Evaluation of Runway-Assignment and Aircraft-Sequencing Algorithms in Terminal Area Automation

Herman Vandevenne and Mary Ann Lippert

■ **The Federal Aviation Administration has responded to the steady growth of air traffic and the ensuing increase in delays at airports by initiating new programs for increasing the efficiency of existing air traffic control facilities. The Terminal Air Traffic Control Automation (TATCA) program is intended to increase efficiency by providing controllers with planning aids and advisories to help them in vectoring, sequencing, and spacing traffic arriving at busy airports. Two important algorithms in this system allocate arrivals to multiple runways and set up the best sequences for landing aircraft. This article evaluates the potential for such algorithms to achieve higher throughput with less delay. The results show that, at airports with multiple active runways, the introduction of algorithms for systematic allocation of runways increases throughput considerably. These algorithms are in fact more effective than algorithms that aim at generating optimal landing sequences based on aircraft weight-class inputs. This result is fortuitous because algorithms for optimal sequencing are significantly more difficult to implement in practice than are algorithms for runway allocation. This study also provides a scientific basis for estimating future benefits of terminal automation by using traffic models patterned on actual recorded traffic-flow data, and by proposing a unified method for assessing performance.**

The air traffic control system of today has changed little over the last three decades. During this same period of time, however, the number of aircraft at all major airports has grown enormously, and this increasing traffic is putting a severe strain on air traffic controllers. The Federal Aviation Administration (FAA) is introducing computer automation with planning and decision-making capabilities to assist controllers and increase their productivity. One such automation program is the Terminal Air Traffic Control Automation (TATCA) program. TATCA algorithms are being developed for the FAA by the National Aeronautics and Space Agency (NASA) Ames Research Laboratory. The prototype system is called the Center TRACON Advisory System, or CTAS (where TRACON stands for Terminal Radar Approach Control). This system is being tested and reengineered for field deployment by Lincoln Laboratory. The first installations are planned for Denver International Airport in 1995 and Dallas–Fort Worth International Airport in 1996.

CTAS operates within a radius of approximately 200 nmi of the airport, thus covering part of the en route Air Route Traffic Control Center (ARTCC) and all the TRACON area. The function of CTAS is to assist air traffic managers and controllers by enhancing their situational awareness of present and future traffic flow and weather, and by providing them with specific and efficient plans for handling large numbers of landings and departures to and from multiple runways. These plans are displayed in the form of continuously updated timelines of scheduled landings and departures on the en route controllers' plan-view displays, together with advisories for turns or speed reductions in the TRACON.

Underlying this planning capability is a set of software processes for (1) predicting flight times, based on flight-plan information or a new proposed trajectory, taking into account weather, wind, aircraft information, and airspace constraints; (2) organizing

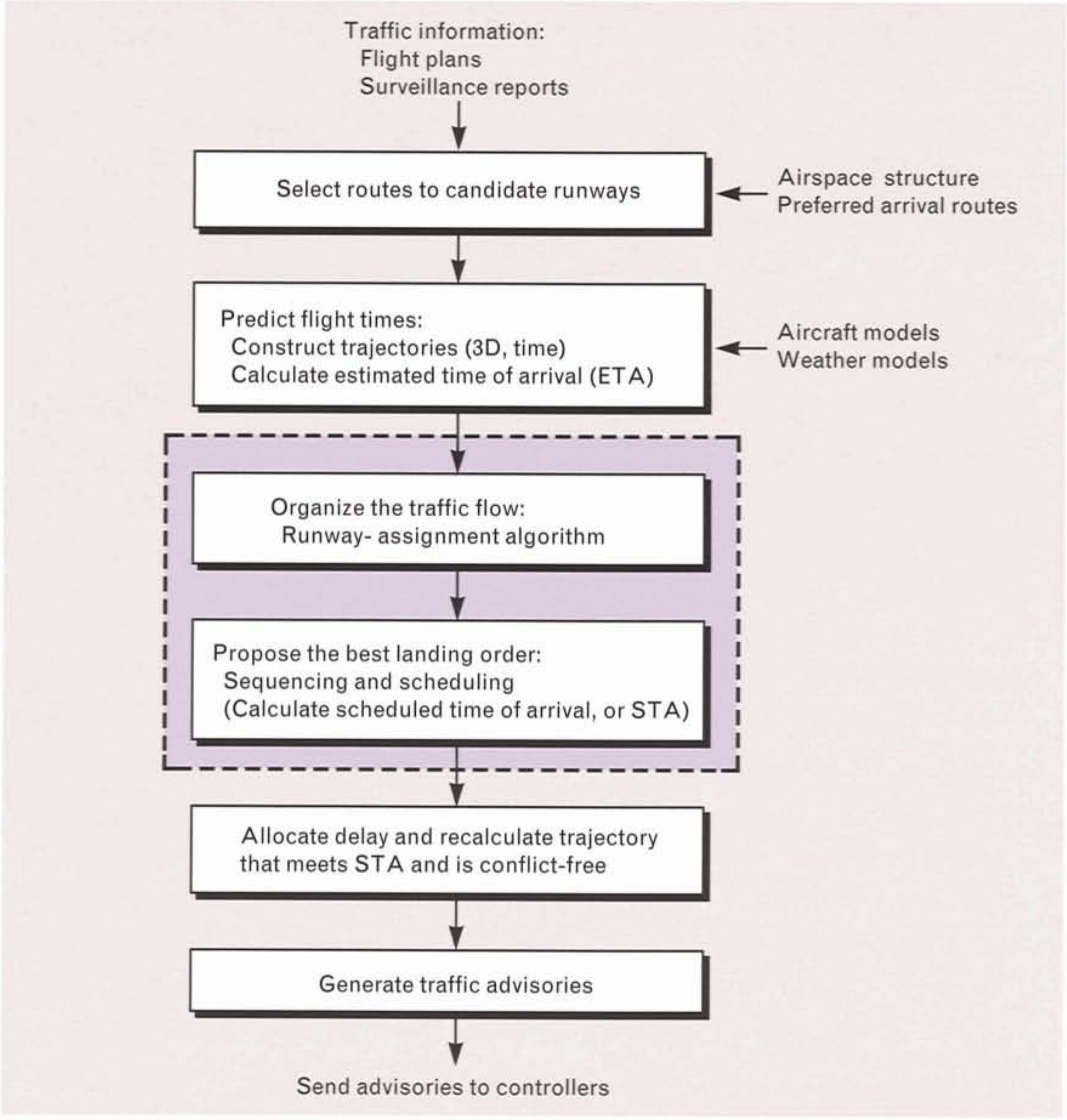

Traffic information:
Flight plans
Surveillance reports
Select routes to candidate runways
Airspace structure
Preferred arrival routes
Predict flight times:
Construct trajectories (3D, time)
Calculate estimated time of arrival (ETA)
Aircraft models
Weather models
Organize the traffic flow:
Runway- assignment algorithm
Propose the best landing order:
Sequencing and scheduling
(Calculate scheduled time of arrival, or STA)
Allocate delay and recalculate trajectory
that meets STA and is conflict-free
Generate traffic advisories
Send advisories to controllers

FIGURE 1. Principal processes of the Center TRACON Advisory System (CTAS). This system assists air traffic managers and controllers by predicting flight times, organizing the traffic flow, selecting the best runway for each aircraft and proposing the best landing sequence, and creating timely advisories to aid the controller in meeting the proposed landing times. The functions specifically described in this article are highlighted inside the dashed lines.

the traffic flow to balance the load over the metering fixes and select the best runway for each aircraft, within prescribed limitations; (3) proposing the best landing sequence and schedule; and (4) creating timely advisories to the controller to assist in meeting the proposed landing times. Figure 1 illustrates these four software processes.

Two important CTAS algorithms are imbedded in processes 2 and 3 for allocating arrivals to multiple runways, and for creating the optimal landing sequence and schedule, taking into account the wake-vortex spacing constraints at the runway threshold. This article describes the results of computer simulations used to evaluate the potential for these two algorithms to achieve higher throughput with less delay.

The notions of optimality and performance are difficult to quantify. In the next section we define a framework for discussing system performance and performance improvements. The section entitled "Evaluation of Scheduling Algorithms" discusses optimal sequencing and what can be expected in terms of increased performance over the common first-come-first-served sequencing method. Finally, the section entitled "Evaluation of Runway-Assignment Algorithms" discusses runway-assignment algorithms and their potential for enhancing performance.

Characterization of Performance

The stated purpose of CTAS is to assist controllers with information and advisories in order to increase aircraft fuel efficiency, reduce delays, provide optimal aircraft sequencing and separation, and improve airport capacity [1]. These stated goals are clearly inter-

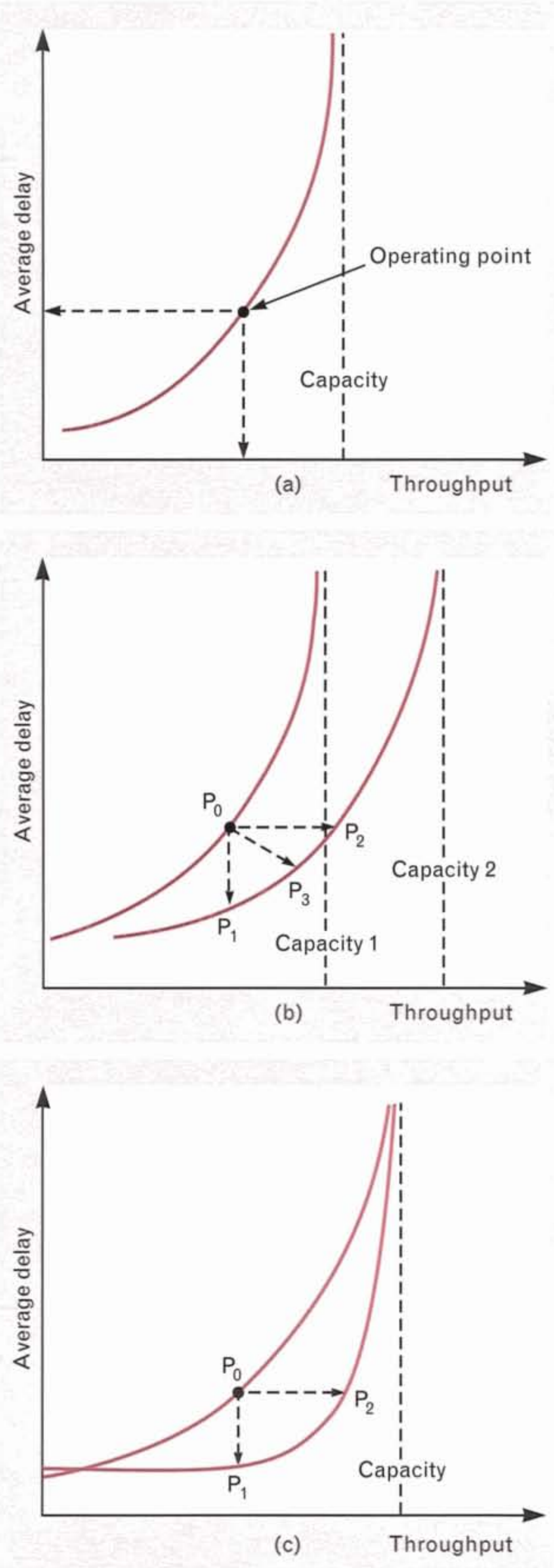


FIGURE 2. The operating curve shows relationships between long-term average throughput and average delay. (a) A specific combination of throughput rate and delay is called an *operating point* on the curve. The maximum throughput for a given operating curve is called the *capacity*. (b) The operating curve shifts to the right when capacity is increased. Operating point P_0 shifts to the new point P_1, P_2, or P_3, each resulting in a different expression for performance improvement: delay reduction, throughput gain, or a combination of the two, respectively. (c) The operating curve shifts downward when methods are used to reduce delays (by efficient runway allocation or by allowing aircraft to be expedited to meet a scheduled time).

related; for example, reducing delays will increase fuel efficiency, and optimal sequencing will reduce delays and increase capacity. A more unified measure of performance—borrowed from queuing theory—is the so-called *operating curve* in which average scheduling delay is plotted versus throughput.

