Transportation Research Part C 95 (2018) 570–581

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

An optimal data-splitting algorithm for aircraft scheduling on a single runway to maximize throughput[☆]

Rakesh Prakash, Rajesh Piplani*, Jitamitra Desai

School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore

ARTICLE INFO

Keywords:
Aircraft sequencing problem
0–1 mixed-integer programming
Asymmetric traveling salesman problem with time-windows
Data-splitting algorithm
Runway throughput
Runway optimization

ABSTRACT

In this research, we present a data-splitting algorithm to optimally solve the aircraft sequencing problem (ASP) on a single runway under both segregated and mixed-mode of operation. This problem is formulated as a 0–1 mixed-integer program (MIP), taking into account several realistic constraints, including safety separation standards, wide time-windows, and constrained position shifting, with the objective of maximizing the total throughput. Varied scenarios of large scale realistic instances of this problem, which is NP-hard in general, are computationally difficult to solve with the direct use of commercial solver as well as existing state-of-the-art dynamic programming method. The design of the algorithm is based on a recently introduced data-splitting algorithm which uses the divide-and-conquer paradigm, wherein the given set of flights is divided into several disjoint subsets, each of which is optimized using 0–1 MIP while ensuring the optimality of the entire set. Computational results show that the difficult instances can be solved in real-time and the solution is efficient in comparison to the commercial solver and dynamic programming, using both sequential, as well as parallel, implementation of this pleasingly parallel algorithm.

1. Introduction

Rising air traffic demand and the resulting stress on the entire ATM system (acronyms' meaning are listed in Table 1) is costing airlines, passengers, and the overall economy several billions of dollars each year (Airlines for America, 2017). While injecting additional capacity into the ATM system through infrastructure development can relieve the stress placed on the system, over the short run the existing resources can be more efficiently operated by managing the bottleneck operations related to arrivals, departures, runways, and taxiways (Soomer and Franx, 2008; Anagnostakis et al., 2001). Commonly referred to in the literature as the ASP, it entails optimally sequencing and scheduling flight operations on runways which constitute a key bottleneck in the TMA of an airport. Specifically, in the static version of this problem, given a set of aircraft, along with information on the earliest/latest operation time for each aircraft (be it an arrival or a departure), and the minimum safety regulations to protect trailing aircraft from wake vortices generated by the leading aircraft, the objective is to determine a sequence (that optimizes a predefined objective) while simultaneously achieving safety, efficiency, and equity in the ATM system. Passenger safety is achieved by maintaining the required separation between aircraft; runway efficiency is equivalent to achieving low average delay or high throughput; and airline equity is modeled by implementing a CPS strategy wherein an aircraft cannot be shifted by more than *k* positions (the so-called MPS parameter) from its initial FCFS-based position.

[☆] This article belongs to the Virtual Special Issue on "AI in ATM".

* Corresponding author.

E-mail address: piplani@pmail.ntu.edu.sg (R. Piplani).

https://doi.org/10.1016/j.trc.2018.07.031
Received 31 January 2018; Received in revised form 7 July 2018; Accepted 28 July 2018

Table 1
Table of Acronyms.

Acronym	Meaning
ATM	Air Traffic Management
ASP	Aircraft Sequencing Problem
TMA	Terminal Manoeuvring Area
CPS	Constrained Position Shifting
MPS	Maximum Position Shifting
FCFS	First-Come-First-Served
DP	Dynamic Programming
ATSP	Asymmetric Traveling Salesman Problem
MIP	Mixed Integer Program
DS-ASP	Data-Splitting Algorithm for ASP
FAA	Federal Aviation Administration
ELW	Earliest Landing time Windows

