Branch-and-Price for Personalized Multi-Activity Tour Scheduling

María I. Restrepo
Bernard Gendron
Louis-Martin Rousseau

January 2015

CIRRELT-2015-03

Branch-and-Price for Personalized Multi-Activity Tour Scheduling

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

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

[3] Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal, Canada H3C 3A7

Abstract. This paper presents a branch-and-price approach to solve personalized tour scheduling problems in a multi-activity context. Two formulations are considered. In the first formulation, columns correspond to daily shifts that are modeled with context-free grammars and tours are assembled in the master problem by means of extra constraints. In the second formulation, columns correspond to tours that are built in a two-phase procedure. The first phase involves the composition of daily shifts, while the second phase assembles those shifts to generate tours using a shortest path problem with resource constraints. Both formulations are flexible enough to allow different start times, lengths and days-off patterns, as well as multiple breaks, continuity and discontinuity in labor requirements. We present computational experiments on problems dealing with up to five work activities and one week planning horizon. The results show that the second formulation is stronger in terms of its lower bound and that it is able to find high-quality solutions for all instances with an optimality gap lower than 1%.

Keywords. Multi-activity tour scheduling problem, branch-and-price, context-free grammars, shortest path problem with resource constraints.

Acknowledgements. The authors would like to thank the Fonds de recherche du Québec - Nature et technologies (FRQNT) which supported this work with a grant.

---

* Corresponding author: Maria-Isabel.Restrepo@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2015

1 Introduction

Personnel scheduling problems consist in constructing a set of feasible shift schedules and assigning them to the company staff to satisfy a given demand for staff requirements. These problems arise in diverse organizations such as hospitals, airline companies, retail stores, call centers and banks, where they have become important tasks, due to the necessity to achieve a better level of service, to reduce staff costs and to increase employee satisfaction.

According to Baker [4], three main categories of problems can be distinguished in personnel scheduling: *Shift scheduling*, *Days-off scheduling* and *Tour scheduling*. The first category deals with the specification of work and rest periods to assign to shifts, as well as the selection of a set of those shifts to satisfy the demand for staff requirements. In shift scheduling, the *planning horizon* is usually one day divided into *time intervals* of equal length. The second category involves the selection of *days-off* over a planning horizon of at least one week. Such selection is usually restricted by some *employee preferences* or *workplace agreements*. The last category includes problems that arise from the integration of shift scheduling and days-off scheduling; therefore, the aim of tour scheduling problems is to specify the time intervals of the day and the days of the week in which employees must work.

Complex extensions of classical personnel scheduling problems appear when real applications are considered. For instance, when more than one activity has to be scheduled, the *Multi-activity shift scheduling* and the *Multi-activity tour scheduling* problems arise. In both extensions not only the specification of work and rest periods is necessary, but also the assignment of activities to the shifts. In a multi-activity context, specific characteristics related with *work rules*, *workplace agreements*, and *employee skills* and *preferences* define the rules to build employee schedules.

The problem considered in this paper is the *personalized multi-activity tour scheduling problem* (MATSP). In the MATSP, a tour can be seen as a schedule over a planning horizon of at least one week, where for every time interval it must be specified if the employee is working on an activity, having a break or resting. Aside from multiple activities, undercovering and overcovering of demand are considered. The MATSP is flexible enough to be easily adapted depending on different work rules and scenarios. The number of feasible tours grows fast with the number of activities, the number of employees and the length of the time horizon, making the complete enumeration of tours impractical. Therefore, we propose column generation approaches embedded into branch-and-price (B&P) methods. Two formulations for the MATSP are considered, each giving rise to a different B&P algorithm. First, the *Daily-based* formulation consists of an extension of a multi-activity shift scheduling problem, where columns correspond to daily shifts and tours are assembled in the master problem by means of extra constraints. Second, the *Tour-based* formulation, where columns correspond to tours that are built in a two-phase procedure. The first phase involves the composition of daily shifts, while the second phase assembles those shifts to generate tours.

The outline of the paper is the following. Section 2 presents the literature review related with personnel scheduling problems, as well as some background material. The definition of the problem, the two formulations and some properties of them are presented in Section 3. The B&P algorithms for the Daily-based formulation and the Tour-based formulation are presented in Sections 4 and 5, respectively. Computational experiments are discussed in Section 6. Finally, Section 7 presents the concluding remarks and future work.

2 Background and Related Research

In this section, we present a literature review on the models and methods proposed to solve shift scheduling, multi-activity and tour scheduling problems. Then, we present an introduction on the use of grammars in the context of multi-actity shift scheduling.

2.1 Shift Scheduling

Two different modeling approaches can be distinguished in the literature on shift scheduling problems: *explicit* and *implicit* models. Explicit models allow to consider flexibility in terms of break placement, start times and shift length by representing each feasible shift with a different variable. On the contrary, implicit models define one variable for every shift and break type, seeking to reduce the number of decision variables by compromising model flexibility.

Dantzig [18] was the first author to introduce an explicit model for the SSP. The model is based on a *set covering* formulation in which the objective is to minimize the total labor cost, ensuring that labor requirements at every time interval are met. Trying to reduce the number of variables, Moondra [27] proposes a method that implicitly represents shifts and considers flexibility regarding multiple shift lengths and start times. Break flexibility is considered in Bechtold and Jacobs [5] with an implicit formulation where shifts are grouped into shift types according to their start time, length and break window.

Aykin [2] and Rekik et al. [33] present two extensions of Bechtold and Jacobs' formulation. In the first extension, the author tackles multiple rest, meal breaks and break windows by introducing integer variables for the number of employees assigned to a shift and starting their breaks at different time intervals. The second extension deals with two implicit models that include a reformulation of forward and backward constraints and a transportation structure to match shifts with admissible breaks.

Column generation (CG) and constraint programming (CP) have been recently proposed as alternatives to solve complex shift scheduling problems. CG is used when the incorporation of flexibility in the composition of shifts causes a considerable increase in the number of variables. In this method, two strategies can be adopted to reach integrality in the problem: a heuristic approach, where the master problem is solved by forcing the integrality constraints on the decision variables, and an exact method, where the CG procedure is embedded into a B&P algorithm. In that vein, Mehrotra et al. [26] use B&P and exploit the advantages of a set covering formulation to solve a shift scheduling problem with multiple rest breaks, one meal break and break windows. On the other hand, CP is used to model shift scheduling problems where work rules involve non trivial relationships between variables. As an illustration, Côté et al. [13] take advantage of the expressiveness of a *Deterministic finite automaton* (DFA) that, used in a 0-1 mixed integer programming model, helps to implicitly express a large set of rules and represent all possible patterns for an employee timetabling problem.

2.2 Multi-Activity Shift Scheduling

In one of the first attempts to solve multi-activity shift scheduling problems, Ritzman et al. [35] develop a heuristic approach by integrating a construction method and a simulation component to schedule employees in a post office over a planning horizon of one week. Although the authors tackled the multi-activity context, they do not consider breaks nor rules related with switching between activities.

A few decades later, approaches using CP, formal languages and context-free grammars were suggested in Demassey et al. [19], Quimper and Rousseau [30] and Côté et al. [12, 14]. In the first approach [19], a CP-based column generation algorithm is presented as a way to model complex regulation constraints to solve large instances of multi-activity shift scheduling problems. Quimper and Rousseau [30], solve multi-activity shift scheduling problems with up to ten work activities by using specialized graph structures that are derived from formal languages and solved via Large neighborhood search. Côté et al. [12], propose two models: one that uses an automaton to derive a network flow model, and another one that benefits from context-free grammars to obtain a MIP model in which an and/or graph structure is used. Despite their ability to easily handle complex rules, the models present some scalability issues when the number of employees and the number of activities increase. Côté et al. [14], seek to solve the scalability issues of their previous models by introducing an implicit formulation that encapsulates model symmetry by using integer variables. Computational results show that, in the mono-activity case, the solution times of the model are comparable and sometimes better than the results presented in the literature and that, in the multi-activity case, the model is able to solve to optimality instances with up to ten work activities.

To solve the personalized version of the problem introduced in Côté et al. [14], Côté et al. [15] present a grammar-based column generation method where the pricing subproblems are formulated using grammars and solved with a dynamic programming algorithm. Although the expressiveness of grammars enables to encode a large set of work rules over shifts, some limitations are present regarding shift total length over longer planning horizons (e.g., one week). Boyer et al. [6] extends the previous work, where besides considering multiple activities, it also includes multiple tasks. An extensive study of branching strategies is made, showing that the method is able to find, in a reasonable amount of time, the solution for all test instances with an optimality gap lower than 5%.

Instead of working with explicit and implicit models or with CP, some authors have proposed different methods to tackle multi-activity shift scheduling problems. Some of them include Tabu search [17], heuristic column generation [34], decomposition techniques [20], or a simplification of the multi-activity shift scheduling problem [21, 25] by fixing sequences of work, rest days, shift types and breaks.

2.3 Tour Scheduling

Since the introduction of Dantzig's model for shift scheduling [18], a lot of research has been conducted to consider more realistic and complex versions of the problem. One example of such extensions is the work of Morris and Showalter [28], which introduces the first integer programming formulation to solve tour scheduling problems based on Dantzig's set covering model. The authors combine LP and heuristic methods to solve a two-phase problem. In the first phase, daily schedules are generated and sent to the second phase where weekly tours are constructed. Trying to introduce flexibility without increasing dramatically the size of the problem, Bailey [3] presents a formulation to implicitly model the start times in a tour scheduling problem. A construction heuristic is used to assign shifts under a limited staff size constraint, while a rounding heuristic is used to achieve integrality of the variables.

Jarrah et al. [24] propose an implicit model with aggregated variables that decomposes the weekly tour scheduling problem into seven daily-shift scheduling subproblems. A transportation model and a postprocessor are used to assign breaks to shifts and shifts to tours, respectively. The authors test their method on a real-world application, running several scenarios including flexibility in start times and break allocation. Jacobs and Brusco [23] present a compact implicit model to demonstrate the importance of start time bandwidth flexibility in tour scheduling. Computational results suggest that allowing start-time flexibility reduces significantly the required workforce size when compared with fixed start time.

Attempting to develop more realistic models, Brusco and Jacobs [9] considers both start time and meal break flexibility in an implicit integer programming model to solve the continuous tour scheduling problem. The authors evaluate their approach on a real-world application, showing that the effect of the scheduling policies on the optimal workforce size can vary significantly depending on the level of other policies.

In order to solve a discontinuous tour scheduling problem (shifts are allowed to overlap from one day to the next) in a call center, Çezik et al. [11] propose a model with two components: a daily-shift generator and a days-off generator. In the former, seven daily shift schedules are generated through implicit modeling, while in the latter, weekly requirements are met via network flows. The proposed model is able to solve half of the real instances to optimality by using a heuristic B&P algorithm.

In a recent work, Brusco and Johns [10] introduce an integrated approach to overcome the lack of integration between start time selection and tour construction by using Tabu search and a cutting plane method. The former is useful for the start time selection, while the latter handles tour construction. Computational experiments are conducted with and without consistency in demand patterns. In both cases, the method has a good performance in terms of computational time.