Figure 2 shows three examples of operating curves. The maximum throughput for each of these curves is called the *capacity*; it is illustrated by a vertical dashed line in Figure 2(a). A choice of desired throughput, which is defined as the long-term average landing rate, results in an average delay as specified by the operating curve. Conversely, the specification of a tolerable delay limits the sustainable throughput rate that is achievable. A combination of throughput rate and delay is called an *operating point* on the curve.

We can use the operating curve as a vehicle for comparing the performance of different sequencing methods or for evaluating the effects of a particular runway-assignment algorithm. Figure 2(b) illustrates how the operating curve is expected to shift to the right when sequencing methods that increase capacity are used. Figure 2(c) illustrates how the operating curve can be lowered by applying methods that reduce delays (for example, by applying more efficient runway allocation). Figures 2(b) and 2(c) show that, given a specific operating point on the curve, we can speak of performance improvement as a throughput increase (for constant average delay) or a delay reduction (for constant throughput), or a capacity increase, or any combination of increases as long as the two operating points being compared are on the appropriate curves.

We can make several observations on the use of operating curves to express performance; these observations relate to (1) long-term stable conditions, (2) delay reductions for systems with different capacities, (3) the duration of the traffic sample and the quality of measured performance, and (4) the degree of randomness in the traffic sample. Each of these observations is explained in greater detail below.

Long-term stable conditions. Operating curves can be used to express only long-term (statistically) stable conditions. Under such conditions the average arrival rate (or demand rate) ultimately equals the throughput rate (because all aircraft eventually land, possibly after long delays) and the average arrival rate must therefore be less than or equal to the capacity. This statement does not imply that for short periods of time the arrival rate cannot exceed capacity, as long as this excess is balanced by other periods in which the arrival rate is less than capacity, so that the long-term average relationship holds.

Delay reductions for systems with different capacities. Let us look at a situation in which long-term arrival rate can exceed capacity. When we compare two system implementations that result in different capacities, we can obtain rather arbitrary delay improvements, depending on the choice of operating point. For example, Figure 3 shows that if the average arrival rate is held at value A (below both capacity 1 of the first system and capacity 2 of the second system), then a finite delay ratio is obtained. If we increase the arrival rate to value B (between capacities 1 and 2) then the delay ratio becomes arbitrarily large, because

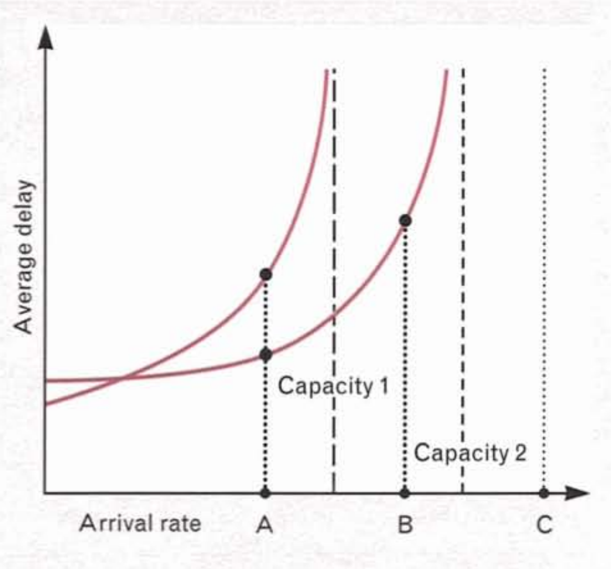


FIGURE 3. Comparing delay performance for two systems with different capacities. At arrival rate A delays are finite; at rate B the lower-capacity system will experience delays that grow indefinitely with the duration of the test since arrival rate exceeds capacity 1; at rate C this growth in delay happens for both systems but delays will accumulate much more slowly for the higher-capacity system. Note that the system with higher capacity does not necessarily have better performance over the full range of arrival rates.

the delay for the first system will increase indefinitely, given enough time. At the arrival rate set to value C, which is greater than capacities 1 and 2, delays for both systems will grow indefinitely, given a performance test of enough duration, but the delays will grow much faster for the system with the lower capacity. For any system, the long-term arrival rate must always be less than the capacity if final delay values are to be obtained. For more information, we refer the reader to the appendix entitled "Illustration of Arrival Rates, Scheduling Delays, and Throughput."

Duration of the traffic sample and the quality of measured performance. We need to discuss the relationship between measured performance and the duration of the experiment designed to measure the performance. Experiments requiring the participation of controllers and/or pilots have limited duration—at most a few hours. Depending on the degree of randomness in the traffic-arrival sample, that duration could be insufficient to measure performance adequately. Let us clarify this point by using Figure 4, which shows the results of one hundred experiments of random arrival traffic (modeled by a Poisson process), in which each experiment was one-and-a-half hours in duration. We measured average delay for two different system designs; in the first design we used a first-come-first-served scheduler and in the second design we used an optimized scheduler (which is the topic of the section entitled "Evaluation of Scheduling Algorithms"). The conditions of the tests are listed in the figure caption. In the figure we can observe that the results vary greatly from test to test, although for each test the ordering of the performance for the two systems is the same.

What is disturbing about this figure is that the delay variation from test to test overwhelms the difference in performance of the two systems. If we take the test results for one system from test *a* and for the other system from test *b*, we could easily draw the wrong conclusion (i.e., the opposite of the conclusion when

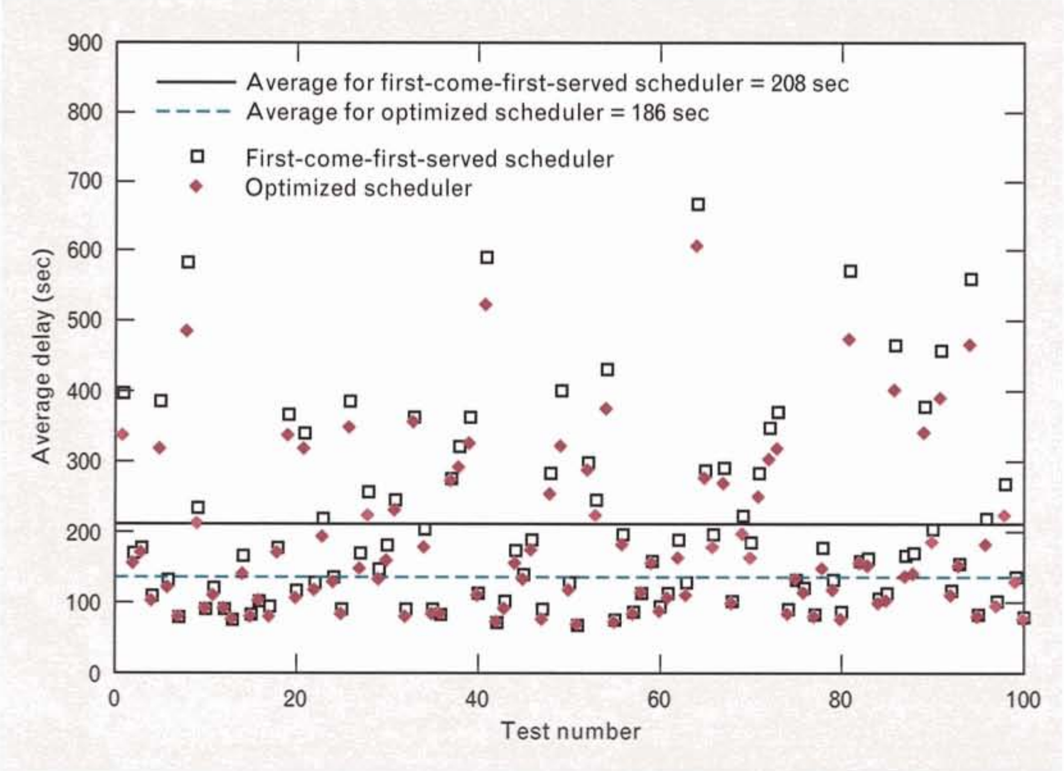


FIGURE 4. Results from one hundred test runs one-and-a-half hours in duration each for a first-come-first-served scheduler and an optimized scheduler. Each test run has a traffic sample taken from a Poisson process with an average arrival rate of 36 aircraft per hour. The mix of aircraft weight classes equals .18/.71/.11 (heavy/large/small). The calculated capacity is 39.4 aircraft per hour.

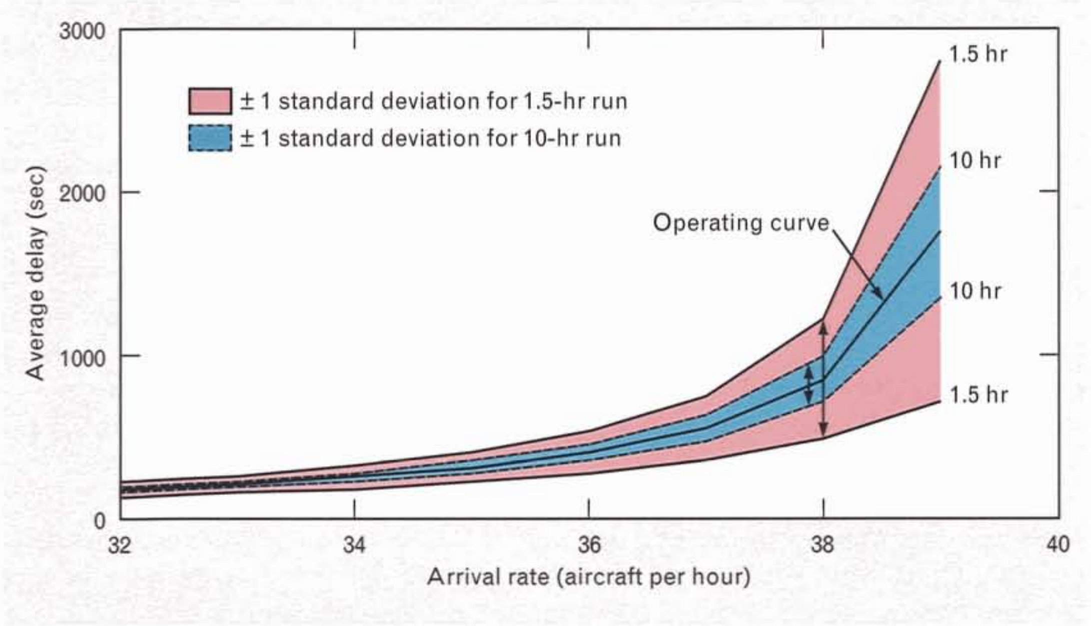


FIGURE 5. Operating curve and expected variability of measured performance for a first-come-first-served sequencing system. These curves are shown as plus or minus one-standard-deviation zones around the true mean for test durations of 1.5 hours and 10 hours.

the same test is run on both systems). Yet, in real life, we typically test one system on, say, one afternoon of traffic and the other system on the next afternoon of traffic. Although the traffic may look similar to the casual observer, we have the situation described above, with test results of relatively little value.

There are two remedies to this problem; first, use identical traffic samples when comparing systems and, second, use long-duration tests. These two remedies can be applied most easily in a computer simulation in which the identical traffic samples can be used for both systems, and tests for traffic durations of longer time periods, such as ten hours, can be completed in a fraction of that time in fast-time simulation mode.

Figure 5 shows the (true or long-term average) operating curve for a first-come-first-served system; the shaded areas around the curve represent the expected spread (in standard deviation) of measured average delay for traffic samples of one-and-a-half hours and ten hours in duration. This figure clearly illustrates that if we are going to compare systems based on operating curves, these curves must be calculated from long-duration traffic samples, and the same samples should be used for both systems. In the remainder of this study we typically use twenty-four hours of steady traffic to measure performance, with the addition of a prescribed degree of randomness.

