878.27	102	15834	4082	4082	**0.00**	2	887	8214	8214	**0.00**	7	1001
_350_50_0	1434	17573	1125.67	94	18000	4132	4132	**0.00**	10	3000	8169	8169	**0.00**	1	747
_350_50_1	1483	5980	303.16	105	18000	4245	4245	**0.00**	4	1794	7843	7843	**0.00**	0	2383
_350_50_2	1349	24896	1745.32	103	17278	4350	51724	1089.17	62	18000	8144	8144	**0.00**	12	1444
_375_30_0	1472	1679	14.06	110	18000	4163	4163	**0.00**	1	894	8306	8306	**0.00**	21	3249
_375_30_1	1440	1440	**0.00**	36	3658	4212	4212	**0.00**	0	705	8516	8516	**0.00**	2	1510
_375_30_2	1429	11941	735.67	78	18000	4393	4393	**0.00**	0	1039	8668	8668	**0.00**	18	2901
_375_40_0	1396	1857	33.00	74	18000	4557	4557	**0.00**	35	8574	8550	8550	**0.00**	2	1486
_375_40_1	1440	2263	57.10	84	18000	4346	4346	**0.00**	6	1654	8850	8850	**0.00**	0	1356
_375_40_2	1374	43445	3061.64	81	13031	4301	4301	**0.00**	152	11745	8388	8388	**0.00**	63	8648
_375_50_0	1418	7498	428.64	86	18000	4310	4310	**0.00**	0	925	8885	8885	**0.00**	1	742
_375_50_1	1265	15156	1098.15	80	13544	4283	49826	1063.28	54	18000	8602	8602	**0.00**	16	2395
_375_50_2	1353	52028	3744.53	88	15535	4298	4298	**0.00**	49	8498	8549	8549	**0.00**	79	7993
_400_30_0	1351	2066	52.93	76	17522	4578	4578	**0.00**	4	1451	8802	8802	**0.00**	31	8255
_400_30_1	1437	14744	926.33	73	17697	4429	4429	**0.00**	11	2407	9311	9311	**0.00**	55	11696
_400_30_2	1304	1354	3.82	114	18000	4327	4327	**0.00**	0	1316	9028	9028	**0.00**	17	2947
_400_40_0	1419	1627	14.68	80	18000	4534	4705	3.78	50	18000	8539	8539	**0.00**	3	1459
_400_40_1	1464	40257	2649.40	68	18000	4534	4534	**0.00**	1	1844	9053	9053	**0.00**	55	13157
_400_40_2	1404	3356	139.00	61	18000	4447	7587	70.61	56	18000	8766	8766	**0.00**	31	5334
_400_50_0	1378	7168	420.17	80	16237	4277	4277	**0.00**	3	2257	8848	8848	**0.00**	7	2023
_400_50_1	1391	2937	111.19	71	13682	4496	50245	1017.43	35	18000	8520	8520	**0.00**	5	1831
_400_50_2	1332	17694	1228.88	80	18000	4439	4439	**0.00**	7	2578	8908	8908	**0.00**	11	3025

Table 9: Basic algorithm solutions to the A-MDTSP instances

Instance A-MDTSP	I					II					III				
	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)
_200_20_0	1395	1395	**0.00**	13	312	3094	3094	**0.00**	0	24	6320	6320	**0.00**	92	834
_200_20_1	1392	1392	**0.00**	16	299	3240	3240	**0.00**	0	25	5896	5896	**0.00**	7	87
_200_20_2	1574	1574	**0.00**	60	548	3567	3567	**0.00**	1	41	5748	5748	**0.00**	3	62
_200_30_0	1333	2340	75.59	213	18000	3313	3313	**0.00**	0	18	5836	5836	**0.00**	6	70
_200_30_1	1437	1437	**0.00**	149	3255	3470	3470	**0.00**	1	40	6102	6102	**0.00**	13	149
_200_30_2	1346	1430	6.24	336	14967	3112	3112	**0.00**	1	42	5927	5927	**0.00**	97	1020
_200_40_0	1444	-	-	217	18000	3149	3149	**0.00**	24	402	5861	5861	**0.00**	4	64
_200_40_1	1453	-	-	206	18000	3450	3450	**0.00**	26	522	6131	6131	**0.00**	120	1061
_200_40_2	1404	-	-	186	18000	3227	3227	**0.00**	0	44	6526	6526	**0.00**	13	115
_300_20_0	1542	1542	**0.00**	25	493	3676	3676	**0.00**	0	84	7660	7660	**0.00**	56	1044
_300_20_1	1474	1474	**0.00**	21	546	3747	3747	**0.00**	0	43	7704	7704	**0.00**	2	130
_300_20_2	1401	1401	**0.00**	59	2104	3833	3833	**0.00**	3	161	7535	7535	**0.00**	34	726
_300_30_0	1479	-	-	115	5670	3848	3848	**0.00**	6	350	7356	7356	**0.00**	14	253
_300_30_1	1517	1882	24.10	116	9789	4245	4245	**0.00**	20	703	7565	7565	**0.00**	107	1772
_300_30_2	1504	-	-	121	7025	3895	3895	**0.00**	9	314	7282	7282	**0.00**	141	2080
_300_40_0	1456	-	-	117	6504	4247	4247	**0.00**	52	1938	7277	7277	**0.00**	6	182
_300_40_1	1452	-	-	115	8312	3876	3889	0.33	168	11819	7440	7440	**0.00**	56	926
_300_40_2	1354	-	-	112	8784	3943	3943	**0.00**	104	3977	7297	7334	0.51	247	7303
_350_30_0	1404	1916	36.47	96	9324	4332	4357	0.58	102	9617	8047	8047	**0.00**	92	1841
_350_30_1	1351	-	-	90	4207	4152	4152	**0.00**	2	234	8169	8169	**0.00**	190	5137
_350_30_2	1323	1342	1.41	197	18000	4361	4361	**0.00**	16	579	8373	8373	**0.00**	100	3272
_350_40_0	1288	-	-	149	18000	4229	-	-	91	6237	8232	8260	0.34	215	8383
_350_40_1	1359	-	-	89	4748	4199	4199	**0.00**	85	5342	8603	8659	0.66	160	4893
_350_40_2	1347	-	-	88	6521	4082	4082	**0.00**	62	5108	8187	8224	0.45	384	8682
_350_50_0	1419	-	-	88	6542	4127	4142	0.37	124	9688	8169	8169	**0.00**	71	1519
_350_50_1	1474	-	-	87	7989	4245	4245	**0.00**	61	5229	7843	7843	**0.00**	7	278
_350_50_2	1341	-	-	88	9604	4335	-	-	91	5501	8101	8161	0.74	150	5452
_375_30_0	1463	-	-	80	5011	4163	4163	**0.00**	8	465	8282	8312	0.37	130	6198
_375_30_1	1438	1446	0.58	156	18000	4212	4212	**0.00**	1	198	8516	8516	**0.00**	33	865
_375_30_2	1418	-	-	79	5079	4393	4393	**0.00**	3	243	8637	8687	0.58	156	6641
_375_40_0	1386	-	-	79	4122	4548	4578	0.65	102	18000	8550	8550	**0.00**	45	1375
_375_40_1	1433	-	-	80	4178	4346	4346	**0.00**	45	2928	8850	8850	**0.00**	126	4915
_375_40_2	1367	-	-	79	3921	4290	4378	2.04	83	10177	8322	8407	1.02	117	4681
_375_50_0	1405	-	-	79	4576	4310	4310	**0.00**	6	402	8885	8885	**0.00**	250	5591
