14 Airline Crew Scheduling

Cynthia Barnhart, Amy M. Cohn
Ellis L. Johnson, Diego Klabjan
George L. Nemhauser, Pamela H. Vance

14.1 Introduction

14.1.1 Crew Scheduling

Crew scheduling can be defined as the problem of assigning a group of workers (a *crew*) to a set of tasks. The crews are typically interchangeable, although in some cases different crews possess different characteristics that affect which subsets of tasks they can complete.

Crew scheduling problems appear in a number of transportation contexts. Examples include bus and rail transit, truck and rail freight transport, and freight and passenger air transportation. There are many common elements to all of these problems, including the need to cover all tasks while seeking to minimize labor costs, and a wide variety of constraints imposed by safety regulations and labor negotiations. Nonetheless, each application also has its own unique characteristics and its own research challenges. In fact, most crew scheduling research focuses on a particular application, rather than the general case.

In this chapter, we focus on the *airline crew scheduling problem*. [For additional details on crew scheduling in the railway industry we refer the reader to Caprara et al., 1998 and for crew scheduling in mass transit systems to Wilson, 1999.] There are a number of reasons for focusing on airlines. First, they provide a context for examining many of the elements common to all crew scheduling problems. Second, the airline problem is truly a planning problem in the sense that airlines typically have a fixed schedule that changes at most monthly. Therefore, substantial time and resources can be (and are) allocated to solving it. Third, airline crews receive substantially higher salaries than equivalent personnel in other modes of transportation; the savings associated with an improved airline crew schedule can be quite significant. Finally, a large number of restrictive rules, mandated both by the FAA (or equivalent governing agencies for non-U.S. carriers) and strong labor unions, greatly restrict the set of feasible solutions, making airline crew scheduling one

of the hardest crew scheduling problems. For all of these reasons, the airline crew scheduling problem has received the greatest level of attention, both from industry and from the academic community.

14.1.2 Airline Planning

Crew scheduling is just one of a number of challenging planning problems faced by airlines, see Figure 14.1. Although these problems are closely interrelated, they are typically solved sequentially, due to their size and complexity. Airlines usually begin by solving a *schedule design problem*, in which they determine the flights to be flown during a given time period. In the next step, the *fleet assignment problem*, they decide what type of aircraft (such as Boeing 767, 727, etc.) to assign to each flight, as a function of the forecasted demand for that flight. The *maintenance routing problem* follows, in which individual aircraft are assigned to flights so as to ensure that each aircraft spends adequate time at specific airports in order to undergo routine maintenance checks. Having completed these three tasks, the airlines then address the problem of scheduling crews.

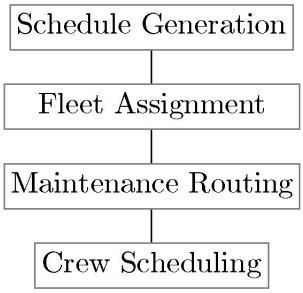

Figure 14.1. Schedule Planning

Within airline crew scheduling, there are significant differences between how international and domestic operations are scheduled. In the U.S., for example, international flight networks tend to be relatively sparse, with a limited number of flights into and out of an airport. U.S. domestic operations, in contrast, are characterized by *hub-and-spoke networks* with large numbers of arrivals followed by departures (called *banks* or *complexes*) occurring at hub airports in relatively short periods of time. International flight networks, however, are characterized by *point-to-point networks* with operations spread throughout the network. Another distinction is that international networks typically operate on a *weekly schedule*, while *daily schedules* are usually assumed

for domestic operations. Moreover, unlike domestic operations, it is not uncommon for international operations to *deadhead* crews, that is fly them as passengers on some of the flights within their schedule in order to re-position them for future assignments. Barnhart et al., 1995 study the deadheading problem. All of these differences affect how crews are scheduled.

There are also significant differences between how *cockpit* and *cabin crews* are scheduled. For example, crews of pilots and other cockpit personnel usually remain together for much of their schedule. Cabin crews tend to vary more frequently, with flight attendants scheduled as individuals, rather than as part of a prescribed crew. Another key difference is that cockpit crews are heavily restricted in the number of fleet types that they are qualified to fly; cabin crews have greater latitude in the range of aircraft types that they can staff.

We will focus our attention on the problem of scheduling cockpit crews. For additional information on other forms of airline crew scheduling, we refer the reader to Day and Ryan, 1997 and Kwok and Wu, 1996.

14.2 The Crew Scheduling Problem

Each cockpit crew is qualified to fly a specific fleet type or set of closely related fleet types, known as a *fleet family*. Therefore, we solve a separate crew scheduling problem for each crew type, which includes only those flights that have been assigned to the corresponding fleet types.

The input to a crew scheduling problem is the set of flights to be covered. Flights are grouped together to form *duty periods*, which are series of sequential flight legs comprising a day's work for a crew. Duties are then strung together to form *pairings*, crew trips spanning one or more work days separated by periods of rest. Finally, monthly *schedules* are made up of multiple pairings with time off in between. These four components, i.e., flights, duties, pairings, and monthly schedules, are the building blocks of crew scheduling.

Associated with each of these building blocks is a distinct set of constraints. These typically come from three sources. First, governing agencies such as the FAA in the U.S. restrict crew scheduling, primarily for safety purposes. Second, labor organizations often enter into collective bargaining agreements concerning the crews' work conditions. Third, the airlines themselves pose added constraints, for example, to make the schedule more robust. In addition to these constraints, each building block is associated with a distinct cost structure. These constraints and cost structures are described in greater detail in the sections that follow.

14.2.1 Work Rules and Pay Structures

The most elemental decision in the crew scheduling problem is to decide which crew to assign to a given flight. The cost of such an assignment is a complex computation. Crews are not salaried, but rather are paid for the time that they spend flying, plus some added compensation for excess time spent on the ground between flights and during rest periods. Given this, we can think of the 'cost' associated with an individual flight simply as the duration of that flight. Because individual crews for a given fleet family cannot be distinguished in the crew pairing problem, crew costs are usually expressed in terms of time rather than cost. The total flying time in the system is clearly fixed and provides a lower bound on the optimal crew cost. The objective in crew scheduling is therefore to minimize *pay-and-credit*, the payments made above and beyond the cost of the actual flying time.

14.2.1.1 Duty Periods. A number of rules restrict what combinations of flights can be flown by the same crew. A sequence of flights that can be flown by a single crew over the course of a work day is a *duty period*. Note that the same crew members typically stay together throughout the duration of a duty period.

Duties are constrained by a number of restrictions. The most obvious of these is that flights must be sequential in space and time. Furthermore, there is a restriction on the *minimum idle time* between two sequential flights, sometimes referred to as *connect time* or *sit time*. There is also a restriction on the *maximum idle time* allowed between two sequential flights. Additionally, there is a *maximum elapsed time* for a duty period. Finally, strict regulations govern the *total number of flying hours*, known as *block time*, that a crew can incur during the course of a single duty period.

The crew cost associated with a duty period is usually expressed as the maximum of three quantities. The first quantity is the *flying time*. The second quantity is a fraction (for example, $\frac{5}{8}$) of the *total elapsed time* of the duty period. The third quantity is a *minimum guaranteed* number of hours. This pay structure primarily compensates crews for flying time, but also provides additional pay for those crews assigned to very short duties or to duties with extensive idle time between the flights. Formally the cost of a duty period d can be expressed as

$$b_d = \max\{f_d \cdot elapse, fly, min\_guar\},$$

where b_d is the cost in minutes, $f_d \cdot elapse$ is a fraction of the elapsed time $elapse$, fly is the number of minutes of flying in the duty period, and min_guar is the minimum guarantee expressed in minutes.

14.2.1.2 Pairings. Often a duty period starts and ends at different airports. Therefore, the crew cannot always return home at the end of a duty period but instead must often *layover* until the next day's duty period begins. Typically, crews spend anywhere from one to five days in a row away from home. A sequence of duties and layovers is known as a *pairing*. In general, a crew will stay together for all of the duties within a pairing.

There are a number of logical constraints on what constitutes a feasible pairing. Clearly, a pairing's first duty period must begin at the crew's domicile, called also the *crewbase*, and the last duty period must end there as well. In addition, each duty period must begin at the same airport where the previous duty period ended.

Pairings are further constrained by a complex array of rest requirements, flying time restrictions, and other constraints. These include the *maximum number of duties* in a pairing, the *minimum* and the *maximum amount of rest* between duties, and the *maximum elapsed time* of a pairing, also known as *time-away-from-base* (TAFB). One particularly complicated constraint is the *8-in-24* rule, which is imposed by the FAA in the U.S. This rule requires extra rest if a pairing contains more than 8 hours of flying in any 24 hour period. Generally, this occurs when the 24 hour period in question spans two consecutive duty periods. It is allowable to have more than 8 hours of flying in a 24 hour period only so long as the *included rest*, i.e., the layover between the two duty periods involved, and the rest following the second duty period, also known as the *compensatory rest*, are of sufficient length.

In the U.S., the cost of a pairing has two components. The first component, similar to the cost of a duty period, is the maximum of three quantities. The first of these quantities is the *sum of the costs of the duties* contained in the pairing. The second quantity is some fraction of the *total elapsed time* of the pairing. The third quantity is a *minimum guaranteed* number of minutes per pairing, which is typically the number of duty periods in the pairing times a fixed minimum guaranteed number of minutes per duty period. In addition to this, we include a second component, which represents the extra costs associated with the rest period between two duties, such as meals and lodging. Formally, the

cost of a pairing p is

$$c_p = \max\{f_p \cdot \text{TAFB}, ndp \cdot mg, \sum_{d \in p} b_d\} + \sum_{\substack{\hat{d} \in p, \bar{d} \in p \\ \hat{d} \to \bar{d}}} e(\hat{d}, \bar{d}) \,,$$

where $d, \hat{d}, \bar{d}$ represent the duty periods in p. Here, $\hat{d} \to \bar{d}$ indicates that duty period $\bar{d}$ immediately follows duty period $\hat{d}$ in p. In addition, mg and f_p are constants, ndp is the number of duty periods in p, and $e(\hat{d}, \bar{d})$ is the extra cost associated with the rest between duty periods $\hat{d}$ and $\bar{d}$.

European carriers tend to have a fixed salary for each crew. In this case the cost of a pairing is either ndp or 1.

14.2.1.3 Schedules. Just as a duty period is a sequence of flights with sit times in between, and a pairing is a sequence of duties with layovers in between, a schedule is simply a sequence of pairings with periods of time off in between. However, a key difference between schedules and the other building blocks is that schedules are associated with *individual crew members*, rather than complete crews. The reason is that each crew member has different needs for time-off throughout the schedule period, which is typically a month. These include vacation time, training time, etc. Thus, in assigning crew schedules, we must take into account the needs and preferences of individual crew members.

In addition to individual crew member needs, we also have constraints similar to those seen for duties and pairings, for example, limits on the *maximum monthly flying time*, the *maximum duty time* in a month, the *minimum number of consecutive days off*, the *minimum total number of days off*, the *minimum rest* between pairings, and so forth.

Given this key difference, it is not surprising that the cost of a schedule is quite different from the other components. Whereas the focus within duties and pairings is on actual labor costs, the cost of a schedule is considered to be more a function of crew satisfaction and of workload balance.

14.2.2 The Crew Pairing and Crew Assignment Problems

The crew scheduling problem is typically divided into two subproblems. First, the *crew pairing problem* is solved. In this problem, we select a set of pairings such that each flight is included in exactly one pairing and pay-and-credit is minimized. Then, the *crew assignment problem* is solved. In this problem, the chosen pairings are combined with rest periods, vacations, training time, and other breaks to create

extended individual work schedules, typically spanning a period of about one month.

