Research Practice 2
Final report

Heuristic and exact solution strategies for the Team Orienteering Problem

Camila Mejía Quintero · Miguel Tamayo Jaramillo · Juan Carlos Rivera Agudelo

May 27, 2016

Abstract The aim of this research practice is to compare exact solution approaches and a matheuristic approach solving the team orienteering problem (TOP). The matheuristic is based on the hybridization of mathematical programming formulations and a large neighborhood search heuristic (LNS). TOP is a variant of the classic vehicle routing problem, where m teams seek to maximize the total collected profit, visiting as many nodes as it can be possible without exceeding a time limit. This research proposes four new mix integer linear programming (MILP) models, and presents a constraint programming (CP) model. The matheuristic method is based on one of the MILP model proposed and it also has a post-optimization phase which improves the solution based on a set partitioning model. The performance of the solution methods are compared by using benchmark instances from the literature.

Keywords Team Orienteering Problem, MILP, CP, vehicle routing, matheuristics, hybrid methods

1 Introduction

The team orienteering problem (TOP) is a generalization of the orienteering problem (OP), which can be defined on a complete indirected graph $G = (V, A)$ where the node-set $V = \{1, ..., n\}$ includes a starting node 1, a finish node n and a subset $V' = V \setminus \{1, n\}$ of profitable nodes. The arc-set $A = \{(i, j) \mid i, j \in V\}$ is composed by edges (i, j) with travel times d_{ij}. There is also a set $K = \{1, ..., m\}$ of m vehicles which are available to visit the nodes in V' and the goal is to determine m routes, without exceeding a given threshold L_{max}, that maximize the total collected prize, due to each node $i \in V'$ has a specific profit p_i. No node can be visited more than once by one or several routes and there is the possibility of not visiting all nodes.

The TOP can be modeled as a multi-level optimization problem. At the first level, it is necessary to select a subset of points to visit for the team in order to maximize the total profit. At the second level, the selected points are assigned to each member of the team. At the third level, it is necessary to construct a feasible route through those points to each member [Chao et al., 1996]. In the figure 1 there is a TOP solution for a specific instance.

C. Mejía
Student of Mathematical Engineering
Universidad EAFIT, Medellín, Colombia. E-mail: cmejia3@eafit.edu.co

M. Tamayo
Student of Mathematical Engineering
Universidad EAFIT, Medellín, Colombia. E-mail: mtamayo6@eafit.edu.co

J.C. Rivera
Tutor Professor
Department of Mathematical Sciences, Universidad EAFIT, Medellín, Colombia. E-mail: jrivera6@eafit.edu.co

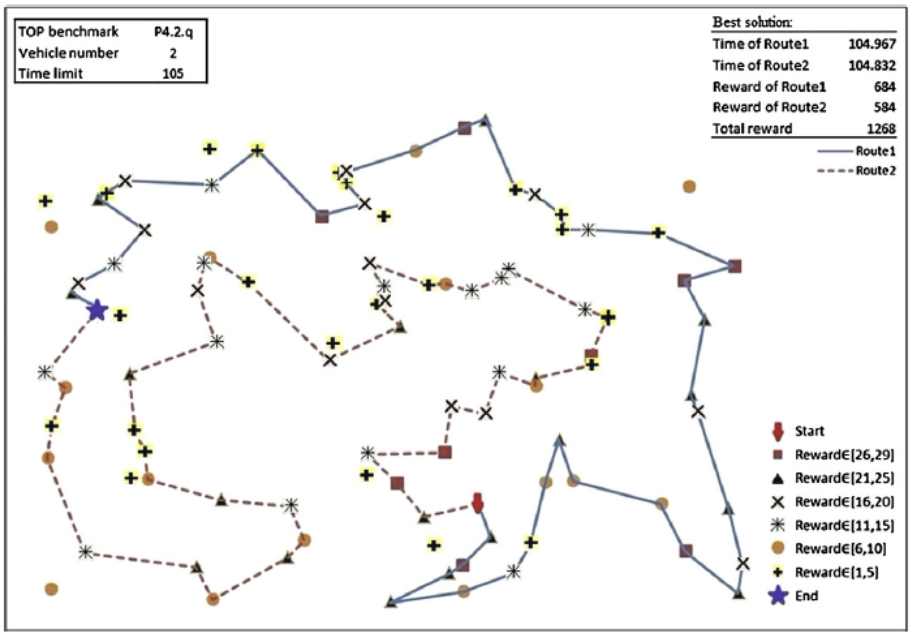

Fig. 1 A TOP problem (instance p4.2.17 and a solution with total reward of 1268). Taken from [Kim et al., 2013] p. 3066

The aim of this research practice is to compare three different paradigms, divided in two groups, to solve the optimization problem already described. The first group, considers exact strategies, constraint programming (CP) and mixed integer linear programming (MILP) are used. The second group, a hybrid mateheuristic method combining LNS, first proposed by [Shaw, 1997], with an exact mathematical models in the rebuilt phase and the post-optimization phase.

The paper is followed by Section 2 with a brief summary of the literature review for the TOP, including a review of which methods have been used to solve it. Then, on Section 3 it is described five mathematical models that are compared, including four MILP and one CP. In Section 4 the proposed matheuristic is described. Finally, Sections 5 and 6 show the results, conclusions and further work.

2 Literature review

The TOP was first proposed and solved by [Butt and Cavalier, 1994], but it was named the Multiple Tour Maximum Collection Problem. They exposed a MILP and a basic constructive heuristic algorithm, which the ratio between the profit and the distance needed to visit a node is the criteria used to add a node to a route. A while after, [Butt and Ryan, 1999] proposed a procedure based on a set-partitioning formulation which makes efficient the use of both column generation and constraint branching. In fact, this algorithm works well for small number of nodes.

In addition, [Chao et al., 1996] benchmarks data sets and solves them with a heuristic model. In its approach, it uses an initial solution created by the closest nodes to the start and finish nodes. Then, a discrete annealing algorithm is applied to improve the routes, it also allows node removal and reinsertion. Then, [Tang and Miller-Hooks, 2005] presents a tabu search (TS) with an adaptive memory process changing between small and large neighborhoods based on greedy procedures.

