Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 347–350
DOI: 10.15439/2022F158
ISSN 2300-5963 ACSIS, Vol. 30

Team Orienteering Problem with Time Windows and Variable Profit

Eliseo Marzal, Laura Sebastia
Valencian Research Institute for Artificial Intelligence
Universitat Politècnica de València
Camino de Vera s/n - Valencia (Spain)
Email: {emarzal, lsebastia}@dsic.upv.es

***Abstract*—This paper introduces the Team Orienteering Problem with Time Windows and Variable Profits (TOPWPVP), which is a variant of the classical Orienteering Problem (OP) and an NP-hard optimization problem. It consists in determining the optimal route for a vehicle to traverse to deliver to a given set of nodes (customers), where each node has a predefined time window in which the service must start (in case this node is visited), and the vehicle may spend an amount of time given by a predefined interval so that the profit collected at this node depends on the time spent. We first propose a mathematical model for the TOPTWVP, then propose an algorithm based on Iterated Local Search to solve modified benchmark instances. The results show that our approach can solve difficult instances with good quality.**

I. Introduction

THE Orienteering Problem (OP) [1] is a combinatorial optimization and integer programming problem whose goal is to obtain the optimal route for a vehicle to traverse to deliver to a given set of customers. The objective is to maximize the total score collected from visited (selected) nodes. The Team OP (TOP) [2] is one of the most studied variants of the OP, where the route for several vehicles must be computed. When a time window is established for each node so that the service at a particular node has to start within a predefined time window, a new variant, called (Team) Orienteering Problem with Time Windows ((T)OPTW) [3], is defined. Another variant is the orienteering problem with variable profit (OPVP) [4] and its generalization, the team orienteering problem with variable profit (TOPVP) [5]. In this case, the profit collected at each node depends on the number of times a node is visited or on the continuous amount of time spent at a given node.

In this paper, we combine the TOPTW and the TOPVP to come up with the **Team Orienteering Problem with Time Windows and Variable Profit** (TOPTWVP). Specifically, each node has a predefined time window in which the service must start (in case this node is visited) and the vehicle may spend an amount of time given by a predefined interval so that the profit collected at this node depends on the time spent.

This extension of the TOPTW and the TOPVP is especially interesting to applications of the vehicle routing problems such as the routing of a reconnaissance vehicle [6], which has the option of staying longer at a location to gather more information and the route planning may depend on time windows when it is safer to perform this reconnaissance task; or the the Tourist Trip Design Problems [7], where a longer stay at a location may provide a higher satisfaction (profit) to the visitor and the visits must be scheduled taking into account the opening times of attractions. For example, the work in [8], [9] and [10] use a modified version of the TOPTWVP which allow the tourist to define some travel-style preferences, such as if they prefer to visit a few or many places or if they prefer to enjoy some free time during the trip. In order to consider these travel-style preferences, each visit is assigned a duration interval so that each visit finally takes the most appropriate time to fit into the tourist preferences.

In general, TOPTWVP tries to model a more realistic situation, where locations can only be visited within specific time windows and where staying longer (also within a given interval) may report a higher profit.

The main contributions of this paper are the following:

- We introduce a mathematical model for the TOPTWVP.
- Due to the limitation of solving large instances, we then propose a heuristic based on Iterated Local Search (ILS).
- We perform experiments with the ILS algorithm on instances for the Solomon benchmark.
- We compare the solutions with respect to several Key Performance Indicators (KPIs): score, number of visits, waiting time (idle time in the route), and execution time.

The remainder of this paper is as follows. The problem description and the mathematical model is detailed in Section II. Section III describes the proposed algorithm based on ILS. Section IV reports numerical experiments performed on benchmark instances. Finally, in Section V, we summarize the main achievements and future works.

II. TOP with Time Windows and Variable Profit

The input values in the **TOPTWVP** are the following:

- a set of nodes $N = \{1, \ldots, |N|\}$, where nodes 1 and $|N|$ represent the start and end nodes
- a number of routes M and total time budget per route T_{max}
- For each node $i \in N$:

This work has been partially supported by the Spanish MINECO project TIN2017-88476-C2-1-R, the AVI project SmartTur+ECO INNEST/2021/233 and the EU project TAILOR 952215.

