Les Cahiers du GERAD ISSN: 0711-2440

The Preferential Bidding System at Air Canada

M. Gamache, F. Soumis
D. Villeneuve, J. Desrosiers
E. Gélinas

G–97–12

March 1997
Revised: December 1997

The Preferential Bidding System at Air Canada

Michel Gamache, François Soumis and Daniel Villeneuve
GERAD and École Polytechnique de Montréal
Montréal, Canada, H3C 3A7

Jacques Desrosiers
GERAD and École des Hautes Études Commerciales de Montréal
Montréal, Canada, H3T 2A7

Éric Gélinas
Ad Opt Technologies Inc.
Montréal, Canada, H3V 1G4

April, 1997
Revised: November, 1997

Abstract

This paper describes the Preferential Bidding Problem solved in the airline industry to construct personalized monthly schedules for pilots and officers. This problem consists in assigning to crew members pairings, days off, annual leaves, training periods, etc., while considering a set of weighted bids that reflect individual preferences. This assignment must be done under strict seniority restrictions: the construction of a maximum-score schedule for a particular crew member must never be done at the expense of a more senior employee. This R&D project has resulted in the PREFERENTIAL BIDDING SYSTEM which has been used at Air Canada since May 1995.

The solution process is summarized as follows. For each employee, from the most senior to the most junior, a so-called residual problem is solved: given an employee and a set of unassigned pairings, the solution to an integer linear program determines the employee's maximum-score schedule while taking into account all the remaining employees. The residual problem is solved by column generation embedded in a branch&bound tree. Integer solutions are obtained by using very efficient cutting planes, without which it would have been impossible to solve some of these residual problems.

Key Words: Airline, crew scheduling, preferential bidding, integer linear programming, column generation, cutting planes.

This research was awarded First Prize in the 1995–1996 Practice Prize Competition of the Canadian Operations Research Society at the IFORS/CORS Conference held in Vancouver, July 1996.

Résumé

Cet article décrit le problème de construction d'horaires mensuels personnalisés avec choix de préférences pour les membres d'équipage (pilotes et officiers) en transport aérienen. Ce problème consiste à affecter aux membres d'équipage des rotations, des repos, des vacances, des entraînements, etc., tout en considérant un ensemble de requêetes pondérées réflétant les préférences de chacun. L'affectation doit tenir compte de l'ancienneté de chaque employé: la construction de l'horaire à pointage maximal pour un employé ne doit jamais s'effectuer au détriment de l'horaire d'un employé ayant plus d'ancienneté. De ce projet de recherche et de développement est né un système informatisé de construction d'horaires personnalisés qui est en usage chez Air Canada depuis le mois de mai 1995.

Le méthode de résolution employé peut se résumer comme suit. Pour chaque employé, du plus ancien au plus récent selon l'ordre d'ancienneté, un problème résiduel est résolu: étant donné un employé et un ensemble de rotations non affectées, la solution d'un problème linéaire en nombres entiers détermine l'horaire à pointage maximal de l'employé tout en considérant les employés résiduels. La résolution du problème résiduel s'effectue par une méthode d'évaluation et de séparation dans un arbre de branchement où, à chaque noeud de branchement, un problème linéaire est résolu par la technique de génération de colonnes. Les solutions entières sont obtenues en utilisant des plans coupants sans lesquels il n'aurait pas été possible de résoudre certains des problèmes résiduels.

Introduction

Management staff in the airline industry must deal with two types of crew scheduling problems that are currently solved sequentially: the well-known *Crew Pairing Problem* followed by the *Monthly Crew Assignment Problem.* The first problem consists in constructing a minimum-cost set of pairings, i.e., round trips composed of legal work days separated by rest periods, which covers all flight legs. The goal of the second problem is to construct monthly schedules for crew members by assigning them pairings, days off, annual leaves, training periods, etc. The solution of each problem must satisfy the collective agreement and security rules.

There are three different methods for constructing monthly schedules: bidline, rostering and preferential bidding. In a *Bidline Problem*, anonymous schedules covering all pairings are first constructed; next, employees choose sequentially, from the most senior to the most junior, their preferred schedule. In a *Rostering Problem*, the construction of personalized schedules must take into account a list of pre-assigned activities. In a *Preferential Bidding Problem*, personalized schedules are constructed as in the rostering problem, while also considering a set of employees' weighted bids which reflect their preferences. In this third approach, the aim is to maximize the award of crew members' preferences while respecting seniority.

This paper describes some aspects of the PREFERENTIAL BIDDING SYSTEM (PBS) used at Air Canada since May 1995 to construct monthly schedules for pilots and officers. It is organized as follows. The first section provides a description of the Preferential Bidding Problem, examines the existing literature and presents the contribution of the paper. Then Section 2 describes the particularities encountered at Air Canada. Section 3 presents some mathematical programming aspects of PBS while Section 4 discusses computational experiments. Finally, we present our conclusions.

1 Problem definition and literature review

In the Monthly Crew Assignment Problem, one must usually consider a set of pre-assigned activities, such as annual leaves and training periods, for each employee. Additionally in the Preferential Bidding Problem, each employee draws up a list of bids. For example, an employee may indicate preferences for a specific pairing, a rest period, a weekend off, flight departures after 8 o'clock in the morning, etc. A

bid may also concern the overall schedule; for example, a pilot may prefer a schedule that has at least 74 flight hours. (At Air Canada, about 75 different kinds of bids can be made.) Each employee associates a weight with each bid and these weights are used to evaluate the *score* of his potential schedules. The aim of the Preferential Bidding Problem is to assign, under strict seniority restrictions, the best personalized schedules to the employees so as to cover all pairings which depart during the month. The problem consists in constructing a maximum-score schedule for each employee in such a way that this maximization is never done at the expense of more senior employees.

