ANALYZING THE RUNWAY CAPACITY OF COMPLEX AIRPORTS

Dr. John N. Barrer[*], Peter Kuzminski[†], and William J. Swedish[‡]
The MITRE Corporation, McLean, Virginia, 22102

This paper describes a simulation modeling system we have developed, called Airport Capacity Analysis Through Simulation (ACATS). Airport capacity, in the sense of the average throughput obtainable during periods of high demand, is determined directly by simulating a constant flow of arrivals and departures for hundreds of hours. The user interface for ACATS provides a fast way to set up the elements of the airport that are essential for calculating runway capacity. It also supports the use of Air Traffic Control (ATC) separation rules that may become feasible as technology improves. The software in the user interface automatically converts the data for any airport into a standardized set of files that are then processed by the ACATS simulation software. At the core of the ACATS software is a simulation engine that is common to all airport analyses. That means that the simulation is driven by data representing the ATC rules, runway layout, and demand characteristics. The output of ACATS includes an animation of the simulation, statistics about the observed throughput, and a set of graphical analysis charts. The animation and graphical results produced by ACATS are important tools in explaining the analysis to the end user and in validating the results of the simulation. This paper will describe 1) the ACATS user interface tool that permits the user to easily describe the problem, 2) the ACATS simulation module, and 3) the methodology that governs the ACATS algorithms.

I. Introduction

Many practical problems in the analysis of potential benefits of aviation systems hinge on estimating the effect of new infrastructure, technology, and procedures on the capacity of an airport's runway system. For many years the Federal Aviation Administration (FAA) and The MITRE Corporation's Center for Advanced Aviation System Development (MITRE/CAASD) have maintained and used the "Enhanced FAA Airfield Capacity Model" (EACM) [1] for assessing the capacity of an airport under differing assumptions about the separation rules, procedures, and the number of runways. This has allowed us to provide answers to questions such as: "Will the new runway be beneficial given the restrictions on its use? Will the airport be able to satisfy the projected demand ten years from now? If the runways are equipped with a new guidance system resulting in improved separation requirements, by how much will the capacity increase?" In addition, runway capacity serves as an input to many other models of the national airspace system.

This paper describes an analysis tool we have developed, called Airport Capacity Analysis Through Simulation (ACATS), for estimating the runway capacity of any airport. The ACATS system combines the desired features of simulation and the EACM. Our motivation for this work was to overcome the analytical limitations of the EACM by developing a simulation-based modeling methodology and to overcome the drawbacks of a traditional simulation by avoiding its costly set-up process. That this was even possible was uncertain at the start of this project, so it was

[*] Lead Systems Analyst, Airport and Airspace Analysis, N370, 7515 Colshire Drive, McLean, VA 22102, Corporate member.
[†] Lead Technical Staff, Modeling and simulation, N590, 7515 Colshire Drive, McLean, VA 22102, Corporate member.
[‡] Principal Staff, Airport and Airspace Analysis, N370, 7515 Colshire Drive, McLean, VA 22102, Corporate member.

funded as a MITRE internal research and development project. MITRE/CAASD is a not-for-profit Federally Funded Research and Development Center sponsored by the FAA.

The definition of runway capacity in the EACM is *average maximum sustainable throughput* for a given fleet mix and arrival-departure ratio. ACATS also estimates capacity using this definition. *Throughput* is defined as the number of aircraft that use the runway system per unit time, in a use pattern obeying the arrival-departure ratio and aircraft fleet mix. The fleet mix is the percentage of each aircraft type that uses the airport. The concept of *sustainability* is that if sufficient demand is present, the airport can maintain this average throughput indefinitely. The average is over time. Although the fleet mix remains constant, the sequence and performance characteristics of the aircraft vary, so the throughput per unit time varies with these factors.

The EACM, while easy to use, begins to fail when the complexity of the runway system increases, requiring significant post processing by an experienced analyst. Consequently, MITRE/CAASD, as well as many Civil Aviation Authorities and consultants, have relied on simulation models to study the effect of new systems on runway throughput at these more complex airports. A well-constructed simulation of a particular airport can model much more subtle effects than can be measured by the EACM, and is particularly useful in studying the effects of different traffic management strategies. The chief drawback of simulation models is that they have to be custom built for the particular airport, a task which often requires months of work to collect the supporting data, validate the simulation, and perform the analysis.

