The Multi-period Multi-trip Container Drayage Problem with Release and Due Dates

M. Bruglieri[a,*], S. Mancini[b,d], R. Peruzzini[c], O. Pisacane[c]

[a]*Dipartimento di Design, Politecnico di Milano, Italy*
[b]*Department of Operations, Energy, and Environmental Management, University of Klagenfurt, Austria*
[c]*Dipartimento di Ingegneria dell'Informazione, Universitá Politecnica delle Marche, Italy*
[d]*Dipartimento di Matematica e Informatica, Universitá di Cagliari, Italy*

Abstract

The Container Drayage Problem (CDP) aims at routing a fleet of trucks, based at a common terminal, to serve customers while minimizing the total travel distance. Each trip starts from and ends at the terminal, and handles a subset of customers. Each customer requires either that a container is picked up or delivered. We introduce a more realistic variant, i.e., the Multi-trip Multi-period CDP with Release and Due Dates (MM-CDP-RDD), in which the planning horizon is composed of several periods (days). On each day, each truck may perform more than one trip respecting the Release and Due Dates (RDD) associated with customer services, corresponding to the first and the last day on which the service can be carried out, respectively. Drivers' contracts impose limitations on the maximum driving time allowed on each day, on two consecutive days and on the whole weekly planning horizon. To model the MM-CDP-RDD, we propose both an Arc-based Integer Linear Programming (ILP) formulation and a Trip-based ILP formulation that receives as input all the feasible non-dominated trips. To efficiently address medium/large-sized instances of the problem, we also design six Combinatorial Beneders' Cuts approaches. All the methods are compared on a rich set of instances generated for this new problem.

Keywords: Routing, Multi-trip Vehicle Routing, Multi-period Vehicle Routing, Combinatorial Benders' Cuts

1. Introduction

In the Port Logistics sector, the term *Container Drayage* refers to the goods transportation between terminals (e.g., sea ports, intermodal terminals, inland ports, border points) and customers, where containers are used

[*]Corresponding author: maurizio.bruglieri@polimi.it

as bins. The drayage operating cost contributes significantly to the total door-to-door container transportation cost (Figure 1). For a 500-mile haul, it represents about 42% of the total door-to-door cost [44]. This leads to the need to properly route the trucks used for the drayage operations.

The *Container Drayage Problem* (CDP) aims at efficiently routing a fleet of trucks, based at a common terminal, in order to serve geographically distributed customers, while minimizing the total travel distance. In a *trip*, a truck starts from the terminal, serves a subset of customers and returns to the terminal. The customers are distinguished into two categories: *import* customers who require a container delivery and *export* customers who instead require a container pickup. Therefore, the drayage activities mainly concern the planning of truck trips to move full/empty containers between the terminal and the import and the export customers [23].

According to the *International Standard Organization*, several container sizes are permitted (e.g., 10 feet (ft), 20 ft, 40 ft, 45 ft, 48 ft, 53 ft), although the most commonly used ones are TEUs (20 ft equivalent unit) and FEUs (40 ft equivalent unit) ([21], [46],[50],[37]) and the truck capacity is usually equal to 40 ft. This means that each truck can transport either one 40 ft sized container or two 20 ft sized containers, simultaneously. Due to these loading restrictions, the maximum number of customers that can be served in a single trip is equal to 4. Figure 2 shows a feasible CDP solution in which 7 trips are performed to serve both export and import customers identified by a positive and negative demand, respectively. For instance, a -40 ft demand means that the customer requires receiving a 40 ft sized container.

Since the inland destinations are usually not far from the terminal, short-duration trips are very frequent. For this reason, in the literature, the multi-trip variant of the CDP has been introduced, where each truck is assumed to perform more than one trip during a working day [28, 35, 9].

In this paper, we introduce a new variant of the CDP where, in addition to the multi-trip assumption, we address the need of both serving the customers only in specific periods (e.g., days) and respecting the contractual restrictions imposed on the drivers' working time. For this purpose, we divide the whole planning horizon into discrete time periods (e.g., days) and each customer can be served only in specific consecutive periods e.g., Release and Due Dates (RDD). Simultaneously, we address the problem of assigning containers to the trucks and then designing their routes.

The main contributions of this work are:

- the introduction of the Multi-period Multi-trip (MM) CDP with Release and Due Dates, i.e., MM-CDP-RDD, a more realistic CDP variant in which the planning horizon is divided into several discrete periods (multi-period), a truck is allowed to perform more than one trip in each period (multi-trip), RDDs are associated with each customer and finally, restrictions on the trip duration in each period, in two

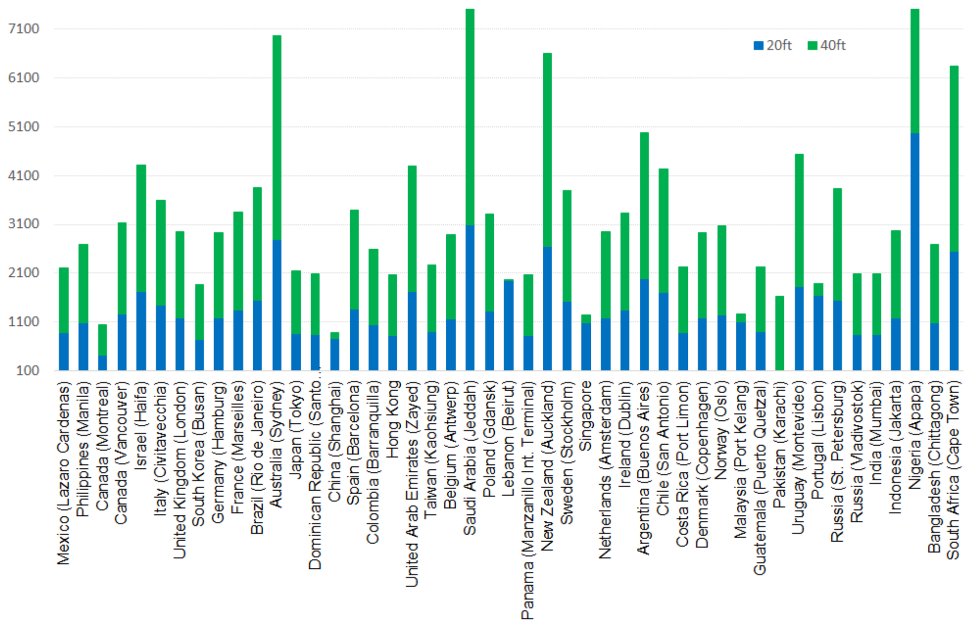

Figure 1: Total costs (in $) to move both 20 ft and 40 ft containers from New York (USA) to other countries (port city). Source: https://moverdb.com/freight-costs-usa/

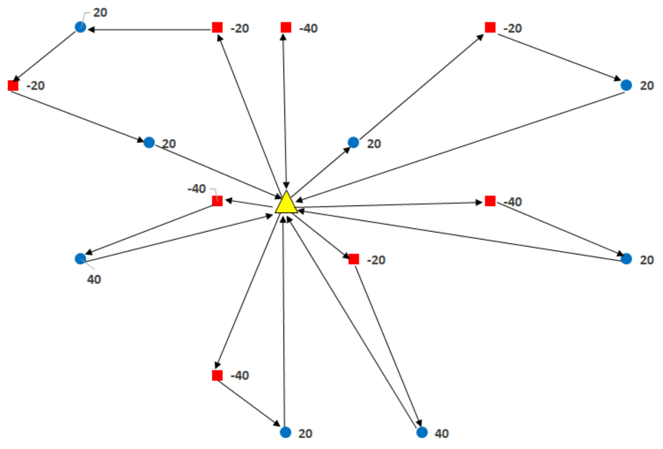

Figure 2: An example of feasible solution of the CDP.

consecutive periods and in the whole planning horizon are imposed. The latter assumptions allow accounting for contractual limitations on the drivers' work hours.

- the formulation of an Arc-based Integer Linear Programming (A-ILP) model;
- the design of an exact two-stage approach together with the definition of both trip feasibility and dominance rules. In the first stage, all the feasible non-dominated trips are generated. In the second stage, a Trip-based ILP (T-ILP) model is solved;
- the design of several Combinatorial Benders' Cuts (CBC) approaches, based also on valid inequalities defined ad hoc for the MM-CDP-RDD;
- the generation of a rich set of instances for this new problem, starting from the traditional benchmark instances proposed for the Vehicle Routing Problem by both Solomon [45] and Gehring & Homberger [26];
- the numerical comparisons among A-ILP, T-ILP (also by adding valid inequalities) and the CBC approaches.

The rest of the paper is organized as follows. Section 2 reviews the main literature contributions on the CDP and its variants. In Section 3, the statement of the problem is given together with the notation used. In Section 4, the A-ILP model for the MM-CDP-RDD is described, while Section 5 proposes a two-stage approach. Section 6 describes several variants of the CBC approach specifically designed for the MM-CDP-RDD. In Section 7, benchmark instances for this new problem are generated and numerical comparisons among the solution methods are discussed. Finally, Section 8 draws some conclusions and outlines future research directions worthy of investigation.

2. Literature Review

The CDP and its variants belong to the most general class of Vehicle Routing Problem (VRP) with Pickups and Deliveries, i.e., the Pickup and Delivery Problem (PDP), because customers can require either a container pickup or its delivery ([39], [38]). Moreover, as specified by the recent literature review [23] on the CDP and its variants, the scientific contributions can be classified into two main groups, depending on the operation policy adopted: Stay-with and Drop&Pick. According to the Stay-with policy, a truck carrying a full container to an import customer waits until the container is emptied and then carries away the empty container, which can be either transported back to the terminal or used to serve an export customer within the same trip. Even in this case, the truck has to wait at the customer location until the container is fully loaded and can be transported to the terminal. This policy has advantages and disadvantages. The main advantage is that a container used to serve an import customer, once emptied, can be immediately reused to serve an export customer without coming back to the depot. On the other hand, this policy has two main disadvantages. Firstly, trucks, and consequently drivers, spend a large part of the working shift waiting for containers to be emptied or filled, without having the possibility of performing other tasks in the meantime, resulting in a strong under-use of the available resources. Secondly, containers which are immediately reused after being emptied do not undergo a proper disinfection procedure, with consequent risks of contamination of goods belonging to different categories and of propagation of infections. Furthermore, containers that are damaged during emptying operations, requiring to be repaired before reusing them, may yield a disruption in the routing planning. The second operation policy, Drop&Pick, allows trucks to deliver a container, full or empty, to the customers and immediately restart the journey without waiting for the container to be emptied or filled. Compared to the Stay-with policy for which 4 kinds of requests may occur, import of full or empty containers and export of full and empty containers, with this approach we consider just two groups of requests, import and export. Indeed, the fact that the container is empty or full does not impact on the further tasks that can be performed in the same trip. The main advantage is that resources (i.e. trucks) can be better exploited, and can perform more routes during the day since they do not have to wait for containers to be filled or emptied. The disadvantage is that containers cannot be reused along the same trip and must be picked up later by another or the same truck. This policy is not convenient when customers are very far from the terminal, while it is convenient when they are close enough to the terminal. As one can evince from [23], both policies have been addressed in the literature and none of them outperforms the other. In this paper, we study a multi-trip problem, in which customers are close enough to the terminal and trucks can perform several trips per day. Therefore, we chose to adopt the Drop&Pick strategy.

In this paper, we introduce the following two new features: *Multi-period* (MP), i.e., the planning horizon is divided into discrete time periods (e.g., days) and then, the release and due dates at customers are specified in terms of periods in which they have to be served; *Multi-trip* (MT), i.e., each truck can perform more than one trip in each period. The latter extension is inspired by the increasing interest of the researchers in this topic, in recent years, mainly motivated by new city logistic distribution systems, as observed in the survey [12]. In this survey, four, three and two index mathematical formulations are presented, depending on considering both the vehicle and the trip index, only the vehicle index, neither of the two, respectively. Regarding the exact approaches, both branch-and-cut [30] and branch-and-price [36] algorithms have been proposed. Regarding the heuristics, two-stage algorithms (e.g., [48], [40]) and meta-heuristics, e.g., tabu search [7] and population-based algorithms [13] have been also proposed. Concerning the multi-period, the survey of [6] shows an increasing interest of the researchers especially since 2009 (e.g., [24]). Both exact approaches (e.g., [17]) and meta-heuristics, like Variable Large Neighborhood Search (e.g., [25]) and Adaptive Large Neighborhood Search (e.g., [42]), have been proposed.

We discuss the main literature contributions on CDP and its variants with regard the aforementioned features.

The Drop&Pick CDP with Time Windows (TW) at customers, with only one container per truck, has been extensively studied in the literature. In [29], the CDP with intermediate facilities between terminals and customers in a metropolitan area is addressed and modeled as a multi-Traveling Salesman Problem with TW (m-TSPTW) at both origins and destinations. An exact solution approach based on dynamic programming is proposed. In [20], the problem is modeled as a Multi-Resource Routing Problem with flexible tasks and formulated as a set partitioning problem with a weighted objective function minimizing both the fleet and the variable distance cost. Meanwhile, a cluster method together with a reactive Tabu Search (TS) is proposed in [58]. In [57], the problem is addressed as an m-TSPTW and solved by a modified version of the method of [51]. In [59], it is modelled as an m-TSPTW with resource constraints and solved through a reactive TS.

Allowing that a tractor can be assigned to a different trailer to perform a new task, in [53], a node-arc formulation is modelled coordinating the empty containers that move between customers. A TS algorithm is also designed for solving instances with up to 400 customers, while a maxmin ant colony optimization algorithm is proposed in [52]. In [54], the same problem has been addressed by a Combinatorial Benders' Cuts approach in which the problem is decoupled in a Master Problem involving only arc variables, contributing to the objective function, while the Slave Problem involves tractors-to-trailers assignment variables, responsible for feasibility. A Combinatorial Benders' Cuts approach has also been adopted to address the Multi-Trip CDP in [8].

A Drop&Pick CDP without TW at the customers, with only one container per truck, is addressed in [5], minimizing simultaneously the number of vehicles used and the total travel distance. Both a hybrid deterministic annealing algorithm and a TS are designed.

A Stay-with CDP with more than one container per truck is modelled as a VRP with multiple visits and heterogeneous trucks in [32] and solved through a variant of the Clarke-and-Wright algorithm. An Adaptive Guidance meta-heuristic is proposed in [31].

The Stay-with CDP with TW at customers is addressed in [11], where the problem is solved through a local search, based on three neighborhoods, with an initial solution computed by a two-phase insertion heuristic. Several techniques already proposed in the literature for the VRPs are extended in order to schedule pre- and end-haulage of containers in [41] and several versions are studied, i.e., both managing multiple empty container depots and balancing empty container depot levels.

In the CDP proposed in [43] for taking control of the harmful emissions, collaboration among truckers is permitted and a mathematical model based on the m-TSPTW is formulated. Finally, in [23], the authors, after introducing the possibility to move more than one container per truck in a trip and by exploiting the limited number of feasible trips, propose a set-covering formulation tested on real-world case studies.

Recently, further extensions of the CDP have been introduced. In [55], the authors address the possibility of using a truck platooning operation mode in which a platoon is formed by a leading truck followed by a set of trucks using semi-automated technologies, where only the leading truck is human-driven. To solve the problem, a mathematical formulation and ant colonies based heuristic are provided. In [19], the authors addressed long-haul CDP where trips may last for multiple days and driver rest periods along the trips are considered, for which they propose an ILP formulation. A bi-objective CDP is addressed in [56], where the number of trucks employed and the total covered distance are simultaneously minimized. The problem is solved by a Large Neighborhood Search heuristic. For a more detailed review and classification of CDP problems, we refer the reader to [23] and [56]

In particular, multi-trip CDPs are addressed in both [35] and [9], the latter considers only 20 ft sized containers. However, neither of them considers multiple periods. Therefore, to the best of our knowledge, no contributions already exist in which the multi-period is combined with the multi-trip, the RDDs of the customers, the contractual restrictions on the trip duration, allowing both moving more than one container per truck and managing two container sizes (20 ft and 40 ft).

3. Problem statement and notation

The MM-CDP-RDD aims at efficiently routing a fleet of $|K|$ trucks, based at a common terminal 0, to serve import/export customers (set C), while minimizing the total travel distance over a given planning horizon. For this purpose, the planning horizon is divided into ν discrete time periods, each one representing a working day. Hereafter, we suppose $T = \{1, 2, \ldots, \nu\}$. Within the same period, each truck can perform more than one trip. However, due to drivers' contractual limitations, each truck can travel no longer than T_1^{max}, T_2^{max} and T_3^{max}, in a period, in two consecutive periods and in the whole planning horizon (generally, a week), respectively,

Table 1: Notation of the MM-CDP-RDD

Set	**Meaning**
C	Set of customers
K	Set of trucks
T	Set of periods
T_i	Set of periods in which customer i can be served
T_{ij}	Set of periods in which both customer i and customer j can be served
C_π	Set of customers served by the trip π
$\tilde{T}_\pi$	Set of periods in which the trip π can be performed

Parameter	**Meaning**
0	Terminal
v	Average truck speed
Q	Loading truck capacity
T_1^{max}	Maximum travel time per truck per period
T_2^{max}	Maximum travel time per truck in two consecutive periods
T_3^{max}	Maximum travel time per truck in the planning horizon
d_{ij}	Travel distance between $i \in C \cup \{0\}$ and $j \in C \cup \{0\}$
r_i	Demand of customer i
τ_i	Service time at customer i
d_π	Total distance travelled in the trip π
$a_{i\pi}$	1 if customer $i \in C$ is served in the trip π; 0 otherwise.

[34]. Each truck, whose average speed is equal to v, also has a limited loading capacity Q equal to 40 ft meaning that it can simultaneously move either only one container of 40 ft or two containers of 20 ft.

The demand r_i of each customer $i \in C$ can be either positive (export customer) or negative (import customer) and of either 40 ft or 20 ft. The service time at each customer $i \in C$ (i.e., the time to pick up or drop off the container at customer's site) is indicated by τ_i. In addition, for each node $i \in C \cup \{0\}$, T_i indicates the set of consecutive periods in which the service can be performed at node i. In particular, T_0 contains all the periods of the planning horizon, i.e., $T_0 = T$. Moreover, for each pair (i, j) such that $i, j \in C \cup \{0\}, i \neq j$, T_{ij} indicates the subset of T containing the periods in which both the nodes i and j can be served, i.e., $T_{ij} = T_i \cap T_j$. For each pair $(i, j) \in C \cup \{0\}$, the travel distance d_{ij} is known. Table 1 summarizes the notation used in this paper.

A trip π starts from the terminal, serves a sub-set of customers C_π and then returns to the terminal. Its total distance is indicated by d_π and the periods in which it can be performed is denoted by $\tilde{T}_\pi = \cap_{i \in C_\pi} T_i$. Finally, we also introduce a customers' coverage matrix in which the entry $a_{i\pi}$ is 1 if and only if the customer $i \in C$ is served in the trip π, i.e., i belongs to C_π; 0, otherwise.
Since each truck can move either a 40 ft container or two containers of 20 ft, all possible kinds of feasible trips can be enumerated according to the following definition.