When considering a direct mathematical formulation to solve this Preferential Bidding Problem, the weights must be chosen so as to differentiate the bids selected by an employee and to reflect the priority of senior over junior employees. Given the numerous kinds of bids an employee can make, it requires a 32-bit computer word to differentiate those selected by a single employee. Therefore, the score of an employee is an integer number that can be as large as $2^{32} \simeq 10^9$. To reflect the strict seniority between two employees, two computer words are necessary, that is 64 bits, i.e., a number close to $2^{64} \simeq 10^{19}$. Given m employees, seniority restrictions require an objective function taking values up to $2^{m \times 32}$. Hence, for a problem involving 100 employees, the value of the objective function can be as large as $2^{100 \times 32} \simeq 10^{963}$. This huge number clearly shows the necessary recourse to a sequential approach to adequately solve the Preferential Bidding Problem.

At first sight, this problem may seem easy to solve since the seniority restrictions impose a sequential approach. Unfortunately, the search for the best schedule for an employee is a very difficult problem. On the one hand, we look for the best schedule to assign to an employee and on the other hand, we need a set of compatible schedules that cover all pairings. Consequently, finding a good sequential approach is not easy and the few methods presented in the literature are heuristics.

Literature Review. To our knowledge there are only two relevant papers in the airline area. Given the list of weighted bids for each employee, Moore, Evans and Ngo (1978) for Qantas, and Byrne (1988) for CP Air have proposed similar heuristics that sequentially construct a good schedule for each employee, from the most senior to the most junior. For a given employee, the most-desired legal activity is first assigned to the *skeleton schedule*, i.e., the schedule that is under construction. The process continues with the next most-desired legal activity. As a feasibility test one has to

check, for each day of the month, if it is still possible to cover all the residual pairings with the remaining employees, i.e., if the number of employees is at least equal to the number of pairings. The activity assignment to the current skeleton schedule stops when an admissible schedule is obtained. This schedule is then assigned to the given employee and all the assigned pairings are removed from the residual set of pairings. When all schedules are constructed, exchanges of activities between schedules are allowed so as to improve the solution.

The above method has several drawbacks. For example, juxtaposing the most-desired activities does not ensure that the skeleton schedule is optimal or even admissible; in the latter case the building of this individual schedule will require backtracking. Moreover, since the feasibility test is incomplete, the optimization process may require backtracking among employees. Even if a partial solution is optimal, there could exist alternative optimal solutions allowing better schedules for more junior employees.

Contribution of the paper. The method that we propose in this paper also uses a sequential approach based on seniority order. For each employee, from the most senior to the most junior, we solve a *residual problem.* For a given employee and a set of already assigned schedules, we formulate an integer linear program to cover the residual pairings taking into account all the remaining employees. This residual problem is solved by a column generation scheme embedded in a branch&bound tree which provides an optimal schedule for the given employee and ensures a feasible solution for the others. Integer solutions are obtained by using a very efficient type of cutting plane. Average integrality gaps are of the order of 15% but may reach 99% in some instances. Without the use of these cutting planes, the solution of several residual problems would be practically impossible.

2 The Preferential Bidding Problem at Air Canada

At Air Canada, the PREFERENTIAL BIDDING SYSTEM is used to construct monthly schedules for technical crew members, i.e., pilots, first officers, and second officers. These three groups of employees represent three independent problems. Moreover, since each employee is qualified to fly on only one type of aircraft, problems are divided into as many instances as there are aircraft types. Finally, problems are also

divided according to the four bases of the company: Montreal (YUL), Toronto (YYZ), Winnipeg (YWG), and Vancouver (YVR). Consequently, a problem involves a group of employees assigned to the same base and performing the same function on board a specific type of aircraft. With this partitioning, typical problem sizes range from 20 to more than 100 employees.

The solution must satisfy the rules of the collective agreement which govern the composition of each schedule, the set of schedules, and the technical crew assigned to any pairing, giving rise to three types of constraints. These rules depend on parameter values that can be changed every month. The given values are those used for the computational experiments.

Local constraints. Local constraints govern the construction of each schedule. Here is a short description of some of these rules.

Flight credits: flight credits are flight hours or the equivalent for activities such as training periods, union meetings, annual leaves, etc. The sum of flight credits must be between 70 and 78.

Working days: the number of consecutive working days between rest periods is at most 7.

Rest periods: an employee must have at least 10 days off during the month.

Fatigue factor: the fatigue factor is evaluated by using a score that increases with the covering of each pairing and decreases with rest periods; it must be no greater than 1500 at the beginning of any pairing and must never exceed 3000 during the month.

There are four types of schedules: normal, low time, empty and open time. A schedule that satisfies all constraints and has at least 72 flight credits is called a *normal schedule.* A feasible schedule for which flight credits are between 70 and 72 is called a *low time schedule.* If the schedule construction of an employee cannot respect a constraint, this employee is temporarily assigned an *empty schedule* and becomes a reserve crew member. Finally, an *open time schedule* can be used to cover a sequence of uncovered pairings that cannot be assigned to any employee's schedule. Pairings included in open time schedules are subsequently used to construct the schedules of reserve crew members.