14.2.2.1 The Crew Pairing Problem. The domestic U.S. crew pairing problem is typically solved in three stages: *daily, weekly exceptions,* and *transition.*

The first stage, the daily problem, considers the set of flights which are flown at least four days per week. In this first stage, we treat these flights as though they all operate daily. We therefore want to find a minimum cost set of feasible pairings such that every flight in this set is covered exactly once. The pairings in this solution are then assumed to be repeated daily. The daily problem forms a good approximation since in the U.S. most of the flights operate every day, with a few exceptions on weekends.

For pairings that span multiple days, we assume that one crew will be assigned to each of the different duties within that pairing on any given day. For example, suppose the solution includes a three-day pairing made up of duties A, B, and C. On any given day, there will be one crew starting their trip with duty period A, another crew that began the trip the day before and is now covering duty period B, and a third crew on the final day of their trip, covering duty period C. This, in conjunction with the constraint that pairings cannot cover the same flight more than once, ensures that on any given day, every flight will be covered by exactly one crew.

Note that a solution to the daily problem will not be completely feasible in practice, because it assumes that all flights are flown every day of the week. Pairings that cannot be flown on certain days of the week because one or more of the flights do not operate on that particular day are referred to as *broken pairings.* The second crew pairing stage, the weekly exceptions problem, constructs new pairings to correct these broken pairings and also to cover those flights that are flown three or fewer days per week. Thus, in the weekly exceptions problem, we must associate flights with a specific day of the week. Accordingly, pairings become specific to days-of-week as well. Typically, deadhead flights are also needed in order to find good solutions to the weekly exceptions problem. Combined, the daily and weekly exceptions pairing solutions cover each flight in the weekly schedule exactly once.

Finally, note that airlines change their flight offerings on a regular basis, often quarterly and, to some degree, even monthly. Therefore, multi-day pairings can be problematic at the end of a monthly flight schedule. For example, on the last day of the month, a new crew must begin each pairing to cover that day's flights. However, the remaining

days of the pairing may not be valid given that different flights might be offered in the next month's schedule. We therefore must solve a third stage of the crew pairing problem, the transition problem. This problem constructs pairings to cover flights for a small number of days spanning the changeover from one monthly flight schedule to another.

In all three of these problems, we emphasize that the object is to minimize pay-and-credit, the labor costs above and beyond the minimum required flying time.

The three stages described above are typical of U.S. domestic problems. More generally, we can think of two types of crew pairing problems, *weekly problems* and *dated problems*. Weekly problems yield sets of pairings that are repeated weekly; pairings that start at the end of the week wrap back around to the beginning of the week. Dated problems, on the other hand, correspond to specific days of the month. In the U.S. case, the daily pairing problem and weekly exceptions problem collectively solve the weekly problem, while the transition problem is a dated problem.

14.2.2.2 The Crew Assignment Problem. Given the solution to the crew pairing problem, i.e., a minimum cost set of pairings that cover all flights throughout a monthly period, we must then assign specific individuals to these pairings. This occurs in the *crew assignment problem*.

Just as the crew pairing problem selects a minimum cost set of pairings (strings of sequential flights that satisfy a variety of rules) such that every flight is covered, the crew assignment problem selects a set of schedules (strings of sequential pairings that satisfy a variety of rules) such that every pairing is covered. In this context, a pairing corresponds to specific days in the schedule.

In spite of their similarities, these two problems are addressed separately, both in industry and in the academic literature. There are two primary reasons for this. First, in the crew pairing problem we assign complete crews to flights, while in the crew assignment problem crew members are scheduled individually, with each pairing being covered by multiple crew members. Second, the crew pairing problem focuses on minimizing labor costs, while in the crew assignment problem greater emphasis is placed on satisfying crew requests and seeking a balanced distribution of work.

In the U.S., the crew assignment problem is solved in two stages. In the first stage, a set of schedules is constructed such that each pairing is included in exactly as many schedules as are needed to fully staff the flight. Then, in the second stage, these schedules are assigned to

individual crew members using a *bidline approach*, where schedules are allocated to crew members through a system in which crew members bid on their preferred work schedules. The schedules are then awarded by the airline based on crew priority, often related to seniority.

In Europe, on the other hand, individualized schedules, called *rosters*, are often constructed directly, taking into consideration the particular needs or requests of each crew member and, in some cases, crew seniority as well.

14.3 Formulations

14.3.1 The Crew Pairing Problem

Crew pairing models are typically formulated as *set partitioning problems*, in which we want to find a minimum cost subset of the feasible pairings such that every flight segment is included in exactly one chosen pairing.

Let F be the set of flight segments to be covered and let P be the set of all feasible pairings. Decision variable y_p is equal to 1 if pairing p is included in the solution, and 0 otherwise. Column p has a 1 in row i of the constraint matrix if flight i is included in pairing p and a 0 otherwise.

The crew pairing problem is

$$\begin{aligned} \min & \sum_{p \in P} c_p y_p \\ & \sum_{p: i \in p} y_p = 1 \qquad\qquad i \in F \qquad\qquad (14.1) \\ & y_p \in \{0,1\} \qquad\qquad p \in P\,. \end{aligned}$$

Note that this formulation requires the explicit enumeration of all pairings. Enumerating pairings can be difficult both because of the numerous work rules that must be checked to ensure legality and, more importantly, because of the huge number of potential pairings. In fact, for most real instances, explicit enumeration of the constraint matrix is not possible. For example, a domestic problem on a hub-and-spoke network with several hundred flights typically has billions of pairings. Thus, heuristic local optimization approaches or column generation methods (described in Section 14.4) are used to solve all but the smallest of problem instances.

This basic set partitioning model is used for all three phases of crew pairing optimization. The models differ in the set of flights F that define the constraints of the problem. For the daily problem, there is one constraint for each flight that is repeated four or more times per

week. The underlying assumption in solving this problem is that each pairing in the solution will be flown starting each day of the week. Recall that this presupposes that pairings are constrained to cover a flight leg at most once.

In the weekly and dated problems, F contains all of the flights in the flight schedule. In the weekly problem the flights are associated with a specific day of the week whereas in the dated problem they are associated with a specific date. Note that in the dated problems, we can relax the restriction that a pairing does not cover the same flight more than once since flights are associated with specific dates in these problems. For weekly problems, the same relaxation is valid for all of the pairings with time away from base shorter than a week.

14.3.1.1 Balancing Constraints. Many airlines also add *crewbase balancing constraints* to the basic crew pairing model. These constraints ensure that the distribution of work over the set of crewbases is matched to the crew resources. They require that the number of hours of work contained in the chosen pairings which originate at a given crewbase must be between specified lower and upper bounds, which are a function of the number of crews stationed at that crewbase. Constraints of this form are known as *two-sided knapsack constraints*.

Example. We illustrate the crew pairing formulation using an example composed of the following seven flights.

Flight #	**Orig**	**Dest**	**Start**	**End**	**Frequency**
1	A	B	08:00	09:00	not 67
2	B	C	10:00	11:00	
3	C	D	13:00	14:00	not 7
4	C	A	15:00	16:00	
5	D	A	15:00	16:00	not 6
6	A	B	17:00	18:00	
7	B	C	11:00	12:00	not 67

The last column indicates the flight schedule. For example, flight 1 is operated every week day, while flight 5 is operated every day except Saturday. We assume, for simplicity, that all of the airports are in the same time zone.

We first consider the daily problem. Suppose that the valid duty periods are

$$\begin{array}{llll} D_1 = \{1\} & D_2 = \{2\} & D_3 = \{3\} & D_4 = \{4\} \\ D_5 = \{5\} & D_6 = \{6\} & D_7 = \{7\} & D_8 = \{1,2\} \\ D_9 = \{1,7,3\} & D_{10} = \{2,3\}\,. & & \end{array}$$

Assuming that airports A, C, and D are crewbases, we have six pairings, which can be expressed in terms of the duty periods as

$$\begin{array}{lll} P_1 = \{D_4, D_8\} & P_2 = \{D_9, D_5\} & P_3 = \{D_5, D_6, D_{10}\} \\ P_4 = \{D_4, D_6, D_7\} & P_5 = \{D_1, D_7, D_4\} & P_6 = \{D_5, D_7, D_9\}\,. \end{array}$$

Pairing P_6 covers flight 7 twice and therefore it is not considered in the daily problem. Notice that an additional pairing could have been defined by the set of duties $\{D_4, D_1, D_2\}$. However, this pairing covers the same flights as pairing P_1. Given that both pairings originate at the same crewbase, only one of the two pairings (the less costly) need to appear in the model. In this example, we assume that $\{D_4, D_8\}$ has lower cost than $\{D_4, D_1, D_2\}$.

Assuming pairing costs $c_1 = c_2 = c_3 = c_4 = 4$ and $c_5 = 5$, from (14.1) we obtain the following formulation.

$$\begin{array}{llllllll} \min & 4y_1 & +4y_2 & +4y_3 & +4y_4 & +5y_5 & & \\ & y_1 & +y_2 & & & +y_5 & = \quad 1 & \text{(flight 1)} \\ & y_1 & & +y_3 & & & = \quad 1 & \text{(flight 2)} \\ & & y_2 & +y_3 & & & = \quad 1 & \text{(flight 3)} \\ & y_1 & & & +y_4 & +y_5 & = \quad 1 & \text{(flight 4)} \\ & & y_2 & +y_3 & & & = \quad 1 & \text{(flight 5)} \\ & & & y_3 & +y_4 & & = \quad 1 & \text{(flight 6)} \\ & & y_2 & & +y_4 & +y_5 & = \quad 1 & \text{(flight 7)} \\ & y_1, & y_2, & y_3, & y_4, & y_5 & \in \quad \{0,1\} & \end{array}$$

If we require that at least 3 hours and at most 6 hours of pay be assigned to crewbases A and D, and at most 5 hours of pay to crewbase

C, then the crew balance constraints are

$$\begin{aligned} 3 \leq & 4y_2 + 3y_5 \leq 6 && \text{(Crewbase A)} \\ 0 \leq & 3y_1 + 3y_4 \leq 5 && \text{(Crewbase C)} \\ 3 \leq & \quad 4y_3 \leq 6 && \text{(Crewbase D).} \end{aligned}$$

An optimal solution to this problem uses pairings 3 and 5, for a total cost of 9.

To obtain a solution to the weekly problem, we need, in addition to solving the daily problem, to solve a weekly exceptions problem to repair the broken pairings. The weekly exceptions problem consists of the following flights.

	P_3	P_5
Friday	5	1
Saturday	6	
Sunday	6,2	4
Monday	2,3	7,4
Tuesday		4

Alternatively, we could have solved the problem as a single weekly problem. Such a problem would have 43 flight segments and thus 43 cover constraints. Each pairing would appear multiple times, associated with the appropriate days of the week. For example, pairing P_1 would have five copies, one starting on each day of the week except Friday and Saturday. The copy that starts on Sunday wraps around in time since the next duty period is on Monday. In addition, this weekly problem also includes the pairing P_6. □

14.3.2 The Crew Assignment Problem

In this section, we explain the rostering problem, which has been the focus of much of the crew assignment literature.

Separate rostering problems are solved for each *crew type*, where a crew type is specified both by the crew member rank (such as Captain, First Officer, Flight Engineer, etc.) and the fleet family (such as Boeing 767, Airbus 320, etc.) the crew members are qualified to fly. For a given crew type, the model input includes the set of pairings that must commence each day, and the number of crew members of the specified type that must be assigned to each of these pairings. The constraints of the rostering model require that:

1 Each pairing in the crew pairing solution is contained in the appropriate number of selected schedules. Note that the rostering

model contains one constraint for each pairing commencing on a given day, for each day in the rostering period.

2 Each crew member is assigned to exactly one work schedule. If the airline is not required to use all crew members, a crew member might be assigned to an *empty* or *null* schedule – that is, a schedule containing no work.