Degree of randomness in traffic sample. The operating curve is affected by the degree of randomness in the arrival traffic stream. Clearly, if all aircraft arrived

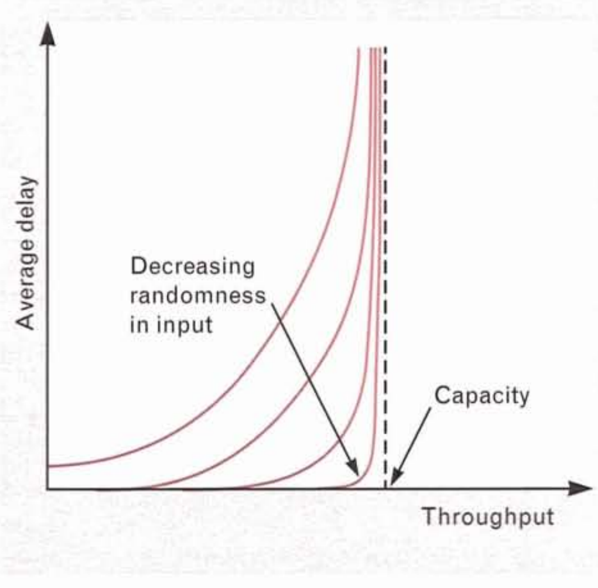


FIGURE 6. The effect of arrival-stream randomness on the shape of the operating curve.

in the terminal area neatly spaced, so that after merging onto the final approach path, they all occupied their assigned landing slot, then no scheduling delay would be incurred in the terminal area. In general, the operating curve becomes shallower with decreased randomness, as shown in Figure 6. This dependency of the operating curve on the degree of randomness is an issue for TATCA, because other FAA systems such as flow control or metering with miles-in-trail attempt to derandomize the traffic flow into the terminal area [4]. The computer simulations in this study typically used, as input, arrival traffic modeled as a Poisson process, although many of the performance tests were run by using recorded traffic from the Dallas–Fort Worth or Denver TRACONs as input.

Simulation Setup

Figure 7 shows the setup for our performance study, which is based on fast-time discrete event simulation. Depending on the algorithm to be evaluated, we must set up a number of arrival routes and a number of destinations (runways). The traffic model consists of defining arrival rates and ratios of traffic rates over all routes, and selecting the mix of aircraft types. In most cases we can describe the aircraft type by its weight class; the aircraft are labelled *heavy* if gross takeoff weight exceeds 300,000 lb, *large* if weight is between 300,000 and 12,000 lb, and *small* if weight is less than 12,000 lb. In other cases we need to be more specific about the airframe because it affects deceleration profiles and landing speeds, which in turn affect the landing separations to be selected.

We adopt the Poisson model for the distribution of the various estimated times of arrival (ETA), which are the inputs to the runway-assignment and scheduling algorithms. In this model, the ETA events are occasionally bunched (possibly causing a short-term arrival rate that exceeds capacity) and they occasionally have large gaps (possibly causing an irretrievable waste of capacity).

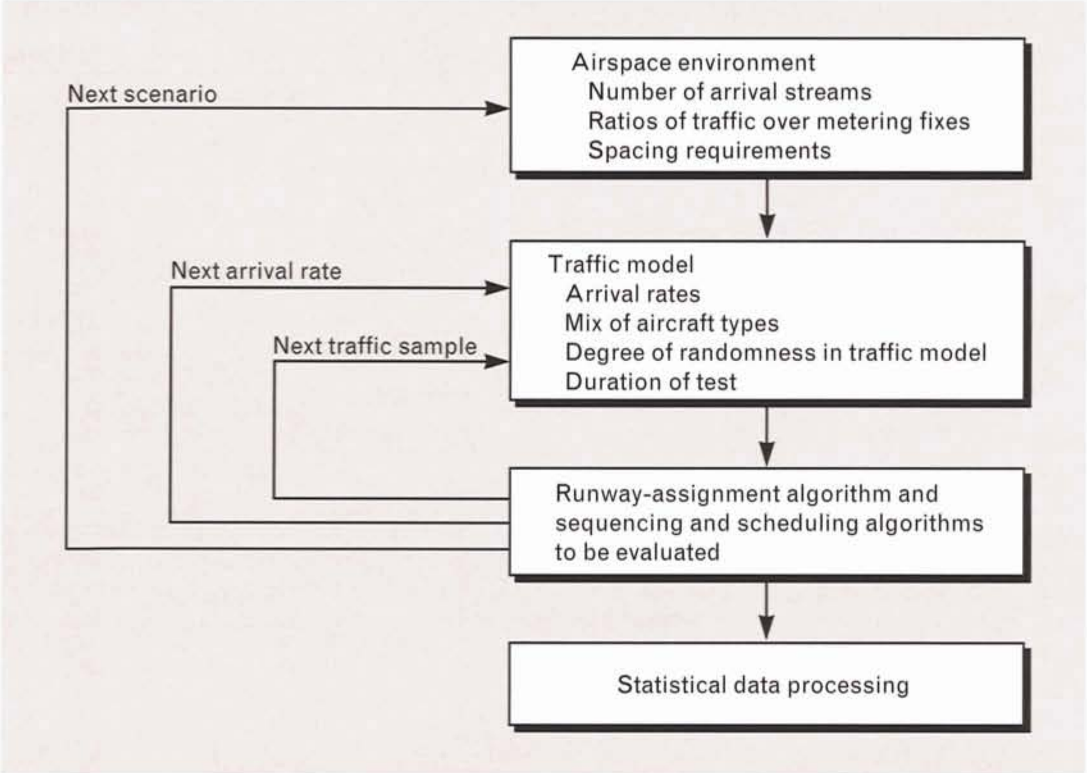

Next scenario
Airspace environment
Number of arrival streams
Ratios of traffic over metering fixes
Spacing requirements
Next arrival rate
Next traffic sample
Traffic model
Arrival rates
Mix of aircraft types
Degree of randomness in traffic model
Duration of test
Runway-assignment algorithm and sequencing and scheduling algorithms to be evaluated
Statistical data processing

FIGURE 7. Elements in the performance study, based on fast-time discrete event simulation. First the airspace environment and traffic model are defined, including a Poisson distribution for the various estimated times of arrival, and these elements are then used as inputs to the runway-assignment and scheduling algorithms being studied.

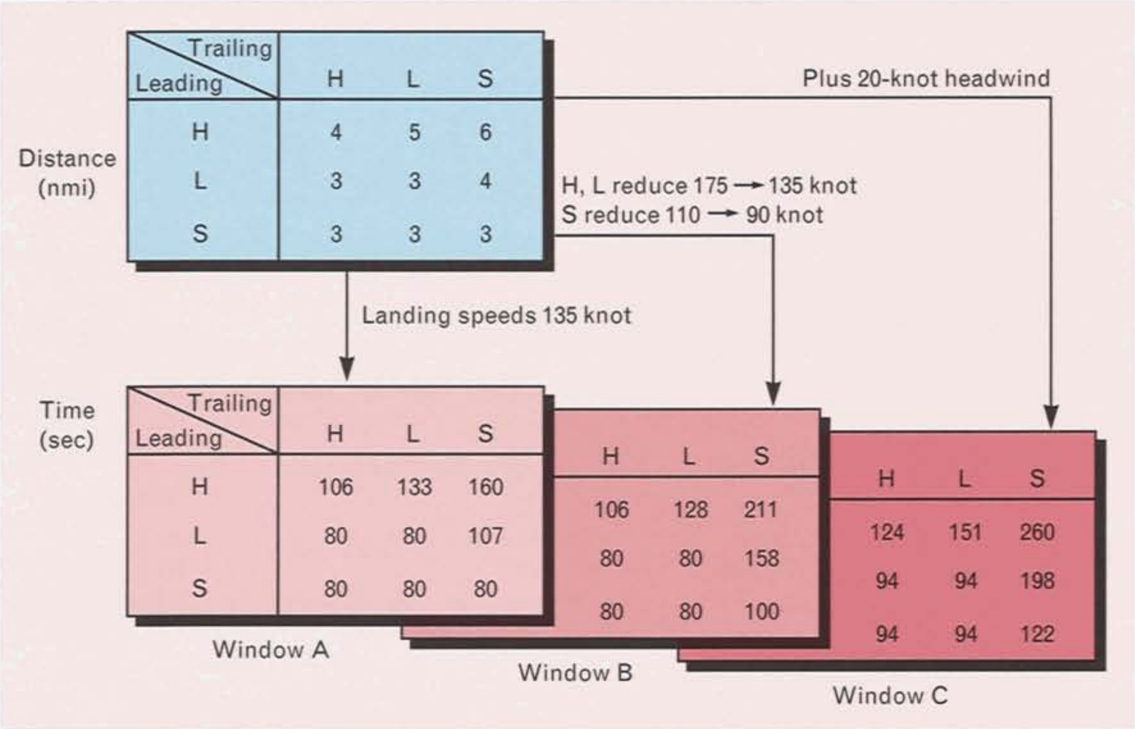
Evaluation of Scheduling Algorithms

The most common landing order scheduled by controllers is first-come-first-served, but other landing orders could be more beneficial in terms of reduced delays and increased throughput and capacity. This conclusion is the case when the landing aircraft represent different weight classes (for example, we must avoid scheduling a small aircraft after a large aircraft because this order would require an extra large separation). A scheduling algorithm that fine-tunes the landing order with the purpose of minimizing a parameter such as the average delay, or of maximizing throughput or any similar goal, is loosely referred to as an *optimal scheduling algorithm.* The algorithm performs sequencing and scheduling, where sequencing is referred to as setting up the planned landing order, and scheduling is referred to as determining the landing times, based on the minimum required separations. Often, however, we use the term scheduling to cover both functions.

In this section we discuss potential benefits of optimal sequencing and scheduling. First we establish upper bounds on the capacity increase, independently of any specific algorithmic implementation. Next we discuss the effect of a specific implementation issue, namely, the necessity for using a finite scheduling window. The use of a finite scheduling window results in a narrower bound on achievable capacity gain. Then we discuss the criteria used for selecting an optimal sequence, and finally we describe some actual algorithms and express their performance as a full operating curve.

Upper Bounds on Capacity Increase

Capacity (or maximum throughput) can be defined as the inverse of the average landing time interval when all aircraft are landing at their legal minimum



Trailing Leading	H	L	S
H	4	5	6
L	3	3	4
S	3	3	3

Trailing Leading	H	L	S
H	106	133	160
L	80	80	107
S	80	80	80

	H	L	S
H	106	128	211
L	80	80	158
S	80	80	100

	H	L	S
H	124	151	260
L	94	94	198
S	94	94	122

FIGURE 8. Minimum landing-time intervals between leading and trailing aircraft of types heavy, large, or small, as calculated for three scenarios: window A has uniform landing speeds and no wind; window B has different landing speeds for heavy and large aircraft than for small aircraft, and no wind; window C has the same conditions as in window B but with a 20-knot head wind. These landing-time intervals are all calculated from the standard (3,4,5,6) separation matrix, where the numbers 3, 4, 5, and 6 represent separation in nautical miles between heavy, large, and small aircraft.

separations. The notion of capacity is complicated because it depends on many variables. The first set of variables are those which affect the translation of the required minimum landing separation, specified by the FAA as a distance, into time units. The translation of the minimum separation into time units is necessary because the separation is used in that form by the scheduler. These variables include landing speeds, deceleration profile used by leading and trailing aircraft, wind, and the length of the common path of the two aircraft.

Figure 8 shows an example of how the standard (3,4,5,6) matrix (where the numbers 3, 4, 5, and 6 represent the FAA-specified minimum landing separation distance in nautical miles between heavy, large, and small aircraft) is translated into time separations for three choices of values for the indicated variables. The average separation (from which capacity is derived) then depends on the mix of aircraft types occurring in the arrival stream for which the algorithm is setting up a schedule. This frequency of use involves a second set of variables that describe the mix of aircraft type, namely, the percentages of heavy, large, and small aircraft. Any aircraft is mapped into one of these three weight classes before the required separation can be selected from the separation matrix on the basis of the type of the present aircraft and the succeeding aircraft in the proposed sequence.

