EFFICIENT AND EQUITABLE DEPARTURE SCHEDULING IN REAL-TIME: NEW APPROACHES TO OLD PROBLEMS

Hamsa Balakrishnan, Massachusetts Institute of Technology, Cambridge, MA. hamsa@mit.edu
Bala Chandran, University of California, Berkeley, CA. chandran@ieor.berkeley.edu

***Abstract*–** The efficient scheduling of departure runways is an important part of surface operations planning, with the goal of increasing the throughput of airports. Departure scheduling is a complex problem that needs to address the needs of diverse stakeholders including the airport operators, air traffic control and the airlines. The challenge lies in optimizing different objective functions such as maximizing departure throughput, minimizing average delay, and ensuring fairness among the airlines, while simultaneously enforcing wake-vortex separation minima, safely accommodating active runway crossings by arrival aircraft, and complying with downstream flow constraints imposed by the terminal airspace, in a dynamic and uncertain environment. This paper presents a new class of techniques based on dynamic programming that can determine, in real-time, efficient departure schedules that satisfy the various upstream and downstream constraints imposed on the departure runway system, thereby providing a valuable asset to departure management.

1 Introduction

The safe and efficient planning of terminal-area operations, particularly airport surface operations, is essential for meeting the expected increase in demand in the Next Generation Air Transportation System (NGATS) without incurring unacceptably large delays. This has motivated several initiatives, both in the United States and in Europe, for the enhancement of terminal-area capacities (TACEC [1], TARMAC [2]). An important aspect of terminal-area operations is the scheduling of departure operations, which is termed departure management.

The runway system has been identified as the primary bottleneck in the departure process, primarily because of the different constraints imposed on runway operations [3]. These constraints include wake-vortex separation requirements that fundamentally constrain the capacity of a runway, the scheduled demand at peak hours which lead to congestion on the surface and large taxi times, controller workload constraints, active runway crossings, and flow restrictions in the airspace downstream of the runway. These downstream flow restrictions are used by air traffic controllers to merge different streams of aircraft through metering at departure fixes, and also by traffic management initiatives such as Ground Delay Programs (GDPs) at destination airports. Downstream constraints must be considered while scheduling departures since they represent the critical interface between the airspace and the airportal systems. Such constraints gain further importance in super-density multi-airport terminal-areas, where operations at airports that share departure fixes are coupled through downstream constraints on the departure runways [4].

The terminal-area is a dynamic and uncertain environment, with constant updates to aircraft states being obtained from surveillance systems and airline reports [5]. The dynamic nature of the terminal-area necessitates the development of scheduling algorithms that are computationally efficient, and therefore amenable to replanning when new events occur or new data updates are obtained. The challenge of departure scheduling lies in simultaneously achieving safety, efficiency, and equity, which are often competing objectives [6, 7, 8], and doing so in a reasonable amount of time. While there is broad consensus on what constitutes safety (wake-vortex avoidance, downstream metering constraints), efficiency (high throughput, low average delay), and equity (limited deviation from the nominal order), as well as decades of research, no solution approach has been able to adequately model and optimally solve the departure planning problem in a computationally tractable manner. One reason for this computational hurdle is that most runway scheduling models are, from a theoretical perspective, inherently hard to solve [9]. Consequently, most practical implementations resort to heuristic or approximate approaches that produce "good" solutions in a short time [6, 10, 11]. The difficulty in solving these scheduling models arises primarily because the solution space allows for the optimal sequence to deviate arbitrarily from the nominal sequence.

However, Dear [12] recognized that, in the short term, it is often unrealistic to allow large deviations from the nominal sequence for two reasons: (i) the system may afford controllers limited flexibility in reordering aircraft, and (ii) large deviations from a nominal or "priority" schedule may be unacceptable to airlines from a fairness standpoint. This observation led to the Constrained Position Shifting (CPS) framework for scheduling aircraft, which

stipulates that an aircraft may be moved up to a specified maximum number of positions from its FCFS order. For example, if the maximum position shift allowed were 2, an aircraft that is in the 8^{th} position in the FCFS order can be placed at the 6^{th}, 7^{th}, 8^{th}, 9^{th}, or 10^{th} position in the new order. Several researchers in both the United States and Europe have since used CPS to model fairness, and worked toward developing fast solution techniques for scheduling within the CPS framework [13, 14, 7]. While some variants of CPS were shown to be solvable in polynomial time [15, 16], they were unable to handle all the operational constraints that arise in practice [17]. More importantly, these methods lack a unifying theory that allows their results to generalize to other interesting scheduling problems under CPS, even resulting in a conjecture that in general, scheduling under CPS may have exponential complexity [7].

This paper presents an overview of new breakthroughs in techniques for solving departure scheduling problems within the CPS framework, while accounting for various operational constraints. We propose efficient (polynomial time) algorithms for several variants of the problem including balancing departure runway operations and departure scheduling in the presence of active runway crossings. We then extend these ideas to develop algorithms that generate schedules that are robust to system uncertainties, and can quantify the tradeoff between efficiency and robustness. We describe a prototype implementation which demonstrates that these algorithms are fast enough to be used for real-time departure management.

2 Problem definition

The objective of departure scheduling is to help controllers determine effective departure sequences and the optimal take-off times. The optimal or efficient departure schedule is one that optimizes one of several possible objectives, the most important of which are the runway throughput and the average delay incurred by departing aircraft. Other objectives include ensuring equity in the departure sequence, and incorporating airline preferences. Departure runway operations need to be optimized while accommodating the various constraints imposed on the system, such as spacing and sequencing requirements, simultaneous runway operations, and downstream flow constraints such as traffic management initiatives.

2.1 Minimum spacing requirements

An aircraft faces the risk of instability if it interacts with the wake-vortex of the aircraft taking off in front of it. To prevent this, the The Federal Aviation Administration (FAA) mandates minimum spacing requirements between departing aircraft, which depends on the size of the leading and trailing aircraft. We define the minimum time-separation matrix by Δ, where the element δ_{ab} is the minimum required time between takeoffs, if the leading aircraft belongs to class a, and the trailing aircraft belongs to class b.