Let K be the set of crew members of a given type, let S^k be the set of work schedules that are feasible for employee $k \in K$, and let P be the set of dated pairings to be covered. [A *dated pairing* is a pairing together with the starting date of the pairing.] n_p represents the minimum number of crew members that must be assigned to pairing $p \in P$ and γ_p^s is 1 if pairing $p \in P$ is included in schedule s and 0 otherwise. Decision variable x_s^k equals 1 if schedule $s \in S^k$ is assigned to employee $k \in K$ and 0 otherwise. c_s^k, the cost of schedule $s \in S^k$ for employee $k \in K$, represents the schedule cost, which might represent how close the schedule is to the crew member's stated preferences, or be set so as to minimize the number of crew members used. The latter is done by assigning very low costs to null assignments.

Given this notation, we write the crew rostering formulation, Gamache and Soumis, 1998, for a given crew member type as

$$\begin{aligned}
&\min \sum_{k \in K} \sum_{s \in S^k} c_s^k x_s^k \\
&\sum_{k \in K} \sum_{s \in S^k} \gamma_p^s x_s^k \geq n_p && \text{for all } p \in P \qquad (14.2)\\
&\sum_{s \in S^k} x_s^k = 1 && \text{for all } k \in K \\
&x_s^k \in \{0,1\} && \text{for all } s \in S^k, \text{ for all } k \in K\,.
\end{aligned}$$

14.4 Solution Algorithms

At their core, the crew pairing and crew assignment models are set partitioning and set covering models with one constraint for each task to be performed (i.e. a flight or pairing to be covered) and one variable for each feasible combination of the tasks.

These problems are difficult for three reasons. First, even determining whether a combination of tasks is feasible can be difficult, given the wide array of rules and regulations that must be enforced. Second, these problems often have an enormous number of variables – often in the hundreds of millions or more. Third, these variables are all integer, further complicating the solution process.

In this section, we discuss solution approaches to address these difficulties. For the purpose of illustration, we focus our attention on the crew pairing problem. However, these ideas are applicable to the crew assignment problem as well.

14.4.1 Historical Solution Approaches

One of the major challenges in solving the crew pairing problem arises from the shear size of the problem. It is not uncommon for a problem containing 300 flights to have billions of pairings. Consequently, in early work on the pairing problem, only a subset of the pairings were constructed, using heuristic rules of thumb to guide and limit the construction process. In fact, until the 1990's, local improvement heuristics represented the state-of-the-art in crew pairing optimization.

A local improvement heuristic for the crew pairing problem starts with a feasible solution to the set partitioning problem. Because most airlines only make minor changes to their flight schedules from one planning period to the next, feasible solutions can usually be constructed manually by modifying the solution used in the previous planning period. Then, to find improved schedules, the heuristic randomly selects a small number of pairings in the current solution and searches for a better solution for the flights covered by that subset of the pairings. The search is usually performed by enumerating all possible pairings for the subset of flights and solving the small set partitioning IP to optimality using branch-and-bound. Often, these small set partitioning problems can be solved quickly since the LP relaxations of set partitioning problems with a small number of rows frequently have integral or near-integral solutions. This process is repeated until no further improvements are found or until some preset time limit is reached. This approach is taken in Anbil et al., 1991 and Gershkoff, 1989.

14.4.2 Pairing Generation

In each iteration of a local search heuristic the current incumbent solution is improved by considering only a small subset of the flights and the pairings covering only these flights. Therefore these heuristics lack the ability to consider the whole flight network in a single step and they need a large number of iterations before finding a good solution. An additional drawback of the local search heuristics is that they do not provide a lower bound on the best possible solution value. Thus, it is hard to estimate how far the current solution is from the optimum. To circumvent these two obstacles more global approaches are needed where at each iteration pairings covering all of the flights are generated.

14.4.2.1 Network Structure for Pairing Generation.

There are two main types of networks that have been developed in the literature for generating pairings. The first, called a *flight network,* has an arc for each flight in the schedule and arcs representing possible connections between flights. The second type of network, a *duty period network,* has an arc for each possible duty period and arcs representing possible overnight connections between the duties.

The network used to model international problems is typically *duty period,* rather than *flight,* based. That is, nodes represent the start or end of a duty period and an arc is included in the network for each possible duty period. *Connection arcs* between duties are included if two duties can be flown consecutively by the same crew. Domestic operations, on the other hand, typically use flight networks, because of the large number of feasible duties.

Each crew pairing is represented by a network path, but only the subset of paths satisfying certain requirements represent pairings. For example, in a flight network, a sequence of flights may give a path through the network, but that does not guarantee that the resulting duty periods will contain no more than the allowable hours of flying or that the resulting pairing will contain fewer than the maximum number of duty periods allowed.

Flight Network A typical flight network, Minoux, 1984, Desrosiers et al., 1991, has nodes representing the departure and arrival of each flight as well as a source s and a sink t. There is an arc representing each flight in the schedule. If there is a sparsity of flights arriving and departing an airport, it is often necessary to include potential deadhead flights in order to achieve a good, or even feasible, solution. For daily problems each flight arc is replicated as many times as the maximum number of calendar days allowed in a pairing but pairings are generated only from flights operating on the first day. For weekly problems, flight arcs are replicated as many times as the maximum number of weeks allowed in a pairing. Because pairings are often not allowed to exceed one week in duration, flight arcs need to be replicated only once so that pairings that cross over from the end of the week to the beginning of the next week, i.e. from Sunday to Monday, can be generated. Of course we could allow pairings to cross over without replicating flights by introducing arcs from the last day of the week back to the first. However, maintaining an acyclic network by replicating flights simplifies the shortest path algorithm for finding attractive pairings.

The source node is connected to the departure node of each flight that originates at a specified crewbase. The arrival node of every flight

that ends at that crewbase is connected to the sink. There are also arcs representing legal connections between flights. A pair of flights will have a connection arc between them if the arrival airport of the first is the same as the departure airport of the second and the time between the two flights is either a legal connection within a duty period or a legal overnight rest. Note, however, that the required duration of an overnight rest might be a function of the attributes of the duty period that precedes it and plus perhaps other attributes of the pairing.

Figure 14.2 shows a partial flight network for the following flight schedule.

Flight 1: AIRPORT A – AIRPORT B 08:00 – 09:00
Flight 2: AIRPORT B – AIRPORT C 10:00 – 11:00
Flight 3: AIRPORT C – AIRPORT D 13:00 – 14:00
Flight 4: AIRPORT D – AIRPORT A 15:00 – 16:00

The network spans a two-day time horizon and contains two copies of each flight. The solid arcs represent flights. With each flight we have a transition through both time and space from the departure airport and departure time to the arrival airport and arrival time. The dotted arcs represent possible connections between flights. Of course, connections are only allowed between pairs of flights that arrive and depart from the same airport. Because of minimum and maximum connection time limits, the set of connection arcs in the network will generally be a proper subset of the set of all possible connections. Note that in the figure each arrival node has two connections emanating from it, one to the next departure and the other to the same departing flight one day later. In order to generate pairings originating at a crewbase, for example, Airport A, we would add a source node s and sink node t to this network. We would then connect s to the departure node of every flight arc originating at Airport A and connect the arrival node of every flight arriving at A to node t.

It is easy to see that every legal pairing is represented by some $s - t$ path in this network. However, there are many $s - t$ paths that do not represent legal pairings. The network structure guarantees that we will not connect two flights that do not have their respective arrival and departure at the same airport, but it does not prevent us from violating other rules like the maximum number of hours of flying allowed in a duty period or the maximum TAFB in a pairing.

Using a duty period network it is possible to build the duty period rules into the network, resulting in a much larger arc set.

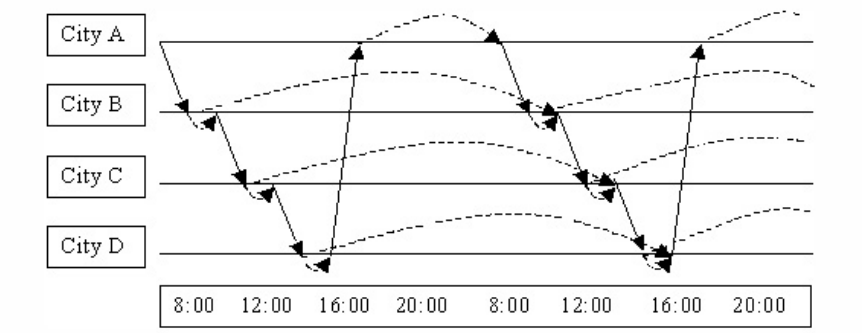

Figure 14.2. Flight Network

Duty Period Network This network, Lavoie et al., 1988, Anbil et al., 1994, Vance et al., 1997b, has nodes representing the departure and arrival of each duty period as well as a source and sink. There are arcs representing each possible duty period in the flight schedule as well as arcs representing legal connections between duties. For daily problems, each duty period is replicated as many times as the maximum number of calendar days allowed in a pairing. Similarly, for weekly problems, each duty period is replicated as many times as the maximum number of weeks allowed in a pairing.

A pair of duties will have a connection arc between them if the arrival airport of the first is the same as the departure airport of the second and the time between them is a legal overnight rest. Remember that the required duration of an overnight rest might be a function of the attributes of the duty period that precedes it and possibly other attributes of the pairing. With the duty period network, unlike the flight network, it is possible to build explicitly into the network the requirements involving the preceding duty period.

For daily problems, the no repeated flight rule can be *partially* enforced by allowing connection arcs only between duties that do not share a common flight. That is, we can ensure that no two consecutive duties share a common flight, but for nonconsecutive duties, e.g. the first and third duties, we cannot prevent flight legs from being repeated in this manner.

Klabjan et al., 2001b propose an approach to store the network compactly. They assume that the duty periods are sorted based on the departure airport and the duty periods originating at the same airport are sorted in increasing order of the departure times. For each node rep-

resenting a duty period arrival they store two pointers. The first pointer points to the earliest connecting duty period and the second pointer to the last connecting duty period. Due to the imposed order on duty periods, all possible connections are obtained by scanning all duty periods between the two stored duty periods. While compact, this network representation cannot embed into the network rules that involve two duty periods, for example, sharing common flights between two duty periods.

Figure 14.3 shows a two-day duty period network for the schedule shown in Figure 14.2. The solid arcs represent duty periods and the dotted ones represent connections between duties. The lighter solid arcs are the single-flight duty periods corresponding to each of the four flights in the schedule while the darker solid lines correspond to two additional duties, one composed of flights 1 and 2 and the other composed of flights 3 and 4. It is possible to build more duty periods from this set of four flights, but we have chosen to add only two to maintain the simplicity of the example. Note that the single flight duty period arcs arrive much later than the corresponding flight arcs in the flight network. This is because we include the time of the overnight rest in the duration of the duty arc.

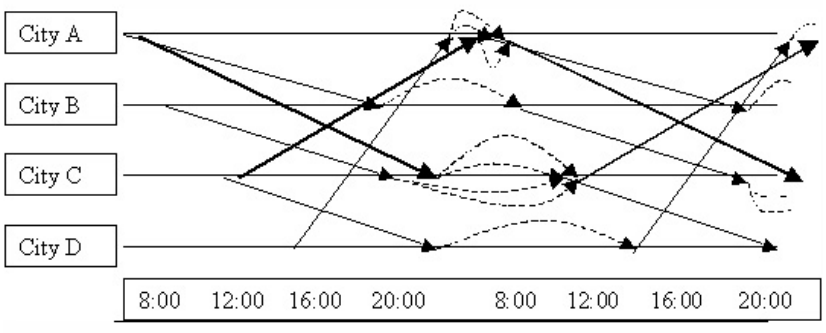

Figure 14.3. Duty Period Network

To generate pairings in the flight network starting and ending at a crewbase, we add a source and sink node. The source node is connected to the departure node of each duty period that originates at the specified crewbase. The arrival node of every duty period that ends at that crewbase is connected to the sink.

