AIAA Guidance, Navigation, and Control Conference and Exhibit
16 - 19 August 2004, Providence, Rhode Island

AIAA 2004-5408

Towards Automated Airport Surface Traffic Control: Potential Benefits and Feasibility

Brian J. Capozzi, Ph.D.* and John DiFelici†
Metron Aviation, Inc., Herndon, VA, 20170

Ray Jakobovits, Ph.D.‡
Consultant, Reston, VA, 20165

The control and management of traffic in the future National Airspace System (NAS) will be increasingly automated. The airport surface, as a key bottleneck to throughput in the current NAS, is an ideal place to begin research into the feasibility, potential benefits, and resultant costs and risks associated with the introduction of significant automation. Toward this end, this paper describes an initial architecture and core functions which implement a basic automated airport surface traffic planning and control system. In particular, we present several algorithmic approaches to the generation of conflict-free, time-based surface trajectories. The relative performance of these algorithms is compared in terms of efficiency, runway throughput, and an estimate of terminal airspace complexity. Results from these preliminary experiments indicate the potential for relatively substantial improvements in surface movement efficiency at the expense of imposing additional requirements relative to the control of timing of aircraft entry into the airport system.

Nomenclature

E	=	set of edges contained within a graph
e_k	=	the k-th edge contained in a given path through a graph
$e_k(t_a,t_b)$	=	denotes the usage of a given edge from time t_a to time t_b
F_A	=	subset of arrival flights
F_D	=	subset of departure flights
G	=	Graph or network
$G[k]$	=	State of graph at time k
J	=	objective function
$N(\mu,\sigma^2)$	=	Gaussian normal distribution with mean μ and standard deviation σ
$\hat{t_{OUT}},\hat{t_{ON}}$	=	estimated spot request time for departures, estimated landing time for arrivals
t_{OFF},t_{IN}	=	planned runway departure time, planned spot request time for arrivals
V	=	set of vertices contained within a graph
$v_k(t_a,t_b)$	=	denotes the usage of a given vertex from time t_a to time t_b
Δ_{sep}	=	the time separation between successive arrivals into a given terminal area
λ_s	=	parameter for Poisson distribution governing the rate of departure handoff spot requests
$\pi(v_0,v_f)$	=	path through a network from a designated initial vertex to a final vertex

I. Introduction

THE airport surface is where the majority of delays in the current National Airspace System (NAS) are felt by passengers and airlines. Recent trends regarding the rate of growth in air passenger travel indicate that these delays are likely to get worse rather than better. Although these delays are not always directly related to capacity constraints or operational inefficiencies at the local airport, this is often the place where aircraft are held in order to deal with traffic congestion or weather situations elsewhere in the NAS. The current hub and spoke system

* Senior Analyst, Aviation R&D, AIAA Member.
† Senior Software Analyst, Aviation R&D.
‡ Consultant, Aviation R&D.

Copyright © 2004 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

exacerbates this problem as delays tend to propagate and grow as airlines struggle to recover from unanticipated delays. Current research and development both in the U.S. and Europe is targeted at the development of near-term decision-support tools to assist both the Air Traffic Service Provider (ATSP) and NAS users in better utilizing existing airspace and surface capacity. The Surface Management System (SMS)[1] is a NASA-funded decision-support tool (DST) that provides improved common situation awareness across ATSP domains as well as between ATSP and airlines. SMS, currently transitioning to the FAA for deployment, provides a number of displays and predictive tools enabling improved coordination between traffic management on the airport surface and constraints imposed by neighboring domains. Similar research activities in Europe[2] include Arrival Manager (AMAN), Departure Manager (DMAN), and Linking Existing On Ground, Arrival, and Departure Operations (LEONARDO).

A number of longer-term research efforts are developing future NAS concepts of operation for the year 2020 and beyond. The goal of many of these efforts is to revolutionize the nation's air transportation system in order to provide the necessary increases in capacity to mitigate the forecasted delays associated with the projected growth in demand. One vision for the future stipulates that, in order to better utilize scarce NAS resources, the future NAS will become increasingly automated[3]. The rationale for this view is that algorithmic planning and coordination of aircraft movements, along with complementary improvements in a number of enabling technology areas, will enable more efficient solutions to be found then is possible given current ATSP roles and responsibilities, information flow, and control paradigms. To realize this future state, a transition from a strictly *human-in-control* to a *human-guided* system is required. Under such a paradigm, automation systems will carry out the details of developing and disseminating a given plan and continuously updating the plan to account for changes in the traffic situation. Humans will provide high-level guidance for the underlying algorithms, specifying preferences, constraints, or conditions that are difficult for computers to obtain from data alone.

This paper focuses on conflict-free surface trajectory planning, a core functionality required for the realization of future automated aircraft planning and control on the airport surface. A number of approaches to this problem have been proposed. Gotteland et al.[4] describe an approach to surface traffic planning based on genetic algorithms. Brinton et al.[5] discuss both a space/time network approach and a collaborative co-evolutionary technique, the latter technique designed to simultaneously evolve complete, conflict-free routes for a set of aircraft given a known start and end point. Simultaneous evolution of complete routes in this fashion overcomes many of the shortcomings of local decision-making in which flights are routed in a pre-determined order using frontier-based search. Smeltink et al[6] and Visser and Rolling[7] use a mixed-integer linear programming formulation to schedule flights along a pre-determined route for each flight. This paper extends this idea by examining the additional efficiency gains to be had by considering the aircraft routing as an additional degree of freedom – both in terms of the number of routes explored between a given source and sink as well as allowing the runway assignment for each flight to be dynamically determined rather than fixed a priori as is common today. Acknowledging the computational difficulty associated with solving the problem to optimality, several candidate heuristic planning algorithms are explored.

The remainder of this paper is organized as follows. Section II provides a brief summary of the future airport surface concept that is the impetus for this research. In Section III, the scheduling algorithms themselves are presented along with the network framework and aircraft motion models used to instantiate the surface routing problem. Section IV presents the results obtained when exercising these algorithms over a given demand set along with a number of observations regarding performance characteristics. Finally, Section V concludes with a summary and a brief discussion of future research plans following on from the work presented here.

II. Concept Overview

The Automated Airport Surface Traffic Control concept[3], the genesis for the algorithmic research reported here, is founded on the following three core ideas, as summarized graphically in Figure 1:

- **Automated Surface Traffic Control and Monitoring**: Surface movement clearances are automatically generated and delivered to pilots, eliminating the need for controllers and pilots to verbally coordinate and confirm clearances and overcoming many of the operational inefficiencies in current surface operations.
- **Collaborative, Terminal Area Wide Movement Planning**: Based on updated estimates of gate pushback and runway arrival times, time-based surface trajectories are algorithmically generated for each aircraft from/to the runway to/from the user's ramp area. In essence, runway usage and surface routings are generated simultaneously. These trajectories are generated in order to attempt to optimize one or more objective functions, subject to the constraint that the planned trajectories are conflict-free. Trajectory specification includes one or more time checkpoints or required times of arrival that must be met with a given navigational precision in order for aircraft to remain free of conflict. The planning algorithms reserve the right to negotiate with terminal airspace controllers regarding the assignment of aircraft to runways and the desired landing time for arriving aircraft, subject to the ability of individual aircraft to be expedited or delayed.

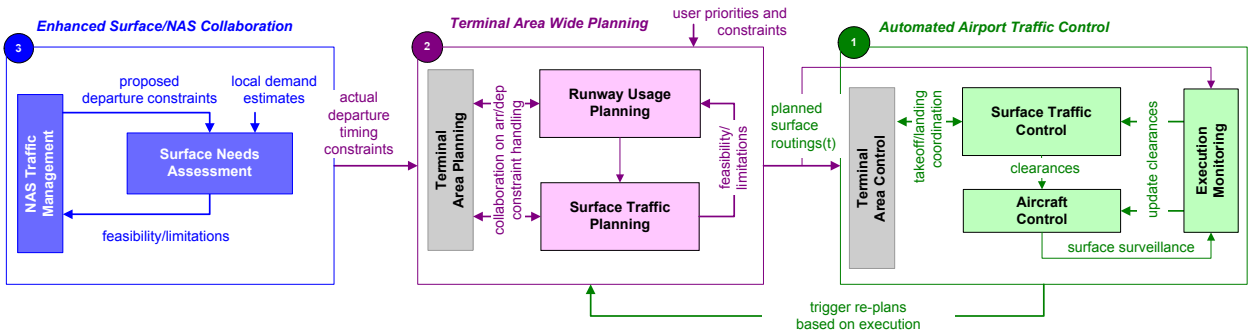


Figure 1 Concept block diagram – focus of this paper is on planning

- **Increased Integration of Surface and En Route Operations**: Efficient terminal area operations alone will not suffice to increase NAS capacity. Rather, terminal area operations must be planned and conducted explicitly in the context of the NAS outside of the local terminal area, bridging the gap between strategic traffic management planning and tactical implementation which exists today. This core idea is built on the principle of explicit planning and coordination of the degrees of freedom available in each domain – surface, terminal, transition, and en route – for accelerating or delaying flights relative to a nominal timeline. Instead of planning surface movements independently from the planning and management of the airborne trajectory, the aircraft's timing through the surface and into en route airspace should be explicitly planned across domains. Multiple routing options (user preferences and constraints) should be specified by users to facilitate allocation of scarce resources and to help manage uncertainty in demand and weather.