_375_50_1	1253	-	-	79	4576	4269	-	-	81	14512	8566	8616	0.59	149	4634
_375_50_2	1340	-	-	79	3981	4290	4311	0.49	133	18000	8516	8577	0.72	122	5516
_400_30_0	1347	-	-	172	18000	4578	4578	**0.00**	77	7497	8792	8805	0.14	521	18000
_400_30_1	1430	-	-	72	3828	4429	4429	**0.00**	29	1278	9263	9346	0.90	103	5066
_400_30_2	1300	1365	4.98	173	18000	4327	4327	**0.00**	6	497	9010	9031	0.23	218	8003
_400_40_0	1412	-	-	71	4492	4529	4556	0.60	154	18000	8539	8539	**0.00**	256	7222
_400_40_1	1457	-	-	71	5482	4534	4534	**0.00**	20	1044	8992	9104	1.25	96	4217
_400_40_2	1395	-	-	70	5580	4433	5189	17.06	84	6636	8726	8779	0.61	113	5501
_400_50_0	1370	-	-	69	5305	4277	4277	**0.00**	63	7739	8824	8862	0.43	167	4951
_400_50_1	1380	-	-	70	4162	4484	-	-	71	7642	8506	8523	0.20	555	18000
_400_50_2	1318	-	-	69	5515	4439	4439	**0.00**	34	1356	8908	8908	**0.00**	235	5805

Table 10: Branch and cut framework solutions to A-MDmTSP instances

Instance	I					II					III				
A-MDmTSP	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)
_50_5_0	1294	1294	**0.00**	0	2	1893	1893	**0.00**	0	1	2856	2856	**0.00**	0	10
_50_5_1	1187	1187	**0.00**	0	12	2487	2487	**0.00**	0	2	2901	2901	**0.00**	3	16
_50_5_2	1356	1356	**0.00**	1	10	1938	1938	**0.00**	0	1	3286	3286	**0.00**	10	60
_50_10_0	1681	1681	**0.00**	0	6	2680	2680	**0.00**	0	2	2595	2595	**0.00**	0	12
_50_10_1	1410	1410	**0.00**	0	9	2079	2079	**0.00**	0	3	2868	2868	**0.00**	1	6
_50_10_2	1372	1372	**0.00**	1	8	1902	1902	**0.00**	0	3	3013	3013	**0.00**	0	3
_60_5_0	1736	1736	**0.00**	16	221	2613	2613	**0.00**	6	32	3250	3250	**0.00**	73	494
_60_5_1	1636	1636	**0.00**	22	182	2424	2424	**0.00**	0	6	3431	3431	**0.00**	0	13
_60_5_2	1650	1650	**0.00**	97	1275	2507	2507	**0.00**	19	109	3883	3883	**0.00**	161	4967
_60_10_0	1354	1354	**0.00**	2	14	2421	2421	**0.00**	1	5	3140	3140	**0.00**	0	8
_60_10_1	1772	1772	**0.00**	0	10	2372	2372	**0.00**	0	2	3432	3432	**0.00**	0	12
_60_10_2	1873	1873	**0.00**	30	150	2528	2528	**0.00**	0	6	3200	3200	**0.00**	43	214
_70_5_0	1959	1997	1.94	177	18000	2568	2568	**0.00**	103	1949	3601	3601	**0.00**	2	46
_70_5_1	1428	1428	**0.00**	8	108	2661	2661	**0.00**	25	271	3511	3539	0.79	318	18000
_70_5_2	1651	1651	**0.00**	17	166	2757	2757	**0.00**	0	9	3559	3559	**0.00**	0	4
_70_10_0	1578	1578	**0.00**	0	19	2286	2286	**0.00**	0	7	3684	3684	**0.00**	14	122
_70_10_1	1538	1538	**0.00**	13	52	2479	2479	**0.00**	1	11	3332	3332	**0.00**	1	30
_70_10_2	1415	1415	**0.00**	8	21	2555	2555	**0.00**	0	12	3819	3819	**0.00**	0	8
_75_5_0	1630	1630	**0.00**	1	23	2590	2590	**0.00**	2	22	3851	3851	**0.00**	3	49
_75_5_1	1802	1802	**0.00**	1	30	2744	2744	**0.00**	2	23	3761	3761	**0.00**	3	93
_75_5_2	1827	1827	**0.00**	57	3631	2420	2420	**0.00**	12	149	4024	4024	**0.00**	105	2566
_75_10_0	1350	1350	**0.00**	6	66	2630	2630	**0.00**	119	4881	3545	3545	**0.00**	13	93
_75_10_1	1514	1551	2.43	182	18000	2807	2807	**0.00**	15	197	3987	3987	**0.00**	25	279
_75_10_2	1771	1826	3.08	169	18000	2725	2725	**0.00**	462	17452	3610	3610	**0.00**	325	5391
_80_5_0	1482	1510	1.87	126	18000	2413	2413	**0.00**	48	647	4296	4296	**0.00**	4	169
_80_5_1	1423	1467	3.13	215	18000	2622	2622	**0.00**	47	555	4248	4248	**0.00**	5	317
_80_5_2	1719	1719	**0.00**	34	644	2778	2778	**0.00**	173	6659	4096	4161	1.59	103	18000
_80_10_0	1344	1344	**0.00**	4	35	2525	2525	**0.00**	4	26	3721	3721	**0.00**	1	43
_80_10_1	1499	1499	**0.00**	22	484	2379	2379	**0.00**	14	63	3976	3976	**0.00**	436	6180
_80_10_2	1559	1559	**0.00**	25	419	2672	2672	**0.00**	3	24	3831	3831	**0.00**	0	33
_80_20_0	1352	1352	**0.00**	72	393	2075	2075	**0.00**	2	14	3744	3744	**0.00**	1	42
_80_20_1	1321	1321	**0.00**	39	166	2813	2813	**0.00**	34	314	3641	3641	**0.00**	1	30
_80_20_2	1584	1584	**0.00**	306	3182	2635	2635	**0.00**	53	626	3891	3891	**0.00**	40	355
_85_5_0	1721	1731	0.60	185	18000	2686	2686	**0.00**	9	106	4222	4244	0.53	233	18000
_85_5_1	1478	1611	9.01	118	18000	2844	2901	2.02	144	18000	4136	4136	**0.00**	1	56
_85_5_2	2043	2195	7.42	108	18000	2588	2588	**0.00**	221	11483	4264	4328	1.51	197	18000
_85_10_0	1559	1597	2.43	145	18000	2772	2772	**0.00**	11	120	4234	4234	**0.00**	386	15270
_85_10_1	1589	1613	1.51	180	18000	2634	2634	**0.00**	211	9653	3843	3926	2.15	186	18000
_85_10_2	1648	1735	5.26	126	18000	2644	2644	**0.00**	195	11243	4268	4268	**0.00**	412	10685
_85_20_0	1451	1451	**0.00**	18	124	2360	2360	**0.00**	21	221	3922	3922	**0.00**	0	50
_85_20_1	1480	1480	**0.00**	21	229	2659	2659	**0.00**	141	2463	3688	3688	**0.00**	0	21
_85_20_2	1390	1390	**0.00**	140	1469	2520	2520	**0.00**	108	1808	4033	4033	**0.00**	84	2197
_90_5_0	1761	2006	13.89	85	18000	2924	3088	5.62	172	18000	4279	4279	**0.00**	8	365
_90_5_1	1530	1530	**0.00**	7	131	2732	2732	**0.00**	0	22	4167	4167	**0.00**	105	4709
_90_5_2	1494	1494	**0.00**	45	3378	2879	2879	**0.00**	23	506	4098	4172	1.80	225	18000
_90_10_0	1740	2191	25.96	60	18000	2605	2662	2.18	162	18000	4066	4066	**0.00**	4	238