Decomposition techniques and column generation are also often used in the context of tour scheduling problems. As an illustration, Rekik et al. [32] use Benders decomposition to solve a continuous tour scheduling problem where the subproblems are modeled with a transportation structure, and the forward and backward constraints introduced by Bechtold and Jacobs [5] are used as a set of initial feasibility cuts. After conducting an extensive analysis, the authors conclude that the proposed model considerably decreases the number of variables at the cost of a small increase in the number of constraints. Ni and Abeledo [29] present a B&P algorithm to solve the continuous version of the problem where weekly tours are decomposed into daily shifts and start-time patterns. Computational experiments show that for large-scale instances where implicit methods often fail to find a feasible solution, the proposed method is able to find near-optimal solutions.

Because achieving integrality for large-scale instances is a difficult task, Al-Yakoob and Sherali [1] propose a heuristic column generation to schedule, over one week, different categories of employees. Employee preferences for work centers, shift types and days-off are considered. Given previously defined shift types, the method is able to generate, in a reasonable amount of time, employee schedules for up to 90 stations and 336 employees.

Combining implicit and explicit shift definitions and developing an exact B&P algorithm, Brunner and Bard [7] solve a discontinuous tour scheduling problem over a one-week planning horizon for postal service employees. The authors analyze the flexibility impact on workforce size, costs and utilization rate by considering several scenarios in the computational experiments.

Brunner and Stolletz [8] present a stabilized B&P algorithm to solve a discontinuous tour scheduling problem that includes flexibility regarding labor regulations and assignment of lunch breaks. The master problem is based on a set covering formulation. Computational experiments show that the model convergence is faster when the stabilization technique is used.

The reader is referred to Van den Bergh et al. [36] for a comprehensive review of recent papers on tour scheduling problems.

The literature review on tour scheduling problems reveals that no method has been proposed to integrate tour scheduling and multi-activity shift scheduling problems. In particular, no rules for the allocation and transition between activities were ever taken into account in the existing research. The literature review on multi-activity shift scheduling problem (Section 2.2) shows that some authors have addressed these problems over planning horizons longer than one day, but only in situations where either weekly patterns were previously defined or rules concerning total tour length and days-off were not considered.

The present paper addresses these gaps in the literature by proposing models and methods that integrate the tour scheduling and multi-activity shift scheduling problems. To efficiently handle the multi-activity context, we make use of context-free grammars, which are reviewed next.

2.4 Grammars and Shift Scheduling Problems

We define a *context-free grammar* as a tuple $G = \langle \Sigma, N, P, S \rangle$ where Σ is an alphabet of characters called the *terminal symbols*, N is a set of *non-terminal symbols*, $S \in N$ is the starting symbol and P is a set of *productions* represented as $A \rightarrow w$, where $A \in N$ is a non-terminal symbol and w is a sequence of terminal and non-terminal symbols. The productions of a grammar can be used to generate new symbol sequences until only terminal symbols are part of the sequence. A *Context-free language* is the set of sequences accepted by a context-free grammar.

A *parse tree* is a tree where each leaf is labeled with a terminal and each inner-node is labeled with a non-terminal. A grammar recognizes a sequence if and only if there exists a parse tree where the leaves, when listed from left to right, reproduce the sequence. An *and/or graph* is a graph where each leaf corresponds to an assignment that can either be true or false. An and-node is true if all of its children are true. An or-node is true if one of its children is true. The root is true if the grammar accepts the sequence encoded by the leaves. The and/or graph embeds every possible parse tree of a grammar.

Finally, a *DAG* Γ is a directed acyclic graph that embeds all parse trees associated with words of a given length n recognized by a grammar. The DAG Γ has an and/or structure where the and-nodes represent productions from P and or-nodes represent non-terminals from N and letters from Σ. The DAG Γ is built by a procedure proposed in Quimper and Walsh [31].

In the shift scheduling context, the use of grammars allows both to include work rules regarding the definition of shifts and to handle the multi-activity context in an easy way. Thus, feasible shifts can be represented as words in a context-free language. For example, words $rw_1w_1bw_2$ and $w_1bw_2w_1r$ are recognized as valid shifts in a two-activity shift scheduling problem where letters w_1, w_2, b and r represent working on activity 1, working on activity 2, break and rest periods, respectively. The time horizon consists of five time intervals, shifts have a length of four periods and must contain exactly one break of one period that can be placed anywhere during the shift except at the first or the last period. The grammar that defines the shifts on this example follows:

$G = (\Sigma = (w_1, w_2, b, r), N = (S, F, X, W, B, R), P, S)$,
where productions P are: $S \rightarrow RF|FR$, $F \rightarrow XW$, $X \rightarrow WB$, $W \rightarrow WW|w_1|w_2$, $B \rightarrow b$, $R \rightarrow r$, and symbol $|$ specifies the choice of production.

In the previous example, productions $W \rightarrow w_1$, $W \rightarrow w_2$, $B \rightarrow b$ and $R \rightarrow r$ generate the terminal symbols associated with working on activity 1, working on activity 2, having a break or having a rest period inside of the shift, respectively. Production $W \rightarrow WW$ generates two non-terminal symbols, W, meaning that the shift will include a working subsequence. Production $X \rightarrow WB$ means that the shift will include working time and then it will be followed by a break. Production $F \rightarrow XW$ generates a subsequence of length four (the daily shift), which includes working time followed by a break to finish with more working time. Finally, the last two productions are $S \rightarrow RF$ and $S \rightarrow FR$. The former generates a sequence starting with a period of rest followed by the daily shift. The latter generates a sequence starting with the daily shift followed by a period of rest.

Let O^{π}_{dil} be the or-nodes associated with $\pi \in N \cup \Sigma$, i.e., with non-terminals from N or letters from Σ, that generate a subsequence at day d, position i of length l. Note that if $\pi \in \Sigma$, the node is a leaf and l is equal to one. On the contrary, if $\pi \in N$, the node represents a non-terminal symbol and $l > 1$. $A^{\Pi,k}_{dil}$ is the kth and-node representing productions $\Pi \in P$ generating a subsequence at day d, from position i of length l. There are as many $A^{\Pi,k}_{dil}$ nodes as there are ways of using P to generate a sequence of length l from position i during day d. The sets of or-nodes and and-nodes from day d are denoted by O_d and A_d, respectively. The root node is described by O^{S}_{d1n} and its children by $A^{\Pi,k}_{d1n}$. Figure 1 represents the DAG Γ associated with the grammar of the example, where dashed-line or-nodes are part of the parse trees associated with and-node $A^{S\rightarrow RF,1}_{15}$, while continuous-line or-nodes are part of the parse trees associated with and-node $A^{S\rightarrow FR,1}_{15}$. For clarity, we did not include the subscript of the day in the notation of the nodes.

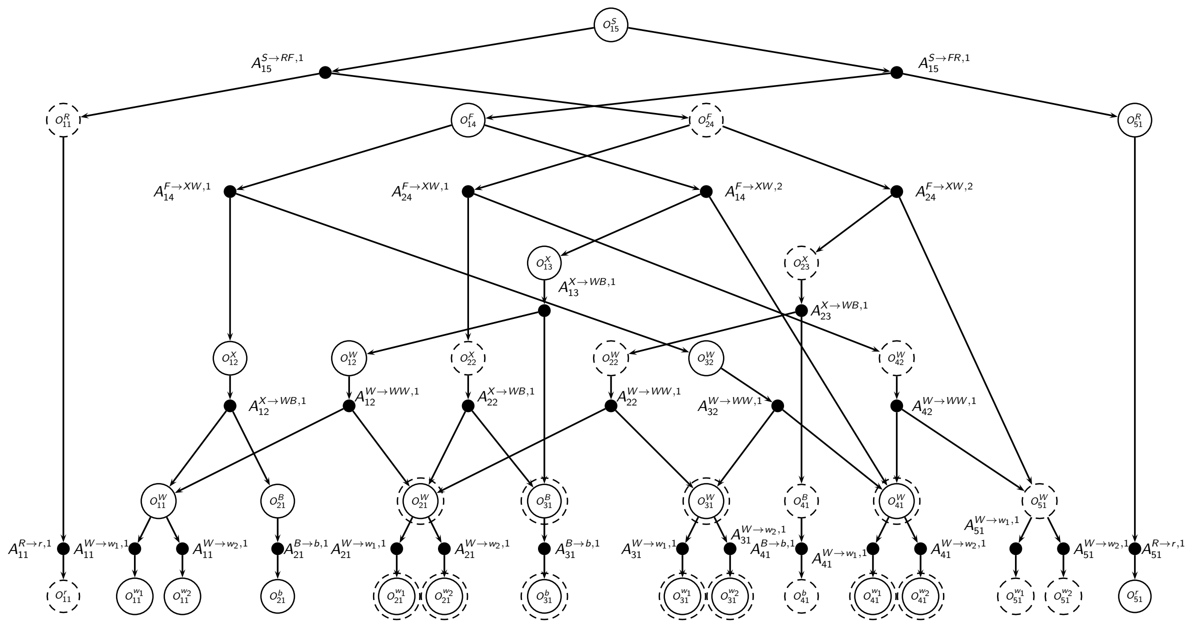
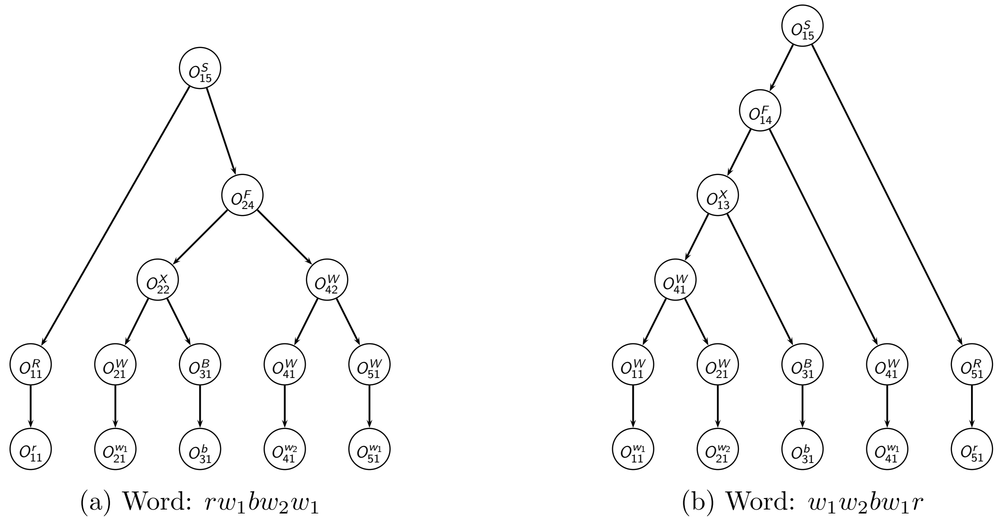


Figure 1: DAG Γ on words of length five and two work activities.

