New cuts and a branch-cut-and-price model for the Multi-Vehicle Covering Tour Problem

Bruno Oliveira[1*], Artur Pessoa[2†] and Marcos Roboredo[2†]

[1]Pós-Graduação em Engenharia de Produção, Universidade Federal Fluminense, Rua Passo da Pátria, Niterói, 24210-240, Rio de Janeiro, Brazil.

[2]Departmento de Engenharia de Produção, Universidade Federal Fluminense, Rua Passo da Pátria, Niterói, 24210-240, Rio de Janeiro, Brazil.

*Corresponding author(s). E-mail(s): bruno_mattos@id.uff.br; Contributing authors: arturpessoa@id.uff.br; mcroboredo@id.uff.br;
†These authors contributed equally to this work.

Abstract

The Multi-Vehicle Covering Tour Problem ($\boldsymbol{m}$-CTP) involves a graph in which the set of vertices is partitioned into a depot and three distinct subsets representing customers, mandatory facilities, and optional facilities. Each customer is linked to a specific subset of optional facilities that define its coverage set. The goal is to determine a set of routes with minimal cost that satisfy the following constraints: each route begins and ends at the depot; every mandatory facility is visited exactly once on a single route; each route visits not more than $\boldsymbol{p}$ facilities and have a maximum cost of $\boldsymbol{q}$; for each customer, at least one optional facility from its coverage set must be visited by one of the routes. In this paper, we present the following contributions for the $\boldsymbol{m}$-CTP: an exact branch-cut-and-price algorithm; a new family of capacity-like cuts; and a new set of benchmark instances. We report several experiments that prove the effectiveness of the proposed algorithm and cuts. The results show that the proposed algorithm outperforms the best exact method from the literature and that the proposed cuts further improve its performance by one order of magnitude. The proposed algorithm and cuts allow us to effectively solve 287 out of 288 literature instances.

Keywords: Routing, Covering, Inequalities, Branch-Cut-and-Price

1 Introduction

In many variations of the Vehicle Routing Problem (VRP), customers are serviced directly by vehicles that move to their locations to meet their demands. However, in certain real-world scenarios, it may be necessary that vehicles avoid direct customer visits and, instead, service demands from nearby facilities. For instance, in the event of a disaster in a specific region, essential supplies like food, medical items, and humanitarian aid must be delivered. Such deliveries are usually made to specialized facilities like schools, hospitals, etc., located close to the affected residents. Another example is the distribution of vaccines, where they are transported to dedicated centers, hospitals, etc., in proximity to those in need of vaccinations. This context gives rise to the focal point of this research, known as the Multi-Vehicle Covering Tour Problem (m-CTP).

The m-CTP was proposed by [1] and it is formally defined as follows. Let $G = (V, E)$ be a complete undirected graph. The node set is $V = \{0\} \cup M \cup O \cup C$, where $\{0\}$ represents the depot, $M = \{1, ..., m\}$ represents the set of mandatory facilities, $O = \{m+1, ...m+o\}$ represents the set of optional facilities and $C = \{m+o+1, ..., m+o+c\}$ represents the set of customers. Each edge $e \in E$ is associated with a travel cost c_e. Each customer $j \in C$ is associated with a coverage set $\phi(j) \subseteq O$. The goal of the m-CTP is to design a set of finite routes with minimal total traveled cost satisfying the following constraints: each route must start and end at the depot; each mandatory facility $i \in M$ is visited exactly once across all the routes; for each customer $j \in C$, there is at least one optional facility $i \in \phi(j)$ that is visited by one of the routes; Each route visits at most p facilities; the maximum cost of each route is q. For the particular cases of the m-CTP with $q = +\infty$ or $p = +\infty$, we refer them to respectively m-CTP-p and m-CTP-q. Throughout this paper, we may use the term m-CTP to denote the problem instances where both types of constraints are present, which should be clear from the context.

In order to illustrate the m-CTP, we present a toy instance with $M = \{1\}$, $O = \{2, 3, 4\}$, $C = \{5, 6\}$, $p = 2$, $q = +\infty$, $\phi(5) = \{3, 4\}$ and $\phi(6) = \{2\}$. We omit the travel costs. Figure 1 shows an example of a feasible solution for the proposed instance, where the nodes marked in yellow, red, green, and blue represent respectively the depot, the mandatory facility, the optional facilities, and the customers. Around each customer, we delineate a dashed red circumference. The optional facilities within the circumference represent the customer's coverage set.

This paper presents the following contributions for the m-CTP:

- A new branch-cut-and-price (BCP) model;
- A family of capacity-like cuts that can be added to the proposed BCP algorithm on demand;
- A new set of benchmark instances with up to 393 facilities;

To demonstrate the effectiveness of our proposed algorithm and cuts, we conducted computational experiments divided into two parts. Firstly, we applied our proposed algorithm (with and without the proposed cuts) to literature benchmark instances that are generated based on the classical Traveling Salesman Problem (TSP). We achieved optimal solutions for these m-CTP instances, which, to the best of our knowledge, have not been achieved before. Subsequently, we compared our approach with the best exact

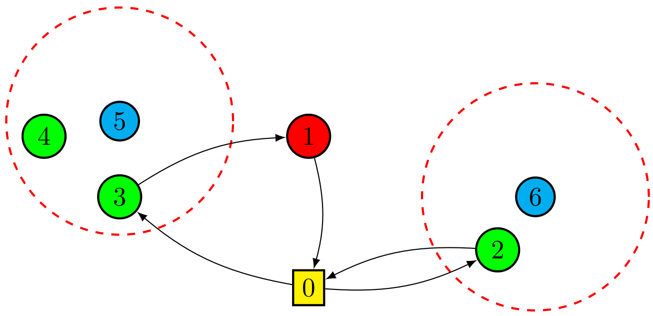

Fig. 1: Example of a m-CTP instance and a possible feasible solution.

method from the m-CTP-p literature, a branch-price algorithm, which was proposed by [2]. Even without the proposed cuts, our algorithm consistently outperformed the previous method. With the cuts, the observed performance was improved by one order of magnitude. Finally, we tested our BCP algorithm on a new set of instances with up to 393 facilities. The results for these instances showed that they are more challenging, and therefore serve well as a benchmark for future algorithms.

This paper is organized as follows: In Section 2, we present a literature review of the m-CTP. In Section 3, we present the proposed BCP algorithm for the m-CTP. In Section 4, we present a new family of capacity-like cuts for the problem, and show how to separate them. In Section 5, we present several computational experiments. Finally, In Section 6, we summarize our conclusions.