_90_10_1	1457	1457	**0.00**	114	4606	2521	2521	**0.00**	14	100	4058	4058	**0.00**	72	1288
_90_10_2	1293	1293	**0.00**	8	65	2742	2742	**0.00**	2	41	4174	4259	2.04	192	18000
_90_20_0	1460	1460	**0.00**	12	83	2439	2439	**0.00**	1	23	3991	3991	**0.00**	27	265
_90_20_1	1377	1377	**0.00**	18	103	2610	2610	**0.00**	13	223	3759	3759	**0.00**	0	20
_90_20_2	1538	1538	**0.00**	319	4723	2678	2678	**0.00**	51	735	3998	3998	**0.00**	0	46
_95_5_0	1592	1592	**0.00**	171	10456	2827	2827	**0.00**	8	266	4521	4521	**0.00**	4	266
_95_5_1	1486	1498	0.84	190	18000	2833	2833	**0.00**	25	424	4410	4410	**0.00**	20	362
_95_5_2	1452	1481	2.02	137	18000	2706	2706	**0.00**	137	7579	4682	4682	**0.00**	76	10126
_95_10_0	1590	1626	2.26	106	18000	2832	2832	**0.00**	135	6648	4508	4533	0.55	192	18000
_95_10_1	1602	1748	9.08	131	18000	3096	3096	**0.00**	14	868	4152	4173	0.50	355	18000
_95_10_2	1724	2006	16.36	75	18000	2610	2653	1.65	190	18000	4039	4039	**0.00**	17	282
_95_20_0	1555	2978	91.48	53	18000	2621	2840	8.34	191	18000	4152	4291	3.36	163	18000
_95_20_1	1603	1603	**0.00**	37	170	2660	2660	**0.00**	9	136	4202	4202	**0.00**	227	7944
_95_20_2	1232	1235	0.22	302	18000	2613	2613	**0.00**	13	202	4144	4181	0.90	237	18000
_100_5_0	1790	1857	3.73	98	18000	3293	3293	**0.00**	41	898	4291	4291	**0.00**	4	141
_100_5_1	1653	1680	1.63	122	18000	2958	3052	3.18	115	18000	4542	4542	**0.00**	0	68
_100_5_2	1859	1931	3.86	104	18000	3125	3181	1.79	170	18000	4374	4550	4.02	117	18000
_100_10_0	1655	1903	15.00	52	18000	3129	3301	5.49	99	18000	4733	4733	**0.00**	191	9395
_100_10_1	1575	3362	113.50	38	18000	2795	2878	2.97	121	18000	4300	4422	2.83	122	18000
_100_10_2	1604	1631	1.68	156	18000	2880	2880	**0.00**	38	576	4403	4403	**0.00**	2	126
_100_20_0	1592	1992	25.14	54	18000	2793	2933	5.01	427	18000	4299	4299	**0.00**	21	629
_100_20_1	1533	1533	**0.00**	182	3311	2390	2390	**0.00**	3	90	4077	4077	**0.00**	1	54
_100_20_2	1268	1274	0.46	245	18000	2638	2638	**0.00**	85	3190	4321	4321	**0.00**	48	1058

Table 11: Basic algorithm solutions to A-MDmTSP instances

Instance A-MDmTSP	I LB	I UB	I Gap	I Nod.	I time (s)	II LB	II UB	II Gap	II Nod.	II time (s)	III LB	III UB	III Gap	III Nod.	III time (s)
_50_5_0	1294	1294	**0.00**	0	0	1893	1893	**0.00**	0	1	2856	2856	**0.00**	0	2
_50_5_1	1187	1187	**0.00**	0	1	2487	2487	**0.00**	0	0	2901	2901	**0.00**	4	14
_50_5_2	1356	1356	**0.00**	3	16	1938	1938	**0.00**	0	1	3286	3286	**0.00**	4	8
_50_10_0	1681	1681	**0.00**	1	4	2680	2680	**0.00**	0	0	2595	2595	**0.00**	0	0
_50_10_1	1410	1410	**0.00**	1	3	2079	2079	**0.00**	2	6	2868	2868	**0.00**	1	1
_50_10_2	1372	1372	**0.00**	3	5	1902	1902	**0.00**	0	2	3013	3013	**0.00**	0	1
_60_5_0	1736	1736	**0.00**	27	883	2613	2613	**0.00**	10	46	3250	3250	**0.00**	188	2216
_60_5_1	1636	1636	**0.00**	65	961	2424	2424	**0.00**	3	13	3431	3431	**0.00**	1	5
_60_5_2	1624	1726	6.28	260	18000	2507	2507	**0.00**	53	1403	3863	3883	0.52	449	18000
_60_10_0	1354	1354	**0.00**	4	10	2421	2421	**0.00**	2	6	3140	3140	**0.00**	1	5
_60_10_1	1772	1772	**0.00**	10	53	2372	2372	**0.00**	0	3	3432	3432	**0.00**	2	14
_60_10_2	1873	1873	**0.00**	80	1578	2528	2528	**0.00**	1	6	3200	3200	**0.00**	36	198
_70_5_0	1896	-	-	85	18000	2568	2568	**0.00**	138	12716	3601	3601	**0.00**	31	411
_70_5_1	1428	1428	**0.00**	39	602	2661	2661	**0.00**	34	425	3477	3561	2.41	187	18000
_70_5_2	1651	1651	**0.00**	119	6286	2757	2757	**0.00**	1	7	3559	3559	**0.00**	0	2
_70_10_0	1578	1578	**0.00**	3	21	2286	2286	**0.00**	5	18	3684	3684	**0.00**	30	246
_70_10_1	1538	1538	**0.00**	32	249	2479	2479	**0.00**	4	15	3332	3332	**0.00**	7	58
_70_10_2	1415	1415	**0.00**	53	165	2555	2555	**0.00**	11	130	3819	3819	**0.00**	0	4
_75_5_0	1630	1630	**0.00**	7	52	2590	2590	**0.00**	13	279	3851	3851	**0.00**	22	306
_75_5_1	1802	1802	**0.00**	3	71	2744	2744	**0.00**	11	116	3761	3761	**0.00**	13	591
_75_5_2	1815	1840	1.36	134	18000	2420	2420	**0.00**	50	1218	4024	4024	**0.00**	240	6160
_75_10_0	1350	1350	**0.00**	40	1097	2630	2630	**0.00**	202	17265	3545	3545	**0.00**	13	168
_75_10_1	1438	-	-	88	18000	2807	2807	**0.00**	176	9565	3987	3987	**0.00**	126	1483
_75_10_2	1718	-	-	80	18000	2677	2796	4.45	194	18000	3610	3610	**0.00**	359	14642
_80_5_0	1442	2318	60.71	65	18000	2413	2413	**0.00**	99	4170	4296	4296	**0.00**	16	676
_80_5_1	1377	5641	309.56	88	18000	2610	2633	0.88	148	18000	4248	4248	**0.00**	33	1530
_80_5_2	1686	-	-	73	18000	2732	2975	8.88	112	18000	3997	5396	34.99	62	18000
_80_10_0	1344	1344	**0.00**	6	51	2525	2525	**0.00**	27	295	3721	3721	**0.00**	9	74
_80_10_1	1499	1499	**0.00**	94	4618	2379	2379	**0.00**	28	176	3953	3994	1.04	435	18000
_80_10_2	1529	2828	84.97	100	18000	2672	2672	**0.00**	7	16	3831	3831	**0.00**	3	29
_80_20_0	1352	1352	**0.00**	274	2953	2075	2075	**0.00**	11	69	3744	3744	**0.00**	2	15
_80_20_1	1321	1321	**0.00**	180	2716	2813	2813	**0.00**	53	1129	3641	3641	**0.00**	8	57
_80_20_2	1548	-	-	165	18000	2635	2635	**0.00**	400	6849	3891	3891	**0.00**	125	781