Figure 2 shows two of the 32 parse trees that are embedded into the DAG Γ presented in Figure 1. Note that the leaves correspond to letters from Σ that form a word, in this case of length five, when listed from left to right.



(a) Word: $rw_1bw_2w_1$ (b) Word: $w_1w_2bw_1r$

Figure 2: Parse trees derived from DAG Γ on words of length five and two work activities.

3 Problem Definition and Formulations

In this section, we first introduce the problem studied and its notation to later define the mathematical models for the Daily-based and the Tour-based approaches. We then show that the Tour-based formulation yields a better linear programming (LP) relaxation bound than the Daily-based formulation.

3.1 Problem Definition and Notation

The problem addressed in this paper is a Tour scheduling problem in a multi-activity context where the set of activities is denoted by J. The planning horizon is at least one week in which each day $d \in D$ is divided into I_d time intervals of equal length. Each employee $e \in E$ have different skills, meaning that the personalized version of the problem is considered. The set of feasible tours for each employee e is denoted by $\mathcal{T}^e$, while the set of feasible shifts for each employee e at each day d is denoted by $\mathcal{S}_d^e$. The notation used in both formulations is as follows:

General parameters

b_{dij}: staff requirements for day d, time interval i and activity j;

c^e_{dij}: cost of employee e working on day d, time interval i and activity j;

c^+_{dij}, c^-_{dij}: overcovering and undercovering costs of employee requirements for day d, time interval i and activity j, respectively;

J^e_{di}: set of work activities that employee e can perform at period i of day d.

Daily shift parameters

ϵ: minimum resting time between two consecutive daily shifts;

$\mathcal{S}^e_{di}(1)$: set of shifts that finish at time interval i during day d for employee e;

$\mathcal{S}^e_{di}(2)$: set of shifts that cover from i to $i+\epsilon$ time intervals during day d for employee e;

δ^e_{dijs}: parameter that takes value 1, if shift s covers time interval i and work activity j during day d for employee e, and assumes value 0 otherwise;

c^e_{ds}: cost associated to shift s during day d for employee e ($c^e_{ds} = \sum\limits_{i \in I_d} \sum\limits_{j \in J^e_{di}} \delta^e_{dijs} c^e_{dij}$);

Every daily shift s from day d has a set of attributes: its start period t_{ds}, its length l_{ds} (considering breaks), its working time w_{ds}, and its end period $f_{ds} = t_{ds} + l_{ds} - 1$.

Tour parameters

Δ_l, Δ_u: minimum and maximum number of working days in a tour, respectively;

Θ_l, Θ_u: minimum and maximum tour length in time intervals, respectively;

$\Phi = |D| - \Delta_l$: maximum number of days-off in a tour;

ϕ^e_{dst}: parameter that takes value 1 if tour t includes shift s at day d for employee e, and assumes value 0 otherwise;

ρ^e_{dijt}: parameter that takes value 1 if tour t covers time interval i and work activity j at day d for employee e, and assumes value 0 otherwise ($\rho^e_{dijt} = \sum_{s \in \mathcal{S}^e_d} \phi^e_{dst}\delta^e_{dijs}$);

c^e_t: cost of tour t for employee e ($c^e_t = \sum_{d \in D}\sum_{i \in I_d}\sum_{j \in J^e_{di}} \delta^e_{dijt} c^e_{dij} = \sum_{d \in D}\sum_{s \in \mathcal{S}^e_d} \phi^e_{dst} c^e_{ds}$).

3.2 The Daily-Based Formulation

In order to solve the MATSP, we propose, as a first formulation, a natural extension of the model presented in Côté et al. [15]. In this extension, daily shifts are linked into tours by means of extra constraints that assure the minimum and maximum tour length, the minimum and maximum number of working days per tour and the minimum resting time between consecutive shifts. We define the following variables:
x^e_{ds}: binary variable that takes value 1 if shift s on day d is assigned to employee e, and assumes value 0 otherwise;
y^+_{dij}, y^-_{dij}: slack variables representing overcovering and undercovering of employee requirements for day d, time interval i and activity j, respectively.

The Daily-based formulation, denoted $F_{\mathcal{S}}$, is as follows:

$$f(F_{\mathcal{S}}) = \min \sum_{e \in E}\sum_{d \in D}\sum_{s \in \mathcal{S}^e_d} c^e_{ds}x^e_{ds} + \sum_{d \in D}\sum_{i \in I_d}\sum_{j \in J} c^+_{dij}y^+_{dij} + \sum_{d \in D}\sum_{i \in I_d}\sum_{j \in J} c^-_{dij}y^-_{dij} \tag{1}$$

$$\sum_{e \in E}\sum_{s \in \mathcal{S}^e_d} \delta^e_{dijs}x^e_{ds} - y^+_{dij} + y^-_{dij} = b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J, \tag{2}$$

$$\sum_{s \in \mathcal{S}^e_d} x^e_{ds} \leq 1,\ \forall\, e \in E,\ \forall\, d \in D, \tag{3}$$

$$\Delta_l \leq \sum_{d \in D}\sum_{s \in \mathcal{S}^e_d} x^e_{ds} \leq \Delta_u,\ \forall\, e \in E, \tag{4}$$

$$\Theta_l \leq \sum_{d \in D}\sum_{s \in \mathcal{S}^e_d} w_{ds}x^e_{ds} \leq \Theta_u,\ \forall\, e \in E, \tag{5}$$

$$\sum_{s \in \mathcal{S}^e_{di}(1)}\sum_{j \in J} \delta^e_{dijs}x^e_{ds} + \sum_{s \in \mathcal{S}^e_{di}(2)}\sum_{j \in J} \delta^e_{dijs}x^e_{ds} \leq 1,\ \forall\, e \in E,\ d \in D,\ i \in I_d, \tag{6}$$

$$x^e_{ds} \in \{0,1\},\ \forall\, e \in E,\ d \in D,\ s \in \mathcal{S}^e_d, \tag{7}$$

$$y^+_{dij},\ y^-_{dij} \geq 0,\ \forall\, d \in D,\ i \in I_d,\ j \in J. \tag{8}$$

The objective of $F_{\mathcal{S}}$, (1), is to minimize the total staffing cost plus the penalization for overcovering and undercovering of demand. Constraints (2) ensure that staff requirements per day $d \in D$, time interval $i \in I_d$ and work activity $j \in J$ are met. Constraints (3) guarantee that every employee is assigned to at most one shift per day. Constraints (4) and (5) enforce a minimum and maximum number of working days (Δ_l and Δ_u) and tour length (Θ_l and Θ_u), respectively. Constraints (6) ensure a minimum rest time between consecutive shifts. Finally, constraints (7)-(8) set the binary nature of variables x^e_{ds} and the non-negativity of variables y^+_{dij}, y^-_{dij}.

3.3 The Tour-Based Formulation

The Tour-based formulation makes use of slack variables as in the Daily-based formulation, but also includes the following decision variables related to tours:
x^e_t: binary variable that takes value 1 if tour t is assigned to employee e, and assumes value 0 otherwise.

Hence, the constraints related with the link between daily shifts and tours are considered in the definition of feasible tours and not in the mathematical model. The Tour-based formulation, denoted $F_{\mathcal{T}}$, is as follows:

$$f(F_{\mathcal{T}}) = \min \sum_{e \in E} \sum_{t \in \mathcal{T}^e} c^e_t x^e_t + \sum_{d \in D} \sum_{i \in I_d} \sum_{j \in J} c^+_{dij} y^+_{dij} + \sum_{d \in D} \sum_{i \in I_d} \sum_{j \in J} c^-_{dij} y^-_{dij} \tag{9}$$

$$\sum_{e \in E} \sum_{t \in \mathcal{T}^e} \rho^e_{dijt} x^e_t - y^+_{dij} + y^-_{dij} = b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J, \tag{10}$$

$$\sum_{t \in \mathcal{T}^e} x^e_t = 1,\ \forall\, e \in E, \tag{11}$$

$$x^e_t \in \{0,1\},\ \forall\, e \in E,\ t \in \mathcal{T}^e, \tag{12}$$

$$y^+_{dij},\ y^-_{dij} \geq 0,\ \forall\, d \in D,\ i \in I_d,\ j \in J. \tag{13}$$

The objective of $F_{\mathcal{T}}$, (9), is to minimize the total staffing cost plus the penalization for overcovering and undercovering of demand. Constraints (10) ensure that staff requirements per day $d \in D$, time interval $i \in I_d$ and work activity $j \in J$ are met. Constraints (11) guarantee that every employee is assigned to exactly one tour. Finally, constraints (12)-(13) set the binary nature of variables x^e_t and the non-negativity of variables y^+_{dij}, y^-_{dij}.

3.4 Comparison Between the Two Formulations

Let $f(\overline{F_{\mathcal{S}}})$ and $f(\overline{F_{\mathcal{T}}})$ be the LP relaxation bounds of the Daily-based formulation and the Tour-based formulation, respectively, with $\overline{F_{\mathcal{S}}}$ and $\overline{F_{\mathcal{T}}}$ the corresponding LP relaxations.

Proposition 1: $f(\overline{F_{\mathcal{S}}}) \leq f(\overline{F_{\mathcal{T}}})$.

Proof: Consider the Daily-based formulation (1)-(8) to which we add the following redundant constraints:

$$y^+_{dij},\ y^-_{dij} \leq b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J. \tag{14}$$

Note that the LP relaxation of the resulting model is equivalent to $\overline{F_{\mathcal{S}}}$. Using the resulting model, we dualize constraints (2) to obtain the following Lagrangian subproblem, denoted by $F_{\mathcal{S}}(\beta)$, where β_{dij} are the Lagrange multipliers associated to constraints (2):

$$f(F_{\mathcal{S}}(\beta)) = \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}\beta_{dij}b_{dij}+$$
$$\min \sum_{e\in E}\sum_{d\in D}\sum_{s\in\mathcal{S}_d^e} c_{ds}^e x_{ds}^e - \sum_{e\in E}\sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}\sum_{s\in\mathcal{S}_d^e}\beta_{dij}\delta_{dijs}^e x_{ds}^e$$
$$\sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}(c_{dij}^- - \beta_{dij})y_{dij}^- + \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}(c_{dij}^+ + \beta_{dij})y_{dij}^+$$
subject to constraints (3)-(8) and (14).

Note that $F_{\mathcal{S}}(\beta)$ decomposes into two subproblems: one expressed in terms of the x variables only, with feasible set X, and the other involving only the y variables, with feasible set Y. The subproblem that depends only on the x variables is defined by:

$$\min \sum_{e\in E}\sum_{d\in D}\sum_{s\in\mathcal{S}_d^e}\left(c_{ds}^e - \sum_{i\in I_d}\sum_{j\in J}\beta_{dij}\delta_{dijs}^e\right)x_{ds}^e$$
subject to constraints (3)-(7).