The FAA divides aircraft into three weight classes (Heavy, Large and Small) based on the maximum takeoff weight capacity [18]. The Boeing 757 is treated similar to a Heavy for wake avoidance. Unlike arrivals, where wake vortex separation is the responsibility of the air traffic control only during IFR operations, increased separation is mandated for departure scheduling during both VFR and IFR operations. These separation requirements can be used to determine the minimum time required between consecutive departures [14]. The matrix of minimum time separations (for departures on a single runway) is given in Table 1.

	Trailing Aircraft	
Leading Aircraft	Heavy/B757	Large/Small
Heavy/B757	90	120
Large/Small	60	60

Table 1: Minimum separation (in seconds) between departures [14].

The departure runway schedule must also be capable of satisfying downstream separation requirements, such as Miles-in-Trail (MIT) or Minutes-in-Trail (MINIT) constraints at the departure fixes. For example, a particular departure fix may require a spacing of 20 nm, which would impose separation requirements between two (non-consecutive) departures which are assigned to that fix.

Separation constraints can be further classified as *successive* and *complete* constraints [9]. Successive separations constraints are those which are imposed between consecutive operations at the runway. In contrast, complete constraints are constraints between non-consecutive operations at the runway. Typically, downstream metering constraints are imposed on aircraft assigned to the same departure fix, or same destination, which may not correspond to consecutive runway operations.

We note that the wake vortex separation requirements, shown in Table 1, satisfy the *triangle inequality*, that is

$$\delta_{ik} \leq \delta_{ij} + \delta_{jk}, \forall i, \forall k \neq i, \forall j \neq i, k. \quad (1)$$

In contrast, the separation constraints imposed by downstream requirements need not necessarily satisfy the triangle inequality. For example, typical MIT spacings require inter-departure separations of 5 nm, while a traffic management initiative at one of the departure fixes might enforce a 20 nm spacing. This would imply that there could be two departure operations between consecutive aircraft assigned to that fix, that is, the triangle inequality would not be satisfied. If all separation constraints satisfy

the triangle inequality, then complete constraints would be equivalent to successive constraints.

2.2 Time-window constraints

There are also constraints on the possible departure times of a particular aircraft in the schedule. Constraints of this form could arise because of acceptable levels of delay for an aircraft on the airport surface, but also because of downstream traffic flow management. Time-window constraints can be used to represent constraints such as the Departure Sequencing Program (DSP), Expected Departure Clearance Times (EDCTs) which are used as part of Ground Delay Programs (GDPs) at destination airports, Approval Request (APREQ) procedures [7]. These time-windows could be quite restrictive, about 3 min for a DSP [7] and about 10 min for EDCTs [19]. These constraints impose an earliest and a latest time of departure for the aircraft. The proposed approach can also handle situations in which an aircraft's scheduled departure time could be in one of a number of disjoint time intervals.

2.3 Fairness: position shift constraints

Since the airlines are a major stakeholder in the air transportation system, it is important that an increase in efficiency is not achieved at the expense of an equitable allocation of resources. This could happen if an aircraft that would have had an early departure in the first-come-first serve (FCFS) sequence is rescheduled to depart last, thereby incurring a disproportionate amount of delay. CPS ensures some degree of fairness since it does not allow the final schedule to deviate significantly from the FCFS schedule. The maximum number of position shifts allowed under CPS is denoted k, and the resultant scenario is referred to as a k-CPS scenario. There is clearly a tradeoff between the value of k and the level of fairness – typical values of k for both arrival and departure scheduling are between 1 and 3 [14, 20].

Carr [7] states that for departure scheduling, it is often necessary to consider asymmetric CPS, where the number of forward shifts allowed is different from the number of backward shifts allowed. This is to accommodate mixed operations (departures and arrivals) on a runway, when the departures maybe allowed a larger number of backward shifts than forward ones. He notes that most prior research on CPS algorithms have focussed on the symmetric case [12, 13, 16]; the techniques described in this paper are applicable to both symmetric and asymmetric position shift constraints.

2.4 Precedence constraints

Finally, we consider precedence constraints on the departure sequence. A source of such constraints is the airlines themselves, who have precedence constraints due to banking operations, or priority flights. Precedence constraints can also represent the restricted freedom available to taxiing aircraft which are not allowed to overtake each other [7]. Precedence relations can be represented by a matrix $M = \{m_{ij}\}$ such that element $m_{ij} = 1$ if aircraft i *must* land before aircraft j, and $m_{ij} = 0$ otherwise.

3 Basic CPS framework

In this paper, we build on the basic technique that we proposed in our prior research [17], where we addressed the following problem: Given n aircraft indexed $1, \cdots, n$, earliest and latest departure times $e(i)$ and $\ell(i)$ for each aircraft i, separation matrix Δ, precedence matrix M, and the maximum number of position shifts k, compute the k-CPS sequence and corresponding departure times, t_i that minimize the makespan of the sequence (the departure time of the last aircraft in the sequence). Since we schedule aircraft in batches as their estimates of their earliest pushback times become available, minimizing the makespan is equivalent to maximizing the runway throughput. For simplicity, we assume that the aircraft are labeled $(1, 2, \cdots, n)$, according to their position in the FCFS sequence. We demonstrated that every k-CPS sequence can be represented as a path in a directed graph whose size is polynomially bounded in n and k. We now briefly describe the structure of this network and its properties.

The network consists of n *stages* $\{1, \cdots, n\}$, where each stage corresponds to an aircraft position in the final sequence. A node in stage p of the network represents a subsequence of aircraft of length $\min\{2k+1, p\}$ where k is the maximum position shift[1] For example, for $n = 6$ and $k = 1$, the nodes in stages $3, \cdots, 6$ represent all possible sequences of length $2k+1 = 3$ ending at that stage. Stage 2 contains a node for every possible aircraft sequence of length 2 ending at position 2, while stage 1 contains a node for every possible sequence of length 1 starting at position 1. For convenience, we refer to the last aircraft in a node's sequence as the *final aircraft* of that node.