Global constraints. Global constraints govern the composition of all selected schedules. They are grouped in five subsets.

Set Partitioning: each pairing must be assigned to only one employee.
Schedule assignment: each employee must be assigned a normal, low time or empty schedule.
Low time schedules: the solution must contain no more than 2 low time schedules.
Open time schedules: the solution must contain no more than 2 open time schedules.
Open time duration: the uncovered pairings cannot last more than 120 hours.

The separation between local and global constraints is a prelude to a solution process based on column generation. Indeed, the coordination problem keeps all the global constraints while the column generator takes care of the local restrictions. Details of the solution approach are provided in the next section.

Crew Composition. Crew composition constraints concern the teams of crew members assigned to pairings. There are two rules. Firstly, at most one crew member with a medical restriction can be assigned to a pairing. Secondly, an officer may prefer to fly with a specific pilot. These rules are easily treated since each type of problem is solved independently and according to a hierarchical order: pilots, first officers, and second officers. The crew composition constraints are imposed in first and second officers' problems, as needed.

3 The Solution Approach

The solution method sequentially constructs the schedule of each employee, from the most senior to the most junior. For a given employee, this involves finding an optimal integer solution to a so-called *residual problem.* Additionally to the sequential solution strategy, an external branch&bound tree links these residual problems to allow backtracking when necessary.

3.1 The Residual Problems

Let $\mathcal{P}$ be a Preferential Bidding problem to solve and m be the number of employees. Let $m+1$ denote the index for open time schedules and consider it as the index of

a fictitious employee. Assuming that schedules have already been assigned to the $k-1$ most senior employees, then a residual integer problem $\mathcal{IP}^k$, $1 \leq k \leq m+1$, has to be solved to find the best score schedule for the k^{th} employee.

Let P^k be the set of residual pairings, with $|P^1| = n$, and Ω_e^k be the set of residual schedules for employee e, $k \leq e \leq m+1$, with $\Omega^k = \bigcup_{e=k}^{m+1} \Omega_e^k$. Note that for $e = k, \ldots, m$, Ω_e^k is the set of normal, low time and empty schedules, while for $e = m+1$, Ω_{m+1}^k is the set of open time schedules. Let also w^k denote the residual number of low time schedules that can be constructed, with $w^1 = 2$. Five classes of parameters are defined as follows: c_s is the score of schedule s if $s \in \Omega_k^k$, and 0 otherwise; $a_{ps} = 1$ if schedule s includes pairing p, and 0 otherwise; $l_s = 1$ if schedule s is a low time schedule, and 0 otherwise; and d_s is the duration of schedule s if it is an open time schedule, and 0 otherwise. The decision variable Y_s, $s \in \Omega^k$, takes value 1 if schedule s is selected, and 0 otherwise. Given the above defined parameters and decision variables, the update formula for w^k is:

$$w^k = w^1 - \sum_{i=1}^{k-1} \sum_{s \in \Omega^k} l_s Y_s.$$

The mathematical formulation of the residual problem $\mathcal{IP}^k$ expressed as a Generalized Set Partitioning Problem is given by:

$$(\mathcal{IP}^k) \qquad Maximize \sum_{s \in \Omega_k^k} c_s Y_s \tag{1}$$

subject to:

$$\sum_{s \in \Omega^k} a_{ps} Y_s = 1, \quad \forall p \in P^k \tag{2}$$

$$\sum_{s \in \Omega_e^k} Y_s = 1, \quad e = k, k+1, \ldots, m \tag{3}$$

$$\sum_{s \in \Omega^k} l_s Y_s \leq w^k \tag{4}$$

$$\sum_{s \in \Omega_{m+1}^k} Y_s \leq 2 \tag{5}$$

$$\sum_{s \in \Omega_{m+1}^k} d_s Y_s \leq 120 \tag{6}$$

$$Y_s \in \{0,1\}, \ \forall s \in \Omega^k. \tag{7}$$

The residual problem $\mathcal{IP}^k$ consists in maximizing the score of employee k's schedule (1), while satisfying the already mentioned five subsets of global constraints: set partitioning (2), schedule assignment (3), low time schedules (4), open time schedules (5) and open time duration (6).

Optimizers. The GENCOL optimizer developed at the GERAD research center in Montréal has been used to solve $\mathcal{IP}^k$-problems. This optimizer uses a branch&bound algorithm where the linear relaxation $\mathcal{LP}^k$ of the Generalized Set Partitioning Problem is solved at each node by using column generation. This method proceeds by alternately solving the master problem and the subproblems. The *master problem* consists of $\mathcal{LP}^k$ defined on the current set of schedules. Its optimal solution is obtained by the CPLEX 3.1 optimizer (CPLEX 1992) as a core module of GENCOL and it provides dual variables used to price out new schedules. The schedules are generated by solving the *subproblems*, one being associated with each employee e, $k \leq e \leq m+1$. These subproblems handle all the local constraints. They are defined as constrained longest path problems and solved by a specialized dynamic programming algorithm (see Desrosiers *et al.* 1995a). The column generation stops when no schedules can improve the master problem solution. At this point, branch&bound starts using branching decisions and cutting planes to solve $\mathcal{IP}^k$, and new schedules are generated as needed at each node of the enumeration tree.