This subproblem can itself be decomposed by employee and its feasible set is defined as the finite set of points $X = \Pi_{e\in E}X^e$, where X^e corresponds to (3)-(7) for each employee e. The subproblem involving only the y variables is defined by:

$$\min \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}(c_{dij}^- - \beta_{dij})y_{dij}^- + \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J}(c_{dij}^+ + \beta_{dij})y_{dij}^+$$
subject to the non-negativity constraints (8) and the bound constraints (14).

Therefore, the feasible set $X \times Y$ of the Lagrangian subproblem $F_{\mathcal{S}}(\beta)$ can be written as $X \times Y = (\Pi_{e\in E}X^e) \times Y$.The corresponding Lagrangian dual can be written as $Z = \max_\beta f(F_{\mathcal{S}}(\beta))$. By Lagrangian duality theory [22], this Lagrangian dual is equivalent to the following LP model:

$$Z = \min \sum_{e\in E}\sum_{d\in D}\sum_{s\in\mathcal{S}_d^e} c_{ds}^e x_{ds}^e + \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J} c_{dij}^+ y_{dij}^+ + \sum_{d\in D}\sum_{i\in I_d}\sum_{j\in J} c_{dij}^- y_{dij}^-$$
$$\sum_{e\in E}\sum_{s\in\mathcal{S}_d^e}\delta_{dijs}^e x_{ds}^e - y_{dij}^+ + y_{dij}^- = b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J,$$
$$(x, y) \in conv(X \times Y).$$

where $conv(A)$ is the convex hull of any set A. Clearly, $Z \geq f(\overline{F_{\mathcal{S}}})$, since $conv(X \times Y)$ is contained in the feasible set of $\overline{F_{\mathcal{S}}}$.

Since $X \times Y = (\Pi_{e\in E}X^e) \times Y$ and Y is a bounded polyhedron, we have $conv(X \times Y) = (\Pi_{e\in E}conv(X^e)) \times Y$. Now, for any employee $e \in E$, every point x^e in $conv(X^e)$ can be written as a convex combination of the extreme points of $conv(X^e)$. Any such extreme point $\xi^e(t)$ corresponds to a tour $\mathcal{T}^e$ for employee e. Thus, we can write $x^e = \sum_{t \in \mathcal{T}^e} \mu^e(t)\xi^e(t)$, where $\mu^e(t)$ is the convex combination weight associated to tour $t \in \mathcal{T}^e$, i.e., $\sum\limits_{t \in \mathcal{T}^e} \mu^e(t) = 1$, $\forall\, e \in E$ and $\mu^e(t) \geq 0$, $\forall\, t \in \mathcal{T}^e$, $e \in E$. Thus, we have:

$$Z = \min \sum_{e \in E} \sum_{t \in \mathcal{T}^e} \left(\sum_{d \in D} \sum_{s \in \mathcal{S}_d^e} c_{ds}^e \xi_{ds}^e(t) \right) \mu^e(t) + \sum_{d \in D} \sum_{i \in I_d} \sum_{j \in J} c_{dij}^+ y_{dij}^+ + \sum_{d \in D} \sum_{i \in I_d} \sum_{j \in J} c_{dij}^- y_{dij}^-$$

$$\sum_{e \in E} \sum_{t \in \mathcal{T}^e} \left(\sum_{s \in \mathcal{S}_d^e} \delta_{dijs}^e \xi_{ds}^e(t) \right) \mu^e(t) - y_{dij}^+ + y_{dij}^- = b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J,$$

$$\sum_{t \in \mathcal{T}^e} \mu^e(t) = 1,\ \forall\, e \in E,$$

$$\mu^e(t) \geq 0,\ \forall\, t \in \mathcal{T}^e, e \in E,$$

$$0 \leq y_{dij}^+,\ y_{dij}^- \leq b_{dij},\ \forall\, d \in D,\ i \in I_d,\ j \in J.$$

We remark that $\xi_{ds}^e(t) = \phi_{dst}^e$, for any $e \in E$, $d \in D$,$s \in \mathcal{S}_d^e$ and $t \in \mathcal{T}^e$. If we let $\mu^e(t) = x_t^e$, for each $t \in \mathcal{T}^e$ and $e \in E$, we obtain the LP relaxation model of the Tour-based formulation (9)-(13) with the redundant constraints (14). Thus, $f(\overline{F_{\mathcal{T}}}) = Z \geq f(\overline{F_{\mathcal{S}}})$. $\square$.

Appendix A presents an example where a feasible solution of $\overline{F_{\mathcal{S}}}$ is not feasible for $\overline{F_{\mathcal{T}}}$, which implies that, in this case, the optimal LP relaxation value of the Daily-based formulation is lower than the optimal LP relaxation value of the Tour-based formulation.

4 B&P for the Daily-Based Formulation

In this section, we present a B&P algorithm for the Daily-based formulation. First, we introduce the restricted master problem. Second, we describe the pricing subproblems. Finally, we present the branching rule used to find near optimal integer solutions.

4.1 Restricted Master Problem

In practical scenarios, the complete enumeration of the set of feasible shifts $\mathcal{S}_d^e$ for every employee e and every day d is intractable due to the incorporation of flexibility regarding shift length, break placement, shift start times, among others. Therefore, we define a restricted master problem $F_{\widetilde{\mathcal{S}}}$ as the LP relaxation of problem (1)-(8) over restricted sets of shifts $\widetilde{\mathcal{S}}_d^e \subseteq \mathcal{S}_d^e$.

Let β_{dij}, λ_d^e, δ^e, θ^e and α_{di}^e be the dual variables associated with constraints (2) to (6), respectively. The reduced cost $\bar{c}_{ds}^e$ of column (shift) s during day d for employee e is given by:

$$\bar{c}_{ds}^e = \sum_{i \in I_d} \sum_{j \in J_{di}^e} (c_{dij}^e - \beta_{dij})\delta_{dijs}^e - \lambda_d^e - \delta^e - w_{ds}\theta^e - \alpha_{df_{ds}}^e - \sum_{i=f_{ds}-\epsilon}^{f_{ds}} \alpha_{di}^e \tag{15}$$

The objective in the column generation procedure is to find, at every iteration, columns with negative reduced cost, i.e., $\bar{c}^e_{ds} < 0$. Hence, (15) is used in the objective function of the pricing subproblems for every employee and every day of the planning horizon. Under a B&P algorithm, if no negative reduced cost columns can be generated, the restricted master problem is optimal for the node under evaluation.

4.2 Pricing Subproblems for Daily Shift Generation

New columns are generated based on the work presented in Côté et al. [15] and the concepts of Section 2.4. For each employee e and day d, a grammar G^e_d is generated. Such grammar represents the possible shifts that an employee can perform during the day and its generation is made by considering the employee skills, its preferences, availability, work rules and regulations such as minimum and maximum shift length, start times, minimum and maximum number of activities and breaks per shift, position of breaks and rules for the transition between activities.

From each grammar G^e_d, we generate a DAG Γ^e_d that is used to find negative reduced cost shifts for every employee e and day d. In Γ^e_d each children of the root node $A^{\pi,k}_{d1n} \in ch(O^S_{d1n})$ represents a daily shift and its corresponding reduced cost is computed by solving the dynamic programming algorithm presented in Quimper and Rousseau [30]. The algorithm traverses the DAG from the leaves to the root node by summing up the costs of the children of the and-nodes and choosing the minimum cost child of the or-nodes. It should be noted that, at every iteration of the column generation, the cost of every node in Γ^e_d has to be updated. If the node is a leaf that corresponds to working on activity j at period i during day d, its cost is updated with $c^e_{dij} - \beta_{dij}$; if the node is an and-node $A^{\pi,k}_{d1n} \in ch(O^S_{d1n})$ its cost is updated with $(-\lambda^e_d - \delta^e - w_{ds}\gamma^e - \alpha^e_{df_{ds}} - \sum\limits_{i=f_{ds}-\epsilon}^{f_{ds}} \alpha^e_{di})$; otherwise, the cost is zero.

4.3 Branching Rule

We use the branching rule suggested in Côté et al. [15], where at each node of the B&P algorithm, we select the employee e' with the highest fractional solution. For that employee, we select two daily shifts, $s^{e'}_d(1)$ and $s^{e'}_d(2)$, corresponding to the associated variables with the largest fractional values. Then, we identify the first divergent position i' between $s^{e'}_d(1)$ and $s^{e'}_d(2)$, meaning that both shifts differ in their activities (work-activity, rest or break). Let $j(1) \in J$ and $j(2) \in J$ be the assigned activities at period i' for shifts $s^{e'}_d(1)$ and $s^{e'}_d(2)$, respectively. As mentioned before, $J^{e'}_{di'}$ is the set of work activities that employee e' can perform at period i' during day d. A partition of $J^{e'}_{di'}$ into subsets $J^{e'}_{di'}(1)$ and $J^{e'}_{di'}(2)$ is created, such that $j(l) \in J^{e'}_{di'}(l)$, for $l = 1, 2$. The rest of the activities in $J^{e'}_{di'}$ are equally distributed between the two partitions. After generating the partitions, two nodes are created. At each node $l = 1, 2$ it is ensured that employee e' does not perform the activities in $J^{e'}_{di'}(l)$ at period i'. The rule is easily handled in the subproblem, since if an activity j is forbidden at period i during day d, the associated leaf O^j_{di1} receives a large cost in the DAG Γ^e_d ensuring that tours containing shifts with activity j at position i for day d will not be generated. Therefore, the suggested branching rule preserves the structure of the pricing subproblem.

5 B&P for the Tour-Based Formulation

In this section, we present a B&P algorithm for the Tour-based formulation. First, we introduce the restricted master problem. Second, we describe a two-phase procedure to solve the pricing subproblems. Finally, we present the branching rule used to identify integer solutions.

5.1 Restricted Master Problem

The complete enumeration of the set of feasible tours $\mathcal{T}^e$ for every employee is intractable due to the incorporation of shift and tour flexibility. Hence, we define a restricted master problem $F_{\widetilde{\mathcal{T}}}$ as the LP relaxation of problem (9)-(13) over restricted sets of tours $\widetilde{\mathcal{T}}^e \subseteq \mathcal{T}^e$.

Let β_{dij} and σ^e be the dual variables associated with constraints (10) and (11), respectively. The reduced cost of column (tour) t for employee e is given by:

$$\bar{c}_t^e = \sum_{d \in D} \sum_{i \in I_d} \sum_{j \in J_{di}^e} (c_{dij}^e - \beta_{dij}) \rho_{dijt}^e - \sigma^e \tag{16}$$

Expression (16) is used in the objective function of the pricing subproblems for every employee. Under a B&P algorithm, if a new column with negative reduced cost is found, the column is added to the restricted master problem $F_{\widetilde{\mathcal{T}}}$. If no negative reduced columns can be generated, the solution of $F_{\widetilde{\mathcal{T}}}$ is optimal for the node under evaluation.