The network for $n = 6$ and $k = 1$ is shown in Figure 1. For each node in stage p, we draw directed arcs to all the nodes in stage $p+1$ that can follow it. For example, a sequence (1–2–3) in stage 3 can be followed by the sequences (2–3–4) or (2–3–5) in stage 4. This results in a network where every directed path from a node in

[1] This network can easily be extended to asymmetric CPS as follows: Let f be the maximum number of forward position shifts allowed, and b the maximum number of backward shifts. Then, a node in stage p of the network would represent a subsequence of aircraft of length $\min\{f+b+1, p\}$. The rest of the network generation procedure would be similar to the symmetric case. This logic can also be extended to scenarios where the CPS constraint depends on the particular aircraft.

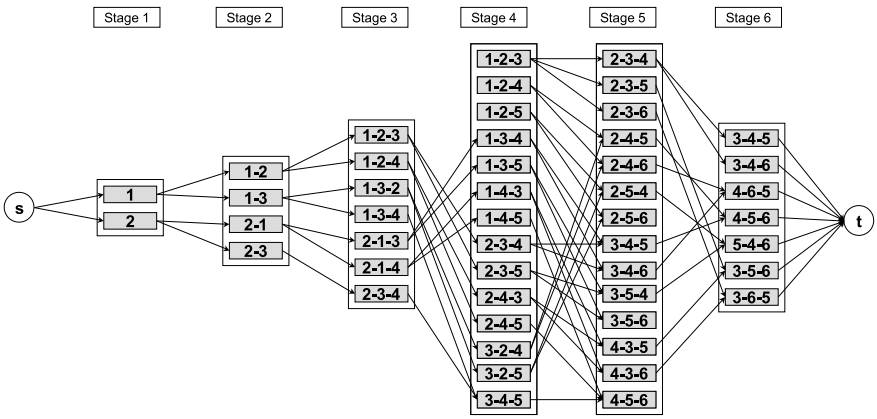

Figure 1: Network for $n = 6$, $k = 1$.

stage 1 to one in stage n represents a possible k-CPS sequence. For example, the path (2) → (2–1) → (2–1–3) → (1–3–4) → (3–4–6) → (4–6–5) represents the sequence 2–1–3–4–6–5. Nodes that violate precedence relationships and those that cannot belong to a path from s to t (such as node (1–2–4) in stage 4 in the figure) are removed from the network to generate a "pruned" network. In practice this pruned network is significantly smaller than the original network. The following properties (derived in [17]) are satisfied by the network, forming the basis of our CPS solution framework:

(i) Every possible k-CPS subsequence of length $2k+1$ or less is contained in some node of the network.

(ii) Every feasible sequence (one that satisfies maximum position shift constraints and precedence constraints) can be represented by a path in the network from a node in stage 1 to a node in stage n.

(iii) Every path in the network from a node in stage 1 to a node in stage n represents a feasible k-CPS sequence.

3.1 Dynamic programming recursion

Given two nodes i and j, the arc connecting them (if it exists) is denoted by (i, j). Let $e(i)$ denote the earliest time that the sequence of node i can begin, which is the earliest departure time of the final aircraft of node i. Each arc (i, j) in the network is associated with a "distance" δ_{ij}, which is the minimum separation between the final aircraft of node i and that of node j, if they were to takeoff consecutively and in that order. This separation is determined by the weight classes of the two final aircraft. Arcs that lead into the sink and out of the source have zero distance associated with them. We define:

$\ell_1(i)$ Departure time of the last aircraft of node i.

$\ell_2(i)$ Departure time of the second-from-last aircraft of node i.

$e_1(i)$ Earliest possible departure time of the last aircraft of node i.

$e_2(i)$ Earliest possible departure time of the second-from-last aircraft of node i.

$\delta_{21}(i)$ Minimum separation between the second-from-last and the last aircraft of node i.

$P(i)$ Set of nodes that are predecessors of node i.

We wish to find $\ell_1(t)$, the earliest time that the entire sequence can be completed, which is equal to the makespan. The values of $\ell_1(\cdot)$ can be computed by the following dynamic programming recursion.

$$\ell_2(i) = \min_{j \in P(i)} \ell_1(j);\ \ell_1(i) = \max\{\ell_2(i) + \delta_{21}(i), e_1(i)\} \quad (2)$$

The recursion is solved using the boundary condition $\ell_1(\cdot) = e(\cdot)$ for all nodes in stage 1.

3.1.1 Complexity

The complexity of the algorithm for finding the minimum makespan for n aircraft and maximum position shift of k is $O(n(2k+1)^{(2k+2)})$ [17]. While it is exponential in k, it is of little consequence, since k is typically small (at most 3 in practice) [14]. The linear growth in n is useful since increasing the number of aircraft does not pose much of a computational burden.

3.1.2 Monte Carlo simulations

We wish to estimate the benefit from CPS over the FCFS sequence. Consider the following example.

Example 1 (Basic departure scheduling:)

We are given 6 aircraft, with no precedence constraints and a nominal FCFS order such that they all have the same earliest departure time ($t = 0$ sec) and latest departure time ($t = 600$ sec). Let their corresponding weight classes be H, S, H, S, L, and L, and the CPS parameter $k = 1$. The makespan of the FCFS schedule would be 420 sec, with the departure times being at 0, 120, 180, 300, 360 and 420 sec respectively. However, the optimal departure sequence would be 2–1–3–4–5–6, with a makespan of 390 sec, an improvement in throughput of 30 sec, or 7%.