In a first-come-first-served landing order the relative frequency of occurrence of certain aircraft pairs is determined by the mix of aircraft types, and the capacity is therefore easily calculated. In other scheduling algorithms in which the sequence is manipulated to achieve certain goals, the calculation of capacity can be difficult. Given a separation matrix and an aircraft mix, however, we can calculate an upper bound to the smallest average separation distance (and hence the largest capacity) independently of the algorithm by considering the best possible reordering of aircraft (away from first-come-first-served sequencing) that achieves the calculated upper bound.

Figure 9 shows such an upper bound as a function of aircraft mix, where the fraction of heavy aircraft is plotted on the *x*-axis. The separate curves represent specific choices for the fraction of small aircraft. The separation matrix used for the calculation of this upper bound is the one shown in window B of Figure 8. Figure 9 shows that, given the choice of matrix, the gain in capacity will be less than 20% even for the most favorable mix of aircraft, whatever the optimal scheduling algorithm. In fact, for a more realistic mix of aircraft equal to .18/.71/.11 (heavy/large/small), which is a representative mix of aircraft weight classes for Logan Airport in Boston, the upper bound indicates there will be no more than a 10% gain in capacity. And most of that gain will be whittled away when some real-life constraints are taken into consideration, as discussed in the following section.

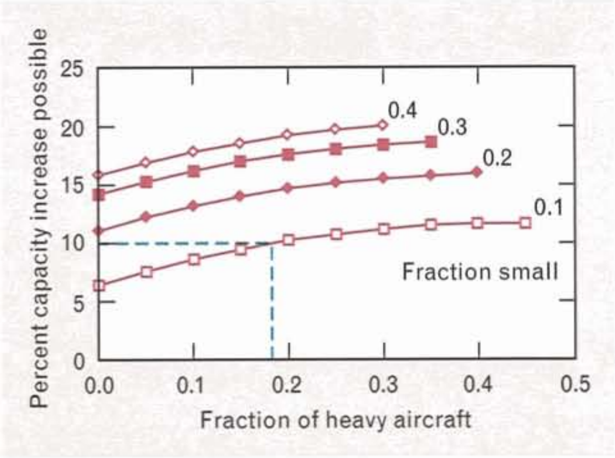


FIGURE 9. The upper bound on increase in capacity, determined by the ratio of best capacity over first-come-first-served capacity, as a function of the mix of aircraft type. The choice of separation parameters is shown in window B of Figure 8. For a realistic mix of heavy/large/small aircraft equal to .18/.71/.11 (as shown by the dashed lines), which is representative of Logan Airport in Boston, the upper bound indicates there can be no more than a 10% gain in capacity.

The Effect of Finite Scheduling-Window Size

Unlimited reordering of aircraft is allowed in the calculation of the upper bound for capacity, as described above, but in practice the set of aircraft that can be considered for reordering is limited. The CTAS system accepts an aircraft as a candidate for scheduling when it enters a zone of approximately 200-nmi radius (or about forty-five minutes of flying time) from the airport, and the aircraft's position in the landing sequence is fixed when it crosses the freeze horizon (about thirty-five minutes before landing). This window of some ten minutes of flying time before the aircraft crosses the freeze horizon is called the *scheduling window*, and it determines the set of aircraft considered for reordering at any given time. This practical restriction implies that only a fraction of the benefits achievable by unlimited reordering can be realized in an actual scheduling environment.

That achievable fraction of benefits can be estimated, as shown in Figure 10 for a specific example of aircraft mix and separation matrix (window B of Figure 8). For a ten-minute scheduling window the upper bound on improvement from *any* scheduling algorithm is approximately one aircraft per hour over the first-come-first-served capacity of 39.2 aircraft per hour, or a gain of 2.7%.

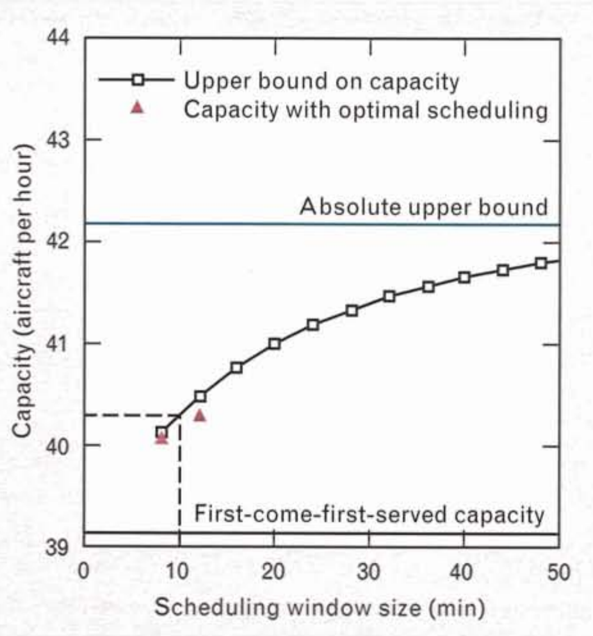


FIGURE 10. Upper bound on capacity as a function of the size of the scheduling window. The values chosen for minimum separations are those shown in window B in Figure 8. The mix of aircraft types is equal to .18/.71/.11 (heavy/large/small). First-come-first-served capacity is 39.2 aircraft per hour, and the upper bound on capacity is 42.3 aircraft per hour. Upper bound for a ten-minute scheduling window is 40.3 aircraft per hour, as shown by the dashed lines. Capacities realized by optimal scheduling algorithms (shown as red triangles) are slightly less.

Sequencing Constraints

Other constraints exist that can reduce the achievable gain further. For example, we usually cannot allow aircraft that follow a common approach path to overtake one another in order to achieve a proposed sequence, or for another example, the controller, for whatever reason, might have imposed a particular landing order on some aircraft. While the former constraint can easily be made part of the search algorithm, which in fact greatly limits the number of sequences to be evaluated, the latter must be accepted by the automated algorithm as a given. Such constraints diminish the potential benefits of an optimal sequencing algorithm, but these constraints could also make the automated sequencing solutions more acceptable to controllers. The effect of these constraints on performance depends heavily on the number of converging traffic streams, and is therefore dependent on the particular airspace configuration around the airport. The more streams and the more similar their volumes of traffic, the less the effect of the constraints and the greater the value of the sequencing algorithm.

Dynamic Aspects of Optimal Sequencing

Aircraft are sequenced and scheduled while their ETA is within the scheduling window. This sequencing and scheduling operation in CTAS is repeated nominally every twelve seconds, during which time any aircraft might leave the window or a new aircraft might enter the window. In general, the sequencing algorithm considers each aircraft numerous times (as many as fifty) to determine its relative position in the landing sequence. While the static optimal sequencing problem is easy to visualize, the end result of scheduling with a sliding scheduling window, in which the set of candidate aircraft slowly changes in time and the optimal solution of the present window can undo the optimal solution from a previous window, is more difficult to envision. The end effect of optimizations is reflected in the sequence of frozen positions of aircraft when they leave the scheduling window, one at a time. This sliding-window mechanism introduces some difficult algorithmic issues on whether to base the set-inclusion-and-freezing decisions on estimated landing times, first-come-first-served scheduled times, or optimally scheduled times.

We might also see the expected performance gain reduced by this dynamic aspect of scheduling reflected in the sliding-window mechanism. Simulation results, however, show that this decrease in performance does not occur. When the results of using a static (stepping) window were compared with the results of using a sliding window, only a very minor improvement, on the average, was noticed.

Objectives of Scheduling Optimization

Up to this point we have focused on scheduling methods and their potential for capacity increase. The upper-bound algorithm was based on unlimited reordering that could be done only in a fully saturated condition. In normal operating conditions long-time saturation rarely occurs, so there are practical limits to the possible ultimate improvements in capacity. Long-term observations over busy afternoons at Dallas–Fort Worth Airport have shown that the average long-term (five hours) throughput rarely exceeded 80% of capacity. Under those conditions the performance of the scheduling algorithm is better described by the operating curve, rather than just its upper limit (which describes the capacity).

If we focus on a specific scheduling window, with *N* aircraft to be scheduled to a single runway, we can in principle consider *N*! possible sequences, most of which would not make any sense operationally. The computer must quickly search through the possible sequences and identify the feasible ones, and out of these feasible sequences identify the optimal one according to some criterion. Many criteria have already been mentioned, including highest throughput, minimum total scheduling delay (possibly with delays for some aircraft types weighted more heavily than others), minimum value of a function of delay (e.g., quadratic), and minimum fuel burn. Even if these criteria are not completely independent (because the operating curve shows the interdependency of average delay, average throughput, and capacity), for a specific window the resulting sequences can be significantly different.

Figure 11 illustrates the variety of resulting sequences for a particular scheduling window. In this example we assume there are three arrival streams with no scheduled overtakes allowed within a stream. The figure shows three schedules: one resulting from a first-come-first-served (at the runway) scheduling discipline, one in which optimality is defined in terms of the minimum of the sum of the delays, and one in which optimality is defined in terms of the minimum of the sum of the squares of delay.

The choice of optimization criterion matters, especially when the dynamic aspects of scheduling are included in the consideration. For example, a linear criterion (sum of delays) could easily result in aircraft of certain types (e.g., heavy) being delayed inordinately through consecutive scheduling windows. The quadratic criterion would prevent this kind of excessive delay, as would the imposition of additional outside constraints—for example, an additional term in the criterion when the scheduled delay for an aircraft reached a threshold of some multiple of the average delay.

Figure 12 shows an example of a meaningful optimization criterion. The dependent variable (*y*-axis) represents scheduling cost, while the independent variable (*x*-axis) represents scheduling delay. The curve is called the *cost function.* The optimization criterion consists of finding a landing sequence that minimizes the total cost of all aircraft being scheduled. Delay here is defined as the time difference between the time an aircraft could have landed if no other aircraft were around and the time the aircraft is scheduled to land in the presence of all other aircraft (minimum required separations are maintained).

A negative delay means that an aircraft will land earlier than it would nominally; for example, the controller instructs the pilot to keep up approach speed or to cut short the usual downwind-upwind trombone-shaped path. Even negative delay costs a little because the procedures involved are not fuel-optimal or nominal, and they may not be conducive to passenger comfort, even though expediting one aircraft might save time on all other aircraft in the queue. A positive delay small enough to be implemented by slowing the aircraft somewhat earlier than is nominally the case contributes quadratically until it reaches a value at which speed controls no longer suffice and path stretching must be invoked. An additional cost penalty is then imposed. Finally, when positive delay reaches a second threshold at which the scheduling delay exceeds what the repertoire of path-stretching

