CIRRELT

Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

Grammar-Based Integer Programming Models for Multi-Activity Shift Scheduling

Marie-Claude Côté
Bernard Gendron
Louis-Martin Rousseau

Janvier 2010

CIRRELT-2010-01

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · UQÀM · HEC Montréal · École Polytechnique Montréal · Université de Montréal

Grammar-Based Integer Programming Models for Multi-Activity Shift Scheduling

Marie-Claude Côté[1,2], Bernard Gendron[1,3,*], Louis-Martin Rousseau[1,2]

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

[2] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, C.P. 6079, succursale Centre-ville, Montréal, Canada H3C 3A7

[3] Department of Computer Science and Operations Research, Université de Montréal, C.P. 6128, succursale Centre-Ville, Montréal, Canada H3C 3J7

Abstract. This paper presents a new implicit formulation for shift scheduling problems, using context-free grammars to model regulation in the composition of shifts. From the grammar, we generate an integer programming (IP) model having a linear programming (LP) relaxation equivalent to that of Dantzig set covering model. When solved by a state-of-the-art IP solver on problem instances with a small number of shifts, our model, the set covering formulation and a typical implicit model from the literature yield comparable solution times. On instances with a large number of shifts, our formulation shows superior performance and can model a wider variety of constraints. In particular, multi-activity cases, which cannot be modeled by existing implicit formulations, can easily be handled with grammars. We present comparative experimental results on a large set of instances involving one work-activity and we experimentally demonstrate the interest of our modeling approach on problems dealing with up to ten work-activities.

Keywords. Shift scheduling, implicit models, mixed integer programming, context-free grammars.

Acknowledgements. This work was supported by a grant from the Fonds de recherche sur la nature et les technologies (FQRNT). We would like to thank Claude-Guy Quimper for his useful comments on our work.

---

* Corresponding author: Bernard.Gendron@cirrelt.ca

Dépôt légal – Bibliothèque nationale du Québec,
Bibliothèque nationale du Canada, 2010

1 Introduction

In this paper, we consider *shift scheduling problems* defined over a planning horizon of one day, divided into multiple periods. In this context, a shift is defined by its starting and ending times and by the activities, work tasks or breaks, to be performed at each period. The assignment of activities to a shift is constrained by different rules mainly arising from work regulation agreements and ergonomic considerations.

We define the *multi-activity shift scheduling problem* as follows. Given a planning horizon I divided into periods of equal length, a set of activities J, the set of all feasible shifts Ω, and the number of employees b_{ij} required at each period $i \in I$ for each activity $j \in J$, one must select from Ω a subset that covers the required number of employees at minimum cost, given that each feasible shift $s \in \Omega$ has an associated cost $c_s \geq 0$. The following integer programming (IP) model, denoted D, extends in a straightforward manner the original set covering formulation proposed in Dantzig (1954) for the shift scheduling problem first described in Edie (1954). This IP model explicitly enumerates all feasible shifts and therefore, is often called the *explicit model*.

$$f(D) = \min \sum_{s \in \Omega} c_s x_s$$

$$\sum_{s \in \Omega} \delta_{ijs} x_s \geq b_{ij}, \qquad \forall i \in I, j \in J, \tag{1}$$

$$x_s \geq 0 \text{ and integer}, \qquad \forall s \in \Omega, \tag{2}$$

where $\delta_{ijs} = 1$ if activity $j \in J$ is assigned to period $i \in I$ in shift $s \in \Omega$, and variable x_s gives the number of employees assigned to shift $s \in \Omega$. We will assume that the cost of each feasible shift $s \in \Omega$ can be decomposed by employee and by activity as follows: $c_s = \sum_{i \in I} \sum_{j \in J} \delta_{ijs} c_{ij}$, where $c_{ij} \geq 0$, for each $i \in I$ and $j \in J$.

In practice, such explicit models can only be solved when Ω is relatively small, or else, by using column generation approaches as in Demassey et al. (2006), Mehrotra et al. (2000). In this paper, we present a new formulation based on assignment variables y_{ij} indicating the number of employees assigned to activity $j \in J$ at period $i \in I$. These variables are related to the variables in the explicit model by the simple equations $y_{ij} = \sum_{s \in \Omega} \delta_{ijs} x_s$. Our formulation belongs to the class of *implicit models* which do not use the set Ω and rather represent implicitly all feasible shifts by using linear inequalities defined over nonnegative integer variables. More precisely, the set $\{x_s \geq 0$ and integer$|y_{ij} = \sum_{s\in\Omega} \delta_{ijs}x_s,\ i \in I,\ j \in J\}$ is described equivalently by an additional set of integer variables $v \geq 0$ such that $(y, v) \in H$, where H is a bounded polyhedron. The implicit model that we propose, denoted Q, has therefore the following form:

$$f(Q) = \min \sum_{i\in I} \sum_{j\in J} c_{ij}y_{ij}$$

$$y_{ij} \geq b_{ij} \text{ and integer}, \qquad \forall i \in I, j \in J, \tag{3}$$

$$(y, v) \in H, \tag{4}$$

$$v \geq 0 \text{ and integer}. \tag{5}$$

Côté et al. (2009), Côté et al. (2007) exploit automata and context-free grammars to formulate similar IP models that represent all feasible shifts for any single employee. However, when the number of employees or activities increase, these implicit models do not scale well as performance degrades rapidly. In this paper, assuming that all employees are identical, we show how to derive grammar-based models with tractable size, allowing us to handle large-scale problems with multiple activities. Assuming that any feasible shift can be represented by a word in a context-free language, we will show how to derive polyhedron H from the context-free grammar G defining the language. Moreover, we will show that polyhedron H is integral, and therefore that Q and D have equivalent linear programming (LP) relaxations. To the best of our knowledge, our approach is the first implicit modeling technique that is able to accurately formulate, and solve efficiently, *multi-activity* shift scheduling problems.

The remainder of the paper is organized as follows. In the next section, we present a literature review on shift scheduling problems and we introduce formal languages and grammar theory. In Section 3, we describe our modeling methodology using grammars to formulate shift scheduling problems. In Section 4, we present theoretical results relevant to our model; in particular, we demonstrate that our model has the same LP relaxation as Dantzig set covering model. Finally, we present comparative computational results in Section 5 on classical shift scheduling problems from Mehrotra et al. (2000) and on a set of large-scale problem instances with multiple activities.

2 Background Material

This section reviews the literature on shift scheduling problems and presents basic notions of context-free grammars, which are relevant to our study.

2.1 Shift Scheduling

For many organizations, finding the best schedule satisfying all their requirements and constraints is an important, but difficult task. Consequently, several studies were dedicated to this problem. Ernst et al. (2004a,b) present an exhaustive overview of models and methods for problems related to staff scheduling and rostering.