We compute the average improvement in the throughput (makespan) of CPS over that of FCFS, for different rates of departure operations, using 1000-trial Monte Carlo simulations. We consider sequences of aircraft generated using Poisson processes at different rates, corresponding to different levels of demand for the departure runway. The type of aircraft is determined from among S, L and H using the specified mix. We then enforce the minimum wake-vortex separations between the departures in the FCFS schedule, if necessary. The time-windows for departures are assumed to be 10-min long, and there are no precedence constraints. The capacity of a single runway used for departures for a traffic mix of 20% Small, 40% Large and 40% Heavy is estimated to be 45 aircraft/hour assuming FCFS sequencing, as is done traditionally [14]. For a fleet mix of 20% Small, 30% Large and 50% Heavy, the capacity is estimated to be 43 aircraft/hour. The results for different numbers of maximum position shifts, $k = 1, 2, 3$ are shown in Figure 2. For an operating departure rate of 43-45 aircraft/hour, 3-CPS yields an increase in the throughput of about 3% *per* runway, when compared to FCFS. In congested terminal-areas, throughput increases of even a few aircraft an hour are desirable, because of the associated delay benefits. For example, for departure sequences generated at 45 aircraft/hour, the average delay savings compared to FCFS are about 15%, 23% and 25% for $k = 1$, 2 and 3 respectively. We note that while the objective here was to maximize the throughput, there are also benefits in terms of reduction of average delay. Similar results have been observed for arrival scheduling [17].

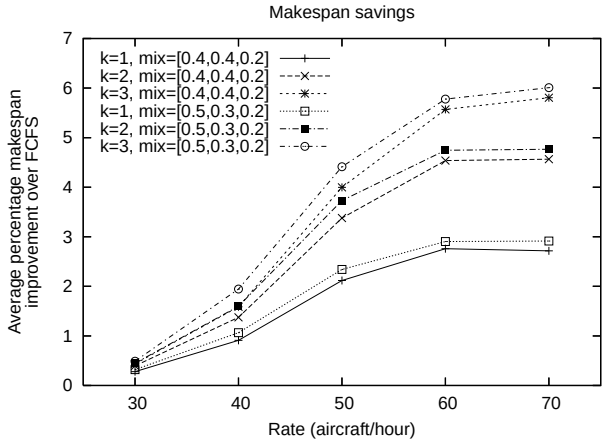

Figure 2: Average throughput improvement over FCFS for basic departure scheduling with 10-min departure time-windows.

We also use the Monte Carlo simulations to gauge if the technique is amenable to real-time applications, where the computation of schedules for a batch of aircraft (with a 30-min or 1-hour horizon, say) must be completed in less than a few minutes. Table 2 shows the average computation times (on a Mac with a 2.33 GHz Intel Core 2 Duo CPU and 2 GB of RAM) for 2- and 3-CPS for the simulations described above. 1-CPS takes less than 0.01 sec for all cases considered. We find that for as many as 70 aircraft, the computation takes about 10 sec for 3-CPS, making the technique suitable for real-time scheduling. We also note that, as expected, the computation time scales linearly with the number of aircraft n (10 aircraft add about 0.01 seconds to the 2-CPS runtime, and about 2 seconds to the 3-CPS runtime).

Num. aircraft	30	40	50	60	70
2-CPS	0.01	0.02	0.02	0.03	0.04
3-CPS	3.12	4.64	6.55	8.37	10.19

Table 2: Average computation time (in seconds) for 2-CPS and 3-CPS for basic departure scheduling.

4 Departure scheduling under CPS

Departure scheduling presents several variations of the basic CPS problem that we can solve by appropriately modifying the basic CPS network and DP recursion, such as: the objective of minimizing average delay or a weighted sum of throughput and average delay, multiple-runway scheduling, scheduling with complete (as opposed to successive) spacing constraints, robust scheduling, and scheduling simultaneous runway operations (active runway crossings).

4.1 Minimizing average delay

While we have so far considered the objective of maximizing the throughput of the runway, it may also be desirable to minimize the average delay incurred by aircraft. This can be achieved using a modification of the basic CPS network described in Section 3, with a moderate increase in computational complexity.

To generate the network for minimizing average delay, we begin with the basic CPS network. Each node at stage p of the original network corresponds to a cluster of $n-p+1$ nodes, such that the connectivity of the original network is maintained. The nodes in a cluster in stage p are associated with the function $t_1+t_2+\cdots+t_{p-1}+jt_p$, where $j = 1, \cdots, n-p+1$. This is shown for the sequence 2–1–3–4–6–5 in Figure 3. Corresponding to every predecessor of a node in the original network, there are now two predecessors. We would like to minimize the function $t_1+t_2+\cdots+t_n$ over all nodes in stage n, where t_i is the departure time of the ith aircraft in the final (optimal) sequence.

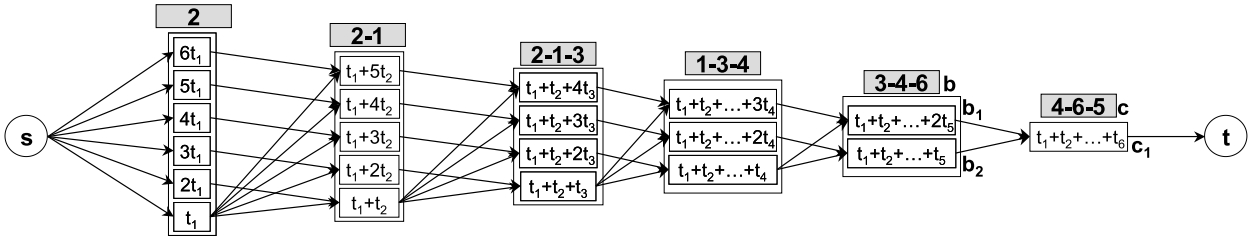

Figure 3: Representation of a path in the average delay network for $n = 6$ and $k = 1$.

Consider a node (denoted c_1) in stage n of the average-delay network, part of a cluster corresponding to a node c in the original network. We denote one of its predecessors in the original network b, and the two corresponding predecessors in the average-delay network b_1 and b_2. We wish to minimize the function

$$f(c_1) = t_1 + t_2 + \cdots + t_n.$$

Then, it is evident that either $t_n = t_{n-1} + \delta_{bc}$ or $t_n = e_1(c)$. Looking at the structure of the average-delay network and the definition of nodes in a cluster, the former condition means that $f(c_1) = f(b_1) + \delta_{bc}$ and the latter means that $f(c_1) = f(b_2) + e_1(c)$. In other words,

$$f(c_1) = \min_{b \in P(c)} \left(\max\{f(b_1) + \delta_{bc}, f(b_2) + e_1(c)\}\right).$$

This can be written recursively for all the previous stages. The recursion is solved by imposing the boundary condition that $t(\cdot) = e(\cdot)$ for all nodes in stage 1.