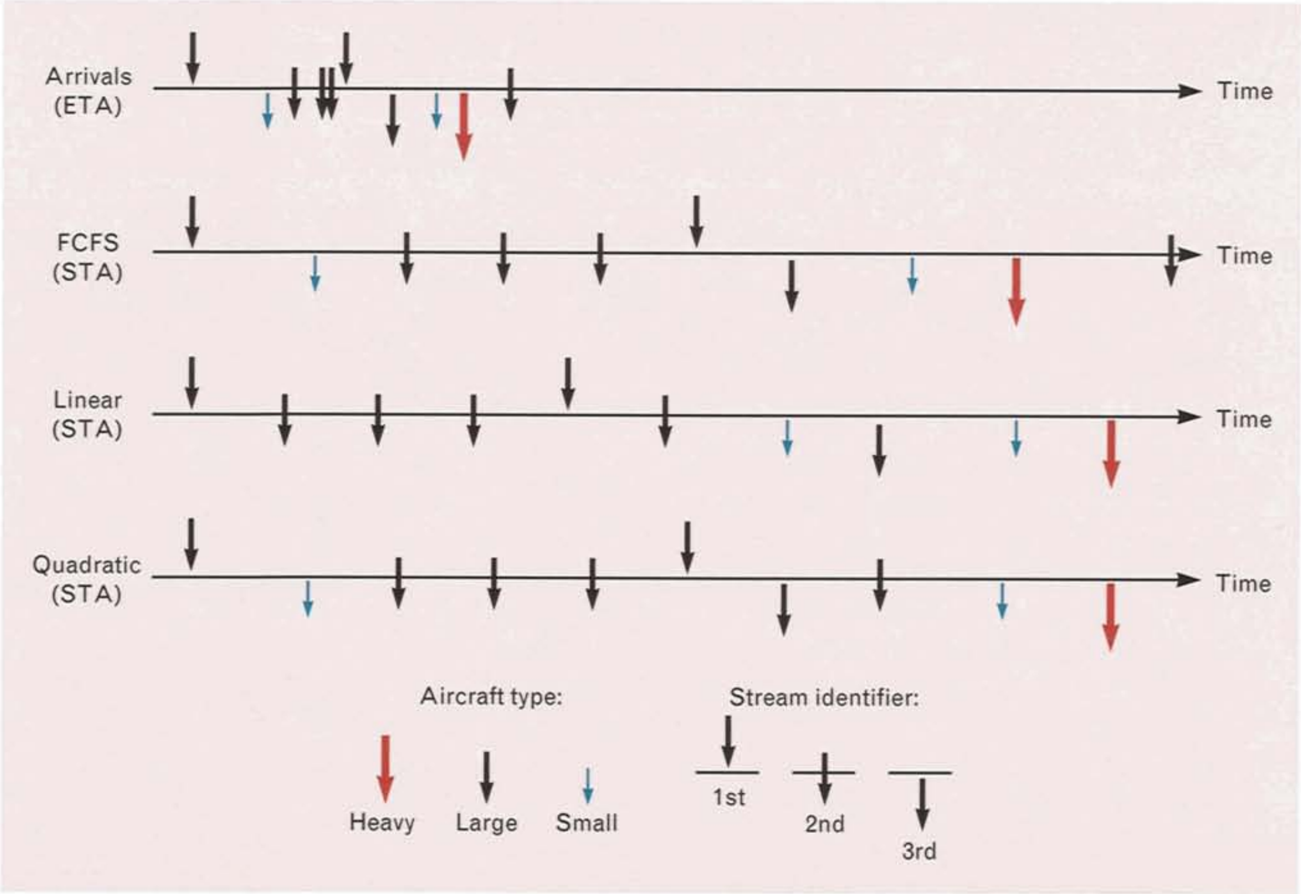


FIGURE 11. Example showing how aircraft in three different arrival streams are scheduled by using three different scheduling disciplines. The first discipline is a first-come-first-served (FCFS) scheduler, the second is a linear scheduler defined in terms of the minimum of the sum of the delays, and the third is a quadratic scheduler defined in terms of the minimum of the sum of the square of the delays.

techniques can accommodate, an additional cost term is added. This cost term is added when the delay exceeds the controllability of the terminal airspace.

Controllability is defined as the delay that can be achieved by speed reductions along the nominal path and by well-defined path-stretching procedures. The notion of controllability is crucial in an aspect of terminal automation not discussed in this article; namely, the automation must propose not only a landing time but a new trajectory (in time and space) that delivers the aircraft to the runway at the proposed landing time, and the automation must produce timely advisories to be relayed by the controller to the pilot to make the new landing time and trajectory happen. When delay exceeds controllability, the automation can no longer propose a trajectory solution and it must signal to the controller to exercise "extraordinary" measures, such as holding maneuvers, to meet the scheduled time. The implication here is that the automation is relied on to propose solutions for routine operations and the controller intervention is rarely called upon to solve unusual cases. Clearly, a proposed landing sequence that would require a particular aircraft to be delayed to such an extent would be undesirable, hence the second penalty term.

Comparing Performance of Several Scheduling Algorithms

The special-purpose simulator, which was built to study scheduling and sequencing algorithms and the effect on the algorithms of such parameters as cost functions, size of the scheduling window, and overtake constraints, was run in fast-time mode, in which all important parameters could be varied at will. The

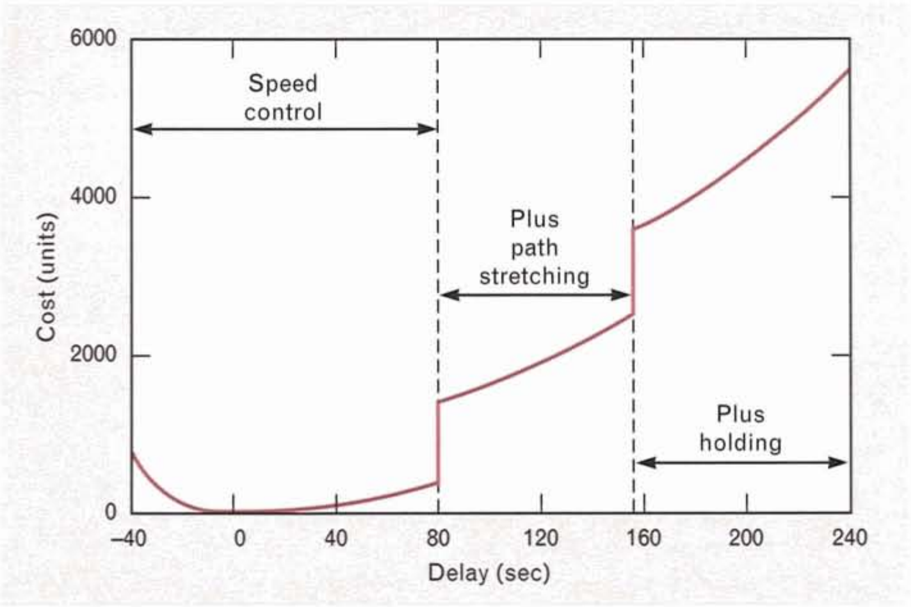


FIGURE 12. Example of scheduling delay as a meaningful optimization criterion; the curve showing the resulting scheduling cost is called a *cost function*. The optimization criterion consists of finding a landing sequence that minimizes the total cost of all aircraft being scheduled.

simulation was run on a Sun workstation in the TATCA laboratory at Lincoln Laboratory.

Rather than provide an elaborate report on all our performance tests of all the algorithms, we illustrate the results for a single choice of parameter settings. The scheduling was done for a single runway, for a time period comprising one thousand arrivals. Three independent arrival streams were examined, with a combined average arrival rate varying from 32 aircraft per hour up to capacity, with interarrival times having an exponential distribution (Poisson process). The minimum time-separation matrix was derived from the standard (3,4,5,6) distance matrix with the assumption that all aircraft have the same deceleration profile and land at 135 knots (as shown in window A of Figure 8). The average mix of aircraft type was fixed at .18/.71/.11 (heavy/large/small). The size of the scheduling window was chosen to be twelve minutes. The optimal schedulers (with linear or quadratic cost functions, and no expediting) operated with the constraint that no overtakes within a particular arrival stream are allowed. The scheduler searched over all possible sequences to find the optimal sequence, irrespective of computation requirements (which nevertheless turned out to be a critical issue for the CTAS scheduler). Figure 13 shows the resulting operating curves for the three scheduling algorithms at these parameter settings.

The most striking feature of these curves is that their upper limit, which is the capacity achieved by the system design represented by the operating curve, stays far below the upper bound. Given the parameter choices made to produce these curves, the upper bound for capacity was 42.4 aircraft per hour versus 39.4 aircraft per hour for first-come-first-served scheduling, or a scant 7.6% gain. Imposition of a finite time for the scheduling window (here twelve minutes) reduced the capacity to 40.3 aircraft per hour, or a 2.7% gain.

Observe how the operating curves for optimal schedulers with linear or quadratic cost functions are hardly distinguishable, even though we have shown that the actual sequences produced in every scheduling window can be significantly different. These operating curves clearly show that only in a condition of protracted saturation, in which the arrival rate equals or exceeds the capacity for a long time, would there be a great difference in average delay between a first-come-first-served scheduler and an optimal scheduler. For most operating conditions, saturation of short durations (half an hour or so) is insufficient to produce large differences in the delay. In fact, long-term throughput rates during so-called heavy traffic seem to average 80% of capacity.

The operating curves show that we would be hard pressed to observe any difference in performance over the range of 32 to 37 aircraft per hour, which covers the 80% to 94% range of first-come-first-served capacity from a single comparison test. Even at 32 aircraft per hour, where the delay difference is 8.9%, we must remember the warning given earlier on the variability of test results, which can invalidate the accuracy of any given measure of performance difference.

Evaluation of Runway-Assignment Algorithms

At large airports such as Dallas–Fort Worth Airport, as many as four or more runways can be simultaneously active for landing operations. Although as a general rule aircraft from a certain sector nominally land on a corresponding runway, the controllers often assign them on an individual basis to other runways to equalize the load. This act of load balancing clearly does not increase runway capacity, but it can reduce delays considerably. In fact, if we go beyond merely equalizing the rates to runways, and focus on minimizing flight times for individual aircraft, the average delay reduction becomes proportional to the number of active runways even when all runways initially had equal arrival rates.

The consequence of such algorithms to land aircraft as soon as possible is that short-term throughputs are maximized, but only at the price of heavy *crossover* traffic (i.e., aircraft that no longer go to a nominal or preferred runway). And this maximized throughput exacts another price in terms of increased controller work load, especially if executing crossovers involves sending aircraft through narrow corridors over the top of the airport, as is the case, for example, at Dallas–Fort Worth Airport. An algorithm for runway allocation must therefore aim at minimizing de-

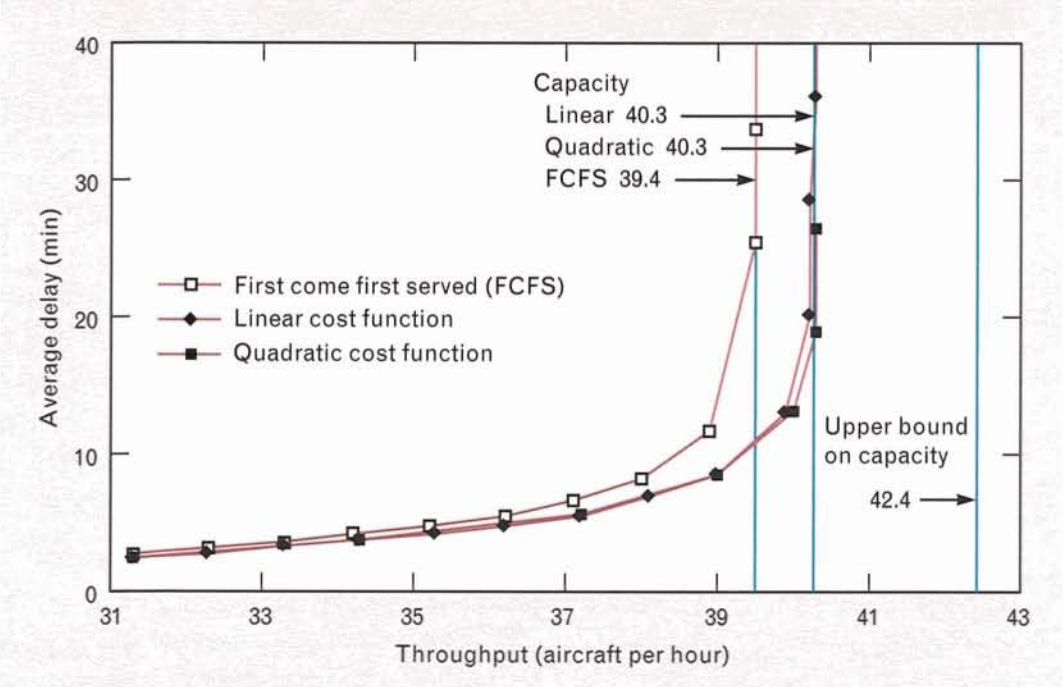


FIGURE 13. Performance curves for first-come-first-served scheduling versus optimized scheduling. The mix of aircraft types is .18/.71/.11 (heavy/large/small); separations are defined as shown in window A in Figure 8, and the scheduling window is twelve minutes. The traffic model is three arrival streams (interarrival times for each stream have a Poisson distribution) with identical fix loadings and mix. No overtakes or expediting is allowed.

lay while at the same time holding down the crossover rate.

Theoretical Bounds on Performance Improvement from Runway-Allocation Algorithms

Obviously, when arrival traffic is unbalanced, shifting the extra load to other runways greatly reduces overall delays. Less obviously, even if the arrival traffic is more or less balanced (i.e., the long-term average rates are the same), allowing individual aircraft to land on whichever of the n runways they could land on the earliest can further reduce the average delay by a factor proportional to n. We show results from two theoretical models, which are discussed in greater detail elsewhere [4].