The ACATS user interface provides a very fast way (requiring minutes or hours, not days) to set up the elements of the airport that are essential for calculating runway capacity and permits the testing of existing Air Traffic Control (ATC) separation rules as well as those that may become feasible as technology improves. The software in the user interface converts the data for any airport into a standardized set of files that are then processed by the simulation software. At the core of the ACATS software is a simulation engine that is common to all airport analyses.

With this model we can start from a blank page for any airport in the world and be running case studies within minutes. The animation and graphical results produced by ACATS are valuable tools for explaining the analysis to the end user.

This paper will describe 1) the ACATS user interface tool that permits the user to easily describe the problem, 2) the ACATS simulation module, and 3) the methodology that governs the ACATS algorithms.

II. Background

The estimation of airport runway capacity is fundamental to all research and planning for airport planning and improvement. All civil aviation authorities either have their own models for estimating capacity, or use consultants to perform the analysis.

For more detailed analyses of complex interactions, analysts use simulation models and there are numerous such models. An internet search will reveal many simulation studies that have been performed on airports around the world. The disadvantage of simulations is that they are costly to set up and often take months to complete. One of the best known models is the Total Airspace and Airport Modeler (TAAM) model [2] and we at MITRE/CAASD are one of the largest TAAM users in the world. We have also used several other well-known models over the years, such as SIMMOD. Typically, these models require a detailed schedule of arrivals and departures to be input for a specific time period, such as 16 or 24 hours.

Other models attempt to estimate the average flow of traffic using closed-form equations to calculate the expected flow rates. The EACM [1] is one of them. They have the advantage of being relatively fast. No traffic schedule is required, just a description of traffic characteristics such as fleet mix. However, this speed comes at the cost of reduced flexibility. The EACM models complex airport configurations by decomposing them into a number of sub-configurations consisting of individual runways or pairs of runways, then adding the capacities of each sub-configuration. For example, a pair of independent runways is modeled as two separate single runways. There is a separate equation, and a separate module in the ACM program, for each sub-configuration. If airport operations are not adequately represented by the standard configurations modeled by the EACM (which happens frequently), the analyst needs to perform the decomposition manually and then use a spreadsheet to calculate the overall airport capacity.

With the ACATS model we are trying to overcome the analytical limitations of the EACM by developing a simulation-based modeling methodology, and to overcome the drawbacks of a traditional simulation by avoiding its costly set-up process.

III. ACATS Results

The simulation engine processes the standardized data and records the results of the simulation in both graphical and numeric representations. This takes several minutes for a large airport.

There are two primary results of an ACATS analysis. The first is a set of capacities. Each capacity is an estimate of the average maximum sustainable rate at which the airport can service demand of a particular fleet mix and operations (arrival and departure) mix, while respecting separation requirements. For a given fleet mix, the average capacities for different operations mixes may be plotted on an arrival vs. departure graph and connected by lines to form what is commonly referred to as the *capacity curve*, for use as input to other models and in broader discussions. An example is shown in Figure 1.