Many more rules are satisfied by all paths from source to sink in the duty period network than the flight network. However, there are still some rules, such as the 8-in-24 rule and the no repeating flight rule

for nonconsecutive duties, that cannot be enforced through the network structure.

14.4.2.2 Pairing Enumeration. Duty period enumeration can be accomplished by a depth-first search approach on the flight network. For each flight arc we construct all duty periods that start with this flight. We attempt to extend the duty period with a flight if there is a corresponding sit connection arc in the flight network and all of the other duty feasibility rules are satisfied. In order to enumerate all duty periods we have to backtrack whenever we have exploited all sit connection arcs originating at a node corresponding to a flight arrival.

Pairings can be enumerated in a similar way either from the flight network or from the duty period network. In pairing enumeration the generation is started from every flight or duty that originates at a crew base. Depth-first search is then used to extend partial pairings or backtrack.

Partial Generation of Pairings Several methodologies for the crew pairing problem require a generation of only a subset of pairings since all of them cannot be handled explicitly. An easy way to achieve this is by generating pairings only on a subset of flights. This approach is taken in Anbil et al., 1991 and Gershkoff, 1989. It is substantially more difficult to generate a subset of pairings that cover all of the flights in the schedule. Andersson et al., 1998 give some details on how this operation is carried out at Carmen Crew Pairing. To generate a subset of pairings, for each flight they limit the number of possible connections. A certain number of short connections is selected and possibly some historically useful connections. Their idea is to solve a flight matching problem for each airport before pairing enumeration. Knowledge of 'good' connections is essential. Moreover, an experienced user might also prune the generation by recognizing useless connections.

Klabjan et al., 2001b propose the generation of random pairings. When extending a branch during enumeration, they choose connections at random. They use the connection times as greedy estimates; that is, the probability of selecting a connection depends on the connection time. Given that short connections are more likely to yield pairings with low cost, the smaller the connection time, the larger the probability of selecting the connection. Because in hub-and-spoke flight networks there are many connections, the connection selection strategy has to be implemented carefully. They propose a similar approach for generating random duties. Based on this random pairing generation they develop an algorithm for the crew pairing problem.

14.4.3 Solving the LP relaxation

Early work on approximately solving the LP relaxation of (14.1) involves considering a large number of pairings and solving the LP relaxation over these pairings. Anbil et al., 1992 found an optimal solution to the LP relaxation of (14.1) over a large subset of the pairings for an 800 flight instance of a U.S. domestic daily problem. Five and a half million feasible pairings were enumerated and the optimal LP solution was found over this set using a specialized approach referred to as *SPRINT* in which several thousand columns are loaded into the LP solver and the LP is optimized over those columns. Then, most of the nonbasic columns are discarded, and several thousand more columns are added. This process is continued until all columns have been considered. At the end, however, it is necessary to price out all nonbasic columns to prove optimality. Bixby et al., 1992 used a combination of an interior point method and the simplex method to find the optimal LP solution to a very large crew pairing model. Hu and Johnson, 1999 propose a primal-dual algorithm for solving the LP relaxation over a given number of pairings. Their algorithm maintains a dual feasible vector and in every iteration increases the objective value by considering convex combinations of dual solutions.

As optimization solvers and computers became more sophisticated, there was a shift to dynamic column generation techniques that implicitly consider all possible pairings in solving the LP relaxation, Anbil et al., 1994, Desaulniers et al., 1998. In column generation the set partitioning problem with all possible pairings is referred to as the *master problem*. Thus (14.1) is the master problem. A *restricted* master problem is one that contains only a subset of the possible pairing columns. The *column generation algorithm* to solve the crew pairing LP involves the following steps:

- *Step 1: Solve the Restricted Master Problem*– Find the optimal solution to the current restricted master problem containing only a subset of all columns.

- *Step 2: Solve the Pricing Subproblem*– Generate one or more columns that may improve the solution. If no columns are found, STOP: the LP relaxation is solved.

- *Step 3: Construct a New Restricted Master Problem*– Add to the restricted master problem the columns generated in solving the subproblem and return to Step 1.

The solution and construction of the restricted master problem (steps 1 and 3) can be achieved using optimization software such as CPLEX or OSL. The solution of the pricing subproblem (step 2), however, should be tailored to exploit the network structure of the problem. The idea is to represent every pairing as a path in a network so that the huge number of pairings can be represented efficiently. This network is then used to identify variables that may improve the solution, without examining all variables. This often can be achieved either by solving *multi-label shortest path problems* on the specially structured network, Desrochers and Soumis, 1989, or by enumeration, Marsten, 1994 and Makri and Klabjan, 2001.

14.4.3.1 Considerations in Solving the Restricted Master Subproblem. Until recently it was believed that the primal simplex algorithm is the most efficient procedure for solving the restricted master subproblem. Given that a primal solution is available from the previous iteration, primal simplex can be warm started. However primal simplex has two drawbacks. First, the LP relaxations of (14.1) tend to be extremely degenerate and therefore primal simplex tend to perform many degenerate steps, and second, the extreme point optimal dual solutions give misleading reduced costs and several iterations of column generation are needed. A standard trick for removing degeneracy is to randomly perturb the right hand sides. After the perturbed LP is solved, the perturbation is removed and the LP is solved to optimality. A different approach is presented in du Merle et al., 1999 by adding surplus and slack variables with penalties. This corresponds to requiring soft lower and upper bounds on the dual variables and penalizing the dual variables if they lie outside the bounds. For crew pairing problems this technique substantially reduces the number of iterations in column generation. Interior point algorithms yield an interior point dual solution, which is a much better indicator of the 'usefulness' of a column, but lack the benefit of warm starting.

Barahona and Anbil, 1998 and Barahona and Anbil, 1999 propose a variant of the subgradient algorithm, which they call the *volume algorithm*. At every iteration the dual vector is improved in the direction of the subgradient and a primal feasible solution is obtained by taking a convex combination of previously obtained primal feasible vectors. The volume algorithm is fast, does not have large memory requirements, and it produces excellent dual vectors for use in column generation. Barahona and Anbil, 1999 claim significant performance improvements over both interior point and simplex algorithms.

14.4.3.2 Pricing. Pricing is the problem of selecting pairings that are then added to the restricted master problem in step 2. There are two main questions in pricing: what is the criteria to select the pairings and how to find pairings that meet the criteria.

Traditionally, pairings are selected by the reduced cost criterion. Recently alternative strategies have been proposed. Bixby et al., 1992 use the pairing cost divided by the sum of the dual values of the legs in the pairing as the selection criteria. They report a significant decrease in the number of iterations. Hu and Johnson, 1999 present a primal-dual algorithm to select the columns with the lowest reduced cost based on a dual feasible vector, which is updated in every iteration. They also report a significantly lower number of iterations. Another, yet unexplored, strategy in the context of column generation would be to use the steepest edge pricing rule, Forrest and Goldfarb, 1992.

There are two approaches for finding pairings that best meet the selection criteria. One is combinatorial by using a shortest path algorithm, and the second is the brute force approach of enumerating the pairings. In the following sections we describe both approaches.

Pricing with Shortest Path Algorithms Until recently shortest path approaches have been designed only for the reduced cost criterion. Many algorithms solve the pricing problem to find attractive pairings using multilabel or constrained shortest path methods on specially structured networks, Desrochers and Soumis, 1988. In either the flight or the duty period network, only basic requirements can be built into the network structure. Requirements that cannot be built into the network structure are enforced through the use of labels. For example, we can maintain a label to track the number of hours of flying in the current duty period, the number of duties in the pairing, and the 8-in-24 rule. In addition to the labels that control the pairing feasibility rules, we need labels to capture the nonlinear components of the pairing cost structure.

Multilabel shortest path approaches differ from single-label approaches in that it might be necessary to keep many paths to each intermediate node in the network. For example, in solving the crew pairing problem, often it is not known which of the cost factors will dominate or which rules might prevent a path from resulting in a pairing until the complete pairing is specified. Consequently, it is necessary to keep track of all *nondominated paths* to each node between s and t. A path is nondominated if there does not exist another single path which is 'better' with respect to all the costs and rules. By better, we mean that either it is cheaper with respect to one of the cost criteria, or it is less restricted with respect to one of the rules. For example, if two paths to the same node have

all labels identical except one has more time-away-from-base than the other, by dominance the one with the larger time-away-from-base can be eliminated.

Figure 14.4 illustrates a label update in a multilabel shortest path procedure. Each path carries four labels: the first gives the flying time in the current duty period, the second the elapsed time in the current duty period, the third the number of segments in the current duty period, and the final one, the number of duties in the pairing. At the arrival node of arc A the label values are (3.0, 6.0, 2, 1). For the arrival node of arc B the labels are (4.0, 5.0, 4, 1). Now consider the departure node of arc C. Two connection arcs both terminate at that node. The connection arc from flight A has a duration of two hours and the connection from flight B has a duration of one hour. Thus the two possible paths will have labels (3.0, 8.0, 2, 1) and (4.0, 7.0, 4, 1) respectively. Neither path can be eliminated by dominance because one contains less flying time and the other less elapsed time. Thus, we must now maintain two sets of labels at the departure node of flight C. Note that in this simple example we have only a small number of labels. As the number of labels grows, it is generally more difficult to eliminate paths by dominance so that a large number of potential paths to each node must be stored.

Lavoie et al., 1988 and others were successful in using the multi-label shortest path procedure to solve the pricing subproblem over duty-based networks. This approach works especially well when the number of duty periods is not excessive, as demonstrated by Anbil et al., 1994, who used a duty-based network to solve international crew problems containing about two to three times as many duties as flights. Vance et al., 1997a were also successful in using a duty period network to solve a relatively small domestic daily problem.

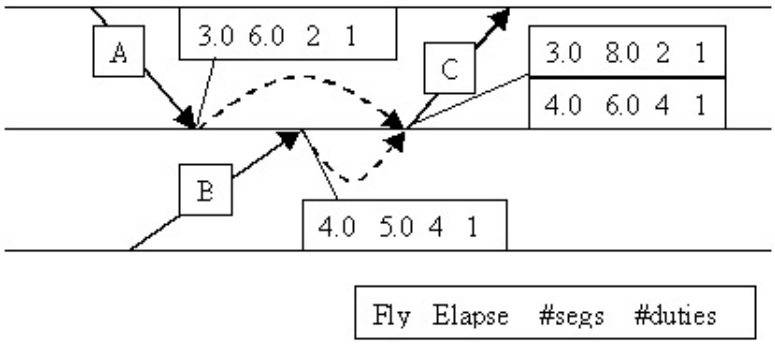

Figure 14.4. Constrained Shortest Path Example

Pricing By Enumeration The approach of generating all the pairings is an alternative solution to pricing. Note that the pairing feasibility rules and the cost structure are very complex and therefore a shortest path approach typically requires many labels, which makes dominated paths a rare occurrence. In addition, it might not even be possible to capture some of the feasibility rules with labels. If a change or an update of a feasibility rule is required, major changes in the shortest path code might be necessary. Therefore crew scheduling software vendors prefer to use pairing enumeration in pricing because they have many customers, each with its own feasibility rules.

For medium and large crew pairing instances enumerating all the pairings can be prohibitive and therefore strategies have to be designed to avoid this. Marsten, 1994 and Anbil et al., 1998 describe crew pairing optimizers that use partial enumeration in pricing. Both of these approaches use the reduced cost criterion.

Makri and Klabjan, 2001 use the selection criterion, introduced by Bixby et al., 1992,

$$\min_{p \in P} \{ \frac{c_p}{\sum_{i \in p} y_i^*} | \sum_{i \in p} y_i^* > 0 \} ,$$

where y^* is the optimal dual vector to the restricted master subproblem. Pairings are enumerated, however they prune the enumeration by providing upper bounds on the score of a pairing p, which is defined as $\frac{c_p}{\sum_{i \in p} y_i^*}$.