- a travel time t_{ij} from node i to j, known for all vertices
- a non-negative profit P_i, i.e. the maximum profit collected when visiting a node; P_1 and $P_{|N|}$ are 0
- a time window $[O_i, C_i]$, which indicates that a visit to a node can only start during this time window
- a service time interval $[minD_i, maxD_i]$, i.e. the minimum and maximum amount of time that can be spent at a node

The goal of the TOPTWVP is to determine M routes, each limited by T_{max}, that visit a subset of N within the respective time windows, during a service time in the interval duration and that maximize the total collected profit. The TOPTWVP can be formulated as an integer programming model with the following decision variables:

- $x_{ijm} = 1$, if in route m, a visit to node i is followed by a visit to node j; 0, otherwise
- y_{im}=1, if node i is visited in route m; 0, otherwise.
- s_{im} is the start of the service at node i in route m
- d_{im} is the service time (duration) at node i in route m

And the following constraints:

$$\sum_{m=1}^{M} \sum_{j=2}^{|N|} x_{1jm} = \sum_{m=1}^{M} \sum_{i=1}^{|N|-1} x_{i|N|m} = M \tag{1}$$

$$\sum_{m=1}^{M} y_{km} \leq 1; \forall k = 2, \ldots, (|N| - 1) \tag{2}$$

$$\sum_{i=1}^{|N|-1} x_{ikm} = \sum_{j=2}^{|N|} x_{kjm} = y_{km}; \quad \forall k = 2, \ldots, (|N| - 1); \forall m = 1, \ldots, M \tag{3}$$

$$O_i \leq s_{im} \leq C_i; \forall i = 1, \ldots, |N|; \forall m = 1, \ldots, M \tag{4}$$

$$s_{im} + d_{im} + t_{ij} - s_{jm} \leq L(1 - x_{ijm}); \quad \forall i, j = 1, \ldots, |N|; \forall m = 1, \ldots, M \tag{5}$$

$$minD_i \leq d_{im} \leq maxD_i; \forall m = 1, \ldots, M \tag{6}$$

Constraints 1 establish that each route starts from node 1 and ends in $|N|$. Constraints 2 ensure that each node can only be visited at most once in all routes. Constraints 3 ensure the connectivity of each route. Constraints 4 force that the service starts within the time window for each route. Constraints 5 ensure the timeline of the route (L is a large constant, that can be equal to T_{max}), explicitly considering the variable duration of the service. Constraint 6 guarantees that the service time is within the corresponding interval of duration.

$$Maximize \sum_{m=1}^{M} \sum_{i=2}^{|N|-1} P_i d_{im} y_{im} \tag{7}$$

The maximization function 7 establishes that the profit of the final plan not only depends on whether a node is visited or not but also it depends on how long a node is visited. Without loss of generality, in this definition of the TOPTWVP, we assume that each time unit spent at the node collects P_i. Therefore, the total score of a visited node i results from the product of the profit P_i and the time spent at a node d_{im} (in a route m).

Since the TOPTW and TOPVP are NP-hard, the TOPTWVP is also NP-hard. Both dealing with time windows and with an interval of possible service times makes this problem harder to solve. We have performed some experiments (not shown in this paper due to space restrictions) that prove that our implementation of this mathematical model is able to solve only small instances. Moreover, as some works referred to in this paper also state, in general, variations of TOP problems with a significant number of nodes are solved with heuristic approaches, and exact algorithms are only feasible for problems with a small number of nodes. For this reason, the following section introduces a heuristic approach to solve the TOPTWVP.

III. Heuristic approach to TOPTWVP

The more successful heuristic approaches for both the TOPTW and TOPVP are based on the ILS algorithm ([11],[12],[5]). Specifically, in this work, we take the ILS implementation for solving the TOPTW given in [11] and [12] and adapt some steps in order to consider the variable profit.

The general scheme of the ILS is shown in Algorithm 1. In a nutshell, this algorithm constructs an initial feasible solution (Construction), which is further improved by Iterated Local Search (ILS). ILS comprises the components LocalSearch, Perturbation and AcceptanceCriterion. We refer the reader to [11] and [12] for further details in the ILS implementation. In this section, we only focus on the modifications introduced to solve the specific TOPTWVP.

