Large Scale Crew Rostering

Hamid Kharraziha, Marek Ozana, and Sami Spjuth

Carmen Systems AB, Odinsgatan 9, SE-411 03 Göteborg, Sweden.
hamid.kharraziha@carmensystems.com

Abstract. Carmen Systems' crew rostering product is currently used by several major European airline and railway companies. In the largest problem that is solved today, around 7,000 crew members are assigned roughly 22,000 tasks. Production quality rosters are produced within 14 hours of optimisation time. In this article, we will describe the sequence of algorithms that are applied and the way these have been modified and extended in order to cope with the largest problems.

1 Introduction

The cost associated with crew is a large portion of the costs of managing transportation companies. Several of the major transportation companies are nowadays using advanced optimisation methods to manage crew costs and other aspects of crew scheduling such as quality of life.

Crew planning is usually separated into a crew pairing and a crew rostering phase. A pairing is a sequence of flight duties which start from a home-base and, within a few days, return back. Cost effective pairings are usually pairings that cover the scheduled flights in the least number of days. In the rostering phase, these paring are assigned to named crew members. In this case one takes into account quality of life aspects as well as the objective to cover all pairings.

Carmen Systems AB is a developer and vendor of resource optimisation software. Among the customers are major airlines such as Air France, British Airways, Iberia, KLM, Lufthansa, Northwest Airlines, SAS and Singapore Airlines and the rail companies Swedish Railways (SJ and Green Cargo) and Deutsche Bahn. We will here describe benchmarks that were performed at two major European airlines. In particular, we will describe the recent changes that were made to the optimiser in order to solve these problems.

For a review of the crew rostering problem in general, and Carmen Systems' rostering optimiser in particular, see [1]. In the literature there are several references describing practical aspects of crew rostering, see e.g. [2], [3], [4], [5] and [6]. For descriptions of the pairing problem, see e.g. [7], [8].

The outline of the article is as follows: First we will describe the rostering problem. Then we will describe the main optimisation methods that we use to solve it and the recent modifications. Finally, we will describe the benchmark results and summarise.

2 The Problem

In a rostering optimisation run the work schedules (rosters) for a number of crew members are determined. The input is rosters with the history from previous months together with some preassigned activities in the planning period. These locked activities may be vacation, medical checks or pairings that are assigned and locked by the planner. The input also consists of a list of assignable pairings. Normally the pairings can be assigned in several positions e.g. Captain and First officer in cockpit problems. It may also be that there is a need of several crew members in a position.

The main objective of the optimisation run is to cover all production (assign all parings) in rosters that satisfy employment contract regulations. In addition to covering production one tries to create rosters that provide convenient work patterns, to make sure parings and free-days are fairly distributed, to make sure the rosters are stable against disruptions and also to meet personal crew-requests (bid satisfaction).

The rules and costs that apply to any single roster are called horizontal rules and costs. (This is because a roster in a Gantt-chart is a horizontal row. see fig 1). Common rules and optimisation costs examples are found in table 1.

Table 1. Examples of horizontal rules and costs

Type	Example
Rule	Less than 80 hrs flight time during the month
Cost	Quadratic penalty for deviation from 60 hrs flight time
Rule	Maximum six days of consecutive work
Cost	Large penalty for not assigning a paring
Rule	Minimum rest time
Cost	Penalty for short rest time
Cost	Penalty for early start after days off
Cost	Penalty for the assignment of a pairing that creates a conflict with time-off bids

In addition to the horizontal rules and costs there are a number of vertical constraints, for example the crew rostering constraint that a pairing must be assigned in all positions. Further, there might be other requirements that apply to all assignments to a pairing e.g. that at most one crew member may be inexperienced, or for cabin crew, that at least one is Spanish speaking. There might also be vertical constraints that apply to the whole problem, so

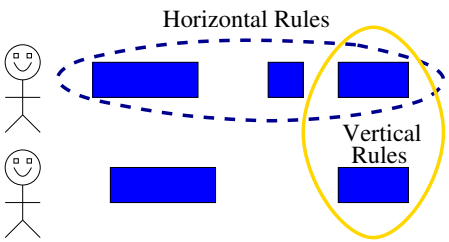

Fig. 1. Horizontal rules: rules which apply to a single roster, e.g. the total flying time has to be smaller than some limit. Vertical rules: rules which concern more than one roster, e.g. at most one inexperienced on a pairing.

called global constraints. A global constraint can for example be used to add an extra penalty in case the number of unassigned pairings on a single day is above a certain limit.