5.2 Pricing Subproblems for Tour Generation

Tours are composed of a combination of daily shifts and days-off over a time horizon of at least one week. Considering the above, the approach presented in this section seeks to build tours with a two-phase process. In the first phase, daily shifts are generated with the same procedure introduced in Section 4.2, except for a change in the cost of every node in Γ_d^e. In this case, if the node is a leaf that corresponds to working on activity j at period i from day d, its cost is $c_{dij}^e - \beta_{dij}$, otherwise the cost corresponds to zero. In the second phase, the proposed method assembles shifts into tours by considering the constraints related with the tour length, the number of working days and the minimum resting time between consecutive shifts. Thus, in this second phase of the subproblem, the multi-activity property does not affect the assembling of feasible tours and the only attributes that must be considered are the shift starting time and shift length. The two-phase procedure is exact because, as it will be described further, at every iteration of the column generation, we include the shift with the lowest reduced cost for every start time and length at each day in the planning horizon.

The objective in the second phase of the subproblem is to generate tours with negative reduced cost for every employee. Taking this into consideration, let $\mathcal{K}^e = \bigcup_{d \in D} A_{d1n}^{\pi,k} \in ch(O_{d1n}^S)$ be the union over the set of days of all the children of the root node O_{d1n}^S of Γ_d^e. Therefore, $\mathcal{K}^e$ can be seen as the set of shift "shells" containing all the possible combinations of shift starting times and lengths but without considering breaks and activity allocation. Let $G^e(\mathcal{N}, \mathcal{A})$ be a directed acyclic graph with a set of nodes $\mathcal{N} = \{v_k \mid k \in \mathcal{K}^e \cup \{v_s, v_f\}\}$, where v_k corresponds to shift k for employee e, while v_s and v_f are the source and sink nodes, respectively. The set of arcs $\mathcal{A}$ is divided into three types: arcs going from the source node to a "shift" node $\mathcal{A}_1 = \{(v_s, v_k) \mid v_k \in \mathcal{N}, d_k \leq \Phi + 1\}$; arcs connecting two "shift" nodes $\mathcal{A}_2 = \{(v_k, v_{k'}) \mid v_k, v_{k'} \in \mathcal{N}, k \neq k^{'}, t_{k'} - f_k \geq \epsilon, d_{k'} - d_k \leq \Phi + 1\}$; and arcs connecting a "shift" node to the sink, $\mathcal{A}_3 = \{(v_k, v_f) \mid v_k \in \mathcal{N}, d_k \geq \Delta_l\}$.

Each node in graph $G^e(\mathcal{N}, \mathcal{A})$ has, besides a list of immediate successors $\mathcal{N}(v_k) = \{v_i \in \mathcal{N} | (k, i) \in \mathcal{A}\}$, a set of attributes (start period t_k, length l_k, considering breaks, working time w_k, end period $f_k = t_k + l_k - 1$, day d_k and reduced cost $\bar{c}_k$) inherited from its corresponding daily shift. The source node has a cost equal to zero, the sink node has a cost equal to the negative of the dual variable σ^e and the remaining nodes have a cost equal to $\bar{c}_k$. The list of successors of each node is generated according to the work rules for tour composition, employee preferences and availability, as expressed in the arc types definition. Thus, successors of source node v_s are nodes that, depending on their start day, allow enough time to meet the constraints related with the minimum number of working days required in a tour. In the same way, sink node v_f is a successor of node v_k if the finish day of v_k is greater or equal to the minimum number of working days required in a tour. Finally, a node $v_{k'}$ is a successor of node v_k if first, its start time allows to guarantee that there is a minimum rest time between both shifts and secondly, if its start day allows to meet the constraints related with the minimum and maximum number of days-off and their consecutiveness.

Tours are resource constrained paths along graph $G^e(\mathcal{N}, \mathcal{A})$. In this case, if the cost of a tour is negative, that means that a column with negative reduced cost was found and it deserves to be sent to the restricted master problem. It should be noted that every feasible path should meet the constraints related with the resources considered in the problem: the total tour length and the number of working days. Furthermore, the method is optimal, since graph $G^e(\mathcal{N}, \mathcal{A})$ contains the shifts with the lowest reduced cost for all the possible shift structures.

Figure 3 presents an example of graph $G^e(\mathcal{N}, \mathcal{A})$. In the graph, the number of days in the planning horizon is 7, the minimum and maximum number of working days are 5 and 6, respectively, the minimum and maximum tour length are 15 and 18, respectively, and shifts are built with the DAG Γ^e_d presented in Figure 1. Nodes with an odd number represent shifts obtained from and-node $A^{S \to RF,1}_{d15}$ that generates the structures $rwbww$, $rwwbw$, whereas nodes with an even number represent shifts obtained from and-node $A^{F \to FR,1}_{d15}$ with structures $wbwwr$, $wwbwr$. In this case, for every and-node there are sixteen possible shifts, where $rw_1bw_1w_1$, $rw_1bw_1w_2$,..., $rw_1w_2bw_2$,...,$rw_2w_2bw_2$ are some possible shifts for the first structure and $w_1bw_1w_1r$, $w_1bw_1w_2r$,..., $w_1w_2bw_2r$,...,$w_2w_2bw_2r$ are some possible shifts for the second. The shift considered at every node is the one with the minimum reduced cost.

For clarity, Figure 3 does not include all the possible arcs, but three examples of paths that consider every arc type are presented: a path with a total length of 18 working periods and one day-off (dashed path): $s - 1 - 3 - 5 - 8 - 10 - 12 - e$, a path with a total length of 15 periods and two days-off in a row (dotted path): $s - 2 - 4 - 6 - 8 - 13 - e$ and path with a total length of 15 periods and two nonconsecutive days-off (bold path): $s - 3 - 7 - 9 - 11 - 14 - e$.

Two processes are considered in the solution method for the tour composition: a graph preprocessing and the solution of a shortest path problem with resource constraints (SPPRC). The aim of the graph preprocessing is to reduce the size of the tour graph $G^e(\mathcal{N}, \mathcal{A})$ by deleting nodes that are similar to each other. Therefore, a node $v_k \in \mathcal{N}$ is deleted if it is dominated by a node $v_{k'} \in \mathcal{N}$, $k^{'} \neq k$. Considering the above, $v_{k'}$ dominates v_k if both nodes represent

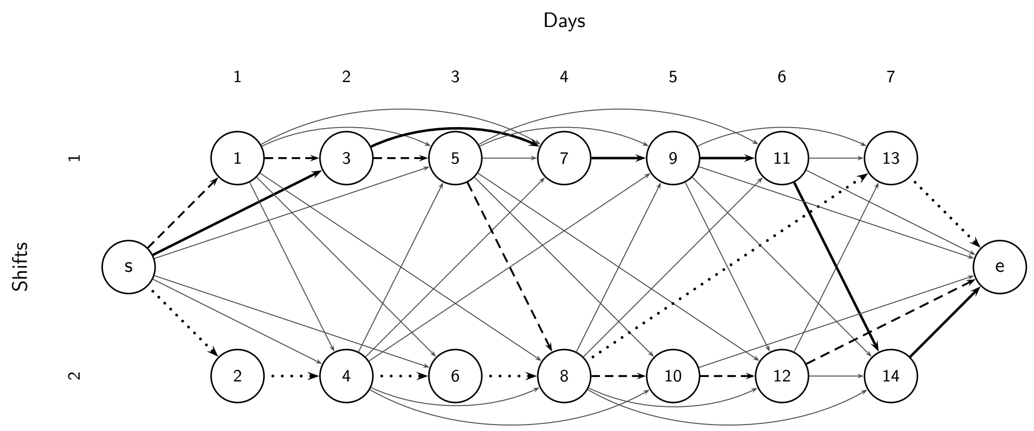


Figure 3: $G^e(\mathcal{N}, \mathcal{A})$ over a planning horizon of seven days.

shifts starting the same day, if both cover the same number of periods and if the cost of $v_{k'}$ is lower than the cost of v_k. Note that the preprocessing step allows to solve the pricing subproblems in a fast but heuristic way, since some nodes that could be in the optimal solution of the SPPRC are deleted. Therefore, to guarantee the optimality of the column generation approach, when no more columns with negative reduced cost can be found, we call the pricing subproblems once again, this time without performing the preprocessing step, and solve the SPPRC with the complete graph.

To solve the shortest path problem with resource constraints, we implemented a label setting algorithm where the total length of the tour and the number of working days are global resources that are consumed by the labels while they are extended. In order to avoid exploring paths that in some point will be infeasible or dominated, we included pruning strategies related with bounds, the consumption of global resources and the dominance property between labels. The pseudocode of the algorithm and its description are presented in Appendix B.

5.3 Branching Rule

Let e_1 and e_2 be the two employees with the highest fractional solutions at the node under evaluation. e_1 holds the first largest value and e_2 holds the second largest value, $x^{e_1}_{t(1)} \geq x^{e_2}_{t(1)}$. Let $x^{e_2}_{t(2)}$ be the second largest variable value for employee two, hence $x^{e_2}_{t(1)} \geq x^{e_2}_{t(2)}$. The branching rule used for the Tour-based formulation is an aggressive variable fixing strategy combined with the rule presented for the Daily-based formulation. In this way, at each node of the B&P tree, we choose employees e_1 and e_2, fix to one variable $x^{e_1}_{t(1)}$ for e_1 and select two tours, $t^{e_2}(1)$ and $t^{e_2}(2)$, for e_2 corresponding to variables $x^{e_2}_{t(1)}$ and $x^{e_2}_{t(2)}$ to apply the branching rule presented in Section 4.3.

6 Computational Experiments

In this section, we present the computational experiments to test the performance of the B&P algorithms. First, we generate random instances to test and compare the algorithms.

Second, we test the Tour-based B&P algorithm for the mono-activity tour scheduling problem introduced by Brunner and Bard [7].

6.1 Results on Random Instances

We perform the computational experiments on several random instances that are generated as follows. We start creating a set of feasible schedules for each employee to randomly choose one. From these schedules, we derive the associated demand profile along the planning horizon. Undercovering and overcovering of employee requirements are generated by randomly adding or removing demand. All instances are defined over a one-week planning horizon where days are divided into 96 periods of 15 minutes and shifts are not allowed to span from one day to another (discontinuous version of the problem). The number of working days in the tour should fall between 5 and 6 and the tour length should fall between 35 and 40 working hours per week. Activities have a minimum and maximum length that ranges between two and ∞ time periods, depending on the instance. Each tour has an associated cost corresponding to the sum of the costs of performing activity j at time interval i plus the sum of transition costs c_{tr} between activities. Instances are divided into three groups and are labeled with the format `E_A_D_V_G` where `E`, `A`, `D`, `V` and `G` represent the number of employees, number of activities, length of the planning horizon in days, version of the instance and group, respectively. The instances and the productions used in the grammars to create the shifts are described as follows, where: s_d and e_d are the first and last time intervals in day d with demand greater than zero, respectively; a_l and a_u are the minimum and maximum length of activity a, respectively; A_e is the set of skills of employee e, and artificial activity r represents either a break or a resting time.

