Modeling and Simulation Tools for Analysis of Terminal Airspace Operations

Monish D. Tandale*, Jason Kwan†, Sydney Lin‡, and V. V. S. Sai Vaddi§

Optimal Synthesis Inc., Los Altos, CA, 94022-2777

Terminal airspaces for the major airports in the National Airspace System (NAS) have been identified as areas of high traffic density and congestion. Consequently researchers in academia, industry and government have focused on analysis of terminal airspace operations to relieve system-wide congestions in the NAS. This paper presents two software tools developed for analysis of terminal airspace operations. Queue Generator (Q-Gen) is a modeling tool that develops a high-fidelity route model for a terminal airspace and segments the routes into servers based on the inter-aircraft separation criteria. The Discrete Event Queuing Simulator (DEQS) propagates the flights through the various servers of the Queuing Network Model (QNM), while implementing queuing behavior to enforce the capacity limits of each server. Rapid Monte Carlo simulation with arrival time uncertainty and spatio-temporally correlated wind uncertainty can be performed using DEQS. DEQS can perform 10,000 Monte Carlo iterations of a 6-hour terminal area simulation in less than 90 seconds. Stochastic analysis results for the LAX terminal airspace are presented to demonstrate the utility of the developed tools.

Nomenclature

ASDO Airspace Super Density Operations
ATC Air Traffic Control
BADA BAse of Aircraft Data
CIFP Coded Instrument Flight Procedures
DAFIF Digital Aeronautical Flight Information Files
DEQS Discrete Event Queuing Simulator
FCFS First Come First Serve
GUI Graphical User Interface
IAP Instrument Approach Procedure
LAX Los Angeles International Airport
NAS National Airspace System
NFD National Flight Database
Q-Gen Queue-Generator
QNM Queuing Network Model
RAP RAPid refresh
RUC Rapid Update Cycle
SID Standard Instrument Departure
SFO San Francisco International Airport
STAR Standard Terminal Arrival Route

* Senior Research Scientist, 95 First Street, monish@optisyn.com, Senior Member AIAA
† Research Engineer, 95 First Street, jason@optisyn.com
‡ Research Engineer, 95 First Street, slin@optisyn.com
§ Senior Research Scientist, 95 First Street, vaddi@optisyn.com, Senior Member AIAA

American Institute of Aeronautics and Astronautics
I. Introduction

Terminal airspaces for the major airports in the National Airspace System (NAS) have been identified as areas of high traffic density and congestion. Consequently researchers in academia, industry and government have focused on analysis of terminal airspace operations to relieve system-wide congestions in the NAS. NASA’s research in terminal airspace operations has been classified under the research focus area of Airspace Super Density Operations (ASDO). This paper describes modeling and simulation tools for the analysis of terminal airspace operations. The tools were developed with the primary objective of performing terminal airspace analysis using a queuing simulation framework. Details on the queuing framework are available in Reference 1. The modeling tool is called Queue-Generator (Q-Gen) and the simulation tool is called Discrete Event Queuing Simulator (DEQS). Note that the terminal airspace modeling tool can be used to perform modeling in a non-queuing setting as well.

Figure 1 illustrates an overview of the software architecture. The Q-Gen software tool has modules that perform the functions of (i) generating the terminal area route network from terminal airspace adaptation data, (ii) segmenting the route network into queuing servers based on the separation criteria and (iii) extracting historic demand on the various routes from recorded radar track data. The demand data can be provided to any external sequencing and scheduling algorithm to obtain a flight schedule. Note that a sequencing and scheduling algorithm is currently not a part of the tool set. The DEQS module performs a discrete event queuing simulation that propagates the flights through the various servers of the Queuing Network Model (QNM), while implementing queuing behavior to enforce the capacity limits of each server. The transit time for a flight in any server can be obtained from the Base of Aircraft Data (BADA)\(^3\) aircraft performance data coupled with the ambient wind field obtained from Rapid Update Cycle (RUC)\(^4\) or Rapid Refresh (RAP)\(^5\). Monte Carlo iterations can be performed using the DEQS simulations, with uncertainties such as (i) schedule time adherence uncertainty at the input and (ii) wind prediction uncertainty. A spatio-temporally correlated wind uncertainty model described in Reference 6 can be used to perform the Monte Carlo simulation. Each iteration of the Monte Carlo simulation provides the entry/exit times and delays for every flight at every sever. By combining data from all iterations, the statistics of various metrics of interest such as delays, throughput and traffic flow efficiency can be calculated.