2 Literature Review

The m-CTP was proposed by [1]. The authors developed three heuristics for the problem that were tested on randomly generated and real data. [3] proposed the first exact approach for the m-CTP, that is a branch-and-price (BP) algorithm, in which the pricing subproblem reduced to the Elementary Shortest Path Problem with Resource Constraints (ESPPRC), [4], and solved by a dynamic programming algorithm. The authors also proposed a column generation heuristic based on a set partitioning formulation. [5] proposed another BP algorithm based on a set partitioning formulation, in which the subproblem encountered during the column generation is a variant of the profitable tour problem. The authors reduced this subproblem to a ring-star problem and solved it using a branch-and-cut algorithm. They also proposed a set of m-CTP-p and m-CTP-q instances based on classical TSP ones. Although the proposed algorithm allows the inclusion of both the constraints on the number of visited facilities and the route cost, no instance with both constraints were used in that paper.

Some researchers considered specifically the m-CTP-p. For this problem, [6] proposed an exact formulation, valid inequalities, and two metaheuristics. The authors solved the formulation through a branch-and-cut algorithm. The metaheuristic proposed by those authors is a two-phase hybrid algorithm, where the first phase aims to select facilities to cover all customers, which can be seen as a Set Covering Problem. In the second phase, the metaheuristic solves a Capacitated Vehicle Routing Problem

(CVRP) instance with unit demand considering only the facilities selected in the first phase. [7] proposed a Variable Neighborhood Search heuristic to solve the m-CTP-p. The heuristic proposed by these authors outperforms the one proposed by [6] in time and solution quality. Recently, [2] proposed a BP algorithm based on set partitioning formulation for the m-CTP-p, in which each pricing subproblem is formulated as an ESPPRC. As the ESPPRC can be time-consuming to solve, they also applied ng-route relaxation, stabilization techniques, and a heuristic to solve the pricing subproblem. The authors presented computational experiments that show that their exact algorithm outperforms the exact one proposed by [6].

Some researchers consider variants of the m-CTP. In this context, [8] introduced the Multi-Depot Covering Tour Vehicle Routing Problem (MDCTVRP). In this problem, a customer's demand can be fulfilled either by directly visiting the customer or by visiting a nearby facility. The authors proposed two Mixed Integer Linear Programming (MIP) formulations, and a hybrid metaheuristic approach that combines GRASP, Iterated Local Search, and Simulated Annealing algorithms. [9] introduced the Multi-Vehicle Cumulative Covering Tour Problem (m-CCTP) whose main difference from m-CTP is that in m-CCTP objective is to minimize the sum of arrival times (latency) at each visited facility. The authors proposed a MIP formulation and a GRASP algorithm for the problem. [10] introduced another variant of the m-CTP named Multi-Vehicle Multi-Covering Tour Problem, where each customer must be covered several times instead of a single one. The authors developed a branch-and-cut and a genetic algorithm for that variant. [11] also introduced a variant of the m-CTP, the Multi-Vehicle Probabilistic Problem, where each customer has a probability of being covered by a given facility. The objective function is then to maximize the expected customer demand covered. For this variant, the authors developed a branch-and-cut algorithm and a local search heuristic based on Variable Neighborhood Search. Finally, [12] presented a m-CTP variant for surveillance that involves multiple vehicles, designed to monitor targets (referred to as mandatory passive nodes) for a specified duration. These vehicles move between waypoints (referred to as optional active nodes), and they have the flexibility to adjust their speed to extend the time spent covering a particular node, ensuring it remains within the coverage range. While the primary objective is to cover all passive nodes, the optimization goal is to maximize the coverage time for nodes that hold greater importance, rather than minimizing the total tour length. To tackle this problem, the authors employ a branch-and-price algorithm that takes into account speed adjustments. Additionally, they introduce a heuristic construction strategy to address the challenge effectively.

3 Proposed BCP model

In this section, we present a new BCP model for the m-CTP. A BCP model is composed of a MIP formulation where the variables are called *original variables*, and a subproblem that defines an additional set of variables (or columns), each one associated to a subproblem solution. The original variables in the MIP formulation are then required to be a linear combination of the columns, further strengthening the linear relaxation of the MIP formulation. In the case of the m-CTP, the subproblem is a Resource Constrained Shortest Path Problem (RCSPP), and each original variable counts the number of times that an edge of G is traversed by a solution. In order to define the RCSP subproblem, we show how to translate G into a directed graph G' that is used as an input to the RCSP solver. We call it the *path generator graph*. To employ advanced techniques that improve the BCP performance, we also define *packing-sets*, which are subsets of vertices of G' that can be used at most once in a complete solution.

The remainder of this section is organized as follows. Sections 3.1 and 3.2 present respectively the path generator graph and the MIP formulation that compose the proposed BCP model. Finally, in Section 3.3, we define the *packing-sets* that are allows activating state-of-the-art BCP elements such as ng-path relaxation [13], limited memory Chvátal-Gomory Rank-1 Cuts, [14] and path enumeration, [15].

3.1 Path generator graph

Let $G' = (V', A')$ be a directed graph such that, $V' = \{v_0, v_1, \dots, v_{|M|+|O|}\}$ and $A' = \{(v_i, v_j), (v_j, v_i) | i, j \in V', i < j\}$. Let ρ be a path starting and ending at node v_0 such that this node is not visited in the middle of the path. The path ρ is associated with two non-negative numbers S^1_ρ and S^2_ρ that represent the accumulated consumption of two resources. S^1_ρ and S^2_ρ are calculated in the following way. When ρ starts, $S^1_\rho = S^2_\rho = 0$. Whenever an arc $(v_i, v_j) \in A'$ is traversed, the values of S^1_ρ and S^2_ρ are increased according to the following equations:

$$S^1_\rho = \begin{cases} S^1_\rho + 0.5 & \text{if } i = 0 \text{ or } j = 0 \\ S^1_\rho + 1.0 & \text{otherwise.} \end{cases} \qquad S^2_\rho = \begin{cases} S^2_\rho + c_{(i,j)} & \text{if } i < j \\ S^2_\rho + c_{(j,i)} & \text{otherwise.} \end{cases}.$$

We define the increase of S^1_ρ at both the incoming and outgoing arcs of v_0 as 0.5 to make G' symmetrical with respect to forward and backward traversals. This feature is explored by BCP specialized solvers to enhance their performances.

