Annals of Operations Research 140, 305–337, 2005

Airline Crew Scheduling: State-of-the-Art

BALAJI GOPALAKRISHNAN bgopalak@ima.umn.edu
Institute for Mathematics and its Applications, University of Minnesota, Minneapolis, MN 55455

ELLIS. L. JOHNSON ejohnson@isye.gatech.edu
School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332

Abstract. The airline industry is faced with some of the largest scheduling problems of any industry. The crew scheduling problem involves the optimal allocation of crews to flights. Over the last two decades the magnitude and complexity of crew scheduling problems have grown enormously and airlines are relying more on automated mathematical procedures as a practical necessity. In this paper we survey different approaches studied and discuss the state-of-the-art in solution methodology for the airline crew scheduling problem. We conclude with a discussion about promising areas for further work to make it possible to get very good solutions for the crew scheduling problem.

Keywords: airline crew scheduling, crew scheduling, crew pairing optimization

Airline planning comprises many interdependent and challenging planning problems. A list of things that need planning are: crews, flights, maintenance, inventory, equipment purchases and crew training to name a few. Each planning problem has its own considerations, complexities, set of time horizons and objectives. One can quantitatively approach each one of these problems in order to support the planning process.

During the past thirty years there has been a phenomenal increase in problem sizes in all areas of airline planning due to rapid expansion of some of the major airlines and an explosion in the number of passengers using air transport for travel needs. This impact is most felt in the airline crew scheduling stage of airline planning. Even though crew scheduling continues to pose new challenges to the operations research community, big strides have been made in solving these problems to near optimality in many cases. Two main factors can be attributed for this: significant development in algorithms and heuristics for solving large-scale linear and integer programming problems and the advent of state-of-the-art hardware and software technologies for computational purposes.

Airline scheduling consists of *fleet scheduling* and *crew scheduling*. In this paper, we shall focus our attention on airline crew scheduling, which is widely considered to be one of the more challenging problems in airline planning. It is difficult to overestimate the importance of airline crew scheduling for the airlines. Next to fuel costs, crew costs represent the largest single cost factor for the airlines. In 1991, American Airlines reported spending $1.3 billion on crew, see Anbil et al. (1991), Northwest Airlines spent $1.05 billion in 1989, see Barutt and Hull (1990), and United Airlines spent $0.6 billion

just on pilots, see Graves, McBride, and Gershkoff (1993). Since crew costs is the biggest expenditure an airline can control, effective assignment of crews to flight legs is a very important aspect of airline planning. Much research has been devoted to scheduling of crews over the past few decades. Even small percentage savings amount to substantial dollar amount for the airlines. Some of them have reported substantial savings in operational costs by using advanced optimization techniques combining exact mathematical programming algorithms and heuristics for the airline crew scheduling problem.

Major U.S. airlines periodically schedule more than 2500 flights per day to over 150 cities using nearly 500 jets. The airline must assign a specific cockpit crew, flight attendant crew and aircraft for each of these flight legs. This involves scheduling more than 5000 cockpit crews and nearly 10000 flight attendants on a monthly basis. Further, this crew schedule must obey several FAA rules, labor contract rules and other airline specific rules. Airlines recognized the difficulty that schedulers had performing this task manually and crew scheduling was the first to receive significant attention within the OR community, see Arabeyre et al. (1969). The crew scheduling problem is relatively easy to model mathematically and interpret as an optimization problem. Still, the task of solving the model efficiently and producing a practical crew schedule can be very challenging.

Airline scheduling consists of the following five planning stages:

1. **Flight Schedule** A schedule consisting of all flights to be flown is constructed. The construction is typically based on market demands for the flight segments. For example, we may schedule a flight from Atlanta Hartsfield to Minneapolis-St.Paul airport on Monday, Wednesday and Saturday departing at 8:30 a.m.
2. **Fleet Assignment** In this stage, available aircrafts are allocated to flight legs. The revenue from a flight leg depends on the market for the flight leg and the size of the aircraft that is used for the leg. The objective is to maximize revenue with the constraint that requires all the flight legs to be flown using the fleet that is available. Several other constraints also have to be satisfied.
3. **Aircraft Routing** The aircraft routing problem involves the routing of aircraft such that maintenance constraints are satisfied, all flights flown by the fleet are covered and through revenues are maximized.
4. **Crew Pairings** A pairing is a sequence of flight legs or segments that begin and end at a crew base such that in a sequence the arrival city of a flight leg coincides with the departure city of the next flight leg. It is also referred to by some as a *trip* or *rotation*. Each pairing has a cost associated with it. The objective is to find a subset of these pairings with minimal cost that covers all the flight legs in the schedule exactly once (sometimes more than once depending on the model used for this stage). As in the previous stage a large number of regulations and other constraints apply during this stage also.
5. **Bidlines/Rosters** In this stage a monthly schedule that can be flown by the crew is drawn using the optimal set of pairings generated from the previous stage. This

monthly schedule is called a bidline (or roster) for the crew. It is called bidline because pilots can bid on the generated lines based on seniority and other considerations. This stage determines the exact number of cockpit crew members that the airline will require for the month. Again, each bidline/roster must satisfy several constraints similar to the previous two stages.

The last two stages in airline planning are usually referred to as airline crew scheduling. The crew scheduling process begins with the daily crew-pairing optimization problem. In the daily problem, all flight legs are assumed to be flown every day. After solving the daily problem, adjustments are made for weekly exceptions and crew base balancing. Once pairings are found that exactly cover every flight leg for the month, bid lines or rosters are made up for the month and are assigned by a bidding process to crew members.

Most of the excess crew cost incurred in the planning stage results from the solution to the daily problem. So from the standpoint of savings, optimizing daily pairings is the most crucial step in the process and offers the greatest opportunity to reduce crew costs through the application of OR techniques. Hence, we restrict most of our discussion in this paper to the daily crew pairing problem.

The crew pairing problem varies between different airlines. Solving the crew pairing problems for smaller airlines tends to be simpler than the problem of larger airlines, but there are also major differences between the crew pairing problems between North American airlines and their European counterparts, see Anderson et al. (1997) for a discussion on this. Crew pairing problems in North American airlines are well documented in Anbil et al. (1991) and Barutt and Hull (1990). The crew pairing problem has been solved using many and varied techniques. The approach is usually one of integer programming with binary variables as a common mathematical modeling technique for assigning crews to flights. This problem is usually approached by first generating a huge number of pairings. From this huge collection of legal pairings, a set of pairings that has minimal cost and which ensures that each flight is manned by exactly one crew is determined.

The crew scheduling problem continues to be challenging to solve due to the following reasons:

1. The number of pairings is extremely large for many airlines. It is not uncommon to generate 100 million legal pairings for moderate size fleets and several billions of legal pairings for some of the large North American fleets.
2. Many complex work rules and FAA safety regulations have to be satisfied. One such rule is the 8-in-24 rule. One version of this rule requires that not more than eight hours of flying are assigned during any twenty-four hour period (gliding time interval) unless (i) an intervening rest of twice the flying time in the previous duty period is given, and (ii) at least 14 hours of rest is given after the duty period during which the eight hour limit is violated. There are several other similar rules which the pairings have to satisfy.
3. Crew costs depend on complex crew pay guarantees and are highly nonlinear.

The rest of this paper is organized as follows: In Section 2 a more formal discussion of the crew pairing optimization problem is given. In Section 3 we discuss the state-of-the-art in crew scheduling and its impact on the airline industry. In Section 4 some future research directions for the crew pairing problem are discussed.

1. The Crew pairing optimization problem

The purpose of the airline crew pairing problem is to generate a set of minimal cost crew pairings covering all flight legs. On any flight leg crews of different categories are required. Some of the crew categories are captains, first officers, flight attendants etc.. Pilots are qualified to fly only certain aircraft types and different rules apply for each category, so that the pairing problem decomposes by crew category, see Graves, McBride, and Gershkoff (1993) for more details. The advantage in doing this is it results in smaller size crew scheduling problems with one problem for each aircraft type. The disadvantage is that deadheading (see definition below) the pilots on any aircraft may not be possible. Scheduling flight attendants is similar to scheduling pilots, but the flight attendant problem tends to be much larger since a flight attendant may be qualified to serve on more than one type of aircraft. Thus, we may not be able to consider the various aircraft types individually. Also, the rules that govern the assignment of flight attendants to flight legs may differ significantly from those of pilots. So the pilot and flight attendant scheduling problems can be treated separately. However, these two problems do have a similar general structure and hence, the pilot methodology can be applied. The focus of this paper is on the scheduling of pilots or cockpit crews since they are the most important ones due to the high salaries for these crews. In the rest of the paper, crew scheduling problem means the pilot scheduling problem, unless otherwise specified.

Before proceeding with the rest of the discussion on the crew pairing problem, we start with a few definitions. A *flight leg* or *segment* is a single nonstop flight. A *duty period* consists of a sequence of flight legs with short rest periods or *sits* separating them. Also included in the duty period are *brief* and *debrief* periods at the beginning and end of the duty period respectively. A duty period can be viewed as a single workday for the crew that is sandwiched between two overnight rest periods. A *pairing* can also be viewed as a sequence of duty periods with overnight rests between them. Each pairing begin and end at the same *crew base*, which is where the crew is stationed. In some cases the crew can fly as passengers in a pairing. This type of flight is called a *deadhead*. Deadheads are typically used to reposition a crew to a city where they are needed to cover a flight, or to enable the crew to return to their base at the end of a pairing. *Elapse* is defined as the elapsed time of a duty period including the brief and debrief periods. *Flying time* is defined as the total number of hours of actual flying time. Time away from base (TAFB) is defined as the total elapsed time including the overnight rests between duty periods in a pairing.

1.1. Rules

For a solution to the crew pairing problem to be considered feasible, the pairings must obey FAA regulations, union contract requirements and other airline specific rules. Such restrictive rules help reduce the size of the problem but can make the problem of determining an optimal crew schedule extremely complex. In this section we discuss some important rules governing legal pairings.

Within a duty period there are prescribed maximum and minimum sit times between flight legs called *maxsit* and *minsit* respectively. Typical values for maxsit and minsit are 4 hours and 45 minutes respectively. The elapsed time in a duty period must be less than an allowable time limit called *maxelapse*, which is usually 12 hours. The total actual flying time in a duty must not exceed *maxfly*, which is typically 8 hours. A duty conforming to these rules is said to be legal or valid. Similarly a valid or legal pairing must satisfy some rules. Legal pairings may be composed of up to a maximum number of duty periods called *maxduties*. A pairing must allow a minimum number of hours of rest between duties, called *minrest*. This *minrest* may need to be extended when the flying time in a twenty-four hour period exceeds eight hours.