G1: Instances with flexible start times and three types of shifts

In this group of instances, shifts may start at any time interval of the day allowing enough time to complete its length and three types of shifts are considered: 8-hour shifts with a 1-hour lunch break in the middle, 4-hour and 6-hour shifts without breaks.

$S_{[s_d,e_d]} \rightarrow RPR \mid RFR \mid RQR \mid PR \mid RP \mid FR \mid RF \mid QR \mid RQ;$

$P_{[16,16]} \overset{c_{tr}}{\rightarrow} J_j J_{\bar{j}} | J_j,\ \forall j \in A_e;$

$Q_{[24,24]} \overset{c_{tr}}{\rightarrow} J_j J_{\bar{j}} | J_j,\ \forall j \in A_e;$

$F \rightarrow PLP;$

$J_{a[a_l,a_u]} \rightarrow J'_a,\ \forall a \in A_e;$

$J'_a \rightarrow a | aJ'_a,\ \forall a \in A_e;$

$J_{\bar{j}} \rightarrow J_{j'},\ \forall j \in A_e,\ \forall j' \in A_e \backslash \{j\};$

$J_{\bar{j}} \overset{c_{tr}}{\rightarrow} J_{j'} J_{\bar{j}'},\ \forall j \in A_e,\ \forall j' \in A_e \backslash \{j\};$

$R \rightarrow r | rR;$

$L_{[4,4]} \rightarrow r | rR.$

G2: Instances with flexible start times and two types of shifts

In this group, we allow flexibility in terms of shift start time, meaning that shifts may start at any time interval of the day and two types of shifts are considered: 8-hour shifts with a 1-hour lunch break in the middle and 4-hour shifts without breaks.

$S_{[s_d,e_d]} \rightarrow RPR \mid RFR \mid PR \mid RP \mid FR \mid RF;$

$P_{[16,16]} \overset{c_{tr}}{\rightarrow} J_j J_{\bar{j}} | J_j,\ \forall j \in A_e;$

$F \rightarrow PLP$;
$J_{a[a_l,a_u]} \rightarrow J_a^{'}$, $\forall a \in A_e$;
$J_a^{'} \rightarrow a|aJ_a^{'}$, $\forall a \in A_e$;
$J_{\bar{j}} \rightarrow J_{j'}$, $\forall j \in A_e$, $\forall j^{'} \in A_e \backslash \{j\}$;
$J_{\bar{j}} \overset{c_{tr}}{\rightarrow} J_{j'} J_{\bar{j}'}$, $\forall j \in A_e$, $\forall j^{'} \in A_e \backslash \{j\}$;
$R \rightarrow r|rR$;
$L_{[4,4]} \rightarrow r|rR$.

G3: Instances with restricted start times and three types of shifts

In this third group of instances, shifts cannot start at any time interval, therefore a set of start times is considered. Three types of shifts are generated: 8-hour shifts with a 1-hour lunch break in the middle, 4-hour and 6-hour shifts without breaks. The start times are 1, 17, 33, 49 and 61.
$S_{[36,36]} \rightarrow PR \,|\, QR \,|\, FR$;
$P_{[16,16]} \overset{c_{tr}}{\rightarrow} J_j J_{\bar{j}}|J_j$, $\forall j \in A_e$;
$Q_{[24,24]} \overset{c_{tr}}{\rightarrow} J_j J_{\bar{j}}|J_j$, $\forall j \in A_e$;
$F \rightarrow PLP$;
$J_{a[a_l,a_u]} \rightarrow J_a^{'}$, $\forall a \in A_e$;
$J_a^{'} \rightarrow a|aJ_a^{'}$, $\forall a \in A_e$;
$J_{\bar{j}} \rightarrow J_{j'}$, $\forall j \in A_e$, $\forall j^{'} \in A_e \backslash \{j\}$;
$J_{\bar{j}} \overset{c_{tr}}{\rightarrow} J_{j'} J_{\bar{j}'}$, $\forall j \in A_e$, $\forall j^{'} \in A_e \backslash \{j\}$;
$R \rightarrow r|rR$;
$L_{[4,4]} \rightarrow r|rR$.

Table 1 presents the average size of the graphs for the instances with flexible and restricted start times. The name of the instance is presented in Column 1. Column 2 shows the average number of nodes in the DAG Γ for each employee at every day in the planning horizon. Columns 3 and 4 present the average number of nodes and arcs in graph $G^e(\mathcal{N}, \mathcal{A})$.

The computational experiments were performed on a 64-bit GNU/Linux operating system, 96 GB of RAM and 1 processor Intel Xeon X5675 running at 3.07GHz. The B&P algorithms were implemented in C++ using the object-oriented branch-and-bound library (OOBB) developed by Crainic et al. [16]. The RMP was solved by using the barrier method of CPLEX version 12.5.0.1.

6.1.1 Solution at the Root Node.

Tables 2 - 3 show the computational effort, at the root node, for the B&P algorithms for the proposed formulations: the Daily-based formulation ($F_{\widetilde{\mathcal{S}}}$) and the Tour-based formulation ($F_{\widetilde{\mathcal{T}}}$) for instances with flexible and restricted start times, respectively. We report the value (LP val.) and the required time in seconds (CPU time), to solve the LP relaxation of the RMP. The difference between the lower bound of $F_{\widetilde{\mathcal{T}}}$ against the lower bound of $F_{\widetilde{\mathcal{S}}}$ is presented in Column 6.

According to the results, $F_{\widetilde{\mathcal{T}}}$ achieved better lower bounds for all the 21 instances when compared with $F_{\widetilde{\mathcal{S}}}$. It is worth noting that the difference between lower bounds tends to increase as the problem flexibility increases. This can be seen in the results where the average LB difference for instances in G1-G2 (flexible start time) is 3.79%, while the average LB

Instance	Nodes DAG Γ	Nodes $G(\mathcal{N},\mathcal{A})$	Arcs $G(\mathcal{N},\mathcal{A})$
20_1_7_1_G1	6,980	998	291,606
20_1_7_2_G1	11,899	1,496	627,300
25_1_7_1_G1	7,006	1,001	292,689
25_1_7_2_G1	11,998	1,505	633,753
40_1_7_1_G1	7,006	1001	292,689
40_1_7_2_G1	11,998	1,505	633,753
20_3_7_1_G2	24,802	652	124,072
20_3_7_2_G2	19,420	983	270,476
20_3_7_3_G2	19,147	972	265,352
20_1_7_1_G3	300	63	1,215
20_1_7_2_G3	300	96	2,700
25_1_7_1_G3	300	63	1,215
25_1_7_2_G3	300	105	3,141
40_1_7_1_G3	300	63	1,215
40_1_7_2_G3	300	105	3,141
20_3_7_1_G3	3,918	63	1,215
20_3_7_2_G3	1,851	101	2,906
20_3_7_3_G3	1,851	99	2,802
20_5_7_1_G3	1,131	53	1,127
20_5_7_2_G3	1,154	51	1,007
20_5_7_3_G3	1,379	84	2,285

Table 1: Average number of nodes and arcs in the graphs for instances with flexible and restricted start times.

difference for instances in G3 (restricted start time) is 1.77%. Regarding the computational time to solve the LP relaxation at the root node, the Daily-based formulation shows a better performance in 15 out of 21 instances and, as expected, results also show that the average computational time of both formulations increases as the number of activities or employees grow.

The reason for the differences in the time performance for both formulations is mainly due to the structure of the pricing subproblems. Subproblems of $F_{\widetilde{S}}$ only deal with the composition of daily shifts, while subproblems in $F_{\widetilde{T}}$ handle the composition of daily shifts and the assembling of tours. However, as it will be shown in the computational results for the B&P, it is worth to invest more time in the solution of the root node if the lower bounds obtained are better, especially when the model exhibits symmetry that makes difficult to prove optimality due to many equivalent solutions.

	Tour-based		Daily-based		
Instance	LP val.	CPU time	LP val.	CPU time	LB Dif.
20_1_7_1_G1	52,080	**21.02**	50,080	24.72	3.84%
20_1_7_2_G1	49,440	30.84	47,440	**18.20**	4.05%
25_1_7_1_G1	60,560	**16.10**	58,060	33.47	4.13%
25_1_7_2_G1	72,660	**6.03**	70,160	32.51	3.44%
40_1_7_1_G1	100,410	**10.52**	96,410	52.64	3.98%
40_1_7_2_G1	98,390	89.17	94,390	**52.95**	4.07%
20_3_7_1_G2	55,270	185.11	53,195	**70.20**	3.75%
20_3_7_2_G2	60,120	133.69	58,045	**54.36**	3.45%
20_3_7_3_G2	60,450	97.80	58,375	**51.51**	3.43%
Average	-	65.60	-	49.40	3.79%

Table 2: Computational effort to solve the LP relaxation at the root node, for instances with flexible start times.

	Tour-based		Daily-based		
Instance	LP val.	CPU time	LP val.	CPU time	LB Dif.
20_1_7_1_G3	82,825	3.59	82,145	**0.68**	0.82%
20_1_7_2_G3	64,675	4.4	63,775	**1.79**	1.39%
25_1_7_1_G3	99,995	2.83	98,732.5	**0.91**	1.26%
25_1_7_2_G3	72,660	**1.31**	70,160	3.03	3.44%
40_1_7_1_G3	202,050	**1.35**	198,050	1.99	1.98%
40_1_7_2_G3	117,677	9.49	116,280	**6.84**	1.19%
20_3_7_1_G3	73,440	61.05	71,440	**7.69**	2.72%
20_3_7_2_G3	61,655.5	130.6	59,635.6	**17.32**	3.28%
20_3_7_3_G3	63,575	86.61	61,560	**15.89**	3.17%
20_5_7_1_G3	95,078.3	509.53	94,865.6	**8.44**	0.22%
20_5_7_2_G3	93,962.2	156.92	93,280.1	**10.25**	0.73%
20_5_7_3_G3	131,824	126.63	130,432	**12.22**	1.06%
Average	-	91.2	-	7.25	1.77%

Table 3: Computational effort to solve the LP relaxation at the root node, for instances with restricted start times.

A heuristic approach in which we solve the IP version of the problem at the root node was tested. Tables 4 - 5 present the best upper bound found (IP RMP) and the computational time required (CPU time) to find it, for the Tour-based formulation and the Daily-based formulation on instances with flexible start time and restricted start time, respectively. The value of the integrality gap for both formulations is computed as: $\frac{\text{(IP RMP - LP val.)}}{\text{IP RMP}}$ and reported in Columns 4 and 7. We set a time limit of 600 sec. or a relative MIP gap tolerance of less than 1% to solve the problem when the integrality constraints are considered.