III. Conflict-Free Surface Trajectory Planning

The purpose of this paper is to demonstrate the technical feasibility of developing efficient, conflict-free surface routings and evaluate the relative performance of different heuristic algorithmic approaches. In this paper, we evaluate performance primarily in the context of transit delays on the airport surface and runway system throughput. In the current study, however, it should be emphasized that there are no external constraints imposed on the solution such as Miles-In-Trail (MIT) restrictions, Expected Departure Clearance Times (EDCT), or other departure restrictions common in today's NAS. Thus, in some sense, we are not truly evaluating the benefits of such algorithmic routing approaches in a system-wide efficiency sense, but rather from the greedy perspective of individual flights.

A. Modeling the Airport Surface

For the purposes of this paper, we frame the surface trajectory planning problem in terms of the allocation and scheduling of a set of surface resources. Specifically, we model the airport surface as a graph structure, G, comprised of a set of vertices, V, and a set of edges, E. Each vertex in this graph represents a physical intersection on the airport surface. The edges in this graph represent the taxiway and runway segments connecting the intersections. The path of an aircraft through this network from its initial node, v_0, to its final node, v_f, can be described in terms of an alternating sequence of edges and vertices: $\pi(v_0,v_f) = \{v_0,e_1,v_2,e_2,\ldots,v_f\}$. Similarly, a *trajectory* through this network can be described as a sequence of appointments on each resource, each with a start and end time: $\pi(t) = \{v_0(t_0,t_1),\ e_1(t_1,t_2),\ldots,\ v_f(t_{f-1},t_f)\}$. Both nodes and links in the network are modeled with a finite size represent the time required for an aircraft to transit the resource at a given speed. The capacity of all intersections is set to unity – it is assumed that only one aircraft can occupy an intersection at any given time. The capacity of each of the links in the network is defined by the number of aircraft that can physically be placed tip to tail along its length, given an assumed separation of 150 feet between aircraft.

Generally, each taxiway resource has a separate resource manager, responsible for managing appointments. Each resource maintains a list of appointments that have previously been scheduled. This dynamic appointment list is used as the basis for determining and resolving potential conflicts. Nodes and links that comprise an active runway are managed by one or more runway resource managers – if a runway node forms an intersection between two active runways, appointments for that node are coordinated between both runway managers. This provides a mechanism for modeling dependent runways. Special treatment is used to allocate all runway resources "in front of" an arriving or departing aircraft so that no other aircraft can enter the runway while for the duration that the runway is anticipated to be in use. Logic is included to release runway resources "behind" the arriving or departing aircraft

to allow runway crossings or other taxi operations on the runway which do not pose a safety threat. A landing aircraft frees the runway resources once it has taxied onto the runway exit link. A departing aircraft frees its runway resources once it has reached a given takeoff speed.

B. Modeling Aircraft Motion

In order to develop viable surface movement trajectories, an aircraft motion model is required. This model is used as the basis for estimating surface resource occupancy into the future enabling de-confliction of future conflicts. For the purposes of this paper, we assume a simple kinematic model. Landing aircraft motion is parameterized by a touchdown speed, a deceleration constant (e.g., braking rate), and a maximum runway exit speed. Departing aircraft are similarly parameterized by an acceleration rate and a takeoff speed. The parameters for landing and departing aircraft are in general a function of aircraft weight class. Given these parameters, it is possible to determine multiple feasible points for the initiation of takeoff roll (e.g., for aircraft that do not require the use of the entire runway) and for arrivals to exit the runway. Such flexibility can be valuable from the perspective of improving surface movement efficiency and achievable runway sequences.

C. Problem Description

The objective is to find a trajectory for each aircraft through the network for each aircraft that optimizes a certain performance objective function, subject to the constraint that the trajectories remain conflict-free over their entire length. We specifically use the term trajectory instead of path to denote the fact that the output of this process is a time-based description of the surface routing for each vehicle, including necessary holds at the start point, one or more intersections, or in general, at any arbitrary point along the route. Classes of conflicts that we detect and resolve in the process of scheduling include: (a) intersection crossing conflicts in which more than one aircraft desire to use an intersection at the same time, (b) overtake conflicts in which one vehicle attempts to pass another on a given resource when traveling in the same direction, and (c) head-to-head conflicts that occur when aircraft attempt to transit a particular resource in opposite directions.

An idealized objective function that seeks to minimize the total surface transit time over all flights can be expressed as:

$$J = \sum_{F_A}\left(t_{IN} - \hat{t}_{ON}\right) + \sum_{F_D}\left(t_{OFF} - \hat{t}_{OUT}\right) \tag{1}$$

where F_A represents the set of arrival flights, F_D represents the set of departing flights, t_{IN} represents the planned gate arrival time, $\hat{t}_{ON}$ is the estimated runway touchdown time, t_{OFF} represents the planned runway departure time, and $\hat{t}_{OUT}$ is the estimated gate departure time.

In this paper, we do not attempt to solve this optimization problem directly due to the combinatorial size of the search space. Rather, we explore algorithmic techniques which seek to optimize the transit time of individual flights, one at a time, given the appointments made by any previous flights on the airport network. Given this approach, we have several degrees of freedom available. First and foremost, we explore the order in which flights are scheduled. Secondly, we explore the benefit associated with allowing the runway assignment to be dynamically determined based on minimizing an individual flight's transit time rather than being set a priori by a nominal departure scenario as is common in today's NAS. Thirdly, we investigate the impact of searching over multiple routes between a given source and sink node rather than being restricted to a single route.

D. Algorithm I – Coupled Trajectory Search and Conflict Avoidance

An initial approach to allocating airport surface network resources to a set of N flights has been implemented, based on performing a search (extending Dijkstra's method[8]) through a sequentially constrained network. The objective is to create a conflict-free Gantt chart representing the time-based allocation of resources to each of the N flights within a given planning window. This algorithm assigns delay as necessary along the scheduled trajectory in order to resolve any projected conflicts identified in the course of performing the search.

1. Basic Algorithm

Basic steps of the algorithm include:

- **Step 1**: Assume a list that determines the order in which the flights will search the network. This list can be constructed either on the basis of a heuristic (FCFS) or as a random perturbation of the list ranging from 1 to N. All resources in the graph network to be searched, *G[0]*, are initialized as available over all time periods.

- **Step 2**: For the kth flight, use Dijkstra's algorithm to find the shortest-time trajectory through the network *G[k]* to any available runway (for departures) or to the assigned gate (for arrivals) given the outcome of any previous (k-1, k-2, … etc.) searches through the network. This trajectory consists of a sequence of resources and the time span across which they are needed by a flight. Note that for k = 0, this shortest-time trajectory will correspond to the shortest distance path connecting the flight from source to sink
- **Step 3**: Allocate any needed resources to the kth flight (thus making the resources unavailable to other flights over the corresponding time period). This creates a modified network, *G[k+1]*, to be used as the basis for the (k+1)th search. If all flights have been processed, then exit. Otherwise, go to Step 2.

Note that this algorithm is sub-optimal by definition since it processes the flights in a sequential fashion (as determined by some heuristic) and thus only explores a small portion of the overall search space. Nonetheless, this approach provides a viable initial step toward more sophisticated algorithmic approaches. Embedded within this algorithm is a recursive procedure for resolving head-on conflicts over one or more common trajectory segments.

2. Conflict Detection and Resolution Illustration

The following example illustrates the action of the conflict detection and resolution (CD&R) algorithm embedded within the Dijkstra's search (Step 2 above). Consider the situation shown below in Figure 2, where aircraft A has a taxi route from N1 to N5 and aircraft B has a taxi route from N4 to N1. The independent scheduling of these taxi routes is shown in the Gantt chart at left – here, each aircraft was scheduled forward in time from its current position based on its current speed over a given time horizon. In this case, aircraft A is scheduled first since its estimated start time is earlier than that for aircraft B. Note that, if no action is taken, the aircraft would come into conflict along link L2, as indicated by the shaded region on the Gantt chart. In this case, aircraft B enters link L2 while aircraft A is still traversing it.