Figure 1. Overview of the Software Architecture

The remainder of the paper describes each of the modules in detail. Section II describes the Q-Gen software used for generating a queuing network modeling of the terminal airspace. The DEQS tool used for performing queuing simulations using the developed queuing network models is described in Section III. Section IV describes
the various queuing metrics that can be calculated using the outputs of a DEQS simulation. Section V presents the results of stochastic analysis of LAX operations performed using Q-Gen and DEQS. Finally, summary and concluding remarks are presented in Section VI.

II. Queue Generator (Q-Gen)

This section describes terminal airspace modeling using the Q-Gen software. Q-Gen performs the following functions:

1. Generation of the route network model for a terminal airspace from the NAS adaptation data
2. Spatial discretization of the terminal area routes into servers based on the separation criteria (known as serverization)
3. Generation of the demand data by associating historic radar track data with the terminal area queuing network model

A. Creation of a Terminal Airspace Route Network

i. Display of Recorded Radar Tracks in the Background

The Q-Gen GUI displays a map in the background generated by tiling map data that can be obtained from Google Maps®, MapQuest® or Bing Maps®. The GUI provides basic functionality like zooming in or out, translation and return to a default screen center and zoom level. To assist the user in creation of the terminal area route network, Q-Gen provides a display of the recorded radar tracks on a satellite map background. Figure 2 displays the recorded radar tracks for arrivals (cyan) and departures (magenta) at SFO on a satellite map background obtained from Google maps.
The display of recorded radar tracks enables even a novice end user to create operationally accurate routes for a terminal airspace, even when the end user is not familiar with the operating procedures in the modeled terminal airspace.

**ii. Creation of the Route Network Using the NAS Data and Manual Editing**

The NAS data included in Q-Gen is obtained from the Coded Instrument Flight Procedures (CIFP) dataset, formerly known as the National Flight Database (NFD). The CIFP data can be used to support both enroute and terminal GPS navigation. The CIFP is updated every 28 days and is distributed by the Federal Aviation Administration (FAA).

The NAS data is displayed in 4 sub-panels:
1. **SIDs**: Standard Instrument Departure Routes
2. **STARs**: Standard Terminal Arrival Routes
3. **IAPs**: Instrument Approach Procedures
4. **Waypoints**: Named airspace fixes and navaids

![Figure 3. Using the NAS Adaptation Data and Manual Editing to Create the Route Network](image)

The SIDs, STARs and IAPs (routes) are displayed in the left panel. Checking the boxes for any route adds the route to the model and displays it on the map as a sequence of waypoints connected by directional arrows. The Waypoints panel lists all the waypoints that are in the current scope of the map window. Checking on the waypoint check box adds the waypoint into the model and displays it on the map display. The waypoints are arranged in alphabetical order and can be searched by clicking on the panel and typing the name of the waypoint. With radar tracks displayed in the background as a reference, the user can add the appropriate SIDs, STARs, IAPs to create the terminal area route network.

Some of the SIDs, STARs and IAPs do not contain complete route information leading to the runway. Frequently, STARs published in the CIFP end at a waypoint close to the runway with an instruction ‘Expect vectors
to runway’. These incomplete routes can be manually completed by adding named or user-defined arbitrary waypoints and links connecting the various waypoints. The GUI allows for adding, deleting and moving waypoints and links. Figure 3 shows a link that was manually inserted between the waypoints PIRAT and OSI. The radar track display in the background is especially useful in manual completion of incomplete routes.

