Scheduling Aircraft to Reduce Controller Workload

Joondong Kim* Alexander Kröller* Joseph S.B. Mitchell*
Girishkumar R. Sabhnani†

Abstract

We address a problem in air traffic management: scheduling flights in order to minimize the maximum number of aircraft that simultaneously lie within a single air traffic control sector at some time t. Since the problem is a generalization of the NP-hard no-wait job-shop scheduling, we resort to heuristics. We report experimental results for real-world flight data.

Keywords: Air Traffic Control, trajectory scheduling, flight plan scheduling, no-wait job shop.

1 Introduction

In the air traffic control system, the volume of airspace in the altitude range that aircraft utilize is partitioned into a set of *sectors*. We consider the set of all trajectories flown between city pairs. Any one trajectory is modeled as a polygonal path, from each vertex (*way point*) being specified by a point, (x, y, z, t), in space-time. For a given set of sectors and a given set of trajectories, we can compute the *occupancy count*, $n_\sigma(t)$, of a sector σ at any time t. For purposes of air traffic control, it is important that $n_\sigma(t)$ not be "too large"; often the occupancy count is compared with the *Monitor Alert Parameter* (MAP) value of the sector σ, which is related to the "capacity" of the sector. Depending on the timing and routing of the flights, though, the MAP values of certain congested sectors are often predicted to be exceeded (if current flights remain on filed flight plans), resulting in the rerouting of aircraft to avoid those sectors that are anticipated to be at or near full capacity during some period of time.

We consider the following scheduling problem: For a given set of trajectories and a given sectorization of the NAS, determine alternate departure times ("close" to the originally scheduled times) so that the modified trajectories result in minimizing $\max_{\sigma,t} n_\sigma(t)$, the maximum occupancy count of a sector over a time window of interest.

Problem Statement: Formally, the problem is defined as follows: Given a set Σ of *sectors* and a set Θ of periodic *flight plans*. The common period of all plans is T, e.g., $T = 24$ hours. Each flight plan θ defines a sequence $\Sigma_\theta = (\sigma_{\theta,1}, \sigma_{\theta,2}, \ldots)$ of the sectors it visits, where $\sigma_{\theta,k} \in \Sigma\ \forall k$. It also defines a *departure time* $d_\theta \in [0, T)$, and for each sector $\sigma_{\theta,k}$ the *dwell time* $t_{\theta,k}$.

Assuming a flight θ departs daily with a delay of Δ_θ, it will therefore be in sector $\sigma_{\theta,k}$ during the intervals

$$I_\theta(\sigma_{\theta,k}, \Delta_\theta) := [\sum_{\ell<k} t_{\theta,\ell}, \sum_{\ell\le k} t_{\theta,\ell}) + d_\theta + \Delta_\theta + T\mathbb{Z}. \quad (1)$$

Therefore, at time $t \in [0, T)$ (and also $t + kT$ for any $k \in \mathbb{Z}$), a total of

$$n_\sigma(t) := |\{\theta \in \Theta : t \in I_\theta(\sigma, \Delta_\theta)\}| \quad (2)$$

flights will be in sector $\sigma \in \Sigma$.

Our goal is to find delays $(\Delta_\theta)_{\theta\in\Theta}$ to minimize the overall maximum occupancy count $\max_{\sigma,t} n_\sigma(t)$. The delays are constrained to be within the range $[0, D]$ for parameter D. Note that additionally allowing flights to leave early, i.e., $\Delta_\theta < 0$, does not change the problem due to the periodicity of flight plans: A delay range $[-a, b]$ is equivalent to $[0, a + b]$, for $a, b > 0$.