Apart from the input data and decision variables detailed in the previous section, we also keep the following data for each node, to reduce the time devoted to checking the feasibility of a route, thus making the algorithm more efficient [11]:

- $arrive_{im}$: arriving time to a node. If $arrive_{im} \in [O_i, C_i]$, then the arriving time and the start time s_{im} take the same value.
- $wait_{im}$: waiting time to visit a node because the arriving time $arrive_{im}$ is previous to the opening time O_i.
- $MaxShift_{im}$: maximum time that a visit can be delayed so that the route is still feasible, i.e. it keeps track of how much a certain visit can be shifted in time, without violating any time window in the route.

Regarding the Construction process, the first solution only contains the start node or depot, as initial and final nodes. Then, a set F with all feasible candidate nodes that can be visited is generated. All possibilities of inserting an unscheduled node in position p of route m are examined. To check the feasibility of inserting node j between nodes i and k, we

Algorithm 1 ILS(N,M)

1: $S_0 \leftarrow \mathsf{Construction}(N, M)$
2: $S_0 \leftarrow \mathsf{LocalSearch}(S_0, N^*, N', M)$
3: $S^* \leftarrow S_0$
4: NoImprovement $\leftarrow 0$
5: **while** TimeLimit has not been reached **do**
6: $S_0 \leftarrow \mathsf{Perturbation}(S_0, N^*, N', M)$
7: $S_0 \leftarrow \mathsf{LocalSearch}(S_0, N^*, N', M)$
8: **if** S_0 better than S^* **then**
9: $S^* \leftarrow S_0$
10: NoImprovement $\leftarrow 0$
11: **else**
12: NoImprovement $\leftarrow$ NoImprovement $+ 1$
13: **end if**
14: **if** (NoImprovement $+ 1$) MOD ThresholdImpr $= 0$ **then**
15: $S_0 \leftarrow S^*$
16: **end if**
17: **end while**
18: return S^*

Algorithm 2 IncreaseServiceTime(S_0,M)

1: **for** each route $m \in M$ **do**
2: **while** $\exists i : MaxShift_{im} > 0$ **do**
3: $G = \{i : MaxShift_{im} > 0\}$
4: **for** $i \in G$ **do**
5: $\Delta_{im} \leftarrow min(MaxShift_{im}, maxD_i - d_{im})$
6: **end for**
7: $n^* = argmax_{\forall i \in G}(P_i * \Delta_{im})$
8: $d_{n^*m} \leftarrow d_{n^*m} + \Delta_{n^*m}$
9: **for** each node j in route m: $s_{jm} > s_{n^*m}$ **do**
10: $\mathsf{Update}(arrive_{jm}, s_{jm}, Shift_{jm}, MaxShift_{jm})$
11: **end for**
12: **for** each node k in route m: $s_{km} <= s_{n^*m}$ **do**
13: $\mathsf{Update}(MaxShift_{km})$
14: **end for**
15: **end while**
16: **end for**
17: return S_0

compute $Shift_{jm}$, which is time added or reduced when a new node is inserted or removed in a route:

$$Shift_{jm} = t_{ij} + wait_{jm} + d_{jm} + t_{jk} - t_{ik}$$

where: $wait_{jm} = max(0, O_j - arrive_{jm})$ and $arrive_{jm} = s_{im} + d_{im} + t_{ij}$. Therefore, it is feasible to insert a node j between nodes i and k in route m if $Shift_{jm} <= wait_{km} + MaxShift_{km}$. In the TOPTWVP, the service time d_{jm} is not a value known a priori. Therefore, to perform the calculations above, it is necessary to compute an estimate of the service time, which is based on using all the available time without causing infeasibility in the route:

$$d_{jm} = max(minD_j, min(maxD_j, MS')), where:$$

$$MS' = wait_{km} + MaxShift_{km} - t_{ij} - wait_{jm} - t_{jk} + t_{ik}$$

The set F contains all the feasible candidate nodes; one of these nodes i is selected according to the attractiveness of the insertion, which is computed as $P_i^2/Shift_i$. Once a node i is selected, it is inserted in the corresponding route and position, and S_0, N', and N^* are updated accordingly. The service time d_{im} for the selected node is set to the estimate computed above. Consequently, the value of $arrive_{jm}$, $wait_{jm}$, s_{jm}, $Shift_{jm}$ and $MaxShift_{jm}$ of the subsequent nodes is also updated. Additionally, $MaxShift_{jm}$ of the previous nodes is also recomputed. The construction process of the first solution is terminated when $F = \emptyset$, that is, when no more feasible candidate nodes are found. This solution will be further improved by the ILS.