	Tour-based			Daily-based		
Instance	IP RMP	CPU time	Gap	IP RMP	CPU time	Gap
20_1_7_1_G1	**52,080**	**0.29**	0.00%	52,280	10.51	4.21%
20_1_7_2_G1	49,660	**0.16**	0.44%	**49,640**	100.77	4.43%
25_1_7_1_G1	**60,560**	**0.59**	0.00%	60,760	38.09	4.44%
25_1_7_2_G1	**72,660**	**0.29**	0.00%	73,100	27.36	4.02%
40_1_7_1_G1	100,850	**0.25**	0.44%	**100,810**	49.89	4.36%
40_1_7_2_G1	**98,940**	**1.35**	0.56%	98,990	551.02	4.65%
20_3_7_1_G2	57,625	600	4.09%	**55,380**	89.97	3.94%
20_3_7_2_G2	62,350	600	3.58%	**60,450**	264.32	3.98%
20_3_7_3_G2	62,050	600	2.58%	**60,700**	116.32	3.83%
Average	-	200.33	1.30%	-	138.69	4.21%

Table 4: Computational effort to find an upper bound at the root node, for instances with flexible start times.

	Tour-based			Daily-based		
Instance	IP RMP	CPU time	Gap	IP RMP	CPU time	Gap
20_1_7_1_G3	**82,860**	**0.07**	0.04%	83,280	0.65	1.36%
20_1_7_2_G3	65,280	**1.51**	0.93%	**65,040**	9.18	1.94%
25_1_7_1_G3	**100,380**	**0.64**	0.38%	100,650	3.46	1.90%
25_1_7_2_G3	73,100	**0.3**	0.60%	**72,860**	2.02	3.71%
40_1_7_1_G3	**202,250**	**0.03**	0.10%	203,450	1.45	2.65%
40_1_7_2_G3	118,170	**0.97**	0.42%	118,170	13.50	1.60%
20_3_7_1_G3	76,030	600	3.41%	**74,070**	7.16	3.55%
20_3_7_2_G3	65,445	600	5.79%	**62,245**	45.20	4.19%
20_3_7_3_G3	67,330	600	5.58%	**64,285**	59.45	4.24%
20_5_7_1_G3	102,170	600	6.94%	**96,035**	9.30	1.22%
20_5_7_2_G3	102,860	600	8.65%	**94,690**	11.40	1.49%
20_5_7_3_G3	137,135	600	3.87%	**133,430**	55.74	2.25%
Average	-	300.29	3.06%	-	18.21	2.51%

Table 5: Computational effort to find an upper bound at the root node, for instances with restricted start times.

According to the results, the Tour-based formulation achieved better solution times for all the 12 mono-activity instances when compared to the Daily-based formulation. On the other hand, the Daily-based formulation had a better performance as the number of activities grow. Observe that if feasible integer solutions are required for multi-activity problems, avoiding the B&P process and using only the variables generated at the root node with the Daily-based formulation could be a good option. On the contrary, as shown by the results of the next section, if the objective is to solve the problems to optimality, a B&P algorithm should be used taking advantage of the formulation with the best lower bound, in this case the Tour-based formulation.

6.1.2 Branch-and-Price.

Tables 6 - 7 show the results of the B&P algorithm for the instances with flexible start time and restricted start time, respectively, with both formulations, the Tour-based formulation and the Daily-based formulation. Columns 2 and 6 present the value of the best integer solution found. Columns 3 and 7 report the total computational time required to find the integer solutions, while Columns 4 and 8 contain the number of nodes explored in the B&P trees. Finally, Columns 5 and 9 show the value of the integrality gap (defined as the percentage difference between the best upper bound minus the best lower bound). The algorithm stops when the gap is less than 1% or when the total time reaches one hour. The search strategy used was a depth-first search where the upper bound of the algorithm was initialized with the heuristic integer solution found at the root node.

For the Tour-based formulation, our method was able to find high-quality integer solutions for all the instances in the three groups. All the mono-activity instances were closed with the heuristic at the root node in less than two minutes. On the contrary, for the multi-activity instances, it was necessary to explore several nodes in the B&P tree to find integer solutions with an optimality gap less than 1%. Note that, as can be seen in the results for instances with five activities, the computational time and the number of nodes explored in the B&P tree increase with the number of activities. The average optimality gap for instances in G3 is the highest among the three groups, with a value of 0.56%.

On the other hand, the B&P algorithm implemented for the Daily-based formulation did not have a good performance when tested with the three groups of instances. The algorithm was not able to improve, within one hour time limit, the integer solution found at the root node, and when compared with the Tour-based formulation, it only achieved better integer solutions in 2 out of 21 instances. As a result, if the objective is to find near optimal solutions, the Tour-based formulation could be considered stronger because it exhibits a better LP relaxation bound that makes easier to close the optimality gap.

	Tour-based				Daily-based			
Instance	IP val.	CPU time	Nb. Nodes	Gap	IP val.	CPU time	Nb. Nodes	Gap
20_1_7_1_G1	**52,080**	21.31	1	0.00%	52,280	3,600	777	4.21%
20_1_7_2_G1	**49,440**	31	1	0.44%	49,640	3,600	229	4.43%
25_1_7_1_G1	**60,560**	16.69	1	0.00%	60,760	3,600	369	4.44%
25_1_7_2_G1	**72,660**	6.32	1	0.00%	73,100	3,600	545	4.02%
40_1_7_1_G1	**100,520**	10.77	1	0.44%	100,810	3,600	217	4.36%
40_1_7_2_G1	**98,390**	90.52	1	0.56%	98,990	3,600	57	4.65%
20_3_7_1_G2	55,380	2366.81	39	0.20%	55,380	3,600	51	3.95%
20_3_7_2_G2	**60,120**	1735.07	35	0.00%	60,450	3,600	159	3.98%
20_3_7_3_G2	60,780	1507.83	39	0.54%	**60,700**	3,600	33	3.83%
Average	-	643	13	0.24%	-	3,600	271	4.20%

Table 6: Computational effort in the B&P algorithm for instances with flexible start times.

	Tour-based				Daily-based			
Instance	IP val.	CPU time	Nb. Nodes	Gap	IP val.	CPU time	Nb. Nodes	Gap
20_1_7_1_G3	**82,860**	3.66	1	0.04%	83,280	3,600	615	1.36%
20_1_7_2_G3	**64,730**	5.91	1	0.93%	65,040	3,600	677	1.94%
25_1_7_1_G3	**100,250**	3.47	1	0.38%	100,650	3,600	703	1.9%
25_1_7_2_G3	**72,660**	1.61	1	0.6%	72,860	3,600	1055	3.71%
40_1_7_1_G3	**202,050**	1.38	1	0.1%	203,450	3,600	971	2.65%
40_1_7_2_G3	**117,730**	10.46	1	0.42%	118,170	3,600	1393	1.6%
20_3_7_1_G3	**73,625**	1,070.66	39	0.25%	74,070	3,600	1113	3.55%
20_3_7_2_G3	**62,125**	1,408.06	37	0.76%	62,245	3,600	405	4.19%
20_3_7_3_G3	**63,950**	1,370.42	37	0.59%	64,285	3,600	953	4.24%
20_5_7_1_G3	**95,890**	2,033.89	41	0.85%	96,035	3,600	465	1.22%
20_5_7_2_G3	94,800	2,256.76	127	0.88%	**94,690**	3,600	431	1.49%
20_5_7_3_G3	**133,035**	1,929.55	91	0.91%	133,430	3,600	410	2.25%
Average	-	841.32	32	0.56%	-	3,600	766	2.51%

Table 7: Computational effort in the B&P algorithm for instances with restricted start times.

6.2 Results on Instances From Brunner and Bard [7]

This mono-activity problem consists of a discontinuous tour scheduling problem over one week with two types of employees: full time workers (Reg) and part time workers (Flex). In the problem, each day is divided into 48 time intervals of 30 minutes each. Each Reg employee must be given an 8.5-hour shift on each working day and the starting time of shifts can vary by day (nine start times). In contrast, Flex employees can be assigned to one of five shift types ranging from 4 to 8.5 hours, which can have different starting times (12 start times). If an employee works 6 hours or more a day, its assigned shift must have a 0.5-hour lunch break.

In Brunner and Bard [7], the master problem is based on a set covering model that seeks to minimize the total cost of the tours plus the cost of undercovering of employee requirements. The model guarantees that the total number of employees that are on duty cover the requirements for each period during the time horizon, and that the ratio between the number of Reg employees and the number of Flex employees is at least a given value.

Table 8 presents the description of the eleven scenarios analyzed. Column 1 gives the name of the instance. Column 2 presents the value of the ratio between Reg and Flex employees. Columns 3 and 4 specify the types of employees having shifts with flexible length and start time, respectively. Columns 5 and 6 show if shifts have a break and if two days off in a row must be considered in the tour, respectively. The full description of the parameters to build daily shifts and weekly tours, as well as, the employee requirements are presented in the appendix of the work in Brunner and Bard [7].

Table 9 shows the output statistics for the Tour-based formulation and the model presented in Brunner and Bard [7], now called Brunner's model. Since the number of employees is not known, the Daily-based formulation cannot be used in the context of this problem because tours are composed for each employee through constraints (3)-(6). In the same way, the proposed B&P algorithm cannot be used in the context of this problem, because the branch-

Instance	Ratio	Flex_length	Flex_start	Breaks	2 Days off in a row
B1	4	Flex	No	Yes	No
B2	4	No	Flex	Yes	No
B3	4	Flex	Flex	Yes	No
B4	4	Flex	Reg,Flex	Yes	No
B5	4	No	Flex	No	No
B6	4	Flex	Flex	No	No
B7	4	Flex	Reg,Flex	No	No
B8	1	Flex	Reg,Flex	No	No
B9	2	Flex	Reg,Flex	No	No
B10	3	Flex	Reg,Flex	No	No
B11	5	Flex	Reg,Flex	No	No

Table 8: Description of Brunner and Bard [7] problem's scenarios.

ing rule is based on employee branching. That is why we decided to solve the Tour-based formulation with the root node heuristic presented in Section 6.1.1, where we observed a good performance on mono-activity instances (average optimality gap of 0.33% in less than 2 sec. of computational time). Columns 2 and 4 present the IP value at the root node for the eleven instances evaluated with the Tour-based formulation and Brunner's model, respectively. As it was done in Brunner and Bard [7] we set a time limit of 300 sec. to solve the IP at the root node. Columns 3 and 5 show the total CPU time in sec. to solve the LP relaxation and obtain the IP value for both formulations. Finally, the value of the best IP solution found with the B&P algorithm presented in Brunner and Bard [7] and the total computational time in seconds to obtain that value are reported in Columns 6 and 7, respectively.

	Tour-based		Brunner's model			
Instance	IP val.	CPU time	IP val.	CPU time	Best IP val.	Total time
B1	95,944	**303.95**	95,864	488.06	95,640	2,882.39
B2	95,120⋆	**304.34**	95,640	496.62	95,120	5,743.36
B3	95,056⋆	**331.24**	95,832	492.20	95,120	14,428.84
B4	94,960⋆	**341.52**	95,920	406.75	95,456	6,293.98
B5	93,442.5	365.60	93,402.5	348.35	93,282.5	2,514.85
B6	93,290.5⋆	342.93	93,426.5	322.10	93,322.5	1,909.29
B7	93,266.5	**304.50**	93,298.5	322.93	93,178.5	1,871.9
B8	83,501⋆	**323.80**	84,421	331.44	84,061	1,853.11
B9	88,295⋆	**340.10**	88,727	342.46	88,439	1,919.47
B10	90,967.5⋆	344.34	91,135.5	335.27	90,999.5	1,891.51
B11	94,812.5	321.32	94,964.5	316.10	94,796.5	1,860.59
Average	-	329.42	-	382.02	-	3,924.48

Table 9: Computational effort at the root node to solve the integer problem.