Relation to Job-Shop Scheduling: When there is no constraint on the maximum delay, i.e., $D \ge T$, our problem is equivalent to no-wait job-shop scheduling. We represent each flight plan as a job and each sector as a machine. We seek to minimize *makespan*, i.e., the smallest time in which all jobs can be processed, where no two jobs can be on the same machine at the same time. The no-wait constraint ensures that, once started, a job can neither be delayed between machines nor suspended while being processed on one. An optimal solution to the job-shop problem with makespan M can be converted

*Dept. Applied Mathematics and Statistics, Stony Brook University, {jdkim,kroeller,jsbm}@ams.sunysb.edu
†Dept. Computer Science, Stony Brook University, gk@cs.sunysb.edu

	$\max_{\sigma,t} n_\sigma(t)$	
	set1	set2
Original flight plan	38	58
Shifting	30	49
Incremental	27	43
Randomized rounding	29	39
Lower Bound	20	32

Table 1: Workload improvement of algorithms

	$\max_\theta \Delta_\theta$		$(\sum_\theta \Delta_\theta)/$ $\lvert\{\theta \lvert \Delta_\theta \neq 0\}\rvert$	
	set1	set2	set1	set2
Shifting	0.017	0.018	0.005	0.004
Incremental	0.042	0.041	0.009	0.013
Randomized rounding	0.041	0.041	0.003	0.008

Table 2: Rescheduling statistics ($T = 1$, $D = 0.042$, and $\Delta_\theta \in [0, D]$)

trivially to a flight plan solution with maximum occupancy $\lceil M/T \rceil$. Vice versa, an algorithm for flight plan scheduling also solves job-shop by finding the largest λ for which a flight plan with all processing times scaled by λ can be scheduled with maximum occupancy 1. This can be achieved using binary search.

No-wait job-shop scheduling has attracted various researchers (see, e.g., [4, 6, 7, 5, 3]). [1] gives a *PTAS* for a special case of the problem and shows hardness of approximation for another case. [2] provides a survey of scheduling algorithms, defining the various terms and known results for some of the basic problems. Since the job-shop problem is NP-hard, so is flight plan scheduling.

2 Results

We designed several heuristic algorithms and compare them with a given flight plan and a lower bound.

The **shifting** heuristic is a greedy algorithm selecting a flight trajectory and adjusting its start time to lexicographically decrease the workload vector. The selected trajectory is one of those participating the most crowded interval of the sector with currently maximum workload. We repeat this shifting until no more shifts can reduce the workload vector. The **incremental** algorithm repeatedly applies the shifting heuristic while adding trajectories one at a time, in random order.

The **randomized rounding** algorithm solves a linear programming formulation whose variables describe a probability distribution for each flight plan. Then, a solution is generated by drawing delays from these distributions.

For **lower bounds**, we solve a linear programming relaxation of a formulation similar to the one used in randomized rounding.

We used real-world data for our experiment, consisting of 57 sectors and 12123 trajectories for set1 and 1281 sectors and 11986 trajectories for set2. Table 1 shows the maximum occupancies in the given flight plans and heuristic solutions, as well as the lower bound. Table 2 shows how much they are altered. The results show a considerable improvement over the originally scheduled flight times. Future work will specifically aim to improve the lower bound, as we believe that the heuristically produced solutions are already almost optimal.

Acknowledgements. The data used for the experiments was provided by Metron Aviation. We thank Michael Bender and Bob Hoffman for helpful discussions. This work was partially supported by NSF (CCF-0431030, CCF-0528209, CCF-0729019), NASA Ames, and Metron Aviation.

References

[1] N. Bansal, M. Mahdian, and M. Sviridenko. Minimizing makespan in no-wait job shops. *Math. Oper. Res.*, 30(4):817–831, 2005.

[2] D. Karger, C. Stein, and J. Wein. Scheduling algorithms. *CRC Handbook of Computer Science*, 1997.

[3] P. M. Lennartz. *No-Wait Job Shop Scheduling, a Constraint Propagation Approach.* PhD thesis, UU Universiteit Utrecht, Netherlands, 2006.

[4] A. Mascis and D. Pacciarelli. Job shop scheduling with blocking and no-wait constraints. *Eur J. Oper. Res.*, 142:498–517, 2002.

[5] C. J. Schuster. No-wait job shop scheduling: Tabu search and complexity of subproblems. *Mathematical Methods of Operations Research*, 63(3):473–491, July 2006.

[6] C. J. Schuster and J. Framinan. Approximative procedures for no-wait job shop scheduling. *Oper Res Lett*, 31:308–318, 2003.

[7] G. J. Woeginger. Inapproximability results for no-wait job shop scheduling. *Oper. Res. Lett.*, 32:320–325, 2004.