# Multi-Agent Orienteering Problem with Time-Dependent Capacity Constraints

Cen Chen, Shih-Fen Cheng, Hoong Chuin Lau

Singapore Management University
80 Stamford Road, 178902, Singapore
{cenchen.2012, sfcheng, hclau}@smu.edu.sg

## 1 Introduction

The Orienteering Problem (OP), as originally defined by Tsiligirides [6], is the problem of cross-country sport in which participants get rewards from visiting a predefined set of checkpoints. As Orienteering Problem can be used to describe a wide variety of real-world problems like route planning for facility inspection, patrolling of strategic location, and reward-weighted traveling salesman problem, it has attracted continuous interests from researchers and a large number of variants and corresponding algorithms for solving them have been introduced. The most important variants of the orienteering problems include: 1) The Team Orienteering Problem (TOP), in which a group of centrally controlled agents are sent to collect rewards by visiting check points [1, 2], 2) The Orienteering Problem with Time Windows (OPTW), in which service time windows are specified for each node [3], and 3) the combination of the above two variants (The Team Orienteering Problem with Time Windows, or TOPTW) [5].

In this paper, we introduce a new variant of the OP which we believe to be first of its kind. Our proposal includes one major new features: 1) The inclusion of individual preference and time-dependent rewards. 2) Nodes are now subject to time-dependent capacity constraints. We call this new variant the Multi-agent Orienteering Problem with Time-Dependent Capacity Constraints (MOPTCC).

The application domain that motivates our creation of this new variant is the crowd control problem in the tourism industry. For tourism operators, one critical task they need to perform on a daily basis is to provide proper guidance to visitors. Such guidance may be passive, which is delivered via signboard or staff's ground instructions. The guidance can also be active, in which case individual visitors may be guided by following instructions delivered in real time to their mobile devices. As the venue operator needs to watch closely how queues build up at different attractions throughout the day, it needs to generate recommendations for individual visitors, following their respective preferences, while observing capacity constraints it sets for different attractions throughout the day.

The "multi-agent" feature of MOPTCC allows us to produce schedules based on individual visitor's preferences. On the other hand, the feature of "time-dependent node capacities" allows us to set the limit of visitors at different attractions to avoid over-crowding. Besides formally formulate the MOPTCC, we propose two propose two solution approaches: 1) a mixed integer linear program (MILP) that computes the exact solution; and 2) an effective sampled fictitious play algorithm [4] that considerably reduces the computational time. Experimentations have been done to evaluate the solution quality and effectiveness.

## 2 Mathematical Programming Formulation

In the MOPTCC, a set of $n$ nodes is given with attached utility scores $s_{ik}^t$ which differs with time and an agent's own preferences. Two fixed nodes,the start node 1 and the exit node $n$, are set during the graph construction. Each node is associated with a service time $v_j$ and time-dependent waiting times $Q^{jt}$. The traveling time from node $i$ to node $j$ is given as $t_{ij}$. $m$ Agents are sent to collect the utility rewards. They may or may not be able to visit all the nodes, as they are constrained by their own time budget $T_n^k$ which is less than the whole network expire time $T$.

The MOPTCC can be formulated as an integer programming problem with the following decision variables: $x_{ijk}^t = 1$ if agent $k$ leaves node $i$ at time $t$ and goes for node $j$; otherwise, 0. It checks whether the edge between the two nodes $i$, $j$ is to be visited at time $t$. $Q_d^t$ is the waiting time of node $d$ at time $t$.

$$\max \sum_{t=1}^{T}\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} s_{ik}^{t} x_{ijk}^{t}, \tag{1}$$

$$s.t. \sum_{t=1}^{T}\sum_{j=1}^{n} x_{1jk}^{t} = \sum_{t=1}^{T}\sum_{i=1}^{n} x_{ink}^{t} = 1, \qquad \forall k = 1,\ldots,m, \tag{2}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n} x_{idk}^{t} = \sum_{t=1}^{T}\sum_{j=1}^{n} x_{djk}^{t}, \qquad \forall d = 1,\ldots,n; \forall k = 1,\ldots,m, \tag{3}$$

$$\sum_{t=1}^{T}\sum_{j=1}^{n} x_{ijk}^{t} \leqslant 1, \qquad \forall i = 1,\ldots,n; \forall k = 1,\ldots,m, \tag{4}$$

$$Q_{d}^{t} = Q_{d}^{t-1} + \sum_{k=1}^{m}\sum_{i=1}^{n} x_{idk}^{t-t_{id}} - \sum_{k=1}^{m}\sum_{j=1}^{n} x_{djk}^{t}, \qquad \forall d = 1,\ldots,n; \forall t = 1,\ldots,T, \tag{5}$$