Since the beginning of the 1980s, the GENCOL optimizer has been used in various areas of the vehicle routing and crew scheduling field. The most recent applications realized with GENCOL, include its integration in the ALTITUDE and HASTUS systems. The first is an airline optimizer system used to solve aircraft routing, crew pairing and bidline/rostering problems (Desrosiers *et al.* 1995b). The second is an urban transit system designed for driver scheduling (Desrochers and Soumis 1989, Rousseau and Desrosiers 1995).

In the following sections, additional details in the context of our application are provided on subproblem features, resource variables, and the way to get integer solutions to $\mathcal{IP}^k$ and $\mathcal{P}$.

3.2 The Subproblems

In a given residual problem $\mathcal{IP}^k$, there is a subproblem for each employee. In each subproblem, a longest path problem with resource constraints is solved on an acyclic graph. For $e = k, \ldots, m+1$, let $G_e^k = (V_e^k, A_e^k)$ be the graph for employee e, where V_e^k the set of nodes while A_e^k is the set of arcs. The node set V_e^k is composed of $N_e^k \cup \{o, d\}$, where N_e^k consists of nodes that can be visited from source o to sink d. These last two nodes represent the beginning and the ending of the monthly period, respectively. The arcs in A_e^k correspond to residual activities of the month such as pairings, rest and training periods, etc.

At iteration k, each schedule for employee e may be represented by an $o-d$ path in G_e^k. The opposite is not true as a path may not respect all the local constraints essential for a legal schedule. Let R be a set of resources used to model the local constraints of the Preferential Bidding Problem. The number of flight credits, the number of days off and the number of consecutive working days are examples of constraint resources considered in this problem. Let t_{ij}^r be the units of resource r consumed on arc (i, j), for all $r \in R$. An interval constraint $[a_i^r, b_i^r]$ is associated with each resource $r \in R$ and with each node $i \in V_e^k$.

A marginal score is also associated with each arc. This is given in terms of the original activity scores and the dual variables provided by the solution of the master problem. For employee e, let $c_{ij}(e)$ represent the sum of weights of all bids applying to arc (i, j). Since the objective function of the residual problem $\mathcal{IP}^k$ consists solely in maximizing the score of employee k's schedule, the score of an arc depends on the subproblem considered. Given that π_{ij}^k represents the sum of dual variables applying to arc (i, j), the marginal scores $\bar{c}_{ij}^k(e)$ of arc (i, j) for employee e are computed as follows:

$$\bar{c}_{ij}^k(e) = \begin{cases} c_{ij}(k) - \pi_{ij}^k, & \text{for } e = k; \\ -\pi_{ij}^k, & \text{for } e = k+1, \ldots, m+1. \end{cases}$$

Let $X_{ij}(e)$ be the flow variables defined on the arc set A_e^k and $T_i^r(e)$, $r \in R$, be the resource variables defined on the node set V_e^k. For $e = k, \ldots, m+1$, the subproblem formulations are given by:

$$Maximize\ \bar{c}^k(e) = \sum_{(i,j) \in A_e^k} \bar{c}_{ij}^k(e) X_{ij}(e) \tag{8}$$

subject to:

$$\sum_{j \in V_e^k} X_{ij}(e) - \sum_{j \in V_e^k} X_{ji}(e) = \begin{cases} +1, & i = o \\ 0, & \forall i \in N_e^k \\ -1, & i = d \end{cases} \tag{9}$$

$$X_{ij}(e)\left(T_i^r(e) + t_{ij}^r - T_j^r(e)\right) \leq 0, \qquad \forall (i,j) \in A_e^k, \quad \forall r \in R \tag{10}$$

$$a_i^r \leq T_i^r(e) \leq b_i^r, \qquad \forall i \in V_e^k, \quad \forall r \in R \tag{11}$$

$$X_{ij}(e) \geq 0, \qquad \forall (i,j) \in A_e^k \tag{12}$$

$$X_{ij}(e) \ \ binary, \qquad \forall (i,j) \in A_e^k. \tag{13}$$

For each employee's subproblem, the objective function (8) seeks to minimize the total marginal score. Constraints of type (9) and (12) define flow conditions on the graph G_e^k, resource restrictions appear in (11) and compatibility requirements between flow and resource variables are given in (10). The solution to problem (8)–(13) is a schedule with a marginal score $\bar{c}^{\,k}(e)$. If the corresponding schedule has been computed for employee $e = k$, its real score is given by $\sum_{(i,j) \in A_k^k} c_{ij}(k) X_{ij}(k)$; otherwise, its real score is simply zero.

3.3 The Resource Variables

Resource constrained shortest (or longest) path subproblems such as the ones described in Section 3.2 have been used in many routing and scheduling applications and the modeling of several local constraints has been described in detail. This is the case for the urban transit crew scheduling problem (Desrochers and Soumis 1989, Desrochers *et al.* 1992), for the airline crew pairing problem (Desaulniers *et al.* 1996), for the airline crew rostering problem (Gamache *et al.* 1994), and for the vehicle routing problem with time windows (Desrochers, Desrosiers and Solomon 1992). The following paragraphs provide the reader with some key features of the modeling of the four local constraints described in Section 2 for the Preferential Bidding Problem.