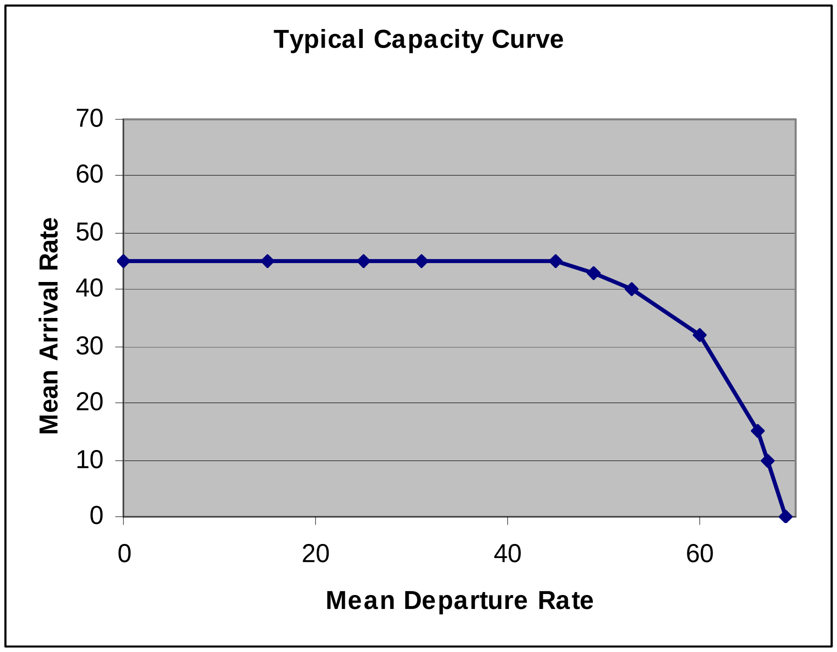


Figure 1: Example of a Capacity Curve

The second primary result of an ACATS analysis is a log of its simulated operations. This record may be used for analysis of particular operational phenomena, including bottlenecks and tactical patterns. In addition, this record

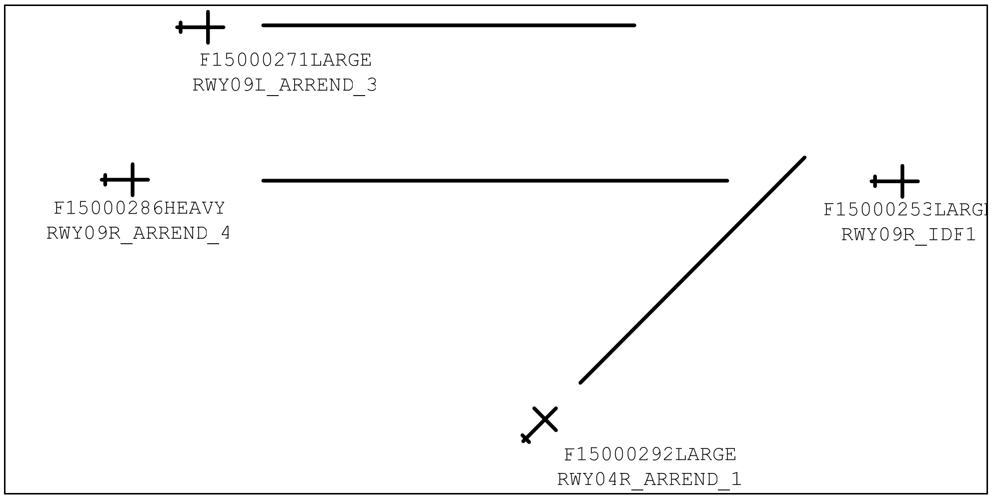


Figure 2. Animation of Simulation Results

can be animated to provide an easily understood visualization (see Figure 2) of how the airport has been modeled and how traffic behaves.

Because the ACATS engine simulates distinct flights, it may be used for other purposes besides estimating capacity. If presented with flights either from a schedule or an external simulation, the ACATS engine can model the experience of each flight in the vicinity of the runway system and thereby estimate induced delay due to separation requirements, tactical sequencing, and runway assignment decisions. In particular, it is possible to estimate the delay induced by the runway layout and ATC rules used for separation, a broadly used measure of system performance.