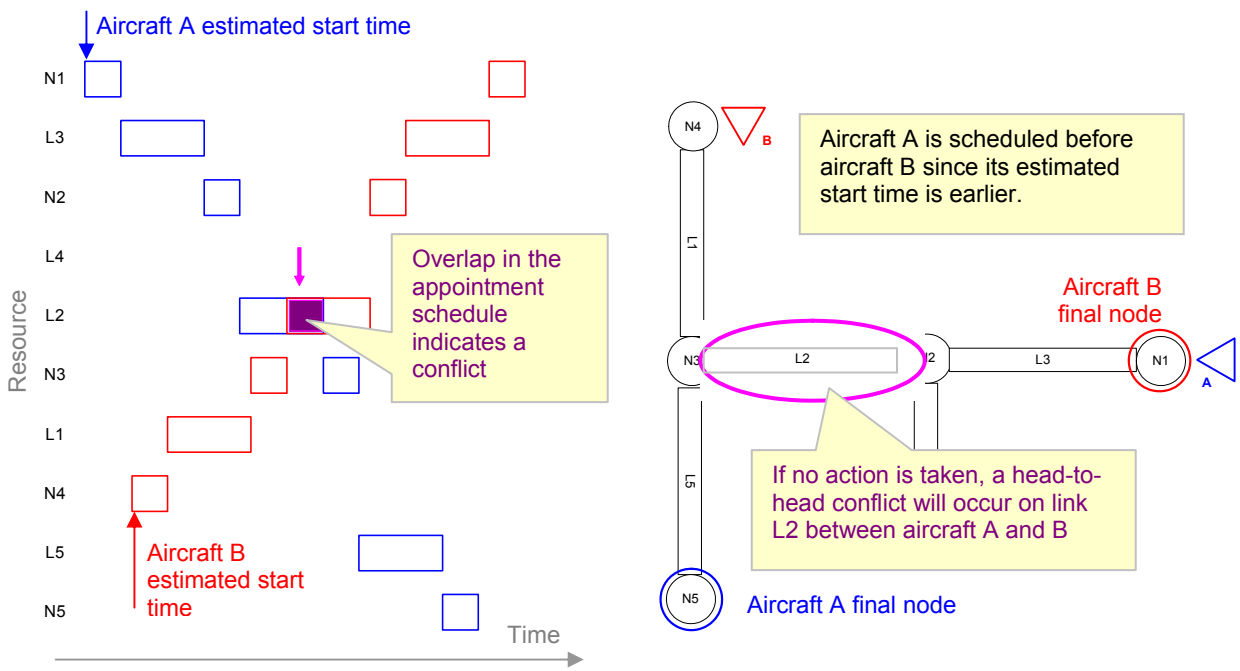


Figure 2: Example schedule and geometry used to illustrate CD&R algorithm

Having detected some sort of conflict, we now need to determine the nature of the conflict (at this point, technically we do not yet know that it is a head-on conflict). Given that aircraft A reaches link L2 first, we now check to see if aircraft A's next resource is the same as aircraft B's previous resource. Here, we see that this is indeed the case – aircraft A desires to use node N3 next. Now we have categorized the potential conflict as one in which resource contention will occur. At this point, we continue to check for common resources until the previous path for aircraft B and the next path for aircraft A diverge. In this case, since aircraft A's next desired resource is L5 while aircraft B's previous resource is L1, we are finished. The resolution is for aircraft B to not enter node N3 until after aircraft A has left it. This effectively creates a target time constraint for aircraft B regarding entry into intersection N3. In some sense, so long as aircraft B is not adversely impacting other aircraft, we are not concerned with the details of how that time constraint is satisfied. The necessary delay could be absorbed via a hold at node N4, a hold at the end of L1, or slowing along its current segment (L1). Figure 3 illustrates one such resolution.

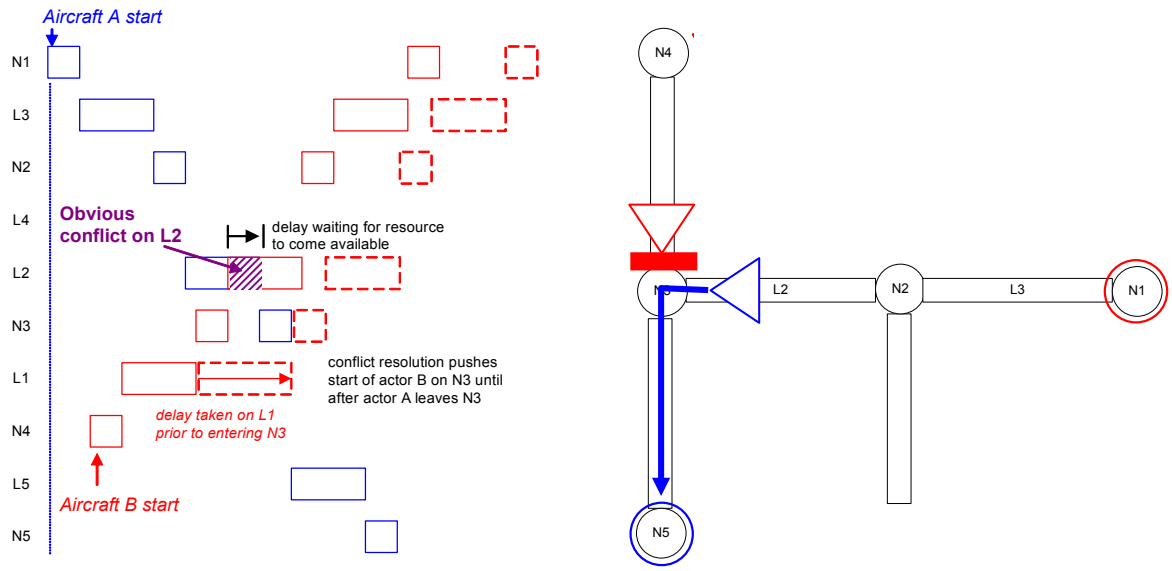


Figure 3 Illustration of resolution – aircraft B gives way to aircraft A, holding prior to entering vertex N3

In general, the steps of the conflict resolution algorithm are as follows. First, determine which aircraft of the pair arrives earliest at the contended resource. Call this aircraft A, and the later-arriving aircraft B. Attempt to 'back up' aircraft B (e.g., absorbing delay) successively along its sequence of requested resources between its current resource and the contended resource in order to try and find a point of divergence between the future trajectories of the two aircraft within the planning horizon. In the worst-case, aircraft B will have to be backed up all the way to its starting location (e.g., gate or handoff spot for departures, runway touchdown point for arrivals).

Note that, since the CD&R algorithm is embedded within the Dijkstra shortest-time path search, the delay required in order to resolve conflicts is added to the cumulative transit time stored in the search tree as the basis of establishing the current search frontier. As such, should the delay required be excessive, it is likely that a lower cost node exists in the search tree and will be selected as the next point to be expanded on the search frontier.

E. Algorithm II – Delay Aircraft at Initial Node, Unimpeded Transit

The algorithm described above tends to distribute delays necessary to resolve any projected conflicts starting at the resources closest to the conflict and spanning backwards away from the conflict as necessary. As an alternative approach, we consider a trajectory planner which searches to determine the minimum delay required to enable each aircraft to execute its planned route without stopping. In doing so, the planner adjusts the time at which each aircraft enters the airport surface. For arrivals, this amounts to vectoring or speed adjustment to absorb any planned delay. For departures, this equates to a hold at the gate prior to pushback.

The algorithm consists of two phases.

- Initial Planning Phase: Each aircraft is scheduled independently, ignoring any interactions with other traffic, to compute the time required for the aircraft to transit the airport surface using each of the options in its Route Option Set. The route with the shortest transit time is selected and appointments are established on the corresponding resources.
- Conflict-Free Planning Phase: Given the set of schedules constructed above, determine if there are any overlaps in appointments on any of the resources. If no conflicts exist, then exit with the initial set of planned trajectories. Otherwise, order the flights according to some priority rule and perform the following steps.
    - Loop over flights. For each flight –
    - Loop over the set of routes contained in the flight's Route Option Set. For each route –
    - Compute the delay required for a given route option to be traversed unimpeded given the motion model for the current flight, given all appointments made by any previously scheduled flights
    - Having computed the delay for all route options, select the route associated with the minimum delay. Set appointments on all resources given the route and motion model.

We now turn to a description of the experiments conducted in order to evaluate these algorithms.

IV. Evaluation of Planning Algorithms

In this section we describe a set of experiments conducted in order to evaluate the relative performance of the various planning algorithms presented in the previous section. In each experiment, the various heuristic planning algorithms are applied to the same demand set. In evaluating performance, each planner is assumed to have perfect information regarding the timing of entry of aircraft into the airport system and the time required for aircraft movement along trajectory segments. Each generated schedule is assumed to be executed perfectly.

A. Experiment Setup

The network layout used for these experiments is based on the physical layout of Newark Liberty (EWR) airport, as illustrated below in Fig. 4. For purposes of this paper, we utilize only the primary runways, 22L and 22R, and assume that the airport remains in a south flow configuration over the entire experiment. Note that we do not, in this paper, model the detailed layout of the airport in the vicinity of aircraft parking gates. Rather, we assume that departure aircraft originate at one of several designated ramp area handoff locations. Arrival aircraft are similarly assumed to exit from ATSP control once reaching one of these handoff locations.

A similar abstraction is used in modeling flows entering and leaving the airport via arrival and departure fixes. Two virtual fixes are defined near the arrival and departure end of each of the active runways. The virtual arrival fixes represent a collection or set of actual arrival fixes whose terminal area flows ultimately merge prior to reaching the airport runways. Similarly, aircraft flows are assumed to diverge once having passed the virtual departure fixes after departing from the airport. In the default fix/runway mapping, all arrival flights, regardless of their arrival fix, are assigned to the outboard runway (22L). Departure flights are likewise by default assigned to the inboard runway independent of their filed departure fix. Note, however, that network links do exist from the virtual fixes to each active runway. These links are required to enable the Flexible Runway Assignment heuristic utilized in some experiments.