Implicit formulations provide an interesting alternative to the explicit model D. These models do not assign breaks to shifts a priori. Rather, they introduce the notion of *shift types*, which are characterized only by starting and ending times, giving no details about how breaks are assigned within the shifts. Typically, these models capture the number of employees assigned to each shift type and to each break with different sets of variables. From an optimal solution to such an implicit model, one can retrieve the number of employees assigned to each shift type and each break, and construct an optimal set of shifts through a polynomial-time procedure.

Rekik et al. (2004) give such an implicit model based on a transportation problem to assign breaks to shifts. They show that the LP relaxation of their model is equivalent to the LP relaxations of two other classical implicit formulations, namely those proposed in Aykin (1996), Bechtolds and Jacobs (1990). Since Dantzig set covering model has the same integrality gap as Aykin model, the LP relaxations of these four models are equivalent. Rekik et al. (2005) propose extensions to allow more flexibility in the definition of breaks. However, to this date, we are not aware of any implicit formulation that can accurately represent *multi-activity* shift scheduling problems.

An alternative to existing explicit and implicit models is to use formal languages to model work regulations. Côté et al. (2007) propose an IP model based on a regular language, represented by a finite deterministic automaton, to formulate the constraints defining a shift, and to represent all feasible shifts using a network flow formulation. Côté et al. (2009) extend these results by using context-free grammars in modeling shift scheduling problems. From a grammar describing work regulations, they generate an IP model, based on assignment variables y_{ije}, that implicitly describe all feasible shifts for each employee e. Since the number of employees is bounded below by $\max_{i\in I}\sum_{j\in J} b_{ij}$, this model generates a large number of variables. Moreover, in the case where many employees are alike, this model has symmetry issues.

Multi-activity shift scheduling problems. With the use of formal languages, many constraints in the planning of shifts can be considered. In particular, these modeling methods can deal with contexts where multiple activities can be performed during the same shift, each activity having its own labor requirements. Compact models for multi-activity problems were seldom studied in the literature. Among the few papers adressing this topic, Loucks and Jacobs (1991), Ritzman et al. (1976) model the tour scheduling problem (shift scheduling over one week) with Boolean assignment variables, specifying the number of employees assigned to a given task at any given time. Since such modeling approaches yield very large IP formulations, both papers propose heuristic methods to construct and improve the solutions. Moreover, they do not place breaks or meals during the shifts, nor do they handle regulations concerning the transition between activities. Approaches using column generation were suggested in Bouchard (2004), Demassey et al. (2006), Vatri (2001). The first two propose approaches to schedule air traffic controllers. While Vatri (2001) uses a heuristic method to build the schedule without taking into account break placement, Bouchard (2004) extends his work to include break placement and solves the problem with a heuristic column generation approach. Demassey et al. (2006) propose a column generation procedure based on constraint programming, that solves efficiently the LP relaxation of the problem stated in Section 5.2 for up to 10 work-activities. However, they report that branching to find integer solutions is difficult and succeed only for the smallest instances.

In the following, we study some basic properties of grammars and show how they can be used in the context of shift scheduling problems.

2.2 Grammars

A context-free grammar defines a language over a given alphabet by means of a set of rules called *productions*. A production is a rule that specifies a substitution of symbols. These symbols are of two types: the *terminal symbols* are letters of the alphabet, generally represented by lower case letters, and the *non-terminal symbols* designate a subsequence that could be rewritten using the associated productions, generally represented by upper case letters. More formally, a production is represented as follows: $\alpha \to \beta$, where α is a non-terminal symbol and β is a sequence of terminal and/or non-terminal symbols. The productions of a grammar can be used recursively to generate new symbol sequences until only terminal symbols are part of the sequence. A sequence of terminal symbols is called a *word*.

Definition. A *context-free grammar* G is characterized by a tuple (Σ, N, P, S) where:

- Σ is an alphabet;
- N is a set of non-terminal symbols;
- P is a set of productions;
- S is the starting non-terminal.

A word, or sequence of letters from alphabet Σ, is *recognized* by a grammar G if it can be generated by successive applications of productions from G, starting with non-terminal S.

In the following, we will use the term grammar to refer to a context-free grammar and we will assume that, except when specified otherwise, all grammars are in Chomsky normal form, meaning that all productions are of the form $X \to \beta$ where $X \in N$ and $\beta \in N \times N \cup \Sigma$. Note that this assumption is not restrictive since any context-free grammar can be converted to Chomsky normal form; see Hopcroft et al. (2001) for more information on formal languages.

Example 1 *The following grammar G defines all feasible shifts for a simple shift scheduling problem. A shift must have a duration equal to the planning horizon and contain one break of one period anywhere during the shift except at the first or the last period. Work and break periods are respectively represented by letters w and b.*

Table 1: Derivation of word $wwbw$ from grammar G of Example 1

P	**CS**
$-$	S
$S \to XW$	XW
$X \to WB$	WBW
$W \to WW$	$WWBW$
$W \to w$	$wWBW$
$W \to w$	$wwBW$
$B \to b$	$wwbW$
$W \to w$	$wwbw$

$G = (\Sigma = (w, b), N = (S, X, W, B), P, S)$, *where* P *is:*

$S \to XW, \qquad X \to WB, \qquad W \to WW \mid w, \qquad B \to b,$

where the symbol $\mid$ *specifies a choice of production. The shifts* wbw, $wwwwwbw$ *and* $wbww$, *among others, are recognized by* G. $wbwb$ *is not recognized by* G. *Word* $wwbw$ *is obtained by the derivation shown in Table 1, where* ***P*** *is the production used and* ***CS*** *is the current sequence, obtained from the previous sequence by applying the production on the left side.*

A common way to illustrate the derivation of a word from a grammar is to use a tree, called *parsing tree*, where the root node is the starting non-terminal S, the innernodes are non-terminals and leaves are letters of the alphabet. A production $X \to YZ$ is represented by nodes Y and Z as left and right children of node X, while $X \to a$ is represented by node X and a unique child, leaf a. When listed from left to right, the leaves form a word recognized by the grammar. Figure 1 shows the two parsing trees induced by grammar G from Example 1 on words of length four ($wwbw$ and $wbww$).

A parsing tree representing a word ω of length n has the following properties:

- An innernode and its children represent a production in P.
- A leaf is associated with a position $i \in \{1, \ldots, n\}$ in ω and represents the letter from Σ taking place at position i.
- Any innernode is the root of a tree inducing a subsequence of ω, starting at position $i \in \{1, \ldots, n\}$ with length $l \in \{1, \ldots, n - i + 1\}$.

