Optimizing airline crew scheduling using biased randomization: A case study

A. Agustín[1], A. Gruler[2], J. de Armas[2], and A. A. Juan[2]

Public University of Navarra[1], Pamplona, Spain
Open University of Catalonia[2], Barcelona, Spain
{albamaria.agustin}@unavarra.edu
{agruler,jde_armasa,ajuanp}@uoc.edu

Abstract. Various complex decision making problems are related to airline planning. In the competitive airline industry, efficient crew scheduling is hereby of major practical importance. This paper presents a metaheuristic approach based on biased randomization to tackle the challenging Crew Pairing Problem (CPP). The objective of the CPP is the establishment of flight pairings allowing for cost minimizing crew-flight assignments. Experiments are done using a real-life case with different constraints. The results show that our easy-to-use and fast algorithm reduces overall crew flying times and the necessary number of accompanying crews compared to the pairings currently applied by the company.

Keywords: Biased Randomization, Airline Planning, Metaheuristics, Crew Pairing Problem, Crew Scheduling

1 Introduction

The airline industry is highly competitive, leading to constant pressure on commercial airlines to reduce costs wherever possible. After fuel, flight crew expenses are hereby the second highest source of costs. Thus, efficient crew scheduling and the resulting operational challenges are of major practical importance [6].

Airline planning consists of a number of consecutive steps, each related to different operational challenges: *(i)* the schedule design, during which the company decides which airports to serve and the corresponding connections and their frequencies, *(ii)* the fleet assignment problem in which the aircraft type for each connection is defined, *(iii)* aircraft maintenance routing to ensure an efficient flight schedule considering different maintenance requirements, and *(iv)* crew scheduling [5]. Optimization problems concerning the efficient scheduling of cockpit and cabin crews to reach cost minimizing crew-flight assignments can be further divided. On the one hand, a sequence of flights (so called *pairings*) to be served by a single crew is defined in the Crew Pairing Problem (CPP). On the other hand, the Crew Assignment Problem arises in the so called rostering process, during which individual crews are assigned to one of the established pairings. This paper addresses the CPP, whose intermediate position in the complete planning process can be seen in Figure 1.

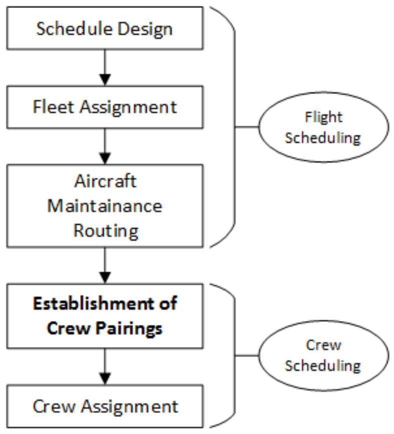

Fig. 1. Position of the CPP in the complete airline planning process

In this paper, a metaheuristic approach based on biased randomization to solve the CPP is presented. The algorithm is tested using a real-life data set provided by a commercial airline (which will not be named due to the existing agreement with the company). Results show that the pairings established by the applied metaheuristic decreases overall crew flying times as well as the necessary number of crews to serve all scheduled flights. Hereby different solutions are provided in only a few seconds, making the algorithm adaptable to short term flight schedule changes and allowing for more decision making flexibility.

This work is structured as follows: the CPP and biased randomization are reviewed and discussed in Section 2. Our solution approach is outlined in Section 3. In Section 4, the suggested algorithm (implemented as C++ application) is tested on a set of real-life data. Finally, the work concludes and discusses possible future work in Section 5.

2 Biased Randomization as solution approach for the CPP

2.1 The Crew Pairing Problem

Given a set F of flights, the aim of the CPP is to establish sequences of flight legs (or segments) from a set P of possible pairings in such a way that overall flying times during a given planning horizon are minimized. Thereby it needs to be ensured that each flight $i \in F$ is included in exactly one pairing $p \in P$ [11]. On the one hand, the complexity of the CPP is driven by the problem size which increases exponentially with a growing number of flight legs that need to be scheduled [21]. On the other hand, various problem constraints need to be considered. As such, established flight schedules defining the date and time of flights need to be considered. Furthermore, labor regulations and internal company rules need to be adhered to: each pairing must start and end at the same airport (the base location of the crew), a minimum transfer time between flight connections must be met, and a limit in daily crew working hours as well as consecutive working days cannot be breached.