Moreover, [Archetti et al., 2007] also presents a TS algorithm allowing both feasible and infeasible movements and a variable neighborhood search (VNS) with feasible move. The latter, outperforms [Chao et al., 1996] and [Tang and Miller-Hooks, 2005]. Then, [Ke et al., 2008] proposes an ant colony

optimization (ACO) based on the sequential, deterministic-concurrent, random-concurrent, and simultaneous methods as the construction algorithms.

Also, [Vansteenwegen et al., 2009] exposes an MILP model and an algorithm that merges a guided local search with a diversifying model named disturb. However their computational time is shorter in comparison with the former models, its results do not outperform other methods. Furthermore, [Dang et al., 2011] present a particle swarm optimization based on memetic algorithm (PSOMA) which uses a tour-splitting method and genetic mixture improvement. Afterward, they extend their research and present a PSO inspired algorithm (PSOiA) [Dang et al., 2013]. In this case, they improve the algorithm and find the best values so far for every benchmark instance. Finally, [Kim et al., 2013] presents different local search heuristic approaches attempting to improve solutions via incremental adjustments to current solutions. They find the same results as the PSOiA algorithm, but in shorter times.

3 Mathematical formulations

This problem can be modeled as a mixed integer linear program (MILP) as [Tang and Miller-Hooks, 2005] shows. There are different ways to describe it by a mathematical model. Some constraints and the objective function do not variate, the difference is the decision variables' dimensions, as well as adding or removing them.

Here, it is described the model exposed in [Tang and Miller-Hooks, 2005]. In addition, four MILP models and one CP model are proposed.

In order to better describe the mathematical models, two new subsets of nodes are defined as follows: The subset $V^1 = V \setminus \{1\}$ is the set of all nodes without the starting point, which is useful to limit the possible destination nodes of traversed arcs. Similarly, the subset $V^2 = V \setminus \{n\}$ is the set of all nodes without the finish point, which is useful to limit the possible origin nodes of traversed arcs.

3.1 Model 1

This model uses the binary decision variables y_{ik}, which represent whether the node $i \in V'$ is visited ($y_{ik} = 1$) or not ($y_{ik} = 0$) by the vehicle k, and x_{ijk}, which represents the number of times that the arc (i, j) is traversed by the vehicle k. The model proposed by [Tang and Miller-Hooks, 2005] includes service times s_i for each node $i \in V'$, nevertheless the instances solved in this research always consider $s_i = 0$. It is important to note that this formulation also considers that the starting and finish nodes are the same and the decision variable x_{ijk} indicates the number of times an arc is traversed but not the direction. The latter implies that variables x_{ijk} only exist if $i < j$, saving some memory and computations, and when a route visits only one node variable x_{0jk} can be equal to 2.

$$max \sum_{i \in V'} \sum_{k \in K} p_i \cdot y_{ik} \tag{1}$$

$$s.t. \sum_{j \in V'} \sum_{k \in K} x_{1jk} = 2m \tag{2}$$

$$\sum_{i<j} x_{ijk} + \sum_{i>j} x_{jik} = 2y_{jk} \qquad \forall\, i, j \in V', k \in K \tag{3}$$

$$\sum_{i \in V'} \sum_{j>i} d_{ij} x_{ijk} + \sum_{i \in V'} s_i y_{ik} \leq L_{max} \qquad \forall\, j \in V', k \in K \tag{4}$$

$$\sum_{k \in K} y_{ik} \leq 1 \qquad \forall\ i \in V' \tag{5}$$

$$\sum_{i \in U} \sum_{j \in U} x_{ijk} \leq |U| - 1 \qquad \forall\ U \subset V^1, 2 \leq |U| \leq n-2, k \in K \tag{6}$$

$$x_{1jk} \in \{0,1,2\} \qquad \forall\ j \in V, k \in K \tag{7}$$

$$x_{ijk} \in \{0,1\} \qquad \forall\ k \in K, i,j \in V', i \leq j \tag{8}$$

$$y_{ik} \in \{0,1\} \qquad \forall\ i \in V', k \in K \tag{9}$$

Objective function is given by (1) which is to maximize the total collected profit. In fact, (2) ensures that there are m routes starting and finishing at node 1. The connectivity of each node is checked on (3) and the time limit of each route is verified on (4). Then, (5) ensures that each node is visited at the most ones. The sub-tours are forbidden by (6), where the subsets U contain the nodes which perform a loop in a given solution. Note that this set of constraints requires that the model must be solve iteratively, adding new constraints on each iteration when a loop is found. Finally, (7) to (9) defines the domain of the decision variables.

3.2 Model 2 - MILP$_1$

The first MILP formulation proposed is flow based models for vehicle routing problems. The decision variables are x_{ij}, which are equal to 1 if arc (i,j) is traversed and 0 otherwise, and L_i, which is the crossed distance from node 0 to node i when the latter is visited.

$$\max z = \sum_{i \in V^1} p_i \cdot x_{1j} \tag{10}$$

$$\sum_{j \in V^1} x_{1j} \leq m \tag{11}$$

$$\sum_{i \in V^2} x_{in} \leq m \tag{12}$$

$$\sum_{\substack{i \in V^2 \\ i \neq j}} x_{ij} \leq 1 \qquad \forall j \in V' \tag{13}$$

$$\sum_{\substack{j \in V^1 \\ i \neq j}} x_{ij} \leq 1 \qquad \forall i \in V' \tag{14}$$

$$\sum_{\substack{i \in V^2 \\ i \neq j}} x_{ij} = \sum_{\substack{i \in V^1 \\ i \neq j}} x_{ji} \qquad \forall j \in V' \tag{15}$$

$$L_j \geq L_i + d_{ij} \cdot x_{ij} - L_{max} \cdot (1 - x_{ij}) \qquad \forall\ i \in V^2, j \in V^1, i \neq j \tag{16}$$