1.2. Problem categories

There are three types of crew pairing problems: daily, weekly, and the fully dated problem. In each of these problems we want to find a set of pairings for a given crew category and fleet type with minimal cost that covers all the flight legs. We start with a discussion on the daily problem.

1. **The daily problem**
   In the daily problem all flights are assumed to be flown every day of the week. This type of problem closely resembles flight operations in North American airlines but is quite different from European airlines where the flight schedules are more irregular. So most North American airlines seek a good solution for the daily problem. A solution to the daily problem consists of a set of pairings such that all the flight legs are covered on a single day. From this solution a dated version for the required time horizon is obtained by repeating itself after shifting the entire pairing by a day. This will ensure that all flights on all days of the time horizon are covered.
2. **The weekly problem**
   The assumption of the weekly problem is that the timetable repeats itself every week. The weekly problem is usually solved by first solving the daily problem. In most North American airlines the task is usually to modify the daily solution to fit the weekly exceptions. This typically results in a good weekly solution. In European airlines the weekly exceptions are many and modifying the solution of the daily problem may result in a solution which is far from optimal.

3. **The dated problem**
   The time horizon for the dated problem is usually a month. In order to solve this problem one must handle differences between weeks. Just as the daily solution may not be feasible for the weekly one, it may not be possible to extend the weekly solution for the dated problem due to holidays and changes of time table. Changes of time table occur due to flight cancelations or introduction of new flights during some week in the month. The dated problem is a more challenging problem than both the daily and weekly problems. In this problem the regularity property of the daily and weekly problems may not exist and can result in an explosion in the size of the problem. It can also lead to poor solutions if we a try to use the weekly solution to solve the dated problem.

Most of the discussion in this paper will be on the daily crew pairing problem since it is the most commonly solved problem among the three problems described above. Also a significant portion of published research on crew pairing optimization is on the daily problem. In the next subsection we present an example of a pairing problem.

1.3. An Example

Table 1 shows a flight schedule with six flights to be flown every day of the month. Each row in the flight schedule corresponds to a single flight leg. Sometimes two or more flights combined to be flown by the same crew are shown as a single flight leg. Each flight leg is uniquely identified by a flight number, departure station and time, and arrival station and time. In this example we assume that A is the only crew base. So all legal pairings have to start and end at A. Also we assume all times to be with respect to crew base time.

For the sake of illustration we follow some simple rules to classify legal pairings in this example. In reality rules that govern legal pairings are very complex. We consider a pairing to be legal if it satisfies the following three rules: (a) flying time of a duty is at most eight hours and (b) number of duties in a pairing is at most two (c) The overnight rest (OR) between duties must be at least nine hours. The total flying time of a pairing

Table 1
Flight schedule.

Leg number	Dep. station	Dep. time	Arr. station	Arr. time
1	A	08:00	B	13:00
2	B	15:00	A	20:00
3	A	07:00	C	10:00
4	C	12:00	A	15:00
5	B	07:00	C	10:00
6	C	11:00	B	14:00

Table 2
Legal pairings.

Pairing number	Flight leg sequence	Pairing cost
1	1⇒ OR ⇒ 2	26
2	1 ⇒ OR ⇒ 5 ⇒ 4	20
3	3 ⇒ 4	2
4	3 ⇒ OR ⇒ 4	26
5	3 ⇒ 6 ⇒ OR ⇒ 5 ⇒ 4	20
6	3 ⇒ OR ⇒ 6 ⇒ 2	26

is defined as the aggregate of the flying times of all the legs in the pairing. The flying time of a leg is the difference between the arrival time and departure time of the leg. As defined before, a duty is defined as a day's worth of work for the crew.

The cost of a pairing in this example is defined as the difference between the time away from base and the flying time. The cost of a pairing in this case is basically the number of hours in the pairing that the crew spends without flying. This is sometimes referred to as the pay-and-credit.

Table 2 shows the list of all legal two day pairings originating from the crew base A. We have six legal pairings and against each one of them we have a cost assigned to the pairing. The first legal pairing in Table 2 has cost 26 because the time away from base for this pairing is 36 hours (from 8 a.m the first day to 8 p.m the following day) and the total flying time is 10 hours. Each pairing will be flown by one crew but the solution to the crew pairing optimization problem does not say which crew will be flying these pairings or how many crews are required to repeat the schedule on all days of the month. This will be determined at the bidline (or roster) generation phase. At the crew pairing optimization stage we are interested in finding a partition of the flight legs using legal pairings that have minimum cost. The daily cost of operating the flight legs in the schedule is equal to the cost of the daily crew pairing solution. In Section 2.5 we describe a commonly used model for solving the crew pairing optimization problem.

1.4. Cost structure

The FAA and union contract rules define the structure and cost of legal duty periods and pairings. The salaries of crew dominate the overall personnel costs. The crew pairing model considers only the "pay-and-credit" cost, i.e., all the incremental costs such as the cost associated with the length of time the crew is away from base, and deadheading. Basically it is the excess cost in addition to the flying time. Besides the fuel cost, which is fixed per legal pairing, the main causes for excessive cost of a pairing are (a) long and frequent sit times within a duty period (b) long rest periods (c) deadheading.

The cost of a pairing is typically computed using the following formula: max{#duties *PMDG, TAFB_Factor * TAFB, Total_Duty_Cost}. PMDG or pairing

minimum duty guarantee is the minimum number of hours that the crew is guaranteed to be paid for each duty in a pairing irrespective of the length of the duty period, TAFB_Factor is a fraction associated with TAFB and Total_Duty_Cost is the sum of the costs of all duties in the pairing. The duty cost is computed using a formula similar to the pairing cost. It is given by: DUTY_COST = max{MDG, Elapse_Factor * elapse, flying time}, where the minimum duty guarantee (MDG) is the guaranteed minimum number of hours of pay for the crew in a duty and Elapse_Factor is a fraction associated with the duty elapse time. Typical values for some of the cost parameters are PMDG = 5 hours, MDG = 3 hours, TAFB_Factor = 2/7 and Elapse_Factor = 4/7.

The total hours of flying in a schedule will be a lower bound on the cost of a given schedule. Pairings that have high TAFB relative to the total flying time in the pairing are expensive pairings. However, a few such expensive pairings may be necessary to fly all the flight legs in the schedule with least cost. The main objective of crew pairing optimization is to find a set of pairings that cover all the flight legs exactly once and has cost as close as possible to the total flying time of the schedule.

1.5. The Set partitioning model

Any solution to the crew pairing problem must satisfy some constraints and these have to be taken into consideration in the mathematical model. Clearly all the pairings in the solution must be legal, i.e., they must satisfy all the FAA rules and other rules that govern legal pairings. Since it is almost impossible to characterize mathematically a legal pairing, one must enumerate a pairing to check for legality. Usually the crew pairing problem is solved in two steps: first all legal pairings are generated and their costs computed, and then a good subset of these pairings is chosen to cover all the flight legs. Smart enumeration and compact storage schemes for pairings have been developed for some moderate size pairing problems, see Hu (1996) and Hu and Johnson (1999). The basic set partitioning model described below assumes that all feasible pairings have been explicitly enumerated and their costs computed. In reality it may not be possible to generate all legal pairings apriori since there are too many of them. A column generation scheme is usually employed for the implicit generation of legal pairings as and when required.

The crew pairing problem of finding a set of pairings with least cost such that each flight leg is covered by exactly one pairing can be formulated as a set partitioning problem (SPP) shown below:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} c_j x_j \\ \text{s.t} \quad & Px = e \\ & x_j \in \{0, 1\} \quad \text{for } j = 1, 2, \ldots, n \end{aligned}$$

The right hand side vector e is a vector having m entries equal to one. Each row of the matrix P represents a flight leg and each column represents a legal pairing. The cost vector $\boldsymbol{c}$ is such that c_j is the cost associated with the jth column or pairing. The binary decision vector $\boldsymbol{x}$ is such that x_j is a 0-1 variable associated with the jth pairing. If column j is selected then $\mathrm{x}_j = 1$ and 0 otherwise. The matrix P is constructed as follows:

$$p_{ij} = \begin{cases} 1 & \text{if flight leg } i \text{ is covered by pairing } j, \\ 0 & \text{otherwise.} \end{cases}$$

Additional constraints on the crew-scheduling problem called the "crew base constraints" may be required at times. These constraints basically restrict the total number of flying hours that a crew can spend away from its base location to be within specified limits. These constraints significantly constrain the allocation of available crews among flights and contribute to the difficulty of solving the crew scheduling problem. These constraints can be modeled as follows:

$$L_k \leq \sum_{j \in BASE_k} TAFB_j x_j \leq U_k$$

$BASE_k$ represents the set of all pairings that start at the kth crew-base, and k varies over all the crew-base stations. $TAFB_j$ is the time away from base for pairing j and L_k and U_k are the lower and upper limits (in hours) for the total time away from base for all the crews stationed at the kth crew base station.

For the pairing problem discussed in Section 2.3 the cost vector $\mathbf{c}$ is given by

$$c = [26 \quad 20 \quad 2 \quad 26 \quad 20 \quad 26]$$

and the matrix P is given by

$$P = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

The decision vector is $x \in \{0, 1\}^6$.

The LP dual of (*SPP*) is

$$\begin{aligned} &\min \boldsymbol{\pi}\mathbf{1} = \Sigma_i \boldsymbol{\pi}_i \qquad (D\text{-}SPP) \\ &\text{s.t } \boldsymbol{\pi}\mathbf{P} \leq \mathbf{c}. \end{aligned}$$

Given all the pairings and their costs, an initial feasible solution π^1 to the dual problem (*D-SPP*) can be obtained by finding the minimal ratio of the cost of pairing and

the number of flight segments in it, i.e., we can use

$$\pi_i^1 = \min_j \{\mathbf{c}_j / \# \text{ of flight legs in pairing j}\} \quad \text{for } i = 1, 2, \ldots, m. \tag{1.2}$$

Since the cost of a pairing is never lesser than the pay for the actual flying time of a pairing, we can utilize the flying times of flight segments to find a better initial dual feasible solution. We can let the initial dual feasible solution π^2 be

$$\pi_i^2 = f \text{ x (flying time of flight i)} \quad \text{for } i = 1, 2, \ldots, m \tag{1.3}$$

where f is chosen as large as possible subject to π^2 being dual feasible.