The upper bound on the number of nodes in the average-delay network is n times the number of nodes in the original network. The number of arcs is at most twice the number of nodes. Using this network, we can compute the complexity of a dynamic programming algorithm that minimizes the average delay incurred by all the aircraft.

4.1.1 Complexity

The complexity of scheduling runway operations with time-window constraints, k-CPS, and minimum separation requirements, with the objective of minimizing the average delay of the aircraft is $O(n^2(2k+1)^{(2k+2)})$.

It can be shown quite easily that the average-delay network can be slightly modified to minimize a weighted sum of average delay and throughput (as attempted in [11] using a heuristic) with no further change in complexity, by defining the function $f(c_1) = t_1 + t_2 + \cdots + \left[1 + \frac{w_{throughput}}{w_{delay}}\right] t_n$, and proceeding as before.

4.2 Robust departure scheduling

The objective of achieving an efficient departure schedule is further complicated by the presence of uncertainty. The sources of uncertainty include the variability in the pushback times as well as the times that aircraft take to taxi from their gates to the runway departure queue. In order to increase the predictability of the departure process, there has been much research into modeling and quantifying the uncertainties in airport surface operations [7, 21, 22]. The presence of uncertainty in the system motivates the development of robust schedules for runway operations. The notion of robustness is one that can be defined in several ways. In the context of departure scheduling, the uncertainty in the system could result in the aircraft violating important safety constraints such as wake separation minima, thereby necessitating rescheduling on the part of the controllers; therefore, we consider a runway sequence robust if there is a high probability that a controller does not have to intervene once the schedule has been determined.

In prior work [23], we presented a technique to determine robust departure schedules that can potentially improve runway productivity, while still satisfying the various constraints that we have described above. We also showed that the CPS framework can be used to develop robust schedules in a computationally efficient manner, with complexity that scales linearly with the number of aircraft, and as the cube of the largest difference between the latest and earliest arrival time over all aircraft. In contrast to the methods described in the previous sections, the output of the robust variant is not a single schedule, but a tradeoff between the probability of controller intervention (reliability) and the time to complete runway operations for the given set of aircraft (makespan of the sequence). Using extensions of the networks developed for the deterministic scenarios, the robust variant of CPS gives system designers the ability to select the appropriate threshold which determines the tradeoff between robustness and efficiency. In addition to scheduling, the robust CPS framework can be used to decide broader policy issues such as the benefit (in terms of throughput and safety) of introducing on-board or ground-based systems to decrease the uncertainty in the system.

4.3 Complete separation requirements

Complete separation requirements are spacing constraints that need to be satisfied between departure operations, irrespective of whether or not they are consecutive operations. As we saw in Section 2.1, if all separation re-

quirements satisfy the triangle inequality, we need to only consider consecutive operations, as in the case of successive separation requirements. However, if the triangle inequaity is violated, then for each aircraft we need to keep track of more than just the last predecessor in order to ensure that separation is maintained. We denote the number of predecessors we need to keep track of until separation is guaranteed as λ, and the scenario as a λ-look-ahead policy. If the triangle inequality is satisfied, $\lambda = 1$. We can define $\ell_1(i)$, $\ell_2(i)$, $e_1(i)$, $e_2(i)$, $\delta_{21}(i)$ and $P(i)$ for any node i as we have done for Equation 2. Then, our objective is to minimize $\ell(i)$ for each node i in stage n. If the triangle inequality is satisfied, then we need a 1-look-ahead policy, and the recursion is given by Equation 2. This is implicitly the recursion that we proposed in our prior work [17].

We consider a discrete-time network, as described by the authors in [23]. The required output accuracy is denoted ε, and L denotes the largest difference between the earliest and latest arrival times over all aircraft, that is, $L = \max_i\{\ell(i) - e(i)\}$. Then, as in the case of robust scheduling described above, the schedule that minimizes the completion time of the sequence can be determined with complexity that scales linearly with the number of aircraft, and as a power of (L/ε).

4.3.1 Complexity

The number of nodes in the network is $O\left(n(2k+1)^{\max\{2k+1,\lambda+1\}}\right)$. The number of arcs is the number of nodes multiplied by $2k+1$. The work done per arc is $O((L/\varepsilon)^{(\lambda+1)})$. Therefore, the computational complexity of requiring a λ-look-ahead policy is $O\left(n(2k+1)^{\max\{2k+2,\lambda+2\}}(L/\varepsilon)^{(\lambda+1)}\right)$.

We note that λ is also typically small – for the case of metering at departure fixes with 20 nm MIT restrictions instead of the usual 5 nm MIT requirement, the time-based spacing is 218 sec, based on a ground-speed of 330 knots at the departure fix [24]. This implies that there can be up to 3 other departures in between two metered aircraft, thereby requiring a 4-look-ahead policy.

4.4 Multiple runways

We now consider the problem of scheduling and sequencing departures on multiple parallel runways. Even when there are multiple parallel runways being used simultaneously for takeoffs at an airport, operations on the runways are not necessarily independent of each other. Under current regulations, if the runway centerlines are less than 2500 ft apart, the separation requirements are the same as the inter-departure separation for the single runway case (Table 1). We note that the FAA has identified closely spaced parallel runway (CSPR) departures as a promising weather-dependent solution for reducing wake separation. The principle behind CSPR departures is that if the crosswind is such that the wake will not travel upwind of a Heavy aircraft into the adjacent runway, the imposed wait on the adjacent runway could be eliminated. CSPR solutions attempt to take advantage of the accuracy of short-horizon forecasts of crosswinds [25], and will therefore require the ability to plan and replan parallel runway departure sequences tactically based on wind predictions.

There are two variants of the multiple runway scheduling problem, depending on whether or not we wish to assign runways to the departing flights, in addition to determining the optimal departure sequence and takeoff times for operations on each runway.