IV. The User Interface

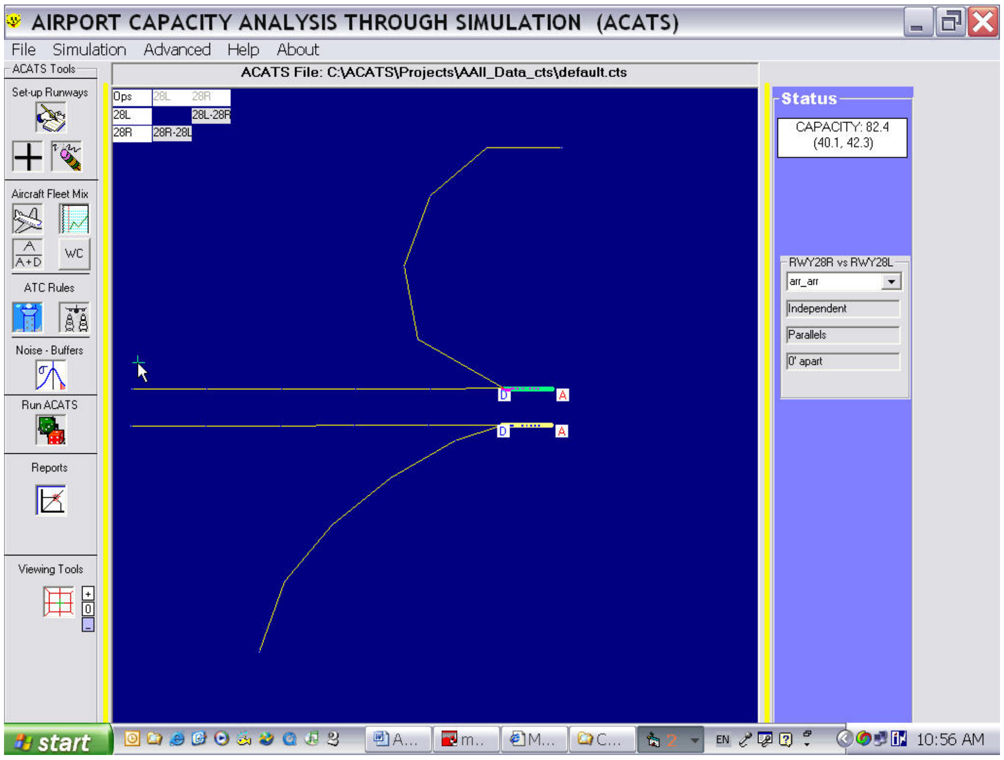
The ACATS software runs on a PC running Microsoft Windows® and requires a license for the ACATS program, for the SLX® simulation software used by the simulation engine, and for Microsoft Office® to generate reports from the simulation data.

The ACATS graphical user interface (GUI) (see Figure 3) provides tools to help the user create the data for a case study. A case may include many different scenarios of airport operations and configurations.

The main categories of data required during the case creation phase are:

1. Runway layout
2. Wake vortex separation requirements
3. Fleet mix
4. Aircraft performance parameters
5. Arrival-departure ratio
6. ATC rules used for separation
7. Simulation control attributes

The GUI allows the user to specify the ATC rules and runway geometry of the airport. The main screen of the GUI is shown in Figure 3. From this screen the user can transfer to other screens in order to enter the parameters



ACATS File: C:\ACATS\Projects\AAll_Data_cts\default.cts

Ops	28L	28R
28L		28L-28R
28R	28R-28L	

CAPACITY: 82.4
(40.1, 42.3)

RWY28R vs RWY28L
arr_arr
Independent
Parallels
0' apart

Figure 3. Main Screen From ACATS User Interface

needed to set up the airport.

To develop a detailed diagram of a new airport, the user can import a previous case and modify it, create a new diagram from a template, or create the modified airport starting from a single runway. For example, the user can start with a three-runway airport having one runway oriented 090 degrees, and simply state that there is to be an additional runway, parallel to Runway 9, 3000 feet to the right with length 10,000 feet. The user does not have to enter any latitude or longitude coordinates unless desired. (It is sometimes useful to model the runways by latitude and longitude when the approach and departure fixes are only given by their coordinates.)

The user must specify the ATC rules that are used to maintain separation between all pairs of aircraft on all pairs of runways. This is facilitated by allowing the user to select from previously stored settings that correspond to known sets of rules (e.g., rules for operating dependent parallel runways). In addition, the user can create a custom pairwise separation rule that is based on either time or distance. For example, the user could specify that an aircraft of type A at the threshold of Runway 3R will be separated from a leading aircraft of type B at the Runway 3L threshold by a minimum of 37 seconds. This is a rule that does not exist today because there is no technology to support it, but one could envision a highly automated environment of the future in which such a time based separation would be feasible. The GUI is very flexible and allows the user to select where the separation is to be applied from a list of common points (e.g., the threshold, final approach fix, or departure fix), whether it is a time-based or distance-based rule, and whether the rule is applied equally for all aircraft or whether it is based on the wake vortex classes. The ACATS model can use this rule-creation capability to model variations in any ATC rule, and thereby easily adapt the model to all airports.

The user may also select a very precise fleet mix of aircraft that can use the airport and further partition that set among the individual runways. For example, the user can specify that small jets cannot use a particular runway but small props can. This can be decided based on any of the 87 individual aircraft types in the accompanying database.