1.6. Handling deadheads

A straightforward approach to allow deadheading in the crew pairing model is to use the set-covering formulation instead of the set-partitioning model used above. In such a situation, one occurrence of the row represents a flying leg for a crew and others are treated as deadheads. Since regular flight legs and deadheads tend to have different costs it can result in some distortion in the column cost for a pairing with deadheads. This is because the optimzer does not differentiate between a regular flight and a deadhead. This in turn can result in a suboptimal solution for the crew pairing problem. The set covering model can be useful when the cost of deadheading versus flying a flight leg are almost the same, see Bornemann (1982). Alternately, deadheads can also be modeled using the set-partitioning approach, see Rubin (1973) and Anbil et al. (1991). In this case some pairings can have missing flight legs. The missing flight legs can be considered as deadheads in the pairing. The additional cost incurred when the crew uses the deadheads to fly as passengers by either using a flight on the same airline or another airline, can be factored into the cost of the pairing. In order to generate pairings containing deadheads, one must include all possible flights that can be used as deadheads in the flight schedule before generating the legal pairings. However, this is usually not done because it can not only result in an explosion in the size of the problem but also because only a small fraction of the deadheads will be used in the final solution. One usually uses historical data and other techniques to select a small fraction from the set of all possible deadhead flights. Once a legal pairing is generated and its cost computed taking into account the presence of deadheads, the deadheads do not have to be included in the set-partitioning model. Hence they are not provided to the optimizer as a part of the set-partitioning model. The advantage in this approach of handling deadheads is the number of rows in the set-partitioning model does not increase. The disadvantage is new columns that represent pairings with deadheads have to be included in the model and for some problems this can be a very large. Also generating legal pairings takes more time since all flights considered as deadheads are distinguished from flying legs of the flight schedule.

In the approach of Barnhart, Hatay, and Johnson (1995), all deadhead possibilities can be considered using a network where an arc represents a flight leg, see Lavoie,

Minoux, and Odier (1988). This methodology was shown to work well for the long-haul crew pairing problem. All flights that either originate or terminate at a station not in the flight schedule are eliminated. After this, for each flight that can be used as a deadhead, a pairing that contains this flight and prices out properly is considered. If no such pairing exists, the flight leg is dropped from the set of deadhead possibilities. The process of choosing such a pairing can be done efficiently by calculating a partial pairing with minimum cost that starts at the crew base and ends at the departure station of the deadhead flight leg. Another partial pairing with minimum cost originating now at the departure station of the deadhead flight leg and ending at the crew base is calculated. Combining these two partial pairings gives a pairing with the combined cost of the two partial pairings. If this cost prices out favorably the deadhead is chosen. The deadheads obtained using this method for a major North American airlines were very different from the set of deadheads considered by approaches discussed above. Also the integer feasible solution obtained using the set of deadheads obtained using this method was reported to be significantly better than previously used methods.

1.7. Network structure

Billions of legal pairings can be generated for as few as 1000 flight legs. One reason for this is due to the *hub-and-spoke* network structure of most airlines in North America. A hub is a central airport that flights are routed through, and spokes are the routes that planes take out of the hub airport. Most major airlines have multiple hubs. The schedule is constructed such that passengers arriving at a hub can make connections to many outgoing flights without much delay. This means that a large number of flights arrive at the hub within a short time interval and also depart the hub in short interval of time. The purpose of the hub-and-spoke system is to save airlines money and give passengers better routes to destinations but this kind of network structure leads to an explosion in the number of legal pairings that can be constructed, see Graves, McBride, and Gershkoff (1993).

A good example of a hub-and-spoke system is that of Delta Airlines, for which Hartsfield Atlanta International Airport is one of its hubs. In order to fly from Charleston, SC, to Memphis, TN the airline flies from Charleston to Atlanta, and then from Atlanta to Memphis via a connecting flight since there is probably not a lot of demand for a Charleston-to-Memphis flight. The hub-and-spoke system is not very common among European airlines and therefore the number of pairings tend to be smaller for crew scheduling problems arising from European airlines.

2. State-of-the-art solution methodologies

A variety of solution methodologies have been suggested for the crew pairing optimization problem over its nearly three decade history. The problem is usually divided into two distinct phases: (a) pairing generation, and (b) optimization. Although pairing generation

is simple, even for relatively small instances several millions of legal pairings can be generated. This makes it extremely difficult to identify a relatively small set of legal pairings with minimum cost to cover all the flight legs in the schedule.

In this section we survey some of the recent advances that have been made in solving the crew pairing problem. A survey on some of the past work on airline crew scheduling can be found in Arabeyre et al. (1969), Etschmaier and Mathaisel (1985), and Gershkoff (1989). The set-partitioning model is one of the commonly used models for solving the crew pairing problem. But alternate models have also proven to be successful, see Desaulniers et al. (1997) and Vance et al. (1995). Most of our discussion in this section is restricted to advances made in pairing generation and solving the resulting set-partitioning problem. At the end of the section we discuss some alternate formulations for the crew pairing problem and the corresponding solution methodologies used.

Most crew pairing systems use one of three broad approaches described below to generate legal crew pairings in order to derive the final schedule for the crew. The performance of these systems depends mainly on techniques for obtaining optimal or close to optimal integer solutions for the crew pairing set-partitioning problem. The efficiency of such techniques rely on three things: the algorithm that is used to solve large-scale linear programs efficiently, clever branching heuristics and efficient cut and column generation routines.

2.1. Three Approaches to pairing generation

1. **Row approach**
   One of the earliest approaches has been to apply local optimization to an initial feasible solution for the crew pairing problem. Such a solution is usually available as a warm start solution obtained from an earlier run of a similar problem, see Rubin (1973). If no such solution is available, efficient heuristics can be used to obtain an initial feasible solution. Note that an initial feasible but very expensive solution would be to use a crew for each flight leg. A small number of pairings are chosen, almost randomly from the current pairing solution. A subproblem consisting of all possible legal pairings that can be generated using flight legs from the selected pairings is formed and an integer optimal solution for the subproblem set-partitioning problem is computed. Since the integer optimal solution for the subproblem cannot be worse than the old solution for the subproblem, it is maintained for the next iteration. That is, the optimal columns of the subproblem replace the set of columns from which it was generated. The random selection of columns begins the next iteration which creates a new subproblem for the optimizer to solve. This process continues until a certain time has elapsed or significant improvement in the objective function has not been obtained for several iterations. The disadvantage of such a local search method is it may end in a local minimum. Some heuristics can be used to move the solution out of a local optimal solution. Anbil, Johnson, and Tanga (1991) present a method to handle more than 5 million columns and report excellent savings over a local improvement

approach. In Housos and Elmroth (1997), the authors present an iterative scheme which is reported to be very successful. However, it is not clear if these iterative methods can produce global convergence.

Work on crew pairing reported in Anbil et al. (1991), Baker, Bodin, and Fisher (1985), Ball and Roberts (1985), Etschmaier and Mathaisel (1985), Gershkoff (1989), Graves, McBride, and Gershkoff (1993), and Rubin (1973) is based on the row approach. Some crew pairing systems that have used this procedure to solve the crew pairing problem are the TPACS system (Rubin, 1973) previously used by several major airlines including United Airlines, the TRIP system (Anbil et al., 1991) used by American Airlines and Continental Airlines and the ALLPS system (Gerbracht, 1978).

2. **Column approach**

In this strategy the columns in the subproblems are generated by considering all the rows simultaneously. It may not be possible to select the columns for the subproblem after generating all legal pairings since billions of legal pairings can be generated in some instances. Some form of column generation approach is used to generate candidate pairings for the subproblem. Usually, one selects for the subproblem, a representative number of pairings that have small reduced cost from a relatively large set of legal pairings. Once the subproblem is obtained an optimizer is called to solve the subproblem to optimality. A new subproblem that has a better objective value than the current one is formed in the next iteration. The process of subproblem generation and optimization continues till an optimal solution to the whole problem is obtained. Hu and Johnson (1999) use this approach after explicitly generating all the columns.

A combination of the row approach and column approach has also been considered. The row approach has the advantage that the process can be terminated at any step with an integer feasible solution. The column approach does not have such an attractive feature, but one can obtain a global optimal or close to global optimal solution using the column approach.

3. **Network approach**

Problems involving the movement of vehicles can usually be formulated as routing problems in graphs or digraphs, where the vertices are associated to locations, the edges or arcs are associated with connections between locations and their length describes the cost of using such a connection. Recently many crew pairing systems have started using a network approach for generating legal crew pairings. The network approach is basically the column approach where the columns are generated using the flight leg or duty network. This approach was first introduced by Lavoie, Minoux, and Odier (1988) and has since been used by many crew pairing systems. A time-space network of flights is used. There is an arc in the network corresponding to a flight in the flight schedule. Overnight rests and sits are also represented as arcs in the network. The costs associated with the arcs are usually linear with respect to time spent on the arc. Given such a network one can generate all legal pairings by generating paths that start and end at crew bases. Any two flight legs on this path must satisfy the property

that the departure time of the second flight is later than the arrival time of the first. All other rules that govern legal pairings must also be satisfied.

Legal pairings can also be generated using a Dantzig-Wolfe column generation technique, see Dantzig and Wolfe (1960). A subproblem consisting of a relatively small set of legal pairings is solved and using the dual values corresponding to the flights in the pairing, additional columns are selected in an iterative procedure. The legal pairings are generated by solving a shortest path problem on the above network. If a legal pairing corresponding to the shortest path that originates and returns to the crew base has a negative reduced cost, it is selected for inclusion in the next subproblem. The new linear program has a better objective function than the previous one. This iterative procedure terminates when the shortest path has a nonnegative reduced cost. At this point the current solution to the linear program is optimal. The advantage of this method is it can provide an optimal solution to the LP relaxation of the crew pairing problem without generating all legal pairings apriori. However, an integer optimal solution is not guaranteed since some of the pairings may not be priced properly using the dual values corresponding to the flight legs. To overcome this, one has to generate and add all legal pairings whose reduced cost is less than the gap between an optimal LP solution and a known feasible integer solution. If this gap is large, a huge number of pairings have to be generated. For large problems the size of and time to create the network can also be prohibitive.