We say that the path ρ is resource-constrained if $S^1_\rho \le p$ and $S^2_\rho \le q$. Figure 1 illustrates the path generator graph for the proposed toy example where we marked in yellow, green, and red the depot, and the optional and mandatory facilities, respectively. On each arc, we denote by s_1 and s_2 the increase in the value of respectively S^1_ρ and S^2_ρ.

Since $p = 2$ and $q = +\infty$, the following paths are resource-constrained on the graph from Figure 1: $\rho_1 = (v_0, v_1, v_0)$, $\rho_2 = (v_0, v_2, v_0)$, $\rho_3 = (v_0, v_3, v_0)$, $\rho_4 = (v_0, v_4, v_0)$, $\rho_5 = (v_0, v_1, v_2, v_0)$, $\rho_6 = (v_0, v_2, v_1, v_0)$. $\rho_7 = (v_0, v_1, v_3, v_0)$, $\rho_8 = (v_0, v_3, v_1, v_0)$, $\rho_9 = (v_0, v_1, v_4, v_0)$, $\rho_{10} = (v_0, v_4, v_1, v_0)$, $\rho_{11} = (v_0, v_2, v_3, v_0)$, $\rho_{12} = (v_0, v_3, v_2, v_0)$, $\rho_{13} = (v_0, v_2, v_4, v_0)$, $\rho_{14} = (v_0, v_4, v_2, v_0)$, $\rho_{15} = (v_0, v_3, v_4, v_0)$, $\rho_{16} = (v_0, v_4, v_3, v_0)$. To facilitate the subproblem resolution, we do not require that the generated paths are elementary in our model. Although such paths are avoided when state-of-the-art elements are activated, such as ng-route, the MIP formulation is responsible for ensuring that any solution that satisfies all constraints corresponds to a valid m-CTP solution.

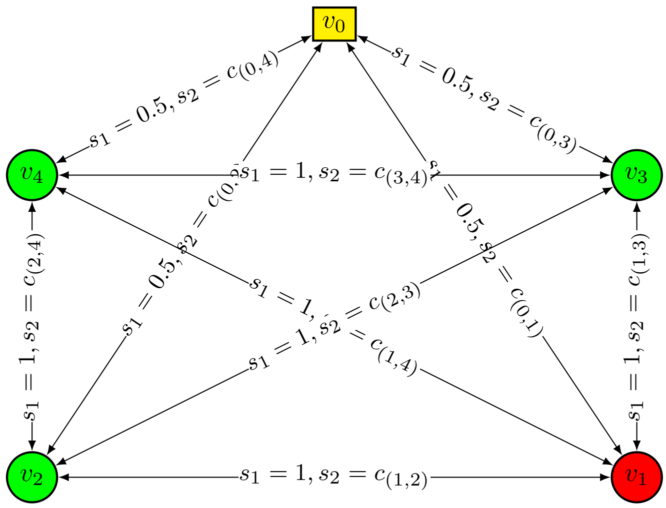

Fig. 2: Path generator graph for the proposed toy example.

3.2 MIP formulation

In this section, we introduce a MIP formulation that considers path variables over G' defined in Section 3.1. The idea behind the proposed formulation is that each route that visits at most p facilities and with a total cost less or equal to q is associated with a path variable on G'. Specifically, the formulation uses an integer variable x_e, for each $e \in E''$, where $E'' = \{e = (i,j) \in E | i,j \in \{0\} \cup M \cup O\}$. Each variable x_e indicates how many times the edge e is used in the solution. Let P be the set of resource-constrained paths over the graph G'. For each $\rho \in P$, the formulation considers an integer variable λ_ρ to indicate the number of times that the path ρ is used at the solution. Besides the variables, the formulation uses the following sets: the set $\delta(S) = \{(i,j) \in E'' | (i \in S \wedge j \notin S) \vee (i \notin S \wedge j \in S)\}$, for a given $S \subseteq \{0\} \cup M \cup O$; the constant h^a_ρ indicates the number of times that an arc $a \in A'$ is used in the path $\rho \in P$; the set $M(e) = \{(v_i, v_j), (v_j, v_i)\}$, for each $e = (i,j) \in E''$. The formulation follows.

$$\text{Min} \sum_{e \in E''} c_e x_e \tag{1a}$$

$$\text{s.t.} \sum_{e \in \delta(\{i\})} x_e = 2, \quad i \in M; \tag{1b}$$

$$\sum_{e \in \delta(\{i\})} x_e \leq 2, \quad i \in O; \tag{1c}$$

$$\sum_{e \in \delta(\phi(j))} x_e \geq 2, \quad j \in C; \tag{1d}$$

$$x_e = \sum_{\rho \in P} \sum_{a \in M(e)} h_\rho^a \lambda_\rho, \quad e \in E''; \tag{1e}$$

$$1 \leq \sum_{\rho \in P} \lambda_\rho \leq |M| + |O|; \tag{1f}$$

$$x_e \in \mathbb{Z}, \quad \forall e \in E''; \tag{1g}$$

$$\lambda_\rho \geq 0, \quad \rho \in P \tag{1h}$$

The objective function (1a) minimizes the total cost of the routes. Constraints (1b) guarantee that each mandatory node is visited exactly once. Constraints (1c) guarantee that each optional node is visited at most once. Constraints (1d) ensure that there is at least one visited node in $\phi(j)$, for each customer $j \in C$. Constraints (1e) ensure the relation between the λ and x variables. For a given $e = (i, j) \in E''$, the value of x_e is given by the number of times that the arcs in $M(e)$ are traversed considering all the paths of the optimal solution. Constraints (1f) guarantee that the solution has at least 1 path and at most $|M| + |O|$ paths. Constraints (1g) and (1h) ensure the domain of the variables.

Since we allow non-elementary paths as RCSPP solutions, it's possible to find an integer solution for the x variables, but a fractional solution for the λ variables where the degree of an optional facility $i \in O$ is equal to 1, i.e., there exists $e^* \in \delta(i)$ such that $\sum_{e \in \delta(i)} x_e = x_{e^*} = 1$. To avoid having to branch on λ variables, we insert the following constraint on demand, by inspection: $x_{e^*} \leq \sum_{e \in \delta(i) \setminus \{e^*\}} x_e$.