Different solution approaches to the CPP have been presented in the literature. [4] combine large neighborhood search with exact enumeration methods and integer programming to solve a large scale CPP over a monthly planning horizon. Another hybrid approach is discussed by [1], who present three different solution approaches based on genetic algorithms, column generation, and a knowledge based random algorithm. Case studies in which the CPP is applied in practical problem settings have also been investigated in the past. In [22], column generation for the improvement of crew pairings of a commercial Taiwanese airline is proposed. Furthermore, [14] establish a robust CPP model considering the short term integration of new flights for small local Turkish airlines. For an extensive overview over the robust CPP, see [20]. While the CPP is discussed as individual optimization issue in the works mentioned in this paragraph, some approaches to integrate the CPP with other airline planning problems exist. An integrated approach over all planning steps is discussed in [16, 18]. Furthermore, [15] combine the aircraft routing problem with crew scheduling, while the CPP and the crew assignment problem is solved globally by [13].

To illustrate the CPP, a simple example with flight legs between three Cities (A. B, C) on a three day planning horizon can be seen in Table 1. The proposed solution (when considering airport A as crew base) shows that the scheduled flights can be completed by two crews (see Table 2).

Table 1. Simple flight scheduling example

Day	**Day 1**			**Day 2**			**Day 3**	
Flight	**A-B**	**A-C**	**B-A**	**A-B**	**C-A**	**B-A**	**A-C**	**C-A**
Departure	10:00	10:00	13:00	8:00	15:00	16:00	9:00	13:00
Arrival	11:35	13:05	14:35	9:35	18:05	17:35	12:05	16:05

Table 2. Possible crew pairings

Pairing	**Day**	**Flight Sequence**
X	1	A-B, B-A
	2	A-B, B-A
Y	1	A-C
	2	C-A
	3	A-C, C-A

2.2 Metaheuristics based on biased randomization

By trading the guarantee for optimality against faster computation times, approximate metaheuristic methods are often the method of choice for large scale optimization problems [19]. Hereby the use of randomization techniques in the construction and/or local search phase is a common approach to tackle combinatorial optimization problems (COPs) [2, 3]. Randomization techniques are often included in multi-start methods, which use multiple algorithm iterations to increase the considered COP solution search space [12].

Biased randomization can be seen as extension of the well-known Greedy Randomized Search Procedure (GRASP) introduced by [17]. A COP typically consists of a finite ground set of elements $E = \{1, ..., n\}$, a set of feasible solutions $F \subseteq 2^E$, and an objective function $f : 2^E \to \mathbb{R}$. GRASP uses various algorithm iterations to build a feasible COP solution using a randomized construction process. At each construction step a new solution element is added, according to the myopic costs of including the element in the currently constructed solution. However, instead of always choosing the element with the highest potential benefits, a list with the most promising elements is created. In the original GRASP, the element to add to the currently constructed solution is chosen from this list according to a uniform distribution.

When using biased randomization, the elements are added according to a skewed probability distribution. Hereby, all available solution elements are eligible at each step. Their probability of being included in the current solution is based on a bias function calculated according to some criteria (e.g., ranking in a list, heuristic value, etc.) [9]. While the element on the first position of the list has the highest chances of being included in the current COP solution, all elements are potentially eligible. Note that the figure shows a geometric distribution, but other theoretical (e.g. triangular) or empirical probability distributions could also be used. For some application examples of biased randomized approaches the reader is referred to [7, 8, 10]

3 Solution Approach for the CPP

To solve the CPP we suggest the use of a multi-start metaheuristic based on biased randomization. A pseudo-code for the algorithm illustrating the following descriptions can be seen in Algorithm 1.

Step 1: Given the flight schedule —i.e., the destinations, days, and exact times of the flights— an initial solution (*initSolution*) is created by assigning one crew per flight. This CPP solution (the worst solution possible) is set as *bestSolution* found so far. Furthermore, the transfer times between any two flights i and j are calculated. If the connection time for $transfer_{ij}$ does not break any flight- or crew related constraints, the corresponding transfer is added to the $transferlist$ of possible flight connections. Once all possible flights have been checked, the created list is ordered from lowest to highest transfer time.

