RC 21642 (01/14/00)
Computer Science/Mathematics

IBM Research Report

The Airline Crew Scheduling Problem: Dual Simplex, Volume, and Volume/SPRINT Solutions

J. Snowdon, R. Anbil, G. Pangborn

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York

Research Division
Almaden · T.J. Watson · Tokyo · Zurich

The Airline Crew Scheduling Problem: Dual Simplex, Volume, and Volume/SPRINT Solutions

J. Snowdon R. Anbil G. Pangborn

January 2000

Abstract

The problem addressed in this paper is airline crew pairing optimization where the objective is to find a minimum cost set of legal duties and pairings which covers every flight leg. The reported computational results contrast the three linear programming methods Dual Simplex, Volume Algorithm, and Volume/SPRINT Algorithm in their ability to improve the solution quality and performance of these computationally difficult problems. Tests were conducted on an IBM RS6000, Model 240 and a Sparc Ultra 1 using Mini-CPOS. Mini-CPOS is a smaller, non-commercial version of IBM's Crew Pairing Optimization System (CPOS) which is used for research and development purposes.

Key Words:
Large-scale linear programming, Set Partitioning, crew pairing, column generation.

1 Introduction

Airline schedule planning is generally broken down into several steps: creation of the flight segment timetable given to customers, assignment of fleet types and planes to flight segments, crew pairing, bid-line assignment, and yield management. The crew pairing step has been the subject of great interest for many years because it has never been solved to optimality, and because crew costs are second only to fuel costs in the operating budget. [5].

For the crew pairing problem an airline generally starts with the schedule of flight segments (which each need to be covered by exactly one crew) and their corresponding fleet assignments. This problem is decomposed into separate problems for different crew types (pilots, flight attendants) and for different fleet types. The flight legs are joined together to form duties, which are essentially work shifts. These duties are then combined into trips/pairings which are one to five days in length and which start and end at a crew's home base. There are many Federal Aviation Authority and union regulations governing the legality of duties and pairings. Some typical ones may include: a limit on the number of hours of flying time in a duty, a limit on the total length of a duty (possibly stricter for a nighttime duty), a limit on the number of duties in a pairing, upper and lower bounds on the rest time between two duties, and a limit on the number of routing breaks in a duty.

The generation of good crew pairings is complicated by the fact that a crew's pay is determined by several guarantees which are based on the structure of the duties in the pairing. Some common rules governing crew compensation may include: a crew is paid overtime for more than five hours of flying in a duty, a crew is guaranteed a minimum level of pay per duty period, a crew is guaranteed that a certain percentage of the duty time will count as flying time, a crew is paid a per diem when they rest away from their base, a crew receives a bonus if the time they spend away from their base exceeds a specified length, and a crew receives a bonus for overnight flying. Some companies also include additional constraints and objective function penalties in order to avoid marginally legal pairings, tight connections, and excessive ground transportation.

The airline crew scheduling problem can be formulated as a Set Partitioning problem (SP):

$$\begin{aligned} \min c^t x & \\ \text{s.t. } Ax &= 1 \\ x \text{ binary} & \end{aligned}$$

where each row of A corresponds to a flight leg that must be staffed, each column of A corresponds to a legal crew trip, and c corresponds to the costs of the trips.

The trouble is that these crew scheduling problems are enormous. The hub spoke networks employed by most domestic carriers result in an explosion in the number of legal pairings; just as customers are given the opportunity to make a multitude of connections so are crews. For example one United Airlines instance has nearly 2000 flights a day, and each flight into their Chicago hub has as many as 100 possible connections [3]. Anbil, Forrest, and Pulleyblank [4] provide several examples of the prohibitive size of these problems. For problem UA737, pairings may contain up to 4 duty days, there are 773 flight segments to be covered, and seven crew bases. There are 568,000 possible duties, and 100,000,000,000,000 possible pairings. For problem USDC9, pairings may contain up to 4 duty days, there are 478 flight segments to be covered, and four crew bases. There are 562,000 possible duties, and 105,000,000,000,000 possible pairings. For problem AAF10, pairings may contain up to 4 duty days, there are 307 flight segments to be covered, and three crew bases. There are 55,000 possible duties, and 63,200,000,000 possible pairings.