First, with the interlanding times modeled by an exponential distribution with mean $1/\mu$, where μ is the capacity of the individual runway, we obtain a relationship between the average delay W_1, without crossovers, and W_n, with crossovers, to n runways:

$$W_n = \frac{\mu}{\lambda} P_\psi W_1 ,$$

where P_ψ is the so-called Erlang C formula, which represents the fraction of time all n runways are busy (and also the fraction of all aircraft that will experience delay), and λ is the total arrival rate.

The quantity P_ψ depends strongly on how busy the airport is through the relations

$$\rho = \frac{\lambda}{n\mu}$$

and

$$P_\psi = p_0 \frac{(n\rho)^n}{n!(1-\rho)} ,$$

where

$$p_0 = \left[\sum_{j=0}^{n-1} \frac{(n\rho)^j}{j!} + \frac{(n\rho)^n}{n!(1-\rho)} \right]^{-1} .$$

For an airport with two runways at 90% capacity ($\rho = 0.9$), $P_\psi = 0.853$ and $W_2/W_1 = 0.472$. Similarly, for three runways, $P_\psi = 0.817$ and $W_3/W_1 = 0.3026$. In this exponential interlanding time model the average delay is given by

$$W_1 = \frac{\rho}{\mu(1-\rho)} .$$

In a second model with constant interlanding time, we put the interlanding times equal to the statistical average obtained from the matrix of time separations and the appropriate aircraft mix. We can then prove that

$$W_n \cong \frac{P_\psi}{n} W_1 ,$$

which is virtually the same relationship as the one shown above. In this case, however, the average delay is

$$W_1 = \frac{\rho}{2\mu(1-\rho)} ,$$

i.e., half of what it was with the exponential model.

The important result is that, whatever the distribution of interlanding times, roughly the same performance improvement can be obtained when aircraft are allowed to land at their earliest convenience on any available runway. The price for this improvement is that, for the example above for two runways, approximately 50% of the aircraft will execute crossovers. In the next section we discuss algorithms that attempt to strike a balance between delay reduction and crossover-rate increases.

A Runway-Assignment Algorithm

We propose a simple runway-assignment algorithm based on the following premises: (1) for each aircraft the set of candidate runways and the ETAs to these runways are known, and (2) for each aircraft there is a preferred runway. The mechanism for limiting crossovers is a simple time threshold T. When an aircraft has its turn to be assigned a runway, it is tentatively scheduled at first on all its candidate runways. The aircraft is then assigned its preferred runway unless the schedules indicate it could land at least T minutes earlier on an alternate runway. If several such options exist, then the alternate runway with the earliest scheduled time is assigned.

Figure 14 shows a simple three-runway example

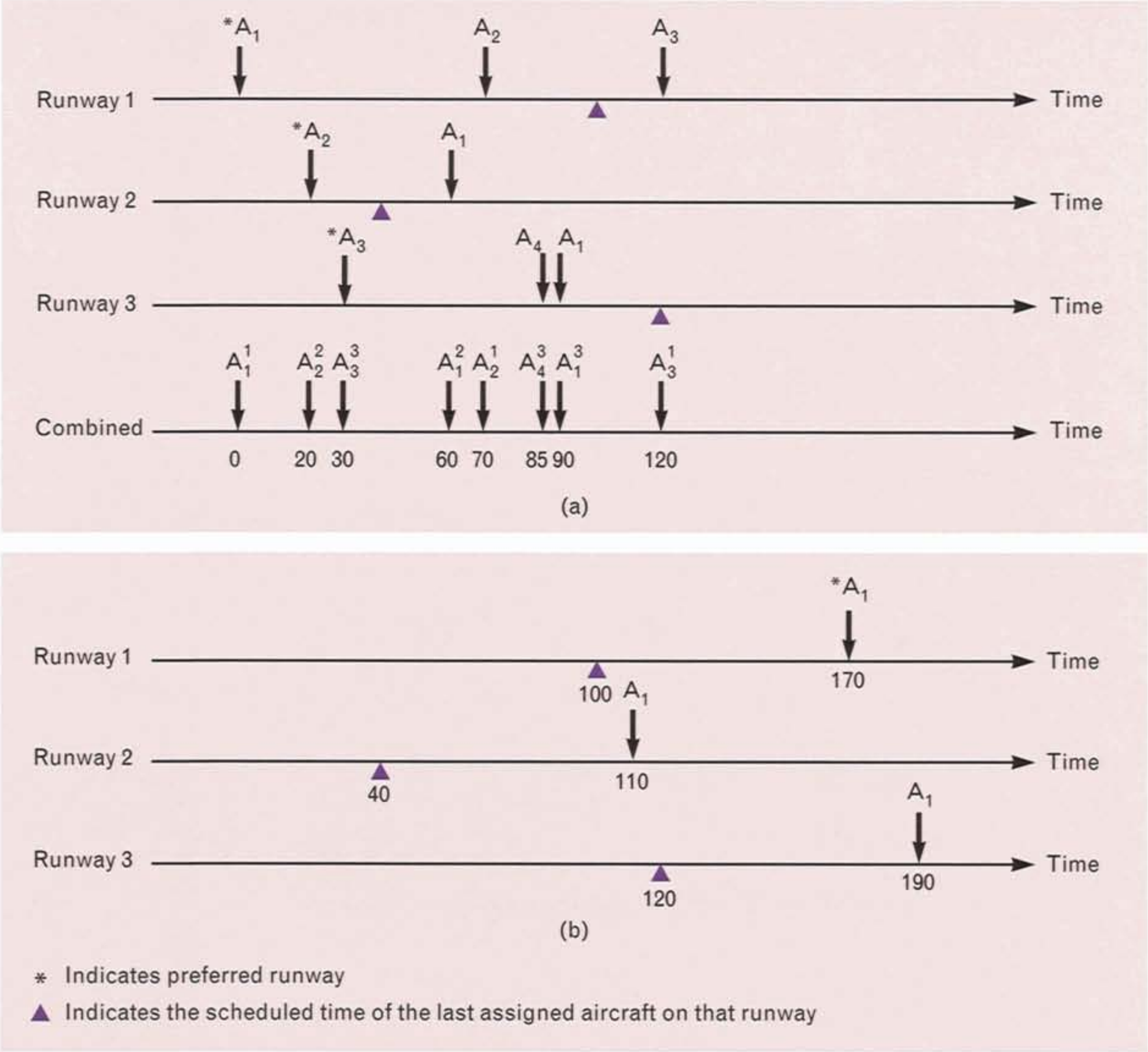


FIGURE 14. Demonstration of the runway-assignment algorithm with a three-runway example. (a) The bottom time axis shows the earliest landing times of a sequence of arriving aircraft. The top three time axes show these same inputs on the three separate runways. In this figure, aircraft A_1 is ready for runway assignment. (b) Aircraft A_1 and competing aircraft are preliminarily scheduled (by a first-come-first-served scheduler) on each of the three runways before CTAS makes the final assignment decision.

that clarifies the proposed runway-assignment algorithm. The bottom time axis of Figure 14(a) shows the inputs to the algorithm in the form of a time line of ETAs, where A_i^j in the figure refers to aircraft A_i on runway j. Time is being counted down continuously. When some arrow reaches time zero, as is the case for aircraft A_1, the runway-assignment decision is made by CTAS. The first three time axes show the same inputs but on the three individual runways, along with an indication of the time of the last scheduled aircraft. ETAs of aircraft on their preferred runway are marked with an asterisk. We assume here a time threshold T of 120 sec. Figure 14(b) shows how aircraft A_1 would tentatively be scheduled on all runways. For simplicity we assume a required minimum separation of 70 sec for all aircraft. The resulting scheduled times of A_1 are now compared to see if A_1 could be advantageously scheduled on a non-preferred runway (i.e., whether it could be scheduled to land T minutes earlier on the non-preferred runway).

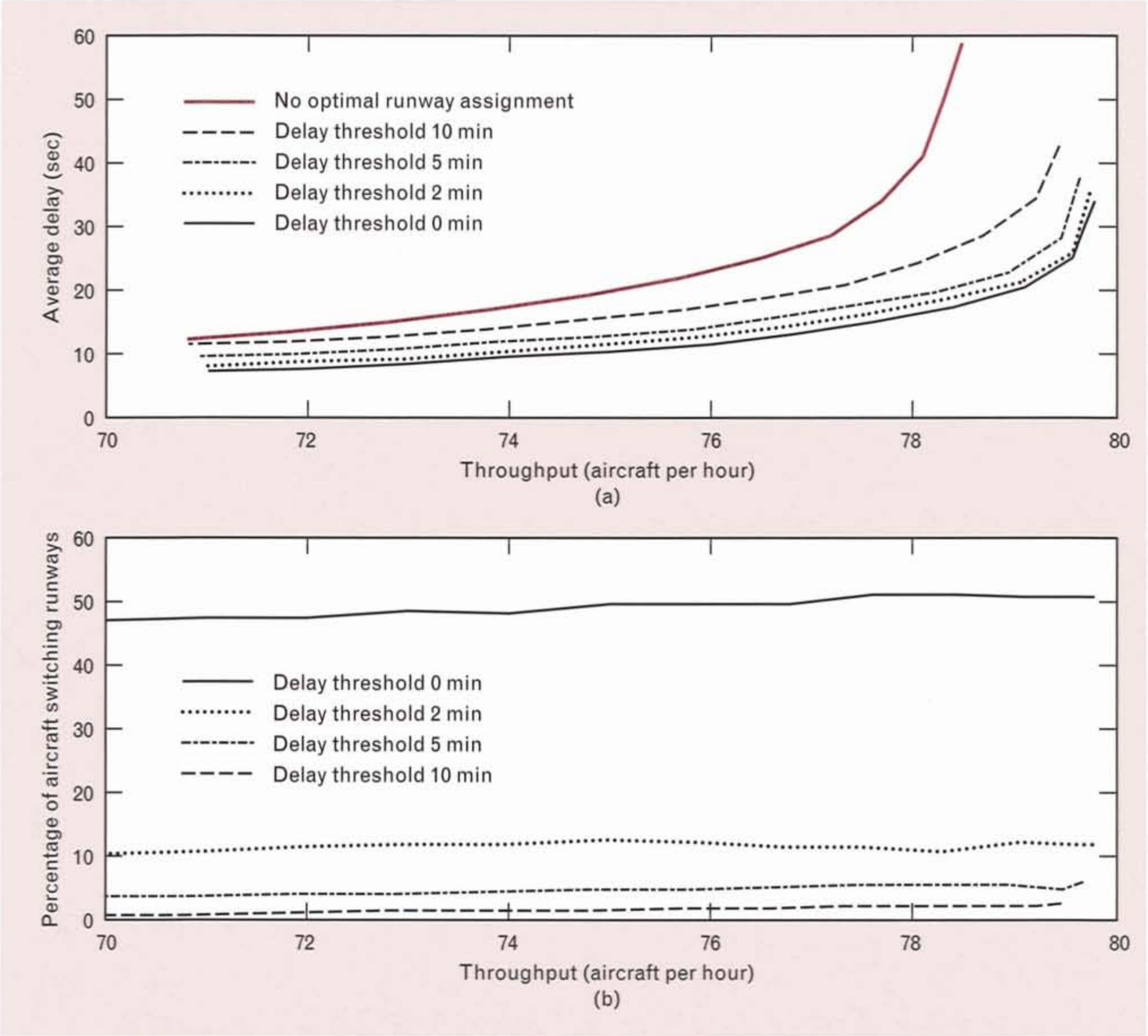


FIGURE 15. Operating curves for the two-runway balanced traffic case in a first-come-first-served scheduling system. Capacities are 40 aircraft per hour per runway. (a) These curves show the effect of delay threshold on performance when a policy of "reluctant" crossovers is used. (b) Crossover rates as a function of the crossover delay threshold.