3.3 Packing Sets

In this section, we define *packing-sets* that can be explored by BCP solvers to activate advanced features. Let $\mathcal{S}^V \subset 2^{V' \setminus \{v_0\}}$ be a collection of mutually disjoint subsets of $V' \setminus \{v_0\}$. We say that the sets in $\mathcal{S}^V$ are *packing-sets* if there is at least one optimal solution to formulation (1), satisfying the following constraints:

$$\sum_{\rho \in P} \left(\sum_{v \in S} h_v^\rho \right) \lambda_\rho \leq 1, \quad S \in \mathcal{S}^V, \tag{2}$$

where, h_v^ρ indicates how many times the node v appears in the path ρ. In other words, for each packing set $S \in \mathcal{S}^V$, at most one node in this set can be traversed by a path and it can occur at most once. In our model, we define a different packing set $\{v_i\}$ for each $i \in M \cup O$ once each facility is visited at most once. For details about how state-of-the-art BCP elements that can be activated based on provided packing sets, we refer to [16].

4 A new family of cuts for the m-CTP

In this section, we introduce a set of valid inequalities for the m-CTP. These inequalities can be seen as a generalization of the following constraints proposed by [17] for a version of the m-CTP that considers just a single route:

$$\sum_{e\in\delta(S)} x_e \geq 2,\ \forall S \subseteq M\cup O\colon S\cap M \neq \emptyset \text{ or } \phi(j)\subseteq S,\ \text{for some } j\in C \tag{3}$$

Constraints (3) ensure that a given set $S \subseteq M \cup O$ must be visited by at least one vehicle (that enters and leaves S) when there is a mandatory facility in S, or when all facilities of the coverage set of a given customer are in S. Note that, if we have $S = \phi(j)$, $j \in C$, we get the Constraints (1d).

The proposed set of valid inequalities aims to identify cases where the number of vehicles that must visit a given subset S of facilities is greater than one. For that, we rely on the limit p on the number of facilities that can be visited by each vehicle. If one can prove that at least α facilities of S must be visited in any feasible solution, then the number of vehicles that enter (and leave) S must be at least $\lceil \alpha/p \rceil$. In the proposed inequalities, α is computed as the number of mandatory facilities in S plus the optimal objective value to a linear relaxation of a Set Covering Problem with a modified objective function that minimizes the number of optional facilities of S that are visited. This approach leads to the following inequality:

$$\sum_{e\in\delta(S)} x_e \geq 2K(S),\ \forall S \subseteq M\cup O, \tag{4}$$

where

$$K(S) = \left\lceil \left(|S\cap M| + \min_{\xi\in\mathbb{R}_+^{|O|}} \left\{ \sum_{i\in S\cap O} \xi_i \mid \sum_{i:i\in\phi(j)} \xi_i \geq 1, \forall j\in C \right\} \right) /p \right\rceil .$$

From now on, we will refer to inequality (4) as a *Covering-Capacity Cut* (*CCC*). To illustrate the inequality, consider an m-CTP instance with $p = 2$. For this instance, we take $S = \{i_1, i_2, i_3\}$, where $i_1 \in M$, $i_2, i_3 \in O$ and there are two customers $j_1, j_2 \in C$ such that $\phi(j_1) = \{i_2\}$ and $\phi(j_2) = \{i_3\}$. The Figure 3 illustrates the instance and the set S.

For the proposed example, we have $K(S) = \left\lceil \dfrac{1+2}{2} \right\rceil = 2$. In other words, for any feasible solution, at least two vehicles are necessary to visit the set S. It happens because the mandatory facility must be visited due to the problem definition and the two optional facilities i_2 and i_3 also must be visited to cover respectively customers j_1 and j_2.

The Proposition 1 proves that CCC are valid for the m-CTP.

Proposition 1. *The CCC are valid for the m-CTP.*

Proof. Let $S \subseteq M \cup O$. First, we prove that at least $|S \cap M| + z^*$ facilities in S must to be visited in any feasible m-CTP solution, where $z^* =$

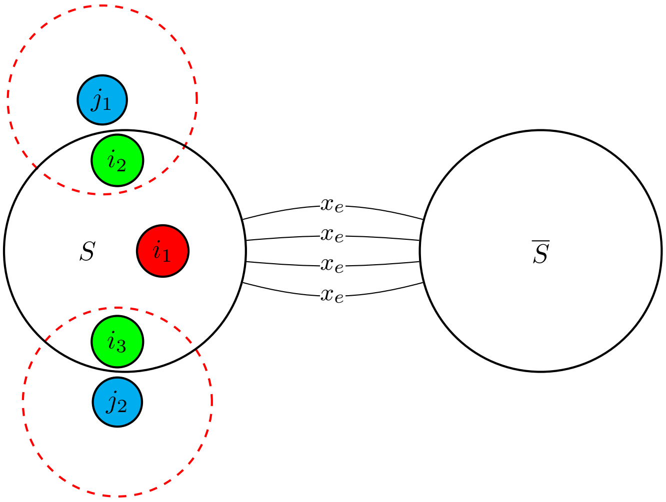

Fig. 3: An Illustration for the proposed cuts.

$\min_{\xi \in \mathbb{R}_+^{|O|}} \left\{ \sum_{i \in S \cap O} \xi_i \mid \sum_{i: i \in \phi(j)} \xi_i \geq 1, \forall j \in C \right\}$. In fact, by the m-CTP definition, all $|S \cap M|$ mandatory facilities in S must be visited. Regarding the optional facilities in S, z^* represents a lower bound for the minimum number of visited optional facilities in S in any feasible solution. Thus, no m-CTP solution can visit less than $|S \cap M| + z^*$ facilities in S. Now, let $\overline{x}$ be a feasible solution to (1). Since no route can visit more than p facilities, the total number of times that the solution traverses an edge from $\delta(S)$ must be at least $2\lceil (|S \cap M| + z^*)/p \rceil$. Hence, (4) must be satisfied by $\overline{x}$. □

4.1 The separation algorithm

The strategy adopted here to find violated CCC is to solve one separation problem for each fixed value of $K(S)$ by modeling it as a MIP. Namely, we seek the set $S \subseteq M \cup O$ that minimizes the left-hand side of (4) such that $K(S) \geq k$, for $k = 1, ..., k_{\max}$, where $k_{\max} = K(M \cup O)$. Note that $K(M \cup O)$ is the value of $K(S)$ when S contains all mandatory and optional nodes, thus representing the maximum value that it can assume. Let $\bar{x}$ be a relaxed solution for Formulation (1) and k be an integer number such that $1 \leq k \leq k_{\max}$. We start with a bi-level formulation that uses two types of first-level binary variables. For each $i \in M \cup O$, we define a binary variable z_i indicating if $i \in S$ ($z_i = 1$) or not ($z_i = 0$), and, for each $e \in E''$, we define a binary variable r_e indicating if $e \in \delta(S)$ ($r_e = 1$) or not ($r_e = 0$). The second-level variables are the continuous variables introduced in the definition of $K(S)$. The formulation follows.