$$x_{ii} = 0 \qquad \forall i \in V' \tag{17}$$

$$0 \leq L_i \leq L_{max} \qquad \forall i \in V \tag{18}$$

$$x_{ij} \in \{0,1\} \qquad \forall\ i \in V^1, j \in V^2 \tag{19}$$

The objective function, given by (10), refers to the total profit maximization. Therefore, (11) makes sure that there are just at most m routes and they have to start from node 1. Then, (12) verifies that at most m arcs (i,n) are traversed. Also, (13) and (14) check that each node is not visited more than once and (15) guarantee that the number of arcs in is equal to the number of arcs out. In other words, if a vehicle visits node j, it must leave it. If $x_{ij} = 1$, the distance L_j has to be greater than the distance

that has been crossed at L_i plus the distance between them, d_{ij}, otherwise is useless, this constraint is represented by (16). Then, (17) avoids the creation of loops from a node i to itself. Finally, the decision variables' domain are specified on (18) and (19).

3.3 Model 3 - MILP$_2$

The second MILP model proposed is based on the previous one, so it has the same parameters and decision variables. However, a new decision variable y_i is added, which is equal to 1 if node i is visited. Specifically, (10), (13), (14) and (15) are reformulated by using the new variables.

$$\max z = \sum_{i \in V'} p_i \cdot y_i \tag{20}$$

$$\sum_{j \in V^1} x_{0j} = m \tag{21}$$

$$\sum_{i \in V^2} x_{in-1} = m \tag{22}$$

$$\sum_{i \in V^2} x_{ij} = y_j \qquad \forall j \in V' \tag{23}$$

$$\sum_{j \in V^1} x_{ij} = y_i \qquad \forall i \in V' \tag{24}$$

$$L_j \geq L_i + d_{ij} \cdot x_{ij} - L_{max} \cdot (1 - x_{ij}) \qquad \forall\, i \in v^2, j \in v^1, i \neq j \tag{25}$$

$$0 \leq L_i \leq L_{max} \qquad \forall i \in V \tag{26}$$

$$x_{ii} = 0 \qquad \forall\, i \in V \tag{27}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, i \in V^1, j \in V^2 \tag{28}$$

$$y_i \in \{0, 1\} \qquad \forall\, i \in V \tag{29}$$

The objective function is given by (20) and seek to maximize the total collected profit. Therefore, (21) and (22) verify that there are exactly m routes. Then, (23) and (24) ensure that if a node $i \in v'$ is visited ($y_i = 1$) it must be a vehicle which enters and leaves the node, otherwise ($y_i = 0$) which implies that $x_{ij} = 0$. Finally, (25) to (28) are the same constraint as (16) to (19) and (29) is the domain constraint of the y_i.

3.4 Model 4 - MILP$_3$

This model is based on the model propsed by [Rivera, 2014] for the multitrip cumulative capacitated single-vehicle routing problem. This model has two decision variables: y_i^k which is equals to 1 if node i is visited by vehicle k and w_{ij}, which is equals to 1 if node j is visited after i, but it does not have to be just after. It is important to note that $y_i^k \neq y_i$.

The objective function is to maximize the profit and is given by (30). Therfore, (31) and (32) ensure that if node i is visited before j, then j cannot be visited before i and vice versa. (33) constraint ensure that each node is visited just once. Then, (34) and (35) verify that if node i is visited before j then the distance traversed until node j (L_j) must to be greater than the distance L_i plus the distance between them. Afterwards, (36) checks that the time threshold is not exceeded. And finally, (37) and (39) are the decision variables' domains.

$$\max z = \sum_{k \in K} \sum_{i \in V'} p_i \cdot y_i^k \tag{30}$$

$$w_{ij} + w_{ji} \geq y_j^k + y_i^k - 1 \qquad \forall i \in V^2, j \in V^1, k \in K, i \neq j \tag{31}$$

$$w_{ij} + w_{ji} \leq \sum_{k \in K} y_i^k \qquad \forall i \in V^2, j \in V^1, k \in K, i \neq j \tag{32}$$

$$\sum_{k \in K} y_i^k \leq 1 \qquad \forall i \in V' \tag{33}$$

$$L_j \geq L_i + d_{ij} \cdot w_{ij} - L_{max} \cdot (1 - w_{ij}) \qquad \forall i \in V, j \in V, i \neq j \tag{34}$$

$$L_j \geq L_i + d_{ij} \cdot w_{ij} - L_{max} \cdot w_{ji} \qquad \forall i \in V, j \in V, i \neq j \tag{35}$$

$$L_i + d_{i,n} \leq L_{max} \qquad \forall i \in V^2, k \in K \tag{36}$$

$$w_{ij} \in \{0, 1\} \qquad \forall\ i, \quad j \in V' \tag{37}$$

$$y_j^k \in \{0, 1\} \qquad \forall j \in V', \quad k \in K \tag{38}$$

$$L_j \geq 0 \qquad \forall i \in V \tag{39}$$

3.5 Model 5 - MILP$_4$

This model is based on the concept of replenishment arcs presented by [Mak and Boland, 2000, Rivera et al., 2015], but it is adapted to solve the TOP. Replenishment arcs are a trick to replace a set of routes by a large single-route: where the last visited node of a route is i and the first visited node of other route is j, the arcs $(i, 0)$ and $(0, j)$ can be replaced by a replenishment arc (i, j) where the travel time at node j becomes $d_0 j$. For instance, the solution matrix x_{ij} takes each route separately, but x'_{ij} unifies the m routes in a single route. It is important to mention that replenishment arcs are used by [Mak and Boland, 2000] to handle a traveling salesman problem where vehicles can replenish fuel while traversing an arc, and [Rivera et al., 2015] adapts the concept to model multiple trips performed by a vehicle where replenishment arcs mean the transition between a route and its subsequent.