Definition 1: Type of trips

- 1-customer trip $\{0, i, 0\}$, with $r_i \in \{40, -40, 20, -20\}$;

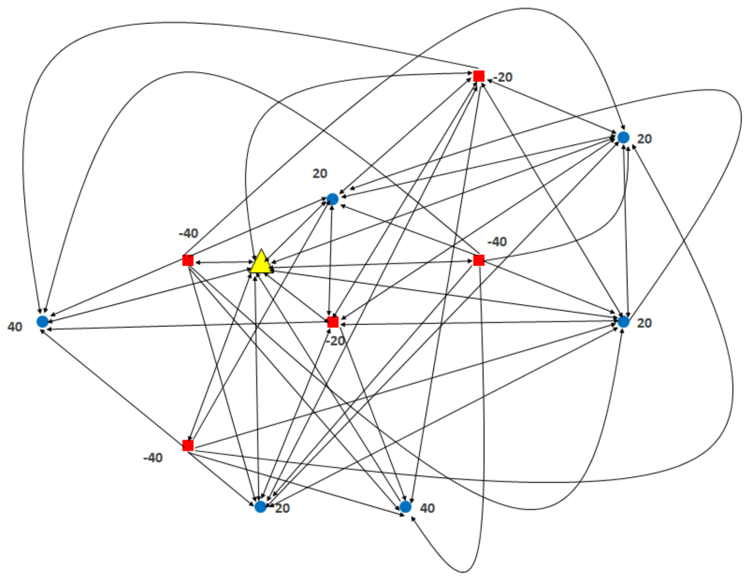

Figure 3: A graph instance of the MM-CDP-RDD.

- 2-customer trip $\{0, i, j, 0\}$, with $(r_i, r_j) \in \{(-20, -20), (20, 20), (20, -20), (-20, 20), (-40, 20), (-20, 40), (-40, 40)\}$;
- 3-customer trip $\{0, i, j, k, 0\}$, where $(r_i, r_j, r_k) \in \{(-20, -20, 20), (-20, 20, 20), (-20, 20, -20), (20, -20, 20), (-20, -20, 40), (-40, 20, 20)\}$;
- 4-customer trip $\{0, i, j, k, v, 0\}$, where $(r_i, r_j, r_k, r_v) \in \{(-20, 20, -20, 20), (-20, -20, 20, 20)\}$.

4. An Arc-based Mathematical Programming Formulation

Similarly to the traditional VRPs, the MM-CDP-RDD can be formally represented on a directed graph $G = (N, A)$, where the set of nodes N contains the set C of customers to be served and the terminal 0, while A is the set of the arcs.

According to Definition 1, A is not complete because, through a proper pre-processing, only some arcs are kept. More specifically, $A = \{(i, j) \in C \times C : i \neq j \wedge (r_i, r_j) \in \{$(-20,-20), (20,20), (20,-20), (-20,20), (-40,20), (-20,40), (-40,40)$\} \wedge T_i \cap T_j \neq \emptyset \wedge d_{ij}/v \leq T_1^{max}\} \cup \{(0, j), j \in C : d_{0j}/v \leq T_1^{max}\} \cup \{(j, 0), j \in C : d_{j0}/v \leq T_1^{max}\}$. Figure 3 shows the graph G generated for the instance of the problem of Figure 2.

The formulation is based on the following decision variables: x_{ij}^{kt}, binary variables equal to 1 if truck k travels from node i to node j in period t and 0 otherwise, $\forall (i, j) \in A$, $\forall k \in K$ and $\forall t \in T$; u_i^+, an integer variable denoting the total quantity of pickup until node i from the last exit from 0, $\forall i \in N$;

u_i^-, an integer variable representing the total quantity of delivery until node i from the last exit from 0, $\forall i \in N$.

The Arc-based Integer Linear Programming (A-ILP) formulation of the MM-CDP-RDD is the following:

$$\min \sum_{(i,j)\in A} \sum_{k\in K} \sum_{t\in T_{ij}} d_{ij} x_{ij}^{kt} \tag{1}$$

s.t.

$$\sum_{k\in K} \sum_{j:(i,j)\in A} \sum_{t\in T_{ij}} x_{ij}^{kt} = 1 \quad \forall i \in C \tag{2}$$

$$\sum_{\substack{j:(i,j)\in A\\ t\in T_j}} x_{ij}^{kt} - \sum_{\substack{j:(j,i)\in A\\ t\in T_j}} x_{ji}^{kt} = 0 \quad \forall i \in N,\, k \in K,\, t \in T_i \tag{3}$$

$$\sum_{\substack{(i,j)\in A\\ t\in T_{ij}}} (\frac{d_{ij}}{v} + 2\tau_j) x_{ij}^{kt} \le T_1^{max} \quad \forall k \in K,\, t \in T \tag{4}$$

$$\sum_{\substack{(i,j)\in A\\ t\wedge(t+1)\in T_{ij}}} (\frac{d_{ij}}{v} + 2\tau_j) x_{ij}^{kt} + \sum_{\substack{(i,j)\in A\\ t\wedge(t+1)\in T_{ij}}} (\frac{d_{ij}}{v} + 2\tau_j) x_{ij}^{k(t+1)} \le T_2^{max} \quad \forall k \in K,\, t = 1,\dots,|T|-1 \tag{5}$$

$$\sum_{(i,j)\in A} \sum_{t\in T_{ij}} (\frac{d_{ij}}{v} + 2\tau_j) x_{ij}^{kt} \le T_3^{max} \quad \forall k \in K \tag{6}$$

$$u_0^+ = u_0^- = 0 \tag{7}$$

$$u_j^+ \ge u_i^+ + max\{r_j, 0\} - (Q + max\{r_j, 0\})(1 - \sum_{k\in K} \sum_{t\in T_{ij}} x_{ij}^{kt}) \quad \forall (i,j) \in A : j \ne 0 \tag{8}$$

$$u_j^- \ge u_i^- - min\{r_j, 0\} - (Q - min\{r_j, 0\})(1 - \sum_{k\in K} \sum_{t\in T_{ij}} x_{ij}^{kt}) \quad \forall (i,j) \in A : j \ne 0 \tag{9}$$

$$x_{ij}^{kt} \le u_i^- - u_i^+ \quad \forall (i,j) \in A : i \ne 0, j \ne 0, r_i = r_j = -20, \forall k \in K, \forall t \in T_{ij} \tag{10}$$

$$x_{ij}^{kt} \le Q - u_i^+ \quad \forall (i,j) \in A : i \ne 0, j \ne 0, r_i = 20, r_j = -20, \forall k \in K, \forall t \in T_{ij} \tag{11}$$

$$u_i^- \le Q \quad \forall i \in C \tag{12}$$

$$u_i^+ \le Q \quad \forall i \in C \tag{13}$$

$$\sum_{k\in K} \sum_{t\in T_i} x_{i0}^{kt} = 1 \quad \forall i \in C : r_i = +40 \tag{14}$$

$$\sum_{k\in K} \sum_{t\in T_i} x_{0i}^{kt} = 1 \quad \forall i \in C : r_i = -40 \tag{15}$$

$$x_{j0}^{kt} \geq x_{ij}^{kt} \quad \forall k \in K, i \in C, j \in C : i \neq j, r_i = r_j = 20, t \in T_{ij} \tag{16}$$

$$x_{ij}^{kt} \in \{0,1\} \quad \forall (i,j) \in A, k \in K, t \in T_{ij} \tag{17}$$

$$u_i^+, u_i^- \geq 0 \ \ integer \quad \forall i \in C \tag{18}$$

The objective function (1) to be minimized represents the total travel distance. Constraints (2) assure that each customer is served exactly once, while constraints (3) guarantee the flow conservation for each node of the graph. Constraints (4)-(6) assure that each truck cannot travel longer than T_1^{max}, T_2^{max} and T_3^{max}, respectively, in a period, in two consecutive periods and in the whole planning horizon. It is worth noting that the service time of each customer is counted twice for taking into account the related service time also at the terminal.

Constraint (7) fixes to 0 the quantity of both pickup and delivery at the terminal, while constraints (8)-(9) count the quantity of pickup and delivery at each node, respectively. Since these constraints are imposed for each arc except those entering in the terminal, they also ensure that the subtours not visiting the terminal cannot be feasible.

Constraints (10) ensure that two consecutive deliveries of a 20 ft size cannot be served after a pickup of 20 ft size, i.e., a trip $\{0, v, i, j, 0\}$, where $r_v = 20, r_i = r_j = -20$, is not allowed. Similarly, constraints (11) avoid that a pickup and a delivery, both of 20 ft size, are consecutively served after a delivery and a pickup, both of 20 ft size, i.e., a trip $\{0, q, v, i, j, 0\}$, where $r_q = r_j = -20, r_v = r_i = 20$, is not permitted.

Constraints (12)-(13) guarantee that the maximum truck loading capacity is never exceeded. In a trip, a customer $i \in C$ with $r_i = 40$ ft has to be served as the last (14), and each customer $i \in C$ with $r_i = -40$ ft has to be served as the first (15). A truck, after consecutively visiting two customers i and j with $r_i = r_j = 20$ ft, has to return to the terminal (16). Finally, constraints (17)-(18) define the decision variables nature.

5. A two-stage trip-based approach

In this section, the two-stage trip-based approach designed for solving the MM-CDP-RDD is described, by exploiting the generation rules (Definition 1). The approach proposed works as in the following. A trip generation procedure is firstly invoked (Section 5.1). In order to further reduce the number of trips to manage, both feasibility and dominance rules are also introduced. This way, the set Π of all feasible non-dominated trips is generated. Then, a Trip-based Integer Linear Programming formulation (T-ILP) is solved by receiving Π as input (Section 5.2).

5.1. Trip generator

From all the possible trips generated according to the rules introduced in Section 3, only the *feasible* are kept, according to the following definition:

Definition 2: Feasible trip.
A trip π is *feasible* if and only if:

- $t_\pi = \frac{d_\pi}{v} + \sum_{i \in C_\pi} 2\tau_i \leq T_1^{max}$;
- $\tilde{T}_\pi = \bigcap_{i \in C_\pi} T_i \neq \emptyset$;
- The truck capacity Q is never exceeded.

Moreover, in order to reduce the number of trips to manage as far as possible, among the feasible ones, we maintain only the set Π of *non-dominated* ones, according to the following definition:

Definition 3: Dominated trip.
Given two feasible trips π_1, π_2, the former dominates the latter if and only if:

- $C_{\pi_1} = C_{\pi_2}$;
- $d_{\pi_1} < d_{\pi_2}$;

Algorithm 1 outlines the trip generation procedure whose time complexity, in the worst case, is $O(|C|^4|T|)$. It is worth noting that $|T|$ is usually much smaller than $|C|$. All details on the pseudo-codes of the routines implemented and invoked by Algorithm 1 and on their time complexity are given in Appendix A.

Algorithm 1 Trips Generator

```
  Input: $C$, 0, $d_{ij} \forall i, j \in C \cup \{0\}$,$T_i, r_i, \tau_i \forall i \in C$, $T_1^{max}$, $Q$, $v$.
  Output: $\Pi$
 1: $\Pi := create()$
 2: $\pi := create\_trip()$
 3: $C_\pi := \varnothing$
 4: $totalTime := \emptyset$
 5: $d_\pi := \emptyset$
 6: $t\pi := \emptyset$
 7: $days := T$
 8: $len := \emptyset$
 9: $l := 0$
10: $\Pi := IncreasedTrip(\pi, C_\pi, d_\pi, t_\pi, totalTime, days, len, l, \Pi)$
```

5.2. Trip-based Integer Linear Programming model

The Trip-based ILP model (T-ILP) receives as input the set Π of all feasible non-dominated trips, generated through Algorithm 1. It is formulated by introducing the following decision variables: z_π equal to 1 if trip $\pi \in \Pi$ is selected; 0 otherwise; $y^t_{\pi k}$ equal to 1 if trip $\pi \in \Pi$ is performed by truck $k \in K$ in period $t \in \tilde{T}_\pi$.

The mathematical formulation is given in what follows.

$$min \sum_{\pi \in \Pi} d_\pi z_\pi \tag{19}$$

$$\sum_{\pi \in \Pi} a_{i\pi} z_\pi = 1 \quad \forall i \in C \tag{20}$$

$$\sum_{k \in K} \sum_{t \in \tilde{T}_\pi} y^t_{\pi k} = z_\pi \quad \forall \pi \in \Pi \tag{21}$$

$$\sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} \leq T^{max}_1 \quad \forall k \in K, \forall t \in T \tag{22}$$

$$\sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} + \sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^{t+1}_{\pi k} \leq T^{max}_2 \quad \forall k \in K, \forall t = 1, \ldots, |T| - 1 \tag{23}$$

$$\sum_{t \in T} \sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} \leq T^{max}_3 \quad \forall k \in K \tag{24}$$

$$z_\pi \in \{0, 1\} \quad \forall \pi \in \Pi \tag{25}$$

$$y^t_{\pi k} \in \{0, 1\} \quad \forall \pi \in \Pi, \forall k \in K, \forall t \in T \tag{26}$$

The objective function (19) to be minimized represents the total travel distance. Customers' coverage is guaranteed by constraints (20), while constraints (21) logically link variables y and z, i.e., a trip π is selected if and only if it is assigned to a truck in a period. Constraints (22)-(24) assure that a truck cannot travel longer than T^{max}_1, T^{max}_2 and T^{max}_3, respectively, in a period, in two consecutive periods and in the whole planning horizon. Finally, constraints (25)-(26) define the variables' nature.

6. Combinatorial Beneders' Cuts approaches

Benders' Decomposition (BD) is a successful and broadly applied exact approach to solve Mixed Integer Programming (MIP) models introduced in the pioneering work of Benders [4]. The core idea, which this approach is based on, consists in fixing a set of variables, which make the MIP hard to be solved, so that the problem can be strongly simplified. In the BD, the problem is decomposed into a Master Problem (MP), in which only a subset of the decision variables is considered and into a Slave Problem (SP), containing the remaining variables. The MP and the SP are iteratively solved in sequence.

At each iteration, the MP is solved first, then, the SP is formulated by fixing the variables' values found by the MP, and solved for the remaining variables. Based on the SP outcome, one or more cuts can be generated and added to MP, preventing it from exploring specific areas of the search space. In the classical application of BD to MIPs, the SP is a Linear Programming (LP) problem, whose dual solution is exploited to derive cuts to be added to the MP. In [22], BD has been generalized and extended to problems in which the SP is not required to be an LP problem.

Logic Based Benders' decomposition, introduced many decades later in [27], is an extension of the classical BD in which, in the MP only the variables that contribute directly to the objective function are considered, while the others are relegated to the SP. This way, the SP becomes a pure feasibility problem. Each time the SP is infeasible, one or more cuts can be generated to cut off infeasible solutions from the MP solutions search space. As soon as a feasible SP is detected, the overall solution obtained is proved to be optimal.

A specific case of Logic Based Benders' decomposition, named Combinatorial Benders' decomposition, has been presented in [15]. This approach is specifically suited for MIP models, involving binary variables and a large number of logical implications through Big-M constraints. In this scenario, each time a given combination of the MP variables yields to an infeasible SP, a Combinatorial Benders' Cut (CBC) is added, forcing at least one of the variables equal to 1 in the current MP optimal solution to be 0. If the number of variables equal to 1 in the MP optimal solution is large, the derived cut may be very weak. Stronger cuts can be obtained by identifying a subset of variables responsible for the infeasibility through the search of the Minimum Infeasible Set (MIS), which can be determined through either an exact or a heuristic approach. Such cuts are stronger since they allow cutting off from the MP search space several solutions at a time, strongly speeding up the convergence towards an optimal solution. However, the algorithm convergence is always guaranteed even by adopting only standard CBC, cutting off one solution at a time as proved in [15].

In the last decade, CBC has been successfully applied to real problems arising in different contexts. The first application is described in [3], where a toll facilities location problem is addressed. In [16], an exact CBC-based approach for the Strip Packing problem is proposed. Several applications in the field of Port Logistics are reported in the literature, e.g., quayside operations at container terminals are studied in [10] and [14], and in [49], the lock scheduling problem is addressed, instead. Other innovative applications can be found in health care, regarding beam intensity modulation in radiotherapy [47], in production, where assembly line balancing is studied [1] and in jobs allocation in computer clusters [33].

All the above cited problems share a common structure where the variables can be partitioned into two subsets. The first one, involved in the MP, contains all the variables directly contributing to the objective function, while the second one, containing variables only responsible for the solution feasibility, is involved in the SP.

In this section, we propose several CBC approaches in which we try to speed up the standard CBC method both providing stronger cuts and adding valid inequalities to the MP.

6.1. CBC1

The first developed CBC approach (CBC1) is based on the classical framework of [15]. The master problem, MP1, aims at finding the optimal subset of trips covering all the customers at minimum cost and, therefore it becomes a pure Set Partitioning problem, quickly solved to optimality.

MP1:

$$min \sum_{\pi \in \Pi} d_\pi z_\pi \tag{27}$$

$$\sum_{\pi \in \Pi} a_{i\pi} z_\pi = 1 \quad \forall i \in C \tag{28}$$

$$z_\pi \in \{0,1\} \quad \forall \pi \in \Pi \tag{29}$$

The objective function (27) to be minimized represents the total travel distance as in the T-ILP model, while constraints (28) and (29) correspond to constraints (20) and (25), respectively.

The slave problem, SP1, checks if a feasible assignment of those trips to both periods and trucks exists such that the duration constraints are respected, becoming a pure feasibility problem.

SP1:

$$\sum_{k \in K} \sum_{t \in \tilde{T}_\pi} y^t_{\pi k} = z^\star_\pi \quad \forall \pi \in \Pi \tag{30}$$

$$\sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} \le T^{max}_1 \quad \forall k \in K, \forall t \in T \tag{31}$$

$$\sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} + \sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^{t+1}_{\pi k} \le T^{max}_2 \quad \forall k \in K, \forall t = 1, \ldots, |T| - 1 \tag{32}$$

$$\sum_{t \in T} \sum_{\pi \in \Pi : t \in \tilde{T}_\pi} t_\pi y^t_{\pi k} \le T^{max}_3 \quad \forall k \in K \tag{33}$$

$$y^t_{\pi k} \in \{0,1\} \quad \forall \pi \in \Pi, \forall k \in K, \forall t \in T \tag{34}$$

where $z^\star_\pi$ is equal to 1 if trip π has been selected in the MP1 optimal solution; 0 otherwise. Constraints (30) imply that if a trip has been selected by MP1, it must be assigned to exactly one period and one truck, while constraints (31)-(34) correspond to constraints (22)-(24) and (26) of the T-ILP model.

MP1 and SP1 are iteratively solved in sequence. If SP1 is infeasible, then a cut is added to MP1, imposing that at least one of the trips selected by MP1 would not be selected in the next iterations:

$$\sum_{\pi \in \Pi^\star} z_\pi \leq |\Pi^\star| - 1 \tag{35}$$

where $\Pi^\star$ is the set of trips selected by MP1 at the current iteration and $|\Pi^\star|$ indicates the number of trips belonging to the set $\Pi^\star$.

The procedure is repeated until SP1 becomes feasible. As soon as a feasible solution for SP1 is found, it is proved to be optimal.