From Table 9, we can conclude that the Tour-based formulation shows competitive solution times when compared with Brunner's model. Regarding the quality of the solution (IP value obtained) the star ($\star$) in Column 2 means that our model was able to achieve, in less than 6 minutes, the same or a better integer solution than the one reported as the best in Brunner and Bard [7], where the computational time to find this value was more than 30 minutes for all the instances (values reported in Column 7). Regarding the problem difficulty, Brunner and Bard [7] found that the instances with a lunch break were the most difficult to solve (instances B1 to B4) and that the time tended to decrease when more flexibility was introduced in the model. On the contrary, we did not observe any significant change in the execution time when considering either more flexibility or breaks.

7 Concluding Remarks

In this paper we introduced two B&P algorithms to solve the personalized multi-activity tour scheduling problem. Two formulations were presented in which the master problem is modeled as a generalized set partitioning problem. With respect to the pricing subproblems, in the Daily-based formulation, columns (daily shifts) are modeled using context-free grammars. In the Tour-based formulation, columns (tours) are built with an exact two-phase procedure. In the first phase, daily shifts are modeled by using context-free grammars, where in the second phase, the daily shifts are assembled into tours by using a shortest path algorithm with resource constraints.

Although our computational experiments suggest that the Daily-based formulation finds solutions for the LP relaxation at the root node in a shorter execution time when compared with the Tour-based formulation, we show that the second formulation is stronger in terms of its LP relaxation lower bound.

Two methods were tested to find integer solutions. A heuristic approach in which we impose the integrality constraints at the root node and an exact approach corresponding to a B&P. The Daily-based formulation exhibited better solution times than the Tour-based formulation for the heuristic approach. On the other hand, the Tour-based formulation had a better performance in the exact approach being able to find, within 1 hour, integer solutions for all the instances with an optimality gap lower than 1%. We also tested the Tour-based formulation on a mono-activity problem presented in Brunner and Bard [7]. The experiments suggested that the solution times and quality of our formulation are comparable with the solution times and quality reported by Brunner and Bard [7].

Despite the ability of our models to handle complex work rules, convergence and scalability issues arise when the number of employees and activities increase. One solution to this problem could be to implement an implicit model in order to avoid the dimension associated with the number of employees. Another option might be to implement new branching strategies related with the shift length in order to reduce the number of nodes explored in the B&P tree and reach integrality in a shorter time.

References

[1] Al-Yakoob, S. M., H. D. Sherali. 2008. A column generation approach for an employee scheduling problem with multiple shifts and work locations. *Journal of the Operational*

Research Society **59** 34–43.

[2] Aykin, T. 1996. Optimal shift scheduling with multiple break windows. *Management Science* **42** 591–602.

[3] Bailey, J. 1985. Integrated days off and shift personnel scheduling. *Computers & Industrial Engineering* **9** 395–404.

[4] Baker, K. R. 1976. Workforce allocation in cyclical scheduling problems: A survey. *Operational Research Quarterly* **27** 155–167.

[5] Bechtold, S. E., L. W. Jacobs. 1990. Implicit modeling of flexible break assignments in optimal shift scheduling. *Management Science* **11** 1339–1351.

[6] Boyer, V., B. Gendron, L.-M. Rousseau. 2012. A branch-and-price algorithm for the multi-activity multi-task shift scheduling problem. *Journal of Scheduling* **17** 185–197.

[7] Brunner, J. O., J. F. Bard. 2013. Flexible weekly tour scheduling for postal service workers using a branch and price. *Journal of Scheduling* **16** 129–149.

[8] Brunner, J. O, R. Stolletz. 2014. Stabilized branch and price with dynamic parameter updating for discontinuous tour scheduling. *Computers & Operations Research* **44** 137–145.

[9] Brusco, M. J., L. W. Jacobs. 2000. Optimal models for meal-break and start-time flexibility in continuous tour scheduling. *Management Science* **46** 1630–1641.

[10] Brusco, M. J., T. R. Johns. 2011. An integrated approach to shift-starting time selection and tour-schedule construction. *Journal of the Operational Research Society* **62** 1357–1364.

[11] Çezik, T., O. Günlük, H. Luss. 2001. An integer programming model for the weekly tour scheduling problem. *Naval Research Logistics* **48** 607–624.

[12] Côté, M.-C., B. Gendron, C.-G. Quimper, L.-M. Rousseau. 2011a. Formal languages for integer programming modeling of shift scheduling problems. *Constraints* **16** 54–76.

[13] Côté, M.-C., B. Gendron, L.-M. Rousseau. 2007. Modeling the regular constraint with integer programming. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, 29–43.

[14] Côté, M.-C., B. Gendron, L.-M. Rousseau. 2011b. Grammar-based integer programming models for multiactivity shift scheduling. *Management Science* **57** 151–163.

[15] Côté, M.-C., B. Gendron, L.-M. Rousseau. 2013. Grammar-based column generation for personalized multi-activity shift scheduling. *INFORMS Journal on computing* **25** 461–474.

[16] Crainic, T. G., A. Frangioni, Gendron B. 2009. OOBB: An object-oriented library for paralel branch-and-bound. *CORS/INFORMS International Conference, Toronto, Canada.*

[17] Dahmen, S., M. Rekik. 2012. Solving multiactivity personalized shift scheduling problems with a hybrid heuristic. Tech. rep., CIRRELT.

[18] Dantzig, G. B. 1954. A comment on Edie's "traffic delays at toll booths". *Journal of the Operations Research Society of America* **2** 339–341.

[19] Demassey, S., G. Pesant, L.-M. Rousseau. 2006. A cost-regular based hybrid column generation approach. *Constraints* **11** 315–333.

[20] Detienne, B., L. Péridy, É. Pinson, D. Rivreau. 2009. Cut generation for an employee timetabling problem. *European Journal of Operational Research* **197** 1178–1184.

[21] Elahipanah, M., G. Desaulniers, E. Lacasse-Guay. 2013. A two-phase mathematical-programming heuristic for flexible assignment of activities and tasks to work shifts. *Journal of Scheduling* **16** 443–460.

[22] Geoffrion, A. M. 1974. *Lagrangean relaxation for integer programming*. Springer.

[23] Jacobs, L. W., M. J. Brusco. 1996. Overlapping start-time bands in implicit tour scheduling. *Management Science* **42** 1247–1259.

[24] Jarrah, A. I. Z., J. F. Bard, A. H. deSilva. 1994. Solving large-scale tour scheduling problems. *Management Science* **40** 1124–1144.

[25] Lequy, Q., M. Bouchard, G. Desaulniers, F. Soumis, B. Tachefine. 2012. Assigning multiple activities to work shifts. *Journal of Scheduling* **15** 239–251.

[26] Mehrotra, A., K. E. Murphy, M. A. Trick. 2000. Optimal shift scheduling: A branch-and-price approach. *Naval Research Logistics* **47** 185–200.

[27] Moondra, S. 1976. A linear programming model for work force scheduling for banks. *Journal of Bank Research* **6** 299–301.

[28] Morris, J. G., M. J. Showalter. 1983. Simple approaches to shift, days-off and tour scheduling problems. *Management Science* **29** 942–950.

[29] Ni, H., H. Abeledo. 2007. A branch-and-price approach for large-scale employee tour scheduling problems. *Annals of Operations Research* **155** 167–176.

[30] Quimper, C.-G., L.-M. Rousseau. 2010. A large neighbourhood search approach to the multi-activity shift scheduling problem. *Journal of Heuristics* **16** 373–392.

[31] Quimper, Claude-Guy, Toby Walsh. 2007. Decomposing global grammar constraints. *Principles and Practice of Constraint Programming–CP 2007*. Springer, 590–604.

[32] Rekik, M., J.-F. Cordeau, F. Soumis. 2004. Using Benders decomposition to implicitly model tour scheduling. *Annals of Operations Research* **128** 113–133.

[33] Rekik, M., J.-F. Cordeau, F. Soumis. 2010. Implicit shift scheduling with multiple breaks and work stretch duration restrictions. *Journal of Scheduling* **13** 49–75.

[34] Restrepo, M. I., L. Lozano, A. L. Medaglia. 2012. Constrained network-based column generation for the multi-activity shift scheduling problem. *International Journal of Production Economics* **140** 466–472.

[35] Ritzman, L. P., L. J. Krajewsky, M. J. Showalter. 1976. The disaggregation of aggregate manpower plans. *Management Science* **22** 1204–1214.

[36] Van den Bergh, J., J. Beliën, P. De Bruecker, E. Demeulemeester, L. De Boeck. 2012. Personnel scheduling: A literature review. *European Journal of Operational Research* **226** 367–385.

Appendix A 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 10 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.

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 10: 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 4 presents the value of the objective function for the evaluated points, as well as the optimal solution $f(\overline{F^{\star}_{\mathcal{S}}}) = 16$ with $x^{\star}_{11} = x^{\star}_{21} = x^{\star}_{31} = 4/9$.

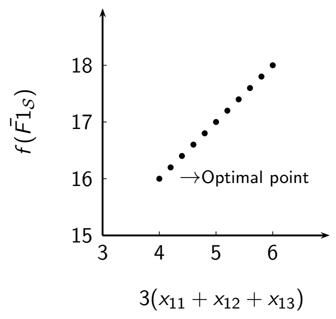


Figure 4: Optimal solution for the LP relaxation of the example using the Daily-based formulation.

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 $f(\overline{F^{\star}_{\mathcal{T}}})$ is 18. Hence $f(\overline{F^{\star}_{\mathcal{S}}}) < f(\overline{F^{\star}_{\mathcal{T}}})$.

Appendix B 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. 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 $\vec{\mathcal{P}^e}$ with negative reduced cost. Line 1 returns an initial set of labels (partial paths from v_s to all its successors). Line 2 selects the first label to be processed according to its 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. 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 successors. A new label l_2 is created and stored in $\mathcal{Q}$ (Line 9) if it is feasible (Line 6), non-dominated (Line 7) and its resident node 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 is stored in $\vec{\mathcal{P}^e}$ (Line 11). The feasibility function checks if the label to be created 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 to be created with the rest of the labels in $\mathcal{Q}$. 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, $l^{'}$ dominates l. 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 $v_i \in \mathcal{N}(v_k)$ 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:                         $\vec{\mathcal{P}} \leftarrow \mathcal{P}$, $t = t + 1$
12:     remove $l_1$ from $\mathcal{Q}$
13:     selectLabel
14: return $\vec{\mathcal{P}}^e$
```

Algorithm 1: Label setting algorithm to solve the SPPRC