$$\max z = \sum_{i \in V'} p_i \cdot y_i \tag{40}$$

$$\sum_{\substack{i \in V^2 \\ i \neq j}} x_{ij} + \sum_{\substack{i \in V' \\ i \neq j}} x'_{ij} = y_j \qquad \forall j \in V' \tag{41}$$

$$\sum_{\substack{j \in V^1 \\ i \neq j}} x_{ij} + \sum_{j \in V', i \neq j} x'_{ij} = y_i \qquad \forall i \in V' \tag{42}$$

$$\sum_{i \in V^2} x_{0i} = 1 \tag{43}$$

$$\sum_{i \in V^1} x_{in-1} = 1 \tag{44}$$

$$\sum_{i \in V'} \sum_{\substack{j \in V' \\ i \neq j}} x'_{ij} = m - 1 \tag{45}$$

$$L_j \geq d_{0j} \cdot y_j \qquad \forall j \in V^1 \tag{46}$$

$$L_j \geq L_i + d_{ij} \cdot x_{ij} - L_{max} \cdot (1 - x_{ij}) \qquad \forall i \in V^2 . j \in V^1, i \neq j \tag{47}$$

$$L_i + d_{in-1} \cdot y_i \leq L_{max} \qquad \forall i \in V^2 \tag{48}$$

$$x_{ii} = 0 \qquad \forall i \in V \tag{49}$$
$$L_j \geq 0 \qquad \forall j \in V' \tag{50}$$
$$x_{ij} \in \{0,1\} \qquad \forall i \in V^2, \quad j \in V^1 \tag{51}$$
$$x'_{ij} \in \{0,1\} \qquad \forall i \in V', \quad j \in V^1 \tag{52}$$
$$y_i \in \{0,1\} \qquad \forall i \in V' \tag{53}$$

As well as the previous model, the objective function (40) is to maximize the collected profit. Therefore, (41) and (42) forbid the presence of any node in 2 or more routes. Then, (43) and (44) ensure that merged route starts on the initial depot and finishes on the last one. Also, (45) establishes that there are exactly m routes. Notes that m routes implies to have exactly $m-1$ replenishment arcs. Afterwards, (46) to (48) guarantee that the time in each node is feasible given the time limit threshold. (49) ensures that there is no loop on any node. Finally, (50) to 52 define the domain for the decision variable.

3.6 Model 6 - CP

In this last formulation, a different paradigm is used to model the TOP based on CP. In this formulation the decision variables a_i and s_i indicate the antecedent visited node and the subsequent visited node of the node $i \in V$, respectively. The decision variable t_i represents the team (member) which visits the node $i \in V$. In addition the decision variable L_i from previous formulations is still valid in this model.

Some different sets and parameters are required. This model supposes that all nodes are assigned to a vehicle, so a dummy vehicle is used to create an extra route for the unvisited nodes with no time limit and no profit. Here H is the set of nodes, including required nodes and start and finish nodes. In order to model some constraints, we replicate m times, once for each vehicle, the start nodes which become the set H^S, and finish nodes which become the set H^F. These sets H^S and H^F belong to H. Similar to previous nodes, H' is the set of required nodes ($H' = H \setminus (H^S \cup H^F)$). H^1 is defined alike V^1 from previous models ($H^1 = H \setminus H^S$). The set of vehicles K' is composed by the set of available vehicles K and the additional dummy vehicle.

With these definitions, the model can be formulated, based on [Kilby and Shaw, 2006], as follows:

$$\max Z = \sum_{j \in H'} p_j \cdot (1 - (t_j = m+1)) \tag{54}$$
$$t_{i-1} = i \qquad \forall\, i \in K' \tag{55}$$
$$t_{n+m+i-2} = i \qquad \forall\, i \in K' \tag{56}$$
$$t_i = t_{a_i} \qquad \forall\, i \in H \tag{57}$$
$$t_i = t_{s_i} \qquad \forall\, i \in H \tag{58}$$
$$a_{i-1} = n + m + i - 2 \qquad \forall\, i \in K' \tag{59}$$
$$s_{n+m+i-2} = i - 1 \qquad \forall\, i \in K' \tag{60}$$
$$a_i \neq i \qquad \forall\, i \in H \tag{61}$$
$$a_{s_i} = i \qquad \forall\, i \in H \tag{62}$$
$$a_i \neq a_j \qquad \forall\, i, j \in H,\ i \neq j \tag{63}$$
$$L_{i-1} = 0 \qquad \forall\, i \in K' \tag{64}$$
$$L_i = (L_{a_i} + d_{a_i i}) \cdot (1 - (t_i = m+1)) \qquad \forall\, i \in H^1 \tag{65}$$
$$L_i * (1 - (t_i = m+1)) \leq L_{max} \qquad \forall\, i \in H^1 \tag{66}$$

$$a_i \in \mathbb{Z}^+ \qquad \forall\, i \in H \tag{67}$$
$$s_i \in \mathbb{Z}^+ \qquad \forall\, i \in H \tag{68}$$
$$t_i \in \mathbb{Z}^+ \qquad \forall\, i \in H \tag{69}$$
$$L_i \geq 0 \qquad \forall\, i \in H \tag{70}$$

The objective function (54) collects the profit of visited vehicles. Note that only vehicles belonging to the set K are considering since the nodes visited by the dummy vehicle are not part of the solution. (55) and (56) fix the initial and final node for each vehicle. Therefore, (57) and (58) indicate that each node must be visited by a vehicle and ensure that a vehicle visits the complete route. Remind that unvisited nodes are visited by a dummy vehicle with no collecting profit. Then, (59) and (60) represent a trick to convert an open route starting in a node from H^S and finishing in a node from H^F into a close one where the predecessor of the initial node is the last one and the successor of the final node is the initial one. In fact, (59) fixes the predecessor of each node in H^S while (60) fixes the successors of nodes in H^F. Then, (61) establish that a node cannot be preceded by itself. (62) guarantee that a route is a sequence of nodes where if node i precedes j then j succeeds i. In (63) ensure that each node precedes exactly one node. In (64) to (66) refer to autonomy of vehicles. In (64) initialize the travel time to zero for all initial nodes. Also, (65) indicate that the travel time until a node j is equal to the travel time at its predecessor plus the distance between them. The total travel distance is limited by L_{max} in (66). Finally, (67) to (70) define the domain of the decision variables.