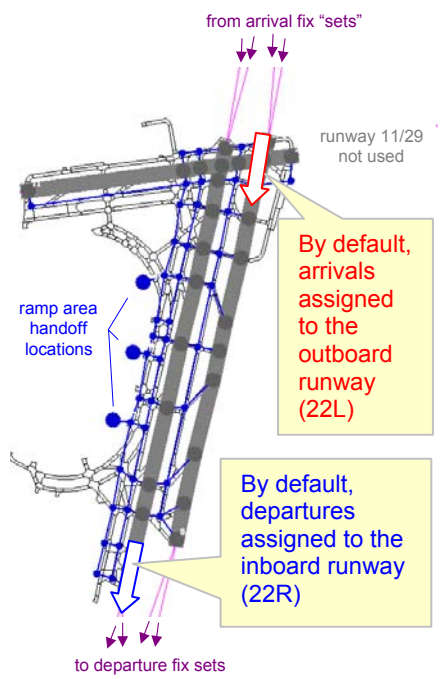
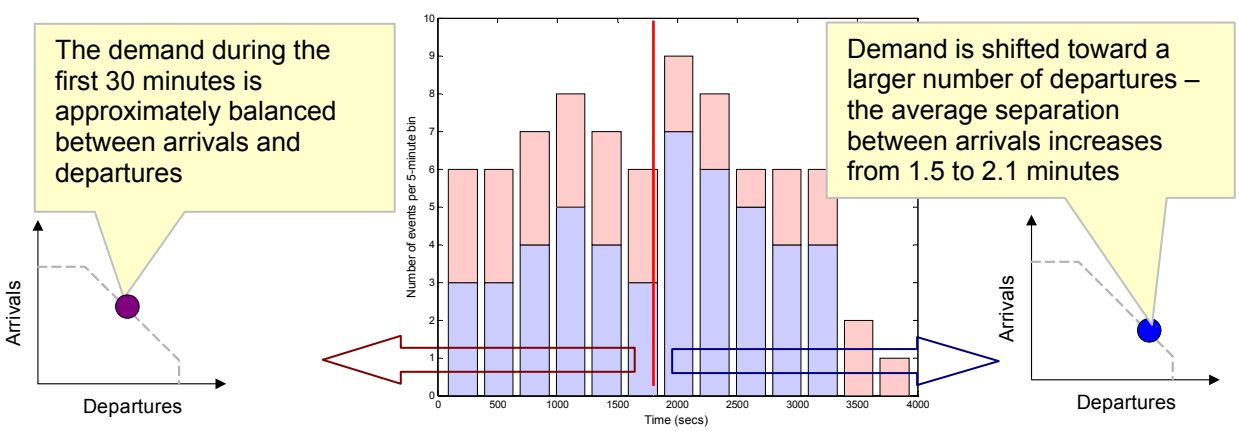


Figure 4 Network layout used for experiments

The demand set used for these experiments is depicted graphically in Fig. 5. This demand set consists of 60 minutes of scheduled traffic, comprised of 49 departure flights and 30 arrival flights. Departure entry into the airport system is governed by an assumed Poisson distribution for the average number of spot handoffs requested per minute. Separation between arrival successive arrivals is modeled by a bounded Gaussian distribution. The distribution of arrivals and departures is designed such that over the first 30 minutes, demand is approximately equal between arrivals and departures. Over the second 30-minute interval, the demand shifts to a heavier departure push with increased average separation between successive arrivals.



$\Delta_{sep} = 1.0 \le G\left(1.5, 0.5^2\right) \le 4.0$, $\lambda_s = 0.5$ $\qquad$ $\Delta_{sep} = 1.5 \le G\left(2.1, 0.5^2\right) \le 5.0$, $\lambda_s = 1$

Figure 5. Demand profile used for evaluating planning algorithms

The metrics used in evaluating the various scheduling algorithms are as follows:

- **Surface Traffic Efficiency**, η, defined as the ratio of the unimpeded surface transit time to the actual transit time. This metric can be computed on an individual flight basis as well as in an average sense over a set of flights, for example, within a given time window. In an ideal case, each flight could transit the airport with zero delay, thus achieving a traffic efficiency value of 1.0.

$$\eta = \frac{unimpeded\ transit\ time}{actual\ transit\ time} = \frac{(planned\ engine\ on\ time) - \sum(planned\ delays)}{(planned\ engine\ on\ time)} \tag{1}$$

  In general, the surface traffic efficiency is bounded from below by zero and above by unity. Note that, in the results that follow, this metric is used only for evaluating Algorithm I; the surface traffic efficiency is, by definition, equal to 1.0 for all solutions produced by Algorithm II since all flights travel unimpeded.
- **Terminal Area Transit Delays** – measured as the difference between the planned aircraft transit time and that which could be achieved if the aircraft could transit unimpeded, initiating motion precisely at its originally scheduled time. Various statistics including the number of delayed flights, the average and maximum delay incurred, and the distribution of delay between arrival and departure operations are generated for each experiment. In addition, for Algorithm I, we record not only the total amount of delay but where it is occurring. For departures, we differentiate between Spot delay, Taxi delays (prior to queuing at the runway) and Runway Queuing delays. For arrivals, we differentiate between Approach delay, Taxi delay, and Spot delay. Spot delay occurs when an arrival must wait for a departure flight to clear the spot prior to entering the ramp area.
- **Runway throughput** – measured as the number of operations per time bin. In this paper we use a 5-minute time bin and track arrivals and departure operation counts separately on both a per-runway and net airport throughput basis.
- **Number of Non-Standard Runway Assignments** – The Flexible Runway Assignment heuristic can result in aircraft being assigned to non-standard runways. This is assumed to correlate to some extent to additional workload required in the terminal airspace in order to keep such flights separated from the nominal traffic flows. The time-density (per 5-minute bin) of such non-standard assignments is used as a proxy for this workload metric.
- **Surface traffic complexity** – This metric tracks the number of unique many routes assigned between a given source/sink in a given period of time. This metric is assumed to be related to the difficulty that a human would face in trying to keep synchronized with the actions of the underlying scheduling algorithms. Such awareness is a critical factor should the human have to step in and assume control of a part of the airport surface in the event of a data or automation failure.

Fundamentally, the experiments run as part of this study are designed to explore several scheduling degrees of freedom. First and foremost, we investigate the impact of different ordering heuristics:

- **Arrival Priority** – The airport schedule of events is ordered such that all arrivals are scheduled first, in first-come, first-served (FCFS) order. Once all arrivals are scheduled, departures are scheduled FCFS
- **Event Order Priority** – Flights are scheduled according to the natural FCFS order as determined by the estimated touchdown time (for arrivals) and estimated spot handoff request time (for departures)

For Algorithm II, we investigate a third priority scheme, referred to as a Minimum Conflict Priority rule. The rationale for this heuristic is to force flights that, in a natural, FCFS schedule, would tend to delay a large number of flights to be pushed later in the scheduling order. For example, if scheduling a given flight causes a line of ten other flights to be delayed along a common resource, it makes sense for the offending flight to be scheduled after the ten other flights.

- **Minimum Conflict Priority** – Each flights is scheduled in an order determined by the number of conflicts in which it is involved based on a naïve, FCFS scheduling of all flights on their shortest-time path starting at their estimated time of entry into the system. Flights are scheduling in order of increasing number of conflicts such that aircraft projected to interfere with a large number of other flights are scheduled last. Ties are broken based on the original estimated event time.

Secondly, we investigate the benefits associated with expanding the routing options available to the scheduler.. A Flexible Runway Assignment scheme is invoked in the context of both Algorithms I and II to allow the scheduler to assign aircraft to runways that results in a lower airport transit time, regardless of the default runway defined by the flight's arrival or departure fix. A Route Option Set is also explored, albeit solely in the context of Algorithm II. The contents of the Route Option Set determine the number of alternative routes available to the Algorithm II Scheduler. We explore two extremes in this paper. At one extreme, the route set is a singleton, comprised of only the shortest-transit time, feasible route between the flight's start and end vertex. At the other extreme, we examine the potential benefit of exploring all possible routes – as defined by all routes in the graph that contain at least one different vertex. Note that the shortest transit-time route is not necessarily always the shortest distance route. For departures, because we have not enabled the shortened takeoff feature, the shortest-time viable route equates to a single route, namely the shortest-distance path from the flight's handoff spot to the departure end of one or more runways. For arrivals, routes from all viable runway exit points (e.g., those greater than a certain distance from the touchdown point) are included in the route set. Depending on the aircraft braking rate, it may, in some cases be faster for an aircraft to take an early runway exit and taxi slightly farther to the handoff spot rather than rolling all the way out to the runway exit corresponding to the shortest-distance path to the spot.

The combinations of heuristics investigated in these experiments are summarized in Table 1. Note that the Minimum Conflict Priority and Route Option Set degrees of freedom are only explored using Algorithm II.