Our LocalSearch procedure is composed by the same six local search moves described in [12], and we add a new move that modifies the service time of some nodes. These seven local search moves are applied in three stages. The first stage contains the moves `Swap1`, `Swap2`, `2-Opt` and `Move` which restructure the current solution trying to decrease the travel total cost, thus increasing the unused time budget.

The second stage, named `IncreaseServiceTime`, consists of a new move aiming to enlarge some nodes' service time, thus increasing the total score. Specifically, Algorithm 2 is applied. Once the first stage has been executed, and at least one move has been applied, a restructured solution with an increased unused time budget is obtained; that is, some idle time can be used to increase the service time of specific nodes. Therefore, our aim at this moment is to find a node whose service time can be enlarged and, consequently, the solution profit improves. Algorithm 2 describes this process. For each route, we build the set G with the nodes whose service time can be increased (line 3) because they have a positive MaxShift. Then, this increment in the service time is computed for each node in G (lines 4-6), and the node n^* with the highest increment in the score is selected (line 7). The new service time d_{n^*m} is computed and, hence, the value of $arrive_{jm}$, $wait_{jm}$, s_{jm}, $Shift_{jm}$ and $MaxShift_{jm}$ of the subsequent nodes and the value of $MaxShift_{jm}$ of the previous nodes are also recomputed.

The third stage in the LocalSearch procedure contains the moves `Insert` and `Replace`, focused on increasing the profit. Finally, for both the Perturbation and AcceptanceCriterion components of ILS, in our implementation, we follow the same scheme as [12].

IV. Experiments

This section shows the experiments performed in order to validate our algorithm to solve the TOPTWVP problem for $m = 1$ and $m = 2$. A set of 56 TOPTW Solomon (set c) instances of vehicle routing problems with time windows were selected (https://unicen.smu.edu.sg/oplib-orienteering-problem-library) and adapted to the TOPTWVP by adding the interval of service time; in particular, min_D was set to the service time indicated in the original problems,

whereas max_D is set to $min_D + 30$. For comparison, we have executed a TOPTW implementation with the original Solomon instances (where the service time coincides with $minD$), denoted as Min and with these same instances where the service time is set to $maxD$, denoted as Max. Table I show the results for score, number of visited nodes, waiting time and execution time for one and two routes, when executing the two baselines Min and Max and our implementation Var; columns under AVG show the average value among the ten runs of our ILS solving the TOPTWVP for each instance and columns under MAX show the maximum value of these ten runs.

For the **score**, in the case of the AVG results, Max obtains a higher score in 21 instances for one route and 25 for two routes. On average, Var obtains a higher score than Max with one route, whereas Max gets a better score than Var with two routes (although the difference is smaller). Regarding MAX results, the solution with Var is the one that provides the highest score in many problems, except in 16 instances out of 56 for one route and 24 for two routes, where Max obtains a higher score. Min is always quite far from the other two approaches, as expected.

Regarding the **number of visits**, the Var approach (almost) always include few more visits in the routes for instances solved with one and two routes. In the case of Max, fewer nodes tend to be incorporated as they have a longer duration.

The **waiting time** is defined as the sum of the waiting time of all visits, that is, the idle time in the routes. Var obtains routes with less waiting time than Max for one route: for AVG results, Var obtains routes with less waiting time in 26 instances versus 14 instances where Max obtains routes with less waiting time. However, for two routes, the results are more similar (26 versus 24 instances). Looking into the individual solutions, we have observed that, in many cases, this worsening of the waiting time is due to an improvement in the score, because the maximization function only considers the score and not the waiting time. That is, it is prioritized to include shorter visits that provide a higher score, although it may imply having idle time in the route waiting for the start time of the time window.

The **execution time** corresponds to the time required to find the best solution, although each execution takes all the given time. It can be observed that Var always needs longer to obtain a solution since an additional dimension is being introduced by having to work with an interval for the duration of the service time. This makes the problem more complicated, and it is reflected in the time needed to find a solution.