4 Matheuristic approach

The matheuristic approach is based on a LNS (Large Neighborhood Search) structure in which every iteration consists on two basic procedures: destroy and rebuilt. In addition, a post-optimization phase based on a set partitioning model selects the best routes founded by different LNS iterations.

In this project several strategies have been implemented in order to destroy and rebuilt solutions. In the sequel those strategies, as well as the main matheuristic components, are described.

4.1 Initial solution

In order to get an initial solution, a heuristic algorithm is implemented. It is used a randomized constructive method similar to the constructive phase of GRASP (Greedy Randomized Adaptive Search Procedure) [Feo and Resende, 1989].

The randomized constructive methods are based on constructive methods, but instead of using the node which fits the best for a specific criteria, it creates a list, called Candidate Restricted List (RCL), with a set of nodes that can produce a feasible solution. The nodes belonging to the RCL are chosen based on a constructive criteria, for instance the ratio profit over distance required to visit it. Thus the RCL contains the α% nodes with greater ratio. RCl can be defined as follows:

$$RCL = \{e \in E \mid c(e) \geq \alpha(c_{max} - c_{min}) + c_{min}\} \tag{71}$$

where c_{max} y c_{min} represent the maximum ratio and the minimum ratio of the eligible nodes, and α is a parameter between 0 and 1 that allows to balance between greedy selection and a complete randomly one. In order to create a solution, on every step a randomly chosen node from RCL is selected to be added to the solution. The procedure finishes where any node can be added without violating the threshold constraint or when all nodes have been added. Note that if $\alpha = 1$ the construction performs as a greedy constructive method and if $\alpha = 0$ it performs as a random search.

After initial solution the matheuristic performs an iterative process in which every iteration consists of a destroy and a rebuilt phase. These phases are described in the following subsections.

4.2 Destroy phase

This phase starts by removing a subset R of randomly chosen nodes from a route k of a solution S. In the sequel, the route k of a solution S is abbreviated as S_k in order to simplify the description. The resulting route M is then composed by a set of nodes which are visited in the same order than they are in the route S_k. Four destroy procedures are designed which differentiate by the way to select the set R. From a given route k, each procedure removes 30% of the visited nodes.

The different ways to remove nodes from a route are as follows:

- *Destroy 1*: In this procedure the removed nodes are chosen randomly.
- *Destroy 2*: This procedure removes the set of nodes with the least profit and, in case of ties, it removes those which save more time.
- *Destroy 3*: This procedure removes sequences of 3 consecutive nodes from the route S_k.
- *Destroy 4*: This procedure starts by inserting one node randomly chosen in the route, and then removes as less nodes as possible to have a feasible route.

4.3 Rebuilt phase

The rebuilt phase considers a subset N of unvisited nodes that can be added to the resulting route M from previous phase. The subset of new nodes N is fulfilled with every node available that, being introduced in at least one part of the route, creates a feasible solution. If the subset N is empty, the probability of being removed grows by 10% and the process is repeated until it gets to 100% or at least one node can be placed in N.

From both sets R and N only seven nodes can be consider, otherwise the extra nodes are eliminated. From R, the seven nodes are chosen randomly and from N are chosen the ones which are closer to any node in S_k.

Moreover, the rebuilt route is the solution of a mathematical model that seeks for the optimal way to integrate the nodes in $R \cup N$ into the route M. Two mathematical models are formulated which are based on Model 4 (MILP_3) described on Section 3.4. Remind that the binary decision variable w_{ij} indicates whether node i is visited before node j in a route (not just before) ($w_{ij} = 1$), or not ($w_{ij} = 0$). Nevertheless, here some constraints are modified because of the features of the new subproblem, for instance, now there are only one route, just a subset of nodes are considered to be included, and the nodes in M must be visited in the same order. The first resulting mathematical model is the following:

$$\max z = \sum_{i \in R \cup N} p_i \cdot y_i \tag{72}$$

$$w_{ij} = 1 \qquad \forall\, i, j \in M \tag{73}$$

$$\sum_{i \in M \cup R \cup N} w_{ij} \geq y_j \qquad \forall\, j \in R \cup N \tag{74}$$

$$\sum_{j \in M \cup R \cup N} w_{ij} \geq y_i \qquad \forall\, i \in R \cup N \tag{75}$$

$$w_{ij} + w_{ji} \leq y_j \qquad \forall\, i \in M, j \in R \cup N \tag{76}$$

$$w_{ij} + w_{ji} \geq y_i + y_j - 1 \qquad \forall\, i \in R \cup N, j \in M \cup R \cup N \tag{77}$$

$$L_j \geq L_i + d_{ij} - L_{max} \cdot (1 - w_{ij}) \qquad \forall\, i, j \in M \cup R \cup N \tag{78}$$

$$L_j \geq L_i + d_{ij} - L_{max} \cdot w_{ji} \qquad \forall\, i, j \in M \cup R \cup N \tag{79}$$

$$L_i + d_{i,n-1} \leq L_{max} \qquad \forall\, i \in M \cup R \cup N \tag{80}$$

$$w_{ij} \in \{0,1\} \qquad \forall\, i,j \in M \cup R \cup N \tag{81}$$
$$y_j \in \{0,1\} \qquad \forall\, j \in R \cup N \tag{82}$$
$$L_j \geq 0 \qquad \forall\, j \in M \cup R \cup N \tag{83}$$

The objective function (72) is maximize the total collected profit. Therefore, (73) force to maintain the nodes of sequence M in the same relative order in the final route from (74) to (77) ensure that if a node j is selected to be added to the route at least for one $i \in M \cup R \cup N$ w_{ij} has to be equal to 1 as well as w_{ji} to connect all nodes in the route. The others constraints are the same as (34) to (39) described previously in the specific case with $k = 1$.