Barnhart et al. (1994) show that long-haul crew pairing problems can be solved efficiently using this approach. This is because the network is smaller in size compared to short-haul problems and TAFB is a good measure for the cost of a pairing. The network approach is used in the crew pairing system in Air France, see Desaulniers et al. (1997). For short-haul problems a duty network can be used in place of the network of flight legs, see Anbil, Forrest, and Pulleyblank (1998), Chu, Gelman, and Johnson (1991), Desrosiers et al. (1991), Lavoie, Minoux, and Odier (1988), and Vance et al. (1995). In this network, each arc represents a potential duty period including overnight rests for the crew. Such a network can handle a more nonlinear cost structure since the cost of the duty periods can be computed using nonlinear cost functions. Such an approach can provide optimal integer solution for the short-haul crew pairing problem. In Barnhart, Hatay, and Johnson (1995), the authors discuss a method for iteratively expanding the duty network.

2.2. Successful solution strategies

Several solution methodologies have been proposed for solving the crew pairing optimization problem. The software currently in use generates legal pairings without explicitly enumerating all of them. This is done by solving relatively small set-partitioning subproblems to optimality and improving the solution further by generating "good" legal pairings using the optimal dual solution from the subproblem. The success of a crew pairing system depends on efficient column selection strategy, fast linear programming

codes to solve the LP relaxation of the subproblems and clever branching strategies. In this section we focus our attention on the state-of-the-art solution methodologies. We also briefly discuss alternate formulations that have resulted in efficient algorithms for solving the crew pairing problem. Most of these techniques have either been developed in the past decade or developed earlier than that but have since been refined and enhanced for superior performance.

2.2.1. TPACS/TRIP approach

Trip Pairing for Airline Crew Scheduling (TPACS) was a creative solution developed about three decades ago by Jerry Rubin (1971, 1973) for the crew pairing optimization problem. This project was started in the IBM New York Scientific Center in consequence of some new Integer Programming, and Set Covering, Set Partitioning algorithms and codes developed by Lemke, Salkin, and Spielberg (1971) and Lemke and Spielberg (1967) and then completed, and TPACS sold to nine major American airlines, in cooperation with the IBM Airlines Marketing group, after a move of the Scientific Center to Philadelphia. TPACS was used by several major airlines for crew scheduling. It resulted in improvement over previous scheduling methods and resulted in substantial savings on operational costs for the airlines. The development of TPACS can be regarded as an outstanding development in crew pairing technology. The trip re-evaluation and improvement program (TRIP) is based on TPACS. The TRIP procedure starts with an initial set of pairings. It iteratively improves this solution by generating subproblems using the row approach. In each iteration roughly five pairings are chosen and all legal pairings using the flight legs in the selected pairings are generated to form a subproblem set-partitioning problem. The set partitioning subproblem is then solved to optimality.

Over the years several enhancements have been made on the TRIP methodology making it one of the most successful techniques used for solving the crew pairing problem, see Anbil et al. (1991). It has been improved to handle weekly pairings that satisfy crew base constraints. TRIP technology has also been improved in subproblem selection, pairing generation and subproblem optimization. These enhancements have translated into several millions of dollars of savings for American airlines. TRIP iteration speed has improved significantly by improved hardware and software technologies. It uses a hybrid algorithm using linear programming and Lagrangian relaxation for solving the set-partitioning subproblems. This algorithm quickly determines a good set of Lagrangian multipliers and a good warm start solution is obtained for the LP. The current TRIP solver is reportedly several times faster than previous versions of TRIP.

Recent versions of TRIP can generate more than 5000 legal pairings in a second and can solve subproblems with more than 100000 columns quickly using column screening techniques. The basic idea behind the column screening approach is to consider pairings that have reduced cost less than some threshold value. The reduced costs are computed efficiently using heuristics based on Lagrangian relaxation. TRIP has been used by American Airlines to obtain globally optimal solutions for some of the smaller fleets and close to optimal solutions for the larger ones.

A major drawback of the TRIP approach is the presence of local optimal solutions due to the subproblem approach that is adopted in this method. This can prevent further improvements in the solution. Progress has been made to address this problem by developing some heuristics to minimize the impact of local minima. One such technique is to allow TRIP to simultaneously consider alternative optimal pairings for the subproblems and the solution paths they generate. This minimizes the possibility of obtaining a local minima because selecting a single solution path may lead to a local minima trap. Another technique is to solve larger subproblems since local minima for larger subproblems are likely to be closer to the global minima for the entire problem.

The TRIP system (or some variations of it) is being used by other major airlines such as Continental Airlines. The single biggest impact on cost savings due to the recent enhancements to TRIP has been its ability to solve large subproblems quickly. ALLPS system (Gerbracht, 1978) marketed by Unisys and previously used by Northwest Airlines, uses a TRIP like approach for solving the crew pairing problem. The success of the TRIP approach can be attributed to its ability to produce feasible solutions, its robustness and the simplicity of the method that makes it easy to implement.

2.2.2. Linear programming algorithms

Some of the successful crew pairing systems have used rapid solution to linear programming relaxations of the crew pairing problem with column generation and specialized branching. In this subsection we discuss two linear programming algorithms that have proven to be very successful in practice for solving crew pairing problems.

1. **SPRINT**

   Solving the LP relaxation of large crew pairing problems can be very time consuming. To tackle this problem (Forrest, 1989) introduced the so-called SPRINT method for solving large-scale linear programming problems quickly. It has been successfully used by some crew pairing systems to solve large-scale crew scheduling LP's. Using the SPRINT method the LP relaxation of crew pairing problems with nearly 850 flight legs and 5.5 million legal pairings have been solved much faster than using some of the traditional methods. The SPRINT iteration counts were quite high, but each iteration was performed very fast.

   The main idea behind the algorithm is to generate the optimal solution of a large scale linear program by solving a sequence of considerably smaller linear programs. In the SPRINT method a small subset of the columns of the problem is chosen and a linear program is solved on this subproblem. Using the optimal dual solution all columns of the original problem are priced out. If the current solution is not optimal for the original problem, then there are columns having negative reduced cost. A new problem is formed by keeping the columns in the optimal basis and then adding a small subset of the columns that have close to zero reduced cost. This is done using a bucket data structure. New columns are selected from buckets based on reduced cost, adding columns from the best buckets. The process continues till an optimal solution for the original problem is obtained.

Recent enhancements to SPRINT has resulted in improved and more robust performance in solving these large-scale LP's. One of the improvements was a slight perturbation of the RHS in the original large-scale LP; since the solution of the perturbed problem is 'almost always' still feasible for the original problem, it is also optimal for the original LP. More precise selection of best reduced cost columns have also resulted in improvements to the performance of SPRINT.

2. **Volume Algorithm**

The volume algorithm developed by Barahona and Anbil (1997) is an extention to the subgradient algorithm to produce primal as well as dual solutions. It is well known that the subgradient algorithm has been used successfully to produce lower bounds for large-scale linear programs, see Held and Karp (1970, 1971) and Held, Wolfe, and Crowder (1974). Due to its low computational cost the subgradient method has proved to be attractive. It can be used to produce very good approximations to optimal dual solutions quickly. The primal solution can be obtained using complementary slackness conditions or using a combinatorial method. Some results on obtaining primal solutions using a subgradient method have been published, see Larzon and Liu (1989) and Shor (1985).

Convergence is an issue with the subgradient method since it does not have well defined stopping conditions. The process is usually terminated based on a limit for the number of iterations or the number of steps without an improvement. Some subgradient algorithms that converge in theory are computationally expensive and those that are successful in practice have no known proof of convergence. The volume algorithm produces approximations to the primal and dual solutions that can be used themselves or used to obtain exact solutions using an exact method. The volume algorithm has good stopping criterion and low computational cost per iteration. It has shown to be very successful in solving large scale set-partitioning and set-covering linear programming problems.

The volume algorithm has similarities with the conjugate subgradient method when dealing with the dual variables, see Lemarechal (1975) and Wolfe (1975). However, in order to obtain the primal solution the volume below the faces that are active at the dual optimal solution is estimated. This volume provides the primal solution. A theorem on LP duality proposed by Barahona and Anbil (1997) is used for justifying their approach in obtaining the primal solution using this method.

Computational results show that the volume algorithm combined with the dual simplex method is very successful in solving the LP relaxation of large-scale set-partitioning problems. This is done by first computing the approximate dual solutions using the volume algorithm and computing the reduced cost of a small set of columns. The dual simplex method is then applied to solve this smaller problem using the reduced cost instead of the original objective. After this, all the remaining columns are priced out. Using the combination of the volume algorithm and the dual simplex algorithm (Barahona and Anbil, 1997) show that an instance of a set-partitioning problem from an airline crew scheduling problem with 2504 rows and 50722 columns

can be solved nearly 20 times faster than the dual simplex algorithm and nearly 10 times faster than an interior point method.

An implementation of the volume algorithm has been successfully used as a subroutine for solving the linear programs encountered when solving the crew pairing problems in a crew pairing system developed recently by IBM for US Airways and Southwest Airlines, see Anbil, Forrest, and Pulleyblank (1998). The volume algorithm has also been used as a crash procedure for the simplex method when the SPRINT method described above performs poorly. The dual solutions obtained by the volume algorithm have a large number of nonzero variables and this tends to be good for column generation. This property is not shared by the simplex algorithm which produces basic solutions.

2.2.3. Integer programming methodologies

In this section we discuss some strategies for obtaining good integer solutions for the crew pairing optimization problem. The size of the crew pairing problem makes it almost impossible to search for an integer solution using traditional branch-and-bound methods on the entire problem. Hence finding a good integer feasible solution for the crew pairing problem is usually restricted to a relatively small subset of columns that have the small reduced cost. Even on this restricted set it may be impossible to fathom the entire brach-and-bound tree. This has lead to the development of several heuristic solution techniques. Some of them such as the branch-and-cut algorithm that is discussed in this subsection have been used to obtain integer solutions to proven optimality. The LP relaxation of small set-partitioning problems generally produces integer solutions but even moderate size set-partitioning linear programs have highly fractional LP solutions, see Gershkoff (1989) and Marsten and Shepardson (1981). This makes the problem of finding integer solutions even harder. Recent advances in solution methodologies for solving set-partitioning and set-covering problems have contributed to the advancement of solution methodologies for the crew pairing optimization problem, see Balas and Padberg (1976), Balas and Ng (1986), Cornuejols and Sassano (1989), Lemke, Salkin, and Spielberg (1971), and Lemke and Spielberg (1967).