The detailed rules and costs differ to a large extent between different clients. Carmen's optimisers provide a high degree of customisibility via a rule modelling language called Carmen Rave. (For a detailed description of Rave and its use in crew rostering optimisation see [1].) Clients can code and maintain all the different aspects of the planning problem such as rules, costs, and quality aspects, in Rave.

2.1 The IP formulation

The rostering optimiser relies to a large extent on a generate-and-optimise method. A number of legal rosters are generated for each crew member. These rosters become columns in a set partitioning type integer programming problem. Each entry in the a column corresponds to an assignable activity and is set to 1 or 0 depending on wether the roster covers that activity. The best solution to the sub-problem is found by solving the IP using a commercial solver. The IP formulation takes into account all the vertical (in the Gantt-chart) rules and costs. As a consequence vertical rules and costs must be definable as linear constraints while roster-level rules and costs may have a very general form.

The exact definition of the IP-formulation for a crew rostering formulation can be found in [1]. Here we just visualise it with a small example:

Example 1. IP-formulation: The first two columns are rosters for Captain A, the next two are rosters for First Officer B. The last column is a slack column in case Pairing 137 is not assigned in Captain position. The slack column carries the unassigned penalty for this pairing. There is an additional vertical constraint which allows only one inexperienced on Pairing 137. In this case both are inexperienced so they cannot fly the same pairings.

1	1	0	0	0	$= 1$	Exactly one roster for Captain A
0	0	1	1	0	$= 1$	Exactly one roster for First Officer B
1	0	0	0	1	$= 1$	Pairing 137 must be assigned one Captain
0	0	1	0	0	$= 1$	Pairing 137 must be assigned one First Officer
1	0	1	0	0	< 2	There must be at most one inexperienced pilot on Pairing 137

3 The Optimisation Methods

We will here give a short description of the optimisation methods in Carmen Systems' rostering optimiser, with the emphasis on recent development. For a review, please refer to [1].

From an optimisation point of view, already the input problem is a feasible solution. However, it is not a good solution, because of a large number of unassigned pairings giving a huge penalty term in the cost function.

The first optimisation method that it applied is an initial method. The objective is to quickly as possible bring the solution to a stage which is close to a good solution. Close means in this case that each roster has a reasonable number of pairings assigned to it and pairings that are better suited for certain crew members, e.g. with respect to quality of life considerations or special qualifications, are assigned to them. The initial method repeatedly tries to assign pairings, in a greedy fashion, without any backtracking. The stopping criterion is that no more pairings can be assigned.

In the main phase of the optimisation run, the solution from the initial method is iteratively improved until satisfactory solution quality is reached. This is done via a generate-and-optimise method. In each iteration, a sub-problem is selected and solved - which usually results in a sub-optimal solution to the main problem. However, the previous solution is guaranteed to be included and the solution quality can only improve. The problem reduction is done by temporary locking some of the pairings, and by allowing only preferable connections between two subsequent pairings.

The preferred connections are found by sorting all feasible connections according to a user defined criterion or cost, e.g. the connection time. Then only the first few connections with the best costs are allowed. This connection cost can optionally be modified with a reward for good crew-pairing matches or in case the pairing is considered hard to assign e.g. longer pairings.

In sub-section 3.2 we will describe two different locking schemes that have been tested for the sub-problem selection.

The rostering optimiser allows for a wide range of problem formulations. One consequence of this is that the optimiser itself is highly customisable. The selection of the best crew-pairing matches in the initial method, as well

as the sub-problem selection in the improvement method, can be customised separately for each type of problem formulation.

