Online Supplement to "Branch-and-Price for Personalized Multi-Activity Tour Scheduling"

María I. Restrepo[*1,2], **Bernard Gendron**[1,3], and **Louis-Martin Rousseau**[1,2]

[1]Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, CIRRELT
[2]Département de mathématiques et de génie industriel, Polytechnique Montréal, Montréal, Canada
[3]Département d'informatique et de recherche opérationelle, Université de Montréal, Montréal, Canada

1 Example for the Comparison Between the Two Formulations

The problem is a mono-activity tour scheduling with a three-day time horizon; each day is divided into four time intervals; the total tour length ranges between 4 and 6 time intervals; the minimum and maximum number of days are 1 and 2, respectively; there are no constraints for the minimum resting time; and the total number of employees is 1. The grammar used to compose the daily shifts is as follows:

$G = (\Sigma = (w, b), N = (S, X, W, B), P, S)$,
where productions P are: $S \to XW$, $X \to WB$, $W \to WW|w$, $B^{[2,2]} \to b$

Daily shifts have a working length of 3 time intervals and must have one break allocated in their second time interval (production $B^{[2,2]} \to b$). Table 1 presents the employee requirements and the structure of the feasible shifts and tours. The cost of the activity per time interval at each day is 1 and the costs of overcovering and undercovering of employee requirements are 1 and 2, respectively.

*Corresponding author: maria-isabel.restrepo-ruiz@polymtl.ca

Day (d)	1				2				3			
Time interval (i)	1	2	3	4	1	2	3	4	1	2	3	4
Empl. req. (b_{dij})	1	1	0	0	1	1	0	0	1	1	0	0
Shift1 (x_{11})	1	0	1	1	-	-	-	-	-	-	-	-
Shift2 (x_{21})	-	-	-	-	1	0	1	1	-	-	-	-
Shift3 (x_{31})	-	-	-	-	-	-	-	-	1	0	1	1
Tour1 (x_1)	1	0	1	1	1	0	1	1	0	0	0	0
Tour2 (x_2)	1	0	1	1	0	0	0	0	1	0	1	1
Tour3 (x_3)	0	0	0	0	1	0	1	1	1	0	1	1

Table 1: Employee requirements, shifts and tours structures.

For the Daily-based formulation, since all the shifts have the same working length (3 time intervals) it is easy to show that the constraint for the minimum and maximum number of working days (4) is redundant and that the summation of the value of the three decision variables (x_{11}, x_{21}, x_{31}) have to fall inside the interval [4/3, 6/3] because of constraints (5). Now, since all the shifts have the same structure and the employee requirements are the same at each day, we can conclude that the value of $f(\overline{F_{\mathcal{S}}})$ is the same when we evaluate the point (1,1,0) or the point (2/3, 2/3, 2/3). Therefore, we can evaluate the value of $f(\overline{F_{\mathcal{S}}})$ when all the variables fall, with the same value, inside the interval [4/9, 6/9]. Figure 1 presents the value of the objective function for the evaluated points, as well as the optimal solution $f(\overline{F_{\mathcal{S}}^{\star}}) = 16$ with $x_{11}^{\star} = x_{21}^{\star} = x_{31}^{\star} = 4/9$.

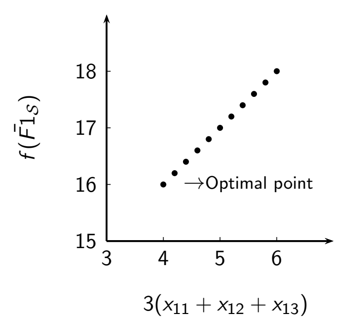

Figure 1: Optimal solution for the LP relaxation of the example using the Daily-based formulation.

The solution of the problem with the Daily-based formulation is not feasible for the Tour-based formulation, since all possible tours must have a length of 6 time intervals and must include 2 working days. As mentioned before, since the employee requirements and the shifts are the same for every day, the value of $f(\overline{F_{\mathcal{T}}})$ is the same when we evaluate all the points where $x_1 + x_2 + x_3 = 1$. In this case the optimal value of $f(\overline{F_{\mathcal{T}}^{\star}})$ is 18. Hence $f(\overline{F_{\mathcal{S}}^{\star}}) < f(\overline{F_{\mathcal{T}}^{\star}})$.

2 Label setting algorithm to solve the SPPRC

Let $\mathcal{Q}$ be the set of labels. Each label $l \in \mathcal{Q}$ has an associated path $\mathcal{P}(l)$ and a set of attributes: its resident node $v(l)$, its predecessor node $p(l)$, its cost $c(l)$, its distance $t(l)$, and its number of working days $d(l)$ accumulated along $\mathcal{P}(l)$.

Algorithm 1 presents the pseudocode of the labeling algorithm to solve the shortest path problem with resource constraints for each employee $e \in E$. The inputs are the tour graph $G^e(\mathcal{N}, \mathcal{A})$ and the maximum number of tours α to generate per iteration. The output corresponds to a vector of paths (tours) $\vec{\mathcal{P}^e}$ with negative reduced cost. Line 1 returns an initial set of labels (partial paths from source node v_s to all of its successors), and initializes the counter t for the number of tours generated ($t = 0$). Line 2 selects the first label l_1 to be processed according to its cost (the selected label is the one holding the lower cost). Line 4 searches to prune by bound the current label before extending it. This pruning is done by calculating an optimistic prediction of the total cost of the path that might be generated by the current label l_1. If such cost is negative, the label is not pruned, otherwise the label is removed from list $\mathcal{Q}$ without being processed. Line 5 seeks to extend label l_1 to all of its successor nodes ($v_i \in \mathcal{N}(v(l_1))$). A new label l_2 is created and stored in $\mathcal{Q}$ (Line 9) if it is feasible (Line 6), if it is non-dominated (Line 7) and if its resident node v_i is different than the sink node v_f. If the resident node is v_f and the cost of label l_2 is negative, a new path $\mathcal{P}(l_2)$ is stored in $\vec{\mathcal{P}^e}$ (Line 11).

The feasibility function checks if the label l_2 to be created (by extending l_1 to node v_i) can reach the minimum number of working days and tour length and, at the same time, if it does not exceed the maximum number of working days and tour length. The dominance function compares certain attributes of the label l_2 to be created with the rest of the labels $l \in \mathcal{Q} | l \neq l_2$. Hence, if the resident node, accumulated time and number of working days are the same for both labels, and if the cost of label l is lower or equal than the cost of label l_2, l dominates l_2. The algorithm stops when either set $\mathcal{Q}$ is empty or the number of tours t generated is greater than or equal to α.

Input $G^e(\mathcal{N}, \mathcal{A}), \alpha$
Output $\vec{\mathcal{P}^e}$

1: `initialization`
2: `selectLabel`
3: **while** $\mathcal{Q} \neq \emptyset \wedge t < \alpha$ **do**
4:     **if** `pruneByBound` (l_1) `= false` **then**
5:         **for** $v_i \in \mathcal{N}(v(l_1))$ **do**
6:             **if** `pruneByFeasibility` (l_1, v_i) `= false` **then**
7:                 **if** `dominance` (l_2) `= false` **then**
8:                     **if** $v_i \neq v_f$ **then**
9:                         $\mathcal{Q} \leftarrow l_2$
10:                     **else**
11:                         $\vec{\mathcal{P}^e} \leftarrow \mathcal{P}(l_2)$, $t = t + 1$
12:     `remove` l_1 `from` $\mathcal{Q}$
13:     `selectLabel`
14: **return** $\vec{\mathcal{P}^e}$

Algorithm 1: Label setting algorithm to solve the SPPRC