1. **Branching Techniques**

   Choosing the appropriate branching rule can be crucial for solving large-scale integer programming problems such as the crew pairing problem. We discuss below three branching rules for the crew pairing optimization problem that have been used successfully in some crew pairing systems.

   (a) **Follow-on Branching**

   The *branch on follow-ons* branching heuristic was introduced by Ryan and Foster (1981). This branching strategy was first used for solving general set-partitioning problems, but has since been used successfully in many crew pairing solvers.

   A *follow-on* is the second of a pair of consecutive flights flown in a pairing. First, the LP relaxation of a subset of pairings of the crew pairing optimization

problem is solved to optimality. Typically several millions of good pairings are considered. Let r and s be any two flight legs. On one branch, called the follow-on branch, r and s are forced to appear consecutively in a pairing with s following r. This includes wraparound so the last flight leg is followed by the first. On the other branch, the non follow-on branch, the two legs cannot appear consecutively. In Vance et al. (1997) the follow-on branching rule was shown to be a valid one, i.e., if F_{rs} denotes the set of all pairings for which flight leg r and s appear consecutively in that order and if x^* is an optimal fractional solution to the LP relaxation of (SPP), then there exist two flight legs r and s such that

$$0 < \sum_{i \in F_{rs}} x_i^* < 1 \tag{2.1}$$

The summation in (3.1) is called the weight of the follow-on pair r and s. The follow-on branching rule can be used by computing the follow-on weights for all pairs of flight legs r and s and fix follow-on's for which the weight is 1 and also a follow-on which has the maximum weight less than 1. Assuming that the basis columns are the ones that have nonzero values there are at most m columns to search, where m is the number of rows (flight legs). This procedure will cut off the current fractional LP solution. The size of the problem decreases by fixing a follow-on; several columns can be discarded and the row corresponding to the follow-on flight leg can be removed as well. This branching rule attempts to produce a totally balanced coefficient matrix using this rule. The theoretical justification behind this rule is that by fixing the follow-on's, matrices that are excluded in the characterization of totally balanced matrices are removed from the coefficient matrix. Hoffman, Kolen, and Sakarovitch (1985), give an overview of the characterization of totally balanced matrices.

The follow-on branching technique can be used to reduce the size of the problem by fixing the follow-on's while the number of pairings remain above a certain number, say 100000 pairings. This is the pruning step. At this stage more sophisticated branching decisions can be combined with the follow-on branching rule to get a good integer solution. We discuss below one such method called *strong branching*. Another approach would be to continue the follow-on branching and see if we get a good feasible integer solution.

In order to do this, the LP relaxation of the reduced problem is solved to optimality and the process is repeated after fixing a follow-on. If it is not possible to cover all the flight legs using this approach, more pairings from the original problem that satisfy all the follow-ons that have been fixed are added to the problem. More pairings from the original problem are also added if there is significant increase in the linear programming objective function value. Other minor variations of the branch on follow-on strategy can also be made.

(b) **Timeline Branching**

The idea of *timeline branching* by Klabjan et al. (2001) for solving crew pairing set-partitioning problems comes from SOS branching by Beale and Tomlin

(1970). If we let $\tilde{F}_r = \bigcup_s F_{rs}$ denote the set of all pairings that contain flight leg r, then $\tilde{F}_r$ can be ordered based on the connection time with flight leg r. The connection time t_p of a pairing $p \in \tilde{F}_r$ is defined as the difference between the departure time of flight leg s and the arrival time of flight leg r. For a given leg r, $\tilde{F}_r$ is partitioned based on the connection time t_p of a pairing p $\in \tilde{F}_r$ and a length of time t; all pairings for which $t_p \leq t$ belong to one branch and the others to another branch. All pairings in $\tilde{F}_r$ that are in the first branch are set to 0 and exactly one in the second branch is set to 1. Pairings that end in leg r are also set to 0. In Klabjan et al. (2001), the timeline branching has been shown to be a valid branching rule under the assumption that no two flights depart from a station at the same time, i.e., under the assumption it was shown that if x^* is a basic fractional solution to the LP relaxation of (SPP) then there exists a leg r and a time t such that:

$$0 < \sum_{p \in \tilde{F}_r, t_p \leq t} x_p^* < 1 \tag{2.2}$$

The summation in (3.2) is called the timeline weight. As in the follow-on branching all flight legs whose timeline weight is 1 are fixed and a flight leg with maximum timeline weight less than 1 is also fixed. If there are flight legs departing from a station at the same time then the timeline branching can be applied by slightly perturbing the departure and arrival times.

(c) **Strong Branching**

In both follow-on and timeline branching strategies, it is important to choose a branching pair that has a good chance of yielding an integer solution or improving the lower bounds on the child nodes, or both. *Strong Branching*, which first appeared in the commercial mixed integer programming solver CPLEX CPLEX Optimization (2000), has been successfully employed for the crew pairing optimization problem. Many variations of strong branching have been proposed, see Linderoth and Savelsbergh (1999) and Klabjan et al. (2001). We motivate the basic idea behind the*strong branching* technique.

Let S be a subset of fractional variables that are candidates for branching. For each variable $p \in S$ two branches are formed and a given number k of dual simplex iterations is performed in each branch. For p let f_p^1 and f_p^0 be the corresponding objective values. These values can be used to choose the branching variable. If the two values for a variable p are relatively large, then p is a good candidate to branch on since there is significant increase in the LP value for both the branches. There are three decisions in the strong branching rule that need to be specified: the choice of the set S, the number of dual iterations k, and a rule for combining the values f^1 and f^0 for selecting the branching variable.

Bixby et al. (1995) choose S to be the set of 10 least integral variables and the number of dual simplex iterations to be 50. The integrality of a variable x_p^* with respect to the current LP solution is defined as $|x_p^* - 0.5|$. Klabjan et al. (2001) have generalized the strong branching ideas by combining it with follow-on and timeline branching rules.

2. **Branch-and-cut**

Branch-and-cut methods are very successful techniques for solving a wide variety of integer programming problems, and they can provide a guarantee of optimality. Branch-and-cut methods are exact algorithms consisting of a combination of a cutting plane method with a branch-and-bound algorithm. For an overview of the branch-and-cut approach see Junger and Thienel (1998) and Padberg and Rinaldi (1991). The branch-and-cut approach has been used to solve to proven optimality large set partitioning problems such as the crew pairing optimization problem, see Hoffman and Padberg (1993). A branch-and-cut solver developed by Hoffman and Padberg has been used to solve crew pairing problems having up to 1000 rows and 1.05 million variables to proven optimality. Most crew pairing solvers do not include side constraints such as base-constraints explicitly in the problem formulation. This makes it almost impossible to accurately measure how far the final solution that satisfies the side constraints are from optimality. By explicitly incorporating any number of base-constraints, the branch-and-cut solver has been used successfully for solving crew pairing problems with side constraints to optimality. The schedules obtained using the branch-and-cut solver were reported to have significant flying time in their duty periods.

The branch-and-cut solver by Hoffman and Padberg (1993) has five components: a branch-and-cut optimizer that preprocesses and tightens the user-supplied formulation; a linear programming solver; a heuristic that yields good integer-feasible solutions quickly; a cut generation procedure that tightens the linear program relaxation and a branching strategy that determines the search tree. The constraint or cut generation component is the most important of the five procedures. Using a cut generation procedure, the LP solution is tightened and this often results in an integer solution without any tree-search related enumeration.

The branch-and-cut solver generates cutting planes based on the underlying structure of the polytope defined by the convex hull of the feasible integer points. The polyhedron is approximated by taking a polyhedral relaxation based on the associated set packing polytope, the set covering polytope, and knapsack constraints from the base-constraints. The general form of the cuts generated using the set packing and set covering constraints are lifted clique and odd hole cuts. Lifted minimal cover cuts are generated for the base-constraints, see Crowder, Johnson, and Padberg (1983), Hoffman, Kolen, and Sakarovitch (1985), and Hoffman and Padberg (1991). These cutting planes are generated and included as constraints in a tree-search algorithm that uses automatic reformulation procedures, heuristics and linear programming technology to assist in the solution. More details can be found in Balas and

Padberg (1976), Balas and Ng (1986), Cornuejols and Sassano (1989), and Padberg (1971, 1972, 1975, 1977, 1979).

3. **Branch-and-Price**

The basic idea of the *branch-and-price* methodology is similar to the branch-and-cut technique. In branch-and-price the focus is on *pricing* or dynamically generating columns rather than row or constraint generation as in branch-and-cut. The process of dynamically generating variables can also be viewed as generating cutting planes for the dual of the current LP relaxation. Hence, LP-based branch and bound algorithms in which the variables are generated dynamically when needed are known as branch-and-price algorithms. Barnhart et al. (1998) provide a thorough review of these methods and Desrochers et al. (1995) shows how the branch-and-price framework can be used for routing and scheduling applications.

In branch-and-price, sets of columns are left out of the LP relaxation because there are too many columns to handle efficiently and most of them will have their associated variables equal to zero in an optimal solution anyway. A subproblem which is a separation problem for the dual LP is solved to try to identify columns to enter the basis. If such columns are found, the LP is re-optimized otherwise the current LP solution is optimal. If the LP solution is not integral, branching occurs. It may seem that branch-and-price is a combination of techniques for solving LP's and branch-and-bound, but there are fundamental difficulties in applying column generation techniques for linear programming in integer programming solution methods, see Johnson (1989).

Vance et al. (1997) show how the branch-and-price method can be incorporated in a heuristic framework to obtain near optimal integer solutions for the crew pairing problem. Pairings are generated using the duty period network, which has an arc for each possible duty period and arcs representing possible overnight connections between the duties. Pairings are generated using a multi-label shortest paths approach on the network. The branch on follow-ons branching rule discussed above (Ryan and Foster, 1981) for general set partitioning problems is used at the nodes in the branch-and-bound tree.

When both variables and cutting planes are generated dynamically during LP-based branch-and-bound, the technique becomes known as *branch-and-cut-and-price* (BCP), see Ralphs, Ladanyi, and Trotter (2001). In such a scheme, there is a symmetry between the treatment of cuts and that of variables. While branch-and-cut-and-price does combine ideas from both branch-and-cut and branch-and-price, combining the two techniques requires much more sophisticated methods than either one requires on its own. The ability to handle constantly changing sets of cuts and variables allows these algorithms to undertake the solution of very large-scale integer programming problems. IBM uses a parallel BCP framework for mixed integer programs that allows the user to develop efficient problem class specific MIP solvers with little implementation effort. The BCP framework has been used in the crew pairing solver developed by IBM, see Anbil, Forrest, and Pulleyblank (1998).