Step 2: Daily flight sequences are created in the following multi-start framework during $maxIter1$ iterations (see Procedure 2). During each iteration, a crew

Algorithm 1: MultiStart (allFlights, alpha, maxIter1, maxIter2)

```
    // Create list of possible tranfers and initial solution
 1  transferList ← readData(allFlights)                          // step 1
 2  initSolution ← createInitialSol(allFlights)                   // step 1
 3  bestSolution ← initSolution                                   // step 1
    // Generate daily parings
 4  for (0 to maxIter1) do
 5      dailyPairing ← initSolution
 6      dailyPairing ← createDailyParing(dailyPairing, tranferList, alpha)
                                                                  // step 2
        // Create pairing over given time horizon
 7      for (0 to maxIter2) do
 8          newSolution ← combineDailyPairings(dailyPairings)     // step 3
            // Ensure that flights return to crew base location
 9          newSolution ← complete(newSolution, allFlights)       // step 4
10          solutionsList ← update(newSolution)                   // step 5
11          if (crewFlightHours(newSolution) < crewFlightHours(bestSolution))
            then
12              bestSolution ← newSolution                        // step 5
            end
        end
    end
13  return bestSolution
```

is assigned to each scheduled flight using biased randomization. That is, pairings consisting of different flight legs are constructed using the sorted $transferList$. According to a skewed probability distribution based on parameter α , the corresponding flights and their associated pairings of the selected transfer are chosen. While α defines the probability of first list entry (i.e. the transfer with the lowest connection time) to be selected, all transfers on the list are potentially eligible. Next, the algorithm checks if the flights are on the same day and that no further constraints are broken. Once a transfer is selected, $transferList$ is updated. This step is repeated until the established list is empty.

Step 3: The daily crew pairings are merged in the next step to establish a complete crew pairing solution over the considered time period (see Procedure 3). According to a uniform distribution, the daily pairings are hereby randomly combined on a single crew duty period when feasible.

Step 4: Until now, constraints concerning the crew base location have not been considered. For each constructed crew pairing solution the base location is therefor included as the first and final destination of the flight sequence as described in Procedure 4 to complete the crew pairings. In practice, this is referred to as so called 'deadheading'. Note that this additional flight connection counts as working time for the crew, leading to scheduled flights can appear more than once in the final crew pairing solution.

Procedure 2: createDailyPairings(dailyPairing, transferList, alpha)

```
1  transferList ← randomSelection (transferList, alpha)
2  while (transferList not empty) do
3      $flight_i$ ← select $flight_i$ of transferList[0]
4      $flight_j$ ← select $flight_j$ of transferList[0]
5      iP ← associatedPairing(dailyPairing, $flight_i$)
6      jP ← associatedPairing(dailyPairing, $flight_j$)
7      if ((time $flight_i$ + time $flight_j$ < max dailyTime) and (iP day = jP day)
       and (iP $\neq$ jP) and ($flight_i$ last in iP) and ($flight_j$ first in jP)) then
8          dailyPairings ← linkSequences(iP, jP)
       end
9      updateList(transferList)
   end
10 return dailyPairings
```

Procedure 3: combineDailyPairings(dailyPairings)

```
1  RandomSort(dailyPairings)
2  for $pos_i$ = 0 to dailyPairings.size() do
3      iP ← dailyPairings[$pos_i$]
4      for $pos_j$ = 0 to dailyPairings.size() do
5          jP ← dailyPairings[$pos_j$]
6          if (iP destination airport = jP origin airport) and (last day iP + 1 =
           first day jP) and (total days iP + total days jP < max pairing days)
           then
7              currentPairings ← combinePairings(iP,jP)
8              $pos_j$ ← 0
           end
       end
   end
9  return currentPairings
```

Step 5: Finally, the constructed solution is included in the solution list (giving decision takers a range of solutions to choose from). The current *bestSolution* is updated when necessary.