The second mathematical formulation is in fact the same program, but it does not consider the set R in its equations. For instance the set R becomes empty. The purpose of this change is to expand the feasible area in order to force the algorithm to get new solutions.

4.4 Post-optimization process

Regarding the post-optimization process, at each iteration the algorithm collects the m routes with their profit and visited nodes. Thus, a set Ω of routes is stored. Each route $k \in \Omega$ has associated a total collected profit P_k and a parameter γ_{ki} which indicates if the node $i \in V'$ is visited ($\gamma_{ki} = 1$) or not ($\gamma_{ki} = 0$) by that route k.

The post-optimization process selects a set of m routes from Ω which reach the maximum total collected profit and visit every node at most once. That selection is given by solving the following mathematical model, where the decision variable χ_k is equal to 1 if the route $k \in \Omega$ is selected and 0 otherwise.

$$\max z = \sum_{k \in \Omega} P_k \cdot \chi_k \tag{84}$$
$$\sum_{k \in \Omega} \gamma_{ki} \cdot \chi_k \leq 1 \qquad \forall\, i \in V' \tag{85}$$
$$\sum_{k \in \Omega} \chi_k = m \tag{86}$$
$$\chi_k \in \{0,1\} \qquad \forall\, k \in \Omega \tag{87}$$

The objective function (84) is to maximize the profit selecting the best routes. Firstly, (85) checks that a node belongs to only one of the selected routes and (86) specifies the number of routes to select. Finally, (87) defines the domain of the binary decision variable χ_k.

5 Computational experiments

In this section the performance of the solution methods described before are evaluated. The data used on this section are the first six sets used in [Chao et al., 1996]. Each set has several instances in which the time limit for a vehicle to do a route (L_{max}) varies and they are solved with two, three and four vehicles. On Tables 1 to 6, the data type is denoted with numbers, where the former represents the set and the latter the amount of vehicles used. The results shown are the mean of the objective function values of the solutions of every instance for each set and vehicle combination. Furthermore, in all tables the best results are highlighted on bold text.

Regarding MILP models, all of them where evaluated and compared in order to decide which one is better, in other words, which model converges faster to the optimal solution. Tables 1 and 2 show the results and the CPU time respectively for four different instances randomly chosen for the MILP models designed and the results given by [Boussier et al., 2007]. In order to have solutions in an appropriated time, a time limit of 600 seconds is set for each instance. In these tables, the third number on first column represents the specific instance used, where it indicates the threshold time of each vehicle, L_{max}.

Table 1 Comparison of MILP models results for some instances

Instance	**Nodes**	**MILP$_1$**	**MILP$_2$**	**MILP$_3$**	**MILP$_4$**	**Best**[1]
1.4.8	32	**45**	**45**	**45**	**45**	**45**
2.3.5	21	**120**	**120**	**120**	**120**	**120**
4.2.1	100	81	43	**206**	**206**	**206**
6.2.4	64	132	114	**192**	**192**	**192**

Table 2 Comparison of MILP models computational time in seconds for some instances

Instance	**Nodes**	**MILP$_1$**	**MILP$_2$**	**MILP$_3$**	**MILP$_4$**	**Best**[1]
1.4.8	32	79.4	233.7	0.7	**0.5**	**0.0**
2.3.5	21	7.6	4.6	**0.5**	1.1	**0.0**
4.2.1	100	600.0	600.0	**28.6**	299.2	**0.0**
6.2.4	64	600.0	600.0	309.0	**15.9**	**0.0**

[1] Taken from [Boussier et al., 2007].

Given the results in Table 1, it is hard to tell which of MILP$_3$ and MILP$_4$ works better. In order to have clearer results, these models were tested with the complete set 1 with 3 different number of vehicles. The time limit is also set to 600 seconds for every instance and the results are presented in Table 3 and 4. Particularly for Table 4, as the third number increases on the instance reference, L_{max} increases as well.

Table 3 Comparison of best MILP for an specific data set

Data	**MILP$_3$**	**MILP$_4$**
1.2	**129.4**	106.9
1.3	**109.2**	99.2
1.4	**84.2**	81.1
Time (s)	**11583**	13982

As Tables 3 and 4 show, MILP$_3$ clearly outperforms MILP$_4$. In addition, Table 5 exposes the results for the matheuristic method. These results are found varying some parameters such as the number of iterations of the LNS, the number of initial solution before LNS algorithm and the post-optimization procedure. Particularly, the methodology to generate the initial solution uses $\alpha = 0.1$ on the randomized constructive algorithm. For the columns Mh$_1$ and Mh$_2$, in Table 5 the number of iterations of LNS and post-optimization procedure are settled in 10 and 30 respectively, while the number of iterations for the initial solution is fixed in 1 for Mh$_1$ and in 100 for Mh$_2$. The idea behind this selection is to evaluate the impact of the number of initial solutions in the final results. Regarding the last two columns, Mh$_3$ and Mh$_4$, the same number of iterations are set for the three parameters, 20 for Mh$_3$ and 10 for Mh$_4$. In addition, in the four cases a 1800 seconds time limit is used for each instance, i.e., if solving one instance the model takes more than 1800 seconds, it continues to the next instance regardless the number of iteration selected as well as if in 1 iteration the solution is not improved.

However, for Mh_3, it continues to the next instance when after five straights iterations there is not improvement in the solution.