2.2.4. Parallelization

Impressive progress has been made by crew pairing solvers due to the development of a wealth of theoretical results and clever solution methodologies. Another equally important factor for this progress has been the dramatic increase in computing power over the last decade, both in terms of processor speed and memory. This increase in the power of hardware has accelerated the development of state-of-the-art software for optimization. Further, the introduction of parallel computing has produced significant improvements in the computational power of optimization tools.

Many crew pairing solvers are increasingly using techniques based on branch-and-cut, branch-and-price and branch-and-cut-and-price (BCP) for producing good solutions quickly. In particular, the BCP technique is very attractive since both variables and constraints can be dynamically generated. Branch-and-bound algorithms are well suited for parallelization because of the partitioning of work that occurs when the branching operation is performed to generate new subproblems. Gendron and Crainic (1994) give a survey on parallel branch-and-bound algorithms.

The BCP technique has been parallelized quite efficiently in some of the commercial crew pairing solvers, see Ralphs, Ladanyi, and Trotter (2001). There are two sources of parallelism in the BCP. First, any group of subproblems on the current candidate list can be processed simultaneously. Once a subproblem has been added to the list, it can be properly processed before or after the processing of any other subproblem. The second major source of parallelism is to parallelize the processing of individual subproblems. This can be done by allowing separation to be performed in parallel with the solution of the linear programs. Alternately, it is also possible to separate over several classes of cuts simultaneously. The most straightforward parallel implementation is a master slave model, in which there is a central manager responsible for partitioning the work and sending it out to the various slave processes that perform the actual computation. Ralphs, Ladanyi, and Trotter (2001) provide an in depth review of an implementation of parallel BCP.

Panayiotis et al. (2000) have developed a scalable parallelization of an approximation algorithm based on Lagrangian relaxation described in Fisher (1981) and Wedelin (1995) for solving the crew scheduling problem. This is being currently employed by the CARMEN crew scheduling solver (Carmen Systems, 1998) used by several European airlines. Enhancements to the algorithm based on an active set strategy which requires less work and is better adapted to the memory hierarchy properties have also been developed. Parallelization of the active set iterations is being undertaken currently. Significant performance improvements have been reported due to parallelization and new enhancements made to the previously used algorithms.

Primal-dual algorithms are another class of algorithms that lend themselves well to parallelization. Klabjan, Johnson, and Nemhauser (2000) propose a parallel primal-dual algorithm with a randomized pricing strategy. In this approach, several small linear programs are repeatedly solved in parallel and their dual solutions are combined to obtain

a new dual feasible solution. Impressive computational results have been reported on a wide range of crew pairing optimization problems.

A successful parallel pairing generation technique has been implemented by Klabjan and Schwan (2002). Dynamic domain decomposition methods are used to parallelize this problem. Near linear speedup even on a large number of processors has been reported. This is achieved by exploiting the combined distributed memory architecture of the underlying cluster computing engine through a novel communication mechanism and by the creation of application-specific load balancing algorithms. The parallel pairing generation algorithm can also be incorporated in branch-and-price algorithms.

2.2.5. Other approaches

1. **An Approximation Algorithm**

   An approximation algorithm for solving large-scale 0-1 integer programming problems developed by Wedelin (1995) has been used for solving the crew pairing optimization problem efficiently. In this algorithm, the 0-1 set-partitioning problem is solved in a more direct way rather than solving a sequence of LP's.

   It consists of two parts: the first is a simple dual search algorithm for finding a 0-1 solution to the 0-1 integer programming problem by considering its Lagrangian relaxation. The second part is an approximation scheme that manipulates the cost array **c** as little as possible, to enable convergence and get a unique 0-1 solution for the relaxed problem. The magnitude of this manipulation is governed by a parameter κ. This parameter determines the compromise between the running time and quality of the solution. When $\kappa = 0$, the algorithm has a linear programming interpretation and strives to find optimal solutions, but often fails. At $\kappa = 1/3$ the algorithm can be interpreted in terms of dynamic programming and when $\kappa = 1$ it turns into a greedy algorithm that strives to converge quickly to any feasible solution. So there is no approximation when $\kappa = 0$ and the approximation is maximal when $\kappa = 1$.

   In practice it is not possible to know in advance the value of κ that would work best. One approach would be to start the algorithm with $\kappa = 0$, and slowly increase its value until convergence occurs. However, in practice a more efficient procedure is used for selecting the value of κ for this purpose. The use of the parameter κ is in some sense similar to the use of the temperature parameter in simulated annealing methods.

2. **Alternate Formulations**

   Even though the set-partitioning formulation (SPP) is one of the widely used formulations for the crew pairing problem, other formulations have also been used successfully. We shall discuss two of these below.

   (a) Desaulniers et al. (1997) use an integer nonlinear multi-commodity network flow model with additional resource variables to model the crew pairing optimization problem. The model can also handle nonlinear costs and nonlinearities in a large subset of constraints. A branch-and-bound algorithm based on

an extension of the Dantzig-Wolfe decomposition principle (Dantzig and Wolfe, 1960) is used to solve this model. The master problem remains a set partitioning problem and columns are generated using resource constrained shortest path subproblems.

In this model, a specific commodity is associated with each crew and commodities can be aggregated when they represent identical crews. The underlying network is a time-space network whose nodes represent stations at different times and arcs represent flight legs, deadhead flights, rests etc., see Minoux (1984). A single arc can represent single or multiple crew activities. The current flow between a pair of legs in the underlying network is used for branching decisions. For a selected pair with a fractional flow, value decisions are taken by imposing a flow of 0 on one branch and a flow of 1 on the other. At each node of the search tree, a lower bound is provided by solving the linear relaxation of the model to optimality using a Dantzig-Wolfe decomposition technique. The solution methodology is similar to one used in Desrosiers et al. (1991) for bus driver scheduling.

This approach also implicitly considers all feasible pairings and provides the optimality gap on a feasible solution. The decomposition process isolates all nonlinear aspects of the proposed multi-commodity model in the subproblems which are solved by means of a specialized dynamic programming approach.

(b) In the duty period formulation, the crew pairing problem is modeled using a duty network. In this model a set of duty periods that partitions the flight legs and then a set of pairings that partitions the duty periods are chosen. A good pairing solution is obtained by using good sets of duty periods as building blocks for the pairings. The duty period formulation is given by:

$$
\begin{aligned}
\min \quad & \sum_{d \in D} b_d x_d + \sum_{p \in P} \hat{c}_p z_p \\
\text{subject to:} \quad & \sum_{d : i \in d} x_d = 1 \qquad i \in F \\
& \sum_{p : d \in p} z_p = x_d \qquad d \in D \\
& x_d \in \{0, 1\} \qquad d \in D \\
& z_p \in \{0, 1\} \qquad p \in P
\end{aligned}
$$

where $x_d = 1$ if duty d is chosen, and 0 otherwise; $z_p = 1$ if pairing p is chosen, and 0 otherwise. F is the set of flight legs in the schedule, D is the set of duty periods, P is the set of all legal pairings, b_d is the cost of duty period d, $\hat{c}_p$ the pay-and-credit for pairing p. In the above duty period formulation, the first set of constraints enforce that each flight leg is covered by exactly one duty period. The second set requires that a pairing be chosen to cover each duty that is being used to cover a flight leg.

The duty period formulation described above will have more rows and columns than the set-partitioning model (SPP) used to model the crew pairing problem, but both of them have the same LP bound, see Vance et al. (1995). A slightly different formulation of the above duty period model can be shown to have a tighter LP bound than the set-partitioning model. The crew pairing problem is solved by applying the Dantzig-Wolfe decomposition technique and enforcing the set partitioning constraints in the duty period model.

Computational results reported in Vance et al. (1995) show that the LP relaxation of this formulation provides integer optimal solutions when the LP relaxation of the set-partitioning model does not. This may be due to the fact that the duty period model provides a tighter bound on the optimal IP solution. The computational overhead in this approach involves solving the LP relaxation, which is more difficult to solve than the LP relaxation of the set-partitioning model (SPP).

3. Future directions

3.1. Cost v/s quality

1. **Robustness**

Most of the crew pairing solvers strive to solve the crew pairing problem with the objective of minimizing excess crew costs. High volume air traffic coupled with unforseen weather patterns can be a perfect recipe for delays due to disruptions. Even minor disruptions can easily propagate to the future and cause major delays for subsequent flights. This results in loss for the airlines and causes inconvenience for the passengers. Current crew pairing solvers tend to produce "brittle" solutions, since disruptions that propagate to future flights can breakdown the crew schedule and render it useless. One main factor responsible for this is that most crew pairing solvers do not explicitly take into account the robustness criteria in their models.

Preliminary research on balancing the trade-off between producing a cost effective solution and a robust one have shown promising results. Ehrgott and Ryan (2003) have shown how a bicriteria framework can be effectively used for addressing this problem. They have given empirical evidence that suggests a linear increase in expected delay due to propagation of delays through the flight schedule. In order to minimize the propagation of delays, their model discourages changing of aircraft by the crew if insufficient ground time occurs due to late arrivals. Based on this observation the set partitioning model for the crew pairing problem is solved using multiple objective functions to minimize cost and penalize pairings that are not "robust".

Ehrgott and Ryan have shown that it is not only desirable to have a more robust solution but it is also achievable. More work needs to be done in developing efficient solution methodologies for the crew pairing problem that incorporates a broader definition of robustness.

2. **Crew Fatigue**

Two main factors responsible for *crew fatigue* are sleep loss and and irregular sleep patterns. Even though there are many FAA rules to handle sleep loss in aircrew there are no specific rules directed at minimizing irregular sleep patterns. It is well known that irregular sleep patterns affect the body's *circadian rhythms*, which in turn can cause fatigue, see Gander, Rosekind, and Gregory (1998) and Rosekind et al. (1994). Circadian rhythms are patterns of activity that occur in the body in 24–28 hour cycles. Examples of circadian rhythms are heart rate, motor activity rate and core body temperature. Some of these activities are critical for alertness and others are indication of changes taking place in the body over a 24 hour cycle. Studies show that it is not only the quantity of sleep but also the quality and timing of sleep that is important to mitigate fatigue.