$$\text{Min} \sum_{e \in E''} \bar{x}_e r_e \tag{5a}$$

$$\text{s.t. } r_e \geq z_i - z_j, \quad e = (i, j) \in E'' \tag{5b}$$

$$r_e \geq z_j - z_i, \quad e = (i,j) \in E'' \tag{5c}$$

$$z_0 = 0 \tag{5d}$$

$$\sum_{i \in M} z_i + \min_{z' \in \mathbb{R}_+^{|O|}} \left\{ \sum_{i \in O} z_i \xi_i \mid \sum_{i \in \phi(j)} \xi_i \geq 1, \forall j \in C \right\} \geq (k-1)p + \epsilon \tag{5e}$$

$$r_e \in \{0,1\}, \quad e \in E'' \tag{5f}$$

$$z_i \in \{0,1\}, \quad i \in M \cup O, \tag{5g}$$

where ϵ is a small number that is still sufficiently large to ensure that the left-hand side of (5e) is strictly greater than $(k-1)p$ despite numerical errors.

The objective function (5a) aims to minimize the left-hand side of 4. Constraints (5b) and (5c) ensures the relation between variables r and z. Constraint (5d) ensures that the depot is not in S. Constraint (5e) guarantees that $K(S) \geq k$. To see this, note that it ensures that

$$\left(\sum_{i \in M} z_i + \min_{z' \in \mathbb{R}_+^{|O|}} \left\{ \sum_{i \in O} z_i \xi_i \mid \sum_{i \in \phi(j)} \xi_i \geq 1, \forall j \in C \right\} \right) /p > (k-1). \tag{6}$$

Thus, rounding up the left-hand side of (6) results in at least k.

Since (5) cannot be optimized in its present form, we obtain a formulation equivalent to (5) by replacing the optimization problem that appears in Constraint (5e) with its dual. For that, for each $j \in C$, let y_j be the dual variable associated with the constraint $\sum_{i \in \phi(j)} z'_i \geq 1$. The formulation follows.

$$\text{Min (5a)} \tag{7a}$$

$$\text{s.t. (5b), (5c), (5d), (5f), (5g)} \tag{7b}$$

$$\sum_{i \in M} z_i + \left\{ \begin{array}{ll} \text{Max} \sum_{j \in C} y_j & \\ \text{s.t.} \sum_{j \in C | i \in \phi(j)} y_j \leq z_i, & i \in O \\ y_j \geq 0, & j \in C \end{array} \right\} \geq (k-1)p + \epsilon, \tag{7c}$$

which is equivalent to:

$$\text{Min (5a)} \tag{8a}$$

$$\text{s.t. (5b), (5c), (5d), (5f), (5g)} \tag{8b}$$

$$\sum_{i \in M} z_i + \sum_{j \in C} y_j \geq (k-1)p + \epsilon \tag{8c}$$

$$\sum_{j \in \sigma(i)} y_j \leq z_i, \quad i \in O \tag{8d}$$

$$y_j \geq 0 \quad j \in C. \tag{8e}$$

In the proposed BCP algorithm, we use the MIP formulation (8) to find violated CCC, for $k = 1, \ldots, k_{\max}$.

5 Computational Results

In this section, we present the results of our model on three sets of instances. The first and second ones are the same used by [2] and [10] for the m-CTP-p and the m-CTP, respectively. The third set of instances is proposed in this paper, based on the set of CVRP instances proposed by [18], and having up to 393 facilities. All the instances used in the reported experiments are available at https://github.com/brunomattos1/m-ctp_instances.

The model is solved using the VRPSolver framework, which was developed by [16] aiming to facilitate the implementation of efficient BCP algorithms for vehicle routing problems and other related problems. We opted for the VRPSolver framework to implement the BCP algorithm due to its state-of-the-art features, such as a bi-directional labeling algorithm [19] within the concept of buckets, proposed by [20], to solve the pricing subproblem, ng-route relaxation [13], limited memory Chvátal-Gomory Rank-1 Cuts, [14], route enumeration, [15], strong branching, [21], automatic dual stabilization technique, [22]. And also because its proven success in various VRP and Scheduling variants, as demonstrated in [23], [24], [25], [26] and [16].

All tests were performed on an Intel(R) Core(TM) i7-10700 CPU@ 2.90GHz. The operational system used was Ubuntu and the algorithm were implemented in Julia using the VRPSolver framework v0.4.1a (https://vrpsolver.math.u-bordeaux.fr/). CPLEX 12.10 was used to solve LP and MIP formulations. The tables presented in this paper contain only average statistics. Detailed data for each tested instance are provided in an online supplementary material.

5.1 Results for [2] and [10]

Table 1 presents performance indicators of our method over literature instances, and a comparison between the two versions of the proposed BCP algorithm (with and without CCC) and the best exact literature method for the m-CTP-p proposed by [2]. For m-CTP and m-CTP-p instances, our method uses as initial upper bounds, solution values found by the heuristics proposed by [10] and [7], respectively. Regarding the comparison between our algorithm and the one proposed by [2], we highlight two points. First, each reported runtime for the latter by the literature was divided by 1.5. This factor is the ratio between the scores of the processor used by us and by the literature obtained at www.cpubenchmark.net, for a single thread. The second point is that the literature considered a time limit of 7200s. For this reason, we set a time limit of 4800s (7200/1.5) for experiments over m-CTP-p instances. For the remaining instances, we set a time limit of 7200s. For each problem (m-CTP or m-CTP-p), we indicate the category of the instances (Column *Category*) and the number of instances in the corresponding category (Column *#Inst.*). Besides, for each compared methodologies, we present the number of proven optimal solutions within the time limit (Column *#Opt*) and the average total consumed time in seconds (Column *T(s)*).