	Score		Num. Visits		Wating Time		Exec. Time	
AVG	m=1	m=2	m=1	m=2	m=1	m=2	m=1	m=2
Min	21039,39	34748,93	24,34	47,81	6,77	9,04	485,39	615,44
Max	26523,71	46150,14	11,69	23,29	5,97	6,75	269,83	513,86
Var	26813,11	46031,54	12,63	25,44	3,60	7,43	701,07	792,21
MAX	m=1	m=2	m=1	m=2	m=1	m=2	m=1	m=2
Min	21316,07	35104,82	24,34	48,58	4,31	6,88	562,30	679,10
Max	26907,86	46726,43	11,69	23,80	4,64	5,20	339,02	547,01
Var	27330,80	46948,29	12,71	25,75	2,94	7,26	715,62	760,83

TABLE I: Summary of score, number of visits, waiting time and execution time of the ILS algorithm with Min, Max and Var for $m = 1$ and $m = 2$

V. Conclusions

This paper has introduced the Team OP with Time Windows and Variable Profits where each node has a predefined time window in which the service must start (if visited), and the vehicle may spend an amount of time given by a predefined interval so that the profit collected at this node depends on the time spent. We have proposed a mathematical model for solving the TOPTWVP, and an ILS algorithm to solve modified benchmark instances. The results show that our approach can solve difficult instances with good quality. When compared with two baselines Min and Max, our approach is, in general, obtaining better scores and also better values in other KPIs, such as the number of visits and the waiting time.

As future work, we are developing new procedures to handle the variable profit with the aim of improving the execution time. Moreover, we are working on a new variant of TOPTWVP that allows defining a custom waiting time so that it can also be considered in the optimization function.

References

[1] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011. doi: 10.1016/j.ejor.2010.03.045

[2] I.-M. Chao, B. L. Golden, and E. A. Wasil, "The team orienteering problem," *European journal of operational research*, vol. 88, no. 3, pp. 464–474, 1996. doi: 10.1016/0377-2217(94)00289-4

[3] S. Boussier, D. Feillet, and M. Gendreau, "An exact algorithm for team orienteering problems," *4or*, vol. 5, no. 3, pp. 211–230, 2007. doi: 10.1007/s10288-006-0009-1

[4] G. Erdogan and G. Laporte, "The orienteering problem with variable profits," *Networks*, vol. 61, pp. 104–116, 2013. doi: 10.1002/net.21496

[5] A. Gunawan, K. M. Ng, G. Kendall, and J. Lai, "An iterated local search algorithm for the team orienteering problem with variable profits," *Engineering Optimization*, vol. 50, no. 7, pp. 1148–1163, 2018. doi: 10.1080/0305215X.2017.1417398

[6] F. Mufalli, R. Batta, and R. Nagi, "Simultaneous sensor selection and routing of unmanned aerial vehicles for complex mission plans," *Computers & Operations Research*, vol. 39, no. 11, pp. 2787–2799, 2012. doi: 10.1016/j.cor.2012.02.010

[7] P. Vansteenwegen and D. Van Oudheusden, "The mobile tourist guide: an or opportunity," *OR insight*, vol. 20, no. 3, pp. 21–27, 2007. doi: 10.1057/ori.2007.17

[8] J. Ibañez, L. Sebastia, and E. Onaindia, "Planning tourist agendas for different travel styles," in *Proceedings of the Twenty-second European Conference on Artificial Intelligence*. IOS Press, 2016. doi: 10.3233/978-1-61499-672-9-1818 pp. 1818–1823.

[9] J. Ibánez-Ruiz, L. Sebastiá, and E. Onaindia, "Evaluating the quality of tourist agendas customized to different travel styles," *arXiv preprint arXiv:1706.05518*, 2017.

[10] L. Sebastia and E. Marzal, "Extensions of the tourist travel design problem for different travel styles," *Procedia Computer Science*, vol. 176, pp. 339–348, 2020. doi: 10.1016/j.procs.2020.08.036

[11] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. Van Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Computers & Operations Research*, vol. 36, no. 12, pp. 3281–3290, 2009. doi: 10.1016/j.cor.2009.03.008

[12] A. Gunawan, H. C. Lau, and K. Lu, "An iterated local search algorithm for solving the orienteering problem with time windows," in *Evolutionary Computation in Combinatorial Optimization: 15th European Conference, EvoCOP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings*, 2015. doi: 10.1007/978-3-319-16468-7_6 pp. 61–73.