Figure 4 illustrates a route network model for SFO arrivals in the West-plan configuration that was constructed using Q-Gen. The model is based on the radar track data for October 1, 2010 between 5:00 and 10:00 pm. Q-Gen offers the functionality for saving the developed models which can be loaded for later use.

<table>
<thead>
<tr>
<th>Airport</th>
<th>SFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival Runways</td>
<td>RW 28L, RW 28R</td>
</tr>
<tr>
<td>Start Time</td>
<td>October 1, 2010 5:00:00 PM</td>
</tr>
<tr>
<td>End Time</td>
<td>October 1, 2010 10:00:00 PM</td>
</tr>
</tbody>
</table>

**Figure 4. SFO Arrivals Route Network Model**

**iii. Associating the Radar Track Data with the Queuing Network Model**

Q-Gen associates radar tracks with the routes in the developed QNM. The track-route association is used to assign attributes such as timestamp air/ground speed and altitude windows from the track to the route. The problem of associating the radar track data with the queuing network model involves the following:

1. Associating a route for every track in the radar track data.
2. For an identified radar track and route pair, associating a point on the track data for every waypoint on the route. This will enable association of attributes such as (timestamp, air/ground speed, altitude) from the track data to the route waypoint. The track-route association is performed by segmenting each track and route into a large number of points $N$, by performing interpolation, with a typical value of $N = 400$. The metric $d$ that measures the closeness of association is the root mean square distance between the points on the track and the route.

\[
d = \sqrt{\frac{\sum_{i=1}^{N} (\text{distance}(\text{track}_i - \text{route}_i))^2}{N}}
\]  

**Figure 5.** Associating a Modeled Route for Every Track in the Radar Track Data

**Figure 6.** Associating a Point on the Track Data for Every Waypoint on the Route

**iv. Assignment of Altitude Windows based on Historic Radar Tracks**

The objective of this task is to assign altitude windows to various waypoints along the routes in the terminal area. Figure 7 shows a graphical representation of the altitude windows for an arrival route.

**Figure 7.** Altitude Windows for Various Waypoints along an Arrival Route
After the isolation of radar tracks that closely follow a given terminal area route has been completed as described in Section II.A.iii, the altitude windows can be generated. As explained in Section II.A.iii, each waypoint in the route can be mapped to a point in the associated track that is closest to it. Using this information, the altitude data from the track can be attributed to the route waypoint. Altitude data from multiple tracks are collected at each waypoint to find the minimum and maximum altitude at that waypoint, thus determining the altitude window.

v. Automatic Determination of Route Intersections in 3D

After the assignment of altitude windows for the various routes, the Q-Gen software runs a check to determine 3D intersections between all route segments in the terminal area. After the 3D intersections have been determined, a shared server can be added at the intersection node by the Q-Gen software. This capability is valuable to determine 3D intersections between the arrival and departure route for an airport or for multiple airports in a metroplex, and their subsequent modeling into a queuing framework.

vi. Merging Multiple Models

Q-Gen provides the capability to load and merge multiple QNMs. This capability is useful in (i) creating a combined arrival departure model from separate arrival and departure models of a given airspace or (ii) creating a metroplex model from the models of constituent airports. Figure 8 shows a combined arrival departure model for the SFO terminal airspace created by merging the arrival and departure models that were created separately. Q-Gen checks for 3D intersections after merging multiple models.

Figure 8. Merging Multiple Queuing Network Models
B. Partitioning Terminal Area Routes into Servers

After the route network model of the terminal airspace is created, Q-Gen provides the capability to partition the model into servers based in the separation assurance criteria. The server model can then be used by DEQS to perform the queuing simulation. The server model resulting from the serverization process is shown in Figure 9.

![Figure 9. SFO Arrivals Server Model](image)

The servers are classified according to the following types:

1. **Nominal Servers**: These servers have a single entry point and a single exit point as shown in Figure 10. These servers form the segments of the queuing network model between two merge, diverge or intersection nodes. Note that the entry points are shown in orange and the exit points are shown in green. Nominal servers are created along the branches of the route network, both at straight segments and turns as shown in sub-figure (a) and sub-figure (b) respectively.