			[2]		Proposed BCP			
					Without *CCC*		With *CCC*	
Problem	Category	#Inst	#Opt	T(s)	#Opt	T(s)	#Opt	T(s)
m-CTP-p	$\|V'\| = 100, \|M\| = 0$	32	32	4.2	32	1.7	32	1.9
m-CTP-p	$\|V'\| = 100, \|M\| > 0$	32	31	$\geq$ 151.3	32	55.1	32	2.1
m-CTP-p	$\|V'\| = 200$	32	27	$\geq$ 966.3	32	132.5	32	6.1
m-CTP	$\|V'\| = 100, \|M\| = 0$	64	-	-	64	3.8	64	14.0
m-CTP	$\|V'\| = 100, \|M\| > 0$	64	-	-	64	1.7	64	2.0
m-CTP	$\|V'\| = 200$	64	-	-	63	206.1	63	128.1

Table 1: Results over literature instances

Table 1 shows that our new BCP algorithm (without or with cuts CCC) outperformed the one proposed by [2] in all categories, in time and number of optimally solved instances. It optimally solved all m-CTP-p instances even when the CCC are not used. The following five m-CTP-p instances are optimally solved for the first time in this paper: A2-20-100-100-6, A2-20-100-100-8, B2-1-100-100-8, B2-20-100-100-6, and B2-20-100-100-8 with optimal solution costs equal to respectively 20966, 18415, 13137, 25960, and 22082. When the CCC are separated, our algorithm further improves its performance by more than one order of magnitude.

Regarding m-CTP instances, the two versions of our algorithm optimally solved 195 out of 196 instances. Note that the improvement obtained for these instances with the cuts only occurred for the largest-instances category.

Table 2 shows the gap reduction caused by the use of different types of cuts for large literature instances ($|V| = 200$). For each instance, that table presents the percentage relative difference between the best known solution and the lower bound obtained by pure column generation at the root node (Column *CG Gap*), the percentage gap reduction obtained when only the literature cuts (3) are applied to the column-generation relaxation (Column *Cuts Lit.*), the percentage gap reduction obtained when the CCC are applied to the column-generation relaxation with literature cuts (Column *Cuts CCC*), and the percentage gap reduction obtained when the limited memory Chvátal-Gomory Rank-1 cuts provide by the VRPSolver framework are applied to the column-generation relaxation with both the literature and the proposed cuts (Column *Cuts R1C*).

Table 2 shows the substantial reduction of the remaining root gap obtained with the CCC. The average reduction is higher for m-CTP-p instances. This fact was already expected since the $K(S)$ calculus does not consider the maximum cost of a route q that is constrained in m-CTP instances. Despite this, the smallest average root gap reduction was 42.88%.

Instance	*m*-CTP-*p*				*m*-CTP($q = 500$)				*m*-CTP($q = 250$)			
	CG	Cuts			CG	Cuts			CG	Cuts		
	Gap	Lit.	*CCC*	Rank-1	Gap	Lit.	*CCC*	Rank-1	Gap	Lit.	*CCC*	Rank-1
A2-1-100-100-4	4.22	0.20	77.8	100.00	3.25	0.00	100.00	-	6.80	0.35	39.61	16.73
A2-1-100-100-5	2.14	15.07	100.00	-	1.86	11.58	100.00	-	0.40	2.33	100.00	-
A2-1-100-100-6	6.69	2.39	100.00	-	5.47	4.20	83.62	100.00	3.63	0.54	89.13	100.00
A2-1-100-100-8	6.50	4.57	65.60	100.00	8.32	0.24	26.21	15.13	7.71	0.26	25.82	9.89
A2-20-100-100-4	1.82	0.00	38.84	56.08	1.80	0.21	36.74	60.07	1.91	0.79	44.25	41.28
A2-20-100-100-5	4.15	1.13	48.28	89.74	3.39	0.63	62.44	82.77	3.49	2.56	65.21	100.00
A2-20-100-100-6	4.33	1.76	54.99	100.00	3.42	1.39	74.26	100.00	5.67	1.96	39.85	78.70
A2-20-100-100-8	7.06	1.92	46.12	85.59	5.07	1.07	63.82	86.57	8.46	0.98	38.49	89.67
B2-1-100-100-4	3.49	0.78	21.66	71.94	4.94	0.65	10.04	26.70	4.70	0.80	17.84	34.03
B2-1-100-100-5	2.15	2.05	100.00	-	4.51	1.36	25.58	59.89	5.44	1.33	14.04	41.96
B2-1-100-100-6	4.24	0.79	44.16	75.64	4.86	0.82	37.57	36.06	7.19	0.54	23.26	38.68
B2-1-100-100-8	5.36	1.99	25.51	100.00	6.18	0.49	45.97	81.90	17.16	0.15	2.36	17.84
B2-20-100-100-4	1.35	3.27	100.00	-	1.35	3.49	100.00	-	1.35	3.49	82.84	100.00
B2-20-100-100-5	2.97	0.92	30.72	78.67	2.95	0.81	33.60	73.56	2.95	1.04	31.35	79.97
B2-20-100-100-6	3.01	0.64	39.82	50.32	3.04	1.90	46.84	78.88	3.03	1.65	39.84	45.38
B2-20-100-100-8	4.85	1.68	31.05	100.00	4.83	2.06	33.52	100.00	4.49	1.61	32.21	60.82
Average	4.02	2.44	57.78	83.99	4.08	1.93	55.01	69.34	5.27	1.27	42.88	56.99

Table 2: Gap reduction comparison.

5.2 Results on the new proposed instances

We introduce a new set of m-CTP-p instances, which is derived from CVRP instances belonging to the set X proposed by [18]. Each CVRP instance within this set is labeled as $X - n - k$, with n signifying the number of nodes and k representing the number of vehicles. Given a CVRP instance, we generate an m-CTP-p instance as follows. The first node of the CVRP instance serves as the depot, while the subsequent $|M|$ nodes denote mandatory facilities. The remaining $n - |M| - 1$ nodes, represent both optional facilities and customers. To achieve this representation, we create a duplicate node for each node of this type. The original node represents an optional facility, while its duplicate is a customer. The construction of coverage sets ϕ considers that each facility i covers every customer j if $c_{(i,j)} \leq \frac{1000}{\sqrt{n}}$ in the original CVRP instance. The parameter p is defined as $p = \lceil \frac{n}{k} \rceil$. For each one of the 58 CVRP instances within the set X with $101 \leq n \leq 393$, we generate two m-CTP-p instances by setting $|M| = 0$ and $|M| = \lceil 0.2n - 1 \rceil$.