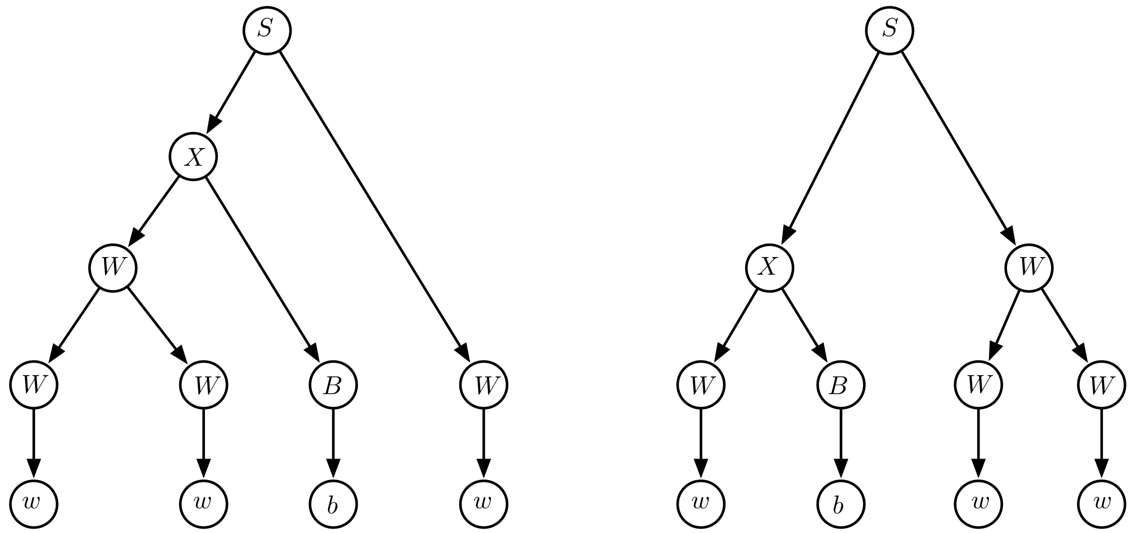

Figure 1: Parsing trees for grammar G from Example 1 on words of length 4.

Using these observations, the next developments characterize a graph embedding all parsing trees associated to words of a given length.

The DAG Γ. In the following, we describe a *directed acyclic graph* (DAG) Γ that encapsulates all parsing trees associated to words of a given length n recognized by a grammar $G = (\Sigma, N, P, S)$. The DAG Γ has an and-or structure containing two types of nodes: nodes O (the or-nodes) represent non-terminals from N and letters from Σ, and nodes A (the and-nodes) represent productions from P. Each node is characterized by its symbol (non-terminal, letter, or production) and the position and length of the subsequence it generates. We define O_{il}^{π} the node associated with non-terminal or letter π that generates a subsequence at position i of length l. Note that if $\pi \in \Sigma$, the node is a leaf and l is equal to one. Also, Γ has a root node described by O_{1n}^{S}. Likewise, $A_{il}^{\Pi,t}$ is the t^{th} node representing production Π generating a subsequence from position i of length l. There are as many $A_{il}^{\Pi,t}$ nodes as there are ways of using Π to generate a sequence of length l from position i. We will refer to this set as the (potentially empty) set $A(\Pi, i, l)$.

The DAG Γ is built in such a way that a path from one node to any other node alternates between or-nodes O and and-nodes A. More precisely, the DAG Γ has the following properties:

- Children of an or-node O^{π}_{il} with $l > 1$, denoted $ch(O^{\pi}_{il})$, are all and-nodes $A^{\Pi,t}_{il}$ such that $\Pi : \pi \to \beta$, $\beta \in N \times N \cup \Sigma$ and $t \in A(\Pi, i, l)$.

- Each or-node O^{π}_{i1}, where π is a non-terminal, has only one child: $ch(O^{\pi}_{i1}) = A^{\Pi,1}_{i1}$ such that $\Pi : \pi \to a$, where $a \in \Sigma$.

- Parents of an or-node O^{π}_{il}, where $\pi \neq S$ is a non-terminal, denoted $par(O^{\pi}_{il})$, are and-nodes of the form $A^{\Pi,t}_{jm}$ such that $\Pi : X \to \pi Z$ or $\Pi : X \to Y\pi$, where $j \leq i$ and $m \geq l$.

- Each or-node O^{π}_{i1}, where π is a letter, has only one parent: $par(O^{\pi}_{i1}) = A^{\Pi,1}_{i1}$ such that $\Pi : X \to \pi$.

- Each and-node $A^{\Pi,t}_{il}$ with $l > 1$ such that $\Pi : X \to YZ$ has exactly two children: O^{Y}_{ik} and $O^{Z}_{i+k,i+l-k-1}$, where $k \leq i + l - 2$.

- Each and-node $A^{\Pi,1}_{i1}$ such that $\Pi : X \to a$, where $a \in \Sigma$, has only one child: O^{a}_{i1}.

- Each and-node $A^{\Pi,t}_{il}$ has only one parent: O^{π}_{il} such that $\Pi : \pi \to \beta$, $\beta \in N \times N \cup \Sigma$, if $l > 1$, and $\Pi : \pi \to a$, $a \in \Sigma$, if $l = 1$.

Figure 2 presents the DAG Γ associated with grammar G from Example 1 on a word of length 4. It is easy to verify the above properties on this DAG.

To derive any parsing tree from Γ, we start at the root O^{S}_{1n}. We visit an or-node O^{π}_{il} by selecting exactly one child, which is necessarily an and-node. We visit an and-node $A^{\Pi,t}_{il}$ by choosing all its children (exactly two if $l > 1$, one otherwise). By traversing Γ in this way until the only remaining unvisited nodes are leaves, we obtain a parsing tree associated to the word defined by the remaining unvisited nodes. Conversely, starting from a given word ω, we can traverse Γ backwards in a straightforward way to derive the parsing tree associated to ω. In practice, Γ is built by a procedure suggested in Quimper and Walsh (2007) inspired by an algorithm from Cooke, Younger, and Kasami (see Hopcroft et al. (2001)).