_85_5_0	1664	4112	147.05	58	18000	2686	2686	**0.00**	33	1617	4207	4252	1.07	234	18000
_85_5_1	1409	-	-	59	18000	2792	6613	136.85	50	18000	4136	4136	**0.00**	2	66
_85_5_2	1994	-	-	61	18000	2537	5508	117.15	70	18000	4207	4546	8.06	98	18000
_85_10_0	1518	7165	371.94	89	18000	2772	2772	**0.00**	24	110	4177	4280	2.47	277	18000
_85_10_1	1549	-	-	73	18000	2579	3216	24.72	87	18000	3794	4028	6.16	94	18000
_85_10_2	1609	-	-	66	18000	2628	2647	0.74	197	18000	4241	4287	1.09	289	18000
_85_20_0	1451	1451	**0.00**	207	3487	2360	2360	**0.00**	204	2122	3922	3922	**0.00**	3	26
_85_20_1	1480	1480	**0.00**	119	6689	2659	2659	**0.00**	226	5341	3688	3688	**0.00**	0	5
_85_20_2	1384	1390	0.45	572	18000	2520	2520	**0.00**	254	4765	4033	4033	**0.00**	216	11528
_90_5_0	1714	-	-	46	18000	2873	21436	646.12	62	18000	4279	4279	**0.00**	30	1712
_90_5_1	1530	1530	**0.00**	44	2599	2732	2732	**0.00**	1	21	4133	4193	1.45	160	18000
_90_5_2	1467	6521	344.42	55	18000	2879	2879	**0.00**	109	10863	4063	4369	7.54	89	18000
_90_10_0	1698	-	-	58	18000	2564	15309	497.07	65	18000	4066	4066	**0.00**	26	777
_90_10_1	1433	1489	3.91	167	18000	2521	2521	**0.00**	59	983	4058	4058	**0.00**	90	5219
_90_10_2	1286	1402	8.99	181	18000	2742	2742	**0.00**	23	434	4103	4334	5.63	96	18000
_90_20_0	1460	1460	**0.00**	63	577	2439	2439	**0.00**	50	742	3991	3991	**0.00**	64	482
_90_20_1	1377	1377	**0.00**	146	716	2610	2610	**0.00**	49	495	3759	3759	**0.00**	0	4
_90_20_2	1496	1853	23.85	160	18000	2678	2678	**0.00**	481	17006	3998	3998	**0.00**	0	11
_95_5_0	1548	1856	19.94	69	18000	2817	2951	4.76	115	18000	4521	4521	**0.00**	17	1675
_95_5_1	1443	6638	359.94	63	18000	2833	2833	**0.00**	65	4147	4410	4410	**0.00**	21	406
_95_5_2	1411	-	-	46	18000	2673	2882	7.84	80	18000	4647	4831	3.95	106	18000
_95_10_0	1561	-	-	50	18000	2790	3253	16.58	104	18000	4422	4736	7.09	61	18000
_95_10_1	1570	-	-	75	18000	3068	9418	206.94	64	18000	4124	4197	1.78	276	18000
_95_10_2	1695	-	-	55	18000	2572	-	-	64	18000	4039	4039	**0.00**	49	1913
_95_20_0	1521	-	-	55	18000	2579	-	-	96	18000	4075	5121	25.67	90	18000
_95_20_1	1603	1603	**0.00**	175	2012	2660	2660	**0.00**	25	300	4196	4202	0.14	260	18000
_95_20_2	1194	-	-	84	18000	2613	2613	**0.00**	80	1591	4098	4209	2.72	134	18000
_100_5_0	1755	14142	705.96	47	18000	3266	3594	10.04	94	18000	4291	4291	**0.00**	13	496
_100_5_1	1623	6316	289.16	55	18000	2924	5091	74.11	41	18000	4542	4542	**0.00**	2	143
_100_5_2	1817	-	-	46	18000	3086	3964	28.44	51	18000	4306	4841	12.41	72	18000
_100_10_0	1627	-	-	44	18000	3083	9222	199.11	51	18000	4678	4826	3.17	125	18000
_100_10_1	1541	-	-	48	18000	2750	-	-	47	18000	4225	4617	9.28	92	18000
_100_10_2	1578	12140	669.55	59	18000	2863	2965	3.55	119	18000	4403	4403	**0.00**	9	106
_100_20_0	1560	-	-	43	18000	2737	-	-	122	18000	4299	4299	**0.00**	55	1426
_100_20_1	1507	3468	130.19	121	18000	2390	2390	**0.00**	96	5809	4077	4077	**0.00**	3	35
_100_20_2	1241	-	-	89	18000	2629	2649	0.76	300	18000	4321	4321	**0.00**	121	2999

Table 12: Branch and cut framework solutions to A-MDCVRP instances

Instance A-MDCVRP	I					II					III				
	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)	LB	UB	Gap	Nodes	time (s)
_40_5_0	1548	1548	**0.00**	1	11	2248	2248	**0.00**	0	1	2838	2838	**0.00**	3	11
_40_5_1	1700	1700	**0.00**	2	21	2235	2235	**0.00**	1	11	2890	2890	**0.00**	0	7
_40_5_2	1549	1549	**0.00**	0	4	2219	2219	**0.00**	4	14	3035	3035	**0.00**	1	13
_40_10_0	1516	1516	**0.00**	0	5	2595	2595	**0.00**	6	25	2558	2558	**0.00**	0	1
_40_10_1	1420	1420	**0.00**	0	5	1776	1776	**0.00**	0	3	2736	2736	**0.00**	0	5
_40_10_2	1560	1560	**0.00**	19	86	2180	2180	**0.00**	9	46	2418	2418	**0.00**	43	569
_45_5_0	1707	1707	**0.00**	0	9	2164	2164	**0.00**	2	12	3264	3264	**0.00**	0	9
_45_5_1	1760	1760	**0.00**	2	8	2488	2488	**0.00**	0	3	3080	3080	**0.00**	119	807
_45_5_2	1427	1427	**0.00**	16	59	2340	2340	**0.00**	331	5021	3456	3456	**0.00**	7	105
_45_10_0	1376	1376	**0.00**	1	11	1817	1817	**0.00**	3	15	2779	2779	**0.00**	300	2188
_45_10_1	1640	1640	**0.00**	19	63	2373	2486	4.75	432	18000	3010	3010	**0.00**	6	52
_45_10_2	1293	1293	**0.00**	0	4	2230	2230	**0.00**	2	16	2920	2920	**0.00**	0	7
_50_5_0	1568	1568	**0.00**	73	338	2224	2331	4.81	463	18000	3692	3692	**0.00**	847	14370
_50_5_1	1370	1370	**0.00**	58	502	2802	2802	**0.00**	274	3190	3291	3305	0.44	701	18000
_50_5_2	1381	1381	**0.00**	3	24	2036	2036	**0.00**	7	21	3378	3378	**0.00**	1	22
_50_10_0	1682	1682	**0.00**	0	6	2699	2699	**0.00**	4	41	2623	2623	**0.00**	0	5
_50_10_1	1498	1498	**0.00**	66	530	2252	2370	5.23	332	18000	3142	3142	**0.00**	0	12
_50_10_2	1372	1372	**0.00**	2	20	1948	1948	**0.00**	7	44	2949	2949	**0.00**	0	4
_55_5_0	1627	1627	**0.00**	45	509	2387	2387	**0.00**	127	1613	3310	3413	3.10	388	18000
_55_5_1	2179	2314	6.20	226	18000	2962	3199	8.01	223	18000	4369	4369	**0.00**	147	5531
_55_5_2	1735	1735	**0.00**	37	229	2880	2880	**0.00**	48	398	3706	3737	0.83	312	18000
_55_10_0	1465	1465	**0.00**	128	2368	2323	2472	6.43	379	18000	3780	3838	1.54	348	18000