In this example, that is not the case and the aircraft is assigned to its preferred runway.

Performance Evaluation

To evaluate the performance of these algorithms we consider two traffic cases, and we show results for a simple case of two runways. In the first case the average flow rate to both runways is the same. We label this case the *balanced traffic* case. In the second case we consider a 64/36 ratio in the average flow rate to the two runways. We label this case the *unbalanced* case. Obviously, we expect heavy crossover rates in the second case. We assume that the chosen conditions (balanced, unbalanced, arrival rates) do not change for the duration of the tests. The tests simulate real-life traffic duration exceeding eight hours, so that we have repeatable performance results.

First we discuss the balanced traffic case by examining the operating curves in Figure 15. The top curve in Figure 15(a) represents the operating curve

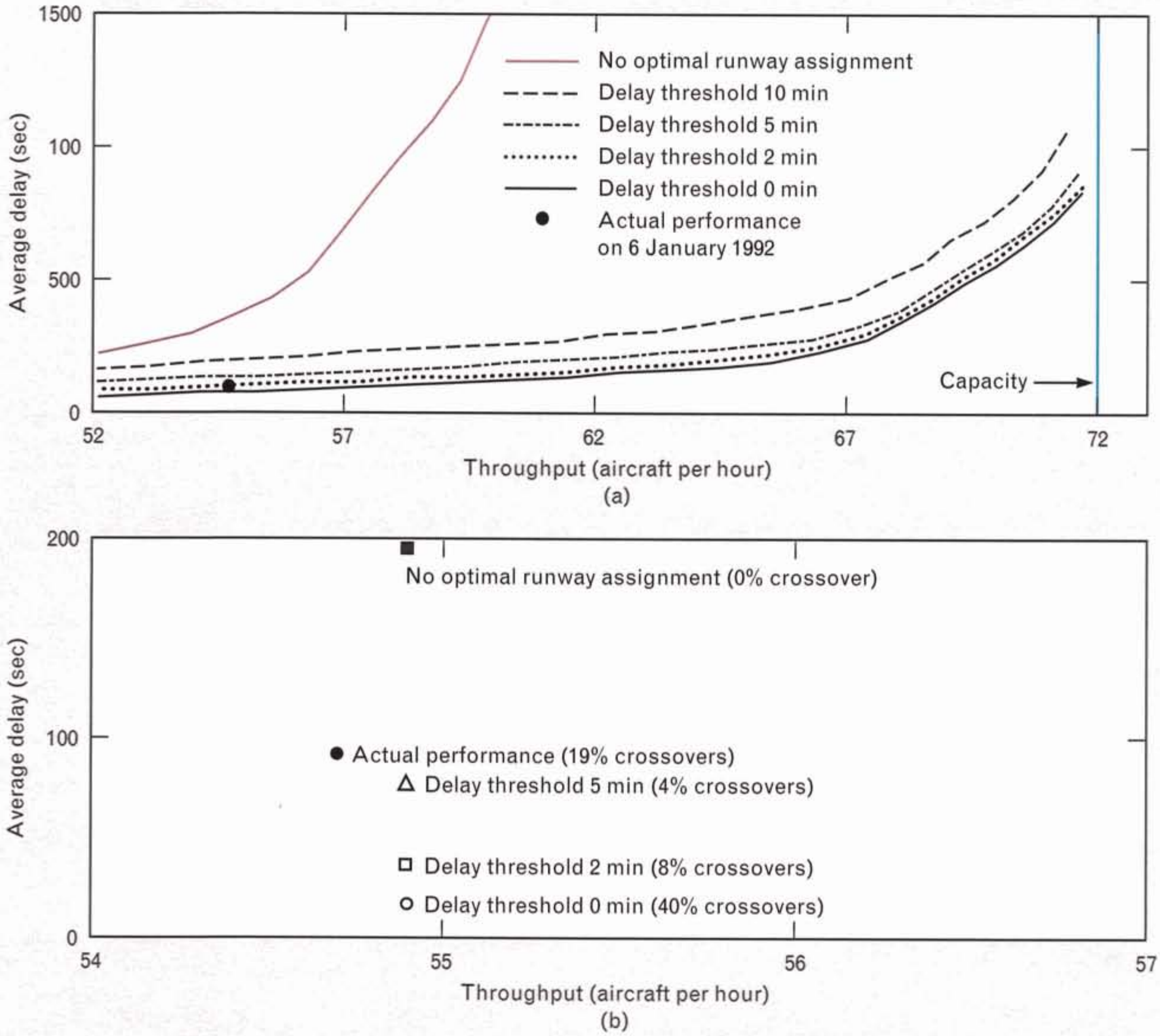


FIGURE 16. Operating curves for the two-runway unbalanced traffic case in a first-come-first-served scheduling system. Unbalanced traffic is in the ratio 64/36, and capacities are 36 aircraft per hour per runway. (a) Simulated and measured performance using optimal runway assignment for the same two runways. (b) Simulated performance using optimal runway assignment when input to the scheduling system is actual Dallas–Fort Worth Airport traffic.

for a first-come-first-served scheduling system in which the ETAs constitute a Poisson process and no crossovers are allowed. The bottom curve is the operating curve for a system in which crossovers are allowed. The expected factor of two for improvement in the average delay rate is clearly visible. The operating curves in between result when crossovers are restricted by the use of a threshold in the delay gain in a crossover. The greater the required delay gain (the threshold) the less the overall delay improvement, i.e., the greater the average delay as expressed by the operating curve, although all operating curves keep hugging the lowest curve. By looking at Figure 15(b), however, we observe that the crossover rate is dramatically reduced when even a small threshold value is chosen. The conclusion is that, with increasing thresholds, performance deteriorates slowly but the crossover rate reduces rapidly. Because crossovers represent increased controller work load, these should be kept low (i.e., below 10%), which implies an actual working threshold on the order of two minutes.

For the unbalanced case the improvements in average delay rate are in principle even greater. The runway-assignment algorithm goes beyond balancing traffic rates to runways; it also reduces individual delays (and therefore the overall average delay). For example, Figure 16(a) shows a series of operating curves similar to Figure 15(a), but for a 64/36 ratio of traffic to two runways. The capacity per runway was chosen to be 36 aircraft per hour. These choices match a traffic situation obtained from recordings made on 6 January 1992 at Dallas–Fort Worth Airport between noon and 5 p.m. The average throughput rate was 54.7 aircraft per hour and the average delays observed were 92 sec with a 19% crossover rate (i.e., aircraft over the easterly corner posts Blue Ridge and Scurry were landing on the more westerly runway 18R and aircraft over the westerly corner posts Bridgeport and Acton were landing on the more easterly runway 17L; the traffic was a south flow pattern). That measured operating point is shown on Figure 16(a) as a large black dot.

From a cursory examination of these data, compared to the operating curves obtained from simulated data, it would seem that manual operations resulted in excellent performance that would be hard to improve. One of the reasons is that the simulated traffic inputs used to obtain the operating curves were Poisson processes and more random than the actual traffic on 6 January 1992. On that day a metering program was in effect in the Dallas–Fort Worth center that helped smooth traffic going into the TRACON. Figure 16(b) shows the comparison when the recorded traffic was replayed through the algorithm. Clearly, a comparable delay performance could have been obtained with a five-minute threshold, but this threshold choice would have resulted in only 4% crossovers instead of the 19% crossover rate observed in actual operations. Or the two-minute threshold could have been used, which would have reduced both the delay (by a factor of four, to sixteen seconds) and the crossover rate (by a factor of better than two, to 8%). We could argue that in the regime of traffic where delays are already less than two minutes, a further delay reduction is of little value. The reason we had such small average delay, however, is that in the recorded traffic sample some delays were absorbed in the en route area (by metering) and the utilization rate of the airport was only 75% (i.e., throughput was 54 and capacity was 72). The improvement ratios from runway-allocation algorithms hold over the full range of utilization rates. The operating curves obtainable with a system with runway-allocation algorithms operating suggest that when the airport is busy (average arrival rates are at 90% capacity), the payoff in reduced crossovers and lower average delay will be considerable.

Synergism between Optimal Sequencing and Runway Assignment

Because the availability of several runways would allow controllers to group landings of aircraft by weight class, optimal runway assignments should reinforce the process of optimal sequencing. Simultaneously optimizing runway assignment and runway sequencing should lead to greater benefits than performing these functions separately and sequentially. Simulation results have indeed shown such a conjecture to be accurate. The incremental performance improvement is small, however, and the costs in terms of algorithmic complexity and computational overhead are too great to warrant this approach.

To illustrate this point, we present Figure 17 as an example of a balanced traffic situation. In this example we have two runways with a capacity of 42.5 aircraft per hour each, a traffic mix of .09/.87/.04, and a sliding scheduling window containing no more than eight aircraft at a time. Figure 17(a) shows four operating curves for this example: the top curve represents simple first-come-first-served sequencing and all aircraft go to their preferred runway (no crossovers). The next curve represents first-come-first-served scheduling with crossovers. The third curve shows optimal scheduling separately for each runway after runway assignment has been done, which is known as *sequential implementation.* The fourth curve employs an algorithm that considers all possible ways to divide the set of candidate aircraft among the runways and all possible ways to sequence these aircraft, and selects the division and sequences leading to the lowest cost.

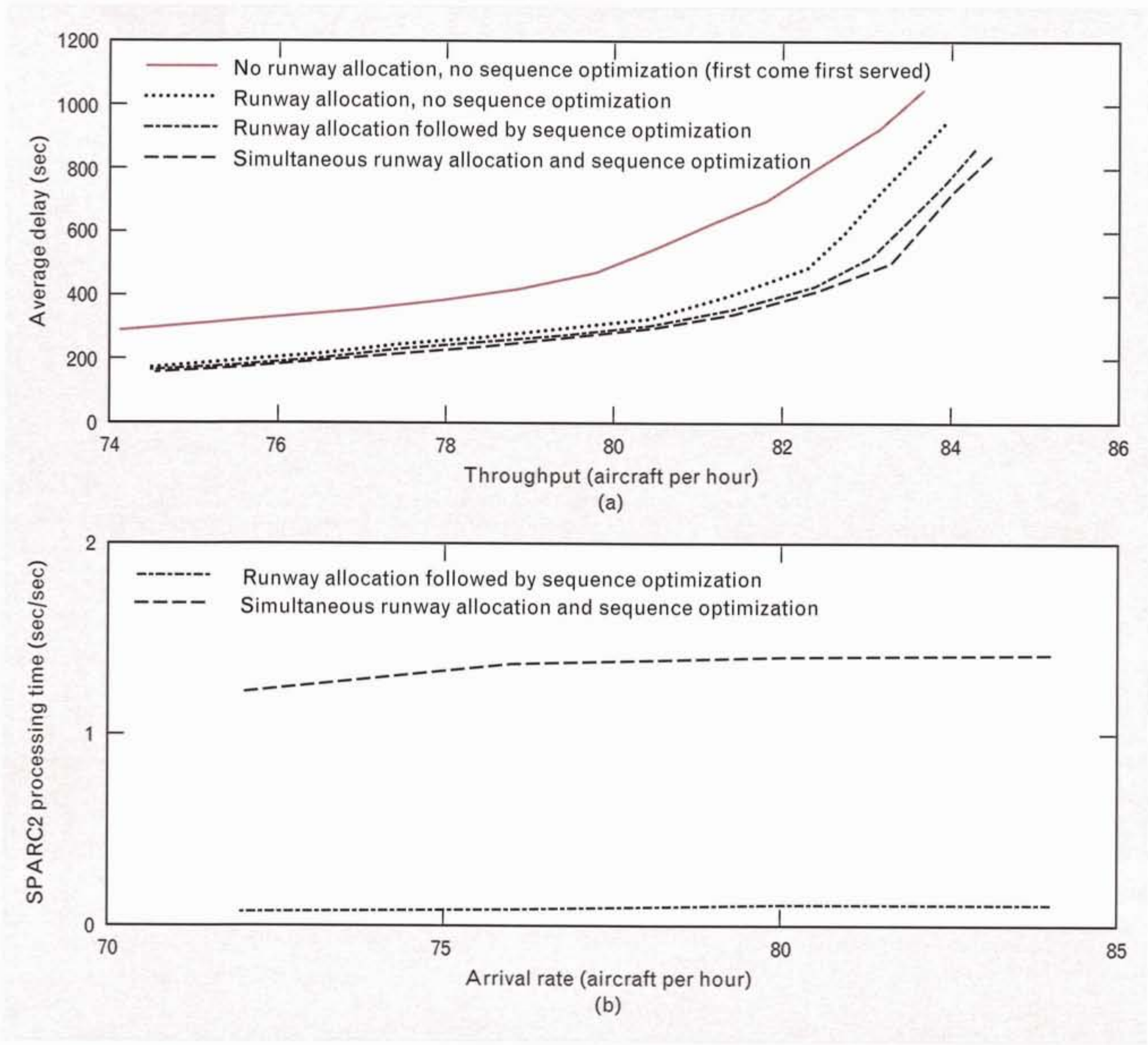


FIGURE 17. (a) Performance comparison of several implementations for the two functions of runway allocation and sequence optimization: (1) neither function, but first-come-first-served scheduling to the preferred runway only, (2) runway allocation only, (3) both functions but applied sequentially, and (4) both functions but applied simultaneously. Choice of parameters: two arrival Poisson streams of equal average rate; the mix of aircraft weight type is heavy/large/small =.09/.87/.04; optimization window is restricted to eight aircraft; capacity is 42.5 aircraft per hour per runway. (b) The processing time for simultaneous implementation of runway allocation and optimal sequencing for a scenario with balanced arrival traffic exceeds the processing time for sequential implementation by a factor greater than ten.