Table 1 Experiment matrix

		Flight Ordering Heuristic		
		Arrival Priority	Event Order	Min. Conflict Priority
Rwy Assign Heuristic	Static, Fix-Based	I/II	I/II	II only
	Flexible, Shortest Transit	I/II	I/II	II only
Route Option Set	Shortest-Time Feasible Route	II only	II only	II only
	All Possible, Feasible Routes	II only	II only	II only

B. Results – Algorithm 1

A high-level summary of the delay statistics resulting from the application of Algorithm I to the demand set across the various combinations of heuristics explored is presented in Table 2. Considering first the top-most rows corresponding to Arrival Priority scheduling, it is clear that, as expected, arrivals are transiting the surface with essentially zero delay. Comparing the static and flexible runway assignment cases, however, it can be seen that the total, average, and maximum departure delays are greatly reduced – even when arrivals are given priority – under a flexible runway assignment scheme. Similar trends are observed in the lower half of the table for the Event Order cases, where the total delay across both arrivals and departures is seen to decrease. However, note that there is a net transfer of delay from departures to arrivals as the average and maximum delays for departures goes down at the expense of a slight increase in said values for arrivals.

Table 2 Summary delay statistics for Algorithm I experiments

	Static Rwy Assignment		Flexible Rwy Assignment	
Ordering Heuristic	Arrivals	Departures	Arrivals	Departures
Arrival Priority				
Total Delay (min)	0.0	283.4	0.04	34.7
Num Flights delayed	0	49	1	37
Average Delay (min)	0	5.78	0.04	0.9
Maximum Delay (min)	0	13.1	0.04	2.6
Event Order				
Total Delay	6.6	79.3	20.9	10.2
Num Flights delayed	19	43	26	27
Average Delay	0.3	1.8	0.8	0.4
Maximum Delay	0.8	4.9	3.7	1.7

Results obtained through evaluation of the various combinations of flight processing order and runway assignment heuristic for Algorithm I are presented below in Fig. 2(a)-(d). In each of these figures, the surface traffic efficiency, computed using Eq. (1), is plotted against each flight's event time. The event time is defined as the planned spot time for departures and the planned runway touchdown time for arrivals. Arrivals are shown in red using open circles while departures are indicated by blue dots. To aid in visualizing the trend for arrival and departures, the average surface traffic efficiency obtained across each 5-minute bin over the duration of the planning window is also shown.

In Fig. 6a, in which Arrival Priority order is used in scheduling with a static runway/fix mapping, it can be observed that all arrival flights had a surface traffic efficiency value of 1.0 while departure flights ranged from a value near 0.9 early in the window to values as low as 0.2 later in the window. Fig. 6b, by comparison, shows the relative improvement in surface traffic efficiency for departures achievable when the departure runway assignment is allowed to be determined independent of departure fix as efficiency values for departing flights remain within 0.6-1.0 across all flights. When Event Order is used in conjunction with a static fix mapping (Fig. 6c), as expected, one observes a reduction in the relative efficiency of arrival movements as compared with the Arrival Priority case in Fig. 6a. Clearly a tradeoff between arrival and departure delays is at play as the trend line for departures in Fig. 2c is lifted in comparison with that shown in Fig. 6a. The best efficiency values, when averaged across both arrivals and departures, are evident in Fig. 6d, suggesting the potential benefits associated with a scheduling algorithm operating in Event Order but taking advantage of flexible runway assignments.

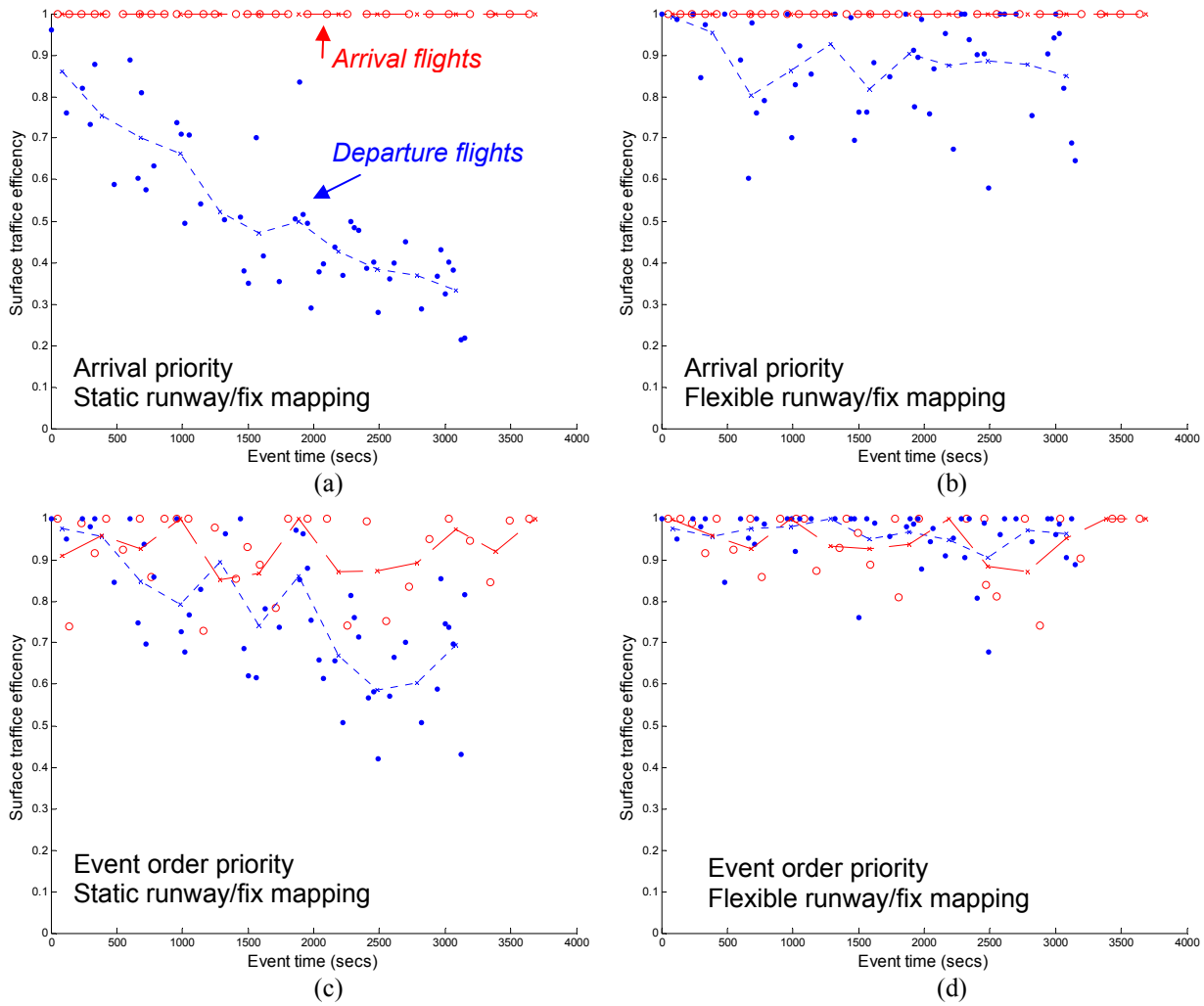


Figure 6 Surface traffic efficiency obtained for demand set 1 using various planning heuristics.

The runway throughput metrics corresponding to the surface traffic efficiencies above are depicted in Fig. 7(a)-7(d). In each of these figures, the number of arrival operations (shown in red) and departure operations (shown in blue) in each 5-minute bin are plotted against the event time. The uppermost subplot in each case corresponds to operations on runway 22L while the lower subplot reflects operations on runway 22R. In Fig. 7a and 7c, in which the mapping between runway and fix is the default mapping, arrival operations are solely on the outboard runway (22R) while departures are confined to 22L. Comparing the runway throughputs between these two plots, it can be seen that, consistent with the efficiency values above, the Event Priority scheme serves the departures faster than the Arrival Priority scheme. Shifting attention to Fig. 7b and 7d, one observes that the resulting schedules on both 22L and 22R are significantly mixed between arrivals and departures when flexible runway assignments are in use. Also shown in Fig. 7b and 7d is the time density of the number of non-standard runway assignments. This metric gives an approximation to the level of workload in the terminal airspace that might be required in order to realize the planned runway assignments. The relative efficiency improvement in terms of serving departures between static and flexible runway assignments can be observed by comparing Fig. 7a and 7b. In fact, the total runway throughput rate is slightly improved in the latter case. When constrained to use the default static mapping, the last departures do not launch until the time bin centered on 3900 seconds. By comparison, under the same Arrival Priority scheme but taking advantage of flexible runway assignment, the last departures are served in the time bin centered on 3300 seconds. Similar trends are observed when comparing Fig. 7c and 7d in which Event Order priority is used. Note, however, that the number of non-standard assignments is reduced for the Event Order priority case (Fig. 7d) relative to that observed in the Arrival Priority case (Fig. 7b).