Due to the enormous size of these problems even solving linear programming relaxations containing just a small fraction of the total number of pairings can take considerable time. Therefore it is very important to generate an effective subset of pairings to consider, and to solve the corresponding LPs efficiently. The Dual Simplex algorithm [6] is frequently used for Set Partitioning problems since primal degeneracy (which is an issue here) can negatively impact the performance of Primal Simplex. The Volume algorithm of Barahona and Anbil [1] is an extension of the subgradient algorithm for linear programming which appears to work very successfully for these Set Partitioning linear programs.

2 Volume Algorithm

The Volume algorithm is able to produce both approximate primal and dual solutions (unlike subgradient algorithms which produce only dual solutions). It is based on a new theorem for linear programming duality. Given an optimal solution to the primal (maximization) problem $(\hat{\pi}, \hat{z})$, consider the subset of constraints that are active at that point as well as a lower bound on the objective function $\bar{z}$. The optimal dual solution can be computed as a function of the volumes between the active constraints and the hyperplane $z = \bar{z}$. The Volume algorithm shows good convergence properties for linear programs derived from combinatorial optimization problems. Furthermore, it uses minimal storage space since it does not require a basis factorization (as Simplex does) or the matrix AA^T (as Barrier methods do).

The Subgradient algorithm starts with a dual feasible solution π, computes the subgradient at π, takes a step in that direction, and repeats this procedure until a nearly optimal solution is found. The Volume algorithm behaves in a similar manner, only in this case the direction of movement is a convex combination of the subgradients computed at each step. Figure 1 demonstrates the progress of these two algorithms.

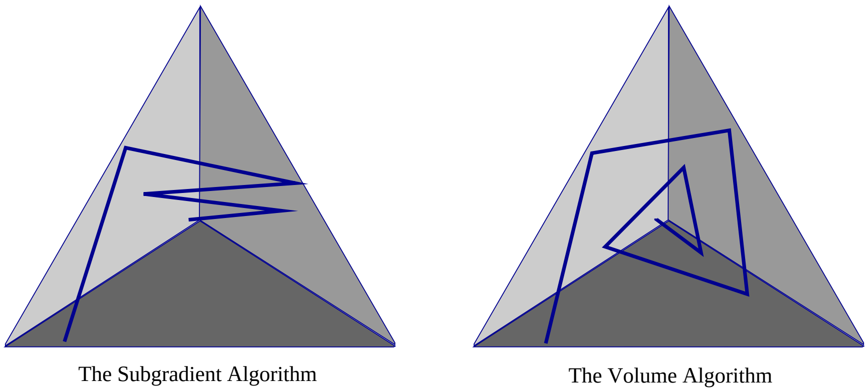

Figure 1: Algorithm Comparison

A more detailed sketch of the Volume algorithm for the Set Partitioning problem is given below:

Consider the Lagrangean Relaxation (LR) of the linear programming relaxation of the crew scheduling Set Partitioning Problem (SP) described above:

$$z(\pi) = \min\ (c - \pi A)x + x\mathbf{1}$$
$$0 \leq x \leq \mathbf{1}$$

Step 0

Given an initial dual solution $\bar{\pi}$.
Solve (LR) to obtain values for $\bar{x}$ and $\bar{z}$.

Set $t = 1$ (the iteration number).

Step 1
Compute the improvement direction $v^t = \mathbf{1} - Ax$.

Step 2
Update the dual solution as $\pi^t = \bar{\pi} + sv^t$, where the step size $s = \lambda(UB - \bar{z})/\|v\|$ and λ is between 0 and 2 depending upon the progress made in previous iterations.

Step 3
Solve (LR) for x^t and z^t using π^t.

Step 4
Update the primal solutions $\bar{x} \leftarrow \alpha x^t + (1-\alpha)\bar{x}$, where $\alpha \in (0, 1)$ and α decreases as the algorithm continues.
Set $\bar{\pi} = \pi^t$ and $\bar{z} = z^t$.