_55_10_1	1556	1556	**0.00**	31	245	2556	2556	**0.00**	124	1861	3427	3427	**0.00**	0	27
_55_10_2	1323	1323	**0.00**	12	36	2488	2488	**0.00**	1	11	2882	2882	**0.00**	0	14
_60_5_0	1935	2063	6.59	217	18000	2913	3129	7.40	202	18000	3488	3488	**0.00**	99	902
_60_5_1	1656	1656	**0.00**	65	971	2516	2516	**0.00**	205	4777	3656	3656	**0.00**	0	26
_60_5_2	1561	1561	**0.00**	8	64	2535	2585	1.98	269	18000	3767	3767	**0.00**	0	41
_60_10_0	1481	1495	0.95	505	18000	2526	2690	6.51	225	18000	3395	3395	**0.00**	1	16
_60_10_1	1817	1817	**0.00**	57	550	2338	2338	**0.00**	9	51	3737	3737	**0.00**	11	143
_60_10_2	2024	2159	6.67	202	18000	2779	3647	31.23	134	18000	3482	3482	**0.00**	8	169
_65_5_0	1483	1483	**0.00**	5	59	2483	2483	**0.00**	48	386	3936	3936	**0.00**	57	851
_65_5_1	1831	1913	4.45	173	18000	2924	3196	9.29	175	18000	4077	4219	3.48	257	18000
_65_5_2	1628	1628	**0.00**	121	2658	2636	2636	**0.00**	212	8915	3826	3826	**0.00**	11	110
_65_10_0	1518	1557	2.58	193	18000	2151	2226	3.47	220	18000	3697	3730	0.89	717	18000
_65_10_1	1314	1314	**0.00**	91	2177	2114	2171	2.70	278	18000	3640	3657	0.47	313	18000
_65_10_2	1720	1914	11.30	181	18000	2640	2945	11.57	186	18000	3474	3474	**0.00**	254	7237
_70_5_0	1955	2011	2.85	233	18000	2731	2975	8.95	259	18000	3928	3928	**0.00**	96	2965
_70_5_1	1492	1492	**0.00**	51	973	2873	2882	0.30	330	18000	3853	3853	**0.00**	1	62
_70_10_0	1644	1644	**0.00**	64	1643	2323	2392	2.99	233	18000	3916	3979	1.61	535	18000
_70_10_1	1668	2189	31.26	117	18000	2793	3571	27.85	95	18000	3977	4026	1.24	249	18000
_70_10_2	1464	1487	1.58	550	18000	2609	2770	6.16	155	18000	4157	4157	**0.00**	3	231
_70_5_2	1917	2080	8.52	134	18000	3128	3341	6.81	192	18000	4200	4200	**0.00**	0	54
_75_5_0	1672	1672	**0.00**	18	139	2753	2753	**0.00**	155	8426	4048	4134	2.11	189	18000
_75_5_1	2013	2112	4.92	177	18000	3146	3373	7.23	128	18000	4341	4529	4.33	107	18000
_75_5_2	1857	1857	**0.00**	146	9111	2499	2544	1.82	211	18000	4482	4482	**0.00**	271	15783
_75_10_0	1350	1350	**0.00**	5	55	2672	2733	2.29	207	18000	3578	3578	**0.00**	4	156
_75_10_1	1570	1745	11.13	126	18000	2902	3500	20.60	99	18000	4531	4531	**0.00**	15	549
_75_10_2	1765	1795	1.68	217	18000	2716	2873	5.78	197	18000	3693	3762	1.87	213	18000
_75_20_0	1495	1678	12.27	110	18000	2604	2718	4.37	125	18000	3987	4024	0.93	206	18000
_75_20_1	1670	1695	1.52	257	18000	2645	2740	3.58	224	18000	3462	3462	**0.00**	2	48
_75_20_2	1435	1624	13.14	167	18000	2495	3045	22.07	113	18000	4077	4077	**0.00**	64	2844
_80_10_0	1366	1366	**0.00**	44	449	2536	2799	10.37	176	18000	3928	3928	**0.00**	29	831
_80_10_1	1629	2040	25.27	86	18000	2506	2733	9.05	111	18000	4367	4428	1.40	277	18000
_80_10_2	1583	1602	1.22	154	18000	2669	2669	**0.00**	43	1166	4051	4051	**0.00**	139	3263
_80_20_0	1358	1358	**0.00**	314	4337	2065	2139	3.59	137	18000	3875	3875	**0.00**	0	24
_80_20_1	1370	1592	16.24	137	18000	2891	3550	22.81	72	18000	3833	3833	**0.00**	41	843
_80_20_2	1585	1623	2.41	338	18000	2563	2944	14.87	128	18000	3974	4016	1.07	284	18000
_85_10_0	1662	2868	72.60	56	18000	2792	3529	26.41	55	18000	5199	5314	2.22	93	18000
_85_10_1	1594	1676	5.18	149	18000	2773	3245	17.03	78	18000	4064	4064	**0.00**	26	1946
_85_10_2	1643	1667	1.45	206	18000	2637	2698	2.30	149	18000	4495	4501	0.13	342	18000
_85_20_0	1479	2458	66.22	63	18000	2387	2907	21.80	52	18000	4353	4485	3.04	127	18000
_85_20_1	1528	1728	13.09	106	18000	2613	3292	25.96	60	18000	4048	4086	0.93	288	18000
_85_20_2	1375	1375	**0.00**	112	995	2431	2650	8.99	107	18000	3968	3968	**0.00**	180	16810

Table 13: Basic algorithm solutions to A-MDCVRP instances

Instance	I					II					III				
A-MDCVRP	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)
_40_5_0	1548	1548	**0.00**	1	2	2248	2248	**0.00**	0	0	2838	2838	**0.00**	2	2
_40_5_1	1700	1700	**0.00**	15	71	2235	2235	**0.00**	2	8	2890	2890	**0.00**	1	5
_40_5_2	1549	1549	**0.00**	1	1	2219	2219	**0.00**	10	23	3035	3035	**0.00**	4	6
_40_10_0	1516	1516	**0.00**	1	1	2595	2595	**0.00**	14	25	2558	2558	**0.00**	0	1
_40_10_1	1420	1420	**0.00**	0	0	1776	1776	**0.00**	1	1	2736	2736	**0.00**	1	3
_40_10_2	1560	1560	**0.00**	52	284	2180	2180	**0.00**	12	23	2418	2418	**0.00**	56	340
_45_5_0	1707	1707	**0.00**	2	7	2164	2164	**0.00**	9	30	3264	3264	**0.00**	0	3
_45_5_1	1760	1760	**0.00**	9	40	2488	2488	**0.00**	2	6	3080	3080	**0.00**	77	349
_45_5_2	1427	1427	**0.00**	93	1320	2297	2370	3.19	373	18000	3456	3456	**0.00**	24	289
_45_10_0	1376	1376	**0.00**	4	12	1817	1817	**0.00**	3	15	2779	2779	**0.00**	313	2512
_45_10_1	1640	1640	**0.00**	167	5382	2273	2956	30.04	240	18000	3010	3010	**0.00**	13	49
_45_10_2	1293	1293	**0.00**	0	2	2230	2230	**0.00**	7	12	2920	2920	**0.00**	0	1
_50_5_0	1568	1568	**0.00**	213	9773	2113	3514	66.33	163	18000	3692	3692	**0.00**	731	12403
_50_5_1	1370	1370	**0.00**	170	5597	2779	2806	0.97	398	18000	3225	3464	7.40	317	18000
_50_5_2	1381	1381	**0.00**	6	55	2036	2036	**0.00**	13	110	3378	3378	**0.00**	2	24