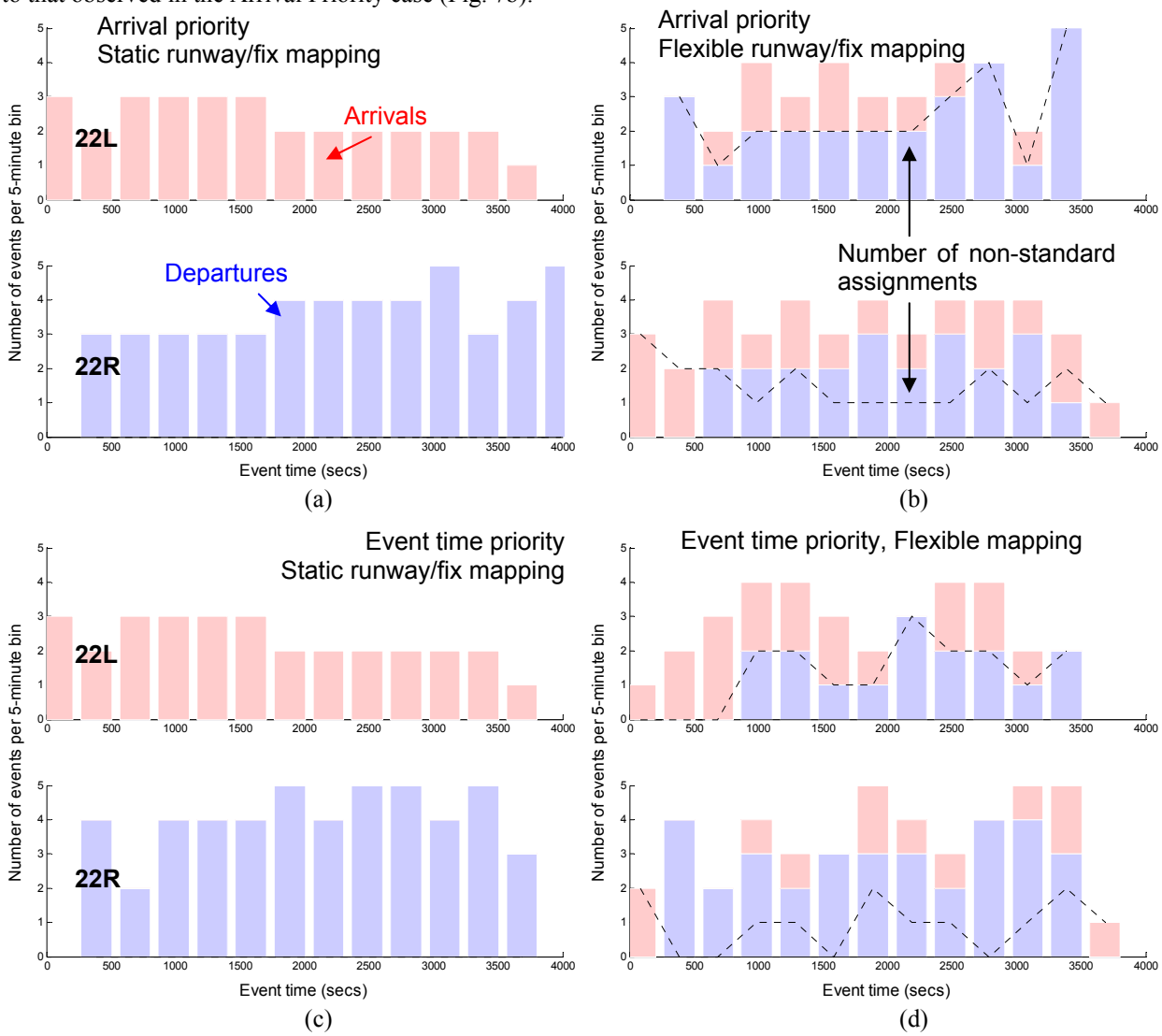


Figure 7: Runway throughput achieved under different planning heuristics

C. Results – Algorithm II

Summary statistics obtained running Algorithm II against the demand set for various combinations of runway assignment scheme and schedule ordering heuristic are presented in Table 3. On the left side of Table 3, the performance of three schedule ordering heuristics is compared when flights are restricted to assignment on their default runway (as defined by the default fix/runway mapping). The total delay absorbed over all flights is seen to range from 82.5 minutes for the Event Order case to nearly 158 minutes for the Minimum Conflict Priority heuristic. In each case, delay tends to be more heavily incurred by departure flights due to the fact that the rate of pushback is set to exceed the nominal single-runway capacity. This effect is obviously most severe in the case of the Arrival Priority heuristic as departures must effectively wait for the majority of arrivals to clear the airport surface prior to beginning taxiing toward the departure runway (due to the sharing of several of the taxiway links between arrivals and departures).

Table 3 Algorithm II delay statistics for various ordering heuristics and runway assignment schemes

Static Runway Assignment		
Metric	**Arr**	**Dep**
Arrival Priority		
Total delay (min)	0.40	95.34
Number of flights delayed	2	47
Average delay (min)	0.20	2.03
Max delay (min)	0.20	5.26
Event Order Priority		
Total delay (min)	37.83	44.78
Number of flights delayed	24	37
Average delay (min)	1.58	1.21
Max delay (min)	4.62	3.35
Min Conflict Priority		
Total delay (min)	10.71	147.32
Number of flights delayed	9	27
Average delay (min)	1.19	5.45
Max delay (min)	4.73	21.51

Flexible Runway Assignment		
Metric	**Arr**	**Dep**
Event Order Priority		
Total delay (min)	36.17	20.02
Number of flights delayed	24	30
Average delay (min)	1.51	0.67
Max delay (min)	3.86	2.38
Min Conflict Priority		
Total delay (min)	21.83	69.64
Number of flights delayed	14	26
Average delay (min)	1.56	2.68
Max delay (min)	4.27	18.08
Min Conflict Priority / Expanded Route Set		
Total delay (min)	21.56	16.00
Number of flights delayed	12	17
Average delay (min)	1.79	0.94
Max delay (min)	4.98	3.742

The right-hand side of Table 3 provides additional summary statistics for the case when additional routing degrees of freedom are allowed to be utilized by the scheduler. Comparing the Event Order Priority heuristic results, one sees that the effect of flexible runway assignment is to reduce the total delay from 82. 5 to 56 minutes. Likewise the Minimum Conflict Priority heuristic shows a reduction from 158 down to 92.5 minutes when Flexible Runway Assignment is used. Expanding the Route Option Set under the Minimum Conflict Priority heuristic results in a further reduction in total delay to 38 minutes – a 76% reduction from the static runway/single route option case.

Per-flight and 10-minute moving average delays obtained when evaluating Algorithm II for the case where the default static runway assignment scheme is used are shown in Fig. 8(a)-(c). Here, the delay metric is shown for three different flight ordering heuristics: Arrival Priority, Event Order Priority, and Minimum Conflict Priority – as defined in the previous section. In all these cases, the Route Option Set for each flight was restricted to consist of a single route, namely that corresponding to the shortest-duration unimpeded path through the airport. In Fig. 8a, similar to the results obtained for Algorithm I, when arrivals are given priority they are serviced with nearly zero delay while departures absorb significant delays. In this case, less than half the departures incur delays less than 100 seconds, and the average delay tends to grow with time as delays compound due to the fact that the single-runway capacity is insufficient to serve the rate of departure demand. In contrast, when Event Order priority is used as the basis of scheduling the flights, the resultant delays are significantly more evenly distributed between arrivals and departures – in fact, the observed departure delay is somewhat lower than the arrival delay on average, with nearly 75% of the departures now receiving delays less than 100 seconds in duration. It is significant to note, however, that the magnitude of delay imposed on several of the arrivals by the scheduler in this case is nearly 5 minutes with several others requiring on the order of 2-3 minutes of delay. Clearly, such delay requirements would need to be coordinated significantly ahead of the aircraft reaching the terminal airspace region to facilitate absorption of delays of that magnitude. Finally, Fig 8c shows equivalent results obtained using the Minimum Conflict Priority heuristic. Due to the limited routing degrees of freedom available, this heuristic in this case results in a small number of flights

absorbing the brunt of the total delay – the maximum delay incurred is approximately 20 minutes. Such a result reinforces the need to consider not only the relative tradeoff in delays between arrivals and departures but also issues related to the distribution of delay across individual flights. If such flights were low priority to the airline, such a solution might be acceptable; in other cases, such a huge disparity in delay distribution would be cause for complaint by the offended NAS user.