On a fast PC, the solution time ranges from below one hour for a problem with a few hundred crew members, up to several hours for the largest problems with several thousand crew members. The initial method normally takes roughly 10% of the cpu-time and the remaining is used in the improvement phase. The evaluation of roster legality and cost constitutes a significant part of an optimisation run.

In the following two sub-sections we will describe the modifications that were useful for the large benchmark problems.

3.1 From assignment-greedy to cost-greedy

As mentioned before, the initial method performs repeated assignment of the pairings, without any backtracking until no more pairings can be assigned.

Originally this has been done by repeatedly solving a weighted matching problem. In each iteration, a bipartite graph is set up with one edge for each possible legal assignment, between a remaining pairing and a crew member. Each edge has a cost. This cost reflects how well the pairing fits into the crew member's current roster, for example if the connection time to the previous pairings is good, or whether the pairing is particularly good for a specific crew member. After the setup of the graph, a matching is performed were the number of assigned pairings is maximised, with minimal cost. The solution assignments are then performed and edges are set up again for a new matching problem. This is repeated until no more feasible matches can be found.

This method turns out to be effective in assigning as many pairings as possible. On the other hand, it often gives low quality rosters because the number of possible assignments is prioritised against the minimisation of the assignment cost.

Another possibility is to reverse the priorities. This can be done by setting up the edges in the same way as above, but to only do the assignment which has the best cost. Then, only the edges for the changed roster are modified and again the overall best assignment is done. As before, this is repeated until no more assignments are possible.

The cost-greedy initial method showed significant overall improvements for the benchmark 1 described in section 4. Due to a large unassigned penalty, the cost-greedy method produced an initial solution with a worse cost, but still, this solution turned out to be a better starting point and it eventually led to a better final solution.

3.2 Time-window locking vs. random locking

We will now describe two different ways of locking pairings to crew members to obtain sub-problems: the time-window approach and the random locking approach.

In the time-window approach a sub-part of the planning period is selected and pairings that are outside of the time-window remain locked to the roster they were assigned to in the previous solution (See fig 2). Typically the time-window length is one week and it is shifted two days in between the iterations. This method has the advantage that it allows for a wide exploration of all different patterns of packing the pairings inside the time window. The disadvantage is that in the rostering problem, there are important rules and costs which apply to the whole planning period. Good examples are the total flight-time limit and various fairness aspects. Using a time-window of a shorter range may easily lead to the situation that the solutions of the sub-problems are trapped in local minima.

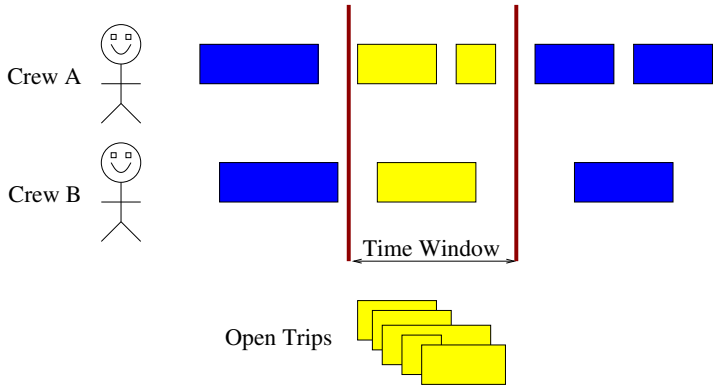

Fig. 2. Only pairings inside the time-window may be de-assigned from a roster and re-assigned to other rosters.

In the random locking scheme the time-window is opened up to the whole planning period. The sub-selection is instead made by randomly locking some, for example 90%, of the pairings. This limits the possibilities on a shorter range, but it allows for better adaption to the long range rules and costs. In a typical run, the best approach may be to do some of the sub-selections with a time-window and some of the sub-selections with random locking in order to allow for improvements on a short as well as long time-scale. For the large benchmarks the random locking was crucial to be able to efficiently cover production.