The main drawback of CBC1 is that if the number of trips selected by MP1 is very large, the cuts may become too weak, and, therefore, the convergence toward an optimal solution may be slow.

6.2. CBC2

In the second CBC approach proposed, (CBC2), we focus our attention on the identification of a subset of the selected trips which is responsible for the SP infeasibility. In particular, we focus on the selected trips which can be performed only in a given subset of consecutive periods S and whose duration imposes that they cannot be paired among them, thus requiring a truck each. The number of selected trips of this type cannot be greater than the number of trucks multiplied by the number of periods in S, in order to obtain a feasible solution.

Both the MP and the SP, namely MP2 and SP2, respectively, exactly correspond to MP1 and SP1. Each time SP2 turns out to be infeasible, for all the periods in S, we check if the number of the selected trips that can be performed only in those periods and cannot be paired with others, is greater than $|S|\cdot|K|$. If this happens, we add to MP2, beyond the traditional cut (35), the following cut:

$$\sum_{\pi \in \Pi_S^\star} z_\pi \leq |S|\cdot|K| \tag{36}$$

where $\Pi_S^\star$ represents the set of trips selected by MP2, that can be performed only in periods belonging to S and cannot be paired among them. The cut (36) becomes very useful when trips are reasonably long with respect to the maximum truck usage duration per period, T_1^{max} and when the set of periods in which a customer can be served is smaller than the total number of periods, $|T|$, e.g., when the number of trips π for which $|\tilde{T}_\pi| = 1$ is high. In fact, the smaller $|S|$, the stronger the related cut is.

6.3. CBC3

The third CBC approach designed, (CBC3), follows the same core idea of CBC2. However, instead of dynamically adding constraints on the maximum number of non-combinable trips per period, whenever this restriction is violated by the current MP, now, these constraints are directly imposed in the MP from the beginning, acting as valid inequalities. Therefore, MP3 can be obtained by adding to MP1 the following valid inequalities:

$$\sum_{\pi \in \Pi_S} z_\pi \leq |S|{\cdot}|K| \quad \forall S = \{t_1, t_1+1, \dots, t_2\} \subseteq T \tag{37}$$

where S is every subset of consecutive periods selected over T and Π_S is the set of all the trips that can be performed only in periods belonging to S and cannot be paired among them. SP3 exactly corresponds to SP1.

It is worth noting that the total number of valid inequalities (37) only increases in a quadratic way in function of $|T|$, since the total number of subsets S of consecutive periods, selected over T, is $|T| - |S| + 1$ and $|S|$ can vary from 1 to $|T|$.

6.4. CBC4

In the fourth CBC approach, (CBC4), we pursue a completely different philosophy to identify stronger cuts. MP4 and SP4 exactly correspond to MP1 and SP1, respectively. The main difference is that whenever SP4 turns out to be infeasible, i.e., no feasible solution exists with all the $|\Pi^\star|$ selected by the MP, we try to compute the smallest integer value α^* for which a feasible solution can be obtained containing at most $|\Pi^\star| - \alpha^*$ trips. This approach is able to find very strong cuts and, consequently, to reduce the number of iterations needed to reach optimality. However, the computational time of each iteration may become slightly higher than that required by the other CBC approaches. It is worth noting that the greater the value of α^*, the stronger the cut, but the longer the computational time required to solve the feasibility check problem.

More specifically, MP4 and SP4 are iteratively solved in sequence. Whenever SP4 turns out to be infeasible, a Feasibility Check Problem, (FCP4), is formulated. FCP4 corresponds to SP4 except for the fact that the constraints (30) are replaced by the following ones:

$$\sum_{k \in K} \sum_{t \in \tilde{T}_\pi} y^t_{\pi k} = z_\pi \quad \forall \pi \in \Pi \tag{38}$$

$$\sum_{\pi \in \Pi^\star} z_\pi \leq |\Pi^\star| - \alpha \tag{39}$$

$$z_\pi \in \{0, 1\} \quad \forall \pi \in \Pi \tag{40}$$

with α initially equal to 2. While FCP4 remains infeasible, we increase α by 1 and reiterate until FCP4 turns out to be feasible. In this way, we exactly identify the maximum number of trips, $\alpha^\star = |\Pi^\star| - \alpha$, among those selected by MP4, that can be selected in a feasible solution. Therefore, we can replace the classical CBC (35), to be added to the MP, by the following stronger cut:

$$\sum_{\pi \in \Pi^\star} z_\pi \leq \alpha^\star \tag{41}$$

The CBC4 pseudo-code is in Algorithm 2.

Algorithm 2 CBC4

```
1: Solve MP4
2: Set, in SP4, the $z$ variables to be equal to the optimal values found by MP4
3: Solve SP4
4: while SP4 is infeasible do
5:    Set $\alpha = 2$
6:    Solve FCP4
7:    while FCP4 is infeasible do
8:       Set $\alpha = \alpha + 1$
9:       Solve FCP4
10:   end while
11:   Add constraint (41) to MP4
12:   Solve MP4
13:   Set, in SP4, the $z$ variables to be equal to the optimal values found by MP4
14:   Solve SP4
15: end while
16: Return the MP4 optimal solution
```

6.5. CBC5

The fifth CBC approach, (CBC5), follows a similar idea to that of CBC2. However, instead of working on the maximum number of non-combinable trips, that can be performed only in a given period t, which can be selected by the MP, it deals with the maximum total duration of the selected trips that can be performed only in a given period t. We then formulate two Feasibility Check Problems whenever the slave problem is infeasible. The first one, FCP_5^1, checks, for each period t, if the total duration of the trips that can be performed only in t, selected by MP5 (indicated by $\Pi_t^\star$), does not exceed T_1^{max}, multiplied by the number of trucks, $|K|$. The same consideration is applied for each pair of consecutive periods $t, t+1$ in the second Feasibility Check Problem, (FCP_5^2). In fact, the total duration of the trips that can be performed only in the subset of periods $t, t+1$, selected by MP5 (indicated by $\Pi_{t,t+1}^\star$), must not exceed the maximum allowed duration T_2^{max}, multiplied by $|K|$. If FCP_5^1 turns out to be infeasible, the following cut is added, together with the classical CBC, (35):

$$\sum_{\pi \in \Pi_t^\star} z_\pi \leq |\Pi_t^\star| - 1 \quad \forall t \in T \tag{42}$$

if FCP_5^2 turns out to be infeasible, the following cut is added, together with the classical CBC, (35):

$$\sum_{\pi \in \Pi^{\star}_{t,t+1}} z_\pi \leq |\Pi^{\star}_{t,t+1}| - 1 \quad \forall t = 1, \dots, |T| - 1 \tag{43}$$

The CBC5 pseudo-code is in Algorithm 3.

Algorithm 3 CBC5

```
 1: Solve MP5
 2: Set, in SP5, the $z$ variables to be equal to the optimal values found by MP5
 3: Solve SP5
 4: **while** SP5 is infeasible **do**
 5:     Solve $FCP_5^1$
 6:     **if** $FCP_5^1$ is infeasible **then**
 7:         Add constraint (42) to MP5
 8:     **end if**
 9:     Solve $FCP_5^2$
10:     **if** $FCP_5^2$ is infeasible **then**
11:         Add constraint (43) to MP5
12:     **end if**
13:     Add constraint (35) to MP5
14:     Solve MP5
15:     Set, in SP5, the $z$ variables to be equal to the optimal values found by MP5
16:     Solve SP5
17: **end while**
18: Return the MP5 optimal solution
```

6.6. CBC6

Finally, the sixth CBC approach, CBC6, is based on the same idea as CBC5 but all the cuts (42) and (43) are directly added to the MP, at the beginning of the search process. According to this, MP6 is equal to MP1 with the additional constraints:

$$\sum_{\pi \in \Pi_t} t_\pi z_\pi \leq |K| \cdot T_1^{max} \quad \forall t \in T \tag{44}$$

$$\sum_{\pi \in \Pi_{t,t+1}} t_\pi z_\pi \leq |K| \cdot T_2^{max} \quad \forall t = 1, \dots, |T| - 1 \tag{45}$$

where $\Pi_{t,t+1}$ indicates the set of trips that can be performed only in the pairs of periods (t,t+1). SP6 exactly corresponds to SP1. Both CBC6 and CBC5 are particularly useful when $|\tilde{T}_\pi|$ is small, but unlike what happens in CBC2 and CBC3, they are effective even when trips are short with respect to T_1^{max}, i.e., when the number of trips which can be performed by the same truck in a period is high.

7. Computational results

In this section, we describe the experimental campaign carried out on several small/medium/large-sized instances, generated ad hoc for the MM-CDP-RDD. In particular, in Section 7.1, the procedure for generating the sets of instances is detailed, while in Section 7.2, numerical comparisons among the proposed CBC approaches and the two ILP formulations are discussed. All the proposed approaches were implemented in Java (in the Eclipse environment) and all the ILP models, including those of the CBC approaches, were solved by ILOGs CPLEX Concert Technology (version 12.9). It is noteworthy that we employed CPLEX default settings and let it use all cores/threads available. The experiments were run on a computer with a 64-bit operating system, 2.39 GHz processor and 32 GB of RAM.

7.1. Problem instances generation and parameter setting

In order to generate significant instances for the MM-CDP-RDD, the dataset proposed for the VRP, available at `https://www.sintef.no/projectweb/top/vrptw/`, was considered. In particular, we used the instance sets with 25 and 100 customers proposed by Solomon and the one with 200 customers of Gehring & Homberger.

A planning horizon of one week, excluding Sunday, was considered, i.e., $|T| = 6$. The number of trucks available is given as input and may change instance by instance. It consists in the minimum number of trucks that makes the instance feasible. It is determined by running the T-ILP with different fleet sizes, starting from a lower bound and increasing it of one unit until the instance becomes feasible. The lower bound is computed as the maximum of two terms. The first one is the ceiling of the ratio between the duration of all the trips serving pickup or delivery requests of 40 ft containers whose deadline is the first period and T_1^{max}. The second term is the ceiling of the ratio between the duration of all the trips serving pickup or delivery requests of 40 ft containers and T_3^{max}. The parameters T_1^{max}, T_2^{max} and T_3^{max} were set to 11, 16 and 40 hours, respectively. These values come from a real application involving a transportation company operating in the North-West of Italy, which was subject of a study developed by one of the authors of the paper. More details about the company cannot be shared due to a non disclosure agreement. The same values were used also in [34]. The truck loading capacity and the truck average speed were set to 40 ft and 60 km/h. The service times for 20 ft and 40 ft sized containers were set to 15 and 30 minutes, respectively.

For generating the RDDs at customers, the procedure proposed in [2] was used. Initializing t equal to 1, the release date of a customer was set to t. The t value was increased by 1 every time a new customer was considered until it reached $|T|$ (customers are considered sequentially). After which, t was again set to 1. The procedure ends when all customers have been considered. The due date of a customer was determined as the release date plus an integer parameter λ that can vary in $[0, 2]$. For feasibility reasons, if the release date of customer i is 6, then $T_i = \{6\}$, while if the release date of customer j is 5, with $\lambda = 2$, then $T_j = \{5, 6\}$. The distance between each pair of customers and between each customer and the terminal was computed using the Euclidean formulas, considering the coordinates indicated in the original files.

The demand of each customer was set as in the following. We first fixed the probability β of having a 40 ft sized demand and the probability γ of having a pickup request. Therefore, for each customer, two integer numbers, n' and n'', were randomly generated with uniform probability, both in $[0, 1]$. If, $n' \leq \beta$, then the request size was set to 40 ft; otherwise, to 20 ft. Moreover, if $n'' \leq \gamma$, the customer request was a pickup; otherwise, a delivery.

Finally, for each of the original six sets with 25, 100 and 200 customers, we randomly took one instance. This way, by varying λ in $\{0, 1, 2\}$, β in $\{0.25, 0.50, 0.75\}$ and γ in $\{0.25, 0.50, 0.75\}$, we generated 162 instances with 25, 100 and 200 customers, for a total number of 486 instances.

7.2. Comparisons among all the solution methods

In this section, we compare the results obtained by the proposed CBC approaches and the ones detected by both A-ILP and T-ILP. While the detailed results are reported in Appendix B, in the following subsections, we summarize them reporting their average values according to the instance sizes (25, 100 and 200 customers), the instance types (clustered C, random R and a mix of the two, RC) and different values of the parameters β, γ and λ.

7.2.1. Results for instances with 25 *customers*

In this section, we discuss the results obtained by A-ILP and T-ILP on the instances with 25 customers. We do not report the results obtained by the CBC approaches since, due to the very small size of the instances, T-ILP shows the best performance on average. In the following tables, column $|K|$ indicates the average number of trucks, columns CPU indicate the computational time required, columns NOS the number of instances where no optimal solution has been detected in the given CPU time limit (one hour) and TD denotes the total travel distance.

In particular, Table 2 compares the average values of A-ILP and T-ILP for each of the three subsets of instances (C, R and RC). In all subsets, both the approaches find the same average travel distance that increases passing from a subset to the next one. However, there is a significant difference in the average computational time, being about 129 seconds for the A-ILP against about 0.6 seconds required by T-ILP. Moreover, the average computational time required by T-ILP does not depend on the particular instance subset, which is always approximately equal to 0.6 seconds. Whereas A-ILP requires by far longer average CPU time for subsets C and RC compared to subset R and in two instances of the former it is even not able to detect the optimal solution. This may be due to the fact that in the instances of C and RC there may be more feasible trips.

Table 2: Average values on instances with 25 customers grouped by subset

		A-ILP		T-ILP		
Subset	$\|K\|$	CPU	NOS	TD	CPU	NOS
C	1	237.44	2	806.17	0.60	0
R	1	5.95	0	941.95	0.59	0
RC	2	142.44	2	1267.64	0.62	0

Table 3 shows the results after increasing the probability of having a 40 ft sized demand. We observe that, while the average computational time required by T-ILP does not depend on the value of β, being between about 0.5 and 0.6 seconds, for A-ILP it is significantly larger for $\beta = 0.25$ and in this case there are even 4 instances where it is not possible to find the optimal solution. This may be due to the fact that in this case the number of feasible solutions increases since in each trip more customers can be served compared to the cases where β is larger.

Table 4 shows the results after increasing the probability of having a pickup request. The best performance in terms of average total distance is reached in the case in which there are half import and half export customers. This behaviour may be due to the fact that when the import and export customers are balanced it is possible to save travel distance, serving first the import customers and then the export customers in the same trip. Concerning the CPU time, it is again possible to observe a more stable behaviour of T-ILP compared to A-ILP. Indeed, for the latter, the instances with less pickup requests ($\gamma = 0.25, 0.50$) are harder to solve, requiring on average a CPU time almost triple compared to the one for $\gamma = 0.75$. They include even 4 instances where it is not possible to detect the optimal solution. Finally, Table 5 shows the results after increasing the number of periods in which a customer can be served. In particular, it is remarkable that a significant saving on the average total distance (about 20%) is obtained by passing from $\lambda = 0$ (i.e., due and release dates coincide) to $\lambda = 1$ (i.e., due and release dates differ by one day). With $\lambda = 2$ (i.e., due and release dates differ by two days), the marginal improvement on the average total distance is only about 6%. This also justifies why it is not significant to increase the λ parameter by more than 2. Concerning the CPU time, one can again observe that T-ILP is more stable compared to A-ILP. Indeed, for the latter, the instances with $\lambda = 2$ are harder to solve requiring an average CPU time that is almost 30 times larger than the one for $\lambda = 1$. They include even 4 instances where it is not possible to detect the optimal solution. This behaviour is due to the fact that the larger λ is, the larger is the number of feasible solutions. Whereas T-ILP does not suffer of this problem thanks to the dominance rules applied in the trip generation.

7.2.2. Results for instances with 100 *customers*

In this section, the results obtained by the two ILP models are compared with those found by the CBC approaches, for the instances with 100

Table 3: Average values on instances with 25 customers grouped by β parameter

		A-ILP		T-ILP		
β	$\|K\|$	CPU	NOS	TD	CPU	NOS
0.25	1	379.56	4	904.91	0.73	0
0.50	1	5.81	0	1011.10	0.60	0
0.75	1	0.45	0	1099.75	0.48	0

Table 4: Average values on instances with 25 customers grouped by γ parameter

		A-ILP		T-ILP		
γ	$\|K\|$	CPU	NOS	TD	CPU	NOS
0.25	1	201.67	2	1036.79	0.55	0
0.50	1	139.86	2	945.18	0.62	0
0.75	1	44.29	0	1033.79	0.62	0

customers. In particular, the average number of trips generated is almost equal to $19,143$ in an average CPU time of about 1.25 seconds, while the number of cuts added is about 11 in CBC1, CBC2 and CBC3, 1 in CBC4 and CBC6 and 4 in CBC5. In all the CBC approaches, the average percentage of the total CPU time necessary to detect the cuts is negligible, being always lower than 0.023%. Moreover, in the following tables, the average CPU time (header CPU), the number of instances where no optimal solution has been detected (header NOS) and the average number of cuts added (header CUT) are reported, for each CBC approach.

Table 6 compares the results of A-ILP, T-ILP and CBC approaches for each of the three subsets of instances. For all the instance types, in almost the same number of cases, A-ILP is not able to find the optimal solution in the given CPU time, while T-ILP and CBC6 always can and the other CBCs fail only in very few instances. Concerning the computational times, T-ILP is more stable since its average computational time varies just between 23 and 26 seconds depending on the subset of instances, while CBCs are by far faster in the subset RC compared to the other subsets, especially for CBC1 and CBC2 where the improvement is about 90% and it is smaller for CBC6 being about 60% compared to subset C and almost insignificant compared to R. This behaviour is due to the fact that the average number of cuts generated is by far lower in the subset RC.

Table 7 shows the results after increasing the probability of having 40 ft sized demand. We observe that, for all methods, both the number of instances where no optimal solution has been detected and the average computational time significantly decrease as β increases. This behaviour is due to the fact that the larger β is, the larger is the number of 40 ft containers considered and the smaller is the number of feasible trips that are given as

Table 5: Average values on instances with 25 customers grouped by λ parameter

		A-ILP		T-ILP		
λ	$\|K\|$	CPU	NOS	TD	CPU	NOS
0	1	0.21	0	1151.13	0.41	0
1	1	13.98	0	957.47	0.58	0
2	1	371.63	4	907.16	0.81	0

input in T-ILP and CBCs or evaluated in the A-ILP.

Table 8 shows the results after increasing the probability of having a pickup request. Similarly to the results for the instances with 25 customers, the best average total distance is reached in the case where there are half import and half export customers (i.e., $\gamma = 0.50$). Concerning the CPU time, for A-ILP it is almost constant on the three values of γ, for T-ILP it is about 15% smaller for $\gamma = 0.5$ than for the other two values of γ, whereas for CBCs it is by far greater for $\gamma = 0.5$. Also the indicator NOS varies accordingly.

Finally, Table 9 shows the results after increasing the number of periods in which a customer can be served. Like for the instances with 25 customers, we can observe that the average total distance significantly decreases passing from $\lambda = 0$ to $\lambda = 1$ (in this case, by about 11%) and by far less from $\lambda = 1$ to $\lambda = 2$ (by about 3%). From the CPU time and NOS values, we can deduce that the instances with $\lambda = 1$ and especially those with $\lambda = 2$ are harder to solve for all the methods.