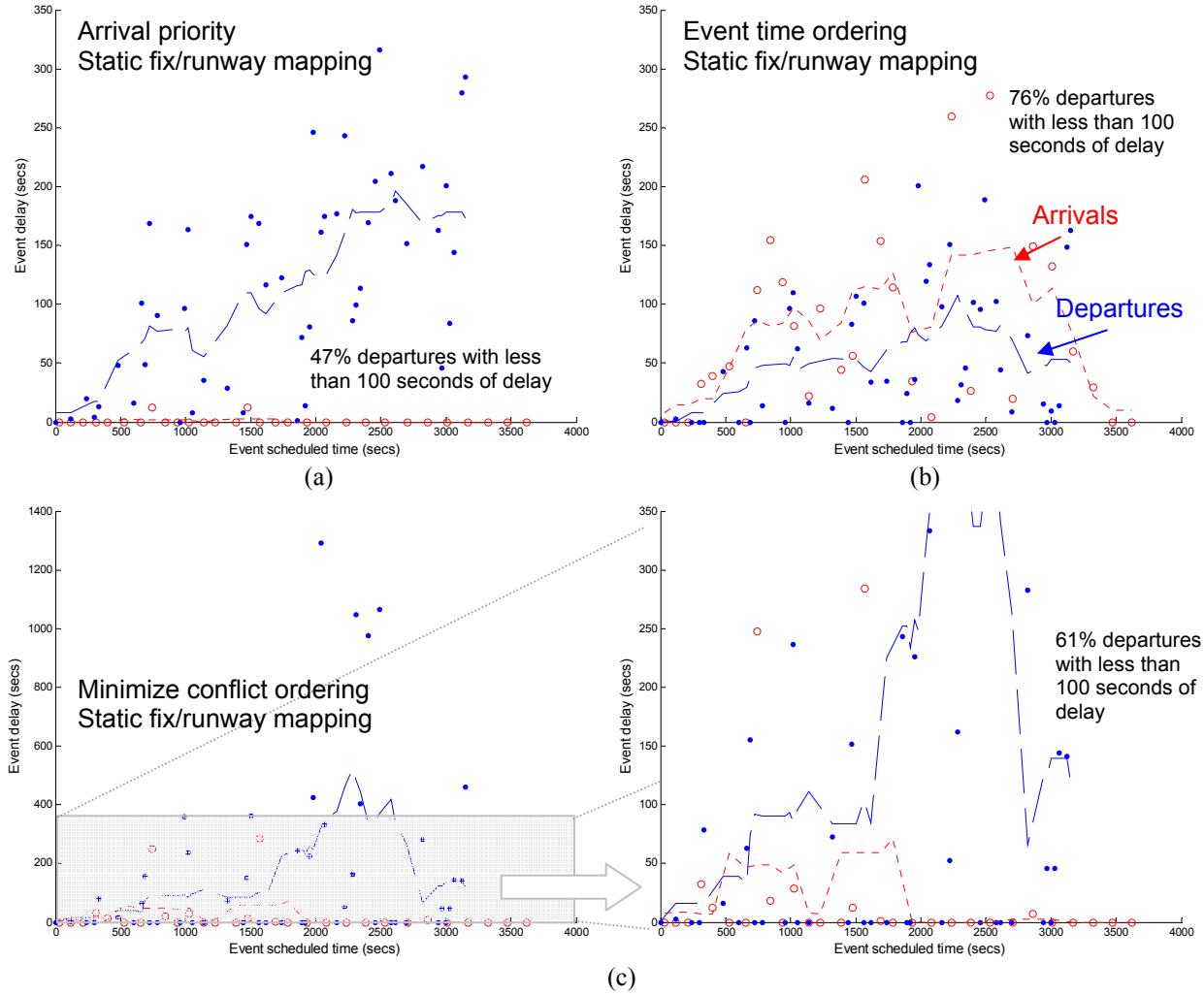


Figure 8 Per-flight delays and 10-minute moving average delay trends associated with various ordering heuristics when using the default static runway assignment scheme.

In order to investigate the relative improvement in the quality of solution possible by utilizing additional degrees of freedom with respect to routing, we repeated several of the cases in Fig. 8 using the Flexible Runway Assignment heuristic. Under this routing heuristic, the Route Option Set for each flight was expanded to include the shortest-time viable route for each aircraft from either of the two active runways. Fig. 9a and 9c show the delay results obtained using this scheme for the Event Order and Minimum Conflict Priority ordering schemes. Comparing Fig. 9a with Fig. 8b and Fig. 9c with Fig. 8c, it can be observed that the scheduler is successful in taking advantage of the additional route option. In the case of the Minimum Conflict Priority heuristic, the total delay over all flights is reduced from 158 minutes to 91.5 minutes. Further, the delay is spread more evenly over more flights, dropping the average departure delay by approximately 3 minutes. The maximum delay absorbed, however, was only slightly reduced from 21 down to 18 minutes. Figure 9b depicts the delay distribution observed for Minimum Conflict Priority ordering and expansion of the Route Option Set to include all routes containing at least one different node than the shortest-time route. In essence, this expansion corresponds to a search over all possible routes between each flight's initial and final vertex. Nonetheless, the results obtained are significant in the sense that the total delay absorbed is drastically reduced (from 158 down to 38 minutes, or 76%). Further, the distribution of delay is much more consistent across both arrivals and departures than in the previous Minimum Conflict cases. Again, however,

several of the arrivals are assigned significant delays relative to their nominal runway time. Still, the arrival delays are on par with the values obtained using the other schedule ordering heuristics while the departure delays are significantly reduced. Thus, the minimum conflict heuristic – when used in conjunction with sufficient routing degrees of freedom – seems capable of developing quality routing solutions from a delay perspective.

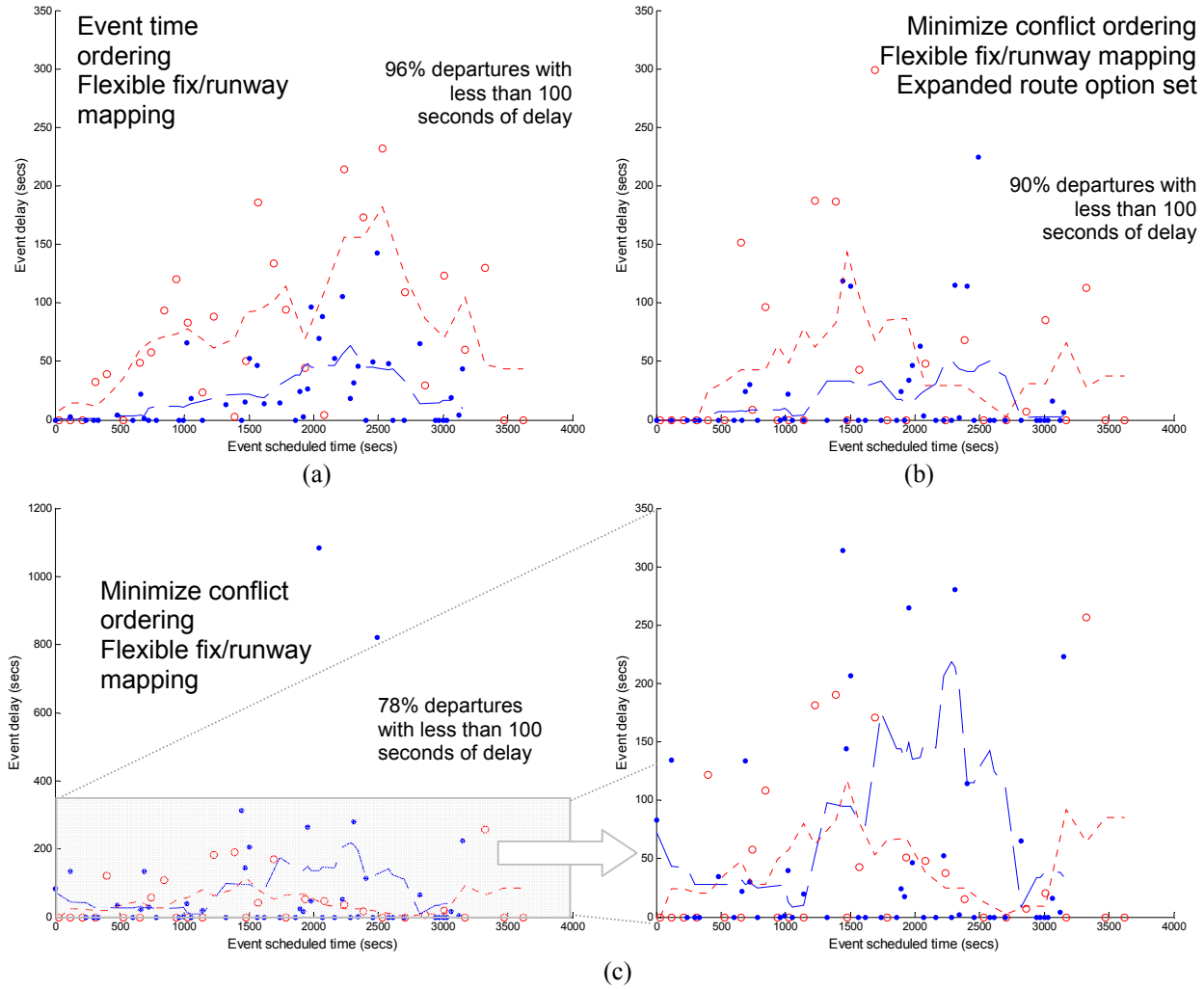


Figure 9 Per-flight delays and 10-minute moving average delay trends associated with various ordering heuristics when using a Flexible Runway Assignment scheme

Figure 10 compares the net runway throughput achieved in each 5-minute time bin for the cases described above in Figure 9.