Grammar-based IP model. Using the structure of the DAG Γ, Côté et al. (2009) present a system of linear equations in 0-1 variables that allow to identify any word recognized by a

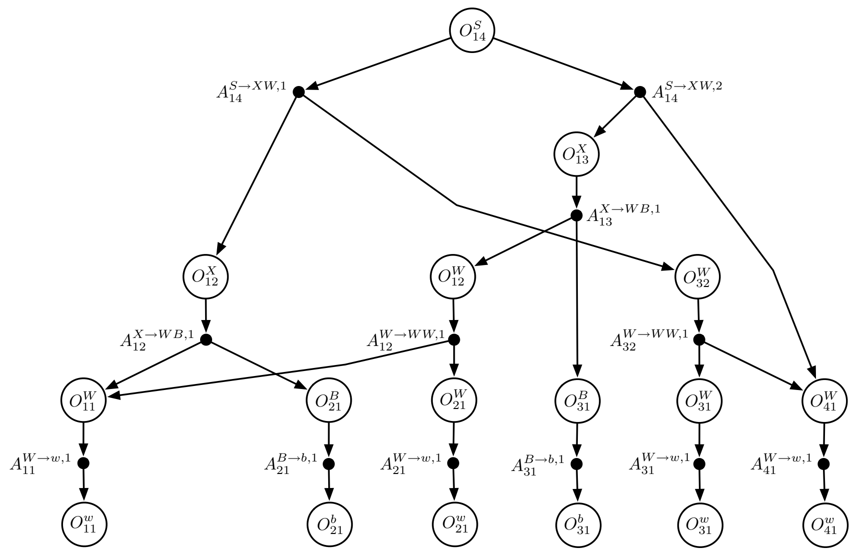

Figure 2: DAG Γ for grammar from Example 1 on a word of length 4.

given grammar G. To each node O_{il}^{π} and $A_{il}^{\Pi,t}$ in Γ are associated 0-1 variables u_{il}^{π} and $v_{il}^{\Pi,t}$, respectively. If we denote by L, the set of leaves in Γ, these equations are as follows:

$$u_{il}^{\pi} = \sum_{A_{il}^{\pi,t} \in ch(O_{il}^{\pi})} v_{il}^{\Pi,t}, \qquad \forall O_{il}^{\pi} \in O \setminus L, \tag{6}$$

$$u_{il}^{\pi} = \sum_{A_{il}^{\Pi,t} \in par(O_{il}^{\Pi})} v_{il}^{\Pi,t}, \qquad \forall O_{il}^{\pi} \in O \setminus O_{1n}^{S}, \tag{7}$$

$$u_{il}^{\pi} \in \{0,1\}, \qquad \forall O_{il}^{\pi} \in O, \tag{8}$$

$$v_{il}^{\Pi,t} \in \{0,1\}, \qquad \forall A_{il}^{\Pi,t} \in A. \tag{9}$$

Constraints (6) ensure that if variable u_{il}^{π} is equal to one, exactly one of the variables associated with its children must be equal to one. Similarly, constraints (7) ensure that if variable u_{il}^{π} is equal to one, exactly one of the variables associated with its parents must be equal to one. Consequently, when we set $u_{1n}^{S} = 1$, if this system of equations has a solution, then, in any solution, the variables equal to one form a parsing tree associated to a word of length n recognized by G. Conversely, let ω be a word of length n on alphabet Σ. If we set to one the u_{i1}^{j} variables that form ω when the letters j are listed from left to right, then, if this system of equations has a solution, ω is recognized by G and the variables of the solution set to one form a parsing tree associated to word ω.

We can rewrite equations (6) and (7) as follows:

$$u_{1n}^{S} = \sum_{A_{1n}^{\Pi,t} \in ch(O_{1n}^{S})} v_{1n}^{\Pi,t}, \tag{10}$$

$$\sum_{A_{il}^{\Pi,t} \in par(O_{il}^{\pi})} v_{il}^{\Pi,t} = \sum_{A_{il}^{\Pi,t} \in ch(O_{il}^{\pi})} v_{il}^{\Pi,t}, \qquad \forall O_{il}^{\pi} \in O \setminus \left\{L \cup \{O_{1n}^{S}\}\right\}, \tag{11}$$

$$u_{i1}^{j} = v_{i1}^{X \to j,1}, \qquad \forall O_{i1}^{j} \in L. \tag{12}$$

This system of equations presents a structure similar to network flow conservation equations, but the two systems are different. Indeed, a solution to equations (10)-(12) does not specify a path in a network, but rather a tree in the DAG Γ, since any variable associated to an and-node, which is equal to one in a solution, will also have its two children with variables equal to one. Hence, (10)-(12) are not flow conservation equations. Further, if we represent system (10)-(12) in matrix notation, we can easily show that the corresponding matrix is not totally unimodular, contrary to the incidence matrix of a network, which is used to represent flow conservation equations. In spite of this, Pesant et al. (2009) have shown (see Section 4) that the polyhedron defined by (10)-(12) is integral, like the polyhedron defined by flow conservation equations.

3 Grammar-Based Model for Shift Scheduling

As explained in Côté et al. (2009), the system of equations (10)-(12) can be used in the context of shift scheduling problems, where the constraints defining any feasible shift are represented by a grammar G, i.e., each word ω recognized by grammar G corresponds to a feasible shift $s \in \Omega$. In this context, the number of periods $|I|$ corresponds to n, the length of any given word recognized by G, while the set of activities J corresponds to Σ, the letters of the alphabet.

Côté et al. (2009) describe an IP model based on this correspondance, using assignment variables y_{ije}, that implicitly describe all feasible shifts for each employee e. But, as explained in Section 2.1, when employees are similar, this model exhibits a lot of symmetry, which makes it impractical to solve large-scale instances. Assuming all employees can be assigned to the same shifts, we introduce here a new grammar-based IP model, that will not suffer from the same performance issues.

In equations (10)-(12), each variable is binary and specifies whether or not its corresponding node is part of the parsing tree selected to generate a word. In the new model, each variable is a nonnegative integer that specifies how many parsing trees the associated node is part of. Since we minimize an objective function with nonnegative costs, the integer variables do not need to be bounded from above.

As in the Introduction, let y_{ij} denote the number of employees assigned to activity $j \in J$ at period $i \in I$. We can replace the leaf variables u_{i1}^{j} by the variables y_{ij}. Model Q presented in the Introduction, can now be explicitly stated as follows:

$$f(Q) = \min \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

$$y_{ij} \geq b_{ij}, \qquad \forall i \in I, j \in J, \tag{13}$$

$$u_{1n}^{S} = \sum_{A_{1n}^{\Pi,t} \in ch(O_{1n}^{S})} v_{1n}^{\Pi,t}, \tag{14}$$

$$\sum_{A_{il}^{\Pi,t} \in par(O_{il}^{\pi})} v_{il}^{\Pi,t} = \sum_{A_{il}^{\Pi,t} \in ch(O_{il}^{\pi})} v_{il}^{\Pi,t}, \qquad \forall O_{il}^{\pi} \in O \setminus \left\{ L \cup \{O_{1n}^{S}\} \right\}, \tag{15}$$

$$y_{ij} = v_{i1}^{X \to j,1}, \qquad \forall i \in I, j \in J, \tag{16}$$

$$u_{1n}^{S} \geq 0 \text{ and integer}, \tag{17}$$

$$v_{il}^{\Pi,t} \geq 0 \text{ and integer}, \qquad \forall A_{il}^{\Pi,t} \in A, \tag{18}$$

$$y_{ij} \geq 0 \text{ and integer}, \qquad \forall i \in I, j \in J. \tag{19}$$

Once Q is solved, an implicit solution is obtained. To find the individual schedules from this solution, we traverse the DAG Γ from the root to the leaves visiting the nodes with value greater than zero. Each time a node is evaluated, its value is decreased by one. When a leaf node is reached, its value is inserted to the current schedule at the right position. Model Q ensures that u_{1n}^{S} words recognized by grammar G can be extracted from the implicit solution. In the context of shift scheduling, variable u_{1n}^{S} thus represents the total number of employees needed to perform all required shifts.