2. **Shared servers**: These servers have multiple entry and exit points. Shared servers are found at merge, diverge and intersection nodes in the queuing network as shown in Figure 11.

![Figure 10. Nominal Servers](image)
Q-Gen provides the capability to specify the sizes of the nominal and shared servers for modeling different separation assurance criteria. For enforcing a constant 3-nmi separation throughout the entire terminal airspace, the sizes of both the nominal and the shared servers can be set to 3 nmi. For enforcing separation based on the wake vortex criteria, 1 nmi nominal servers are created along the branches of the QNM and 3 nmi shared servers are created at the merge diverge and intersection nodes. Note that according to the published FAA ATC regulations, wake vortex separation must be enforced for aircraft directly behind the flight path of the leading aircraft. A constant separation of 3 nmi is required at merge points when the trailing aircraft is not on the flight path of the leading aircraft. The DEQS simulation maintains multiple server separation for the nominal servers to enforce the appropriate wake separation and single server separation across 3-nmi shared servers at merges.

C. Demand Data Export

After the serverization is complete, Q-Gen generates the data required for performing a DEQS simulation such as:

1. The route and server network topology which includes the waypoints, nodes and servers and their respective connectivities.

American Institute of Aeronautics and Astronautics
2. Flight demand data by associating historic radar tracks with the developed queuing network model as described in Section II.A.iii; that includes (i) Nominal Times of Arrival (NTA) for flights, (ii) sequence of servers along the planned flight route and (iii) server transit times based on BADA aircraft performance data. The nominal times of arrival and transit times can be provided to terminal area scheduling algorithm to generate a planned schedule.

III. Discrete Event Queuing Simulator (DEQS)

This section presents the concept development and implementation of the Discrete Event Queuing Simulator (DEQS), which is used to perform fast-time simulation of terminal airspace operations. DEQS performs a discrete-event simulation in which the flights are not propagated by integrating their equations of motion; instead, the simulation propagates by jumping from one significant event to the next. Queuing behavior is enforced by delaying flights until the downstream server is available.

A. DEQS Algorithm for Propagating Flights

The objective of the DEQS algorithm is to propagate flights along their respective server sequences by allowing a flight to enter the next server in its server sequence, only if no other flight is currently occupying the downstream server. To achieve this, a numerical simulation of a queue is performed. The queuing logic is as follows. A flight enters a server and finishes unimpeded service. This instant is noted as serviceEnd Time. If the downstream server is unoccupied, the flight then transitions to the downstream server. The exitTime from the previous server is the same as the serviceEnd Time and the waitTime is zero. If the downstream server is occupied, the flight waits in the current server until the downstream server is available. The flight accrues waitTime in the current server and serviceEnd Time + waitTime = exitTime. Thus, the simulation follows the ‘wait after service’ paradigm and allows for the phenomenon of ‘blocking’.

![DEQS Flow Chart](image-url)

Figure 13. DEQS Flow Chart
The simulation maintains a queue of the next events for all flights in the simulation. The next event is defined as the scheduled entry of a flight at a server. Hence the queue stores the scheduledEntryPoint for the i-th flight into the server that it is supposed to enter next. The flight with the minimum scheduledEntryPoint (next impending event), is selected for processing. The algorithm for processing flights through the queuing network model is summarized in Figure 13.

B. Enforcing Separation Criteria

The DEQS simulation can enforce both the 3-nmi and wake vortex separation criteria as elaborated further in the following sub sections.

i. 3-nmi Separation

DEQS algorithm maintains a separation of 3 nmi between all aircraft by dividing the terminal area routes into 3-nmi servers and by ensuring that every server is occupied by only one aircraft at any given time. This is accomplished by delaying an aircraft in a server until the downstream server is available.

ii. Wake Vortex Separation

According to the published FAA ATC regulations, wake vortex separation must be enforced for aircraft directly behind the flight path of the leading aircraft. The DEQS algorithm allows for variable inter-aircraft in-trail separation based on the wake categories of the leading and the trailing aircraft as shown in Table 1.