4 Computational Experiments

The real life case with which we show the potential of our algorithm consists of 41 flight legs that have to be paired over a five day time period. According to company regulations, the maximum daily flight time per crew is eight hours, while the maximum consecutive working days is limited to three days. Furthermore the minimum transfer time between two flights is 45 minutes. The base location of the flight crews is in Madrid (MAD). Note that deadheading is possible. The set of flights for which the pairings have to be established can be seen

Procedure 4: complete(currentPairings, allFlights)

```
1  for (i = 0 to currentPairing.size()) do
2      iP ← currentPairings[i]
3      if (iP origin airport $\neq$ crew base) then
4          $flight_j$ ← findFlight(iP origin flight)
5          if ($flight_j \neq$ NULL) then
6              jP ← createPairing($flight_j$)
7              iP ← combinePairings(iP, jP)
           end
       end
8      if (iP destination airport $\neq$ crew base) then
9          $flight_j$ ← findFlight(iP destination flight)
10         if ($flight_j \neq$ NULL) then
11             jP ← createPairing($flight_j$)
12             iP ← combinePairings(iP, jP)
           end
       end
13     update(currentPairings, iP)
   end
14 completePairings ← currentPairings
15 return completePairings
```

in Table 3. The current pairings used by the company to schedule their crews on the flights is outlined in Table 4. With the current pairings, the scheduled flights are accompanied by six crews, with an overall flight time of 69 hours 40 minutes. Moreover, there are three deadheading flights.

To test our algorithm, we use a geometric distribution algorithm α of 0.25. Furthermore, we apply 250 iterations for the daily pairing construction phase ($maxIter1$). Each time a daily solution is constructed, they are then combined in an iterative process using 25 iterations ($maxIter2$). The algorithm was implemented with C++ on a personal computer with a Intel Core i5-2400 (3,1GHz) processor and 3GB RAM. Results show that our biased randomization metaheuristic is able to reduce total flight time by two hours to 67 hours and 40 minutes, as only one deadheading flight is necessary. Moreover, the proposed solution only needs five crews to serve all flights in the considered period (instead of the six in the current solution).

Another important attribute of our metaheuristic are the short calculation times. As the complete process can be completed in only a few seconds (approximately 20 seconds considering the parameters and CPU used to obtain the presented results), the algorithm is adaptable to short term flight schedule changes as they regularly occur in the airline industry. Moreover, the multi-start procedure leads to a list of promising solutions, enabling decision takers to choose from more than one option according to individual preferences and necessities.

Table 3. Set of scheduled flights

Day 1		**Day 2**		**Day 3**		**Day 4**		**Day 5**	
Flight	*Time*	*Flight*	*Time*	*Flight*	*Time*	*Flight*	*Time*	*Flight*	*Time*
MAD-BCN	7.00-8.00	BCN-ORY	6.25-8.05	BCN-PMI	5.50-6.30	BRU-MAD	7.15-9.35	LPA-MAD	11.20-13.50
BCN-FCO	8.45-10.25	ORY-BCN	8.55-10.30	MAD-BCN	7.00-8.00	BCN-FCO	8.45-10.25	MAD-FRA	15.05-17.35
FCO-BCN	11.40-13.20	MAD-SCQ	17.45-18.55	PMI-BCN	7.15-8.00	BCN-MAD	8.55-9.55	BCN-MUC	15.15-17.15
				SCQ-MAD	8.10-9.15	MAD-NCE	10.50-12.30	MUC-MAD	18.00-20.35
				BCN-FCO	8.45-10.25	FCO-BCN	11.40-13.20	FRA-MAD	18.25-20.55
				BCN-MAD	8.55-9.55	BCN-ORY	13.15-14.55	AMS-MAD	18.40-21.05
				BCN-BRU	9.00-11.05	NCE-MAD	13.20-15.05		
				MAD-NCE	10.50-12.30	MAD-SCQ	15.05-16.15		
				MAD-BCN	11.30-12.30	BCN-MXP	15.10-16.40		
				FCO-BCN	11.40-13.20	ORY-BCN	15.45-17.20		
				NCE-MAD	13.20-15.05	MAD-LPA	16.35-19.25		
				BCN-SCQ	15.05-16.40	SCQ-MAD	17.05-18.10		
				MAD-BCN	16.00-17.00	MXP-BCN	17.35-19.20		
				SCQ-BCN	17.35-19.05	BCN-MAD	18.45-19.45		
						MAD-AMS	19.10-21.30		