14.4.4 Finding Good Solutions to the IP

Most state-of-the-art approaches combine column generation for solving the LP relaxation of the set partitioning problem with a branch-and-bound algorithm to find good integer solutions. Because of the large number of possible pairings, for all but the smallest problems, these approaches are heuristic in nature. They fall into one of three general classes. In the first class are algorithms where column generation is performed 'off-line'. That is, a subset of pairings is enumerated up front and the integer program is solved to optimality over this subset. An example of this type of approach can be found in Hoffman and Padberg, 1993. Because even moderate sized problems can have billions of variables, these approaches must work on a very small subset.

The second class of approaches uses dynamic column generation to solve the LP relaxation of the set partitioning problem to optimality or near optimality. Then, branch-and-bound is applied to obtain the optimal IP solution over the subset of columns generated to solve the

LP relaxation. Among these approaches is work by Anbil et al., 1994 on the international crew pairing problem and Ryan, 1992 on the rostering problem. Recently Klabjan et al., 2001b proposed an algorithm that solves the LP relaxation of (14.1) and then selects several million pairings with low reduced cost to find an integer solution.

The drawback to these approaches is that there is no guarantee that a good solution, or even a feasible solution, exists among a subset of columns that give a good LP solution.

A third class of algorithms allow dynamic column generation throughout the branch-and-bound tree. We refer to algorithms of this type as *branch-and-price* approaches. Like branch-and-bound, a branch-and-price procedure is a smart enumeration strategy in which an LP relaxation is solved at each node of a branch-and-bound tree. The difference is that the huge constraint matrix requires the use of column generation. Branch-and-price methodology has been applied to a number of problems in transportation, scheduling, and combinatorial optimization. For a survey, see Barnhart et al., 1998.

Recently, a number of groups have developed crew pairing and rostering algorithms using a branch-and-bound framework with column generation, including Desaulniers et al., 1998, Desrosiers et al., 1991, Gamache et al., 1999, Gamache et al., 1998, Gamache and Soumis, 1998, Ryan, 1992, Vance et al., 1997a, and Anbil et al., 1998.

Marsten, 1994 combines dynamic pairing generation with variable fixing to obtain good integer solutions. To find integer solutions, the variables associated with fractional pairings with value close to one are fixed to one sequentially. To limit column generation, new pairings are generated only when the bound from the LP relaxation increases above a pre-set target.

Andersson et al., 1998 decouple pairing generation from the optimization engine. The algorithm generates pairings several times and solves (14.1) over the generated columns. They use the Lagrangian algorithm presented in Wedelin, 1995 to solve the integer programs.

14.4.4.1 Branching Rules for the Crew Pairing Problem.

To be able to generate pairings at any node in the branch-and-bound tree, a branching rule that is compatible with the pairing generation procedure is needed. The standard rule of branching on variable dichotomy is difficult to implement. Using such a branching decision, we would either fix a pairing into the solution ($x_j = 1$) or forbid the use of a pairing ($x_j = 0$) with each decision. It is easy to fix a pairing j into the solution. There is no need to generate any more pairings containing any of the flights covered by pairing j, so these flight arcs may be deleted

from the pairing generation network. However, forbidding the use of a specific pairing is difficult since we must forbid specific paths from being returned by the pairing generation procedure. This could require finding the $(k+1)$st shortest path if k pairings have been forbidden by branching.

The first branching rule presented here is motivated by a general rule for set partitioning problems developed by Ryan and Foster, 1981. Their rule is based on the simple observation that given a fractional solution to the LP relaxation of a set partitioning problem there must exist two columns whose associated variables are fractional such that they both contain coefficients of one in a common row r and there exists another row s where one column has a coefficient of one and the other has a coefficient of zero. This fact leads to a general branching rule where pairs of rows r and s are required to be covered by the same column on one branch and by different columns on the other.

Essentially the same logic can be used for crew pairing optimization, but the rule is modified to maintain tractability. In general, it is difficult to force two specific flights to either appear only in pairings that contain them both (the first branch) or to never appear together in the same pairing (the second branch). However, it is an easy matter to force two flights to appear consecutively in a pairing or not. If flights r and s satisfy the conditions of the Ryan and Foster rule and they appear consecutively in at least one of the fractional pairings that contains them both, branching can be performed by requiring that they appear consecutively in the pairing that covers them at one node and by requiring that they cannot appear consecutively in any pairing in the solution at the other node. This strategy is sometimes referred to as branching on *follow-ons* because it places restrictions on which flights can follow flight r in the solution. The flight pair r, s is often termed a *follow-on.*

Klabjan et al., 2001b present a different branching rule called *timeline branching.* In timeline branching the decision is based on a flight r and a time t. In one branch we only allow pairings with time of the connection immediately following r less than or equal to t. The other branch considers only pairings with time greater than t of the connection immediately following r. They show that this is a valid branching rule if the departure times are all different, which can be achieved by slightly perturbing them. They also combine follow-on branching or timeline branching with strong branching. Strong branching is a branching rule that chooses the branching variable (follow-on in the context of crew pairing) by carrying out several dual simplex iterations for each branching candidate to estimate the change in the lower bound, Bixby et al., 1995, Linderoth and Savelsbergh, 1999.

Master Problem Modification. We discuss the modifications to the master problem for follow-on branching. For timeline branching the modifications are similar. To implement follow-on branching, both the master problem with its existing set of columns and the column generation subproblem must be modified. On the branch where flight r must be followed by flight s in the same pairing, any pairing in the restricted master problem that contains r and/or s but does not have the two flights appearing consecutively is eliminated. On the branch where the flights cannot appear consecutively, any pairing with r and s consecutive is eliminated from the restricted master problem.

Flight Network Modification. If a flight network is used, flight s can be required to follow flight r by eliminating all the connection arcs out of r except the one to s. Connections into s from any flight other than r are also deleted. Note that this second modification is not absolutely necessary since requiring r to be followed by s is sufficient to ensure that s will not be preceded by a flight other than r in any pairing in the basis. However, for the subproblem, it is computationally advantageous to eliminate as many arcs as possible when a branching decision is fixed. Forbidding the connection is implemented by eliminating the arc connecting r to s. These network modifications can be accomplished by removing the arcs or by giving them a very high cost so that they will not be used in attractive pairings. The second approach is generally preferable since it simplifies many of the bookkeeping issues associated with storing the network structure and enables the same network (with modified costs) to be used to generate pairings at any node in the branch-and-bound tree.

Duty Period Network Modification. If a duty period network is used, the implementation will depend on whether the connection between flights r and s is an overnight rest. A connection within a duty period (not an overnight) can be required by eliminating all duties that contain either flight but do not have them appearing consecutively. The connection can be forbidden by eliminating all duty periods containing r and s consecutively. A connection that is an overnight rest can be required by eliminating all duties containing flight r that do not end in r and all duties containing s that do not begin with s. Then arcs connecting any duty period ending in r to a duty period that does not begin with s are deleted, as well as arcs connecting a duty period ending in a flight other than r to a duty period beginning with s. To forbid the connection, we delete connection arcs from any duty period ending in r to a duty period beginning in s.

From the above discussion, we see that the branch on follow-on rule can be implemented with either type of pairing generation network simply by eliminating arcs from the network.

14.4.5 Parallel Approaches to Crew Pairing

Crew pairing problems with as few as 300 legs for hub-and-spoke networks or 2000 legs for point-to-point networks can take as much as 10 to 20 hours of CPU time to solve even approximately. This is particularly problematic when conducting 'what-if' analysis. One way to decrease computation time is to employ parallel algorithms for crew pairing. However, the crew pairing model (14.1) is an integer program and parallel algorithms for integer programs typically do not scale well. Thus, designing parallel algorithms for the crew pairing problem is a challenging problem that has only recently begun to be studied.

One of the most time intensive parts of most crew pairing algorithms is pairing generation. The basic idea of a parallel algorithm for pairing generation is to distribute the legs originating at crewbases (called starting legs) among the processors, with each processor enumerating all the pairings starting with the assigned starting legs. Since the computational times to generate all the pairings starting with a given leg can vary substantially, load balancing algorithms are needed. Goumopoulos et al., 1997 propose a pulling algorithm based on the master/workers paradigm. The master distributes the legs one by one to the workers. Whenever a worker becomes idle, it queries the master for a new starting leg. Klabjan and Schwan, 2000 eliminate the master with the processors exchanging the workload among themselves.

In other research, Alefragis et al., 1998, Sanders et al., 1999, and Alefragis et al., 2000 focus on parallelizing the pairing enumeration and the Lagrangian decomposition algorithm of Andersson et al., 1998. Since the latter algorithm is inherently sequential, they describe the steps for parallelizing an iteration of the algorithm, i.e. updating the Lagrangian multipliers and computing the subgradient. They compute the constraints in parallel and distribute the variables among the processors. Note that due to the fine grain parallelism, this algorithm does not perform well on architectures with high latency such as a cluster of workstations.

An entirely different approach is given in Klabjan et al., 2001b. The LP relaxation is solved in parallel by generating the pairings in parallel and in each iteration the LP is solved with the parallel primal-dual algorithm, Klabjan et al., 2000. The IP over a small subset of pairings is solved with a branch-and-bound algorithm that executes the strong

branching rule in parallel. The pairing enumeration algorithm is scalable but the parallel primal-dual algorithm scales only to 20 processors.

The most promising algorithms for parallelization are branch-and-price since coarse granularity is easily achievable by evaluating the branch-and-bound nodes in parallel. Gedron and Crainic, 1994 give a survey on parallel branch-and-bound algorithms. Klabjan, 2001 describes a parallel branch-and-price algorithm. The algorithm evaluates the branch-and-bound nodes in parallel and in addition, each node is evaluated in parallel. An LP relaxation is solved in parallel by embedding parallel column generation in the parallel primal-dual algorithm.

14.4.6 Open Issues

There are still a number of open questions regarding the best method for crew pairing optimization. Whether to use dynamic network-based pairing generation or a fast pricing procedure like SPRINT is not clear. If pairings can be enumerated quickly and accessed off-line in an efficient manner, the SPRINT approach may be preferable to network-based generation. If network-based generation is used, the type of network that will perform most efficiently might be highly schedule-dependent. For point-to-point crew pairing problems, duty period based networks have proven to be efficient because the number of possible duty periods grows relatively slowly with the number of possible flights. However, for hub-and-spoke networks, the results have been mixed. Another open issue is how to manage effectively the number of pairings in the constraint matrix; that is, how many pairings to add at each iteration and whether or not to delete pairings with high reduced cost.

14.4.7 Crew Rostering Solution Approaches

The solution approaches described above apply to both the crew pairing and crew assignment problems. We conclude by briefly highlighting some of the research conducted specifically for the crew rostering version of the crew assignment problem. We also introduce an alternative approach to generating schedules that has been used in the crew rostering literature but could also be applied to crew pairing generation.

14.4.7.1 Computational Results. Ryan, 1992 solves crew rostering problems with 55 crew members and 120 pairings. This results in problems containing as many as 300,000 variables, with solution times ranging from less than 10 minutes to 2-3 hours.

Gamache et al., 1998 construct individualized monthly work schedules for pilots and officers. Schedules are selected for assignment to crew

members based on considerations of individual preferences and seniority restrictions. They solve 24 instances of problems at Air Canada, containing up to 108 pilots and 568 pairings. Using cutting planes, solution times range from 1 to 8 hours.

14.4.7.2 A Constraint Programming Approach. In some cases, there exist schedule rules and regulations that cannot easily be captured in a constrained shortest path approach. Moreover, pricing by enumeration may be intractable. Recently *constraint programming* (CP) approaches to rostering have been proposed. Thorough discussions of CP can be found in Lustig and Puget, 2001 and Brailsford et al., 1999.