7.2.3. Results for instances with 200 *customers*

In this section, we discuss a comparison between the results obtained by T-ILP model and those of CBC6 in the instances with 200 customers, since the latter has provided the best performances compared to the other CBC approaches, in the previous set of instances.

Since in this new set of instances, T-ILP and CBC6 are not always able to detect the optimal solution, we report for each of them the column TD indicating the average travel distance only in the instances where they can find the optimal solution. Also, in this case we do not report the average percentage of the total CPU time necessary to detect the cuts since it is again negligible being always lower than 0.0001%.

From Table 10 we can observe that for both T-ILP and CBC6 the subsets C and R are more challenging than RC, since in each of them the number of cases where the methods are not able to detect an optimal solution is about double compared to RC and the average CPU time is about 26% and 72% larger for T-ILP and CBC6, respectively. Moreover, CBC6 is by far faster than T-ILP with an average relative percentage improvement of CPU time equal to about 65%.

From Table 11 we can observe that CBC6 is by far faster than T-ILP for $\beta = 0.25$, with a relative percentage improvement of CPU time equal to about 71%, while the percentage decreases to 67% for $\beta = 0.50$. For $\beta = 0.75$, CBC6 becomes slightly slower than T-ILP. This behaviour can be explained by the fact that in the latter case there are less feasible trips for the T-ILP making it easier to be solved. Moreover, as already observed for the instances with 25 and 100 customers, the average total distance increases as β increases.

Table 6: Average values on instances with 100 customers grouped by subset

		A-ILP		T-ILP			CBC1			CBC2			CBC3			CBC4			CBC5			CBC6		
Subset	$\|K\|$	CPU	NOS	TD	CPU	NOS	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT
C	4	3208.56	48	3569.73	25.69	0	91.71	1	11.81	91.69	1	11.81	103.72	1	11.69	164.44	2	0.48	76.45	1	6.70	13.01	0	1.59
R	4	3153.49	47	3088.60	23.21	0	116.02	1	20.63	115.93	1	20.63	123.70	1	20.31	204.79	3	1.44	76.54	1	6.50	6.46	0	0.13
RC	4	3151.00	46	4030.49	24.62	0	10.53	0	0.50	10.53	0	0.50	31.35	0	0.50	72.87	1	0.24	5.90	0	0.09	5.06	0	0.00

Table 7: Average values on instances with 100 customers grouped by β parameter

		A-ILP		T-ILP			CBC1			CBC2			CBC3			CBC4			CBC5			CBC6		
β	$\|K\|$	CPU	NOS	TD	CPU	NOS	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT
0.25	3	3600.00	54	3094.19	55.31	0	148.80	2	13.41	148.92	2	13.41	186.93	2	12.96	235.72	3	0.98	143.14	2	10.50	12.06	0	0.24
0.50	4	3600.00	54	3543.47	15.49	0	67.80	0	19.54	67.60	0	19.54	70.09	0	19.54	204.18	3	1.19	14.01	0	2.80	10.68	0	1.48
0.75	4	2313.04	33	4051.16	2.71	0	1.66	0	0.00	1.64	0	0.00	1.74	0	0.00	2.21	0	0.00	1.73	0	0.00	1.79	0	0.00

Table 8: Average values on instances with 100 customers grouped by γ parameter

		A-ILP		T-ILP			CBC1			CBC2			CBC3			CBC4			CBC5			CBC6		
γ	$\|K\|$	CPU	NOS	TD	CPU	NOS	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT
0.25	4	3222.01	48	3780.77	15.59	0	5.30	0	0.19	5.14	0	0.19	9.59	0	0.19	19.47	0	0.19	4.31	0	0.04	4.08	0	0.00
0.50	4	3153.89	47	3187.48	34.52	0	202.17	2	32.24	202.00	2	32.24	222.64	2	31.80	342.94	5	1.72	148.56	2	13.17	15.33	0	1.72
0.75	4	3137.15	46	3720.56	23.40	0	10.79	0	0.52	11.01	0	0.52	26.54	0	0.52	79.70	1	0.26	6.01	0	0.09	5.13	0	0.00

Table 9: Average values on instances with 100 customers grouped by λ parameter

		A-ILP		T-ILP			CBC1			CBC2			CBC3			CBC4			CBC5			CBC6		
λ	$\|K\|$	CPU	NOS	TD	CPU	NOS	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT	CPU	NOS	CUT
0	4	2429.35	36	3890.66	2.67	0	2.46	0	0.00	2.52	0	0.00	2.62	0	0.00	2.58	0	0.00	2.45	0	0.00	2.36	0	0.00
1	4	3483.69	51	3456.76	13.81	0	3.80	0	0.02	3.81	0	0.02	6.57	0	0.02	5.50	0	0.02	3.87	0	0.02	3.88	0	0.00
2	4	3600.00	54	3341.40	57.04	0	212.00	2	32.93	211.82	2	32.93	249.57	2	32.48	434.03	6	2.15	152.56	2	13.28	18.29	0	1.72

Table 10: Average values on instances with 200 customers grouped by subset

		T-ILP			CBC6			
Subset	$\|K\|$	TD	CPU	NOS	TD	CPU	NOS	CUT
C	9	12217.27	986.57	8	11975.45	344.55	4	4.72
R	10	12671.91	894.42	9	12431.27	358.56	4	4.59
RC	9	12523.97	731.20	5	12347.50	205.13	2	0.17

Table 11: Average values on instances with 200 customers grouped by β parameter

		T-ILP			CBC6			
β	$\|K\|$	TD	CPU	NOS	TD	CPU	NOS	CUT
0.25	8	10698.26	1735.32	18	10502.81	503.69	5	0.91
0.50	10	12374.64	723.40	4	12313.79	239.70	3	3.43
0.75	11	13741.40	153.47	0	13841.64	164.86	2	5.15

From Table 12 we can observe that CBC6 has a large relative percentage improvement of CPU time compared to T-ILP for $\gamma = 0.25$ and $\gamma = 0.75$, being in both cases equal to about 89%, while for $\gamma = 0.50$ it decreases to 38%. Unlike the instances with 100 customers, for T-ILP the CPU time is almost double for $\gamma = 0.5$ than for the other two values of γ and for the CBC6 it is even 10 times greater. Also, the indicator NOS varies accordingly. This behaviour may be due to the fact that in this case the number of feasible solutions, i.e., feasible trips, is larger compared to when $\gamma = 0.25$ or $\gamma = 0.75$. Indeed, according to Definition 1, there are only 2 trips of type 2-customer that serve either only pickup requests or only delivery requests over 7 possible trips. Furthermore, there are not 3-customer trips or 4-customer trips with this feature. Moreover, similarly to the results for the instances with 25 and 100 customers, the best average total distance is reached in the case where there are half import and half export customers (i.e., $\gamma = 0.50$).

Finally, from Table 13 we can observe that CBC6 has a large relative percentage improvement of CPU time compared to T-ILP for $\lambda = 1$ and $\lambda = 0$, being about 90% and 85%, respectively. For $\lambda = 2$ it decreases to 55%. Moreover, like for the instances with 25 and 100 customers, the average total distance decreases passing from $\lambda = 0$ to $\lambda = 1$, in this case by about 8%. Whereas the total distance increases by about 2% from $\lambda = 1$ to $\lambda = 2$ since in this case there are several instances that cannot be solved to optimality. Indeed, 91% of the instances that cannot be solved to optimality by T-ILP are concentrated to the case $\lambda = 2$ and all for CBC6, thus showing that this case is harder to solve. However, if for $\lambda = 1$ we compute the average total distance only in the instances that are solved to optimality for $\lambda = 2$, than it is equal to 12960.27. Therefore, for $\lambda = 2$ we improve the latter value by about 7%.

7.3. Impact of the valid inequalities on T-ILP and comparison with CBC6

In this section, the performances of both T-ILP and CBC6 are compared with those of three variants of T-ILP obtained by adding valid inequalities. In particular, these three variants are obtained by adding constraints (37)

Table 12: Average values on instances with 200 customers grouped by γ parameter

		T-ILP			CBC6			
γ	$\|K\|$	TD	CPU	NOS	TD	CPU	NOS	CUT
0.25	10	13203.15	699.60	5	12930.95	79.89	0	0.00
0.50	9	10992.04	1218.32	12	10782.15	754.43	10	9.48
0.75	10	13005.82	694.28	5	12772.60	73.92	0	0.00

Table 13: Average values on instances with 200 customers grouped by λ parameter

		T-ILP			CBC6			
λ	$\|K\|$	TD	CPU	NOS	TD	CPU	NOS	CUT
0	10	12792.72	46.05	0	12792.72	7.02	0	0.00
1	9	11953.93	714.27	2	11825.63	69.01	0	0.06
2	9	12749.83	1851.88	20	12113.99	832.21	10	9.43

only, (44)-(45) or (37)-(44)-(45), respectively. The goal of this analysis is to show that using the same constraints in a CBC framework is much more efficient than directly adding them to T-ILP as valid inequalities. Since computational times required by T-ILP to solve instances with 25 customers are already very small, it could be difficult to obtain an improvement adding valid inequalities. Therefore, we report the results obtained in larger instances with 100 and 200 customers. The results in tables 14-21 are discussed considering the instance types (R, C and RC) and by varying the value of the three parameters (β, γ and λ).

Globally, constraints (37) considerably slow down the model either if added singularly or in combination with constraints (44)-(45). This behaviour is more evident on the instances with 200 customers (Tables 18-21) for which adding them singularly or in combination with (44)-(45) yields a very strong negative impact on computational times required to solve the model to optimality.

In contrast, on the instances with 100 customers (Tables 14-17), adding constraints (44)-(45) in T-ILP produces a very little speed up (2.32%) against the very large improvement (66.61%) provided by using them in a CBC framework (CBC6). On the instances with 200 customers, instead, adding them in T-ILP fails to provide a speed-up too, yielding slightly higher computational times (0.34%) on average. In contrast, using the same constraints in a CBC framework (CBC6) reduces the computational times by 65%.

However, on the instances with 200 customers, the improvement provided by the addition of valid inequalities is not systematic for both the instance layouts and values of β, γ and λ, differently from what happens in 100 customers instances. Constraints (44)-(45), used as valid inequalities, are effective in R and C type instances but not for RC ones. Similarly, they are effective only for $\beta = 0.5$ and for $\gamma = 0.75$ while they yield larger computational times for the other values of both parameters. For what concerns the parameter λ, all the tested valid inequalities provide a slight speed-up when $\lambda = 0$. With $\lambda = 1$, constraints (37) yield a very strong negative impact on computational times required to solve the model to optimality, either if added singularly or in combination with (44)-(45). In contrast, con-

Table 14: Average values on instances with 100 customers grouped by subset

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
Subset	$\|K\|$	CPU	CPU	CPU	CPU	CPU
C	4	25.69	35.94	24.72	34.99	13.01
R	4	23.21	32.75	22.10	32.15	6.46
RC	4	24.62	42.18	24.98	42.49	5.06
AVG	4	24.50	36.96	23.93	36.54	8.18

Table 15: Average values on instances with 100 customers grouped by β parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
β	$\|K\|$	CPU	CPU	CPU	CPU	CPU
0.25	3	55.31	90.26	53.73	89.08	12.06
0.50	4	15.49	17.92	15.26	17.69	10.68
0.75	4	2.71	2.70	2.81	2.86	1.79
AVG	4	24.50	36.96	23.93	36.54	8.18

straints (44)-(45) provide a small speed-up. Finally, in instances with $\lambda = 2$, we report a negative effect of all the tested valid inequalities. This effect is small for constraints (44)-(45) but it is huge for constraints (37). CBC6 is instead faster than the model for all the values of the analyzed parameters except for $\beta = 0.75$, for which it is slightly slower (around 7%).

In conclusion, the results in tables 14-21 clearly show the advantage of adopting the proposed decomposition approach.

8. Conclusions and future work

In this paper, we introduced a new variant of the Container Drayage Problem (CDP), i.e., the Multi-period Multi-trip CDP with Release and Due Dates (MM-CDP-RDD), in which the planning horizon is divided into discrete time periods (days), each truck can perform more than one trip in a period and customers have to be served in specific periods (Release and Due Dates, RDDs).

We addressed the MM-CDP-RDD through exact approaches. In particular, we proposed an Arc-based Integer Linear Programming (A-ILP) model, representing the problem on a direct graph, properly pre-processed in order to remove infeasible arcs considering both the truck load capacity and the RDDs. Due to both the maximum truck load capacity and the RDDs, the number of feasible trips may be very limited. Therefore, we also designed a Trip-based solution approach where all the feasible non-dominated trips are first determined and then, a Trip-based ILP (T-ILP) model is solved. Moreover, because the number of feasible non-dominated trips may become high

Table 16: Average values on instances with 100 customers grouped by γ parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
γ	$\|K\|$	CPU	CPU	CPU	CPU	CPU
0.25	4	15.59	19.99	15.42	19.83	4.08
0.50	4	34.52	55.96	32.96	54.94	15.33
0.75	4	23.40	34.93	23.42	34.85	5.13
AVG	4	24.50	36.96	23.93	36.54	8.18

Table 17: Average values on instances with 100 customers grouped by λ parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
λ	$\lvert K\rvert$	CPU	CPU	CPU	CPU	CPU
0	4	2.67	2.57	3.02	2.86	2.36
1	4	13.81	16.40	13.62	16.41	3.88
2	4	57.04	91.91	55.16	90.36	18.29
AVG	4	24.50	36.96	23.93	36.54	8.18

Table 18: Average values on instances with 200 customers grouped by subset

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
Subset	$\lvert K\rvert$	CPU	CPU	CPU	CPU	CPU
C	9	986.57	5722.13	923.29	5104.50	344.55
R	10	894.42	5580.44	871.11	5605.92	358.56
RC	9	731.20	3566.02	826.56	3969.91	205.13
AVG	9	870.73	4956.20	873.65	4893.45	302.75

Table 19: Average values on instances with 200 customers grouped by β parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
β	$\lvert K\rvert$	CPU	CPU	CPU	CPU	CPU
0.25	8	1735.32	13701.76	1801.83	13478.52	503.69
0.50	10	723.40	952.72	624.17	999.04	239.70
0.75	11	153.47	214.11	194.96	202.78	164.86
AVG	10	870.73	4956.20	873.65	4893.45	302.75

Table 20: Average values on instances with 200 customers grouped by γ parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
γ	$\lvert K\rvert$	CPU	CPU	CPU	CPU	CPU
0.25	10	699.60	4000.76	748.65	3519.35	79.89
0.50	9	1218.32	7584.07	1225.35	8299.77	754.43
0.75	10	694.28	3283.76	646.96	2861.22	73.92
AVG	10	870.73	4956.20	873.65	4893.45	302.75

Table 21: Average values on instances with 200 customers grouped by λ parameter

		T-ILP	T-ILP+(37)	T-ILP+(44)+(45)	T-ILP+(37)+(44)+(45)	CBC6
λ	$\lvert K\rvert$	CPU	CPU	CPU	CPU	CPU
0	10	46.05	46.33	35.12	41.89	7.02
1	9	714.27	1437.72	656.31	1307.17	69.01
2	9	1851.88	13384.55	1929.53	13331.28	832.21
AVG	9	870.73	4956.20	873.65	4893.45	302.75

in medium/large-sized instances, we also proposed six different Combinatorial Benders' Cuts (CBC) approaches for efficiently solving the T-ILP model, by defining, beyond the traditional CBC, ad hoc both valid inequalities and stronger cuts.

The proposed approaches were tested on 486 instances, with a different number of customers (25, 100 and 200), derived from those proposed for the VRP. In 25-customers instances, the T-ILP model outperformed all the other approaches including the A-ILP model that reached the CPU time limit in 4 cases. In 100-customers instances, on average, the sixth CBC method (CBC6) outperformed the others, although it is shown that, instance by instance, it did not always dominate the others. In the instances with 200 customers, CBC6 strongly outperformed the T-ILP model closing to optimality almost all the instances in an average CPU time that is about 65% less than that required by the T-ILP model (302.75 vs 870.73 seconds). Finally, we compared the results of both the T-ILP model and the CBC6 with those of three variants of the T-ILP model obtained by adding valid inequalities. These three new variants resulted to be strongly outperformed by the CBC6 and only one of them obtained comparable results with those of the T-ILP model.

Future developments on this subject could include the extension of the multi-period multi-trip CDP to the CDP with Time Windows (TWs). This problem will become much more complex to solve because the T-ILP model will be not anymore a packing based formulation but a scheduling based one. In fact, when service time windows are addressed, it becomes significant not only the assignment of trips to vehicles, but also the order in which the trips are carried out. From a CBC point of view, the further complexity added by the introduction of TWs could be managed by the SP problem, letting unchanged the MP. Another possible extension could address different delivery strategies, as the *stay with* policy, in which the driver waits at customer location until the container is filled/emptied. This will only impact on the feasible non-dominated trips generation routine, but not on the T-ILP nor on the CBC approach. From a methodological point of view, the proposed CBCs can be applied to other combined vehicle routing and scheduling problems. Furthermore, it is worth noting that our exact approach can be easily turned into a heuristic method, providing to the model only a subset of the feasible non-dominated trips.

References

[1] S. Akpinar, A. Elmi, and T. Bektaş. Combinatorial benders cuts for assembly line balancing problems with setups. *European Journal of Operational Research*, 259(2):527–537, 2017. doi: `https://doi.org/10.1016/j.ejor.2016.11.001`.

[2] C. Archetti, O. Jabali, and M. G. Speranza. Multi-period vehicle routing problem with due dates. *Computers & Operations Research*, 61:122–134, 2015. doi:`https://doi.org/10.1016/j.cor.2015.03.014`.

[3] L. Bai and P A. Rubin. Combinatorial benders cuts for the minimum tollbooth problem. *Operations Research*, 57:1510–1522, 2009. doi: `https://doi.org/10.1287/opre.1090.0694`.

[4] J. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962. doi: `https://doi.org/10.1007/BF01386316`.

[5] K. Braekers, A. Caris, and G. K. Janssens. Bi-objective optimization of drayage operations in the service area of intermodal terminals. *Transportation Research Part E: Logistics and Transportation Review*, 65:50–69, 2014. doi: `https://doi.org/10.1016/j.tre.2013.12.012`.

[6] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313, 2016. doi:`https://doi.org/10.1016/j.cie.2015.12.007`.

[7] J.C.S. Brandao and A. Mercer. The multi-trip vehicle routing problem. *Journal of the Operational research society*, 49(8):799–805, 1998. doi:`https://doi.org/10.1057/palgrave.jors.2600595`.

[8] M. Bruglieri, S. Mancini, and O. Pisacane. A combinatorial benders' cuts based exact method for the multi-trip containers drayage problem. In *Odysseus 2018, Seventh International Workshop on Freight Transportation and Logistics*, 2018.

[9] C. Caballini, I. Rebecchi, and S. Sacone. Combining multiple trips in a port environment for empty movements minimization. *Transportation Research Procedia*, 10:694–703, 2015. doi:`https://doi.org/10.1016/j.trpro.2015.09.023`.