Table 4. Currently used pairings (*deadheading flights)

Pairing	Day	Flight Sequence					Transfer Time	Total Time
A	4	MAD-SCQ	SCQ-MAD	MAD-AMS			1h30m	6h25m
	5	AMS-MAD					0h00m	2h25m
B	2	MAD-SCQ					0h00m	1h10m
	3	SCQ-MAD	MAD-BCN	BCN-SCQ	SCQ-BCN		5h45m	10h55m
	4	BCN-ORY	ORY-BCN	BCN-MAD			1h55m	6h30m
C	3	MAD-BCN	BCN-FCO	FCO-BCN			2h00m	6h20m
	4	BCN-FCO	FCO-BCN	BCN-MXP	MXP-BCN		4h00m	10h35m
	5	BCN-MUC	MUC-MAD				0h45m	5h20m
D	3	MAD-BCN*	BCN-MAD	MAD-NCE	NCE-MAD	MAD-BCN	3h35m	10h00m
	4	BCN-MAD	MAD-NCE	NCE-MAD	MAD-LPA		3h15m	10h30m
	5	LPA-MAD	MAD-FRA	FRA-MAD			2h05m	9h35m
E	3	MAD-BCN*	BCN-BRU				1h00m	4h05m
	4	BRU-MAD					0h00m	2h20m
F	1	MAD-BCN	BCN-FCO	FCO-BCN			2h00m	6h20m
	2	BCN-ORY	ORY-BCN				0h50m	4h05m
	3	BCN-PMI	PMI-BCN	BCN-MAD*			1h40m	4h05m

5 Conclusion

In this paper a multi-start metaheuristic based on biased randomization to tackle the complex CPP is presented. Computational results on a real-life scenario show that our approach leads to improvements concerning overall flying times due to less necessary deadheading and the overall number of crews to accompany the scheduled flights. The very fast calculation times make the algorithm applicable to short term schedule changes. Furthermore, the range of solutions created in the mutli-start procedure gives decision takers more flexibility in the decision taking process.

Future research will include a more detailed analysis of the algorithm's robustness. Furthermore, an integrated approach together with other decision making problems in the airline planing phase (e.g. concerning aircraft routing) could be done.

Table 5. Our solution (*deadheading flights)

Pairing	Day	Flight Sequence					Transfer Time	Total Time
A	2	MAD-SCQ					0h00m	1h10m
	3	SCQ-MAD	MAD-BCN				2h05m	6h45m
	4	BCN-ORY	ORY-BCN	BCN-MAD			2h15m	4h15m
B	3	MAD-BCN	BCN-SCQ	SCQ-BCN			3h30m	4h05m
	4	BCN-FCO	FCO-BCN	BCN-MXP	MXP-BCN		4h00m	6h35m
	5	BCN-MUC	MUC-MAD				0h45m	4h35m
C	3	MAD-BCN	BCN-FCO	FCO-BCN			2h00m	4h20m
	4	BCN-MAD	MAD-NCE	NCE-MAD	MAD-LPA		3h15m	7h20m
	5	LPA-MAD	MAD-FRA	FRA-MAD			2h05m	7h20m
D	3	MAD-BCN*	BCN-BRU				1h00m	3h05m
	4	BRU-MAD	MAD-SCQ	SCQ-MAD	MAD-AMS		7h20m	6h55m
	5	AMS-MAD					0h00m	2h25m
E	1	MAD-BCN	BCN-FCO	FCO-BCN			2h00m	4h20m
	2	BCN-ORY	ORY-BCN				0h50m	3h15m
	3	BCN-PMI	PMI-BCN	BCN-MAD	MAD-NCE	NCE-MAD	3h25m	5h50m