4 Theoretical Properties of Grammar-Based Models

In this Section, we study the polyhedral properties of the implicit grammar-based model Q and compare it with other models from the literature. First, using the notation from the Introduction, we denote by H the polyhedron defined by equations (14)-(16) along with nonnegativity constraints on all variables. Our first result states that this polyhedron is integral, thus extending the result derived in Pesant et al. (2009) for a similar polyhedron defined over 0-1 variables.

Theorem 2 *H is an integral polyhedron.*

Proof. To simplify the notation, we denote H using matrix notation as follows: $H = \{z \geq 0 | Mz = b\}$. Now, let d be any arbitrary costs associated to variables z. The result will follow if we can prove that there always exists an integer optimal solution to the linear program: $\min\{dz | z \in H\}$. For this, it suffices to construct an integer point z^I in H that satifies the complementary slackness conditions: $(\lambda^* M - d)z^I = 0$, where λ^* is an optimal solution to the dual $\max\{\lambda b | \lambda M \leq d\}$.

Let z^* be an optimal solution to the linear program and assume that $m = \lceil u^S_{1n} \rceil > 0$ (otherwise, if $m = 0$, $z^I = 0$ is an integer point in H satisfying the complementary slackness conditions). Our objective is to construct an integer solution z^I such that for every k for which $\lambda^* M_k < d_k$, we have $z^I_k = 0$. This condition can be easily maintained by enforcing that $z^I_k = 0$ whenever $z^*_k = 0$.

First, set $u^S_{1n} = m$ in z^I. By definition of H, since $u^S_{1n} > 0$ in z^*, there exists at least one variable corresponding to a child of root-node O^S_{1n} that has a value greater than 0 in z^*, say z^*_k corresponding to node $A^{\Pi,t}_{1n}$, with $\Pi : X \to YZ$. We continue our construction by fixing $z^I_k = m$. From constraints (15), since $z^*_k > 0$, the two children of $A^{\Pi,t}_{1n}$, say O^X_{1k} and $O^Y_{k+1,n-k}$ have at least one child each with a corresponding value greater than 0, say $z^*_{k_1}$ and $z^*_{k_2}$. We then fix $z^I_{k_1} = m$ and $z^I_{k_2} = m$. We continue this process, following the children of the nodes and setting them to m in z^I, until we reach the leaves of the DAG Γ.

The variables set to m in z^I form a tree in the DAG Γ that satisfies constraints $Mz = b$ by construction. Furthermore, since we only used variables that were already set to a value greater than 0 in the LP optimal solution z^*, we know that z^I satisfies the complementary slackness conditions with respect to λ^*. Therefore, there always exists an optimal integer solution to the linear program $\min\{dz | z \in H\}$, with arbitrary d, i.e., polyhedron H is integral. ∎

From this result, we observe that integrality constraints (17) and (18) are redundant in model P. However, in practice, we found that leaving these constraints in the formulation helps the IP solver to further presolve the model and, overall, speeds up the solution process. Consequently, the experimentions in Section 5 were performed by leaving the integrality constraints in the models.

The next theorem uses the previous result to establish the equivalence between the LP relaxations of models Q and D, the Dantzig set covering formulation, presented in the Introduction.

Theorem 3 *Q and D have equivalent LP relaxations.*

Proof. The proof is direct using Lagrangean duality arguments. Let $\gamma_{ij} \geq 0$ denote Lagrangean multipliers associated to the requirement constraints, (1) in model D and (3) in model Q. Also, let $f_{LP}(M)$ denote the optimal objective value of the LP relaxation of a model M. We then have:

$$\begin{aligned}
f_{LP}(Q) &= \max_{\gamma \geq 0} \left\{ \sum_{i \in I} \sum_{j \in J} \gamma_{ij} b_{ij} + \min \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} - \gamma_{ij}) y_{ij} \,|\, (y, v) \in H \right\} \right\} \\
&= \max_{\gamma \geq 0} \left\{ \sum_{i \in I} \sum_{j \in J} \gamma_{ij} b_{ij} + \min \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} - \gamma_{ij}) y_{ij} \,|\, (y, v) \in H, (y, v) \text{ integer} \right\} \right\} \\
&= \max_{\gamma \geq 0} \left\{ \sum_{i \in I} \sum_{j \in J} \gamma_{ij} b_{ij} + \min \left\{ \sum_{i \in I} \sum_{j \in J} (c_{ij} - \gamma_{ij}) (\sum_{s \in \Omega} \delta_{ijs} x_s) \,|\, x_s \geq 0 \text{ and integer} \right\} \right\} \\
&= \max_{\gamma \geq 0} \left\{ \sum_{i \in I} \sum_{j \in J} \gamma_{ij} b_{ij} + \min \left\{ \sum_{s \in \Omega} (c_s - \sum_{i \in I} \sum_{j \in J} \gamma_{ij} \delta_{ijs}) x_s \,|\, x_s \geq 0 \text{ and integer} \right\} \right\} \\
&= \max_{\gamma \geq 0} \left\{ \sum_{i \in I} \sum_{j \in J} \gamma_{ij} b_{ij} + \min \left\{ \sum_{s \in \Omega} (c_s - \sum_{i \in I} \sum_{j \in J} \gamma_{ij} \delta_{ijs}) x_s \,|\, x_s \geq 0 \right\} \right\} \\
&= f_{LP}(D).
\end{aligned}$$

■

Since Dantzig set covering model yield the same integrality gap as the models suggested in Aykin (1996), Bechtolds and Jacobs (1990), Rekik et al. (2004) (see Section 2), Theorem 3 implies that the LP relaxation of P is also equivalent to the LP relaxations of these other implicit models. Note however that, to the best of our knowledge, these models cannot be extended to the multi-activity case.

5 Computational Experiments

The objective of our computational experiments is to evaluate the efficiency of our new implicit grammar-based model, when processed by a state-of-the-art IP solver. For this purpose, we will compare model P to the explicit model D and to another implicit model, due to Aykin (1996), under the same conditions. We will use instances from the literature, all having one work activity, as well as large-scale instances with one work activity, but also multiple work activities.