[10] J.X. Cao, D-H. Lee, J.H. Chen, and Q. Shi. The integrated yard truck and yard crane scheduling problem: Benders decomposition-based methods. *Transportation Research Part E: Logistics and Transportation Review*, 46(3):344–353, 2010. doi: `https://doi.org/10.1016/j.tre.2009.08.012`.

[11] A. Caris and G. K. Janssens. A local search heuristic for the pre-and end-haulage of intermodal container terminals. *Computers & Operations Research*, 36(10):2763–2772, 2009. doi: `https://doi.org/10.1016/j.cor.2008.12.007`.

[12] D. Cattaruzza, N. Absi, and D. Feillet. Vehicle routing problems with multiple trips. *4OR*, 14(3):223–259, 2016. doi: `https://doi.org/10.1007/s10288-016-0306-2`.

[13] D. Cattaruzza, N. Absi, D. Feillet, and T. Vidal. A memetic algorithm for the multi trip vehicle routing problem. *European Journal of Operational Research*, 236(3):833–848, 2014. doi:`https://doi.org/10.1016/j.ejor.2013.06.012`.

[14] J.H. Chen, D-H. Lee, and J.X. Cao. A combinatorial benders cuts algorithm for the quayside operation problem at container terminals. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):266–275, 2012. doi: `https://doi.org/10.1016/j.tre.2011.06.004`.

[15] G. Codato and M. Fischetti. Combinatorial benders' cuts for mixed-integer linear programming. *Operations Research*, 54:756–766, 2006. doi: `https://doi.org/10.1287/opre.1060.0286`.

[16] J-F. Côté, M. Dell'Amico, and M. Iori. Combinatorial benders' cuts for the strip packing problem. *Operations Research*, 62(3):643–661, 2014. doi: `https://doi.org/10.1287/opre.2013.1248`.

[17] I. Dayarian, T. G. Crainic, M. Gendreau, and W. Rei. A branch-and-price approach for a multi-period vehicle routing problem. *Computers & Operations Research*, 55:167–184, 2015. doi:`https://doi.org/10.1016/j.cor.2014.06.004`.

[18] R. Elmasri and S. Navathe. Fundamentals of database systems. 7. kansainvälinen p, 2016.

[19] M.P. Fanti, A. Locatelli, G. Stecco, and W. Ukovich. A new ilp formulation for the multi-day container drayage problem. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019. doi: `https://doi.org/10.1109/SMC.2019.8914439`.

[20] P. Francis, G. Zhang, and K. Smilowitz. Improved modeling and solution methods for the multi-resource routing problem. *European Journal of Operational Research*, 180(3):1045–1059, 2007. doi:`https://doi.org/10.1016/j.ejor.2006.03.054`.

[21] J. Funke and H. Kopfer. A model for a multi-size inland container transportation problem. *Transportation Research Part E: Logistics and Transportation Review*, 89:70–85, 2016. doi:`https://doi.org/10.1016/j.tre.2016.02.010`.

[22] A. Geoffrion. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972. doi: https://doi.org/10.1007/bf00934810.

[23] A. Ghezelsoflu, M. Di Francesco, A. Frangioni, and P. Zuddas. A set-covering formulation for a drayage problem with single and double container loads. *Journal of Industrial Engineering International*, pages 1–12, 2018. doi:https://doi.org/10.1007/s40092-018-0256-8.

[24] C. Groër, B. Golden, and E. Wasil. The consistent vehicle routing problem. *Manufacturing & service operations management*, 11(4):630–643, 2009. doi:https://doi.org/10.1287/msom.1080.0243.

[25] V.C. Hemmelmayr, K.F. Doerner, and R.F. Hartl. A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802, 2009. doi:https://doi.org/10.1016/j.ejor.2007.08.048.

[26] J. Homberger and H. Gehring. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 162(1):220–238, 2005. doi: https://doi.org/10.1016/j.ejor.2004.01.027.

[27] J. Hooker and G. Ottoson. Logic-based benders decomposition. *Mathematical Programming*, 96:33–60, 2003. doi: https://doi.org/10.1007/s10107-003-0375-9.

[28] A. Imai, E. Nishimura, and J. Current. A lagrangian relaxation-based heuristic for the vehicle routing with full container load. *European journal of operational research*, 176(1):87–105, 2007. doi:https://doi.org/10.1016/j.ejor.2005.06.044.

[29] H. Jula, M. Dessouky, P. Ioannou, and A. Chassiakos. Container movement by trucks in metropolitan networks: modeling and optimization. *Transportation Research Part E: Logistics and Transportation Review*, 41(3):235–259, 2005. doi: https://doi.org/10.1016/j.tre.2004.03.003.

[30] I. Karaoğlan. A branch-and-cut algorithm for the vehicle routing problem with multiple use of vehicles. *International Journal of Lean Thinking*, 6(1):21–46, 2015.

[31] M. Lai, M. Battarra, M. Di Francesco, and P. Zuddas. An adaptive guidance meta-heuristic for the vehicle routing problem with splits and clustered backhauls. *Journal of the Operational Research Society*, 66(7):1222–1235, 2015. doi: https://doi.org/10.1057/jors.2014.123.

[32] M. Lai, T. G. Crainic, M. Di Francesco, and P. Zuddas. An heuristic search for the routing of heterogeneous trucks with single and double container loads. *Transportation Research Part E: Logistics and Transportation Review*, 56:108–118, 2013. doi: `https://doi.org/10.1016/j.tre.2013.06.001`.

[33] S. Mancini, M. Ciavotta, and C. Meloni. The multiple multidimensional knapsack with family-split penalties. *European Journal of Operational Research*, in press, 2019. doi: `https://doi.org/10.1016/j.ejor.2019.07.052`.

[34] S. Mancini and J. Gonzalez-Feliu. A mip formulation for a combined vehicle routing and driver scheduling problem with real life constraints. In *OPT-i 2014 - 1st International Conference on Engineering and Applied Sciences Optimization, Proceedings*, pages 760–773, 2014.

[35] N. Marković, Ž. Drobnjak, and P. Schonfeld. Dispatching trucks for drayage operations. *Transportation Research Part E: Logistics and Transportation Review*, 70:99–111, 2014. doi: `https://doi.org/10.1016/j.tre.2014.06.016`.

[36] A. Mingozzi, R. Roberti, and P. Toth. An exact algorithm for the multitrip vehicle routing problem. *INFORMS Journal on Computing*, 25(2):193–207, 2013. doi:`https://doi.org/10.1287/ijoc.1110.0495`.

[37] P. Nagl. Longer combination vehicles (lcv) for asia and pacific region: Some economic implications. *UNESCAP Working Papers n2, UNITED NATIONS*, 2007.

[38] S. N. Parragh, K. F. Doerner, and R. F. Hartl. A survey on pickup and delivery models part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008. doi: `https://doi.org/10.1007/s11301-008-0036-4`.

[39] S. N. Parragh, K. F. Doerner, and R. F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1):21–51, 2008. doi:`https://doi.org/10.1007/s11301-008-0036-4`.

[40] R.J. Petch and S. Salhi. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Applied Mathematics*, 133(1-3):69–92, 2003. doi:`https://doi.org/10.1016/S0166-218X(03)00434-7`.

[41] L. B. Reinhardt, D. Pisinger, S. Spoorendonk, and M. M. Sigurd. Optimization of the drayage problem using exact methods. *INFOR: Information Systems and Operational Research*, 54(1):33–51, 2016. doi: `https://doi.org/10.1080/03155986.2016.1149919`.

[42] Mancini S. A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based matheuristic. *Transportation Research Part C: Emerging Technologies*, 70:100–112, 2016. doi:https://doi.org/10.1016/j.trc.2015.06.016.

[43] F. Schulte, E. Lalla-Ruiz, R. G. González-Ramírez, and S. Voß. Reducing port-related empty truck emissions: a mathematical approach for truck appointments with collaboration. *Transportation Research Part E: Logistics and Transportation Review*, 105:195–212, 2017. doi:https://doi.org/10.1016/j.tre.2017.03.008.

[44] S. Shiri and N. Huynh. Optimization of drayage operations with time-window constraints. *International Journal of Production Economics*, 176:7–20, 2016. doi:https://doi.org/10.1016/j.ijpe.2016.03.005.

[45] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265, 1987. doi:https://doi.org/10.1287/opre.35.2.254.

[46] D. W. Song and P. Panayides. *Maritime logistics: A complete guide to effective shipping and port management.* Kogan Page Publishers, 2012.

[47] Z.C. Taşkin and M. Cevik. Combinatorial benders cuts for decomposing imrt fluence maps using rectangular apertures. *Computers & Operations Research*, 40(9):2178–2186, 2013. doi:https://doi.org/10.1016/j.cor.2011.07.005.

[48] É. D Taillard, G. Laporte, and M. Gendreau. Vehicle routeing with multiple use of vehicles. *Journal of the Operational research society*, 47(8):1065–1070, 1996. doi:https://doi.org/10.1057/jors.1996.133.

[49] J. Verstichel, J. Kinable, P. De Causmaecker, and G. Vanden Berghe. A combinatorial benders decomposition for the lock scheduling problem. *Computers & Operations Research*, 54:117–128, 2015. doi:https://doi.org/10.1016/j.cor.2014.09.007.

[50] M. Vidović, M. Nikolić, and D. Popović. Two mathematical formulations for the containers drayage problem with time windows. *International Journal of Business Science and Applied Management*, 7(3):23–32, 2012.

[51] X. Wang and A. C. Regan. Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, 36(2):97–112, 2002. doi:https://doi.org/10.1016/S0965-8564(00)00037-9.

[52] Z. Xue, W.H. Lin, L. Miao, and C. Zhang. Local container drayage problem with tractor and trailer operating in separable mode. *Flexible Services and Manufacturing Journal*, 27(2-3):431–450, 2015. doi:https://doi.org/10.1007/s10696-014-9190-2.

[53] Z. Xue, C. Zhang, W.H. Lin, L. Miao, and P. Yang. A tabu search heuristic for the local container drayage problem under a new operation mode. *Transportation Research Part E: Logistics and Transportation Review*, 62:136–150, 2014. doi:https://doi.org/10.1016/j.tre.2013.12.007.

[54] Z. Xue, C. Zhang, P. Yang, and L. Miao. A combinatorial benders cuts algorithm for the local container drayage problem. *Mathematical Problems in Engineering*, 2015. doi:https://doi.org/10.1155/2015/134763.

[55] J. You, L. Miao, C. Zhang, and Z. Xue. A generic model for the local container drayage problem using the emerging truck platooning operation mode. *Transportation Research Part B: Methodological*, 133:181–209, 2020. doi:https://doi.org/10.1016/j.trb.2019.12.009.

[56] R. Zhang, C. Huang, and J. Wang. A novel mathematical model and a large neighborhood search algorithm for container drayage operations with multi-resource constraints. *Computers & Industrial Engineering*, 139, 2020. doi:https://doi.org/10.1016/j.cie.2019.106143.

[57] R. Zhang, W. Y. Yun, and H. Kopfer. Heuristic-based truck scheduling for inland container transportation. *OR spectrum*, 32(3):787–808, 2010. doi:https://doi.org/10.1007/s00291-010-0193-4.

[58] R. Zhang, W. Y. Yun, and I. Moon. A reactive tabu search algorithm for the multi-depot container truck transportation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(6):904–914, 2009. doi:https://doi.org/10.1016/j.tre.2009.04.012.

[59] R. Zhang, W. Y. Yun, and I. Moon. Modeling and optimization of a container drayage problem with resource constraints. *International Journal of Production Economics*, 133(1):351–359, 2011. doi:https://doi.org/10.1016/j.ijpe.2010.02.005.

Appendix A. Pseudo-codes and their computational complexity

In this Appendix, the pseudo-codes of the routines implemented and invoked by Algorithm 1 are described. Moreover, details on their computational complexity are also given. The set of feasible and non-dominated trips Π is implemented as a hash table data structure [18] to make the implementation of the *Dominated* routine more efficient. The key of Π is a string, i.e., the trip id. In fact, the routine *ID* (invoked in Algorithm 4) receives a trip π and returns its *id* defined as in the following. The ids of the customers served in π are first converted into a string. The terminal id 0 is then added as the first and the last character of the string. This string represents the identifier of the trip, generated by following the customer ids in a lexicographic way. For example, if the trip serves the customers $\{i, j, k\}$ then its identifier becomes the string $'0ijk0'$. This way, the id of the trip $\{k, i, j\}$ coincides with the id of the trip $\{i, j, k\}$. Since, in the worst case, a trip π can serve 4 customers, the time complexity of the *ID* routine is constant, being at most $4log4$.

The routine *create* initializes the hash map Π and the routine *createTrip* creates a trip without customers $\pi := \{0, 0\}$. Then, the set of customers served in π, i.e., C_π is empty and the set of periods ($days$) in which the trip can be performed is initially T. The distance and the duration of the trip are set to zero as well as the parameter $totalTime$ counting the service times of the customers served in the trip. The parameter len denotes the number of customers currently served in the trip and it is initially set to zero. Finally, the parameter l identifies the last customer served in the trip and it is initially set to the terminal.

Then, the recursive routine *IncreasedTrip* is invoked for determining all the feasible non-dominated trips of increasing length. It is worth remarking that the length of a trip represents the number of customers served. Given a feasible non-dominated trip π, with distance d_π, duration t_π, periods $days$ in which it can be performed, set of its customers C_π and length len, this routine tries to generate a new trip π' obtained from π by adding a new customer as the last served (routine *put*). Therefore, the new trip π' has a length equal to $len + 1$.

In order to check the first two feasibility conditions reported in Definition 2, the routine *Feasible* (outlined in Algorithm 5) is invoked. For this purpose, let i be a new customer to add. Then, it computes the intersection between $days$ and T_i and verifies if the new path π' satisfies the first two feasibility rules given in Definition 2. Therefore, the time complexity of the routine *Feasible* is equal, in the worst case, to $O(|T|)$, since the periods are stored in a sorted tree data structure.

If the length of π' is equal to 1 and π' is feasible, it is added to the set Π through the routine *insert*. The latter uses the id of π' as a key to add it in the hash table with its distance. Then, the *IncreasedTrip* is invoked again to increase the length of π'. Otherwise, if the length of π' is greater than 1, the algorithm firstly verifies if π' respects one of the types given in Definition 1 (routine *Capacity* outlined in Algorithm 6), receiving the requests of the customers served by π' and its length. The routine *Req* receives a trip π' and returns an array ρ containing the demands of the customers by following the order according to which they are served in π'. It also verifies that π' is not dominated (routine *Dominated* outlined in Algorithm 7).

The routine *Dominated* returns TRUE if the trip π' is dominated, according to the dominance rules given in Definition 3; FALSE, otherwise. In particular, the routine *search* looks for a given trip id (key) and returns *null* if such a trip does not belong to Π, its total distance, otherwise. Since we are implementing Π as a hash map data structure, the complexity of the *Dominated* routine is constant. If the length of the trip is less than 4, then the routine *IncreasedTrip* is invoked again.

Algorithm 4 IncreasedTrip

```
Input: $\pi$, $C_\pi$, $d_\pi$,$t_\pi$, $totalTime$, $days$, $len$, $l$, $\Pi$.
Output: $\Pi$
1: for ($i \in C \setminus C_\pi$) do
2:     $\pi' := put(\pi, i)$
3:     $d_{\pi'} := d_\pi - d_{l0} + d_{li} + d_{i0}$
4:     $totalTime := totalTime + 2\tau_i$
5:     $t_{\pi'} := totalTime + d_{\pi'}/v$
6:     $len := len + 1$
7:     $l := i$
8:     $C_{\pi'} := C_\pi \cup \{i\}$
9:     $days := Feasible(t_{\pi'}, T_1^{max}, days, T_i)$
10:    $id_{\pi'} := ID(\pi')$
11:    if ($days \neq \varnothing$) then
12:        if ($len = 1$) then
13:            $insert(\Pi, id_{\pi'}, d_{\pi'})$
14:            $\Pi := IncreasedTrip(\pi', C_{\pi'}, d_{\pi'}, t_{\pi'}, totalTime, days, len, l, \Pi)$
15:        else
16:            if ($Capacity(Req(\pi'), len) \wedge !Dominated(id_{\pi'}, d_{\pi'}, \Pi)$) then
17:                $insert(\Pi, id_{\pi'}, d_{\pi'})$
18:                if ($len < 4$) then
19:                    $\Pi := IncreasedTrip(\pi', C_{\pi'}, d_{\pi'}, t_{\pi'}, totalTime, days, len, l, \Pi)$
20:                end if
21:            end if
22:        end if
23:    end if
24: end for
```

Algorithm 5 Feasible

```
Input: $\tau_\pi$, $T_1^{max}$, $days$, $T_i$ with $i \in C$.
Output: $days \cap T_i$ or $\varnothing$
1: $days := days \cap T_i$
2: if ($\tau_\pi \leq T_1^{max} \wedge days \neq \varnothing$) then
3:     return $days$
4: else
5:     return $\varnothing$
6: end if
```

Appendix B. Detailed result comparisons

In the following tables, *instance* denotes the instance name. We point out that we keep the original instance name in the root name, adding information about parameters β, γ and λ, after it. For example, $c107\_25\_25\_0$

Algorithm 6 Capacity

```
Input: $\rho$, $len$.
Output: TRUE or FALSE
if $(len = 4 \wedge \rho[1] = -20 \wedge \rho[4] = +20 \wedge ((\rho[2] = +20 \wedge \rho[3] = -20) \vee (\rho[2] = -20 \wedge \rho[3] = +20)))$ then
    return $TRUE$
else
    if $(len = 3)$ then
        if $(\rho[1] = -40 \wedge \rho[2] = +20 \wedge \rho[3] = +20)$ then
            return $TRUE$
        else
            if $(\rho[1] = +20 \wedge \rho[2] = -20 \wedge \rho[3] = +20)$ then
                return $TRUE$
            else
                if $(\rho[1] = -20 \wedge \rho[2] = -20 \wedge (\rho[3] = 20 \vee \rho[3] = +40))$ then
                    return $TRUE$
                else
                    if $(\rho[1] = +20 \wedge \rho[2] = -20 \wedge (\rho[3] = 20 \vee \rho[3] = -20))$ then
                        return $TRUE$
                    end if
                end if
            end if
        end if
    else
        if $(len = 2)$ then
            if $(\rho[1] = -20 \wedge \rho[2]! = -40)$ then
                return $TRUE$
            else
                if $(\rho[1] = -40 \wedge \rho[2] > 0)$ then
                    return $TRUE$
                else
                    if $(\rho[1] = +20 \wedge (\rho[2] = 20 \vee \rho[2] = -20))$ then
                        return $TRUE$
                    end if
                end if
            end if
        end if
    end if
end if
return $FALSE$
```

Algorithm 7 Dominated

```
Input: $id_\pi$, $d_\pi$, $\Pi$.
Output: TRUE or FALSE
$distance := hash\_search(\Pi, id_\pi)$
if $(distance = null \vee distance > d_\pi)$ then
    return $FALSE$
end if
return $TRUE$
```

means that the instance $c107$ has been elaborated with $\beta = 0.25$, $\gamma = 0.25$ and finally, $\lambda = 0$. Column $|K|$ indicates the minimum number of trucks to make the instance feasible. Columns TD and CPU denote the total travel distance and the CPU time required, respectively. Column $GAP(\%)$ reports the percentage ILP GAP obtained by CPLEX solving the ILP models and the symbol " † " is always used to mark the cases in which the CPU time limit of one hour is reached. Finally, the row AVG1 reports, for each approach, the average values computed for the instances that are solved to optimality by the approach considered. The row AVG2 shows the average values computed for only the instances that are solved to optimality by all the approaches compared.
Tables B.22-B.23 show that T-ILP is suitable to close to optimality in all the instances in an average total CPU time of 0.60 seconds against $1,005.25$ seconds required by A-ILP. It is worth remarking that the total CPU time required by T-ILP counts also the time for generating all the feasible non-dominated trips. The number of trips generated is equal to 174 in about 0.11 seconds, on average. Moreover, A-ILP reaches the CPU time limit of $3,600$ seconds in four instances that correspond to cases in which the value of λ has been set to the maximum one (i.e., 2). Indeed, in these four instances, although A-ILP finds the optimal solution, the solver is not able to certify its optimality within the CPU time limit. In fact, those percentage gaps are high only because the A-ILP bound turns out to be not tight enough, as is often the case for arc-based mathematical programming formulations.