<table>
<thead>
<tr>
<th>Separation Distance (nmi)</th>
<th>Leading Aircraft: Wake Category</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Heavy</td>
</tr>
<tr>
<td>Trailing Aircraft:</td>
<td>Heavy</td>
</tr>
<tr>
<td>Wake Category</td>
<td>B757</td>
</tr>
<tr>
<td>Wake Category</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td>Small</td>
</tr>
</tbody>
</table>

Variable wake separation can be realized in DEQS by dividing the terminal area route network into 1 nautical mile servers and ensuring that appropriate number of servers exists between two aircraft as per Table 1. Wake separation for all aircraft types trailing behind a heavy aircraft is illustrated in Figure 14.

A constant separation of 3 nmi is required at merge points when the trailing aircraft is not on the flight path of the leading aircraft. Hence, the DEQS algorithm enforces wake separation along a branch of the arrival network and 3 nmi separation across merge points. This requires a Queuing Network Model (QNM) in which all branches are divided into 1 nmi servers and 3 nmi shared servers are created at the merge points as shown in Figure 15.
C. Multi-Threaded DEQS implementation for Multi-Core Computers

A multi-threaded implementation of the DEQS algorithm was developed for implementation on multi-core computers. This multi-threaded implementation performs multiple iterations of the DEQS Monte Carlo simulation in parallel. Note that 16 concurrent Monte Carlo iterations can be performed on an 8-core computer which leads to fast execution. The multi-threaded DEQS implementation can perform 10,000 Monte Carlo iterations of a 6-hour LAX simulation with 260 flights in less than 90 seconds.

D. Comparison of DEQS with STASS

This section compares DEQS with Stochastic Terminal Area Scheduling Simulation (STASS)\textsuperscript{10,11} developed at the NASA Ames Research Center, and currently used for stochastic analysis of terminal area operations. STASS is comprised of a terminal area scheduler and a terminal area simulation. DEQS can perform terminal area simulation with higher fidelity than STASS for the following reasons.

1. **Separation Assurance at Finer Spatial Resolutions**: While performing terminal area simulation STASS ensures separation only at the meter fix and the runway. The STASS simulation is currently being enhanced to ensure separation at all merge points in addition to meter fix and runways. DEQS uses a high-fidelity terminal area route model created using Q-Gen. The DEQS simulation ensures separation at the meter fixes, runways and merge/diverge/intersection points and also at every 1 nautical mile interval along the routes in between the merge points.

2. Aircraft **Type Specific Transit Time Models**: STASS classifies aircraft into 3 categories jet, turboprop and piston and uses a transit time for each category. Thus, the transit times for all jet aircraft are aggregated in spite of known differences in the transit time between individual jet aircraft types. DEQS on the other hand uses BADA\textsuperscript{3} aircraft performance models to obtain accurate transit times specific to each aircraft type along high-fidelity routes.

3. **Spatio-Temporally Correlated Wind Uncertainty Model**: STASS introduces perturbations in the arrival times at the meter fix, runway and various merge points by drawing a random number from a Gaussian distribution of arrival times at each point. If the random arrival times at the various points are not correlated, artifacts are introduced in the simulation. On the contrary, DEQS uses a spatio-temporally correlated wind uncertainty model to determine transit time uncertainties for various segments of the terminal area route network.
IV. Metrics for Analysis of Terminal Airspace Queuing Networks

This section describes the various metrics that can be calculated from the output of a DEQS simulation to perform analysis of the terminal airspace network.