Flight credits must be between 70 and 78. On the one hand, the upper bound is like a vehicle capacity restriction in vehicle routing problems, and as such, a resource K is used to cumulate the number of flight credits. Let K_i denote the value of the resource at node i and k_{ij} be the number of flight credits associated with arc (i,j). If arc (i,j) is used in a path, resource K is updated from node i to node

j by $K_j = K_i + k_{ij}$, provided $K_j \leq 78$. On the other hand, the strictly enforced lower bound requires an additional resource. This new resource, denoted $NegK$, is the negative of the previous one so that the restriction simply becomes $NegK = -K \leq -70$. Hence, this lower bound can also be treated as a capacity restriction (with a negative upper bound), and this resource is updated from node i to node j by $NegK_j = NegK_i - k_{ij}$, where the upper bound -70 is enforced only at the sink node d. Therefore, if $NegK_d > -70$, it means that the number of flight credits is not sufficient and the corresponding path is not valid.

This negative valued resource is necessary. To see this, suppose for example that the labels associated with two different paths ending at node i share the same marginal cost, denoted $\bar{c}$, but have different numbers for flight credits, say 65 and 69. This results in labels $(\bar{c}, 65)$ and $(\bar{c}, 69)$. In this case, $(\bar{c}, 65) \leq (\bar{c}, 69)$ and the first label clearly dominates the second label which is eliminated. Now, extend to the sink node the remaining label, with $k_{id} = 3$ units of flight credits: this label becomes invalid as it is below the lower bound of 70 whereas the eliminated label would have been legal. Keeping the negative resource gives $(\bar{c}, 65, -65)$ and $(\bar{c}, 69, -69)$ as labels at node i: no label dominates the other and both are considered for possible extension to the sink node. The first label then becomes invalid while the second is legal and hence kept.

The number of consecutive *working days* between rest periods is at most 7. This is a widely used type of constraint, especially in the airline crew pairing problem, where restrictions on the number of flight hours are enforced at the end of each duty period, i.e., before rest periods. This constraint requires a reset-to-zero feature on all rest arcs. Let the resource W be the number of consecutive working days; at each node, this resource has a value between 0 and 7. Let w_{ij} be the number of working days associated with arc (i, j). If this arc is used in a path, the resource is updated as $W_j = max\{0, W_i + w_{ij}\}$. In order to adequately define a reset function before each rest period, one has now to assign $w_{ij} = -7$ for all (i, j) rest arcs. Since $W_i \leq 7$ before any rest period, the resource value becomes $W_j = max\{0, W_i - 7\} = 0$ after the utilization of a rest arc.

As for the minimum number of flight credits, the minimum number of 10 *days off* during the month can be enforced by using a non-positive resource denoted D.

To control the *fatigue factor*, define the non-negative resource F, and let f_{ij} be the fatigue factor score on arc (i, j). This score is positive on pairing arcs, negative

on rest arcs, and zero otherwise. From node i to node j, this resource is updated by $F_j = max\{0, F_i + f_{ij}\}$. The interval bounds are set to [0, 3000] for all nodes, except for nodes that correspond to the beginning of a pairing for which the upper bound is set at 1500. In this way, the fatigue factor never exceeds 3000 during the month and is less than 1500 at the start of any pairing.

3.4 Integer Solutions

Two types of branching trees are used to obtain integer solutions to problem $\mathcal{P}$. The *internal branching tree* is directly related to the solution of $\mathcal{IP}^k$-problems, $1 \leq k \leq m+1$, while the *external branching tree* maintains the links between these problems.

Given a set of fixed schedules for employees $i \leq k-1$, solutions to $\mathcal{IP}^k$-problems are obtained within the GENCOL optimizer. If the solution of $\mathcal{LP}^k$ is integer, it provides a maximum-score schedule for employee k, and this schedule is fixed as a decision taken within the external branching tree. Otherwise, the optimal solution of $\mathcal{LP}^k$ is fractional and internal branching decisions are taken on the flow variables of the network structures, as described in Desrosiers *et al.* (1995a).

These branching decisions are stopped as soon as an integer solution is found (a schedule is selected) for employee k, regardless of the integrality for the remaining employees, i.e., for $e = k+1, \ldots, m+1$. The maximum-score schedule selected for employee k is then fixed as an external branching decision. Therefore, at a subsequent iteration, if it is not possible to find an integer solution to a residual problem, backtracking occurs in the external branching tree: an alternative maximum-score schedule for a more senior employee must be selected.

In our implementation, we choose to fix only the schedule of employee k since finding schedules for the remaining employees without taking into account their scores is useless. By choosing this strategy, we assume that there exists an integer solution to $\mathcal{IP}^k$ that includes the selected schedule. This hypothesis is validated if there is no need for backtracking in the external branching tree.

The above external tree scheme as proved to be sufficient for most problems as it is usually not necessary to backtrack. However, for the Preferential Bidding Problem, integrality difficulties come from the internal branching tree. The reason is that the integrality gap is generally very large when the fractional optimal solution for employee k is made of a convex combination of schedules of different scores. To

resolve this difficulty, we introduce special cutting planes in the internal branching tree. This is an innovative aspect of the present research.