An example of how constraint programming has been used in crew assignment can be found in the work of Fahle et al., 1999 and Junker et al., 1999. They use CP to solve the pricing problem for generating columns in the crew assignment problem of a major European airline. For each crew, they define a variable that represents the set of duties to be assigned to that crew. The domain contains all feasible subsets of the set of duties to be covered. They are able to specify constraints that capture all of the crew rules and regulations. In addition, they incorporate a shortest path component that leverages dual information from the restricted master. This allows them to reduce the search space significantly. They are able to solve real-world problems successfully, incorporating constraints that could not be captured in a constrained shortest path approach.

14.5 Integrating Crew Pairing with Maintenance Routing and Schedule Design

In airline planning, the schedule design, fleet assignment and maintenance routing problems are all solved before the crew scheduling problem. Their solutions then impact the input to crew pairing. For example, by assigning aircraft types to flights in the fleet assignment problem, we partition the flights into smaller sets and solve a separate crew scheduling problem for each of these sets, since individual crews can only fly certain aircraft types. Thus, solving these planning problems sequentially can lead to sub-optimalities, because decisions are made in the earlier problems without taking into account their impact on crew scheduling. A fully integrated approach to the airline planning process is quite difficult, due to its enormous size and complexity. Nonetheless, benefits can be gained by partially integrating elements of the planning process. We provide examples of this in the sections that follow.

14.5.1 Crew Pairing and Maintenance Routing

Although crew scheduling assigns *crews* and maintenance routing assigns *aircraft*, there is a connection between these two problems. Specifically, this connection deals with the amount of time required between two flights for a connection to be crew-feasible. Recall that there is a minimum idle time required between two consecutive flights in a duty period. This time is needed, in part, so that the crew can move through the terminal from the arrival gate of the first flight to the departure gate of the second flight. However, if both of these flights have been assigned to the same aircraft in the maintenance routing problem, then the crew will remain with the aircraft and therefore this time restriction can be relaxed. Such a crew connection, which is feasible only if both flights share a common aircraft in the maintenance routing solution, is known as a *forced turn*.

When constructing a network for the crew pairing problem, we begin with the set of connections that have adequate sit time and then add to this set those forced turns implied by the maintenance routing solution. These additional connections permit new pairing opportunities and thus can improve the quality of the crew pairing solution. However, because the forced turns are determined in the maintenance routing problem without taking into account the crew pairing objective, solving these two problems sequentially can lead to sub-optimal results.

This potential for sub-optimality is demonstrated in the following example, taken from Cohn and Barnhart, 2002. Consider a network of eight flights, denoted A through H. As shown in Figure 14.5, this network has three potential forced turns, two different feasible solutions to the maintenance routing problem (denoted by MR), and four potential pairings. For each of the maintenance solutions, only a subset of the pairings are feasible, depending on the forced turns implied by the maintenance solution. Thus, given maintenance solution 1, the cost of an optimal crew pairing solution is \$7, while maintenance solution 2 yields a crew pairing problem whose optimal cost is \$5.

This small example highlights how a sequential approach to the maintenance routing and crew pairing problems can lead to sub-optimal results. Three different approaches have appeared in the literature to address this.

In the first approach, Klabjan et al., 2002 solve the crew pairing problem *before* they solve the maintenance routing problem. They include *all* potential forced turns in the crew pairing network. Those forced turns contained in the crew pairing solution then become required aircraft turns when solving the maintenance routing problem. Although this ap-

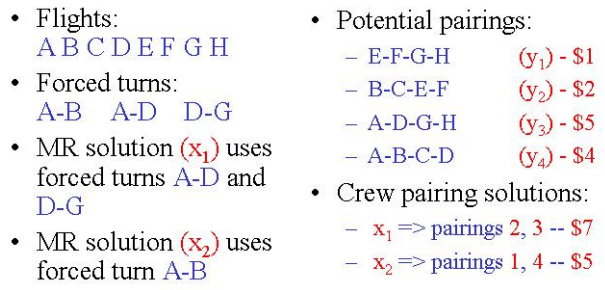

Figure 14.5. Integrated Example

proach can potentially lead to maintenance infeasibility, in practice they found feasible solutions for many hub-and-spoke flight networks. Additionally, note that whenever a feasible crew pairing solution is found to be maintenance feasible, this solution is in fact *optimal* for the integrated problems, when the maintenance routing problem is a feasibility rather than an optimization problem. Furthermore, their approach requires no more computational effort than the original sequential approach.

In the second approach, Cordeau et al., 2001 present an integrated model that guarantees maintenance feasibility. They maintain the original string-based maintenance routing and crew pairing formulations. Like Klabjan et al., 2002, they include all potential forced turns in the crew pairing network. They then link the two models by adding one constraint for each potential forced turn. The constraint for forced turn t states that the number of chosen crew pairings containing forced turn t cannot exceed the number of chosen maintenance routes that contain it. This results in a large-scale integer program which they solve by branch-and-bound, where the LP relaxations are solved by using a Benders decomposition approach and column generation.

A third approach is found in Cohn and Barnhart, 2002. Their approach is similar to that of Cordeau et al., 2001, but in place of maintenance string variables, they use variables representing *complete solutions* to the maintenance routing problem. This dramatically reduces the number of constraints, because all of the original maintenance routing constraints are replaced by a single convexity constraint. This reduction in constraints comes at the cost of a potential explosion in the number of variables. However, they prove that only a small subset of the feasible maintenance solutions need to be considered to ensure an optimal result, thereby greatly reducing the size of the problem. Furthermore, they prove that the integrality of the maintenance solution variables can

be relaxed. Thus, their integrated maintenance routing and crew pairing model has no more binary decision variables than the basic crew pairing model alone.

14.5.2 Crew Pairing and Schedule Planning

Klabjan et al., 2002 study the impact of flight departure times on the crew pairing problem. If we are allowed to change the flight departure times, then some paths in the duty period network that do not satisfy all of the pairing feasibility rules might correspond to a pairing in a retimed flight schedule. Consider two flights i and j depicted in Figure 14.6. In the original schedule, leg j cannot follow leg i in a pairing because it violates the minimum sit connection time. However if leg j departs 5 minutes later, then the connection becomes feasible. In a retimed flight schedule, additional paths in the duty period network become pairings not only due to the minimum sit and rest connection times but also with respect to the maximum duty elapsed time and the 8-in-24 rule.

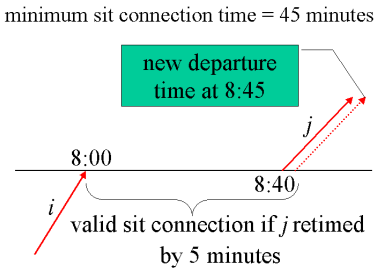

Figure 14.6. Leg Retiming

The model in Klabjan et al., 2002 is identical to (14.1) except that more columns are considered. They develop an algorithm that simultaneously generates paths in the duty period network and new departure times such that the generated paths correspond to pairings in the retimed flight schedule. Each path defines its own flight departure times, but given that in (14.1) every flight segment is covered by exactly one pairing, the solution implies a single departure time for each leg.

They report computational experiments on large fleets of a U.S. domestic carrier. A time window w of either 5 or 10 minutes is imposed, i.e. every departure time can be changed by at most $\pm w$. New flight departure times should not diverge by much from the original times since otherwise it would affect the fleeting cost and would substantially disrupt passenger connections. On average, the improvement of pay-and-credit

for $w = 5$ minutes is 25% and for $w = 10$, pay-and-credit decreases by 35%. These data clearly show that the crew cost can be substantially reduced by slightly retiming departures.

14.5.3 Crew Pairing with Regularity

The crew pairing model (14.1) for the weekly problem minimizes the weekly crew cost. However it is unlikely that the resulting pairings could be repeated many times in the weekly horizon unless such repetition constraints were specifically imposed. Thus the solution would lack regularity. Regularity is important with respect to crew (and aircraft) schedules, since regular solutions are much easier to implement and manage, and, if possible, crews prefer to repeat itineraries.

In Section 14.2.2.1 we described the daily/weekly exceptions methodology for solving the weekly problem. This methodology, first, does not necessarily find the minimum cost crew schedule even if the daily and the weekly exception problems are solved to optimality and second it does not directly take into account regularity. Klabjan et al., 2001a present a new model, called the weekly crew pairing model with regularity, that captures both the crew cost and regularity in a weekly schedule. They solve the model in several stages, where in the first stage they obtain pairings with the highest regularity, i.e. those that can repeat seven days a week, in the second stage the algorithm yields pairings that can be repeated six times in a week, and so forth.

By using approximations and integer programming as a heuristic, they obtain solutions that improve on current practice with respect to both regularity and cost. They report computational results on small and large fleets of a major U.S. domestic carrier. The improvements on crew cost range from 10% to 40% and their solutions have 40-60% higher regularity.

14.6 The Crew Recovery Problem

An airline schedule rarely operates as planned. Maintenance problems, weather conditions, and other unplanned events cause frequent disruptions – on a typical day, several flights will be delayed or canceled. Each disruption can propagate through the system, because it impacts resources such as crews and aircraft that are also needed for subsequent flights. The *crew recovery problem* considers how to modify a crew schedule that has been affected by disruptions.

The recovery problem differs from the planning problem in several ways. One of the most fundamental differences is in the time horizon for solving the problem. Unlike the planning problem, which is solved as

part of a multi-week process, the recovery problem must be solved very quickly – often, in minutes. Thus, the goal of the crew recovery problem is to find a good solution quickly.

The second difference between recovery and planning is that the crew recovery problem must take into account the recent flying history of the active crews. Each crew's options are limited as a function of the work that has already been performed during the current pairing.

Third, *reserve crews* can be considered when solving the crew recovery problem. These crews have a minimum guaranteed pay, measured in flying hours, and cannot fly more than a designated monthly maximum. This pay structure adds a further level of complexity to the problem.

Another difference is in the constraints that determine what constitutes a feasible pairing. Most airlines use tighter restrictions in their scheduling problems than are legally mandated by, for example, tightening the minimum rest connection time and the maximum duty elapsed time. This is precisely so that they will have some added flexibility in recovering from disruptions. In the recovery problem, on the other hand, rules on flying hours are often pushed to their legal limits.

Perhaps the most significant difference between crew pairing and crew recovery is in how the objective function is defined. When a schedule is modified to address disruptions, active crews are usually paid *at least* the cost of their originally scheduled pairings; if a crew is assigned to a modified pairing that has higher cost, they receive the higher amount. It is also desirable to keep down the additional costs incurred by reserve crews. Furthermore, there are other objectives that are important as well, such as returning to the original plan quickly and minimizing passenger disruptions. In addition, crew decisions are not made in isolation. They must be made in conjunction with decisions about delaying or canceling future flights, swapping aircraft, and further issues related to the other resources that have been affected by the disruptions. The recovery problem also faces a host of safety and labor constraints that restrict what changes can be made. Therefore, even deciding what objective to use when recovering from disruption can be a difficult question.

Thus, while crew recovery has much in common with crew pairing, it also poses its own unique set of challenges. Although this problem has been addressed in limited fashion in the literature, much work remains to be done. In the following section, we present one approach to modeling the crew recovery problem, which leverages its similarities with crew pairing. In the subsequent section, we highlight some of the research on solving the crew recovery problem.

14.6.1 A Crew Recovery Model

We present a crew recovery model from Lettovský, 1997 and Lettovský et al., 2000 that is similar to the crew pairing model. However, in this recovery model, each pairing is specific to a particular crew. For a given crew, all potential pairings begin at the next time and location that the crew becomes available, i.e. at the end of their current flight or rest period. Each potential pairing for a crew must not only take into account work already completed in the current pairing when ensuring that all rules and regulations are satisfied, but must also ensure the legality of the crew's remaining schedule. That is, there must be enough time at the completion of this modified pairing for a sufficient rest period before the next scheduled pairing is to begin, and restrictions that span multiple pairings in a schedule (such as monthly flying time limits) must not be violated.