The demand pattern is represented by the aircraft *fleet mix* file and the *arrival-departure ratio* file. The user can select a default setting for both of these or customize the settings. In the arrival-departure ratio file, the user can vary the ratio that is to be used as the simulation progresses in time.

The wake vortex separation rules require time or distance parameters for different weight classes. The user has the option of importing known sets of such data, modifying them, or creating a completely new set.

While the simulation engine works with latitude and longitude points on and around the airport, the GUI software performs all of the mathematical calculations (in spherical coordinates) to convert ATC rules into the input required for the simulation engine.

The user can also set parameters such as the run time for the simulation, or whether the data should be reported hourly, in 15-minute increments, or at intervals specified by the user. There are other options available for advanced users that control how the simulation processes flights.

V. Creating a Standardized Representation of an Airport

The ACATS software automatically converts any airport's description into a standardized format that serves as input to the simulation engine. The conversion requires no intervention by the user and is accomplished in a fraction of a second.

Any airport's system of runways and ATC rules can be abstracted to a standard representation involving the concepts of *runways, traffic streams, points, segments,* and *blocking rules*. The demand pattern is represented by the aircraft *fleet mix* file and the *arrival-departure ratio* file. Each of these files is automatically created by the software based on the inputs from the user.

Separation requirements are enforced by establishing segment-to-segment rules that specify which segments are blocked to other aircraft when a leading aircraft is in a particular segment. These rules are all incorporated into the *point, segment, traffic stream, and blocking* files. The key concept underlying the ACATS algorithm is that the ATC separation rules can be expressed by referring to segments that are blocked for use by one type of aircraft when another type of aircraft is using the same or a different segment. This is the concept we have called *blocking*. For example, with two converging arrival streams to intersecting runways, if one arrival is crossing its runway threshold, an arrival on the other runway must not be within 2 miles of its threshold. Time separation is enforced by having a condition that says the blocked aircraft cannot begin using its blocked segment until a certain amount of time has passed after the blocking aircraft has exited the blocking segment. In some cases an additional amount of time is added to the blocked segment to account for uncertainty. This amount of time is called a *buffer*.

The simulation engine then operates on these sets of data to produce the output. As the airport gets more complex the amount of time required to run the simulation will increase, but the structure of these sets and the simulation engine both remain constant. Consequently, the simulation engine requires no modification by the user and is driven solely by the data files that are automatically generated by the user interface.

A *point set* is a list of latitude/longitude coordinates with an associated point label. All the calculations of latitudes and longitudes are done within the GUI software in spherical coordinates but this is hidden from the user. The GUI also automatically generates the labels for each point.

A *segment* is an unordered set of point labels. It can have a single point as its only element. A *segment set* is a list of all the segments used to enforce ATC rules. All the segments are created by the software without user specification.

A *runway set* consists of the points that are located at each end of the runway and the exits. These are used by the simulation engine when calculating the runway occupancy times and exits used. The user must specify the location of the exits, measured in feet from the threshold; tools are provided to make this task easier.

A *traffic stream set* specifies the path that a type of operation (arrival or departure) will use as well as the classes of aircraft that can use that path. A path is an ordered set of points that is used by each aircraft in that traffic stream. The user must specify whether arrivals, departures, or both use a particular runway, and the types of aircraft that can use the runway. The software automatically creates the description of the stream of traffic and associated points.

A *blocking rule* is a statement reflecting the effect of an ATC separation rule and the runway geometry on the flow of traffic. It is expressed in the format:

Segment-blocking ACtype-X ACtype-Y
Segment-blocked time Buffer

For example, suppose that an ATC rule requires that when an arriving Heavy-class aircraft is crossing the runway threshold of Runway 19 that a following Large-class must be at least 5 miles out. The software would create a segment S1, containing the points *threshold* and *rwy19_5* (five miles before the threshold) and a segment S2 consisting of just the point *rwy19_5*. A blocking rule would be created as:

S1 Heavy Large
S2 0 0

The simulation engine enforces the restriction that when any Heavy aircraft is anywhere within the segment S1, no Large aircraft can *begin* using the segment S2. This prohibits a Large aircraft from following too close behind a Heavy aircraft when near the threshold, but does not restrict the separations when a Large leads a Heavy. There would be another blocking rule for Large leading Heavy.