Let Z^k_{LP} be the optimal (maximum) objective function value for problem $\mathcal{LP}^k$: $\lfloor Z^k_{\text{LP}} \rfloor$ represents an upper bound on the score of employee k's schedules for which $\mathcal{IP}^k$ is feasible. Given constraint set (3) for employee k, this maximum-score Z^k_{LP} is computed as a convex combination of scores of several schedules. If these scores are not all equal, then some are strictly greater than $\lfloor Z^k_{\text{LP}} \rfloor$ and hence, the corresponding schedules can not appear in any integer solution to $\mathcal{IP}^k$. Therefore all the schedules s for which $c_s > \lfloor Z^k_{\text{LP}} \rfloor$ can be excluded from Ω^k_k, i.e.,

$$\sum_{(i,j)\in A^k_k} c_{ij}(e) X_{ij}(e) \leq \lfloor Z^k_{\text{LP}} \rfloor \tag{14}$$

is a valid inequality (see Figure 1). Since this restriction applies only to employee k's schedules, this cutting plane can be imposed locally on employee k's subproblem. In subproblem k, such a cutting plane requires the introduction of a new non-negative resource, denoted C. This maximum-score resource is then simply treated as a capacity constraint with an upper bound of $\lfloor Z^k_{\text{LP}} \rfloor$, i.e., $C_d \leq \lfloor Z^k_{\text{LP}} \rfloor$ at the sink node. Already generated schedules for which the cost exceeds the upper bound $\lfloor Z^k_{\text{LP}} \rfloor$ are discarded from the current restricted master problem and none can be generated again. Therefore, no schedule can reappear as part of a convex combination of schedules in the new $\mathcal{LP}^k$.

Several cuts may be necessary to obtain a convex combination of equal maximum-score schedules for employee k and hence allow the use of decisions on flow variables. However, supplementary cuts only require updating the upper bound on C_d as the value of Z^k_{LP} decreases.

This type of cutting plane is compatible with the column generation scheme and, to our knowledge, it is the first time a cutting plane strategy has been used directly at the subproblem level. The importance of these cutting planes in the present application is demonstrated in the next section.

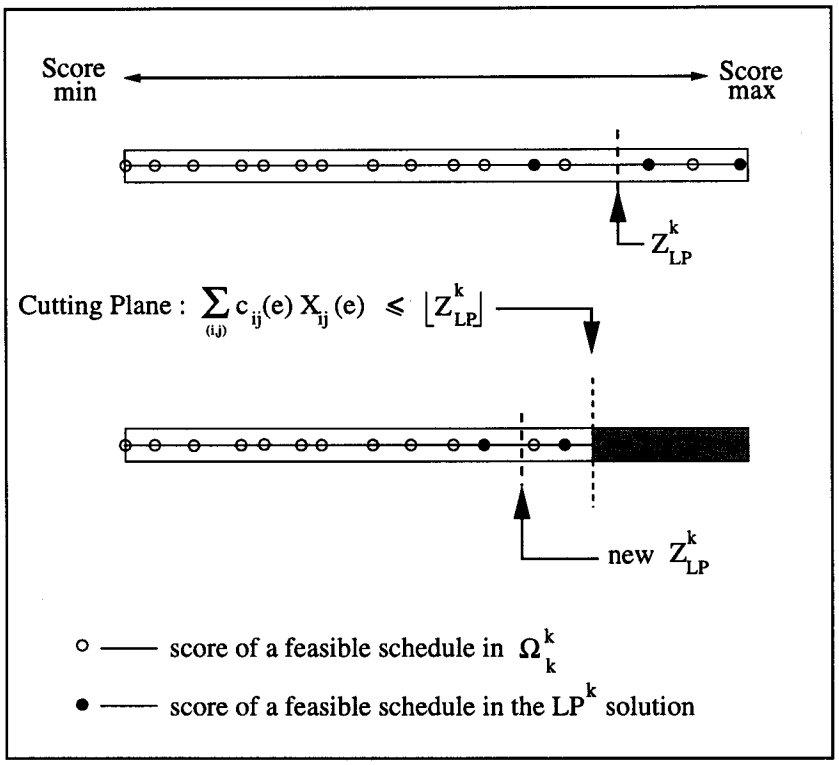

Figure 1: Illustration of a cutting plane

4 Computational experiments

The method has been applied to 24 instances provided by Air Canada, for the months of September, October and November 1994. Each instance involves the pilots of an aircraft fleet assigned to one of the four bases. Firstly, this section shows the strong impact of the cutting planes on the integrality gap and provides some statistics on the strategies used to obtain integer solutions. Secondly, it analyses the results and the solution process on the four largest test-problems. Details for the other instances are provided in the appendix.

The cutting planes described in Section 3.4 have played an important role in the solution process. Among the 24 $\mathcal{P}$-problems solved (giving rise to 710 $\mathcal{IP}^k$-problems), 15 needed such cuts on 51 $\mathcal{IP}^k$-problems. Although these cuts are not used on all $\mathcal{IP}^k$-problems, it should be emphasized that if only one $\mathcal{IP}^k$-problem cannot be solved, then neither can the corresponding $\mathcal{P}$-problem. The average relative integrality gap on these 51 $\mathcal{IP}^k$-problems is 15%, with a maximum of 99% (see

Figure 2). In one of the worst cases, Z_{LP}^k reaches 142 175 units while the optimal score is only 262 units. The branching decisions alone can not be expected to handle such a large gap. Nevertheless, very few of the special cuts are necessary in the present application to completely eliminate the integrality gap: the average taken on the 51 $\mathcal{IP}^k$-problems is only 2.3 cuts, with a maximum of 7. Table 1 presents some statistics on cuts and internal branching decisions. Additionally, the internal branching has been used on 271 $\mathcal{IP}^k$-problems using 3.5 branching decisions on average. None of the 24 $\mathcal{P}$-problems required backtracking in the external branching tree, which indicates that the strategy of fixing only the schedule of employee k instead of completely solving the $\mathcal{IP}^k$-problem has been worthwhile.