The objective of this model is to minimize the cost of adjusted pairings, reserve crews, and deadheaded crews, as well as the cost of canceling flights. The cancelation cost is the cost of re-assigning passengers to other flights as well as hotel and meal costs for affected passengers and some estimate of the loss of good will.

We define the following parameters:

e	equipment type experiencing disruption (this may represent several aircraft types if they are crew compatible),
L_e	set of flight segments to be covered by crews of equipment type e,
K_e	set of crews available for equipment type e (including reserve crews),
P_k	set of pairings that can be flown by crew $k \in K_e$,
c_p	cost of assigning pairing p,
d_l	cost of using flight segment l for deadheading,
q_k	cost estimate of returning the crew to its domicile,
f_l	cost estimate of canceling flight segment l,
β_{pl}	1 if flight segment l is included in pairing p, 0 otherwise.

The variables are:

$$x_p = \begin{cases} 1 & \text{if pairing } p \text{ is assigned to a crew,} \\ 0 & \text{otherwise,} \end{cases}$$

$$v_p = \begin{cases} 1 & \text{if crew } k \text{ has no pairing assigned,} \\ 0 & \text{otherwise,} \end{cases}$$

$$y_l = \begin{cases} 1 & \text{if flight segment } l \text{ is canceled,} \\ 0 & \text{otherwise,} \end{cases}$$

$$w_l = \text{ the number of crews deadheading on flight segment } l.$$

The airline crew recovery problem for a given equipment type e is

$$\begin{aligned} \min \sum_{k \in K_e} \sum_{p \in P_k} c_p x_p + \sum_{l \in L_e} f_l y_l & \\ + \sum_{l \in L_e} d_l w_l + \sum_{k \in K_e} q_k v_k & \\ \sum_{k \in K_e} \sum_{p \in P_k} \beta_{pl} x_p + y_l - w_l = 1 & \qquad \text{for all } l \in L_e, \\ \sum_{p \in P_k} x_p + v_k = 1 & \qquad \text{for all } k \in K_e, \\ w_l + max_l \cdot y_l \le max_l & \qquad \text{for all } l \in L_e, \\ 0 \le w_l \le max_l & \qquad \text{for all } l \in L_e, \\ x \text{ binary}, y \text{ binary}, w \ge 0, v \ge 0\,. & \end{aligned} \tag{14.3}$$

The first set of constraints guarantees that all flight segments are either canceled or covered at least once. The slack variable on these constraints, w_l, has an upper bound max_l defined as the maximum number of crews that can deadhead on flight segment l. The second set of constraints ensures that crew k is either assigned to a pairing or is deadheaded to its crewbase. The third set of constraints forces w_l to be zero if flight l is canceled. Note that the integrality of the variables w_l and v_k are implied and hence need not be imposed.

When solving the crew recovery problem, it is important to find a good solution quickly. Furthermore, it makes sense to take advantage of the fact that the original scheduled pairings were optimal in the undisrupted problem environment. Therefore, when solving the crew recovery problem, we do not want to re-assign all crews. Instead, we want to consider only those crews whose pairings were disrupted, as well as a small number of additional crews deemed likely to introduce good 'swapping' opportunities. Limiting the scope of the problem in this way can significantly reduce the size of the model and thus improve its tractability. Heuristics for selecting the set of crews to be considered can be found in Lettovský et al., 2000 and Lettovský, 1997.

14.6.2 Crew Recovery Solution Approaches

Very little has been published in the open literature on solving the crew recovery problem. Teodorovic and Stojkovic, 1990 developed a sequential approach based on a dynamic programming algorithm, using the first-in-first-out principle to minimize the crews' ground time. Wei and Yu, 1997 presented a heuristic-based framework for real-time crew re-scheduling. Song et al., 1998 presented a multicommodity integer network flow model and a heuristic search algorithm to solve it. Stojkovic et al., 1998 presented a column generation approach similar to that used for crew pairing problems. Solutions approaches to the crew recovery model presented in the previous section can be found in Lettovský et al., 2000 and Lettovský, 1997.

Models such as (14.3) can be solved to obtain optimal solutions for small disruptions and good feasible solutions for medium sized disruptions. However, for major disruptions such as those caused by snowstorms, and particularly those disruptions that affect multiple airports, further refinement and perhaps even a new approach altogether is needed.

14.6.3 Crew Rostering Recovery

We conclude this section by noting that one potential for inefficiency in the crew recovery approach discussed above is that it limits changes to the pairings currently underway. By requiring the modified pairings to be feasible in conjunction with the remainder of the crew's monthly schedule, opportunities may be missed. On the other hand, considering the entire schedule, rather than just the current pairings, yields an enormous problem. Stojkovic et al., 1998 present an initial approach to this challenging task.

14.7 Robustness in Crew Pairing

The crew pairing problem is solved well before the flight schedule becomes operational. In this planning stage, all flights are assumed to have departure times that are both fixed and known. This assumption is often proven wrong when the crew schedule is actually implemented. For example, the U.S. Department of Transportation reported that the total number of *delay minutes* in the system (based on flight delays of 15 minutes or more) had increased by 11% from 1995 to 1999, Bond, 2000. In the summer of 2000, airline delays received national attention in the U.S., when the airline with the *best* performance record had 25% of its flights delayed by 15 minutes or more.

When crew members' schedules are disrupted in operations, they are nonetheless guaranteed to be paid for their original scheduled workload. In addition, if delays increase their flying or sit time, they may be entitled to added compensation. Furthermore, disruptions may require the use of reserve crews to get back on schedule. Clearly, then, the cost associated with implementing a crew pairing solution may vary significantly from the planned cost. Typically, the planned ratio of pay-and-credit to flying time is below 1% for large fleets, but increases on average to 4% when the schedule is implemented. For smaller fleets, the ratio tends to increase from about 3% to 8%. Solutions of large fleets are much more sensitive to disruptions since they have many tight connections. A disrupted short connection can have a significant impact on the entire flight and crew schedule due to the snowball effects. Such increases in planned cost can translate to millions of dollars in unplanned crew costs. There are two ways that carriers can try to minimize these unplanned costs. The first, discussed in Section 6, is to improve the quality of their recovery procedures. The second is to focus in the planning stage on developing more robust schedules – that is, to minimize the expected operational cost of a schedule rather than its planned cost.

14.7.1 Evaluating Crew Schedules

Robustness is not well-defined, in fact, comparing two different schedules to determine which one is more robust can be quite difficult. In general, comparisons are done by using a simulation to approximate the operating cost of a given schedule for a particular time period (typically, one month). Clearly, such a simulation should reflect the airline operations as closely as possible.

Simulations of partial airline operations, for example, aircraft ground movement and passenger flow, have been developed, see Yu, 1998. Only recently have simulations of integrated airline operations been designed.

Kornecki and Vargas, 2000, for example, developed a simulation designed for employee training. Rosenberger et al., 2000 created `SimAir`, a simulation that takes into account most airline operations and has built-in recovery modules. It keeps track of several types of resources, including aircraft, crews, and passengers, and produces a number of statistics such as crew costs and block times. Finally, Schaefer et al., 2000 use a simulation-based approach to design more robust crew schedules.

14.7.2 Models for Robust Airline Crew Pairing

Here we present three approaches for finding robust crew pairings.

14.7.2.1 Expected Pairing Cost Approach. Schaefer et al., 2000 solve a problem very similar to the crew pairing problem (14.1). However, they replace the objective coefficients in this model with $\bar{c}_p$, which they define to be the *expected cost of pairing p*. They then solve this model with the same methodologies as presented in Section 14.4.

Of course, the difficult aspect of this problem is in computing the cost coefficients $\bar{c}_p$, given that the expected cost of a pairing depends in part on the other pairings in the crew schedule. For every pairing they compute the expected cost by running `Simair` under the assumption that the expected cost is independent of the other pairings in a crew schedule. They show that this assumption holds under the push-back recovery procedure. Push-back recovery delays the flights until all of the resources are available.

Once they have computed cost coefficients they solve this modified version of the crew pairing problem and then use `SimAir` to evaluate the quality of their solutions. They report some interesting findings. For example, they observe that it may be preferable to have some pairings in which costs are determined by TAFB or the minimum guarantee pay, rather than flying cost. In addition, they find crew schedules to be more robust when the pay-and-credit of the pairings has low variance and there are not many pairings with zero pay-and-credit. This is intuitive because zero pay-and-credit pairings have minimal connection time and are therefore vulnerable to disruptions.

Schaefer et al., 2000 compared this expected cost approach with a penalty approach that includes penalties in the cost function for such factors as tight connections and elapsed times, and tight 8-in-24 constraints. Better results were obtained by the expected cost model.

14.7.2.2 Maximizing the Connection Time. Ehrgott and Ryan, 2001 and Yen and Birge, 2000 measure robustness as the excess

sit connection time above the minimum sit connection time. If k is a sit connection and t is the connection time, they define a penalty by $w_k(minSit - t)$, where w_k is the penalty factor and $minSit$ is the minimum required sit connection time. They define the *robustness cost of a pairing* as the sum of the penalties over all sit connections in the pairing, excluding the sit connections corresponding to the aircraft turns. Their models find a crew schedule that minimizes the robustness cost.

Yen and Birge, 2000 solve the resulting model as a stochastic integer programming model by assuming that t is a random variable. Given a crew schedule, the recourse problem is a large-scale LP. They develop a heuristic based on follow-on branching for solving the model. They sample 100 disruption scenarios and they show the computational results on a problem with 3000 pairings and 50 legs. Their crew schedules tend to have more sit connections corresponding to the aircraft turns and longer connection times. Ehrgott and Ryan, 2001 assume that the connection time t is deterministic and it is taken with respect to the planned flight schedule. They give computational result on fleets from Air New Zealand.

14.7.2.3 The Crew Pairing Model with Move-up Crews.

When a crew is delayed or has reached a limit on its flying time for a duty or pairing, it would be highly desirable to have an alternative crew available with which it could swap one or more flights. The *crew pairing model with move-up crews*, presented in Klabjan et al., 2001c and Chebalov and Klabjan, 2002, relies on a recovery procedure that uses crew swaps. In addition to the traditional objective of minimizing pairing costs, they introduce a new objective of maximizing the number of opportunities for crew swapping. Thus, their model is a bicriteria optimization model.

A *move-up crew* for a given flight i is a crew that is on the ground for at least the minimum required connection time, originates at the same crew base as the crew covering i, and the two involved crews finish their respective pairings on the same day. If two crews can be swapped in operations, then one crew is a move-up crew. The crew pairing model with move-up crews maximizes the overall number of move-up crews and is solved by a Lagrangian decomposition approach. Computational results show that there are crew schedules with only a slightly higher crew cost but 5 to 10 times more move-up crews than the crew schedules obtained by solving (14.1). Moreover this approach, which attempts to provide protection against uncertainty rather than modeling uncertainty, can be combined with stochastic models that minimize expected cost and/or incorporate penalties.

14.8 Future Directions

Airline crew scheduling has been one of the great successes of operations research, with decision support software installed at all major airlines. Whereas a decade ago solutions to daily problems were typically 10-15% above the lower bound of flying cost, solutions are now typically within at most 1-2% of the lower bound. This improvement in solution quality translates to savings on the order of $50 million annually for a large airline.

Nonetheless, airline crew scheduling is still an active research area with many unsolved problems. We have discussed some recent work on recovery and robust planning in Sections 14.6 and 14.7, but this is clearly just the 'tip of the iceberg'.

Benefits can be gained by developing more efficient schedules for cabin crews. This problem has received less attention than cockpit crew scheduling, both because cabin crews are significantly less costly and also because it is a much larger problem.