Some of the legal pairings, while having the required number of sleep hours according to FAA rest rules, may have rest periods that are highly irregular. Since some of the large fleets can produce billions of legal pairings, it may be possible to optimize over legal pairings that have more regularity in sleep patterns without increasing the crew costs significantly. A preliminary computational study by Gopalakrishnan and Johnson (2003) shows promising results for minimizing crew fatigue at the crew pairing optimization stage. The basic idea behind their approach is to use the intersection of rest periods in a pairing as a measure of regularity of sleep for the air crew. All legal pairings are partitioned into two sets using a *Fatigue Rule*. There are two parameters associated with the rule; the first is the *Rest Intersection* parameter which can be between 5–8 contiguous hours, and the second is the *Overnight Rest Window* parameter, which is a 14 hour time window such as [20:00, 10:00]. The Fatigue Rule selects crew pairings that have a contiguous block of rest given by *Rest Intersection* that is contained in the *Overnight Rest Window* on all days during which the pairings are flown. This idea has also been extended for the crew bidline problem in Weir (2002).

3. **Regularity**

The weekly crew scheduling problem requires finding crew pairings that partition all the flights in the weekly schedule. Minimizing the crew costs in solving the weekly problem often times results in pairings that cannot be repeated in the weekly horizon unless such constraints are specified in the problem. Such a solution lacks *regularity*.

Regularity is important with respect to crew schedules, since regular solutions are not only easier to implement and manage but also crews prefer to repeat itineraries whenever possible. Some of the current methods try to achieve some regularity by solving a two-stage problem. In this approach a daily problem is solved using a subset of the legs followed by the next stage when the remaining legs are covered as a weekly exceptions problem. The main drawback of such an approach is it can substantially reduce the regularity achieved in the first stage.

Klabjan et al. (2001) propose a method which imposes regularity constraints or penalties directly on the weekly problem. In this fashion, their model captures both

cost and regularity in a weekly problem. The basic idea behind their approach is to partition the flight legs into groups in such a way that within each group regularity is easily achievable. The flight legs are partitioned into g-regular groups, $g = 4, 5, 6, 7$, where the flight legs in a g-regular group can be partitioned with pairings that can be repeated g consecutive days of the week. The model uses binary variables for each pairing in the weekly horizon and a binary variable for each flight leg that assigns this flight leg to a g-regular group. The computational results obtained have shown that the solutions obtained using this approach can be used to find good trade-off between cost and regularity by varying the objective function.

3.2. Integration

Airline planning is complex and therefore divided into several stages. The five stages of airline planning are *Schedule Development*, *Fleet Assignment*, *Aircraft Routing*, *Crew Scheduling* and *Crew Assignment*. Ideally all five problems should be solved as a single problem, but this may not be feasible computationally. However, it may be possible to combine some of the more closely related stages in airline planning. Some progress has been made to meet the challenge of integration. Klabjan et al. (2002), have addressed this problem by considering a partial integration of schedule planning, aircraft routing and crew scheduling. Freling, Huisman, and Wagelmans (2000) discuss models and algorithms for integrating vehicle and crew scheduling. Integration of the fleet assignment and crew scheduling problems in airline planning have also been attempted.

It is estimated that by integrating crew and aircraft planning the airline industry could save an additional half a billion dollars per year on crew costs alone. Some of the recent work on integrated airline planning have moved a step closer to achieving the dream of integrated planning. More work needs to be done on developing efficient solution methodologies for some of the complex optimization problems that surface while considering an integrated approach to airline planning.

3.3. Cutting plane technology

Cutting plane methods are a traditional approach for solving integer programming problems, especially when used in a branch-and-cut framework. Recently, the Gomory mixed integer cut has experienced some computational success, see Balas et al. (1996). A new class of cutting planes, described in Gomory and Johnson (2003), generalize the Gomory mixed integer cuts with a class of subadditive functions over the unit interval that can be used to generate cutting planes for any integer programming problem.

To see how the method works, let x_{B_k} be a fractional basic variable in an optimal basic solution to the LP-relaxation of a given pure integer programming problem, and let

$$x_{B_k} + \sum_{j \in N} \bar{A}_{ij} x_j = \bar{b}_k$$

be the corresponding row of the optimal LP tableau. Let $f(u) = u - \lfloor u \rfloor$ represent the fractional part of any real number u. Given a function $\pi : [0, 1) \to [0, 1)$ over the unit interval that satisfies certain *complementarity* and *subadditivity* conditions, the inequality

$$\sum_{j \in N} \pi(f(\bar{A}_{ij}))x_j \geq \pi(f(\bar{b}_k))$$

is a valid inequality for the original IP that removes the current fractional solution from the feasible region. A similar construction applies using the same subadditive functions when continuous variables are also present.

Ongoing research is working on an implementation of this method, and experimenting with different rules to determine which rows should be used to generate cutting planes, as well as which subadditive function should be used to generate a cutting plane from a given fractional basic variable. Combined with other cutting plane methods designed for use specifically with 0-1 integer programming problems, this method should help solve large scale IP's such as those arising in crew-pairing problems, faster. More details on this approach can be found in Araoz et al. (2003) and Gomory, Johnson, and Evans (2003).

References

Anbil, R., E. Gelman, B. Patty, and R. Tanga. (1991). "Recent Advances in Crew-Pairing Optimization at American Airlines." *Interfaces* 21, 62–74.

Anbil, R., E.L. Johnson, and R. Tanga. (1991). "A Gobal Approach to Crew Pairing Optimization." *IBM Systems Journal* 31, 71–78

Anbil, R., J. Forrest, and W. Pulleyblank. (1998). "Column Generation and the Airline Crew Pairing Problem." *Documenta Mathematica—Journal der Deutschen MathematikerVereinigung, number III in extra volume: proceedings of the ICM*.

Anderson, E., E. Housos, N. Kohl, and D. Wedelin. (1997). "Crew Pairing Optimization." In G. Yu, (ed.), *OR in airline industry*, Kluwer Acad. Publ., Boston.

Arabeyre, J. P., J. Feanley, F.C. Stieger, and W. Teather. (1969). "The Airline Crew Scheduling Problem: A Survey." *Transportation Science* 3, 140–163.

Araoz, J., L. Evans, R.E. Gomory, and E.L. Johnson. (2003). "Cyclic Group and Knapsack Facets." *Mathematical Programming* 96(2), 337–408.

Balas, E. and M. Padberg. (1976). "Set Partitioning: A Survey." *SIAM Review* 18, 710–760.

Balas, E. and S.M. Ng. (1986). "On the Set-Covering Polytope I: All Facets with Coefficients {0,1,2}." MSSR-522, GSIA, Carnegie-Mellon U.

Balas, E., S. Ceria, G. Cornuejols, and N. Natraj. (1996). "Gomory Cuts Revisited." *Operations Research Letters* 19, 1–9.

Barahona, F. and R. Anbil. (1997). "The Volume Algorithm: Producing Primal Solutions with a Subgradient Method." *Research Report RC 21103 (94395)*, IBM T.J. Watson Research Center, Yorktown Heights, NY.

Baker, E.K., L.D. Bodin, and M. Fisher. (1985). "The Development of a Heuristic Set Covering Based System for Aircrew Scheduling." *Transportation Policy Decision Making* 3, 95–110.

Ball, M. and A. Roberts. (1985). "A Graph Partitioning Approach to Airline Crew Scheduling." *Transportation Science* 19(2) 107–126.

Barnhart, C., L. Hatay, and E.L. Johnson. (1995). "Deadhead Selection for the Long-Haul Crew Pairing Problem." *Operations Research* 43, 491–499.

Barnhart, C., E.L. Hohnson, R. Anbil, and L. Hatay. (1994). "A Column Generation Technique for the Long-Haul Crew Assignment Problem." In T.A. Cirani and R.C. Leachman (eds.), *Optimization in Industry II*, Wiley.

Barnhart, C., E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance. (1998). "Branch-and-Price: Column Generation for Solving Huge Integer Programs." *Operations Research*, 43, 491–499.

Barutt, J. and T. Hull. (1990). "Airline Crew Scheduling: Supercomputers and Algorithms." *SIAM News* 23(6), 1 and 20–22.

Beale, E. and J. Tomlin. (1970). "Special Facilities in a General Mathematical Programming System for Non-Convex Problems Using Ordered Sets of Variables." *Proceedings of the 5th International Conference on Operations Research*.

Bixby, R., W. Cook, A. Cox, and E. Lee. (1995). "Parallel Mixed Integer Programming." *Technical Report CRPC-TR95554*, Rice University. Available from ftp://softlib.rice.edu/pub/CRPC-TRs/reports

Bornemann, D.R. (1982). "The Evolution of Airline Crew Pairing Optimization." AGIFORS *Crew Management Study Group Proceedings*, Rio de Janeiro (April).

The Carmen Systems, version 5.1, Carmen Systems AB, Göteborg, Sweden.

Chu, H., E. Gelman, and E.L. Johnson. (1997). "Solving Large Scale Crew Scheduling Problem." *European Journal of Operations Research* 97, 245–259.

Cornuejols, G. and A. Sassano. (1989). "On the 0,1 Facets of the Set Covering Polytope." *Mathematical Programming* 43, 101–111.

Crowder, H., E.L. Johnson, and M.W. Padberg. (1983). "Solving Large-Scale 0-1 Linear Programming Problems." *Operations Research* 31, 803–834.

CPLEX Optimization, "Using the CPLEX Callable Library." 7.0 edn, ILOG Inc.

Dantzig, G.B. and P. Wolfe. (1960). "Decomposition Principle for Linear Programs." *Operations Research* 8, 101–111.

Desrosiers, J., Y. Dumas, M. Desrochers, F. Soumis, B. Sasno, and P. Trudeau. (1991). "A Breakthrough in Airline Crew Scheduling." Technical Report G-91-11, Les Cahiers du GERAD.

Desrochers, J., Y. Dumas, M.M. Solomon, and F. Soumis (1995). "Time Constrained Routing and Scheduling." In M.E. Ball, T.L. Magnanti, C. Monma and G.L. Nemhauser (eds.), *Handbook in Operations Research and Management Science*, 8: Network Routing, Elsevier, Amsterdam, 35–140.

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

Ehrgott, M. and D.M. Ryan. (2003). "Constructing Robust Crew Schedules with Bicriteria Optimization." *Journal of Multi-Criteria Decision Analysis* 11, 139–150, 2002.