For a complex multi-runway airport like Chicago O'Hare there could be almost 1,000 blocking rules. These are all created automatically by the software and the user need only have knowledge of the desired ATC rules. The GUI software creates all of these *runways, traffic streams, points, segments, and blocking rules* in response to user inputs expressed as statements about the ATC rules desired. This makes it easier for new users to learn the model.

VI. Methodology

From the beginning we wanted to create a simulation model for estimating capacity that would be easy to use and not require that the user know anything about the simulation language in order to model different airports. Our goal was to avoid requiring that the simulation code be modified or that extensive amounts of data be assembled for each new airport. We managed to achieve this objective and attribute our success to several factors.

First, our problem has a very definite focus. We want to estimate the capacity (the average maximum sustainable throughput) of the airport runway system for a constant demand. Knowing the desired outcome allowed us to limit the scope of the simulation.

Second, ATC rules are *pairwise* statements about the minimum separation requirements between two aircraft. A typical rule says "if an aircraft of type Y is in this area, then an aircraft of type X cannot also be in that area until x seconds has elapsed." This fact limits the number of constraints that must be considered and helps prevent an exponential increase in the problem size as the complexity of the airport increases.

Third, the constraints implied by ATC rules are additive. If, for example, a new runway is added to an existing configuration, all of the existing blocking rules still exist and are still pairwise (often with the same distance or time values) and only new pairwise rules related to the additional runway must be added to the set of rules. One consequence of this is that if we have already created a simulation for an airport and a new runway is added, the user does not have to start from a blank sheet, but rather can just add the runway and its rules to the existing simulation.

The simulation algorithm is conceptually very simple, but involves several interwoven heuristic processes to implement the proper selection of aircraft from the fleet, efficiently assign them to the runways available, and to incorporate the buffered blocking rules.

In simple terms, the ACATS algorithm repeatedly performs the following functions. The algorithm generates a set of aircraft waiting for selection. Next it selects the next aircraft to arrive or depart based primarily on how long it has been waiting (first come, first served), how much disruption it could cause to other traffic, and how soon it could go if selected. Once an aircraft is selected, its trajectory is calculated and the segments that it blocks are prevented from being used by other aircraft according to the blocking rules table. A new aircraft is then added to the list of waiting aircraft, based on the desired fleet mix and arrival-departure ratio, and its earliest release time is calculated based on the segments in its path that are free and blocked.

VII. On-Going Activities

We have performed many comparisons between ACATS and EACM showing that for simple cases the results are equivalent. Because the simulation engine in ACATS does not change from airport to airport, we know that once basic calculations are correct, they will remain that way. That, coupled with the animation, helps insure that all the ATC rules are being applied correctly. The combination also gives confidence that the observed throughput reflects the desired measure of capacity.

To achieve our measure of capacity as maximum average throughput, the simulation engine generates a continuous stream of flights that keep the airport saturated. By replacing our internal flight generator with an actual schedule of operations, we can also measure the delay that would be induced on that set of aircraft. Thus ACATS can serve as both a capacity model and a delay model.

Additional tools in the GUI assist the user in developing demand scenarios for testing purposes. However, we have adopted a flexible input format so that, with some pre-processing, many different types of real-world demand data can be transformed into sets useable by ACATS.

VIII. Conclusions

We have developed a simulation-based runway capacity model that enables the user to estimate the runway capacity of any airport using any set of ATC separation rules. When applied to complex runway layouts, it is more accurate than a steady-state analytical model and can produce results much faster than other simulation models. We expect to use this model in the future to quickly and accurately estimate runway capacity.

IX. Acknowledgments

The authors would like to thank The MITRE Corporation for sponsoring this work under its Sponsored Research Program.

X. Disclaimer and Copyright

XI. References

[1] *Swedish, W. J., 1981, Upgraded FAA Airfield Capacity Model, Volumes I and II, MTR-81W16, McLean, Virginia, The MITRE Corporation , 1981*

[2] *The Preston Group, 2004, Total Airspace and Airport Modeller (TAAM) Users Guide, Richmond 3121 VIC Australia, The Preston Group, 2004*