Table 4 Comparison of best MILP for each instance of an specific data set

Instance	**MILP$_4$**	**MILP$_3$**	**Instance**	**MILP$_4$**	**MILP$_3$**	**Instance**	**MILP$_4$**	**MILP$_3$**
1.2.1	**0**	**0**	1.3.1	**0**	**0**	1.4.1	**0**	**0**
1.2.2	**15**	**15**	1.3.2	**0**	**0**	1.4.2	**0**	**0**
1.2.3	**20**	**20**	1.3.3	0	**15**	1.4.3	**0**	**0**
1.2.4	**30**	**30**	1.3.4	0	**15**	1.4.4	0	**15**
1.2.5	**45**	**45**	1.3.5	**30**	**30**	1.4.5	0	**15**
1.2.6	**80**	**80**	1.3.6	**40**	**40**	1.4.6	**25**	**25**
1.2.7	85	**90**	1.3.7	**50**	**50**	1.4.7	**35**	**35**
1.2.8	100	**110**	1.3.8	**70**	**70**	1.4.8	**45**	**45**
1.2.9	110	**135**	1.3.9	**105**	**105**	1.4.9	**60**	**60**
1.2.10	110	**140**	1.3.10	**115**	**115**	1.4.10	**75**	**75**
1.2.11	140	**155**	1.3.11	130	**135**	1.4.11	**100**	**100**
1.2.12	155	**180**	1.3.12	150	**155**	1.4.12	**120**	**120**
1.2.13	140	**210**	1.3.13	165	**175**	1.4.13	**130**	**130**
1.2.14	180	**190**	1.3.14	175	**180**	1.4.14	**155**	**155**
1.2.15	170	**235**	1.3.15	155	**200**	1.4.15	**165**	**165**
1.2.16	175	**215**	1.3.16	180	**220**	1.4.16	**175**	**175**
1.2.17	195	**240**	1.3.17	205	**220**	1.4.17	180	**190**
1.2.18	175	**240**	1.3.18	215	**240**	1.4.18	195	**210**

Table 5 Parameters comparison for the matheuristic

Data	**Nodes**	**Problems**	**Matheuristic (Mh)**			
			Mh$_1$	**Mh$_2$**	**Mh$_3$**	**Mh$_4$**
1.2	32	18	135.3	138.1	**139.2**	133.9
2.2	21	11	187.3	184.5	**187.7**	182.7
3.2	33	20	473.0	**480.0**	478.5	462.5
4.2	100	20	766.9	776.0	**791.4**	728.8
5.2	66	26	755.8	776.5	**797.5**	734.8
6.2	64	14	593.6	599.6	**604.7**	560.6
1.3	32	18	107.8	107.8	**109.4**	104.4
2.3	21	11	134.5	133.6	**135.0**	133.6
3.3	33	20	395.5	397.5	**401.0**	388.5
4.3	100	20	675.8	698.9	**700.7**	665.4
5.3	66	26	676.7	698.8	**702.1**	668.5
6.3	64	14	417.9	**435.0**	419.6	406.7
1.4	32	18	80.6	80.6	**83.3**	77.8
2.4	21	11	**92.7**	**92.7**	**92.7**	**92.7**
3.4	33	20	325.5	326.5	**329.5**	322.5
4.4	100	20	565.2	**583.4**	582.9	545.5
5.4	66	26	602.7	**619.6**	611.9	588.7
6.4	64	14	231.9	**247.3**	243.9	231.0
		Time (s)	12631.0	17978.0	23420.0	**4608.0**

Finally, in Table 6 are exposed the results of the MILP$_3$, the CP, the best results of the matheuristic model shown in Table 5 and the results on [Kim et al., 2013], which are the best results found in the literature. For the MILP$_3$ and the CP model, the CPU time limit is set at 120 seconds for a single instance.

Table 6 Comparing final results

Data	Nodes	Problems	$MILP_3$	CP	Mh	Best[2]
1.2	32	18	116.7	75.8	**139.2**	**149.1**
2.2	21	11	**190.5**	166.4	187.7	**190.5**
3.2	33	20	430.5	276.5	**480.0**	**496.0**
4.2	100	20	98.8	31.2	**791.4**	**917.1**
5.2	66	26	272.5	249.2	**797.5**	**897.8**
6.2	64	14	307.7	169.3	**604.7**	**819.3**
1.3	32	18	100.3	68.1	**109.4**	**125.0**
2.3	21	11	**136.4**	131.4	135.0	**136.4**
3.3	33	20	364.0	266.5	**401.0**	**411.5**
4.3	100	20	172.4	1.9	**700.7**	**856.2**
5.3	66	26	272.9	216.9	**702.1**	**783.6**
6.3	64	14	306.0	86.6	**435.0**	**454.4**
1.4	32	18	81.1	57.5	**83.3**	**101.0**
2.4	21	11	**94.5**	92.7	92.7	**94.5**
3.4	33	20	323.5	251.0	**329.5**	**336.5**
4.4	100	20	191.5	0.0	**583.4**	**804.1**
5.4	66	26	278.7	171.3	**619.6**	**708.8**
6.4	64	14	199.7	15.3	**247.3**	**255.0**
		Time (s)	24022.7	17755.6	23420.0	**5204.2**

[2] Taken from [Kim et al., 2013].

6 Concluding remarks

This research has shown that the MILP models based on the w_{ij} decision variables and the concept of replenishment arcs, described on Section 3.4 and 3.5 respectively are much faster than the models based on flow models, as it can be seen in Table 2. Therefore, the first two models do not reach the optimal solution for the two last instances, as Table 1 shows, due to its time limit settled at 600 seconds. Although the $MILP_3$ and $MILP_4$ accomplish the best results of the literature, theirs computational time is much higher than the one in [Boussier et al., 2007]. As it is mentioned in Table 2, it is not possible to choose which model is the best, because solving problem 4.2.1 the $MILP_3$ is much faster than $MILP_4$, but solving 6.2.4 is the opposite almost in the same proportions. As a result, it is necessary to do another experiment, so that both models were executed with the complete data set 1 and its results are reflected in Table 3 and 4, where is easily to assure that the $MILP_3$ works better in both computational time and the objective function's value. In fact, there is no instance where $MILP_4$ defeat $MILP_3$ results. Hence, this model is used on the matheuristic approach to solve the resulting subproblems.

Regarding the matheuristic algorithm, its results are illustrated in Table 5. On the first hand, respecting Mh_1 and Mh_2, it is clear that the impact of the initial solution is huge, because in 13 of the 18 cases Mh_2 outperforms Mh_1 on the total profit. Nonetheless, the Mh_2 computational time surpasses by about 5000 seconds the Mh_1 results. On the other hand, concerning Mh_3 and Mh_4, it is evident that the results are improved when the iterations are increased, but it takes much longer to solve it. Furthermore, having a better initial solution impacts the most on the final results, when the number of vehicles is four, since the destroy and rebuilt procedures are designed to improve one randomly route at a time.