_50_10_0	1682	1682	**0.00**	3	19	2699	2699	**0.00**	12	73	2623	2623	**0.00**	0	1
_50_10_1	1498	1498	**0.00**	287	7063	2164	2661	22.98	197	18000	3142	3142	**0.00**	1	4
_50_10_2	1372	1372	**0.00**	6	14	1948	1948	**0.00**	33	254	2949	2949	**0.00**	0	1
_55_5_0	1627	1627	**0.00**	107	3069	2336	2508	7.36	191	18000	3284	3426	4.32	363	18000
_55_5_1	2089	6492	210.83	134	18000	2848	-	-	104	18000	4369	4369	**0.00**	511	15780
_55_5_2	1735	1735	**0.00**	93	2813	2880	2880	**0.00**	229	8640	3679	3774	2.59	246	18000
_55_10_0	1465	1465	**0.00**	249	13058	2247	2743	22.09	183	18000	3732	3872	3.75	298	18000
_55_10_1	1556	1556	**0.00**	78	1316	2556	2556	**0.00**	282	10507	3427	3427	**0.00**	2	28
_55_10_2	1323	1323	**0.00**	14	51	2488	2488	**0.00**	9	17	2882	2882	**0.00**	3	25
_60_5_0	1851	7559	308.40	115	18000	2788	6544	134.72	95	18000	3488	3488	**0.00**	88	1005
_60_5_1	1656	1656	**0.00**	151	7700	2464	2727	10.66	153	18000	3656	3656	**0.00**	1	20
_60_5_2	1561	1561	**0.00**	68	1448	2479	2824	13.93	169	18000	3767	3767	**0.00**	9	180
_60_10_0	1439	1489	3.51	258	18000	2452	3867	57.74	129	18000	3395	3395	**0.00**	4	14
_60_10_1	1791	2684	49.88	172	18000	2338	2338	**0.00**	48	819	3737	3737	**0.00**	20	185
_60_10_2	1933	6059	213.53	139	18000	2688	-	-	110	18000	3482	3482	**0.00**	23	197
_65_5_0	1483	1483	**0.00**	46	1698	2483	2483	**0.00**	98	3345	3936	3936	**0.00**	68	2729
_65_5_1	1746	9186	426.17	90	18000	2841	5778	103.36	89	18000	4037	4260	5.53	283	18000
_65_5_2	1605	1655	3.15	204	18000	2574	3168	23.09	131	18000	3826	3826	**0.00**	35	608
_65_10_0	1460	5511	277.56	92	18000	2085	3575	71.42	99	18000	3681	3744	1.73	916	18000
_65_10_1	1314	1314	**0.00**	248	13526	2062	2234	8.36	169	18000	3634	3658	0.67	352	18000
_65_10_2	1661	-	-	112	18000	2564	-	-	88	18000	3474	3474	**0.00**	321	7050
_70_5_0	1894	2819	48.84	104	18000	2647	7138	169.67	88	18000	3928	3928	**0.00**	81	3762
_70_5_1	1461	2698	84.72	115	18000	2794	3875	38.68	110	18000	3853	3853	**0.00**	1	28
_70_5_2	1854	-	-	87	18000	3059	-	-	82	18000	4200	4200	**0.00**	2	41
_70_10_0	1644	1644	**0.00**	174	11553	2264	2483	9.68	122	18000	3881	4106	5.79	350	18000
_70_10_1	1601	-	-	94	18000	2710	-	-	86	18000	3936	4078	3.60	226	18000
_70_10_2	1407	5452	287.44	173	18000	2546	19978	684.68	92	18000	4157	4157	**0.00**	19	254
_75_5_0	1672	1672	**0.00**	57	2489	2693	4906	82.20	84	18000	3982	4393	10.32	112	18000
_75_5_1	1963	5544	182.47	97	18000	3054	-	-	73	18000	4254	-	-	86	18000
_75_5_2	1815	4383	141.49	91	18000	2449	16598	577.88	77	18000	4402	4550	3.37	148	18000
_75_10_0	1350	1350	**0.00**	32	952	2611	6210	137.87	91	18000	3578	3578	**0.00**	4	58
_75_10_1	1502	-	-	83	18000	2830	-	-	80	18000	4531	4531	**0.00**	61	803
_75_10_2	1715	5683	231.31	94	18000	2659	-	-	77	18000	3626	3878	6.95	190	18000
_75_20_0	1460	2954	102.32	94	18000	2553	7553	195.89	99	18000	3949	4077	3.24	275	18000
_75_20_1	1627	3196	96.45	126	18000	2598	-	-	96	18000	3462	3462	**0.00**	6	56
_75_20_2	1388	-	-	98	18000	2441	-	-	78	18000	4077	4077	**0.00**	146	4666
_80_10_0	1366	1366	**0.00**	80	3761	2488	-	-	78	18000	3928	3928	**0.00**	126	3628
_80_10_1	1579	-	-	63	18000	2442	6913	183.04	68	18000	4316	4508	4.44	175	18000
_80_10_2	1538	-	-	77	18000	2639	3128	18.53	104	18000	4051	4051	**0.00**	196	4438
_80_20_0	1334	1381	3.49	341	18000	2023	-	-	75	18000	3875	3875	**0.00**	2	30
_80_20_1	1319	-	-	104	18000	2840	-	-	67	18000	3833	3833	**0.00**	205	4532
_80_20_2	1549	6904	345.63	123	18000	2521	-	-	92	18000	3955	3988	0.84	488	18000
_85_10_0	1614	-	-	66	18000	2734	-	-	54	18000	5091	6701	31.63	89	18000
_85_10_1	1561	-	-	80	18000	2712	-	-	59	18000	4064	4064	**0.00**	92	16638
_85_10_2	1605	3439	114.21	99	18000	2598	2938	13.10	99	18000	4453	4516	1.42	269	18000
_85_20_0	1435	-	-	66	18000	2325	-	-	51	18000	4287	4494	4.84	140	18000
_85_20_1	1491	-	-	73	18000	2572	-	-	60	18000	4012	4073	1.52	247	18000
_85_20_2	1365	1386	1.50	499	18000	2387	-	-	76	18000	3939	3999	1.53	205	18000

Table 14: Branch and cut framework solutions to A-CLRP instances

Instance A-CLRP	I					II					III				
	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)
_40_5_0	2281	2281	**0.00**	0	5	4222	4222	**0.00**	0	4	4297	4297	**0.00**	0	27
_40_5_1	3048	3048	**0.00**	66	907	3836	3836	**0.00**	8	49	4426	4426	**0.00**	0	10
_40_5_2	2831	2831	**0.00**	0	8	2963	2963	**0.00**	0	6	4110	4110	**0.00**	0	7
_40_10_0	2788	2788	**0.00**	0	4	4064	4064	**0.00**	0	2	4144	4144	**0.00**	15	47
_40_10_1	2256	2256	**0.00**	0	3	2722	2722	**0.00**	0	4	3963	3963	**0.00**	1	20
_40_10_2	2854	2854	**0.00**	2	21	3617	3617	**0.00**	7	50	3981	3981	**0.00**	122	2147
_45_5_0	2769	2769	**0.00**	0	9	3932	3932	**0.00**	13	80	4273	4273	**0.00**	0	14
_45_5_1	2613	2613	**0.00**	0	2	4757	4757	**0.00**	0	10	4533	4533	**0.00**	1	32
_45_5_2	2581	2581	**0.00**	1	17	4622	4622	**0.00**	81	1277	5052	5052	**0.00**	104	3538
_45_10_0	2451	2451	**0.00**	0	7	3135	3135	**0.00**	0	8	4724	4724	**0.00**	100	1733