In literature there exist many instances of the ASP (Ghoniem et al., 2014; Beasley et al., 2000; Ghoniem et al., 2015; Briskorn and Stolletz, 2014; Lieder et al., 2015; Benlic et al., 2016; Sam et al., 2017; Sölveling and Clarke, 2014; Ahmed et al., 2017) which vary depending upon the number of runways (single or multiple), mode of runway operations (segregated or mixed), problem objective (minimizing delay, maximizing throughput, or minimizing cost), constraints, such as inclusion of time windows, width of time-windows (narrow or wide), permissibility of early landings, CPS inclusion, and solution methods (exact or heuristics). An extensive literature survey on the problem ASP can be found in Prakash et al. (2017) and Bennell et al. (2011). In this work, which considers the static ASP on a single runway, we formulate and *exactly* solve a particular version of ASP defined in Balakrishnan and Chandran (2010) as that is the only work to incorporate all practical constraints including CPS, wide time-windows, precedence constraints, and early landings/departures. Further, they also consider the very important objective function of maximizing throughput (or equivalently minimizing the makespan). Being a potential candidate for deployment in arrival-departure management systems, this problem has great practical implications. Nonetheless being equivalent to NP-hard ATSP (Psaraftis, 1980), obtaining optimal solutions in real-time of large-scale realistic instances opens it for the investigation.

However, exploiting one or the other problem characteristics of the ASP, *dynamic programming* provides an efficient solution for a specific set of scenarios. Assuming that the aircraft from the same category are identical, all aircraft are ready to land together, and ignoring the time-window considerations, Psaraftis (1980) provides the first efficient dynamic programming implementation building on the fact that the number of aircraft categories is limited. His algorithm is exponential in number of categories and polynomial in the number of aircraft and can provide a *tractable* solution with or without CPS requirements, under the aforementioned scenario. However, considering that the above formulation cannot accommodate time-windows and precedence restrictions—the two important practical constraints—another DP algorithm is offered by Balakrishnan and Chandran (2010) whose implementation is exponential in k (MPS parameter) and polynomial in n (instance size). Even though their formulation includes most of the practical constraints and is *tractable* under the segregated mode of operations for values of k up to 3, it may face the curse of dimensionality in mixed-mode of operations as violation of triangular inequality for some of the aircraft triplets requires time to be included in the state-space. Furthermore, increasing the value of k above 3 renders it intractable. In addition, of all possible optima, DP approach cannot provide a specific minimum makespan schedule that leads to associated minimum delay[1]; this requires total delay to be included in the objective function which renders DP computationally prohibitive. Table 2 compares the total delay (averaged over ten randomly generated instances of size 30) for the optimal schedule when maximizing throughput is the objective, where the first column represents the delay when total delay is part of the objective function, and second column represents the delay when it is not; notably, without any loss of makespan, around 2 min of delay is saved in the former case. The above factors motivate the development of a *tractable* approach which is more *flexible*. Note that tractability here refers to the ability to solve problems of realistic size in solution times that are appropriate for the given application—not usual polynomial solvability; whereas flexibility refers to tractability over different problem scenarios.

Next, we revisit the formulation of this version of ASP as a 0–1 MIP, an adaptation of Beasley et al. (2000)'s model, enhanced with the inclusion of the CPS constraints. Recognizing that a more flexible approach over current state-of-the-art is needed, this paper proposes an extension to *data-splitting algorithm* (**DS-ASP**) originally proposed in Prakash et al. (2017), that optimizes the flight sequences by a repeated application of the 0–1 MIP on smaller data sets while ensuring the global optimality, and demonstrate the efficacy of the approach. (See Fig. 1 for the schematic representation of DS-ASP.) Another important and intrinsic feature of the proposed data-splitting framework is that it is pleasingly parallelizable thereby making it amenable to rapidly scalable computations. Note that DS-ASP uses LP-based branch-and-cut approach to solve MIPs.

With the aim to reduce the solution search space, DS-ASP algorithm is based on splitting the given flight data-set into several pairs of leading and following sets and independently optimizing these pairs of sets, while preserving the global optimality. In the process of ensuring global optimality, we have to capture the effect of optimization of the leading set onto the following set, which is not straightforward. The intelligence to do this is developed from analysis of the optimal solution of several instances of leading and

---

[1] There can be several optima with the same optimal makespan but different values of delay.