4.4.1 Preassigned runways

We begin with scenarios in which runways are preassigned considering factors such as gate location and flight destination. Clearly, if the FCFS order and CPS constraints are defined separately for each runway, the two runways can be scheduled completely independent of each other. However, if the FCFS order involves operations on both runways (this could happen because of the airport layout: if aircraft share (compete for) the same taxiway segments or intersections to reach their respective runways, the FCFS order, CPS and precedence constraints would have to be be defined on both runways), the schedules will be coupled. The parallel runway sequencing problem is then a special case of complete spacing requirements in which the triangle inequality is violated (if the runways are independent, there are no separation requirements between departures on different runways). For two runways, if operations on each runway satisfy the triangle inequality, we require a $\left(2\lceil\frac{\delta_{\max}}{\delta_{\min}}\rceil - 1\right)$-look-ahead policy, where $\delta_{\max}$ and $\delta_{\min}$ are the maximum and minimum separation requirements ($\lambda = 3$ for Table 1).

4.4.2 Runway balancing

Runway assignments are not always uniquely determined by departure fix or gate assignments – in such scenarios, departure management would involve not only scheduling the takeoff times and the departure sequence, but also the optimal runway assignments. Keeping the runways balanced, that is, maintaining load on both parallel runways is essential for efficient runway utilization. As we have seen, the inter-arrival spacing could constrained, even in the case of parallel runways. In order to schedule multiple runways and balance the load among the runways to maximize throughput, we construct a network with multiple copies of each node in the original network to account for all possible runway assignments. For example, if we had a node corresponding

to the subsequence (a,b,c,d,e) and we want to consider 2 runways, we replace it by 2^5 nodes (a_1,b_1,c_1,d_1,e_1), (a_1,b_1,c_1,d_1,e_2), (a_1,b_1,c_1,d_2,e_1), (a_1,b_1,c_1,d_2,e_2), ..., (a_2,b_2,c_2,d_2,e_2), where the subscript indicates the runway assigned to each aircraft. Thus, for each node in the original network, we would have r^s nodes, where r is the number of runways and s is length of each node sub-sequence. For each arc, we would have r^{s+1} copies, so the complexity would be r^{s+1} times the complexity of scheduling with preassigned runways.

4.5 Scheduling active runway crossings

The typical layout of taxiways at airports imply that some aircraft (arrivals) need to cross a runway that is being used to reach their assigned gates. Such crossings, known as active runway crossings, require the coordination of runway schedules and taxi schedules. In the current system, for reasons of safety, active runway crossings are the responsibility of the local controller who is coordinating operations on the departure runway, and can lead to a significant increase in the controller workload as well as taxi delays [26, 7]. It is desirable to identify gaps in the departure sequence to allow aircraft to cross active runways, and local controllers try to utilize the relatively large wake-vortex separation requirement following a Heavy aircraft to schedule runway crossings. Due to the negligible space for overtaking on the taxiways, runway crossing queues are always processed FCFS.

At current traffic levels, there are sufficient gaps in the departure schedules at most airports to allow for active runway crossings without disrupting the departure schedule. However, studies have shown that as the traffic levels increase and runways operate close to their capacity, scheduling departure schedules independent of runway crossings could result in a substantial increase in runway crossing queue times and the resulting taxi delays [27]. This motivates the development of scheduling algorithms that accommodate active runway crossings into the runway schedule, by possibly altering the spacing in the departure schedule. This would also help keep runway crossing queue times to within acceptable levels (under 3 min by current standards [28]). Such algorithms would need to handle time-window constraints for both departures and runway crossings, and process the departure queue under CPS constraints in coordination with the runway crossing queues in which an aircraft would be processed FCFS relative to the other aircraft in the same runway crossing queue.

In many airports, there is more than one spot at which aircraft can cross a runway. Controllers can therefore simultaneously process multiple runway crossing queues. An additional feature of runway crossings is the *acceleration-delay penalty* that is incurred by the first crossing aircraft that accelerates from a standstill to the taxi speed. If there are several crossing aircraft in the same queue that accelerate simultaneously, the rest of the queue does not incur the same penalty [7]. Observations at Boston Logan airport indicate that due to the acceleration-delay penalty, the runway crossing time of the first aircraft is about 1.7 times that of the aircraft that follow it [29, 7]. In addition, aircraft are kept at least 10 sec in trail while crossing active runways. A schematic of the runway layout, showing the departure queues and runway crossing queues, along with the crossing times, is depicted in Figure 4.

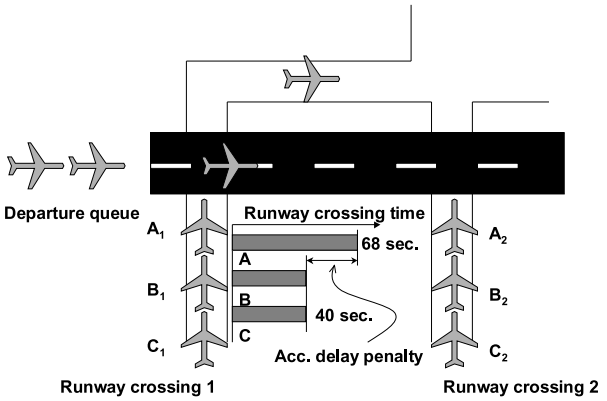

Figure 4: Schematic of a departure runway with two active runway crossing queues.

The dynamic programming approach described in Section 3 can be extended to scheduling departures with a single active runway crossing queue of length n_1 by duplicating the basic CPS network n_1+1 times (thus creating n_1+1 *levels* going from 0 through n) and adding arcs in between the levels depending on the number of aircraft that need to have crossed the runway by that stage in the sequence. As before, for any node, $T(i)$ is the earliest time that the sequence corresponding to node i can possibly begin, in a sequence starting at node s and ending in node i. However, in this case, if node i is in level q, then $T(i)$ is the earliest time that the sequence corresponding to node i can possibly begin, *given that* q aircraft have crossed the runway. The nodes in the last stage of the last level have arcs leading to the sink t. The corresponding network for the instance of $n=6$, $k=1$, $n_1=2$ is shown in Figure 5.