Moreover, in Table 6 are presented the best results for each programming paradigm and ones found on the literature [Kim et al., 2013]. It is important to stick out that the algorithms designed are based on mathematical models, so it is pointless to compare the computational time with the heuristic approach propose in [Kim et al., 2013]. In regard of the CP model, it is evident that the model on Section 3.6 do not work as it was expected. However, it cannot be assure that CP paradigm is useless on solving TOP, as it could be different schemes to model and program it. Additionally, the $MILP_3$ performs the best when the number of nodes is small. In fact, this model

accomplish the best results for each number of vehicles solving set 2. Regarding the matheuristic results, they are evidently, in most of the cases, quite better than MILP and CP results specially when the number of total nodes are greater than 21, as it was expected, since the TOP is a NP-hard problem [Tang and Miller-Hooks, 2005]. As the algorithm is based on mathematical models, the results for the largest instances, 66 and 100 nodes, have a vast difference comparing to the best ones.

As future research directions the results found in this research can be improved by defining new subproblems and methods to solve them. For instance, the method split for vehicle routing problems [Prins, 2004] can be used to solve the TOP as shown by [Bouly et al., 2010]. New subproblems can be defined in order to allow the interchange of nodes between routes, and the subproblems can by solved by dynamic programming approaches.

Regarding the MILP strategies, a different model can be formulated based on set partitioning models as the post-optimization procedure. That kind of models can be solved by Branch & Price algorithm or the approach proposed by [Azi et al., 2007] and [Rivera et al., 2016] in which all feasible routes are first generated before the problem solution.

Finally, the proposed matheuristic can be improved by adding the use of some memory structures and precomputations in order to speed up the solution procedure. When mathematical model solves the subproblems, most of the nodes considered remain the same, so partial solutions from an iteration would be able to be used in the subsequent ones.

Abbreviations

- ACO: Ant Colony Optimization.
- CP: Constraint Programming.
- GRASP: Greedy Randomized Adaptive Search Procedures.
- LNS: Large Neighborhood Search.
- Mh: Matheuristic algorithm.
- MILP: Mixed Integer Linear Programming.
- OP: Orienteering Problem.
- PSOiA: Particle Swarm Optimization inspired Algorithm.
- PSOMA: Particle Swarm Optimization on Memetic Algorithm.
- RCL: Restricted Candidate List.
- TOP: Team Orienteering Problem.
- TS: Tabu Search.
- VNS: Variable Neighborhood Search.

References

[Archetti et al., 2007] Archetti, C., Hertz, A., and Speranza, M. G. (2007). Metaheuristics for the team orienteering problem. *Journal of Heuristics*, 13(1):49–76.

[Azi et al., 2007] Azi, N., Gendreau, M., and Potvin, J.-Y. (2007). An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *European Journal of Operational Research*, 178(3):755–766.

[Bouly et al., 2010] Bouly, H., Dang, D.-C., and Moukrim, A. (2010). A memetic algorithm for the team orienteering problem. *4OR*, 8(1):49–70.

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

[Butt and Cavalier, 1994] Butt, S. E. and Cavalier, T. M. (1994). A heuristic for the multiple tour maximum collection problem. *Computers & Operations Research*, 21(1):101–111.

[Butt and Ryan, 1999] Butt, S. E. and Ryan, D. M. (1999). An optimal solution procedure for the multiple tour maximum collection problem using column generation. *Computers & Operations Research*, 26(4):427–441.

[Chao et al., 1996] Chao, I.-M., Golden, B. L., and Wasil, E. A. (1996). The team orienteering problem. *European Journal of Operational Research*, 88(3):464–474.

[Dang et al., 2011] Dang, D.-C., Guibadj, R. N., and Moukrim, A. (2011). A PSO-based memetic algorithm for the team orienteering problem. In *Applications of Evolutionary Computation*, pages 471–480. Springer.

[Dang et al., 2013] Dang, D.-C., Guibadj, R. N., and Moukrim, A. (2013). An effective PSO-inspired algorithm for the team orienteering problem. *European Journal of Operational Research*, 229(2):332–344.

[Feo and Resende, 1989] Feo, T. A. and Resende, M. G. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8(2):67–71.

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

[Kilby and Shaw, 2006] Kilby, P. and Shaw, P. (2006). *Handbook of Constraint Programming: Vehicle Routing*, chapter Vehicle routing, pages 799–827. Elsevier, New York, NY, USA.

[Kim et al., 2013] Kim, B.-I., Li, H., and Johnson, A. L. (2013). An augmented large neighborhood search method for solving the team orienteering problem. *Expert Systems with Applications*, 40(8):3065–3072.

[Mak and Boland, 2000] Mak, V. and Boland, N. (2000). Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs. *International Transactions in Operational Research*, 7(4-5):431 – 447.

[Prins, 2004] Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

[Rivera, 2014] Rivera, J. C. (2014). *Logistic Optimization in Disaster Response Operations*. PhD thesis, Université de Technologie de Troyes.

[Rivera et al., 2015] Rivera, J. C., Afsar, H. M., and Prins, C. (2015). A multistart iterated local search for the multitrip cumulative capacitated vehicle routing problem. *Computational Optimization and Applications*, 61(1):159–187.

[Rivera et al., 2016] Rivera, J. C., Afsar, H. M., and Prins, C. (2016). Mathematical formulations and exact algorithm for the multitrip cumulative capacitated single-vehicle routing problem. *European Journal of Operational Research*, 249(1):93–104.

[Shaw, 1997] Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, University of Strathclyde.

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

[Vansteenwegen et al., 2009] Vansteenwegen, P., Souffriau, W., Berghe, G. V., and Van Oudheusden, D. (2009). Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research*, 36(12):3281–3290.