4 Benchmark Results

We will here present results from two benchmarks that have been performed at two major European airlines. In both cases the client's requirements on solution quality and time were met.

4.1 Benchmark 1

In the first problem the crew consisted of 7,000 members. The rosters were subject to 29 industrial rules. The cost-greedy method was used to create the initial solution. In the improvement phase, the crew were split into two independent groups separated by their role (flight attendant and purser). The random locking mechanism was used in the improvement phase.

	group 1	group 2
Crew members	4669	2331
Assignable tasks	11197	6009
Initial method CPU-time	3:14	
Initial solution cost	1,198,022	3,296,131
Initial assignment percentage	99	99
Improvement phase CPU-time	9:03	2:58
Final solution cost	-112,099	-109,174
Final assignment percentage	100	100

Table 2. Summary of the results for the first benchmark. Both crew groups were assigned simultaneously in the initial phase. In the improvement phase they were run separately. The final cost of rosters is negative due to awarded crew-requests. The times correspond to the CPU time for a 2.4GHz Intel Pentium Linux machine. Assignment percentages are rounded down.

4.2 Benchmark 2

Here, the goal was to assign 22,457 tasks to 6,353 crew members. The solution had to satisfy 22 industrial rules, which cover over 95% of the airline production rules.

In this case the cost-greedy initial method did not prove so efficient, but the random locking mechanism was widely used in the improvement phase. The result are shown in table 3.

5 Summary

Carmen's crew rostering optimiser can efficiently solve problems with up to 7,000 crew members and around 22,000 assignable tasks. It provides a set of customisable heuristic methods which can be tailored for each type of problem characteristics. In order to handle the largest problems, the optimiser toolbox has recently been enlarged with new heuristic methods.

Crew members	6,353
Assignable tasks	22,457
Initial method CPU-time	1:12
Initial solution cost	135,784
Initial assignment percentage	97
Improvement phase CPU-time	12:41
Final solution cost	29,181
Final assignment percentage	99

Table 3. Summary of the results for the second benchmarks. the times correspond to the CPU time for a 2.4GHz Intel Pentium Linux machine. Assignment percentages are rounded down.

Acknowledgements

We would like to thank our colleagues Geoff Bache, Sergio Lupi and Nidhi Sawhney who have contributed to the recent development of Carmen Systems' crew rostering optimiser.

References

1. N. Kohl, and S.E. Karisch, "Airline crew costering: Problem types, modeling, and optimization". Accepted by *Annals of Operations Research*.
2. P.R. Day and D.M. Ryan, "Flight attendant rostering for short-haul airline operations". *Operations Research*, **45**(5):649-661 (1997).
3. F. Gamache, F. Soumis, D. Villeneuve, J. Desrosiers, and E. Gélinas, "The preferential bidding system at Air Canada". *Transportation Science* **32**(3), pages 246-255, (1998).
4. R. Freling, R. Lentink, and A. Wagelmans. "A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm". In S. Voß and J.R. Daduna, editors, *Computer Aided Scheduling of Public Transport*, pages 73-90. Springer-Verlag. Lecture Notes in Economics and Mathematical Systems, (2001).
5. P. Lučić and D. Teodorović, "Simulated annealing for the multi-objextiva aircrew rostering problem", *Transportation Research Part A* **33**:19-45 (1999) .
6. A. Caprara, P. Toth, M. Fischetti, and D. Vigo. "Modeling and solving the crew rostering problem". *Operations Research*, **46**:820-830 (1998).
7. E. Andersson, E. Housos, N. Kohl and D. Wedelin, "Crew pairing optimization" [In G. Yu, editor, *Operations Research in the Airline Industry*, pages 228-252. Kluwer Academic Publishers, Norwell, MA] (1998).
8. S. Lavoie, M. Minoux, and E. Odier. "A new approach of crew pairing problems by column generation and application to air transport", *European Journal of Operations Research*, **35**:45-58 (1998).