$$Q_{d}^{t} \leqslant Q_{d}^{Max}, \qquad \forall d = 1,\ldots,n; \forall t = 1,\ldots,T, \tag{6}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n}(t + v_d + Q_{d}^{t+t_{id}} + t_{id}) x_{idk}^{t} = \sum_{t=1}^{T}\sum_{j=1}^{n} t x_{djk}^{t}, \qquad \forall d = 2,\ldots,n-1; \forall k = 1,\ldots,m, \tag{7}$$

$$\sum_{t=1}^{T}\sum_{j=1}^{n} t \times x_{1jk}^{t} = T_{1}^{k}; \ \sum_{t=1}^{T}\sum_{j=1}^{n} t \times x_{njk}^{t} \leqslant T_{n}^{k}, \qquad \forall k = 1,\ldots,m, \tag{8}$$

$$x_{ijk}^{t} \in \{0,1\}, \qquad \forall i,j,k,t. \tag{9}$$

The objective function 1 is to maximize the total utility collected. Constraint 2 ensures that for each agent $k$, he starts at node 1 and ends at node $n$. Constraint 3 guarantees the connectivity of the path for each agent $k$. Constraint 4 ensures that for each agent $k$, each node is visited at most once. Constraint 5 defines the waiting time for each node $d$ at time $t$. We assume the service rate is 1 for all the nodes. Thus $Q_d^t$ equals the waiting time of the last time stamp $t-1$ plus the inflow and minus the outflow of current time $t$ for this node. Constraint 6 ensures the waiting time $Q_d^t$ to not exceed its corresponding threshold $Q_d^{Max}$. Constraint 7 ensures consistency of arriving and leaving time of node $d$ for each agent $k$. Constraint 8 ensures that for each agent, the schedule starts at $T_1^k$ and ends before $T_n^k$ as he specifies.

Constraint 5 can be expressed as as constraint 10 by solving the recursion. When $Q_d^t$ is substituted into the constraint 7, this constraint will become non-linear which introduces scalability issue. To address it, we represent the equivalent linear constraints 11,12,13 to transform the original formulation into a mixed integer linear programming (MILP) where $\alpha_{ijdlk}^{st}$ and $\beta_{ijdlk}^{st}$ are intermediate variables.

$$Q_{d}^{t} = Q_{d}^{1} + \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{s=2-t_{id}}^{t-t_{id}} x_{idk}^{s} - \sum_{k=1}^{m}\sum_{j=1}^{n}\sum_{s=2}^{t} x_{djk}^{s}, \qquad \forall d = 1,\ldots,n; \forall t = 1,\ldots,T, \tag{10}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n}(t + v_d + t_{id}) x_{idk}^{t} + \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{l=1}^{m}\sum_{j=1}^{n}\sum_{s=2-t_{id}}^{t} \alpha_{ijdlk}^{st} - \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{l=1}^{m}\sum_{j=1}^{n}\sum_{s=2}^{t+t_{id}} \beta_{ijdlk}^{st}$$
$$= \sum_{t=1}^{T}\sum_{j=1}^{n} t x_{djk}^{t}, \forall d = 2,\ldots,n-1; \forall k = 1,\ldots,m, \tag{11}$$

$$\alpha_{ijdlk}^{st} \leqslant x_{jdl}^{s}, \alpha_{ijdlk}^{st} \leqslant x_{idk}^{t}, \alpha_{ijdlk}^{st} \geqslant x_{jdl}^{s} + x_{idk}^{t} - 1, \qquad \forall i,d,j,k,l,s,t \tag{12}$$

$$\beta_{ijdlk}^{st} \leqslant x_{djl}^{s}, \beta_{ijdlk}^{st} \leqslant x_{idk}^{t}, \beta_{ijdlk}^{st} \geqslant x_{djl}^{s} + x_{idk}^{t} - 1, \qquad \forall i,d,j,k,l,s,t \tag{13}$$

An exact global optimal solution can be reached by solving this multi-agents MILP. However, The computational cost increases exponentially with the number of agents, nodes and the time horizon.

# 3 Solution Approach

Since high quality solutions are required and computational time should be limited to a reasonable time, a sampled fictitious play (SFP in Figure 1) is implemented to improve the solutions and each best response is computed using a mixed integer programming formulation(MILP) for single agent.

The single agent MILP can be obtained by deriving from the multi-agents model and reducing the dimensionality through fixing the rest of the agents' actions as $Q^{\text{input}}$. In this formulation queue constraints can be expressed as constraint 14. Similarly, timeline flow constraint 7 can be rewritten and linearized as constraints 15 for this single agent model.

**K** is a set of agents. **A** contains nodes and associated information. **S** specifies the utility scores for all the agents at each node and each time stamp. **T** is a set of the walking times among all the nodes. **Q** gives all the waiting times. **B** is a set of best responses for every agent given queues as input by fixing the rest of the agents decisions. **D** contains the history of sampled decisions for all the agents.

