The Multi–Constraint Team Orienteering Problem with Multiple Time Windows

Wouter Souffriau
Information Technology, KaHo St.-Lieven, Belgium

Pieter Vansteenwegen
Centre for Industrial Management, Katholieke Universiteit Leuven, Belgium

Greet Vanden Berghe
Information Technology, KaHo St.-Lieven, Belgium

Dirk Van Oudheusden
Centre for Industrial Management, Katholieke Universiteit Leuven, Belgium
Email: wouter.souffriau@kahosl.be

The Orienteering Problem (OP) is a combinatorial routing problem of which the goal is to find a tour that maximises the total score earned by visiting vertices. A set of vertices is given, determined by their coordinates and a score. The pairwise travel times between the vertices are known. The total travel time should not exceed a predetermined time budget. Each vertex can be visited at most once.

The OP serves as the starting point for modelling tourist trip design problems [2, 4]: a tourist who wants to visit a city or region is limited in time, and cannot visit every tourist attraction. Therefore, the tourist has to make a selection of the most interesting places to visit, within the limited time frame. The score represents the estimated personal interest of the tourist in the location. The time budget obviously represents the maximal amount of time the tourist has available. Solving the OP results in a personal trip for the tourist. However, while executing the planned trip, unexpected events often make the solution infeasible. Therefore, a fast algorithm is needed in order to dynamically recalculate the plan. Well known OP extensions are the Team OP (TOP), and the OP with Time Windows (TW). The former allows modelling trip planning problems for multiple days, the latter allows defining a period for each vertex in which the visit has to take place.

The scope of this paper is the Multi–Constraint TOP with Multiple TWs (MCTOPMTW), in

which each location is extended with Z attributes for each day. Z additional knapsack constraints are defined, which limit the selection of locations, together with the time constraints. In the envisioned tourist application (http://www.citytripplanner.com), the additional constraints will involve budget limitations for entrance fees and "max-n type" for each day and for the whole trip (e.g. a maximum number of museums to visit on the first day). In addition, the proposed model addresses the main drawbacks of the TOPTW model of Vansteenwegen et al. [4]. The new model allows defining different TWs on different days and more than one TW per day. The (T)OP with multiple (and different) TWs was recently discussed and tackeled by Tricoire et al. [3], without extra constraints.

This abstract defines the MCTOPMTW mathematically, proposes a fast local search based metaheuristic algorithm and presents promising experimental results.

1 Mathematical Model

The MCTOPMTW can be formulated as an integer program: given are M tours and N locations with a non-negative score S_i, Z attributes and W TWs; location 1 is the starting location, location N is the end location; the shortest path between location i and location j requires time t_{ij}, the Euclidean distance between them; $x_{ijm} = 1$ if, in tour m, a visit to location i is followed by a visit to location j, 0 otherwise; $y_{iwm} = 1$ if location i is visited during TW w in tour m, 0 otherwise; s_{im} is the start of the visit at location i in tour m; O_{iwm} and C_{iwm} are the opening and closing times of TW w of vertex i in tour m; e_{imz} is the cost associated with knapsack constraint z for location i in tour m; E_z is the cost budget of knapsack constraint z; L is a large constant. The total score of the selected visits has to be maximised.

$$Max \sum_{m=1}^{M} \sum_{w=1}^{W} \sum_{i=2}^{N-1} S_i y_{iwm} \tag{1}$$

Subject To:

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

$$\sum_{i=1}^{N-1} x_{ikm} = \sum_{j=2}^{N} x_{kjm} = \sum_{w=1}^{W} y_{kwm}; \forall k = 2, ..., N-1; \forall m = 1, ..., M \tag{3}$$

$$s_{im} + t_{ij} - s_{jm} \leq L(1 - x_{ijm}); \forall i, j = 1, ..., N; \forall m = 1, ..., M \tag{4}$$

$$\sum_{m=1}^{M} \sum_{w=1}^{W} y_{iwm} \leq 1; \forall i = 1, ..., N \tag{5}$$

$$\sum_{m=1}^{M}\sum_{w=1}^{W}\sum_{i=1}^{N} e_{imz}y_{iwm} \leq E_z; \forall z = 1, ..., Z \tag{6}$$

$$\exists w \in 1, ..., W : O_{iwm} \leq s_{im} \leq C_{iwm}; \forall i = 1, ..., N; \forall m = 1, ..., M \tag{7}$$

$$x_{ijm}, y_{iwm} \in \{0, 1\}; \forall i, j = 1, ..., N; \forall w = 1, ..., W; \forall m = 1, ..., M \tag{8}$$