This fourth curve is indeed the lowest (i.e., the best) operating curve. Figure 17(b), however, shows the corresponding computational burden; now the lowest curve (which is significantly lower by a factor of more than 15) is the curve for sequential implementation. Synergism? Yes. Worth the price? No. Designing an algorithm that simultaneously performs runway allocation and sequence optimization is not only a complicated task but is very computationally intensive and the improvement over a design in which these tasks are performed sequentially is minimal. In fact, most of the improved performance derives from runway allocation (based on first-come-first-served scheduling on candidate runways), with only a minor improvement attributable to optimal sequencing.

Conclusion

During the design and early implementation phase of a large and complex system such as terminal automation we must be guided by a vision of where the performance payoffs are most likely to occur. Automation is capable of making decisions based on a more global awareness of traffic converging from all directions to a number of runways, and at the same time taking into account localized and time-changing weather conditions and individual aircraft characteristics. Runway-assignment algorithms not only fulfill a strategic role by equalizing traffic loads, but they also play a tactical role by exploiting opportunities to fill otherwise irretrievable gaps in landing sequences. This increases efficiency by increasing the throughput and reducing average delays.

Whereas optimal runway-assignment algorithms merely exploit existing runway capacity more efficiently, optimal sequencing algorithms reduce the average separation and therefore increase capacity. Several factors conspire to undermine the potential of these algorithms, however. These factors are the following: (1) the number of aircraft in the scheduling window whose landings can be rearranged is usually limited; (2) in most real traffic situations there is a preponderance of a single aircraft type, usually large aircraft (the statistical occurrence of potential heavy-small ordered pairs with which the greatest potential savings could occur is too small to make any great impact); and (3) the implementation costs of optimal sequencing algorithms are overwhelmingly high in terms of the complexity of the code and the computational resources needed.

REFERENCES

1. CTAS Operational Concept Document.
2. H.F. Vandevenne and M.A. Lippert, "Test Duration and Validity of Performance Claim for TATCA Designs," *Technical Report 41L-0396*, Lincoln Laboratory (3 Apr. 1992).
3. H.F. Vandevenne and M.A. Lippert, "The Best, the Worst and FCFS Sequences; Comparison of Capacity," *Technical Report 41L-0378*, Lincoln Laboratory (28 Jan. 1991).
4. H.F. Vandevenne and M.A. Lippert, "Benefits from an Algorithm for Better Multiple Runway Allocation," *Technical Report 41L-0416*, Lincoln Laboratory (15 June 1993).

APPENDIX: ILLUSTRATION OF ARRIVAL RATES, SCHEDULING DELAYS, AND THROUGHPUT

We present Figure A to illustrate the meaning of some important variables discussed in this article: arrival rate (short-term and long-term averages), scheduled delay (short-term and long-term averages), and throughput (and its relation to arrival rate and capacity). We present these variables for the final five hours out of a ten-hour simulation of traffic. Only the last half of the traffic is used in order to avoid initial transients that arise at startup.

The arrival process shown is Poisson with an average rate of 36 aircraft per hour. Each individual arrival event is represented by a vertical line in frame 1 of the figure. We can observe the occasional bunching and the occurrence of large separations (gaps) in the arrival stream. Frame 2 shows hourly averages varying between 24 and 43 arrivals per hour (sometimes exceeding the capacity of 40 aircraft per hour) and a longer term five-hour average arrival rate that settles down to approximately 36 aircraft per hour. Even though the long-term (five hour) average rate was close to 36 aircraft per hour, the hourly arrival rate fluctuates quite a bit. This assertion is often used to justify the use of Poisson processes to simulate the varying daily arrival rate observed at airports, even though the fluctuations in the arrival rate are predictable and the fluctuations occurring in the Poisson process are not.

Frame 3 and frame 4 show the results of a scheduler imposing minimum separations at landing (here simplified to ninety seconds between all aircraft). Individual scheduled delays shown in frame 3 vary between zero and ten minutes. The hourly average, as shown in frame 4, varies considerably less. Observe how the bulge in the hourly average delay occurs later than the bulge in the curve of the hourly average arrival rate. This observation occurs because delays continue to build up as long as high arrival rates persist, and the delays peak at the very end of the high-arrival-rate period and then, when arrival rates are lowered, dissipate only slowly. Such lags are typical in queuing systems.

Frame 5 shows so-called busy and idle periods of runway usage. During busy periods all aircraft are landed as closely spaced as legally allowed. Idle periods consist of the "excess" spacings between landings. If we are allowed to do violence to the definition of arrival rate by restricting the interval over which we average the landings to the busy and idle periods, we could say that the throughput rate has only two values: equal to capacity during the busy period and equal to zero during the idle period. The busy periods, when added together, form a fraction of the time line equal to a variable that is called the *utilization factor* of the runway.

Frame 6 shows the hourly average throughput rate that fluctuates much like the hourly arrival rate, except that the average throughput rate never exceeds capacity, while the average arrival rate can and occasionally does exceed capacity. Delays build up rapidly during periods when the hourly throughput rate equals or is close to capacity. The long-term throughput rate, shown as a dotted line in frame 6, equals approximately 36 aircraft per hour, which closely equals the long-term arrival rate (because all aircraft must ultimately land) and the ratio of long-term throughput rate to capacity equals the utilization factor (which here is 90%). The utilization factor, sometimes referred to as the degree of busyness of the runway, plays a crucial role in the analysis of performance and performance improvements in this article.

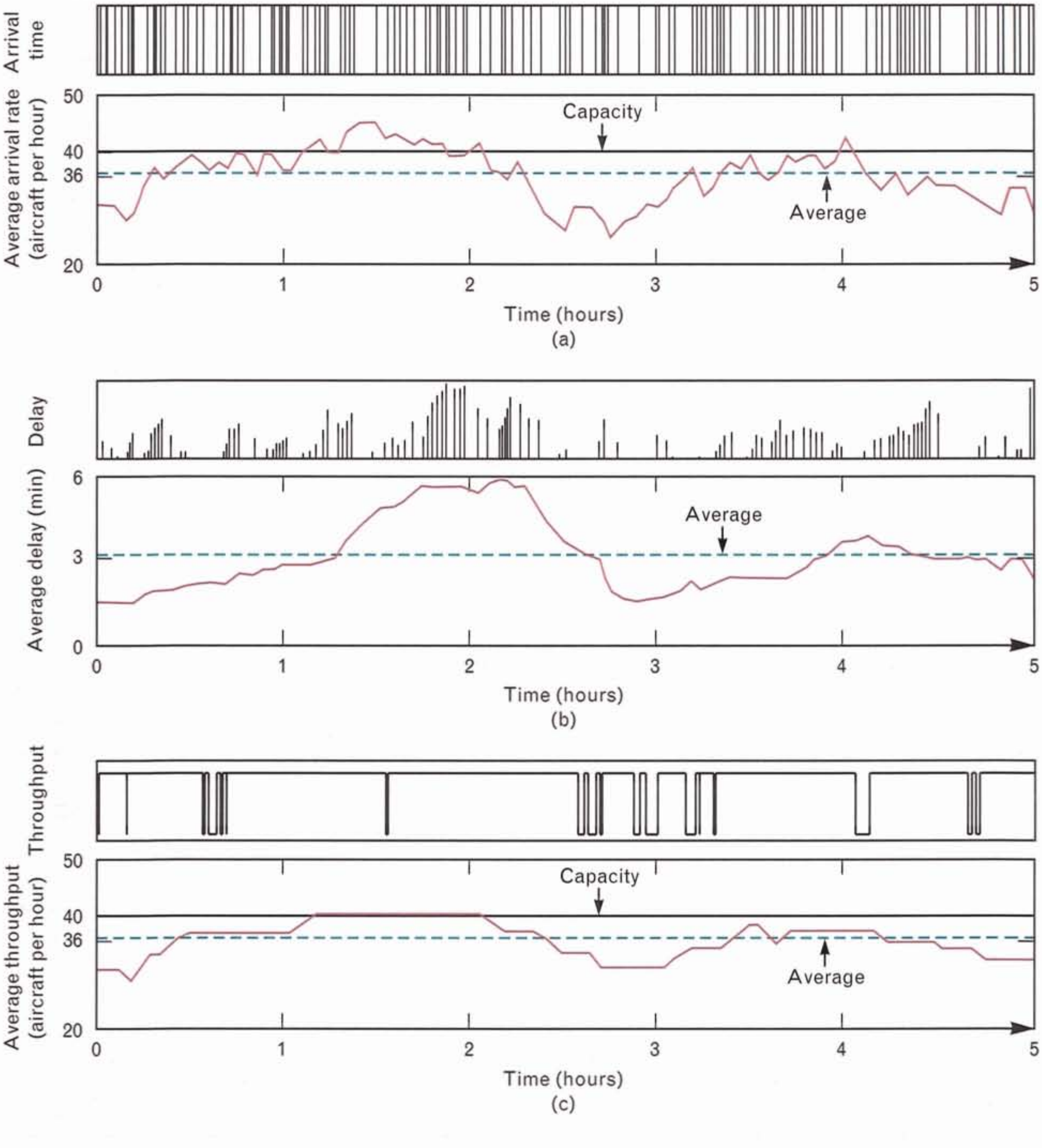


FIGURE A. Illustration of the relationship among salient variables in the scheduling problem, shown for an arrival process that is Poisson. (a) Frames 1 and 2: aircraft arrival times and hourly average (sliding window) and long-term average arrival rates. (b) Frames 3 and 4: scheduled delays for individual aircraft and hourly average delay. (c) Frames 5 and 6: throughput busy and idle periods and hourly and long-term average throughputs.

HERMAN VANDEVENNE
is a staff member in the Air Traffic Automation group. His area of research is in the development of runway-assignment and aircraft-sequencing algorithms for air traffic control. Before he joined the Air Traffic Automation group in 1989, his research at Lincoln Laboratory was in the analysis and design of advanced tracking systems, digital communications, and signal intercept systems. He was a member of the original Mode S design team. Dr. Vandevenne received a master's degree in electronics and a master's degree in computer science from the University of Louvain in Belgium. He holds S.M. and Ph.D. degrees in control and decision engineering from MIT. He was also a NATO Postdoctoral Fellow.

MARY ANN LIPPERT
is an assistant staff member in the Air Traffic Automation group. Her research focuses on writing simulations of aircraft landing sequences. She received a B.A. degree from the University of Detroit and an M.A. degree from the University of Massachusetts, both in mathematics.