Return to **Step 1**

It is also possible to take an approximate solution to the Set Partitioning LP generated by the Volume algorithm and perturb it slightly to find a good starting point for the Dual Simplex algorithm. To do this consider all of the primal variables in the Volume solution that have strictly positive values, as well as a subset of the variables with the best reduced costs. A dual feasible solution can be found by adjusting the current approximate dual solution. (For $\bar{c}_j < 0$ compute $\epsilon = \bar{c}_j / \Sigma_i a_{ij}$ and update the dual variable $\bar{\pi}_i = \bar{\pi}_i + \epsilon a_{ij}$.) Then use the Dual Simplex algorithm to obtain a primal feasible solution. Since not all of the primal variables are being considered the resulting primal solution may not be optimal, but the gap should be reasonably small (Barahona and Anbil [2]).

Barahona and Anbil [2] compare the Volume algorithm to Dual Simplex and the Barrier method on a test bed of Set Partitioning instances that arose in the solution of airline crew scheduling problems. The times for the Volume algorithm were on average 6.5 times faster than for Dual Simplex, and the times for the Volume algorithm with cross over were 4.5 times faster. The Volume times were comparable to those of the Barrier method. The significant gain there was that the Barrier method required 6 to 10 times the space that the Volume algorithm did, and up to eight times the space that the Volume algorithm with cross over (to Dual Simplex) did.

3 SPRINT approach

Anbil, Forrest, and Pulleyblank [4] use the SPRINT methodology for linear programming to solve airline crew scheduling problems. They consider just a small subset (around 5000) of the total number of columns (possibly millions) in the linear program. Dual Simplex is used to solve a small LP defined by the subset of columns. The resulting dual variables are then used to price out the complete set of columns. The next subset of columns is chosen on the basis of the reduced costs (and some at random) and the procedure iterates until an optimal solution is found. They were able to obtain speedups of four to five times over Dual Simplex on problems with 837 rows and 100,000 to 500,000 columns.

4 Computational Experiments

The computational results presented here compare an implementation of the Volume algorithm to an implementation of the SPRINT approach that uses the Volume algorithm instead of Dual Simplex to solve the small LPs. Both of these implementations were developed by Ranga Anbil.

The first dataset was created using the IBM Mini-CPOS initial column generation routine for varying numbers of iterations. (This program is described later on.) These runs were done on an IBM RS6000 model 240. The subset size used for the SPRINT was 1.2$*$(the number of rows)$+$200. The results are presented in Tables 1 and 2.

Table 1: Results for the Volume/SPRINT Code on the First Dataset

rows	columns	nonzeros	time (sec)
2302	8452	62530	74.7
2302	20163	164484	167
2302	26128	218827	170.6
2302	37820	320885	341.8
2302	50092	451983	543.3
2302	74414	732699	672.3
2302	97886	1032832	771
2302	115489	1293251	831

The second dataset contains problems sp1 through sp12 from the airline crew scheduling Set Partitioning dataset used by Barahona and Anbil [2] for their comparisons of the Volume, Dual Simplex, and Barrier methods. The results are presented in Tables 3 and 4.

The results of these runs generally confirm the expected behavior. The SPRINT version usually outperforms the original volume code. For the crew pairing dataset the average speedup was 24%, while for the sp dataset the average speedup was 19%. In particular, SPRINT tends to be useful as the ratio of columns to rows grows and when the number of columns is very large. But the speedup is not as consistent or as striking as it was when SPRINT method was applied to the Dual Simplex algorithm [4].

Table 2: Results for the Volume Code on the First Dataset

rows	columns	nonzeros	time (sec)
2302	8452	62530	78.6
2302	20163	164484	219.7
2302	26128	218827	257.4
2302	37820	320885	470.8
2302	50092	451983	611.7
2302	74414	732699	971.7
2302	97886	1032832	845
2302	115489	1293251	1710

Table 3: Results for the Volume/SPRINT Code on the Second Dataset

	rows	columns	nonzeros	time (sec)
sp1	204	6954	94687	46.9
sp2	173	3686	45065	24
sp3	111	1668	27177	21.5
sp4	368	9144	150880	77.9
sp5	684	13718	162571	235.5
sp6	2504	50722	550643	617.5
sp7	2991	43459	499346	735.5
sp8	4810	91123	1004472	1400
sp9	2917	50013	742545	838
sp10	781	13128	220702	166.5
sp11	104	2275	56685	22.5
sp12	3218	84746	910021	766.5