Etschmaier, M.M. and D.F.X. Mathaisel. (1985). "Airline Scheduling: An Overview." *Transportation Science* 19, 127–138.

Fisher, M.L. (1981). "The Lagrangian Relaxation Method for Solving Integer Programming Problems." *Management Science* 27(1), 1–18.

Forrest, J.J. (1989). "Mathematical Programming with a Library of Optimization Subroutines." presented at the ORSA/TIMS Joint National Meeting, New York.

Freling, R., D. Huisman, and A.P.M. Wagelmans. (2000). "Models And Algorithms For Integration Of Vehicle And Crew Scheduling." No 189 in Econometric Institute Report from Erasmus University Rotterdam, Econometric Institute.

Gander, P.H., M.R. Rosekind, and K.B. Gregory. (1998). "Flight Crew Fatigue." *Aviation, Space and Environmental Medicine* 69, (9 Suppl.), B49–B60.

Gendron, B. and T.G. Crainic. (1994). "Parallel Branch-and-Bound Algoirithms: Survey and Synthesis." *Operations Research* 42, 1042.

Gerbracht, R. (1978). "A New Algorithm for Very Large Crew Pairing Problems." *18th AGIFORS Symposium*, Vancouver, British Columbia, CA.

Gershkoff, I. (1989). "Optimizing Flight Crew Schedules." *Interfaces* 19, 29–43.

Gomory, R.E. (1963). "An Algorithm for Integer Solutions to Linear Programs." In R.L. Graves and P. Wolfe (eds.), *Recent Advances in Mathematical Programming*, pp. 269–302. McGraw-Hill, New York.

R.E. Gomory and E.L. Johnson. (2003). "T-Space and Cutting Planes." *Mathematical Programming Ser.* B 96(2), 341–375.

Gomory, R.E., E.L. Johnson, and L. Evans. (2003). "Corner Polyhedra and their Applications to Cutting Planes." *Mathematical Programming* 96(2), 321–339.

Gopalakrishnan, B. and E.L. Johnson. (2003). "Mitigating Crew Fatigue through Crew Pairing Optimization." Working Paper, School of Industrial and Systems Engineering, Georgia Tech.

Graves, G.W., R.D. McBride, and I. Gershkoff. (1993). "Flight Crew Scheduling." *Management Science* 39(6), 736–745.

Held, M. and R.M. Karp. (1970). "The Travelling Salesman Problem and Minimum Spanning Tress." *Oper. Res*. 18, 1138–1162.

Held, M. and R.M. Karp. (1971). "The Travelling Salesman Problem and Minimum Spanning Tress: Part II." *Mathematical Programming* 1, 6–25.

Held, M., P. Wolfe, and H.P. Crowder. (1974). "Validation of Subgradient Optimization." *Mathematical Programming* 6, 62–88.

Hoffman, K.L. and M. Padberg. (1993). "Solving Airline Crew-Scheduling Problems by Branch-and-Cut." *Management Science* 39, 657–682.

Hoffman, A.J, A. Kolen, and M. Sakarovitch. (1985). "Totally Balanced and Greedy Matrices." *SIAM Journal on Algebraic and Discrete Methods* 6, 721–730.

Hoffman, K.L. and M. Padberg. (1991). "Techniques for Improving the LP = representation of 0-1 Linear Programming Problems." *ORSA J. Computing* 3, 121–134.

Housos, E. and T. Elmroth. (1997). "Automatic Optimization of Subproblems in Schedulign Airline Crews." *Interfaces* 27, 68–77.

Hu, J. (1996). "Solving Linear Programs Using Primal-Dual Subproblem Simplex Method and Quasi-Explicit Matrices." Ph.D. Dissertation.

Hu, J. and E. Johnson. (1999). "Computational Results with a Primal-Dual Subproblem Simplex Method." *Operations Research Letters* 25, 149–158.

Johnson E.L. (1989). "Modeling and Strong Linear Programs for Mixed Integer Programming." In S.W. Wallace (ed.), *Algorithms and Model Formulations in Mathematical Programming*, NATO ASI Series 51, pp. 1–41.

Junger, M. and S. Thienel. (1998). "Introduction to ABACUS - A Branch-and-Cut System." *Operations Research Letters* 22, 83–95.

Klabjan, D., E. Johnson, and G.L. Nemhauser. (2000). "A Parallel Primal-Dual Simplex Algorithm." *Operations Research Letters* 27, 47–55

Klabjan, D., E. Johnson, and G.L. Nemhauser, E. Gelman, and S. Ramaswamy. (2002). "Airline Crew Scheduling with Time Windows and Plane Count Constraints." *Transportation Science* 36, 337–348.

Klabjan, D. and K. Schwan. (2002). "Airline Crew Pairing Generation in Parallel." *Technical report TLI/LEC-99-02*.

Klabjan, D., E. Johnson, G.L. Nemhauser, E. Gelman, and S. Ramaswamy. (2001). "Solving Large Airline Crew Scheduling Problems: Random Pairing Generation and Strong Branching." *Computational Optimization and Applications* 20, 73–91

Klabjan, D., E. Johnson, G.L. Nemhauser, E. Gelman, and S. Ramaswamy. (2001). "Airline Crew Scheduling with Regularity." *Transportation Science* 35, 359–374.

Larson, T. and Z. Liu. (1989). "A Primal Convergence Result for Dual Subgradient Optimization with Application to Multicommodity Network Flows." Research Report S-581 83, Dept. of Math., Linkoping Institute of Technology.

Lavoie, S., M. Minoux, and E. Odier. (1988). "A New Approach to Crew Pairing Problems by Column Generation and Application to Air Transport." *European Journal of Operatiosn Research* 35, 45–58.

Lemarechal, T. (1975). "An Extention of Davidson Methods to Nondifferentiable Problems." *Mathematical Programming Study* 3, 95–109.

Lemke, C., H. Salkin, K. Spielberg. (1971). "Set Covering by Single Branch Enumeration with Linear Programming Subproblems." *Operations Research* 19, 998–1022.

Lemke, C. and K. Spielberg. (1967). "Direct Search Algorithms for Zero-One and Mixed Integer Programming."*Operations Research* 15, 892–914.

Linderoth, J. and M. Savelsbergh. (1999). "A Computational Study of Search Strategies for Mixed Integer Programs." *Informs Journal on Computing* 11, 173–187.

Marsten, R.E. and F. Shepardson. (1981). "Exact Solution of Crew Problems using the Set Partitioning Mode: recent Successful Applications." *Networks* 11, 165–177.

Minoux, M. (1984). "Column Generation Technique in Combinatorial Optimization: A New Application to Crew Pairing Problems." *Proceedings XXIVth AGIFORS Symposium*.

Mitchell, J.E. (2000). "Branch-and-Cut Algorithms for Combinatorial Optimization Problems." *Handbook of Applied Operations Research*, Oxford University Press.

Nemhauser, G.L. and L.A. Wolsey. (1999). "Integer and Combinatorial Optimization." John Wiley, New York.

Padberg, M. and G. Rinaldi. (1991). "A Branch-and-Cut Algorithm for the solution of Large-Scale Traveling Salesman Problems." *SIAM Review* 33, 1–41.

Padberg, M. (1971). "Essays in Integer Programming." Pd.D. Thesis, GSAI, Carnegie-Mellon U., Pittsburg, PA.

Padberg, M. (1972). "On the Facial Struture of the Set Packing Polyhedra." *Mathematical Programming* 5, 199–215.

Padberg, M. (1975). "A Note on 0-1 Programming." *Operations Research* 23, 833–837.

Padberg, M. (1977). "On the Complexity of Set Packing Polyhedra." *Annals of Dicrete Mathematics* 1, 421–434.

Padberg, M. (1979). "Covering, Packing and Knapsack Problems." *Annals of Dicrete Mathematics* 4, 265–287.

Panayiotis, A., P. Sanders, T. Takkula, and D. Wedelin(2000). "Parallel Integer Optimization for Crew Scheduling." *Annals of Operations Research* 99, 141–166.

Ralphs, T.K., L. Ladanyi, and L.E. Trotter, Jr. (2001). "Branch, Cut, and Price: Sequential and Parallel." In D. Naddef and M. Juenger (eds.), *Computational Combinatorial Optimization*.

Rosekind, M.R., P.H. Gander, R.M. Smith, K.J. Weldon, and K.L. McNally. (1994). "Fatigue in Aviation." *Air Line Pilot* 63(10), 22–25.

Ryan, D.M. and J.C. Falkner. (1988). "On the Integer Properties of Scheduling Set Paritioning Problems." *European Journal of Operations Research* 35, 442–456.

Ryan, D.M. and B. Foster. (1981). "An Integer Programming Approach to Scheduling." In A. Wren (ed.), *Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling*. North-Holland, pp. 269–280.

Rubin, J. (1971). "Airline Crew Scheduling—The Non-Mathematical Problem." *IBM Technical Report : 320.3006*.

Rubin, J. (1973). "A Technique for the Solution of Massive Set Covering Problems with Applications to Airline Crew Scheduling." *Transportation Science* 7, 34–38.

Shor, N.Z. (1985). "Minimization Methods for nondifferentiable functions." *Springer*, Berlin.

Vance, P.H., A. Atamturk, C. Barnhart, E. Gelman, E. Johnson, A. Krishna, D. Mahindra, G.L. Nemhauser, and R. Rebello. (1997). "A Heuristic Branch-and-Price Approach and Airline Crew Pairing Problem."

Vance, P.H., C. Barnhart, E.L. Johnson, and G.L. Nemhauser (1995). "Airline Crew Scheduling: A New Formulation and Decomposition Algorithm." *Operations Research* 45(2), 188–200.

Wedelin, D. (1995). "An Algorithm for Large-Sclae 0-1 Integer Programming with Applications to Airline Crew Scheduling." *Annals of Operations Research* 57, 283–301.

Weir, J. (2002). "A Threee Phase Approach to Solving the Crew Bidline Problem with Rules for Mitigating Crew Fatigue." Ph.D. Dissertation, Georgia Institute of Technology.

Wise, T.H (1995). "Column Generation and Polyhedral Combinatorics for Airline Crew Scheduling." Ph.D. Dissertation, Cornell U., Ithaca, N.Y.

Wolfe, P. (1975). "A Method of Conjugate Subgradients for Minimizing Nondifferentiable Functions." *Mathematical Programming Study* 3, 145–173.