Tables B.24-B.25 show that on average CBC6 outperforms all the other approaches and the two ILP models, solving all the instances to optimality in the smallest average CPU time of 8.13 seconds. The other CBC approaches do not solve to optimality two instances (i.e., $c106\_25\_50\_2$ and $r207\_25\_50\_2$). For these instances, the values of the objective function ($2,517.37$ and $2,195.69$, respectively) are lower than the optimal ones ($2,521.93$ and $2,199.11$, respectively) because they are computed in the last solutions found by the MP and are infeasible since the CPU time limit of one hour has been reached. Moreover, CBC4 does not solve to optimality also the instances $c106\_50\_50\_2$, $r109\_50\_50\_2$, $r207\_50\_50\_2$ and $rc108\_25\_75\_2$. This is justified by the fact that, at each iteration, it needs to check not only the feasibility of the current solution found by MP4 but also needs to find α^*. However, observing the results instance by instance, we can conclude that there is not an approach that always dominates the others. Figure B.4 shows the percentage of instances in which each approach dominates the others. Instead, the A-ILP model closes to optimality only 12.96% of instances, while for 28.40% of instances, the solver is not able to certify the optimality in 1 hour. Finally, in 58.64% of instances, the A-ILP model is not suitable to find even a feasible solution in 1 hour (cases in which TD value is indicated by *NFS*, i.e., No Feasible Solution, in the tables).

Tables B.26-B.27 show that on average CBC6 closes the instances to

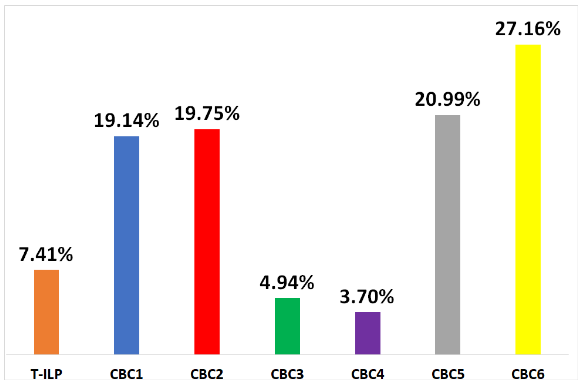

Figure B.4: Percentage of instances in which each approach dominates all the others

optimality in an average CPU time that is about 65% less than that required by the T-ILP model. In 10% of the instances, the T-ILP model is not able to find even a feasible solution (cases marked in the tables with the acronym NFS in the TD column) in the time limit of one hour. Indeed, in these cases, the average number of trips generated is more than 1 million. In all instances, the average number of trips is about $242,688$ generated in an average CPU time of about 13 seconds. Moreover, in 3% of the instances, the T-ILP model reaches the CPU time limit of one hour and therefore, it provides only a feasible solution. Concerning CBC6, in the tables, the cases marked with the symbol '$*$' (about 6% of the instances) are those in which we report the solution found by the MP at the final iteration, i.e., not a feasible solution for the problem, since the CPU time limit of one hour is reached. CBC6 is able to solve to optimality all instances except two, for which it provides only infeasible solutions (while the T-ILP model is able to solve them to optimality).

Table B.22: Results on instances with 25 customers: Part I

		A-ILP			T-ILP		
Instance	$\|K\|$	TD	CPU	GAP(%)	TD	CPU	GAP(%)
c107_25_25_0	1	768.24	0.14	0.00	768.24	0.56	0.00
c107_25_25_1	1	595.29	32.34	0.00	595.29	0.64	0.00
c107_25_25_2	1	559.47	†	8.80	559.47	0.77	0.00
c107_25_50_0	1	898.05	0.13	0.00	898.05	0.39	0.00
c107_25_50_1	1	659.93	5.50	0.00	659.93	0.56	0.00
c107_25_50_2	1	634.78	14.08	0.00	634.78	0.75	0.00
c107_25_75_0	1	792.76	0.25	0.00	792.76	0.61	0.00
c107_25_75_1	1	603.53	204.35	0.00	603.53	1.41	0.00
c107_25_75_2	1	539.03	1844.99	0.00	539.03	1.48	0.00
c107_50_25_0	1	908.66	0.14	0.00	908.66	0.38	0.00
c107_50_25_1	1	761.44	1.08	0.00	761.44	0.42	0.00
c107_50_25_2	1	725.07	11.51	0.00	725.07	0.59	0.00
c107_50_50_0	1	794.07	0.11	0.00	794.07	0.41	0.00
c107_50_50_1	1	586.36	0.38	0.00	586.36	0.86	0.00
c107_50_50_2	1	557.80	8.92	0.00	557.80	0.77	0.00
c107_50_75_0	1	873.61	0.09	0.00	873.61	0.36	0.00
c107_50_75_1	1	810.24	0.69	0.00	810.24	0.47	0.00
c107_50_75_2	1	794.02	12.22	0.00	794.02	1.13	0.00
c107_75_25_0	1	914.85	0.09	0.00	914.85	0.34	0.00
c107_75_25_1	1	858.66	0.14	0.00	858.66	0.39	0.00
c107_75_25_2	1	788.36	0.50	0.00	788.36	0.59	0.00
c107_75_50_0	1	883.23	0.09	0.00	883.23	0.34	0.00
c107_75_50_1	1	701.40	0.17	0.00	701.40	0.38	0.00
c107_75_50_2	1	661.46	0.27	0.00	661.46	0.45	0.00
c107_75_75_0	2	994.82	0.09	0.00	994.82	0.37	0.00
c107_75_75_1	1	891.64	0.13	0.00	891.64	0.42	0.00
c107_75_75_2	1	831.63	0.17	0.00	831.63	0.44	0.00
c204_25_25_0	1	856.01	0.17	0.00	856.01	0.36	0.00
c204_25_25_1	1	738.82	129.41	0.00	738.82	0.61	0.00
c204_25_25_2	1	682.12	3091.02	0.00	682.12	0.88	0.00
c204_25_50_0	1	883.80	0.13	0.00	883.80	0.61	0.00
c204_25_50_1	1	746.62	109.45	0.00	746.62	1.06	0.00
c204_25_50_2	1	714.51	†	12.55	714.51	1.00	0.00
c204_25_75_0	1	999.95	1.33	0.00	999.95	0.59	0.00
c204_25_75_1	1	814.05	1.22	0.00	814.05	0.55	0.00
c204_25_75_2	1	795.20	19.09	0.00	795.20	1.36	0.00
c204_50_25_0	1	1016.70	0.09	0.00	1016.70	0.34	0.00
c204_50_25_1	1	1014.47	0.33	0.00	1014.47	0.55	0.00
c204_50_25_2	1	982.30	2.70	0.00	982.30	0.44	0.00
c204_50_50_0	1	886.50	0.14	0.00	886.50	0.34	0.00
c204_50_50_1	1	710.02	0.61	0.00	710.02	0.50	0.00
c204_50_50_2	1	648.42	8.08	0.00	648.42	0.89	0.00
c204_50_75_0	1	893.21	0.22	0.00	893.21	0.39	0.00
c204_50_75_1	1	724.24	9.59	0.00	724.24	0.88	0.00
c204_50_75_2	1	686.19	107.57	0.00	686.19	0.69	0.00
c204_75_25_0	1	991.22	0.09	0.00	991.22	0.36	0.00
c204_75_25_1	1	877.94	0.14	0.00	877.94	0.47	0.00
c204_75_25_2	1	853.63	0.58	0.00	853.63	0.55	0.00
c204_75_50_0	1	1026.18	0.11	0.00	1026.18	0.42	0.00
c204_75_50_1	1	876.06	0.16	0.00	876.06	0.38	0.00
c204_75_50_2	1	814.15	0.19	0.00	814.15	0.55	0.00
c204_75_75_0	2	1083.95	0.11	0.00	1083.95	0.41	0.00
c204_75_75_1	1	922.15	0.13	0.00	922.15	0.36	0.00
c204_75_75_2	1	906.29	0.28	0.00	906.29	0.45	0.00
r101_25_25_0	1	989.49	0.13	0.00	989.49	0.41	0.00
r101_25_25_1	1	855.85	2.86	0.00	855.85	0.72	0.00
r101_25_25_2	1	809.65	24.48	0.00	809.65	0.84	0.00
r101_25_50_0	1	942.06	0.16	0.00	942.06	0.39	0.00
r101_25_50_1	1	747.08	5.28	0.00	747.08	0.69	0.00
r101_25_50_2	1	696.45	98.79	0.00	696.45	1.61	0.00
r101_25_75_0	1	1019.41	0.13	0.00	1019.41	0.38	0.00
r101_25_75_1	1	813.72	4.55	0.00	813.72	0.72	0.00
r101_25_75_2	1	766.84	85.82	0.00	766.84	0.94	0.00
r101_50_25_0	1	981.95	0.11	0.00	981.95	0.39	0.00
r101_50_25_1	1	800.29	0.69	0.00	800.29	0.89	0.00
r101_50_25_2	1	789.48	10.81	0.00	789.48	0.84	0.00
r101_50_50_0	1	1038.76	0.13	0.00	1038.76	0.36	0.00
r101_50_50_1	1	864.63	0.34	0.00	864.63	0.55	0.00
r101_50_50_2	1	806.89	1.30	0.00	806.89	0.78	0.00
r101_50_75_0	1	1069.61	0.13	0.00	1069.61	0.37	0.00
r101_50_75_1	1	939.93	0.30	0.00	939.93	0.50	0.00
r101_50_75_2	1	907.59	1.61	0.00	907.59	1.22	0.00
r101_75_25_0	1	1119.24	0.08	0.00	1119.24	0.39	0.00
r101_75_25_1	1	1070.84	0.13	0.00	1070.84	0.64	0.00
r101_75_25_2	1	1034.65	0.30	0.00	1034.65	0.70	0.00
r101_75_50_0	1	1093.76	0.09	0.00	1093.76	0.34	0.00
r101_75_50_1	1	969.64	0.17	0.00	969.64	0.39	0.00
r101_75_50_2	1	909.40	0.30	0.00	909.40	0.70	0.00
r101_75_75_0	2	1064.31	0.17	0.00	1064.31	0.41	0.00
r101_75_75_1	1	951.95	0.19	0.00	951.95	0.48	0.00
r101_75_75_2	1	940.24	0.53	0.00	940.24	0.50	0.00
r202_25_25_0	1	1040.78	0.11	0.00	1040.78	0.39	0.00

Table B.23: Results on instances with 25 customers: Part II

		A-ILP			T-ILP		
Instance	$\|K\|$	TD	CPU	GAP(%)	TD	CPU	GAP(%)
*r*202_25_25_1	1	873.25	5.23	0.00	873.25	0.52	0.00
*r*202_25_25_2	1	838.37	51.59	0.00	838.37	0.77	0.00
*r*202_25_50_0	1	988.74	0.14	0.00	988.74	0.39	0.00
*r*202_25_50_1	1	774.23	1.16	0.00	774.23	0.75	0.00
*r*202_25_50_2	1	710.15	10.91	0.00	710.15	1.30	0.00
*r*202_25_75_0	1	1081.92	0.13	0.00	1081.92	0.36	0.00
*r*202_25_75_1	1	957.27	0.56	0.00	957.27	0.44	0.00
*r*202_25_75_2	1	929.94	6.42	0.00	929.94	0.84	0.00
*r*202_50_25_0	1	1119.77	0.11	0.00	1119.77	0.36	0.00
*r*202_50_25_1	1	978.45	0.17	0.00	978.45	0.52	0.00
*r*202_50_25_2	1	931.40	0.41	0.00	931.40	0.48	0.00
*r*202_50_50_0	1	1043.97	0.11	0.00	1043.97	0.41	0.00
*r*202_50_50_1	1	908.72	0.24	0.00	908.72	0.47	0.00
*r*202_50_50_2	1	898.57	2.45	0.00	898.57	0.88	0.00
*r*202_50_75_0	1	1080.52	0.11	0.00	1080.52	0.36	0.00
*r*202_50_75_1	1	905.68	0.20	0.00	905.68	0.47	0.00
*r*202_50_75_2	1	882.81	0.47	0.00	882.81	0.83	0.00
*r*202_75_25_0	2	1124.48	0.09	0.00	1124.48	0.44	0.00
*r*202_75_25_1	1	959.51	0.13	0.00	959.51	0.39	0.00
*r*202_75_25_2	1	959.51	0.17	0.00	959.51	0.53	0.00
*r*202_75_50_0	2	1077.29	0.11	0.00	1077.29	0.41	0.00
*r*202_75_50_1	1	863.14	0.09	0.00	863.14	0.41	0.00
*r*202_75_50_2	1	840.97	0.16	0.00	840.97	0.56	0.00
*r*202_75_75_0	2	1161.85	0.14	0.00	1161.85	0.41	0.00
*r*202_75_75_1	1	979.63	0.14	0.00	979.63	0.44	0.00
*r*202_75_75_2	1	960.76	0.17	0.00	960.76	0.47	0.00
*rc*108_25_25_0	1	1364.73	0.16	0.00	1364.73	0.34	0.00
*rc*108_25_25_1	1	1104.76	2.22	0.00	1104.76	0.53	0.00
*rc*108_25_25_2	1	982.27	40.69	0.00	982.27	0.78	0.00
*rc*108_25_50_0	1	1306.69	0.14	0.00	1306.69	0.38	0.00
*rc*108_25_50_1	1	1041.13	3.30	0.00	1041.13	0.69	0.00
*rc*108_25_50_2	1	871.09	16.75	0.00	871.09	0.84	0.00
*rc*108_25_75_0	1	1334.70	0.17	0.00	1334.70	0.36	0.00
*rc*108_25_75_1	1	1205.54	4.47	0.00	1205.54	0.81	0.00
*rc*108_25_75_2	1	1149.82	48.26	0.00	1149.82	0.88	0.00
*rc*108_50_25_0	2	1493.89	0.13	0.00	1493.89	0.42	0.00
*rc*108_50_25_1	2	1316.91	3.22	0.00	1316.91	0.56	0.00
*rc*108_50_25_2	2	1290.27	11.08	0.00	1290.27	0.69	0.00
*rc*108_50_50_0	2	1401.34	0.23	0.00	1401.34	0.42	0.00
*rc*108_50_50_1	1	1030.03	0.36	0.00	1030.03	0.45	0.00
*rc*108_50_50_2	1	994.76	6.06	0.00	994.76	1.38	0.00
*rc*108_50_75_0	2	1563.45	0.11	0.00	1563.45	0.45	0.00
*rc*108_50_75_1	2	1246.31	3.63	0.00	1246.31	0.52	0.00
*rc*108_50_75_2	1	1146.98	4.23	0.00	1146.98	0.78	0.00
*rc*108_75_25_0	2	1741.50	0.11	0.00	1741.50	0.39	0.00
*rc*108_75_25_1	2	1586.04	0.49	0.00	1586.04	0.49	0.00
*rc*108_75_25_2	2	1536.06	6.31	0.00	1536.06	0.55	0.00
*rc*108_75_50_0	2	1516.28	0.13	0.00	1516.28	0.47	0.00
*rc*108_75_50_1	2	1343.63	1.05	0.00	1343.63	0.49	0.00
*rc*108_75_50_2	2	1240.96	0.73	0.00	1240.96	0.61	0.00
*rc*108_75_75_0	2	1658.87	0.13	0.00	1658.87	0.41	0.00
*rc*108_75_75_1	2	1383.92	0.59	0.00	1383.92	0.48	0.00
*rc*108_75_75_2	2	1379.93	3.25	0.00	1379.93	0.75	0.00
*rc*205_25_25_0	1	1274.65	0.23	0.00	1274.65	0.42	0.00
*rc*205_25_25_1	1	885.27	182.46	0.00	885.27	0.72	0.00
*rc*205_25_25_2	1	781.96	†	10.79	781.96	1.19	0.00
*rc*205_25_50_0	1	1334.79	2.94	0.00	1334.79	0.38	0.00
*rc*205_25_50_1	1	902.44	23.55	0.00	902.44	0.69	0.00
*rc*205_25_50_2	1	794.91	†	1.59	794.91	1.50	0.00
*rc*205_25_75_0	1	1370.37	0.14	0.00	1370.37	0.36	0.00
*rc*205_25_75_1	1	1072.39	4.45	0.00	1072.39	0.89	0.00
*rc*205_25_75_2	1	970.22	14.45	0.00	970.22	0.75	0.00
*rc*205_50_25_0	2	1679.02	0.14	0.00	1679.02	0.48	0.00
*rc*205_50_25_1	2	1405.13	3.17	0.00	1405.13	0.49	0.00
*rc*205_50_25_2	2	1359.23	70.79	0.00	1359.23	0.91	0.00
*rc*205_50_50_0	2	1432.56	0.22	0.00	1432.56	0.42	0.00
*rc*205_50_50_1	1	1,012.82	1.20	0.00	1,012.82	0.52	0.00
*rc*205_50_50_2	1	977.86	22.16	0.00	977.86	0.66	0.00
*rc*205_50_75_0	2	1463.48	0.19	0.00	1463.48	0.44	0.00
*rc*205_50_75_1	2	1,280.18	1.66	0.00	1,280.18	0.84	0.00
*rc*205_50_75_2	1	1183.02	0.77	0.00	1183.02	0.83	0.00
*rc*205_75_25_0	2	1610.07	0.13	0.00	1610.07	0.41	0.00
*rc*205_75_25_1	2	1248.10	0.44	0.00	1248.10	0.47	0.00
*rc*205_75_25_2	2	1156.81	0.59	0.00	1156.81	0.61	0.00
*rc*205_75_50_0	2	1536.81	0.17	0.00	1536.81	0.39	0.00
*rc*205_75_50_1	2	1273.82	0.41	0.00	1273.82	0.49	0.00
*rc*205_75_50_2	2	1170.12	2.52	0.00	1170.12	0.86	0.00
*rc*205_75_75_0	2	1574.34	0.09	0.00	1574.34	0.41	0.00
*rc*205_75_75_1	2	1258.22	0.22	0.00	1258.22	0.53	0.00
*rc*205_75_75_2	2	1212.17	0.44	0.00	1212.17	0.69	0.00
AVG1	**1**	**1012.66**	**128.61**	**0.21**	**1005.25**	**0.60**	**0.00**
AVG2	**1**	**1012.66**	**40.72**	**0.00**	**1012.66**	**0.59**	**0.00**