_45_10_1	3464	3464	**0.00**	9	116	4856	4856	**0.00**	62	444	5355	5551	3.65	503	18000
_45_10_2	2643	2643	**0.00**	4	30	3761	3761	**0.00**	1	9	4943	4943	**0.00**	236	3118
_50_5_0	3331	3331	**0.00**	5	61	4653	4789	2.91	435	18000	6950	7065	1.66	192	18000
_50_5_1	2524	2524	**0.00**	4	38	4718	4718	**0.00**	6	41	4623	4623	**0.00**	414	11049
_50_5_2	3249	3249	**0.00**	1	22	3688	3688	**0.00**	2	24	4641	4641	**0.00**	0	14
_50_10_0	2548	2548	**0.00**	0	3	3945	3945	**0.00**	0	3	4256	4256	**0.00**	79	1412
_50_10_1	2729	2729	**0.00**	0	7	4623	4623	**0.00**	55	625	5596	5596	**0.00**	29	834
_50_10_2	2520	2520	**0.00**	227	6622	3255	3255	**0.00**	4	50	4589	4662	1.60	339	18000
_55_5_0	2693	2693	**0.00**	1	27	4369	4369	**0.00**	9	73	5231	5231	**0.00**	3	90
_55_5_1	3855	3855	**0.00**	150	7796	5408	5408	**0.00**	124	1577	6953	7020	0.97	145	18000
_55_5_2	3091	3091	**0.00**	0	9	4436	4436	**0.00**	26	239	5953	6040	1.47	205	18000
_55_10_0	3679	3679	**0.00**	295	14822	4158	4280	2.95	269	18000	6020	6169	2.47	149	18000
_55_10_1	2731	2731	**0.00**	3	46	5219	5219	**0.00**	77	1852	5376	5376	**0.00**	44	552
_55_10_2	2297	2297	**0.00**	5	48	3537	3537	**0.00**	8	69	4538	4538	**0.00**	66	1781
_60_5_0	3773	3965	5.10	200	18000	5054	5124	1.39	278	18000	6409	6409	**0.00**	36	907
_60_5_1	3193	3193	**0.00**	274	16648	4500	4557	1.27	200	18000	6555	6725	2.59	554	18000
_60_5_2	3467	3467	**0.00**	0	14	3625	3625	**0.00**	7	222	5663	5663	**0.00**	3	139
_60_10_0	3002	3002	**0.00**	5	118	4654	4681	0.59	314	18000	5416	5678	4.84	322	18000
_60_10_1	3046	3046	**0.00**	202	9713	3671	3671	**0.00**	13	359	5096	5096	**0.00**	16	1150
_60_10_2	3955	3974	0.47	147	18000	5318	5747	8.07	163	18000	5637	5836	3.54	253	18000
_65_5_0	2850	2850	**0.00**	17	210	3564	3564	**0.00**	4	190	6186	6382	3.17	142	18000
_65_5_1	3061	3061	**0.00**	97	5943	4952	5025	1.47	263	18000	6262	6262	**0.00**	73	1963
_65_5_2	3670	3670	**0.00**	52	1264	4866	4866	**0.00**	4	108	5275	5275	**0.00**	402	7870
_65_10_0	2756	2822	2.39	210	18000	4089	4089	**0.00**	1	55	6198	6975	12.53	150	18000
_65_10_1	2811	2811	**0.00**	2	74	3163	3163	**0.00**	116	3968	5430	5430	**0.00**	22	1495
_65_10_2	2935	3039	3.54	178	18000	5276	5410	2.54	242	18000	5974	6593	10.36	210	18000
_70_5_0	3487	3487	**0.00**	118	8150	5104	5265	3.16	144	18000	6278	6355	1.23	181	18000
_70_5_1	2598	2598	**0.00**	8	169	5132	5132	**0.00**	16	434	5066	5066	**0.00**	1	46
_70_5_2	3735	3782	1.27	134	18000	6487	6910	6.53	90	18000	6445	6800	5.51	181	18000
_70_10_0	3107	3107	**0.00**	120	8127	4157	4157	**0.00**	28	2070	5783	6041	4.46	172	18000
_70_10_1	3773	4125	9.32	77	18000	5633	6668	18.37	39	18000	6063	6646	9.61	186	18000
_70_10_2	2836	2836	**0.00**	36	1237	4219	4318	2.35	202	18000	6355	6594	3.76	161	18000
_75_5_0	2534	2534	**0.00**	0	19	4754	4754	**0.00**	20	854	6061	6421	5.95	199	18000
_75_5_1	3698	3722	0.64	170	18000	5398	5518	2.22	131	18000	6383	6829	6.99	82	18000
_75_5_2	2750	2750	**0.00**	2	56	3778	3778	**0.00**	3	159	6519	6604	1.30	228	18000
_75_10_0	2970	2970	**0.00**	1	60	3825	3825	**0.00**	7	391	5132	5430	5.80	164	18000
_75_10_1	3502	3625	3.53	63	18000	5083	5464	7.49	110	18000	7106	7866	10.70	102	18000
_75_10_2	2818	2818	**0.00**	19	1422	3912	3971	1.50	184	18000	5200	5709	9.78	97	18000
_75_20_0	2984	2995	0.36	186	18000	4215	4567	8.36	45	18000	5815	6260	7.65	172	18000
_75_20_1	2570	2570	**0.00**	0	38	4117	4117	**0.00**	223	16092	5332	5527	3.66	203	18000
_75_20_2	3766	3815	1.31	66	18000	4894	5656	15.58	41	18000	6230	7353	18.02	87	18000
_80_10_0	2410	2410	**0.00**	1	42	4447	4447	**0.00**	15	1124	6010	6355	5.73	213	18000
_80_10_1	3563	3641	2.20	80	18000	4001	4421	10.51	83	18000	6477	7248	11.90	49	18000
_80_10_2	2897	2981	2.91	127	18000	4286	4436	3.50	80	18000	6729	7458	10.84	119	18000
_80_20_0	2327	2334	0.30	213	18000	3736	3891	4.14	122	18000	5640	6150	9.04	142	18000
_80_20_1	3045	3291	8.07	86	18000	5296	5928	11.94	20	18000	6045	7090	17.28	85	18000
_80_20_2	3016	3203	6.20	59	18000	4559	4835	6.06	138	18000	5863	6488	10.65	113	18000
_85_10_0	3408	3744	9.87	56	18000	6181	6640	7.42	36	18000	6740	7631	13.23	32	18000
_85_10_1	3289	3330	1.24	103	18000	4884	4982	2.01	150	18000	5813	6215	6.92	47	18000
_85_10_2	3094	3196	3.30	98	18000	4002	4184	4.54	104	18000	6486	6972	7.50	108	18000
_85_20_0	3822	4924	28.84	20	18000	5122	6439	25.70	22	18000	6774	7643	12.82	59	18000
_85_20_1	3227	3358	4.06	91	18000	4337	4459	2.82	154	18000	5594	6189	10.63	98	18000
_85_20_2	2580	2580	**0.00**	6	217	4330	4535	4.74	77	18000	5642	6139	8.80	81	18000

Table 15: Basic algorithm solutions to A-CLRP instances

Instance	I					II					III				
A-CLRP	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)	LB	UB	Gap	Nod.	time (s)
_40_5_0	2281	2281	**0.00**	1	7	4222	4222	**0.00**	0	3	4297	4297	**0.00**	6	26
_40_5_1	3048	3048	**0.00**	148	5767	3836	3836	**0.00**	18	63	4426	4426	**0.00**	2	5
_40_5_2	2831	2831	**0.00**	1	4	2963	2963	**0.00**	0	1	4110	4110	**0.00**	0	2