A. DEQS Output

The DEQS implementation calculates the following quantities for every flight, for every server on the flight’s planned route:

1. **entryTime**: The actual time that the flight enters the server. This may be different from the scheduleEntryTime if a wait is encountered prior to entering the current server.
2. **serviceEndTime**: The time when the flight completes its unimpeded service in the sector.

   \[
   \text{serviceEndTime} = \text{entryTime} + \text{serviceTime}
   \]

3. **exitTime**: The time when the flight exits the server. This may be different from the serviceEndTime if the downstream server is busy and wait is encountered.

   \[
   \text{exitTime} = \text{entryTime} + \text{serviceTime} + \text{waitTime}
   \]

   \[
   \text{exitTime} = \text{serviceEndTime} + \text{waitTime}
   \]

All other metrics of interest can be calculated from these quantities as described in the following sections.

B. Metrics from a Single DEQS Simulation

1. **Delay per Flight**: The total delay experienced by a flight. The delay for a flight at a given server is calculated as

   \[
   \text{Delay}_{ij} = \text{exitTime}_{ij} - \text{serviceEndTime}_{ij}
   \]

   where the subscript \( i \) indicates the \( i^{th} \) flight and the subscript \( j \) indicates the \( j^{th} \) server. Hence total delay per flight is given by

   \[
   \text{DelayPerFlight}_i = \sum_j \text{Delay}_{ij}
   \]

   where the index \( j \) runs over all the servers that the flight passes through.

2. **Total Delay at a Server**: The total delay experienced at a given server by all flights that pass through that server.

   \[
   \text{DelayAtServer}_j = \sum_i \text{Delay}_{ij}
   \]

   where the index \( i \) runs over all flights that pass through server \( j \). This metric helps in identification of choke points in the terminal airspace network were congestion delay is large.

3. **Traffic Flow Efficiency**: Traffic Flow Efficiency for a given flight is defined as the ratio of the unimpeded flight time to the total flight time and can be calculated as follows

   \[
   \text{Traffic Flow Efficiency} = \frac{\text{Total Flight Time} - \text{Delay}}{\text{Total Flight Time}} \times 100
   \]

4. **Percentage Utilization of the Runway Servers**
   a. For the 3-nmi separation simulation:

   \[
   \text{Runway Utilization} = \left( \frac{\sum_i (\text{exitTime}_{ij} - \text{entryTime}_{ij})}{\max_i (\text{exitTime}_{ij} - \text{min} \text{ EntryTime}_{ij})} \right) \times 100
   \]

   where subscript \( j \) indicates the runway server and \( i \) indicates all the flights that flew through the runway server. Note that the denominator includes the time from the entry of the first flight into the runway server to the exit of the last flight from the runway server and thus includes the total time for processing all aircraft along with the gaps in between. The numerator is the total time that the runway server was occupied. For arriving flights, the last server on the final approach path for each runway is considered. For departure flights the first server for departures from a runway is considered.
b. For the wake-separation simulation: For the case of variable inter-aircraft separation based on wake vortex criteria, the runway utilization can be calculated as described below. Consider all aircraft landing on a given runway in pairs composed of the leading and the trailing aircraft as shown in Figure 16.

\[
\text{Runway Utilization} = \left(1 - \frac{\sum_{\text{All Landing Pairs}} \text{t}^{\text{Entry}}(t,s(l,t)) - \text{t}^{\text{Exit}}(l,s1)}{\max_{\text{All Landing Pairs}} \text{t}^{\text{Entry}}(t,s(l,t)) - \min_{\text{All Landing Pairs}} \text{t}^{\text{Exit}}(l,s1)}\right) \times 100
\]

Figure 16. Procedure for Calculating Runway Utilization

If the mandated wake-based server separation between the leading and trailing aircraft is denoted by \(s(l,t)\), the earliest time that the trailing aircraft can enter server \(s(l,t)\) is equal to the exit time of the leading flight from the landing server \(s1\). Thus, any delay from this earliest entry time leads to lower runway utilization. This gives us the formula for calculating runway utilization as follows.

5. Number of Probable Separation Violation Events: In the DEQS simulation, when a flight finishes unimpeded service at a server and tries to enter the downstream server before the earliest available time of the downstream server, the flight encounters delay. This can be termed as a probable separation violation event. Thus, the number of probable separation violation events gives an idea of the degree of congestion in the traffic.