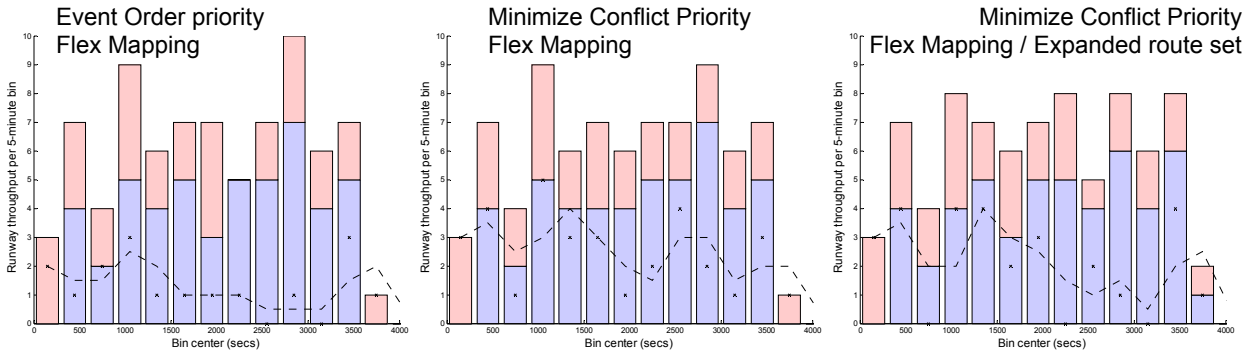
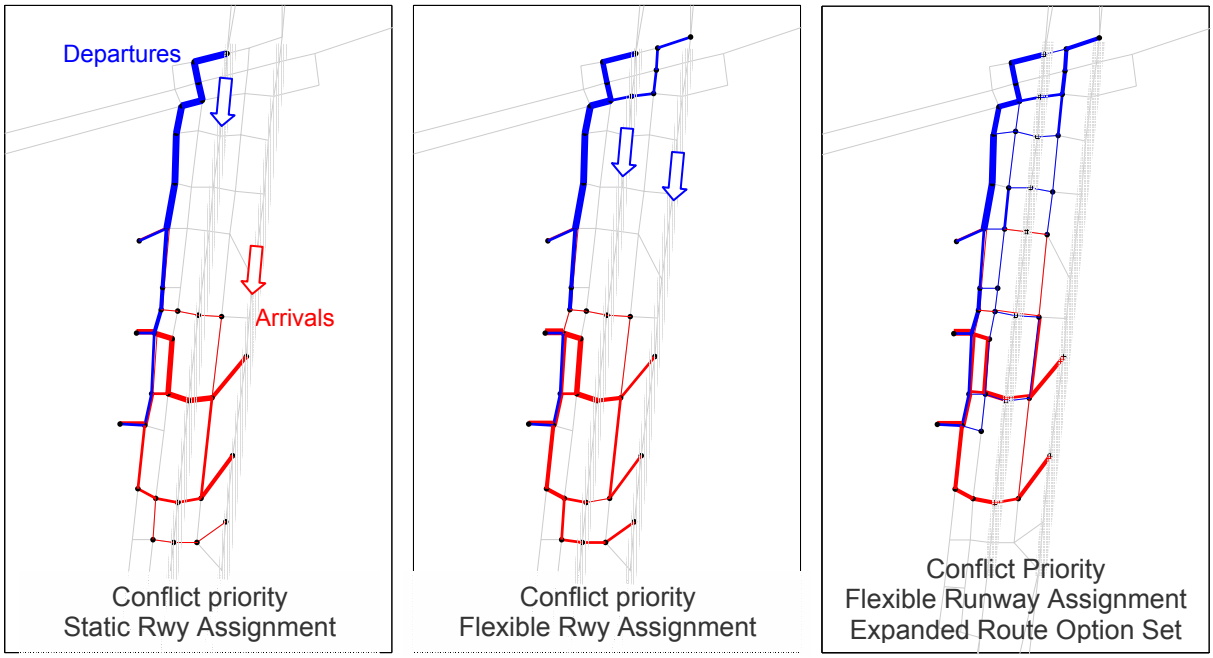


Figure 10 Comparison of net airport runway throughput achieved under Event Order and Minimum Conflict Priority ordering schemes and expanded routing degrees of freedom

In general, it is observed that the resulting runway throughputs are roughly comparable, although the time density of non-standard assignments is seen to be slightly lower on average for the Event Order heuristic.

Clearly, the Conflict Priority ordering heuristic shows a drastic improvement in terms of the reduction in delay achievable as flexible runway assignments and additional route options are added to the search space. However, a natural question is at what cost these improvements in efficiency are achieved in terms of the relative complexity of the resulting surface movements. For this purpose, we interpret complexity as the spatial distribution of routes or, equivalently, as the number of distinct routes assigned to arrival and departure flights between a given initial and final vertex. In order to assess this tradeoff, Fig. 11(a)-(c) depict the surface resource usage frequency observed across the three Conflict Priority runs. In these figures, airport links colored blue are those used by departures while those in red are those used by arrivals. Note that several of the resources are used by both arrivals and departures. The thickness of the link depicts the relative frequency of usage across the set of arrival and/or departure flights, respectively. In the case of the static fix mapping (Fig. 11a), it can be observed that only a single taxi route was assigned to all departing flights and that the majority of arrival flights utilized the closest viable runway exit. When flexible runway assignments are possible (Fig. 11b), the assignment of a fraction of the departing flights to the outboard runway (22R) is readily observed. When the scheduling algorithm is allowed to search over the largest possible route set (Fig. 11c), it can be seen that the number of distinct arrival and departure routes is increased dramatically with an associated increase in the fraction of airport resources that are used by both arrival and departure operations.



(a) (b) (c)

Figure 11 Comparison of surface resource usage frequency for various fix mapping and route set options

V. Conclusions and Future Research

This paper has explored a number of different algorithmic approaches to the planning of conflict-free trajectories on the airport surface. Although no "silver bullet" was discovered, a number of behavioral characteristics of the various algorithms were identified. The performance of these algorithms and their behavior will likely vary depending on the spatial characteristics of the airport being studied. Future studies will be performed to evaluate these algorithms at a number of different airport geometries.

Having evaluated these algorithms based on the assumption of perfect information, the next logical step is to assess the degree to which various sources of uncertainty (navigation, event timing) tend to degrade the quality of the solution obtained. Such an assessment will then naturally lead to the development of requirements to both adjust the information or decision-flows that lead to uncertainty and for dynamic replanning of trajectories in order to compensate for uncertainty as it occurs. It is anticipated that the planning algorithms described herein would be applied in a sliding window fashion in which the airport state would be sampled and used as the basis of planning

over a finite horizon. Adjustment of the plan could either be triggered on a periodic basis or by observation of events that result in significant reduction in plan utility or the potential for conflict.

Several enhancements to the algorithms themselves are planned, including establishing a maximum delay threshold that can be absorbed by an aircraft in a given state. In this fashion, we can mitigate the excessive delays that were observed to be assigned to certain aircraft, particularly arrivals, that would likely be difficult for controllers to achieve. Another area to be investigated is the degree to which explicitly incorporating buffers into the scheduling of resources can help mitigate the effects of certain classes of uncertainties such as navigation errors.

Acknowledgments

This research was funded by NASA Ames Research Center contract NAS2-02071. The guidance of our NASA Technical Monitor, Yoon Jung, Ph.D., is greatly appreciated; his guidance helped direct and focus the effort. We also wish to acknowledge the support of the Virtual Airspace Modeling and Simulation (VAMS) project and the VAMS project manager, Mr. Harry Swenson.

References

[1]Atkins, S., and Brinton, C., "Concept Description and Development Plan for the Surface Management System," *Journal of Air Traffic Control*, 2002.

[2]EuroControl, *EATMP Validation Data Repository*, https://www.eurocontrol.int/eatmp/vdr, [cited August 5, 2004]

[3]Capozzi, B., "Automated Airport Surface Traffic Control", *AIAA Guidance, Navigation, and Control Conference*, Austin, TX, August, 2003.

[4]Gotteland, J-B, Durand, N., Alliot, J-M, and Page, E., "Aircraft Ground Traffic Optimization," *4th USA/Europe Air Traffic Management Seminar*, Santa Fe, NM, Dec., 2001.

[5]Brinton, C., Krozel, J., Capozzi, B., and S. Atkins, "Improved Taxi Prediction Algorithms for the Surface Management System", *Proc. of the 2002 AIAA Guidance, Navigation, and Control Conference*, Monterey, CA, August 2002.

[6]Smeltink, J.W., Soomer, M.J., P.R. de Waal, P.R., van der Mei, R.D., "Optimisation of Airport Taxi Planning", URL: http://www.math.vu.nl/sto/publications/2003-9.pdf, Selected Preprints, [cited March 21, 2004].

[7]Visser, H.G., and P.C. Roling, "Optimal Airport Surface Traffic Planning Using Mixed Integer Linear Programming", *Proceedings of the AIAA's 3rd Annual Aviation Technology, Integration, and Operations (ATIO) Technical Forum*, Denver, Colorado, 2003.

[8]Nilsson, N.J., Principles of Artificial Intelligence, Tioga Pub. Co., Palo Alto, CA, 1980, Chapter 2.