5.1 Shift Scheduling with Multiple Rest Breaks, Meal Breaks, and Break Windows

In this section, we compare our model with a state-of-the-art implicit model, proposed in Aykin (1996), and to the explicit model D on a large set of shift scheduling instances used in Mehrotra et al. (2000) from shift specifications and labor requirements reported in Aykin (1996), Henderson and Berry (1976), Segal (1974), Thompson (1995). The problems differ from one another in the labor requirements, the set of allowed shifts, the planning horizon, the number of breaks, the break windows, the cost structures, and whether the problem is cyclic or not. We refer to Mehrotra et al. (2000) for details on shift generation rules. Here, we present a general description of the three classes of problems studied.

Thompson Set. Thompson (1995) presents two sets of non-cyclic problems. The first set are problems on 15-h demand patterns. Shifts either allow one break or none depending on their length. The second set are problems on 20-h demand patterns. Shifts allow one break of one hour. The planning horizons are divided into periods of 15 minutes and shifts can start at any period that allows them to finish within the planning horizon. The break windows depend on the duration of the shifts and the costs are proportional to the number of work hours in a shift.

Aykin Set. Aykin (1996) presents a set of cyclic problems, with shifts composed of three breaks and differing only in the length of the break windows. The planning horizon is 24-h divided into periods of 15 minutes. All shifts have the same length and must start on the hour or the half-hour. The cyclic case is handled in the same way in the three modeling approaches. The planning horizon is extended to allow shifts to start at any time in the original planning horizon.

Mehrotra Set. Mehrotra et al. (2000) use the same shift generation rules as in Aykin Set, but allow the shifts to have different durations and to start at any period. The problems were tested as cyclic problems using the same labor requirements as in Aykin Set.

Definition of the Grammars. The following presents the grammars used for each set of instances. For the sake of clarity, the grammars are not stated in Chomsky normal form. In the sets of productions P, $\rightarrow_{[min,max]}$ restricts the subsequences generated with a given production to have a length between min and max periods.

Grammar for Thompson Set. Let Φ be the set of feasible shift types. Let bws_l and bwe_l be the break window starting and ending periods for shift type $l \in \Phi$. Let sl_l and bl_l be the shift and break lengths for shift type $l \in \Phi$.

$G = (\Sigma = (w, b, r), N = (S, W, R, A_l, B_l, M_l \;\; \forall l \in \Phi), P, S)$,

where w is a period of work, b is a break period and r is a rest period. P is defined as follows:

$$\begin{array}{lll} S \rightarrow RA_lR \mid A_lR \mid RA_l & \forall l \in \Phi & W \rightarrow Ww \mid w \\ A_l \rightarrow_{[sl_l, sl_l]} M_lW & \forall l \in \Phi & R \rightarrow Rr \mid r \\ M_l \rightarrow_{[bws_l+bl_l, bwe_l+bl_l]} WB_l & \forall l \in \Phi & \\ B_l \rightarrow b^{bl_l} & \forall l \in \Phi & \end{array}$$

Grammar for Aykin and Mehrotra Sets. Let Φ be the set of feasible shift types. Note that in Aykin Set, Φ contains only one element since all shifts have the same length. Let bws_l^n and bwe_l^n be the break window starting and ending periods for shift type $l \in \Phi$ and break $n \in \{1, 2, 3\}$. Let sl_l and bl_l^n be the length of shift and breaks $n \in \{1, 2, 3\}$, respectively, for shift type $l \in \Phi$.

$G = (\Sigma = (w, b, r), N = (S, W, R, B_l^n \;\; \forall n \in \{1, 2, 3\}, A_l, M_l^A, M_l^B, M_l^C \;\; \forall l \in \Phi), P, S)$,

where w is a period of work, b is a break period and r is a rest period. P is defined as follows:

$$\begin{array}{llll} S \rightarrow RA_lR \mid A_lR \mid RA_l & \forall l \in \Phi & B_l^n \rightarrow b^{bl_l^n} & \forall l \in \Phi, n \in \{1, 2, 3\} \\ A_l \rightarrow_{[sl_l, sl_l]} M_l^AW & \forall l \in \Phi & W \rightarrow Ww \mid w & \\ M_l^A \rightarrow_{[bws_l^3+bl_l^3, bwe_l^3+bl_l^3]} M_l^BWB_l^3 & \forall l \in \Phi & R \rightarrow Rr \mid r & \\ M_l^B \rightarrow_{[bws_l^2+bl_l^2, bwe_l^2+bl_l^2]} M_l^CWB_l^2 & \forall l \in \Phi & & \\ M_l^C \rightarrow_{[bws_l^1+bl_l^1, bwe_l^1+bl_l^1]} WB_l^1 & \forall l \in \Phi & & \end{array}$$

Results. We compare our model on these instances to the explicit model D from Dantzig (1954) and to the implicit model from Aykin (1996). We generated the three IP models and solved them with CPLEX 10.1.1 with the default parameters. As in Mehrotra et al. (2000),

Table 2: Number of instances solved to optimality within the four minutes time limit

Models/Sets	Thompson 15-h (21)	Thompson 20-h (80)	Aykin (16)	Mehrotra et al. (16)
Implicit Grammar	21	79	16	16
Aykin	21	66	16	16
Dantzig	21	63	16	16

we gave a four-minute time limit to find the optimal solution. Experiments were run on a 2.3GHz AMD Opteron with 3GB of memory.

Table 2 shows the number of instances solved to optimality within the four-minute time limit by the three models on each set. The numbers in parentheses are the number of available instances in each set.

For the instances solved to optimality by all three models, Table 3 presents a summary of the model size and solution statistics. $|C|$, $|V|$ and $|NZ|$ are the number of constraints, variables and non-zeroes in the models. Nit, $Nnodes$ and $Times$ give the average values of the number of iterations, the number of nodes and the time needed to solve the instances to optimality. Gap is the average IP gap for the instances that were not solved to optimality with the three models, i.e., $Gap = 100(Z_{IP} - Z_{LP})/Z_{IP}$ where Z_{IP} and Z_{LP} are, respectively, the best upper and lower bounds after the time limit has been reached.

The results on Thompson Set show that our model can lead to better solution times in comparison with the two other formulations. Indeed, with the Implicit Grammar model, we manage to solve to optimality 13 instances more than Aykin to optimality and 16 more than Dantzig. However, for instances in Aykin and Mehrotra Sets, our model appears less effective, although it finds the optimal solution for all instances within the time limit, as the two other models do.