Line 2 generates initial solutions, thus populating the 0th row of history matrix **H**. Line 6 performs uniform sampling from each players history independently. If the queue constraints resulting from the sampling are violated, re-sampling is performed. Line 9 computes each agent's best reply $\mathbf{B}_i$ to all the other agents' sampled decision $\mathbf{D}_{-i}$. Line 14 appends **B** at the end of the history matrix **H**.

```
1: procedure SFP(K, A, S, T)
2:     H ← INITIALSOLUTIONS()
3:     k ← 0
4:     B_Best ← H_0
5:     while k < k_max do
6:         D ← SAMPLE(H, k)
7:         for each agent i do
8:             Q_{-i}^input ← AGGREGATEQUEUES(D_{-i})
9:             B_i ← BESTREPLY(K, A, S, T, Q_{-i}^input)
10:            if UTILITY(B_i, D_{-i}) > UTILITY(D) then
11:                B_Best ← (B_i, D_{-i})
12:            end if
13:        end for
14:        H ← UPDATEHISTORY(B)
15:        k ← k + 1
16:    end while
17:    return B_Best
18: end procedure
```

Figure 1: Sampled Fictitious Play

$$Q_d^t = Q_{dt}^{\text{input}} + \sum_{i=1}^{n} x_{id}^{t-t_{id}} \; and \; Q_d^t \leqslant Q_d^{Max}, \qquad \forall d = 1,\ldots,n; \forall t = 1,\ldots,T, \tag{14}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n}(t + v_d + t_{id} + Q_{d,t+t_{id}}^{\text{input}})x_{id}^t + \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ijd}^t = \sum_{t=1}^{T}\sum_{j=1}^{n} t x_{djk}^t, \qquad \forall d = 2,\ldots,n-1; \forall k = 1,\ldots,m, \tag{15}$$

$$\alpha_{ijd}^t \leqslant x_{jd}^t, \alpha_{ijd}^t \leqslant x_{id}^t, \alpha_{ijd}^t \geqslant x_{jd}^t + x_{id}^t - 1, \qquad \forall\, i,d,j,t, \tag{16}$$

**Experimental Results.** In all the experiments, utility values were randomly generated ranging from 1 to 5. All the agents start at time 1 and leave before time $T$. In the Figure 2, we provide a comparison for the results on small instances by MILP and SFP. SFP constantly performed much faster than MILP on small instances we experimented with[1]. In addition, in the Figure 3, we conducted experiments for SFP on large instances to show the efficiency and effectiveness of this heuristic. We compare the SFP results with Greedy solutions (Initially assuming there are no queues for each node and gradually adding in agents greedily). In this experiment, there were 8 nodes and 18 time units involved. We experimented with the different number of agents for 100 times each and averaged over the trials.

| Agents/ Nodes/ T/ Q | Runtime | | Utility collected | |
|---|---|---|---|---|
| | MILP | SFP | MILP | SFP |
| 2 /5 /5 /1 | 11.54s | 2.462s | 15 | 15 |
| 2 /5 /8 /1 | 234.22s | 2.830s | 19 | 19 |
| 3 /5 /8 /2 | - | 4.267s | - | 26 |

Figure 2: Comparison for MILP and SFP on small instances

| Agents/ Nodes/ T/ Q | Runtime | Utility collected | |
|---|---|---|---|
| | | Greedy | SFP |
| 10 /8 /18 /2 | 318.542s | 557 | 635 |
| 20 /8 /18 /2 | 350.123s | 623 | 690 |
| 30 /8 /18 /2 | 479.674s | 608 | 723 |

Figure 3: Results for SFP on larger instances

**Acknowledgment.** This research/project is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

# References

[1] Sylvain Boussier, Dominique Feillet, and Michel Gendreau. An exact algorithm for team orienteering problems. *4OR: A Quarterly Journal of Operations Research*, 5(3):211–230, 2007.

[2] I-Ming Chao, Bruce L. Golden, and Edward A. Wasil. The team orienteering problem. *European Journal of Operational Research*, 88(3):464–474, 1996.

[3] Marisa G. Kantor and Moshe B. Rosenwein. The orienteering problem with time windows. *The Journal of the Operational Research Society*, 43(6):629–635, 1992.

[4] Theodore J. Lambert III, Marina A. Epelman, and Robert L. Smith. A fictitious play approach to large-scale optimization. *Operations Research*, 53(3):477–489, 2005.

[5] R. Montemanni and L. Gambardella. Ant colony system for team orienteering problems with time windows. *Foundations of Computing and Decision Sciences*, 34(4):287—306, 2009.

[6] T. Tsiligirides. Heuristic methods applied to orienteering. *The Journal of the Operational Research Society*, 35(9):797–809, 1984.

[1] With respect to the last MILP result, we terminated it after running it for more than 12 hours.