_40_10_0	2788	2788	**0.00**	0	1	4064	4064	**0.00**	0	1	4144	4144	**0.00**	12	26
_40_10_1	2256	2256	**0.00**	0	1	2722	2722	**0.00**	0	2	3963	3963	**0.00**	8	43
_40_10_2	2854	2854	**0.00**	14	178	3617	3617	**0.00**	19	345	3981	3981	**0.00**	214	1949
_45_5_0	2769	2769	**0.00**	1	16	3932	3932	**0.00**	17	177	4273	4273	**0.00**	0	6
_45_5_1	2613	2613	**0.00**	0	1	4757	4757	**0.00**	0	3	4533	4533	**0.00**	9	40
_45_5_2	2581	2581	**0.00**	12	201	4622	4622	**0.00**	140	3613	4996	5065	1.38	166	18000
_45_10_0	2451	2451	**0.00**	0	4	3135	3135	**0.00**	4	23	4724	4724	**0.00**	303	5541
_45_10_1	3464	3464	**0.00**	37	875	4856	4856	**0.00**	99	2676	5082	5658	11.33	549	18000
_45_10_2	2643	2643	**0.00**	24	540	3761	3761	**0.00**	2	11	4943	4943	**0.00**	438	4061
_50_5_0	3331	3331	**0.00**	24	567	4375	7962	82.01	150	18000	6765	7120	5.25	263	18000
_50_5_1	2524	2524	**0.00**	17	272	4718	4718	**0.00**	19	406	4623	4623	**0.00**	77	1064
_50_5_2	3249	3249	**0.00**	3	66	3688	3688	**0.00**	7	148	4641	4641	**0.00**	0	14
_50_10_0	2548	2548	**0.00**	0	2	3945	3945	**0.00**	0	11	4256	4256	**0.00**	156	1951
_50_10_1	2729	2729	**0.00**	0	3	4623	4623	**0.00**	148	5007	5596	5596	**0.00**	220	5153
_50_10_2	2520	2520	**0.00**	249	12486	3255	3255	**0.00**	19	335	4529	4879	7.72	242	18000
_55_5_0	2693	2693	**0.00**	15	538	4369	4369	**0.00**	15	155	5231	5231	**0.00**	19	473
_55_5_1	3767	4297	14.08	127	18000	5275	7581	43.70	113	18000	6868	7173	4.44	163	18000
_55_5_2	3091	3091	**0.00**	3	162	4436	4436	**0.00**	73	2918	5922	6019	1.64	258	18000
_55_10_0	3568	4382	22.80	129	18000	4037	5368	32.98	138	18000	5854	8283	41.50	184	18000
_55_10_1	2731	2731	**0.00**	41	1389	5219	5219	**0.00**	114	2306	5376	5376	**0.00**	98	1947
_55_10_2	2298	2298	**0.00**	15	188	3537	3537	**0.00**	10	78	4538	4538	**0.00**	89	3286
_60_5_0	3682	4714	28.03	110	18000	4842	9199	89.97	89	18000	6409	6409	**0.00**	182	6885
_60_5_1	3122	3331	6.69	126	18000	4429	5029	13.56	129	18000	6578	6773	2.97	582	18000
_60_5_2	3467	3467	**0.00**	4	169	3625	3625	**0.00**	40	1719	5663	5663	**0.00**	31	344
_60_10_0	3002	3002	**0.00**	53	3191	4330	6977	61.13	91	18000	5444	5752	5.66	385	18000
_60_10_1	2992	3259	8.92	179	18000	3671	3671	**0.00**	44	2745	5096	5096	**0.00**	93	6321
_60_10_2	3848	6403	66.41	88	18000	4963	6208	25.09	84	18000	5336	7565	41.77	156	18000
_65_5_0	2850	2850	**0.00**	68	4776	3564	3564	**0.00**	21	923	6186	6484	4.82	123	18000
_65_5_1	3009	3273	8.77	112	18000	4727	-	-	79	18000	6262	6262	**0.00**	150	6551
_65_5_2	3670	3670	**0.00**	117	13577	4867	4867	**0.00**	23	1036	5275	5275	**0.00**	230	4167
_65_10_0	2715	2849	4.94	158	18000	4089	4089	**0.00**	8	400	5774	7276	26.00	151	18000
_65_10_1	2811	2811	**0.00**	18	784	3088	3622	17.29	108	18000	5430	5430	**0.00**	83	6395
_65_10_2	2843	4859	70.88	87	18000	4901	11607	136.82	85	18000	5296	7044	33.01	194	18000
_70_5_0	3408	4065	19.29	89	18000	4628	-	-	61	18000	6053	6650	9.86	107	18000
_70_5_1	2598	2598	**0.00**	40	2780	5132	5132	**0.00**	51	2329	5066	5066	**0.00**	4	80
_70_5_2	3638	-	-	73	18000	6002	-	-	48	18000	5961	7717	29.46	104	18000
_70_10_0	3090	3119	0.95	160	18000	4052	5966	47.23	69	18000	5618	6527	16.17	105	18000
_70_10_1	3670	-	-	55	18000	5207	-	-	49	18000	5660	-	-	126	18000
_70_10_2	2836	2836	**0.00**	93	9162	3991	-	-	60	18000	6090	7543	23.85	136	18000
_75_5_0	2534	2534	**0.00**	10	405	4754	4754	**0.00**	56	5645	5446	6860	25.96	122	18000
_75_5_1	3635	4965	36.59	83	18000	5349	5554	3.83	121	18000	5974	8567	43.40	74	18000
_75_5_2	2750	2750	**0.00**	14	339	3778	3778	**0.00**	11	1043	5847	8244	41.01	101	18000
_75_10_0	2970	2970	**0.00**	3	136	3810	4148	8.88	103	18000	4909	5523	12.51	112	18000
_75_10_1	3426	-	-	65	18000	4725	-	-	50	18000	6593	9309	41.19	102	18000
_75_10_2	2818	2818	**0.00**	77	13549	3840	5555	44.67	79	18000	4896	5847	19.42	102	18000
_75_20_0	2924	4843	65.61	50	18000	4009	-	-	39	18000	5470	-	-	83	18000
_75_20_1	2570	2570	**0.00**	16	749	4098	4133	0.86	185	18000	5059	5742	13.51	215	18000
_75_20_2	3631	-	-	53	18000	4514	-	-	50	18000	5843	-	-	98	18000
_80_10_0	2410	2410	**0.00**	5	379	4440	4465	0.57	89	18000	5233	12652	141.75	101	18000
_80_10_1	3504	4479	27.83	59	18000	3854	5434	41.00	60	18000	5983	-	-	83	18000
_80_10_2	2860	3053	6.74	117	18000	4141	7753	87.21	46	18000	5998	-	-	80	18000
_80_20_0	2274	3853	69.43	64	18000	3586	-	-	42	18000	5346	6758	26.42	149	18000
_80_20_1	2962	-	-	57	18000	4955	-	-	33	18000	5672	-	-	101	18000
_80_20_2	2953	6250	111.68	39	18000	4309	6528	51.49	40	18000	5343	-	-	70	18000
_85_10_0	3341	-	-	48	18000	5530	-	-	31	18000	6402	-	-	35	18000
_85_10_1	3238	5516	70.33	62	18000	4370	-	-	43	18000	5515	-	-	78	18000
_85_10_2	3041	3987	31.13	45	18000	3951	4809	21.73	52	18000	5931	8782	48.06	101	18000
_85_20_0	3752	-	-	39	18000	4569	-	-	34	18000	6219	-	-	56	18000
_85_20_1	3135	-	-	36	18000	4219	8495	101.35	49	18000	5226	-	-	62	18000
_85_20_2	2570	2699	5.04	91	18000	3906	-	-	35	18000	5412	11256	107.98	69	18000