Mehrotra et al. (2000) present a branch-and-price approach involving specialized branching rules for solving Dantzig set covering formulation. They compare their method with Aykin model solved with CPLEX 4.0 on the same instances stated above.The results show that their method is generally superior. Since CPLEX has evolved considerably since these experiments were performed, it is difficult to deduce from these results a fair comparison

Table 3: Summary for the instances solved to optimality

Model	$\lvert C\rvert$	$\lvert V\rvert$	$\lvert NZ\rvert$	Nit	$Nnodes$	$Time$	Gap
			Thompson set for 15-h demand curves				
Implicit Grammar	6418	9686	28481	400	591	0,46	–
Aykin	421	3673	27521	3467	13	0,93	–
Dantzig	60	3312	85977	185	10	0,34	–
			Thompson set for 20-h demand curves				
Implicit Grammar	13732	22118	65582	1338	11	4,04	0,004
Aykin	838	9208	67912	5866	130	4,89	0,082
Dantzig	80	8450	229215	751	35	3,82	0,140
			Aykin set				
Implicit Grammar	5703	36184	106902	3421	17	3,59	–
Aykin	274	697	3396	271	2	0,07	–
Dantzig	130	4681	149760	860	36	0,77	–
			Mehrotra et al. set				
Implicit Grammar	11201	16488	47683	15629	15	14,50	–
Aykin	1572	6961	33955	2006	2	1,06	–
Dantzig	132	46801	1497113	659	5	8,90	–

between their approach and our model solved with CPLEX 10.1.1.

5.2 Shift Scheduling with Multiple Rest and Meal Breaks, and Multiple Work Activities

This section presents a shift scheduling problem for a retail store, allowing up to ten different work activities. We present the specifications of the problem and compare our model with Dantzig model and an extentsion of Aykin model suggested in Rekik et al. (2005), allowing to model work-stretch duration restrictions, for instances with one work activity.

Problem Definition

1. The planning horizon is 24 hours divided into 96 periods of 15 minutes.
2. A shift may start at any period of the day allowing enough time to complete its duration during the planning horizon.
3. A shift must cover between 3 hours and 8 hours of work activities.
4. If a shift covers at least 6 hours of work activities, it must have a 15 minute-breaks, a lunch break of 1 hour.
5. If a shift covers less than 6 hours of work activities, it must have a 15 minute-break, but no lunch.
6. If performed, the duration of a work-activity is at least 1 hour (4 consecutive periods).
7. A break (or lunch) is necessary between two different work activities.
8. Work activities must be inserted between breaks, lunch and rest stretches.
9. For each period of the planning horizon, labor requirements for every work-activity are available.
10. Overcovering and undercovering are allowed. Costs are associated with overcovering and undercovering the requirements of a work-activity at a given period.

11. The cost of a shift is the sum of the costs of all work-activity/period achieved in the shift.

Definition of the Grammar. The following presents the grammar used for this problem. For the sake of clarity, the grammar is not stated in Chomsky normal form.

$G = (\Sigma = (a_j \ \ \forall j \in A, b, l, r), N = (S, F, P, W, A_j \ \ \forall j \in A, B, L, R), P, S)$,

where A is the set of work-activity, a_j is a period of work on activity $j \in A$, b is a break period, l is a lunch period and r is a rest period. In P, $\rightarrow_{[min,max]}$ restricts the subsequences generated with a given production to have a length between min and max periods. P is defined as follows:

$$
\begin{array}{ll}
S \rightarrow RFR \mid FR \mid RF \mid RPR \mid PR \mid RP & B \rightarrow b \\
F \rightarrow_{[30,38]} WBWLWBW \mid WLWBWBW \mid WBWBWLW & L \rightarrow llll \\
P \rightarrow_{[13,24]} WBW & R \rightarrow Rr \mid r \\
W \rightarrow_{[4,\infty)} A_j \quad \forall j \in A & \\
A_j \rightarrow A_j a_j \mid a_j \quad \forall j \in A &
\end{array}
$$

Results. To compare the different models for this problem, we generated the IP models representing these rules and solved them with CPLEX 10.1.1 with the default parameters. We gave a one-hour time limit to find the optimal solution. Experiments were run on a 3.20 GHz Pentium 4.

First, we compare Dantzig model and the extension of Aykin model suggested in Rekik et al. (2005), called the Aykin/Rekik model, to our model on ten instances with one work-activity, which differ only from their labor requirements. Table 4 presents the results. $|C|$, $|V|$ and $|NZ|$ are the number of constraints, variables and non-zeroes in the models. Note that the differences in the number of variables between the instances for the same model come from the slack variables introduced to requirements constraints to allow overcovering and undercovering. For periods where no employees are required, we suppose that the retail store is closed and that no work should be scheduled. *Nit*, *Nnodes* and *Times* give the number of iterations, nodes and the solution times for the instances solved to optimality within the time limit; otherwise the sign “>” is used.

The comparison between the three models shows that Dantzig model tends to be less competitive on problems with a large number of shifts. In the one-activity case, solving the

Table 4: Model comparison on the one-activity problem

No	$\|C\|$	$\|V\|$	$\|NZ\|$	Nit	$Nnodes$	$Time$
			Implicit Grammar model			
1	16191	66621	198517	137	0	0,52
2	16191	66653	198549	67053	12	132,54
3	16191	66653	198549	762526	1312	838,79
4	16191	66637	198533	13245	37	9,05
5	16191	66629	198525	1154	0	0,69
6	16191	66629	198525	1006	0	0,60
7	16191	66637	198533	26715	95	11,60
8	16191	66653	198549	$>$	$>$	$>$
9	16191	63068	198525	12057	215	3,60
10	16191	66637	198533	1977	0	1,01
			Aykin/Rekik et al. model			
1	50007	78247	930056	604	0	2,31
2	50007	78279	930088	1776156	4313	2607,76
3	50007	78279	930088	231981	1061	415,75
4	50007	78263	930072	12382	140	12,23
5	50007	78255	930064	603	0	2,30
6	50007	78255	930064	825	0	2,33
7	50007	78263	930072	5742	74	7,60
8	50007	78279	930088	$>$	$>$	$>$
9	50007	78255	930064	64378	402	59,38
10	50007	78263	930072	2071	0	2,78
			Dantzig model			
1	96	845176	24605722	23	0	45,42
2	96	845208	24605754	$>$	$>$	$>$
3	96	845208	24605754	561202	71033	1477,83
4	96	845192	24605738	70578	18270	128,75
5	96	845184	24605730	101	20	42,79
6	96	845184	24605730	23	0	46,08
7	96	845192	24605738	232	0	89,07
8	96	845208	24605754	$>$	$>$	$>$
9	96	845184	24605730	353	3266	46,05
10	96	845192	24605738	117	0	42,01

Table 5: Multi-activity problems with the Implicit Grammar model

$NbAct$	$NbShifts$	$\|C\|$	$\|V\|$	$Time$	$Nopt(10)$
2	13404928	18068	69893	423,90	9
3	67752783	19945	73152	337,14	9
4	214010944	21822	76417	212,07	9
5	522350575	23699	79688	182,02	10
6	1082991744	25576	82961	179,51	10
7	2006203423	27453	86246	317,31	10
8	3422303488	29330	89492	274,66	10
9	5481658719	31207	92731	296,37	10
10	8354684800	33084	96026	518,02	10

entire model with an IP solver is still manageable, but both the Implicit Grammar models and Aykin/Rekik models are solved more rapidly (except for instance 9, for which Dantzig model performs slightly better than Aykin/Rekik model). Our model succeeds in proving optimality for 9 out of 10 instances, such as the Aykin/Rekik model, but does so in less times for 7 out of these 9 instances. Note also that the number of variables in the Implicit Grammar model is smaller than in the other two models.