Finally, and perhaps most challenging, is the integration of the crew pairing, fleet assignment, and schedule planning problems, especially since these problems are difficult to solve individually.

References

Alefragis, P., Goumopoulos, C., Housos, E., Sanders, P., Takkula, T., and Wedelin, D. (1998). Parallel crew scheduling in PAROS. In *Proceedings of 1998 Europar*, pages 1104–1113.

Alefragis, P., Sanders, P., Takkula, T., and Wedelin, D. (2000). Parallel integer optimization for crew scheduling. *Annals of Operations Research*, 99:141–166.

Andersson, E., Housos, E., Kohl, N., and Wedelin, D. (1998). Crew pairing optimization. In Yu, G., editor, *Operations Research in the Airline Industry*, pages 228–258. Kluwer Academic Publishers.

Anbil, R., Barnhart, C., Johnson, E., and Hatay, L. (1994). A column generation technique for the long-haul crew assignment problem. In Ciriani, T. and Leachman, R., editors, *Optimization in Industry II*, pages 7–24. John Wiley & Sons.

Anbil, R., Forrest, J., and Pulleyblank, W. (1998). Column generation and the airline crew pairing problem. In *Extra Volume Proceedings ICM*. Available from `http://www.math.uiuc.edu/documenta/xvol-icm/17/17.html`.

Anbil, R., Gelman, E., Patty, B., and Tanga, R. (1991). Recent advances in crew pairing optimization at American Airlines. *Interfaces*, 21:62–74.

Anbil, R., Johnson, E., and Tanga, R. (1992). A global approach to crew pairing optimization. *IBM Systems Journal*, 31:71–78.

Barahona, F. and Anbil, R. (1998). The volume algorithm: Producing primal solutions with a subgradient method. Technical Report RC-21103, T. J. Watson Research Center.

Barahona, F. and Anbil, R. (1999). On some difficult linear programs coming from set partitioning. Technical Report RC-21410, T. J. Watson Research Center.

Barnhart, C., Hatay, L., and Johnson, E. (1995). Deadhead selection for the long-haul crew pairing problem. *Operations Research*, 43:491–499.

Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M., and Vance, P. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329.

Bixby, R., Cook, W., Cox, A., and Lee, E. (1995). Parallel mixed integer programming. Technical Report CRPC-TR95554, Rice University. Available from `ftp://softlib.rice.edu/pub/CRPC-TRs/reports`.

Bixby, R., Gregory, J., Lustig, I., Marsten, R., and Shanno, D. (1992). Very large-scale linear programming: A case study in combining interior point and simplex methods. *Operations Research*, 40:885–897.

Bond, D. (2000). Commercial aviation on the ropes. *Aviation Week & Space Technology*. September issue.

Brailsford, S., Potts, C., and Smith, B. (1999). Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research*, 119:557 – 581.

Caprara, A., Toth, P., Vigo, D., and Fischetti, M. (1998). Modeling and solving the crew rostering problem. *Operations Research*, 46:820–830.

Chebalov, S. and Klabjan, D. (2002). Robust airline crew scheduling: Move-up crews. In *Proceedings of the 2002 NSF Design, Service, and Manufacturing Grantees Research Conference*.

Cohn, A. and Barnhart, C. (2002). Improving crew scheduling by incorporating key maintenance routing decisions. *Operations Research*. To appear.

Cordeau, J., Stojković, G., Soumis, F., and Desrosiers, J. (2001). Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35:375–388.

Day, P. and Ryan, D. (1997). Flight attendant rostering for short-haul airline operations. *Operations Research*, 45:649–661.

Desaulniers, G., Desrosiers, J., Ioachim, I., Solomon, M., and Soumis, F. (1998). A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In Crainic, T. and Laporte, G., editors, *Fleet Management and Logistics*, pages 57–93. Kluwer Publishing Company.

Desrochers, M. and Soumis, F. (1988). A generalized permanent labeling algorithm for the shortest path problem with time windows. *INFOR*, 26:191–212.

Desrochers, M. and Soumis, F. (1989). A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23:1–13.

Desrosiers, J., Dumas, Y., Desrochers, M., Soumis, F., Sanso, B., and Trudeau, P. (1991). A breakthrough in airline crew scheduling. Technical Report G-91-11, Cahiers du GERAD.

du Merle, O., Villeneuve, D., Desrosiers, J., and Hanses, P. (1999). Stabilized column generation. *Discrete Mathematics*, 194:229–237.

Ehrgott, M. and Ryan, D. (2001). Bicriteria robustness versus cost optimization in tour of duty planning at Air New Zealand. Technical report, Univeristy of Auckland.

Fahle, T., Junker, V., Karish, S., Kohl, N., and Vaaben, B. (1999). Constraint programming based column generation for crew assignment. *Journal of Heuristics*. To appear.

Forrest, J. and Goldfarb, D. (1992). Steepest-edge simplex algorithms for linear programming. *Mathematical Programming*, 57:341–374.

Gamache, M. and Soumis, F. (1998). A method for optimally solving the rostering problem. In Yu, G., editor, *Operations Research in the Airline Industry*, pages 124–157. Kluwer Academic Publishers.

Gamache, M., Soumis, F., Marquis, G., and Desrosiers, J. (1999). A column generation approach for large scale aircrew rostering problems. *Operations Research*, 47:247–262.

Gamache, M., Soumis, F., Villeneuve, D., Desrosiers, J., and Gelinas, E. (1998). The preferential bidding system at Air Canada. *Transportation Science*, 32:246–255.

Gedron, B. and Crainic, T. (1994). Parallel branch-and-bound algorithms: Survey and synthesis. *Operations Research*, 42:1042–1066.

Gershkoff, I. (1989). Optimizing flight crew schedules. *Interfaces*, 19:29–43.

Goumopoulos, C., Housos, E., and Liljenzin, O. (1997). Parallel crew scheduling on workstation networks using PVM. In *Proceedings of 1997 EuroPVM-MPI*, volume 1332.

Hoffman, K. and Padberg, M. (1993). Solving airline crew scheduling problems by branch-and-cut. *Management Science*, 39:657–682.

Hu, J. and Johnson, E. (1999). Computational results with a primal-dual subproblem simplex method. *Operations Research Letters*, 25:149–158.

Junker, U., Karisch, S., Kohl, N., Vaaben, B., Fahle, T., and Sellmann, M. (1999). A framework for constraint programming based column generation. In *Proceedings of CP 1999*, pages 261–274.

Klabjan, D. (2001). Next generation airline crew scheduling. Technical report, University of Illinois at Urbana-Champaign. Available from `http://www.staff.uiuc.edu/~klabjan/reports/ngCS.pdf`.

Klabjan, D., Johnson, E., and Nemhauser, G. (2000). A parallel primal-dual algorithm. *Operations Research Letters*, 27:47–55.

Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E., and Ramaswamy, S. (2001a). Airline crew scheduling with regularity. *Transportation Science*, 35:359–374.

Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E., and Ramaswamy, S. (2001b). Solving large airline crew scheduling problems: Random pairing generation and strong branching. *Computational Optimization and Applications*, 20:73–91.

Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E., and Ramaswamy, S. (2002). Airline crew scheduling with time windows and plane count constraints. *Transportation Science*, 36:337–348.

Klabjan, D., Schaefer, A., Johnson, E., Kleywegt, A., and Nemhauser, G. (2001c). Robust airline crew scheduling. In *Proceedings of TRISTAN IV*, pages 275–280.

Klabjan, D. and Schwan, K. (2000). Airline crew pairing generation in parallel. In *Proceedings of the Tenth SIAM Conference on Parallel Processing for Scientific Computing*.

Kornecki, A. and Vargas, D. (2000). Simulation-based training for airline controller operations. In *Proceedings of Society of Computer Simulation 2000 Advanced Simulation Technologies Conference*, pages 162–171.

Kwok, L. and Wu, L. (1996). Development of an expert system in cabin crew pattern generation. *International Journal of Expert Systems*, 9:445–464.

Lavoie, S., Minoux, M., and Odier, E. (1988). A new approach for crew pairing problems by column generation with an application to air transportation. *European Journal of Operational Research*, 35:45–58.

Lettovský, L. (1997). *Airline Operations Recovery: An Optimization Approach*. PhD thesis, Georgia Institute of Technology.

Lettovský, L., Johnson, E., and Nemhauser, G. (2000). Airline crew recovery. *Transportation Science*, 34:337–348.

Linderoth, J. and Savelsbergh, M. (1999). A computational study of search strategies for mixed integer programming. *INFORMS Journal on Computing*, 11:173–187.

Lustig, I. and Puget, J. (2001). Program $\neq$ program: constraint programming and its relationship to mathematical programming. *Interfaces*. To appear.

Makri, A. and Klabjan, D. (2001). Efficient column generation techniques for airline crew scheduling. Technical report, University of Illinois at Urbana-Champaign. Available from `http://www.staff.uiuc.edu/~klabjan/professional.html`.

Marsten, R. (1994). Crew planning at Delta Airlines. XV Mathematical Programming Symposium. Presentation.

Minoux, M. (1984). Column generation techniques in combinatorial optimization: a new application to crew pairing problems. In *Proceedings XXIVth AGIFORS Symposium*.

Rosenberger, J., Schaefer, A., Goldsman, D., Johnson, E., Kleywegt, A., and Nemhauser, G. (2000). A stochastic model of airline operations. *Transportation Science*. To appear. Available from `http://tli.isye.gatech.edu`.

Ryan, D. (1992). The solution of massive generalized set partitioning problems in air crew rostering. *Journal of the Operational Research Society*, 43:459–467.

Ryan, D. and Foster, B. (1981). An integer programming approach to scheduling. In Wren, A., editor, *Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling*, pages 269–280. Elsevier Science B.V.

Sanders, P., Takkula, T., and Wedelin, D. (1999). High performance integer optimization for crew scheduling. In *Proceedings of the HPCS '99*. Available from `http://www.cs.chalmers.se/~tuomo`.

Schaefer, A., Johnson, E., Kleywegt, A., and Nemhauser, G. (2000). Airline crew scheduling under uncertainty. Technical Report TLI-01-01, Georgia Institute of Technology.

Song, M., Wei, G., and Yu, G. (1998). A decision support framework for crew management during airline irregular operations. In Yu, G., editor, *Operations Research in the Airline Industry*, pages 260 – 286. Kluwer Academic Publishers.

Stojkovic, G., Soumis, M., and Desrosiers, J. (1998). The operational airline crew scheduling problem. *Transportation Science*, 32:232–245.

Teodorovic, D. and Stojkovic, G. (1990). Model for operational daily airline scheduling. *Transportation Planning Technology*, 14:273–285.

Vance, P., Atamtürk, A., Barnhart, C., Gelman, E., Johnson, E., Krishna, A., Mahidhara, D., Nemhauser, G., and Rebello, R. (1997a). A heuristic branch-and-price approach for the airline crew pairing problem. Technical Report LEC-97-06, Georgia Institute of Technology.

Vance, P., Barnhart, C., Johnson, E., and Nemhauser, G. (1997b). Airline crew scheduling: A new formulation and decomposition algorithm. *Operations Research*, 45:188–200.

Wedelin, D. (1995). An algorithm for large scale 0-1 integer programming with applications to airline crew scheduling. *Annals of Operations Research*, 57:283–301.

Wei, G. and Yu, G. (1997). Optimization model and algorithm for crew management during airline irregular operations. *Journal of Combinatorial Optimization*, 1:305–321.

Wilson, N., editor (1999). *Computer-Aided Transit Scheduling*. Springer Verlag.

Yen, J. and Birge, J. (2000). A stochastic programming approach to the airline crew scheduling problem. Technical report, University of Washington.

Yu, G., editor (1998). *Operations Research in the Airline Industry.* Kluwer Academic Publishers.