	$\mathcal{P}$-problems 24	$\mathcal{IP}^k$-problems 710
Cutting Planes	15 (63%)	51 (7%)
Internal Branching	24 (100%)	271 (38%)

Table 1: Statistics on Cutting Planes and Branching Decisions

Table 2 summarizes the results obtained on the four largest instances. The first part of the table gives the size of the problems: the number of pilots (m), the number of pairings (n), and the total number of nodes and arcs in the set of networks. The second part presents the total CPU time in hours:minutes and shows its distribution between the master problem and the subproblems. The tests have been run on a HP9000/735 with 128 MBytes of RAM. The last part of the table provides specific problem information on cutting planes and branching decisions required in the internal branching trees.

This table shows that the CPU time required to solve the largest test-problems at Air Canada ranges between 1 and 8 hours. For the smaller sized problems presented in the appendix, the solution time is on average much faster, ranging from a few minutes for several of them to almost 2 hours for the more difficult ones. The size of a problem has a great influence on the CPU time. On the one hand, the number of pilots m represents the number of $\mathcal{IP}^k$-problems to solve for a given $\mathcal{P}$-problem, while $O(m^2)$ subproblems have to be considered in the sequential process. This partly explains the rapid growth of the solution time in terms of the number of employees. On the other hand, the number of pairings n directly influences the size of the networks and therefore the time to solve the subproblems. The solution time is

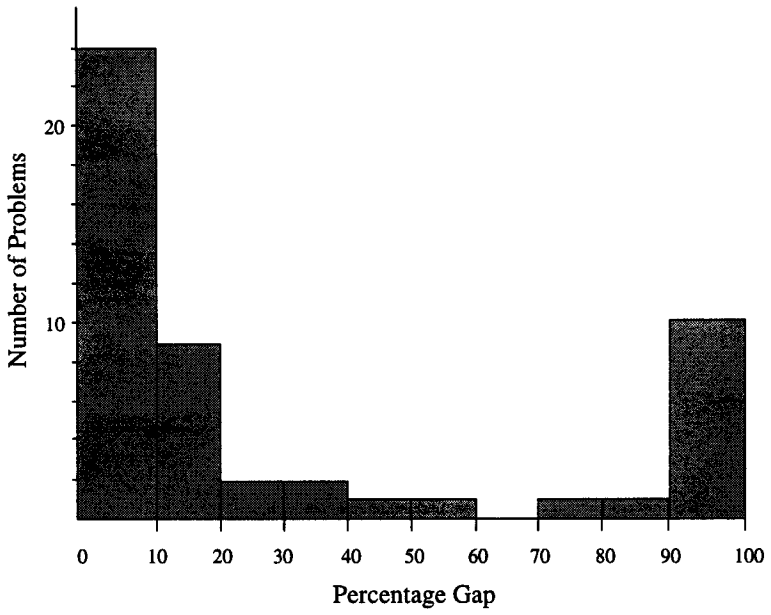

Figure 2: Distribution of the Relative Integrality Gap for 51 $\mathcal{IP}^k$-Problems

also adversely influenced by similarity among the bids selected by the employees. For example, if none of them prefer to work on weekends, then there will come a moment where the solution process will have to find solutions in the opposite direction of the ones indicated by the bid weights in order to find a feasible solution. This has the effect of slowing down the column generation process. The present computational experiments involved actual instances with quite diverse bids. Finally, the number of cutting planes and branching decisions required to obtain integer solutions to $\mathcal{IP}^k$-problems randomly influences the CPU time.

Conclusion

The approach presented in this paper improves in several ways the methods described in the literature. On the one hand, it uses a constrained longest path problem that ensures the construction of the best possible schedule for each employee. On the other hand, the formulation of the master problem as a Generalized Set Partitioning Problem ensures the compatibility of selected schedules. This research has also led to some interesting innovations such as the use of cutting planes at the subproblem level of a column generation process.

	YYZ		YUL	
	320	DC9	320	DC9
Size of Problem				
Number of Pilots (m)	108	82	46	62
Number of Pairings (n)	568	602	267	329
Number of Nodes	77 802	69 837	16 610	28 497
Number of Arcs	128 201	118 347	28 146	45 194
Solution Time				
Total CPU (hours:minutes)	5:42	8:01	1:04	1:10
Master Problem (%)	28	58	10	21
Subproblems (%)	72	42	90	79
Results				
Cutting Planes				
$\mathcal{IP}^k$-problems with Gaps	4	4	5	1
Total Number of Cuts	8	5	9	1
Internal Branching				
$\mathcal{IP}^k$-problems with Fractional Solutions	32	22	13	10
Total Number of Nodes	151	82	46	67

Table 2: Large Scale November Problems

Acknowledgments. This research was supported by the Quebec Government (Programme SYNERGIE du Fonds de Développement Technologique) and by the Natural Sciences and Engineering Research Council of Canada. The authors would also like to thank managers André Allaire at Air Canada, and Pierre Trudeau and Benoît Lacroix at Ad Opt Technologies Inc. for their valuable assistance. Last but not least, Julie Falkner was of precious help in revising the various versions of this paper.

References

J. Byrne, "A Preferential Bidding System for Technical Aircrew," *1988 AGIFORS Symposium Proceedings* 28, 87–99 (1988).

CPLEX Reference Manual. Using the CPLEX Callable Library and CPLEX Mixed Integer Library. *CPLEX Optimization, Inc.*, Incline Village, NV 89451-9436, U.S.A. (1992).