To compute the initial upper bounds for the proposed instances, we used a simple and effective heuristic for m-CTP-p available at https://github.com/brunomattos1/m-ctp_heuristic. Roughly speaking, it generates random subsets of optional facilities that cover all customers and iterates perturbing and evaluating them in the same fashion as [27]. In each subset evaluation, it finds a set of routes visiting them and the mandatory facilities using the heuristic proposed by [28], which is configured to generate only 3 tentative individuals (solutions) and return the best of them. For each instance, we let the heuristic run for 3 minutes.

Table 3 presents average results obtained by the application of the proposed BCP model with CCC to the new set of instances. These instances are categorized into six groups based on two criteria: the value of $|M \cup O|$ (ranging from 100 to 200, 200 to 300, and 300 to 400) and the value of $|M|$ ($|M| = 0$ or $|M| > 0$). Table 3 employs the following columns to present statistical information for each group. Column *#Inst.* denotes the count of instances within the group. Column *#Opt.* represents the number of instances in the group that were optimally solved within a time limit of 18000 seconds. Column *#Nodes.* indicates the average total number of the tree nodes considering only instances that were optimally solved. Column *#Gap_0.* presents the average relative percentage difference between the optimal cost and the lower bound at the root node. This statistic is also limited to instances that were optimally solved. Finally, Column *#T(s).* displays the average total computational time for instances that were optimally solved.

The results present in Table 3 indicate that the proposed instances are considerably more challenging than those found in the existing literature. Regarding instances with $|M| = 0$, the algorithm successfully solved only 16 out of 21 of the smallest instances, with average nodes of 66, average Gap_0 of 1.15%, and average total execution time of 2350.49s. Besides, for instances with $200 \leq |M \cup O| < 300$, the algorithm was able to optimally solve only 5 out of 22 instances. Finally, our algorithm did not optimally solve any instance with $300 \leq |M \cup O|$.

For instances with $|M| > 0$, the algorithm exhibited a similar performance profile, leading us to conclude that the presence of mandatory facilities does not pose a significant challenge. For the smallest group of instances, the algorithm optimally solved 15 out of 21 instances, achieving an average node count of 26.06, an average Gap_0 of 0.60%, and an average execution time of 2953.39 seconds. For instances with $200 \leq |M \cup O| < 300$, the algorithm managed to optimally solve only 3 out of 22 instances. Furthermore, none of the largest instances were optimally solved.

The Figure 4 shows the optimal solution for the instance *X-n101-k25-p5*, with $M = \emptyset$, where the nodes marked in red represent the optional facilities that are visited by some vehicle. Around each one of these nodes i, we delineate a dashed red circumference. Each node j within the circumference is such that $i \in \phi(j)$.

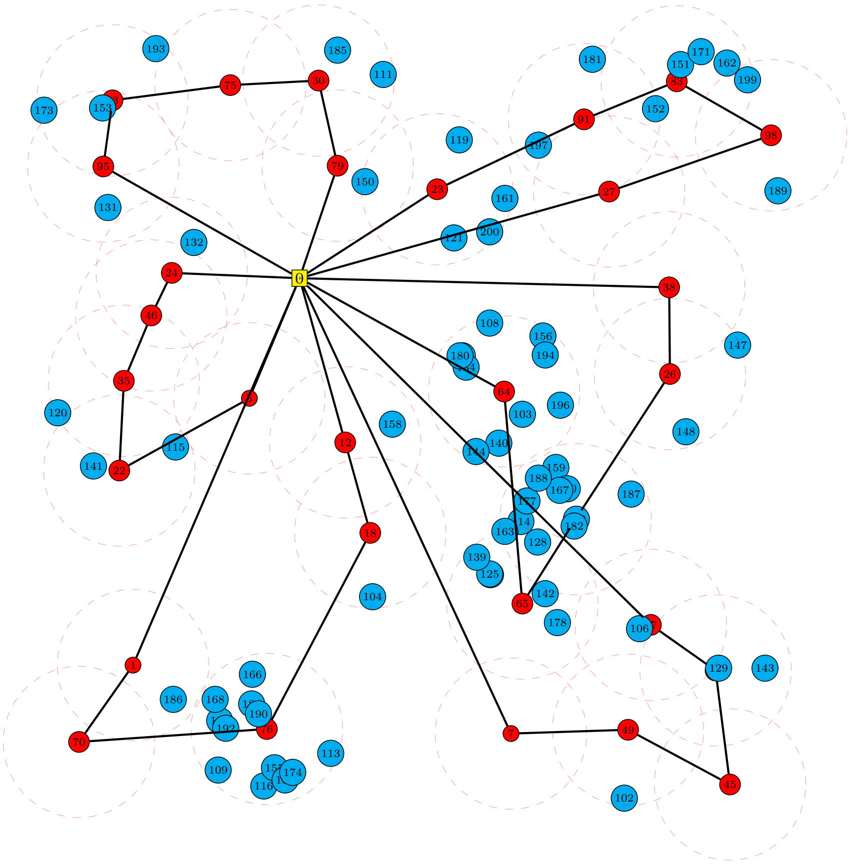

Fig. 4: Solution of the instance *X-n101-k25-p5*.

Note that the customers 5, 7, 27, 38, 45, 70 and 75 are only covered by the facility that is placed at the same place that them. Therefore, they must be visited.

6 Conclusion

In this paper, we addressed the m-CTP, which is a variant of the Vehicle Routing Problems where facilities are visited by the routes to cover customers from a given network. In this problem, the routes are limited by the number of facilities visited and the total cost. For that problem, we devised a BCP model coded within the VRPSolver framework, presented an effective new family of cuts that explores two main structures of the problem, the covering problem and the capacity of the vehicles. We also proposed a set of large instances.

To prove the effectiveness of the proposed method and the proposed cuts, we conducted several computational experiments that are divided into two parts. In the first one, we applied the two versions of the proposed model (with and without proposed cuts) to 288 literature benchmark instances. The results showed that our method optimally solved all but one instance. Besides, we compared our algorithm to the best literature one that is specialized for the problem variant where the routes are limited only on the number of visited facilities. This comparison showed that our approach outperformed the literature one even when we did not consider the proposed cuts. In the second part of the experiments, we applied our approach to the newly proposed instances and concluded that they are considerably more challenging than the literature ones, being suitable for future research developments.

For future research, we intend to adapt the proposed cuts to other VRPs whose routes are also limited on the number of visited nodes.