The objective function (1) maximises the total collected score. Constraint (2) guarantees that all tours start in vertex 1 and end in vertex N. Constraints (3) and (4) determine the connectivity and time line of each tour. Constraints (5) guarantee that every vertex is visited at most once. Knapsack constraints (6) limit the selection by constraining attributes of the vertices, used to define budget and max-n type constraints. Constraints (7) restrict the start of the visit to one of the time windows. Note that the model also enables max-n type and budget constraints to be defined per day, e.g. visit maximum one church on the first day, or spend at most 100$ on the second day. In this case, an extra knapsack constraint is added for that particular day. Moreover, constraints (5) can also be expressed by a special case of general knapsack constraints (6).

2 Algorithm

The algorithm for tackling the MCTOPMTW is based on the Iterated Local Search (ILS) for the TOPTW of Vansteenwegen et al. [4]. In order to add diversification, the ILS approach is hybridised with GRASP (Algorithm Listing 1), as GRASP has proven to work well for the TOP [2].

```
for greed=0.89; greed>0.59; greed-=0.01 do
    while NumberOfIterationsNoImprovement < 100 do
        Solution = GRASP(greed);
        if Solution > BestFound then
            BestFound=Solution;
            NumberOfIterationsNoImprovement=0;
        else
            NumberOfTimesNoImprovement++;
        Shake();
Return BestFound;
```

Algorithm 1: Hybrid ILS–GRASP for the MCTOPMTW

The GRASP procedure iteratively adds visits to the current solution, allowing only feasible solutions. This procedure is governed by a greediness parameter, which varies from 0.89 (close to best–improving local search) to 0.60 (more randomness). In order to escape from local optima, the shake procedure removes R visits from the current solution, starting from vertex S. R and S

are dynamically updated to steer diversification. More details about this updating process can be found in [4].

As the ILS–GRASP algorithm iteratively adds visits to and removes visits from the current solution, an efficient mechanism is designed to evaluate the knapsack constraints. Efficient value propagation dynamically maintains a neighbourhood structure of possible visits for each tour. Also, the knapsack constraints' slack values serve as a basis for calculating the heuristic value of an insertion of a vertex in a tour, which is used by the GRASP procedure.

3 Experimental Results

Garcia et al. [1] designed MCTOPTW test instances based on available test sets for the TOPTW [4]. We extended the MCTOPTW instances with one money budget constraint, ten max-n type constraints and with m ranging from 1 to 4 instead of 2. The instances are designed in such a way that high quality TOPTW solutions are also feasible high quality solutions of the new MCTOPTW instances. Different time windows in different tours are not included in these instances, but that would have no influence on the performance of this algorithm. Multiple time windows are also not included, but these can be modelled by copying the location, giving each copy one time window and adding an extra knapsack constraint.

The algorithm is allowed to run 10 times on a 2.5GHz Intel Xeon processor with 4 GB of RAM. The results are very good: one average run has a score gap (with the known high quality solutions) of only 3.26%, using 1 second of computation time. When the best solution of 10 runs is used, the initial high quality solution was found in 27% of the test instances and this solution was improved in 9% of the instances. More extensive experimental results indicate that hybridising ILS with GRASP significantly increases the performance of the algorithm.

References

[1] A.Garcia, P. Vansteenwegen, W. Souffriau, M. Liniza and O. Arbelaitz, "Solving multi-constrained team orienteering problems to generate tourist routes", *Discrete Optimization*, under review.

[2] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe and D. Van Oudheusden, "A Path Relinking Approach for the Team Orienteering Problem", *Computers & O.R.*, in press.

[3] F. Tricoire, M. Romauch, K.F. Doerner, R.F. Hartl, "Heuristics for the Multi-Period Orienteering Problem with Multiple Time Windows", *Computers & O.R.* 37, 351-367 (2010).

[4] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe and D. Van Oudheusden, "Iterated Local Search for the Team Orienteering Problem with Time Windows", *Computers & O.R.* 36, 3281-3290 (2009).