G. Desaulniers, J. Desrosiers, Y. Dumas, S. Marc, B. Rioux, M. Solomon and F. Soumis, "Crew Pairing at Air France," *European Journal of Operations Research* 97, 245–259 (1997).

M. Desrochers and F. Soumis, "A Column Generation Approach to the Urban Transit Crew Scheduling Problem," *Transportation Science* 23, 1–13 (1989).

M. Desrochers, J. Desrosiers and F. Soumis, "A New Optimization Algorithm for the Vehicle Problem with Time Windows," *Operations Research* 40, 342–354 (1992).

M. Desrochers, J. Gilbert, M. Sauvé and F. Soumis, "Crew-Opt: Subproblem modeling in a column generation approach to urban crew scheduling," in Computer-Aided Transit Scheduling, *Lecture Notes in Economics and Mathematical Systems* 386, M. Desrochers and J.-M. Rousseau (eds.), Springer Verlag, Berlin Heidelberg, 395–406, 1992.

J. Desrosiers, Y. Dumas, M.M. Solomon and F. Soumis, "Time Constrained Routing and Scheduling," in *Handbooks in OR & MS* 8, M.O. Ball *et al.* (eds.), Elsevier Science B.V., 35–139, 1995.

J. Desrosiers, A. Lasry, D. McInnis, M.M. Solomon and F. Soumis, "ALTITUDE: The Airline Operations Management System at Air Transat," *Les Cahiers du GERAD* G–95–23, École des Hautes Études Commerciales, Montréal, Canada, 1995. Forthcoming in *Interfaces.*

M. Gamache, F. Soumis, G. Marquis and J. Desrosiers, "A Column Generation Approach for Large Scale Aircrew Rostering Problems," *Les Cahiers du GERAD* G–94–20, École des Hautes Études Commerciales, Montréal, Canada, 1994. Forthcoming in *Operations Research.*

R. Moore, J. Evans and H. Ngo, "Computerized Tailored Blocking," *1978 AGIFORS Symposium Proceedings* 18, 343–361, 1978.

J.-M. Rousseau and J. Desrosiers, "Results Obtained with Crew-Opt: A Column Generation Method for Transit Crew Scheduling," *Lecture Notes in Economics and Mathematical Systems* 430, J.R. Daduna *et al.* (eds.), 349–358, 1995.

A Results for the 20 Smallest Test-Problems

The following tables present results for the smallest instances. The key is the same as the one given in Section 4. November instance YYZ-L10 has not been solved because of missing data on input.

	YVR		YWG		YYZ		YUL
	320	767	320	DC9	747	L10	767
Size of Problems							
Number of Pilots (m)	39	48	24	45	46	18	26
Number of Pairings (n)	148	178	113	277	172	103	111
Number of Nodes	10 186	13 384	5246	20 250	13 592	3996	4974
Number of Arcs	16 409	20 946	8647	32 403	21 655	7388	8355
Solution Time							
Total CPU (hours:minutes)	0:14	0:13	0:06	1:29	0:14	0:03	0:05
Master Problem (%)	4	4	4	17	4	12	5
Subproblems (%)	96	96	96	83	96	88	95
Results							
Cutting Planes							
$\mathcal{IP}^k$-Problems with Gaps	3	0	1	7	0	0	1
Total Number of Cuts	3	0	3	20	0	0	1
Internal Branching							
$\mathcal{IP}^k$-problems with Fractional Solutions	7	9	4	16	13	7	12
Total Number of Nodes	15	31	6	46	21	23	52

Table 3: September Instances

	YVR		YWG		YYZ		YUL
	320	767	320	DC9	747	L10	767
Size of Problems							
Number of Pilots (m)	39	48	25	45	46	18	26
Number of Pairings (n)	166	203	121	252	196	108	138
Number of Nodes	9 538	11 198	4867	14 954	11 021	3271	5067
Number of Arcs	15 128	17 720	8126	23 647	18 007	5966	8610
Solution Time							
Total CPU (hours:minutes)	0:10	0:24	0:04	1:13	0:30	0:02	0:15
Master Problem (%)	6	4	5	7	4	15	3
Subproblems (%)	94	96	95	93	96	85	97
Results							
Cutting Planes							
$\mathcal{IP}^k$-Problems with Gaps	2	5	1	7	8	1	1
Total Number of Cuts	2	16	3	20	21	1	3
Internal Branching							
$\mathcal{IP}^k$-problems with Fractional Solutions	11	8	5	16	12	7	10
Total Number of Nodes	29	42	7	71	46	14	15

Table 4: October Instances

	YVR		YWG		YYZ		YUL
	320	767	320	DC9	747	L10	767
Size of Problems							
Number of Pilots (m)	39	48	25	46	46	–	26
Number of Pairings (n)	177	198	138	270	162	–	134
Number of Nodes	10 817	11 251	5252	16 846	9 707	–	4855
Number of Arcs	17 027	17 682	9291	28 510	15 810	–	8046
Solution Time							
Total CPU (hours:minutes)	0:11	0:16	0:05	0:28	0:07	–	0:10
Master Problem (%)	7	4	13	21	3	–	4
Subproblems (%)	93	96	87	79	97	–	96
Results							
Cutting Planes							
$\mathcal{IP}^k$-problems with Gaps	0	0	0	0	0	–	0
Total Number of Cuts	0	0	0	0	0	–	0
Internal Branching							
$\mathcal{IP}^k$-Problems with Fractional Solutions	7	21	5	8	9	–	7
Total Number of Nodes	22	51	51	34	11	–	7

Table 5: November Instances