Similarly, when there are multiple (say, c) runway crossings each of length at most n_1, we need to create $O(n_1^c)$ copies of the basic CPS network; and for a node i corresponding to the "level" $q_1 q_2 \cdots q_c$, $T(i)$ is the earliest time that the sequence corresponding to node i can possibly begin, given that q_1 aircraft from the first runway crossing queue, q_2 aircraft from the first runway crossing queue have crossed the runway, and so on.

4.5.1 Complexity

The complexity of maximizing the runway throughput with a single FCFS runway crossing queue of length n_1, time-window constraints, CPS parameter k (for the departure sequence), and minimum separation requirements is $O(nn_1^3(2k+1)^{(2k+2)})$, where the departure sequence contains n aircraft. Analogously, the complexity of maximizing the throughput of departure runway operations with c different active runway crossing queues of maximum length n_1 is $O(cnn_1^{2c+1}(2k+1)^{(2k+2)})$.

Since the number of active runway crossings in a single runway is generally small (≤ 5), and taxiway geometries necessitate small buffers (that is, small n_1), the problem remains tractable for practical scenarios.

Example 2 (Scheduling with runway crossings:)

We consider an extension of previous example, with $n = 6$, $k = 1$, and weight classes H, S, H, S, L, and L. Let there be a single runway crossing queue with two aircraft (denoted A and B) that arrive at the runway at 160 sec and 200 sec respectively. The requirement that the wait time for crossing a runway be under 3 min implies that A must cross between 160 and 340 sec and B must cross between 200 and 380 sec. If both aircraft cross at the same time, A takes 68 sec to cross while B takes 40 sec to cross; on the other hand, if they cross at different times (that is, there is at least one departure in between the crossings), they take 68 sec each. The runway occupancy times for departures are 55 sec each [27].

We have seen earlier that the optimal schedule without runway crossings is

Aircraft	2	1	3	4	5	6
Time (s)	0	60	150	270	330	390

We note that for an aircraft to cross the runway, there must be a gap in the departure sequence of $55 + 68 = 123$ sec. The largest gap in the above schedule is 120 sec, which is insufficient for a crossing. Therefore, it is clear that one would have to extend the gaps to accommodate X_1, which cannot wait until the departures have completed at 390 sec. If a naive (almost FCFS) approach were used, we would obtain the schedule

Aircraft	2	1	3	A	4	5	6	B
Time (s)	0	60	150	205	273	333	393	448

which implies that it would take 516 sec for the whole sequence to clear the runway.

However, using the CPS framework described above, we can compute the optimal schedule to be

Aircraft	2	1	3	A	B	4	5	6
Time (s)	0	60	150	205	243	283	343	403

which implies that the schedule would be completed in 458 sec.

5 Conclusion

We have developed a unified framework for runway scheduling under CPS constraints, and demonstrated that most conceivable problems in departure management including enhancing throughput, decreasing delay, ensuring fairness, satisfying downstream metering constraints, runway balancing, robust scheduling, and accommodating active runway crossings can be effectively modeled and solved in polynomial time; our approaches often scale linearly (and sometimes quadratically) in the number of aircraft. The algorithms are easily implemented, and a prototype implementation demonstrates that the run-times are sufficiently small to enable real-time deployment.

References

[1] K. D. Arkind. Requirements for a novel Terminal Area Capacity Enhancement Concept in 2022. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, Providence, RI, 2004.

[2] D. Boehme. Improved airport surface traffic management by planning, problems, concepts and a solution TARMAC. In *Lecture Notes in Control and Information Sciences 198*, 1994.

[3] H. R. Idris, B. Delcaire, I. Anagnostakis, W. D. Hall, N. Pujet, E. Feron, R. J. Hansman, J.-P. Clarke, and A. R. Odoni. Identification of flow constraint and control points in departure operations at airport systems. In *AIAA Guidance, Navigation and Control Conference*, Boston, MA, August 1998. AIAA-1998-4291 .

[4] NASA. Next Generation Air Transportation System (NGATS) Air Traffic Management (ATM)-Airspace Project: Reference Material. External Release Version, June, 2006.

[5] S. Atkins and C. Brinton. Concept description and development plan for the Surface Management System. *Journal of Air Traffic Control*, 44(1), January-March 2002.

[6] D. Böhme. Tactical departure management with the Eurocontrol/ DLR DMAN. In *6th USA/Europe Air Traffic Management R&D Seminar*, Baltimore, MD, 2005.

[7] F. R. Carr. *Robust Decision-Support Tools for Airport Surface Traffic*. PhD thesis, Massachusetts Institute of Technology, 2004.

[8] I. Anagnostakis, H. R. Idris, J. P. Clarke, E. Feron, R. J. Hansman, A. R. Odoni, and W. D. Hall. A conceptual design of a departure planner decision aid. In *3rd USA/Europe Air Traffic Management R&D Seminar*, Napoli, Italy, June 2000.

[9] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson. Scheduling aircraft landings - the static case. *Transportation Science*, 34(2):180–197, 2000.

[10] W. W. Cooper, R. H. Cormier, J. G. Foster, M. J. Mills, and S. C. Mohleji. Use of the Departure Enhanced Planning and Runway/Taxiway Assignment System (DEPARTS) for optimal departure scheduling at busy airports. In *Digital Avionics Systems Conference*, 2002.

[11] I. Anagnostakis, J.-P. Clarke, D. Böhme, and U. Völckers. Runway operations planning and control: Sequencing and scheduling. *Journal of Aircraft*, 38(6), 2001.

[12] R. G. Dear. The dynamic scheduling of aircraft in the near terminal area. MIT Flight Transportation Laboratory Report R76-9, Massachusetts Institute of Technology, September 1976.

[13] R. G. Dear and Y. S. Sherif. An algorithm for computer assisted sequencing and scheduling of terminal area operations. *Transportation Research, Part A, Policy and Practice*, 25:129–139, 1991.

[14] Richard de Neufville and Amadeo Odoni. *Airport Systems: Planning, Design and Management*. McGraw-Hill, 2003.