Table 4: Results for the Volume Code on the Second Dataset

	rows	columns	nonzeros	time (sec)
sp1	204	6954	94687	84.5
sp2	173	3686	45065	60.2
sp3	111	1668	27177	18.5
sp4	368	9144	150880	117.1
sp5	684	13718	162571	142.6
sp6	2504	50722	550643	605.5
sp7	2991	43459	499346	664
sp8	4810	91123	1004472	1590
sp9	2917	50013	742545	923.5
sp10	781	13128	220702	203.5
sp11	104	2275	56685	54
sp12	3218	84746	910021	947

The IBM Mini-CPOS program starts with an artificial set of pairings (that guarantees a feasible solution), solves the corresponding Set Partitioning linear program, and then uses the resulting dual variable values to generate promising pairings, (pairings with negative reduced cost values), to add to the linear program. It then resolves the LP and repeats this process until there is no further improvement in the objective function value. Figure 2 indicates the speed at which Mini-CPOS was able to bring down the LP objective function when calling Dual Simplex, Volume/SPRINT, and Volume to solve the intermediate linear programs. These runs were completed on a Sparc Ultra 1. The subset size used for SPRINT was 2950 (1.2∗(the number of rows)+200).

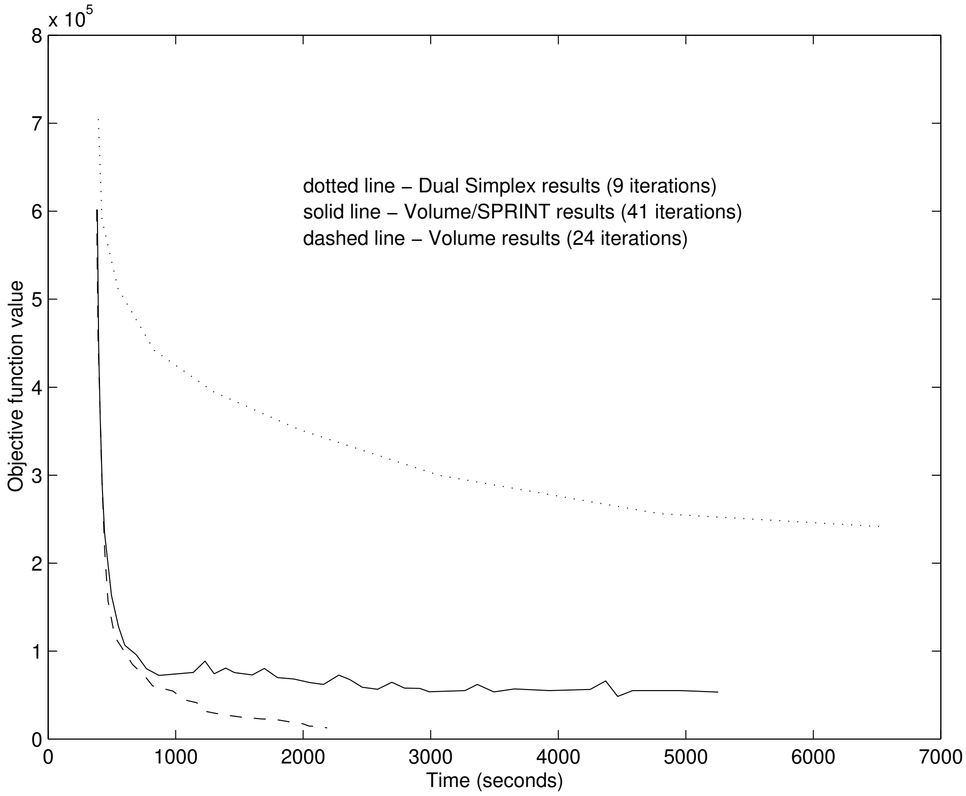

Figure 2: Mini-CPOS LP Solver Comparison

The Volume algorithm is much more effective than the Volume/SPRINT algorithm in this context. Both of these algorithms outperform the Dual Simplex algorithm by an immense margin. Figure 3 indicates the results that were obtained by running Mini-CPOS with different subset sizes (given in parenthesis in the figure) for Volume/SPRINT. Clearly, the larger the subset size for SPRINT the better the results for column generation.