Acknowledgments

Artur Alves Pessoa received support from CNPq, grant 306033/2019-4.

Conflicts of interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

[1] Hachicha, M., Hodgson, M.J., Laporte, G., Semet, F.: Heuristics for the multi-vehicle covering tour problem. Computers & Operations Research, **27**, 29–42 (2000)

[2] Glize, E., Roberti, R., Jozefowiez, N., Ngueveu, S.U.: Exact methods for mono-objective and bi-objective multi-vehicle covering tour problems. European Journal of Operational Research **3**, 812–824 (2020)

[3] Lopes, R., Souza, V.A., DA Cunha, A.S.: A branch-and-price algorithm for the multi-vehicle covering tour problem. Electronic Notes in Discrete Mathematics **44**, 61–66 (2013)

[4] Irnich, S., Desaulniers, G.: Shortest Path Problems with Resource Constraints, pp. 33–65 (2006). https://doi.org/10.1007/0-387-25486-2_2

[5] Jozefowiez, N.: A branch-and-price algorithm for the multivehicle covering tour problem. Networks **64**, 160–168 (2014)

[6] Ha, M.H., Bostel, N., Langevin, A., Rousseau, L.M.: An exact algorithm and a metaheuristic for the multi-vehicle covering tour problem with a constraint on the number of vertices. European Journal of Operational Research **226**, 211–220 (2013)

[7] Kammoun, M., Derbel, H., Ratli, M., Jarboui, B.: A variable neighborhood search for solving the multi-vehicle covering tour problem. Electronic Notes in Discrete Mathematics **47**, 285–292 (2015)

[8] Allahyari, S., Salari, M., Vigo, D.: A hybrid metaheuristic algorithm for the multi-depot covering tour vehicle routing problem. European Journal of Operational Research **242**(3), 756–768 (2015)

[9] Flores-Garza, D.A., Salazar-Aguilar, M.A., Ngueveu, S.U., Laporte, G.: The multi-vehicle cumulative covering tour problem. Annals of Operations Research **258**, 761–780 (2017)

[10] Pham, T.A., Hà, M.H., Nguyen, X.H.: Solving the multi-vehicle multi-covering tour problem. Computers & Operations Research **88**, 258–278 (2017)

[11] Karaoğlan, I., Erdoğan, G., Koç: The multi-vehicle probabilistic covering tour problem. European Journal of Operational Research **271**(1), 278–287 (2018)

[12] Margolis, J.T., Song, Y., Mason, S.J.: A multi-vehicle covering tour problem with speed optimization. Networks **79**, 119–142 (2021)

[13] Baldacci, R., Mingozzi, A., Roberti, R.: New route relaxation and pricing strategies for the vehicle routing problem. Operations Research **59**(5), 1269–1283 (2011) https://doi.org/10.1287/opre.1110.0975

[14] Pecin, D., Pessoa, A., Poggi, M., Uchoa, E.: Improved branch-cut-and-price for capacitated vehicle routing. Mathematical Programming Computation **9** (2016) https://doi.org/10.1007/s12532-016-0108-8

[15] Contardo, C., Martinelli, R.: A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. Discrete Optimization **12** (2014) https://doi.org/10.1016/j.disopt.2014.03.001

[16] Pessoa, A., Sadykov, R., Uchoa, E., Vanderbeck, F.: A generic exact solver for vehicle routing and related problems. Mathematical Programming **183**, 483–523 (2020)

[17] Gendreau, M., Laporte, G., Semet, F.: The covering tour problem. Operations Research **45**(4), 568–576 (1997)

[18] Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., Subramanian, A.: New benchmark instances for the capacitated vehicle routing problem. European Journal of Operational Research **257**(3), 845–858 (2017)

[19] Righini, G., Salani, M.: Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optimization **3**(3), 255–273 (2006) https://doi.org/10.1016/j.disopt.2006.05.007

[20] Sadykov, R., Uchoa, E., Pessoa, A.: A bucket graph–based labeling algorithm with application to vehicle routing. Transportation Science **55**(1), 4–28 (2021) https://doi.org/10.1287/trsc.2020.0985

[21] Achterberg, T., Koch, T., Martin, A.: Branching rules revisited. Operations Research Letters **33**(1), 42–54 (2005) https://doi.org/10.1016/j.orl.2004.04.002

[22] Pessoa, A., Sadykov, R., Uchoa, E., Vanderbeck, F.: Automation and combination of linear-programming based stabilization techniques in column generation. INFORMS Journal on Computing **30** (2018) https://doi.org/10.1287/ijoc.2017.0784

[23] Roboredo, M., Sadykov, R., Uchoa, E.: Solving vehicle routing problems with intermediate stops using vrpsolver models. Networks **81**(3), 399–416 (2023)

[24] Soares, V.C., Roboredo, M.: On the exact solution of the multi-depot open vehicle routing problem. Optim Lett (2023) https://doi.org/10.1007/s11590-023-02072-y

[25] Ben Mohamed, I., Klibi, W., Sadykov, R., Şen, H., Vanderbeck, F.: The two-echelon stochastic multi-period capacitated location-routing problem. European Journal of Operational Research **306**(2), 645–667 (2023) https://doi.org/10.1016/j.ejor.2022.07.022

[26] Bulhões, T., Sadykov, R., Subramanian, A., Uchoa, E.: On the exact solution of a large class of parallel machine scheduling problems. Journal of Scheduling **23** (2020) https://doi.org/10.1007/s10951-020-00640-z

[27] Subramanian, A., Uchoa, E., Ochi, L.S.: A hybrid algorithm for a class of vehicle routing problems. Computers & Operations Research **40**(10), 2519–2531 (2013)

[28] Vidal, T.: Hybrid genetic search for the CVRP: Open-source implementation and swap* neighborhood. Computers & Operations Research **140**, 105643 (2022)

	$\|M\| = 0$					$\|M\| > 0$				
			Avg.					Avg.		
Category	#Inst	#Opts	#Nodes	Gap_0	Time(s)	#Inst	#Opts	#Nodes	Gap_0	Time(s)
$100 \leq \|M \cup O\| < 200$	21	16	66	1.15%	2350.49	21	15	26.06	0.60%	2953.39
$200 \leq \|M \cup O\| < 300$	22	5	35.8	0.70%	5195.40	22	3	82.23	0.58%	14766.98
$300 \leq \|M \cup O\| < 400$	15	0	-	-	-	15	0	-	-	-

Table 3: Results over proposed instances.