[15] H. N. Psaraftis. A dynamic programming approach for sequencing groups of identical jobs. *Operations Research*, 28:1347–1359,

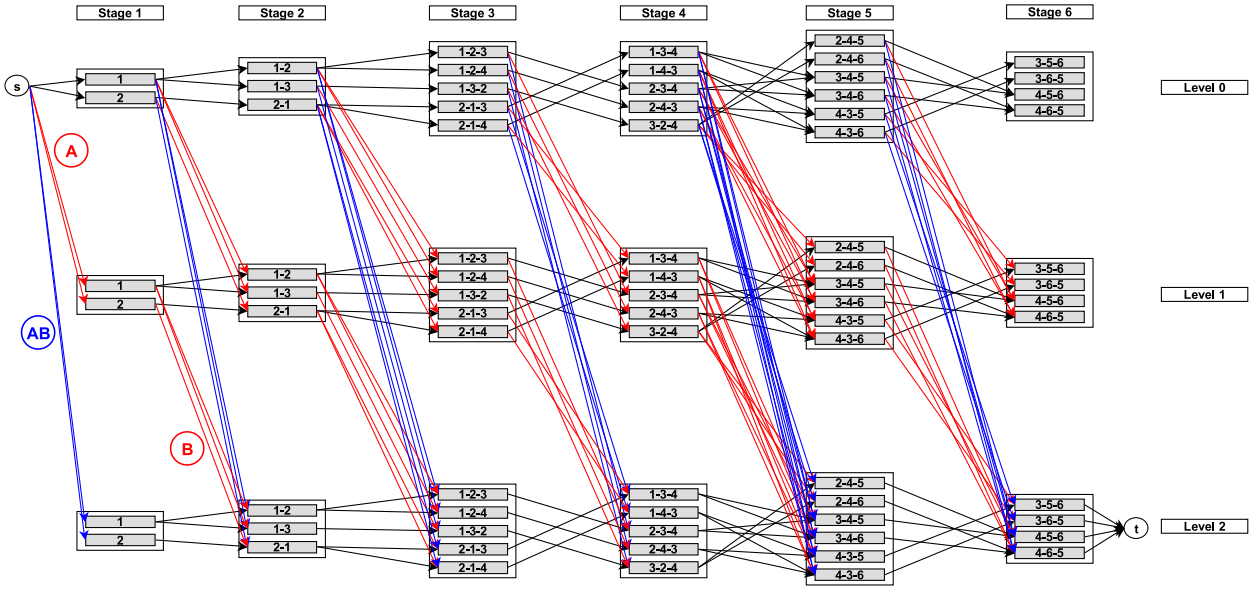

Figure 5: Network for $n = 6$, $k = 1$, $c = 1$, $n_1 = 2$.

1980.

[16] D. A. Trivizas. Optimal scheduling with Maximum Position Shift (MPS) constraints: A runway scheduling application. *Journal of Navigation*, 51(2):250–266, May 1998.

[17] H. Balakrishnan and B. Chandran. Scheduling aircraft landings under Constrained Position Shifting. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, August 2006.

[18] Federal Aviation Administration. Air Traffic Control: Order 7110.65P. Includes Change 1, effective August 3, 2006.

[19] Federal Aviation Administration. Collaborative Decision Making (CDM) Newsletter, February 2006.

[20] H. R. Idris, I. Anagnostakis, B. Delcaire, W. D. Hall, J.-P. Clarke, R. J. Hansman, E. Feron, and A. R. Odoni. Observations of departure processes at Logan airport to support the development of departure planning tools. *Air Traffic Control Quarterly Journal*, 7(4), 1999.

[21] H. R. Idris, J.-P. Clarke, R. Bhuva, and L. Kang. Queuing model for taxi-out time estimation. *Air Traffic Control Quarterly Journal*, 10(1), 2002.

[22] C. Brinton, J. Krozel, B. Capozzi, and S. Atkins. Improved taxi prediction algorithms for the Surface Management System. In *AIAA Guidance, Navigation and Control Conference*, Monterey, CA, 2002.

[23] B. Chandran and H. Balakrishnan. A dynamic programming algorithm for robust runway scheduling. In *American Control Conference*, New York, NY, 2007. To appear.

[24] S. Grabbe and B. Sridhar. Modeling and evaluation of Miles-in-Trail restrictions in the National Air Space. In *AIAA Guidance, Navigation and Control Conference*, Austin, TX, August 2003.

[25] R. E. Cole and S. Winkler. Wind prediction to support reduced wake separation standards for Closely Spaced Parallel Runway departures. In *AIAA Guidance, Navigation and Control Conference*, August 2004.

[26] V. H. L. Cheng, V. Sharma, and D. C. Foyle. Study of aircraft taxi performance for enhancing airport surface traffic control. *IEEE Transactions on Intelligent Transportation Systems*, 2(2), 2001.

[27] H. Balakrishnan and Y. Jung. Benefits of taxiway operations planning at Dallas/Fort Worth airport. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007. To appear.

[28] I. D. Anagnostakis. *A Multi-Objective, Decomposition-Based Algorithm Design Methodology And Its Application To Runway Operations Planning*. PhD thesis, Massachusetts Institute of Technology, 2004.

[29] H. R. Idris. *Observation and Analysis of Departure Planning Operations at Boston Logan International Airport*. PhD thesis, Massachusetts Institute of Technology, 2000.

Keywords: Departure management, runway operations scheduling, active runway crossings, wake vortex separation, constrained position shifting, dynamic programming.

Author Biographies

Hamsa Balakrishnan is an Assistant Professor of Aeronautics and Astronautics and of Engineering Systems at the Massachusetts Institute of Technology. She received her PhD in Aeronautics and Astronautics from Stanford University in April 2006, following which she was a researcher at the University of California, Santa Cruz and the NASA Ames Research Center. Her research interests include algorithms for the scheduling and routing of air traffic, techniques for the collection and processing of air traffic data, and mechanisms for the allocation of airport and airspace resources.

Bala Chandran is currently an analyst at Analytics Operations Engineering, Inc. in Boston, MA. He received his PhD from the Department of Industrial Engineering and Operations Research at the University of California, Berkeley in April 2007. His research interests lie in combinatorial optimization, and algorithm development and implementation.