Table 5 shows our results on the multi-activity instances. We ran experiments on our model on instances ranging from two to ten work activities. For each instance, we tested ten different labor requirements. Column $NbShifts$ gives the number of feasible shifts for each of the problems, which would be the number of variables needed by Dantzig set covering model. $Nopt(10)$ gives the number of instances solved to optimality within the one-hour time limit. $|C|$, $|V|$ and $Time$ are the average number of constraints and variables, and solution times for the instances solved to optimality.

To our knowlegde, no other implicit formulations are capable of modeling multi-activity instances. To solve Dantzig model on these problems, one must consider column generation methods, since the number of feasible shifts is very large. Demassey et al. (2006) present a column generation approach for these problems. Their method does not succeed in finding optimal solutions, even for the single-activity instances. As for our modeling approach, the multi-activity problems can easily be handled, with a few additionnal productions than what is needed for the one-activity grammar, and results show that they can rapidly be solved on almost all available instances. Note that the growth in the number of constraints and variables when increasing the number of work activities is much slower than the increase in the number of feasible shifts.

6 Conclusion

In this paper, we presented a new implicit IP model for solving multi-activity shift scheduling problems. This model differs significantly from the models proposed in the literature, as our modeling approach uses context-free grammars to represent the constraints defining feasible shifts. This model yields the same LP relaxation bound as the classical set covering model from Dantzig (1954), and the other well-known implicit models in the literature, i.e., Aykin (1996), Bechtolds and Jacobs (1990), Rekik et al. (2004).

Our experiments showed that the solution times for our model are competitive with the solution times for Aykin model on one-activity shift scheduling instances from the literature, and slightly superior to an extension of Aykin model suggested in Rekik et al. (2004) on large-scale one-activity instances. We also showed that our model can be solved to optimality efficiently on instances with up to ten work activities. To the best of our knowledge, no other technique in the literature can solve multi-activity instances efficiently.

An interesting feature of our formulation is that the objective function allows many types of cost structures, contrary to classical implicit formulations. In model Q, costs are associated with activities and periods, but one can modify the objective function to have costs on productions ($\sum_{A_{il}^{\Pi,t} \in A} \text{cost}(v_{il}^{\Pi,t}) v_{il}^{\Pi,t}$), on the root-node (minimizing the root-node variable, would be equivalent to minimizing the number of employees needed), or even on subsequences. In fact, $\sum_t v_{il}^{\Pi,t}$ corresponds to the number of words having a subsequence of length l starting in position i generated from production Π. A subsequence can be a shift or a part of a shift shift, such as a task. One could be interested in tracking a task corresponding to a consecutive assignment of work activities and do so by using the corresponding variables v from A. Therefore, the implicit grammar-based model can easily be extended to handle problems that require the simultaneous assignment of tasks and activities.

Acknowledgments

This work was supported by a grant from the Fond québécois de recherche sur la nature et les technologies. We would like to thank Claude-Guy Quimper for his useful comments on our work.

References

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

Bechtolds, S., L. Jacobs. 1990. Implicit optimal modeling of flexible break assigments. *Management Science* **36** 1339–1351.

Bouchard, M. 2004. Optimisation des pauses dans le problème de fabrication des horaires avec quarts de travail. M.Sc. Thesis, Ecole Polytechnique de Montréal.

Côté, M.-C., B. Gendron, C.-G. Quimper L.-M. Rousseau. 2009. Formal languages for integer programming modeling of shift scheduling problem. *Constraints* **doi:10.1007/s10601-009-9083-2**.

Côté, M.-C., B. Gendron, L.-M. Rousseau. 2007. Modeling the regular constraint with integer programming. *Proc. of CPAIOR'07, Springer-Verlag LNCS* **4510** 29–43.

Dantzig, G. 1954. A comment on Edie's traffic delay at toll booths. *Journal of the Operations Research Society of America* **2** 339–341.

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

Edie, L. 1954. Traffic delays at toll booths. *Journal of the Operations Research Society of America* **2** 107–138.

Ernst, A., H. Jiang, M. Krishnamoorthy, B. Owens, D. Sier. 2004a. An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research* **127** 21–144.

Ernst, A., H. Jiang, M. Krishnamoorthy, D. Sier. 2004b. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research* **153** 3–27.

Henderson, W.B., W.J. Berry. 1976. Heuristic methods for telephone operator shift scheduling. *Management Science* **22** 1372–1380.

Hopcroft, J., R. Motwani, J. D. Ullman. 2001. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley.

Loucks, J.S., F.R. Jacobs. 1991. Tour scheduling and task assignment of a heterogeneous work force: a heuristic approach. *Decision Sciences* **22** 719–739.

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

Pesant, Gilles, C.-G. Quimper, L.-M. Rousseau, M. Sellmann. 2009. The polytope of context-free grammar constraints. *Proc. of CPAIOR'09, Springer-Verlag LNCS* **5547** 223–232.

Quimper, C.-G., T. Walsh. 2007. Decomposing global grammar constraint. *Proc. of CP'07, Springer-Verlag LNCS* **4741** 590–604.

Rekik, M., J.-F. Cordeau, F. Soumis. 2004. Using benders decomposition to implicitly model tour scheduling. *Annals of Operations Research* **128** 111–133.

Rekik, M., J.-F. Cordeau, F. Soumis. 2005. Implicit shift scheduling with multiple breaks ans work stretch duration restrictions. *Publication GERAD-2005-15* .

Ritzman, L., L.J. Krajewski, M.J. Showalter. 1976. The disaggregation of aggregate manpower plans. *Management Science* **22** 1204–1214.

Segal, M. 1974. The operator-scheduling problem: A network-flow approach. *Operations Research* **22** 808–823.

Thompson, G. 1995. Improved implicit optimal modeling of the labor shift scheduling problem. *Management Science* **41** 595–607.

Vatri, E. 2001. Integration de la génération de quart de travail et de l'attribution d'activités. M.Sc. Thesis, Ecole Polytechnique de Montréal.