References

1. A. Aydemir-Karadag, B. Dengiz, and A. Bolat, "Crew pairing optimization based on hybrid approaches," *Computers & Industrial Engineering*, vol. 65, no. 1, pp. 87–96, 2013.
2. M. Clerc, *Guided Randomness in Optimization.* London: Wiley-ISTE, 2006, vol. 1.
3. P. Collet and J. P. Rennard, "Stochastic Optimization Algorithms," in *Handbook of Research on Nature Inspired Computing for Economics and Management*, J. P. Rennard, Ed. Idea Group Inc., 2006, pp. 28–44.
4. G. Erdogan, M. Haouari, M. O. Matogl, and O. O. Özener, "Solving a large-scale crew pairing problem," *Journal of the Operational Research Society*, vol. 66, pp. 1742–1754, 2015.
5. C. Gao, "Airline integrated planning and operations," Ph.D. dissertation, Georgia Institute of Technology, 2007.
6. B. Gopalakrishnan and E. Johnson, "Airline crew scheduling: State-of-the-art," *Annals of Operations Research*, vol. 140, no. 1, pp. 305–337, 2005.
7. A. Gruler, A. A. Juan, C. Contreras-Bolton, and G. Gatica, "A Biased-randomized Heuristic for the Waste Collection Problem in Smart Cities," in *Proceedings of the 2015 International Conference of the Forum for Interdisciplinary Mathematics (FIM2015)*, 2015.
8. A. A. Juan, H. Lourenço, M. Mateo, R. Luo, and Q. Castella, "Using Iterated Local Search for Solving the Flow-Shop Problem: Parallelization, parameterization, and randomization issues," *Int. Transactions in Operational Research*, vol. 21, no. 1, pp. 103–126, 2014.
9. A. A. Juan, J. Faulin, A. Ferrer, H. R. Lourenço, and B. Barrios, "MIRHA: Multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems," *Top*, vol. 21, no. 1, pp. 109–132, 2013.
10. A. A. Juan, I. Pascual, D. Guimarans, and B. Barrios, "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem," *International Transactions in Operational Research*, vol. 22, no. 4, pp. 647–667, 2015.

11. D. Klabjan, Y. C. Lee, and G. Stojković, “Crew management information systems,” in *Quantitative Problem Solving Methods in the Airline Industry*, C. Barnhart and B. Smith, Eds. Springer US, 2012, pp. 237–282.
12. R. Martí, M. Resende, and C. C. Ribeiro, “Multi-start methods for combinatorial optimization,” *European Journal of Operational Research*, vol. 226, no. 1, pp. 1–8, 2013.
13. C. P. Medard and N. Sawhney, “Airline crew scheduling from planning to operations,” *European Journal of Operational Research*, vol. 38, no. 3, pp. 1013–1027, 2007.
14. I. Muter, S. I. Birbil, K. Bülbül, G. Şahin, H. Yenigün, D. Taş, and D. Tüzün, “Solving a robust airline crew pairing problem with column generation,” *Computers & Operations Research*, vol. 40, no. 3, pp. 815–830, 2013.
15. F. M. Nurul, M. Z. Zaitul, S. Said, N. H. M., M. Huda, and A. M. Nurual, “A heuristic and exact method: Integrated aircraft routing and crew pairing problem,” *Modern Applied Science*, vol. 10, no. 4, pp. 128–136, 2016.
16. N. Papadakos, “Integrated airline scheduling,” *Computers and Operations Research*, vol. 36, pp. 176–195, 2009.
17. M. G. C. Resende and C. C. Ribeiro, “GRASP : Greedy Randomized Adaptive Search Procedures,” in *Handbook of Metaheuristics*, F. Glover and G. Kochenberger, Eds. Springer New York, 2003, pp. 219–249.
18. J. J. Salazar-González, “Approaches to solve the fleet-assignment, aircraft-routing, crew-pairing and crew-rostering problems of a regional carrier,” *Omega*, vol. 43, pp. 71–82, 2014.
19. E.-G. Talbi, *Metaheuristics: From Design to Implementation.* London: Wiley, 2006.
20. H. Tekiner, “Robust crew pairing for managing extra flights,” Master’s thesis, 2006.
21. P. H. Vance, C. Barnhart, E. L. Johnson, and G. L. Nemhauser, “Airline crew scheduling: A new formulation and decomposition algorithm,” *Operations Research*, vol. 45, no. 2, pp. 183–200, 1997.
22. S. Yan, T. T. Tung, and Y. P. Tu, “Optimal construction of airline individual crew pairings,” *Computers and Operations Research*, vol. 29, pp. 341–363, 2002.