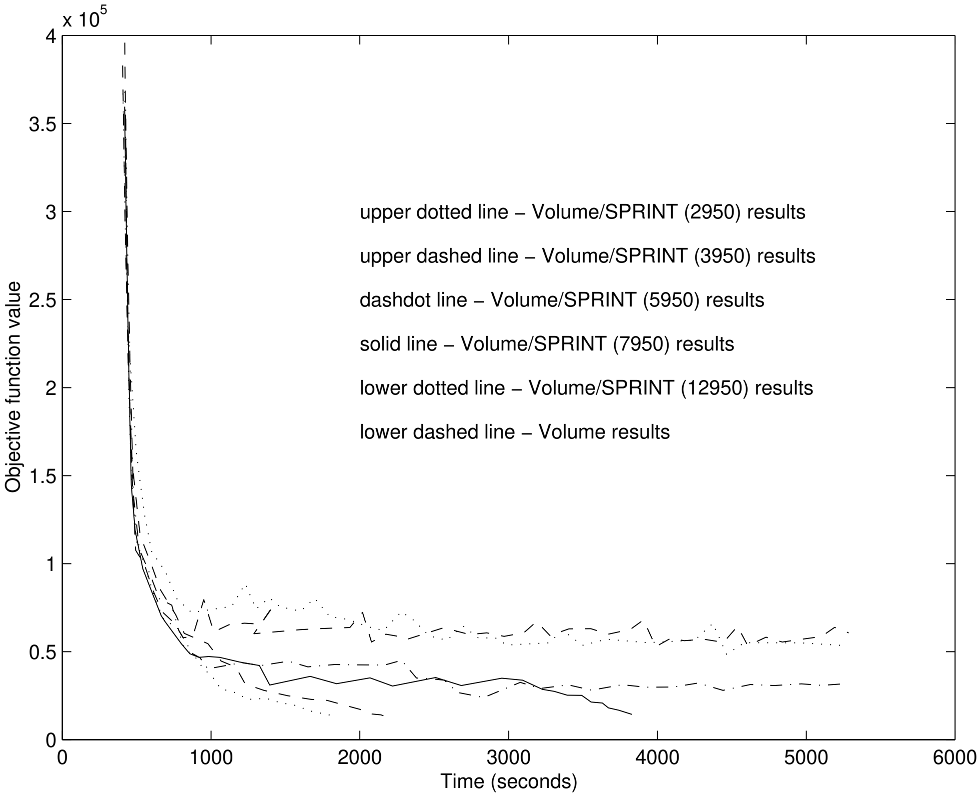

Figure 3: Mini-CPOS Varying SPRINT Subset Size

5 Conclusions

It is not entirely clear why the Volume algorithm does so well at producing 'good' dual solutions to be used for the column generation step. (Or, similarly, why larger SPRINT subset sizes are helpful.) One plausible hypothesis is that the Volume algorithm produces solutions that are further from the corner points of the dual polytope. These solutions may provide more global information about the primal polytope.

While the Volume algorithm remains the best choice for Mini-CPOS's column generation step, it is not clear that it is the best choice throughout the Crew Pairing solution process. When heuristically reducing the problem size, and in the early levels of the Branch and Bound tree, the combined Volume SPRINT may be the best approach, (since it was the fastest approach for approximately solving Set Partitioning LPs). The Dual Simplex algorithm may be the most effective choice for solving later nodes in the Branch and Bound tree when the LPs are smaller and there is some chance of finding an integral solution.

References

[1] Francisco Barahona; Ranga Anbil. The volume algorithm: producing primal solutions with a subgradient algorithm. Technical Report RC 21103, IBM Watson Research Center, 1998.

[2] Francisco Barahona; Ranga Anbil. On some difficult linear programs coming from set partitioning. Technical Report RC 21410, IBM Watson Research Center, 1999.

[3] Glen Graves; Richard McBride; Ira Gershkoff; Diane Anderson;Deepa Mahidhara. Flight crew scheduling. *Management Science*, 39(6):736–745, June 1993.

[4] Ranga Anbil;John Forrest; William Pulleyblank. Column generation and the airline crew pairing problem. *Documenta Mathematica*, pages 500–509, 1998. Extra Volume ICM.

[5] Ranga Anbil;Eric Gelman;Bruce Patty;Rajan Tanga. Recent advances in crew pairing optimization at american airlines. *Interfaces*, 21(1):62–74, January-February 1991.

[6] Robert J. Vanderbei. *Linear Programming Foundations and Extensions*. Kluwer Academic Publishers, 1997.