Table B.24: Results on instances with 100 customers: Part I

		A-ILP			T-ILP		CBC1	CBC2	CBC3	CBC4	CBC5	CBC6
Instance	$\|K\|$	TD	CPU	GAP(%)	TD	CPU	CPU	CPU	CPU	CPU	CPU	CPU
c106_25_25_0	4	3643.51	†	18.81	3640.19	4.06	3.47	3.16	3.03	5.38	3.28	3.27
c106_25_25_1	3	NFS	†	-	3127.14	21.41	5.23	5.11	7.75	6.84	5.22	5.17
c106_25_25_2	3	NFS	†	-	3018.09	89.03	78.96	78.43	128.52	814.40	29.51	12.38
c106_25_50_0	3	NFS	†	-	3258.41	4.25	5.06	5.33	3.34	3.13	3.59	3.14
c106_25_50_1	3	2846.02	†	31.13	2656.92	31.01	5.77	5.55	14.56	6.83	5.58	7.08
c106_25_50_2	3	NFS	†	-	2521.93	151.91	†	†	†	†	†	82.06
c106_25_75_0	3	NFS	†	-	3462.39	4.61	3.27	3.45	3.66	5.08	3.41	4.50
c106_25_75_1	3	3059.36	†	26.27	2908.49	36.78	6.84	6.48	13.66	7.56	6.52	7.13
c106_25_75_2	3	NFS	†	-	2791.04	201.28	49.98	50.62	280.51	442.00	37.61	25.56
c106_50_25_0	4	3946.48	†	7.93	3946.48	3.47	3.56	3.61	3.02	2.92	3.30	3.53
c106_50_25_1	4	3563.41	†	23.34	3541.00	6.27	3.70	4.20	3.67	4.03	3.59	3.17
c106_50_25_2	4	3452.99	†	28.03	3437.90	22.78	5.63	5.50	6.94	5.36	5.53	5.16
c106_50_50_0	4	3528.52	†	4.90	3520.58	5.52	5.09	5.13	3.11	5.08	4.98	3.02
c106_50_50_1	3	NFS	†	-	2879.79	47.45	4.58	4.78	6.03	5.80	4.37	3.53
c106_50_50_2	3	NFS	†	-	2626.06	94.40	1012.09	1005.53	1019.07	†	244.17	367.08
c106_50_75_0	4	3949.40	†	4.80	3949.40	1.50	3.11	3.16	3.20	4.88	3.02	4.06
c106_50_75_1	4	3459.67	†	10.99	3424.65	7.53	2.92	2.84	4.77	2.81	2.59	2.72
c106_50_75_2	4	3427.29	†	15.65	3375.71	30.62	6.33	6.39	15.30	8.44	6.23	6.94
c106_75_25_0	5	4528.44	90.50	0.00	4528.44	0.73	0.55	0.56	0.61	0.61	0.52	0.53
c106_75_25_1	4	4045.87	†	2.87	4045.87	2.24	3.69	2.91	4.31	3.25	3.92	4.72
c106_75_25_2	4	3977.54	†	8.06	3971.76	4.28	1.73	1.44	1.67	2.89	2.66	2.64
c106_75_50_0	4	4011.42	43.95	0.00	4011.42	0.81	0.59	0.56	0.66	0.59	0.53	0.58
c106_75_50_1	4	3493.91	†	2.39	3493.91	3.00	0.89	2.16	1.37	2.34	3.52	0.95
c106_75_50_2	4	3298.12	†	5.51	3294.26	3.61	1.44	1.41	1.61	3.11	2.50	2.36
c106_75_75_0	5	4554.96	15.906	0.00	4554.96	0.69	0.56	0.63	0.61	0.66	0.56	0.53
c106_75_75_1	4	4135.05	†	1.20	4135.05	2.92	2.89	3.08	1.20	2.38	1.05	1.99
c106_75_75_2	4	3977.10	†	3.06	3977.10	3.89	1.33	1.20	1.48	2.77	1.09	3.25
c202_25_25_0	4	3854.04	†	21.49	3835.40	3.24	3.16	3.13	3.27	5.42	3.16	3.17
c202_25_25_1	3	NFS	†	-	3392.28	22.67	4.20	4.56	5.69	4.66	4.80	4.47
c202_25_25_2	3	NFS	†	-	3249.86	72.39	10.87	11.00	35.47	12.91	11.03	13.28
c202_25_50_0	3	NFS	†	-	3291.61	8.97	3.39	5.41	3.23	3.25	5.38	4.70
c202_25_50_1	3	2992.99	†	33.51	2707.84	37.45	5.80	5.61	12.20	6.58	5.75	5.34
c202_25_50_2	3	NFS	†	-	2509.13	119.93	25.28	25.33	127.66	200.78	26.20	15.64
c202_25_75_0	3	NFS	†	-	3421.68	4.16	3.34	3.58	3.55	3.45	3.36	3.25
c202_25_75_1	3	3147.40	†	29.06	2961.19	43.00	6.83	6.92	17.11	7.92	6.83	8.17
c202_25_75_2	3	NFS	†	-	2842.55	183.50	24.70	24.70	196.35	28.95	24.92	27.62
c202_50_25_0	4	4214.37	†	7.88	4214.37	2.64	1.44	2.97	3.81	3.23	3.05	4.83
c202_50_25_1	4	3843.97	†	22.71	3804.34	5.23	3.44	3.49	3.56	3.22	3.61	3.45
c202_50_25_2	4	3769.37	†	27.68	3689.85	21.59	5.31	5.12	9.47	6.28	5.14	4.91
c202_50_50_0	4	3795.12	†	1.36	3795.12	2.41	0.86	0.75	0.95	0.86	0.77	0.86
c202_50_50_1	4	3142.52	†	8.31	3123.99	5.66	2.36	3.75	3.50	3.72	3.45	3.37
c202_50_50_2	4	3108.96	†	14.92	2976.90	16.50	4.34	4.41	7.88	4.41	4.45	3.75
c202_50_75_0	4	4008.06	†	0.82	4008.06	1.30	3.44	3.33	3.17	3.16	3.09	3.05
c202_50_75_1	4	3597.99	†	8.54	3595.81	4.45	3.41	2.36	3.17	3.61	3.58	3.20
c202_50_75_2	4	3642.34	†	18.75	3533.13	18.23	4.36	5.11	9.45	5.88	4.36	4.86
c202_75_25_0	5	4736.41	253.58	0.00	4736.41	0.63	0.59	0.55	0.59	0.66	0.56	0.56
c202_75_25_1	5	4496.52	†	6.65	4496.52	3.20	3.80	3.86	1.14	2.42	1.78	3.63
c202_75_25_2	4	NFS	†	-	4376.46	5.33	2.67	1.63	1.83	2.67	2.56	2.56
c202_75_50_0	4	4121.54	47.33	0.00	4121.54	0.77	0.53	0.55	0.59	0.56	0.55	0.58
c202_75_50_1	4	3521.33	†	0.95	3521.33	5.14	3.45	3.30	3.53	3.48	4.66	3.03
c202_75_50_2	4	3378.33	†	2.73	3378.33	3.28	1.50	1.59	1.59	3.19	1.78	3.28
c202_75_75_0	5	4669.75	10.828	0.00	4669.75	0.64	0.59	0.61	0.66	0.63	0.63	0.55
c202_75_75_1	4	4256.14	†	1.10	4256.14	3.58	1.08	1.13	5.03	5.19	1.05	1.74
c202_75_75_2	4	4157.07	†	3.59	4157.07	5.06	2.56	3.34	2.77	3.44	2.78	1.61
r109_25_25_0	3	3061.91	†	16.83	3055.01	3.31	5.05	3.30	3.30	3.00	3.27	3.17
r109_25_25_1	3	2854.21	†	35.18	2707.94	15.52	4.06	3.91	5.44	4.22	4.33	4.00
r109_25_25_2	3	2862.56	†	42.25	2620.80	72.28	10.64	10.70	49.47	13.23	10.86	11.22
r109_25_50_0	3	2789.39	†	18.57	2785.85	3.34	3.25	3.36	5.16	4.86	3.25	3.08
r109_25_50_1	3	2582.48	†	33.25	2425.50	40.75	8.59	8.72	17.72	9.81	9.13	8.94
r109_25_50_2	3	NFS	†	-	2305.31	196.32	23.06	23.05	195.17	27.53	22.91	25.70
r109_25_75_0	3	3058.76	†	6.86	3058.76	5.95	2.47	5.33	5.58	5.73	5.67	5.20
r109_25_75_1	3	2882.79	†	24.31	2791.33	16.97	3.56	3.84	6.14	5.75	3.61	3.99
r109_25_75_2	3	3083.84	†	37.61	2714.56	83.95	11.84	12.13	53.09	14.45	12.05	13.73
r109_50_25_0	4	3477.66	†	2.93	3477.66	2.63	4.59	3.09	2.92	3.08	5.19	2.97
r109_50_25_1	4	3196.71	†	16.21	3191.45	5.41	3.75	2.48	3.53	3.56	3.55	2.78
r109_50_25_2	4	3126.26	†	20.06	3100.93	18.62	4.17	4.28	6.16	5.56	4.03	4.31
r109_50_50_0	4	3053.22	†	3.84	3053.22	2.08	2.89	3.23	3.02	2.95	3.08	2.97
r109_50_50_1	4	2653.79	†	9.77	2646.40	5.14	3.56	3.69	3.84	3.70	2.25	2.36
r109_50_50_2	3	NFS	†	-	2492.81	37.06	518.62	517.51	522.08	†	65.17	4.27
r109_50_75_0	4	3261.69	†	5.99	3244.50	3.17	3.48	3.06	3.28	3.27	3.08	2.95
r109_50_75_1	3	NFS	†	-	2897.54	11.19	2.98	3.33	4.30	5.03	3.64	3.39
r109_50_75_2	3	NFS	†	-	2833.67	41.26	6.55	6.56	19.39	8.14	6.41	6.89
r109_75_25_0	4	3876.00	46.83	0.00	3876.00	0.69	0.53	0.52	0.56	0.55	0.53	0.55
r109_75_25_1	4	3694.25	†	0.35	3694.25	2.34	2.03	1.05	1.16	2.36	2.02	2.50
r109_75_25_2	4	3648.58	†	8.03	3648.58	2.84	2.27	1.39	2.05	2.72	1.27	2.39
r109_75_50_0	4	3466.09	4.92	0.00	3466.09	0.69	0.63	0.59	0.58	0.63	0.58	0.56
r109_75_50_1	4	3023.52	†	1.44	3023.52	3.61	2.17	1.31	1.50	6.48	2.31	3.70
r109_75_50_2	4	2922.18	†	3.44	2922.18	4.48	2.53	2.72	2.80	2.69	1.62	3.47
r109_75_75_0	4	3863.58	3.38	0.00	3863.58	0.63	0.47	0.52	0.53	0.58	0.52	0.50
r109_75_75_1	4	3670.58	145.25	0.00	3670.58	1.41	0.75	0.73	0.86	0.80	0.75	0.70
r109_75_75_2	4	3624.69	†	1.02	3624.69	3.11	1.59	2.63	2.73	2.66	2.67	1.24
r207_25_25_0	3	NFS	†	-	3270.26	3.75	3.61	3.13	3.23	3.56	3.09	3.05

Table B.25: Results on instances with 100 customers: Part II

		A-ILP			T-ILP		CBC1	CBC2	CBC3	CBC4	CBC5	CBC6
Instance	$\|K\|$	TD	CPU	GAP(%)	TD	CPU	CPU	CPU	CPU	CPU	CPU	CPU
r207_25_25_1	3	3059.99	†	41.02	2988.41	11.50	4.08	3.66	4.23	6.38	4.30	4.36
r207_25_25_2	3	3053.63	†	47.16	2889.33	30.09	5.59	5.48	10.63	6.37	5.44	5.55
r207_25_50_0	3	2734.67	†	13.79	2728.26	4.05	3.31	3.34	5.30	3.33	3.31	3.22
r207_25_50_1	3	2492.90	†	29.00	2344.16	37.83	7.53	7.28	16.16	8.89	7.53	7.64
r207_25_50_2	3	2608.60	†	40.24	2199.11	254.08	†	†	†	†	†	114.32
r207_25_75_0	3	3047.58	†	9.46	3022.39	3.67	2.94	3.33	3.28	2.95	3.88	3.16
r207_25_75_1	3	2780.16	†	23.42	2726.13	30.90	5.66	5.84	18.33	7.81	5.72	5.88
r207_25_75_2	3	NFS	†	-	2640.11	128.76	16.48	17.01	99.56	20.25	16.44	19.36
r207_50_25_0	4	3527.80	†	9.82	3527.80	2.77	3.03	3.31	4.92	3.05	3.09	2.95
r207_50_25_1	4	3190.98	†	25.71	3179.64	5.58	2.77	3.50	3.59	4.16	2.38	3.67
r207_50_25_2	4	3154.99	†	28.32	3131.35	23.01	5.27	5.25	9.03	7.19	5.75	5.41
r207_50_50_0	4	3079.74	†	4.07	3079.74	2.91	3.19	3.41	3.28	3.25	3.17	3.06
r207_50_50_1	3	NFS	†	-	2642.37	10.03	2.78	3.09	3.81	3.36	2.98	4.42
r207_50_50_2	3	NFS	†	-	2455.68	58.43	1936.14	1930.89	1939.28	†	251.72	8.00
r207_50_75_0	4	3481.45	†	2.78	3481.45	5.11	4.33	4.83	3.17	2.97	2.97	3.00
r207_50_75_1	4	3265.55	†	9.66	3265.55	4.50	2.83	2.81	2.62	3.03	3.08	2.86
r207_50_75_2	4	3181.16	†	11.75	3178.10	24.19	4.33	4.12	6.00	6.22	4.16	4.05
r207_75_25_0	4	3926.16	748.126	0.00	3926.16	0.64	0.53	0.55	0.58	0.59	0.56	0.56
r207_75_25_1	4	3678.48	†	1.97	3678.48	3.59	0.78	0.89	1.00	1.02	0.84	0.84
r207_75_25_2	4	3615.82	†	7.83	3615.82	4.89	2.55	2.59	2.09	3.20	2.75	2.27
r207_75_50_0	4	3521.76	123.232	0.00	3521.76	2.34	2.69	2.59	2.64	2.36	2.59	2.61
r207_75_50_1	4	2889.62	†	2.02	2889.62	2.14	1.08	1.11	1.08	1.44	1.03	2.33
r207_75_50_2	4	2815.52	†	4.11	2814.72	4.67	3.39	2.45	3.66	3.44	3.58	3.14
r207_75_75_0	4	3691.74	16.695	0.00	3691.74	0.73	0.56	0.59	0.66	0.58	0.58	0.56
r207_75_75_1	4	3399.89	†	1.44	3399.89	3.34	1.73	4.55	4.83	4.78	5.16	1.09
r207_75_75_2	4	3303.45	†	3.24	3303.45	4.83	1.88	1.67	4.34	5.36	3.14	3.88
rc108_25_25_0	4	4006.06	†	19.20	4005.57	3.19	3.23	4.92	5.22	3.09	3.19	3.05
rc108_25_25_1	4	NFS	†	-	3484.63	11.66	3.83	4.08	6.06	4.23	3.92	4.27
rc108_25_25_2	4	3766.30	†	44.59	3362.70	55.79	9.20	9.16	21.22	11.03	9.20	10.06
rc108_25_50_0	4	3665.40	†	12.69	3660.40	4.70	5.41	5.50	5.23	3.37	5.53	5.20
rc108_25_50_1	3	NFS	†	-	2947.32	32.83	5.25	5.36	20.67	6.72	5.48	5.94
rc108_25_50_2	3	NFS	†	-	2812.14	148.29	15.36	14.91	136.12	19.09	15.47	17.20
rc108_25_75_0	4	3782.00	†	11.49	3776.60	3.80	3.30	3.38	5.25	3.44	3.38	3.14
rc108_25_75_1	3	NFS	†	-	3310.69	49.12	5.77	5.73	12.27	6.95	5.61	6.94
rc108_25_75_2	3	NFS	†	-	3156.96	160.88	324.23	328.51	552.28	†	75.26	22.41
rc108_50_25_0	4	4334.44	†	6.11	4334.44	3.03	0.94	0.92	1.13	1.17	0.88	0.84
rc108_50_25_1	4	4042.23	†	21.12	4011.33	12.64	3.64	3.95	4.13	4.80	3.88	3.73
rc108_50_25_2	4	3933.99	†	24.85	3885.29	57.36	8.77	8.89	23.73	10.37	8.45	9.30
rc108_50_50_0	4	4165.72	†	5.14	4165.72	2.98	0.75	0.88	1.02	0.92	0.92	0.81
rc108_50_50_1	4	3495.09	†	17.76	3465.60	12.00	4.25	4.03	4.56	4.30	4.02	3.94
rc108_50_50_2	4	3442.89	†	24.29	3292.93	50.29	7.06	7.17	21.92	9.33	7.39	7.73
rc108_50_75_0	4	4770.45	†	0.60	4770.45	1.17	5.42	4.74	4.33	4.48	4.12	3.97
rc108_50_75_1	4	4338.01	†	7.85	4338.01	4.59	3.36	3.09	3.37	3.70	3.48	2.92
rc108_50_75_2	4	4292.27	†	11.97	4292.27	12.81	2.97	3.14	3.80	4.53	2.81	5.72
rc108_75_25_0	5	5336.06	7.819	0.00	5336.06	0.56	0.52	0.52	0.61	0.66	0.53	0.49
rc108_75_25_1	5	5049.05	†	1.93	5049.05	1.34	0.83	0.86	0.97	0.88	0.80	0.81
rc108_75_25_2	5	4967.61	†	5.22	4967.61	4.42	1.36	1.30	1.47	2.77	1.25	2.78
rc108_75_50_0	5	4709.75	10.53	0.00	4709.75	0.84	1.64	0.61	0.64	0.70	0.66	0.58
rc108_75_50_1	4	4179.12	780.01	0.00	4179.12	2.84	2.56	2.34	1.59	2.56	2.64	2.08
rc108_75_50_2	4	3994.11	†	2.04	3994.11	4.95	1.58	1.42	2.03	3.81	1.39	2.38
rc108_75_75_0	5	5134.52	6.946	0.00	5134.52	0.67	0.61	0.56	0.66	0.58	0.52	0.52
rc108_75_75_1	5	4785.87	3593.941	0.00	4785.87	3.34	3.58	1.03	1.30	3.67	1.02	4.59
rc108_75_75_2	5	4777.81	†	3.77	4777.81	3.34	2.78	2.48	2.14	2.58	2.56	2.66
rc205_25_25_0	4	3941.90	†	31.09	3923.66	4.02	3.48	3.13	3.42	5.23	3.50	3.19
rc205_25_25_1	3	NFS	†	-	3435.78	25.70	6.70	5.16	14.95	6.78	5.05	6.36
rc205_25_25_2	3	NFS	†	-	3339.83	128.16	17.69	15.19	74.73	18.42	16.39	16.80
rc205_25_50_0	3	NFS	†	-	3512.72	4.45	3.48	3.39	3.20	3.33	3.25	3.64
rc205_25_50_1	3	3132.04	†	36.75	2853.17	50.22	14.78	14.95	50.06	65.56	15.56	11.14
rc205_25_50_2	3	NFS	†	-	2622.99	266.98	32.36	33.00	598.12	42.48	33.56	36.25
rc205_25_75_0	4	4329.62	†	6.43	4328.99	4.95	3.06	3.13	3.14	3.17	2.91	2.78
rc205_25_75_1	4	3899.40	†	19.69	3894.89	8.77	3.28	3.19	3.78	4.36	4.55	3.89
rc205_25_75_2	4	3851.37	†	28.15	3787.87	40.69	7.05	7.06	17.31	8.61	6.66	7.56
rc205_50_25_0	4	4554.98	†	5.39	4554.98	2.86	3.16	3.27	4.98	3.20	3.19	2.94
rc205_50_25_1	4	4202.91	†	23.28	4198.08	4.70	3.56	3.44	3.58	3.63	5.52	2.89
rc205_50_25_2	4	4184.31	†	28.47	4139.36	15.86	4.36	4.05	5.78	5.77	4.83	4.38
rc205_50_50_0	4	4011.97	†	2.63	4011.97	2.91	0.89	1.84	2.41	1.95	0.83	2.75
rc205_50_50_1	4	3406.10	†	17.74	3399.79	9.62	3.23	3.03	5.61	3.80	3.37	3.27
rc205_50_50_2	4	3385.27	†	26.45	3202.24	48.25	6.75	6.59	25.59	8.30	6.61	7.19
rc205_50_75_0	4	4601.11	†	2.35	4601.11	2.84	5.02	3.22	5.11	4.84	4.95	3.06
rc205_50_75_1	4	4234.35	†	10.65	4234.09	5.56	2.34	3.16	3.53	3.70	2.28	5.17
rc205_50_75_2	4	NFS	†	-	4126.74	26.44	3.73	3.98	6.20	4.48	3.97	4.17
rc205_75_25_0	5	4846.58	41.534	0.00	4846.58	0.69	0.56	0.59	0.63	0.69	0.56	0.58
rc205_75_25_1	5	4393.80	†	3.61	4393.80	3.66	4.89	5.08	3.58	3.75	3.48	0.92
rc205_75_25_2	4	NFS	†	-	4260.67	5.55	2.02	2.89	3.30	3.86	1.69	2.72
rc205_75_50_0	4	4306.66	99.869	0.00	4306.66	0.80	0.59	0.55	0.63	0.69	0.56	0.55
rc205_75_50_1	4	3795.39	†	3.68	3795.39	3.78	1.06	1.09	1.28	1.25	1.00	1.03
rc205_75_50_2	4	3707.18	†	6.59	3706.77	5.34	1.98	1.83	3.67	2.31	5.73	1.88
rc205_75_75_0	5	5019.14	13.187	0.00	5019.14	0.64	0.63	0.55	0.63	0.70	0.56	0.59
rc205_75_75_1	5	4653.22	†	1.12	4653.22	2.66	0.97	2.28	1.17	1.33	0.98	1.19
rc205_75_75_2	5	4488.86	†	3.50	4488.86	4.09	2.67	3.63	2.53	3.25	2.50	2.38
AVG1	**4**	**4331.24**	**3171.01**	**11.62**	**3562.94**	**24.50**	**72.75**	**72.72**	**86.25**	**147.37**	**52.96**	**8.18**
AVG2	**4**	**4331.24**	**290.68**	**0.00**	**4331.24**	**1.04**	**0.96**	**0.78**	**0.80**	**0.95**	**0.79**	**0.92**

Table B.26: Results on instances with 200 customers: Part I

		T-ILP			CBC6	
Instance	$\|K\|$	TD	CPU	GAP	TD	CPU
C1_2_1_25_25_0	8	11283.51	68.31	0.00	11283.51	9.53
C1_2_1_25_25_1	8	10404.67	1556.78	0.00	10404.67	159.46
C1_2_1_25_25_2	8	NFS	†	–	10200.63	646.94
C1_2_1_25_50_0	8	10734.76	129.85	0.00	10734.76	13.39
C1_2_1_25_50_1	8	9359.18	1626.06	0.00	9359.18	178.20
C1_2_1_25_50_2	8	NFS	†	–	9019.17*	†
C1_2_1_25_75_0	9	12196.92	69.56	0.00	12196.92	15.20
C1_2_1_25_75_1	8	11305.77	990.47	0.00	11305.77	115.48
C1_2_1_25_75_2	8	11251.82	†	0.02	11064.58	541.15
C1_2_1_50_25_0	11	15078.61	16.47	0.00	15078.61	3.34
C1_2_1_50_25_1	10	14404.63	138.93	0.00	14404.63	19.72
C1_2_1_50_25_2	10	14254.11	656.03	0.00	14254.11	67.76
C1_2_1_50_50_0	10	12549.10	20.02	0.00	12549.10	4.17
C1_2_1_50_50_1	9	11251.10	297.80	0.00	11251.10	21.30
C1_2_1_50_50_2	9	11015.23	2728.24	0.00	11015.23	96.67
C1_2_1_50_75_0	10	14533.41	39.33	0.00	14533.41	4.78
C1_2_1_50_75_1	10	13539.88	106.49	0.00	13539.88	13.30
C1_2_1_50_75_2	10	13422.07	570.63	0.00	13422.07	48.72
C1_2_1_75_25_0	12	16011.96	4.48	0.00	16011.96	1.83
C1_2_1_75_25_1	11	15362.15	28.78	0.00	15362.15	5.34
C1_2_1_75_25_2	11	15289.95	101.07	0.00	15289.95	9.75
C1_2_1_75_50_0	10	13187.62	9.44	0.00	13187.62	1.95
C1_2_1_75_50_1	10	11813.52	109.62	0.00	11813.52	7.30
C1_2_1_75_50_2	10	11466.82	373.78	0.00	11464.51*	†
C1_2_1_75_75_0	11	14475.00	5.41	0.00	14475.00	3.11
C1_2_1_75_75_1	11	13607.58	28.51	0.00	13607.58	7.84
C1_2_1_75_75_2	10	13288.32	1626.34	0.00	13288.32	12.30
C2_2_3_25_25_0	8	10123.42	89.54	0.00	10123.42	13.55
C2_2_3_25_25_1	7	9423.27	3199.97	0.00	9423.27	265.97
C2_2_3_25_25_2	7	NFS	†	–	9121.84	650.19
C2_2_3_25_50_0	7	9251.42	381.03	0.00	9251.42	22.97
C2_2_3_25_50_1	7	8209.24	†	0.01	8161.40	250.99
C2_2_3_25_50_2	7	NFS	†	–	7766.10*	†
C2_2_3_25_75_0	8	10450.37	88.90	0.00	10450.37	15.16
C2_2_3_25_75_1	8	9882.38	1052.79	0.00	9882.38	137.52
C2_2_3_25_75_2	8	NFS	†	–	9696.51	547.51
C2_2_3_50_25_0	10	13105.34	13.98	0.00	13105.34	4.31
C2_2_3_50_25_1	10	12296.73	156.98	0.00	12296.73	17.73
C2_2_3_50_25_2	10	12113.05	644.92	0.00	12113.05	62.81
C2_2_3_50_50_0	9	11342.06	18.20	0.00	11342.06	5.56
C2_2_3_50_50_1	8	9755.95	3343.90	0.00	9755.95	28.28
C2_2_3_50_50_2	8	NFS	†	–	9560.23*	†
C2_2_3_50_75_0	9	11844.75	19.03	0.00	11844.75	5.69
C2_2_3_50_75_1	9	10969.32	159.79	0.00	10969.32	17.87
C2_2_3_50_75_2	9	10789.53	688.15	0.00	10789.53	67.00
C2_2_3_75_25_0	11	13946.50	4.31	0.00	13946.50	1.86
C2_2_3_75_25_1	11	13374.59	18.53	0.00	13374.59	4.42
C2_2_3_75_25_2	10	13314.91	944.55	0.00	13314.91	8.94
C2_2_3_75_50_0	10	12016.59	7.19	0.00	12016.59	2.41
C2_2_3_75_50_1	9	11084.88	566.03	0.00	11084.88	7.56
C2_2_3_75_50_2	9	10909.36	1375.34	0.00	10909.36	18.78
C2_2_3_75_75_0	10	12865.15	6.80	0.00	12865.15	2.30
C2_2_3_75_75_1	10	11711.98	54.23	0.00	11711.98	8.56
C2_2_3_75_75_2	10	11587.07	338.37	0.00	11587.07	29.19
R1_2_5_25_25_0	8	11399.57	132.21	0.00	11399.57	12.14
R1_2_5_25_25_1	8	10653.12	1099.92	0.00	10653.12	134.99
R1_2_5_25_25_2	8	NFS	†	–	10418.51	419.67
R1_2_5_25_50_0	8	9865.13	97.51	0.00	9865.13	21.08
R1_2_5_25_50_1	7	8599.48	1999.45	0.00	8599.48	543.11
R1_2_5_25_50_2	7	NFS	†	–	8231.49*	†
R1_2_5_25_75_0	8	11349.44	67.11	0.00	11349.44	11.66
R1_2_5_25_75_1	8	10402.82	1670.65	0.00	10402.82	151.31
R1_2_5_25_75_2	8	NFS	†	–	10077.01	481.90
R1_2_5_50_25_0	10	13933.99	20.41	0.00	13933.99	5.23
R1_2_5_50_25_1	10	13231.03	235.38	0.00	13231.03	23.81
R1_2_5_50_25_2	10	12930.91	973.64	0.00	12930.91	81.61
R1_2_5_50_50_0	9	11209.97	38.11	0.00	11209.97	6.78
R1_2_5_50_50_1	8	9822.99	513.19	0.00	9822.99	40.50
R1_2_5_50_50_2	8	9612.11	†	0.02	9445.51*	†
R1_2_5_50_75_0	10	12997.89	29.87	0.00	12997.89	5.16
R1_2_5_50_75_1	10	12212.10	253.80	0.00	12212.10	25.64
R1_2_5_50_75_2	10	11950.30	1394.59	0.00	11950.30	107.64
R1_2_5_75_25_0	11	15720.41	4.64	0.00	15720.41	2.06
R1_2_5_75_25_1	11	15167.18	29.55	0.00	15167.18	5.13
R1_2_5_75_25_2	11	15032.09	99.67	0.00	15032.09	10.47
R1_2_5_75_50_0	10	12991.71	9.48	0.00	12991.71	1.73
R1_2_5_75_50_1	10	11672.34	37.86	0.00	11672.34	6.19
R1_2_5_75_50_2	10	11334.87	149.29	0.00	11334.87	523.43
R1_2_5_75_75_0	11	14995.54	5.36	0.00	14995.54	1.97
R1_2_5_75_75_1	11	14439.77	41.73	0.00	14439.77	6.89
R1_2_5_75_75_2	11	14216.79	94.00	0.00	14216.79	13.14
R2_2_9_25_25_0	10	11971.75	69.93	0.00	11971.75	9.31

Table B.27: Results on instances with 200 customers: Part II

		T-ILP			CBC6	
Instance	$\|K\|$	TD	CPU	GAP	TD	CPU
R2_2_9_25_25_1	8	11121.18	713.70	0.00	11121.18	81.33
R2_2_9_25_25_2	8	NFS	†	–	10895.77	335.48
R2_2_9_25_50_0	8	10199.67	97.73	0.00	10199.67	18.11
R2_2_9_25_50_1	8	NFS	†	–	8818.51	197.38
R2_2_9_25_50_2	8	NFS	†	–	8412.13*	†
R2_2_9_25_75_0	9	12104.21	73.46	0.00	12104.21	8.81
R2_2_9_25_75_1	8	11366.36	2583.36	0.00	11366.36	90.90
R2_2_9_25_75_2	9	11388.06	†	0.03	11117.62	393.06
R2_2_9_50_25_0	10	13733.07	19.51	0.00	13733.07	4.55
R2_2_9_50_25_1	10	12980.70	187.96	0.00	12980.70	21.61
R2_2_9_50_25_2	10	12694.18	786.35	0.00	12694.18	74.76
R2_2_9_50_50_0	9	11550.48	31.61	0.00	11550.48	4.69
R2_2_9_50_50_1	9	10152.71	396.81	0.00	10152.71	47.61
R2_2_9_50_50_2	9	9951.32	†	0.01	9813.56*	†
R2_2_9_50_75_0	10	13670.67	20.55	0.00	13670.67	5.22
R2_2_9_50_75_1	10	12818.61	258.38	0.00	12818.61	26.69
R2_2_9_50_75_2	10	12563.97	991.31	0.00	12563.97	100.64
R2_2_9_75_25_0	11	15528.91	23.56	0.00	15528.91	2.45
R2_2_9_75_25_1	11	14954.69	21.87	0.00	14954.69	3.92
R2_2_9_75_25_2	11	14773.89	84.78	0.00	14773.89	9.55
R2_2_9_75_50_0	10	13056.48	7.20	0.00	13056.48	2.17
R2_2_9_75_50_1	10	11572.47	51.67	0.00	11572.47	7.61
R2_2_9_75_50_2	10	11228.07	278.24	0.00	11228.07	856.79
R2_2_9_75_75_0	11	15776.70	82.12	0.00	15776.70	2.53
R2_2_9_75_75_1	11	15239.62	29.06	0.00	15239.62	4.45
R2_2_9_75_75_2	11	15048.26	92.09	0.00	15048.26	9.64
RC1_2_2_25_25_0	8	11465.54	116.04	0.00	11465.54	14.50
RC1_2_2_25_25_1	8	10849.64	787.62	0.00	10849.64	97.15
RC1_2_2_25_25_2	8	NFS	†	–	10680.33	356.81
RC1_2_2_25_50_0	8	9873.04	115.90	0.00	9873.04	20.28
RC1_2_2_25_50_1	7	8485.05	1914.00	0.00	8485.05	324.25
RC1_2_2_25_50_2	7	NFS	†	–	8083.75*	†
RC1_2_2_25_75_0	8	11407.10	72.15	0.00	11407.10	12.67
RC1_2_2_25_75_1	8	10633.50	678.44	0.00	10633.50	95.42
RC1_2_2_25_75_2	8	NFS	†	–	10479.76	336.06
RC1_2_2_50_25_0	10	12685.85	21.67	0.00	12685.85	4.75
RC1_2_2_50_25_1	9	11743.68	285.58	0.00	11743.68	31.83
RC1_2_2_50_25_2	9	11538.25	1919.08	0.00	11538.25	139.03
RC1_2_2_50_50_0	9	11901.84	29.67	0.00	11901.84	7.17
RC1_2_2_50_50_1	9	10389.54	360.47	0.00	10389.54	46.42
RC1_2_2_50_50_2	9	10179.69	2638.32	0.00	10179.69	276.22
RC1_2_2_50_75_0	10	14026.09	17.22	0.00	14026.09	5.77
RC1_2_2_50_75_1	10	13322.63	111.95	0.00	13322.63	15.36
RC1_2_2_50_75_2	10	13159.21	611.49	0.00	13159.21	58.25
RC1_2_2_75_25_0	12	15790.92	4.22	0.00	15790.92	2.14
RC1_2_2_75_25_1	11	15473.16	68.83	0.00	15473.16	4.22
RC1_2_2_75_25_2	11	15315.91	81.53	0.00	15315.91	8.30
RC1_2_2_75_50_0	10	12801.00	4.84	0.00	12801.00	1.73
RC1_2_2_75_50_1	10	11439.60	27.77	0.00	11439.60	5.42
RC1_2_2_75_50_2	10	11147.26	120.09	0.00	11147.26	10.66
RC1_2_2_75_75_0	11	15241.89	4.20	0.00	15241.89	2.20
RC1_2_2_75_75_1	11	14493.83	28.61	0.00	14493.83	4.72
RC1_2_2_75_75_2	11	14408.80	134.99	0.00	14408.80	11.36
RC2_2_2_25_25_0	9	11996.43	67.72	0.00	11996.43	10.38
RC2_2_2_25_25_1	8	11112.52	567.46	0.00	11112.52	68.40
RC2_2_2_25_25_2	8	10880.95	2566.52	0.00	10880.95	257.74
RC2_2_2_25_50_0	8	10288.10	81.67	0.00	10288.10	11.61
RC2_2_2_25_50_1	8	9049.90	1290.97	0.00	9049.90	166.18
RC2_2_2_25_50_2	8	NFS	†	–	8768.07	665.42
RC2_2_2_25_75_0	9	12390.03	38.92	0.00	12390.03	8.12
RC2_2_2_25_75_1	9	11702.26	589.61	0.00	11702.26	59.51
RC2_2_2_25_75_2	9	11554.80	2161.88	0.00	11554.80	201.46
RC2_2_2_50_25_0	10	13935.33	21.01	0.00	13935.33	4.89
RC2_2_2_50_25_1	10	13157.47	203.38	0.00	13157.47	22.00
RC2_2_2_50_25_2	10	13011.45	715.73	0.00	13011.45	73.20
RC2_2_2_50_50_0	9	11753.09	30.14	0.00	11753.09	5.89
RC2_2_2_50_50_1	8	9687.19	505.96	0.00	9687.19	56.43
RC2_2_2_50_50_2	8	9382.32	†	0.01	9271.31	197.02
RC2_2_2_50_75_0	10	13083.61	23.69	0.00	13083.61	4.94
RC2_2_2_50_75_1	9	12309.00	294.35	0.00	12309.00	26.09
RC2_2_2_50_75_2	9	12129.82	1103.73	0.00	12129.82	97.15
RC2_2_2_75_25_0	11	14765.00	6.45	0.00	14765.00	1.83
RC2_2_2_75_25_1	11	13917.72	40.97	0.00	13917.72	7.78
RC2_2_2_75_25_2	11	13700.43	157.77	0.00	13700.43	17.45
RC2_2_2_75_50_0	11	13466.10	5.56	0.00	13466.10	1.81
RC2_2_2_75_50_1	10	11406.55	35.67	0.00	11406.55	6.22
RC2_2_2_75_50_2	10	10803.64	733.54	0.00	10803.36*	†
RC2_2_2_75_75_0	12	16883.64	3.34	0.00	16883.64	1.69
RC2_2_2_75_75_1	12	16476.26	18.69	0.00	16476.26	3.42
RC2_2_2_75_75_2	12	16440.09	65.61	0.00	16440.09	7.77
AVG1	**10**	**12470.75**	**870.73**	**0.00**	**12252.67**	**302.75**
AVG2	**10**	**12490.10**	**440.23**	**0.00**	**12490.10**	**50.91**