6. Number of Probable Separation Violation Pairs: The number of probable separation violation pairs is the unique aircraft pairs that have one or more probable separation violation events.

7. Average Inter-Aircraft Landing Time: This metric gives the average time interval between successive landings at the runways of the queuing model. This is a measure of the runway throughput.

8. Makespan: Makespan is defined as the time interval between the entry of the first flight in the demand data at the meter fix and the landing of the last flight at the runway. Similar definition can be generated for departure flights.

C. Stochastic Metrics from Monte Carlo Simulation

Multiple Monte Carlo iterations can be run for the DEQS simulation with various perturbations such as (i) uncertainty in the entry time to the network and (ii) uncertainty in the transit time for a server due to wind prediction uncertainty. The results of the various Monte Carlo iterations can be combined by generating histograms for the various variables listed in Section IV.B.

V. Analysis of Arrivals in to LAX Terminal Airspace

This section describes the results of the stochastic analysis of arrivals into the LAX terminal airspace queuing network model shown in Figure 17. The results were generated using the a DEQS simulation that enforces wake vortex separation and uses the First-Come-First-Serve (FCFS) queuing discipline at all servers.
Table 2 lists the top 10 flights in descending order of congestion delays. Note that the reported delays include only the tactical delays required for separation assurance and do not include the strategic scheduling delays that modify the nominal flight times of the aircraft. Note that the maximum delay is 97 seconds. Table 3 shows a list of the top 10 servers where the most amount of congestion delay was experienced. Figure 18 shows the locations of the most delayed servers with the server number and its delay rank in parenthesis. Note that the congestion delays mainly occur at (i) servers on final approach close to the runway due to the phenomenon of ‘compression’ and at (ii) merge points in the arrival network where the traffic flows merge.

### Table 2. Delays for the Top 10 Most Delayed Flights

<table>
<thead>
<tr>
<th>No.</th>
<th>Flight Id</th>
<th>Aircraft Type</th>
<th>Delay (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>49</td>
<td>B752</td>
<td>97.17</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>B752</td>
<td>44.08</td>
</tr>
<tr>
<td>3</td>
<td>18</td>
<td>B738</td>
<td>35.34</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>CRJ2</td>
<td>32.72</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
<td>B739</td>
<td>31.79</td>
</tr>
<tr>
<td>6</td>
<td>93</td>
<td>CRJ2</td>
<td>30.35</td>
</tr>
<tr>
<td>7</td>
<td>90</td>
<td>A320</td>
<td>30.08</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>E135</td>
<td>28.47</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>B737</td>
<td>28.12</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>B77W</td>
<td>27.52</td>
</tr>
</tbody>
</table>

### Table 3. Delays for the Top 10 Most Delayed Servers

<table>
<thead>
<tr>
<th>No.</th>
<th>Server Id</th>
<th>Delay (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>214</td>
<td>86.71</td>
</tr>
<tr>
<td>2</td>
<td>215</td>
<td>82.98</td>
</tr>
<tr>
<td>3</td>
<td>229</td>
<td>76.67</td>
</tr>
<tr>
<td>4</td>
<td>228</td>
<td>74.93</td>
</tr>
<tr>
<td>5</td>
<td>213</td>
<td>74.14</td>
</tr>
<tr>
<td>6</td>
<td>230</td>
<td>73.25</td>
</tr>
<tr>
<td>7</td>
<td>212</td>
<td>61.86</td>
</tr>
<tr>
<td>8</td>
<td>227</td>
<td>59.04</td>
</tr>
<tr>
<td>9</td>
<td>140</td>
<td>36.32</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>36.14</td>
</tr>
</tbody>
</table>
Figure 18. Locations of the Top 10 Delayed Servers in the LAX Arrival QNM

Figure 19 through Figure 24 show the histograms for the various queuing metrics generated using 10,000 Monte Carlo iterations. Note that the definitions for all the queuing metrics presented here are provided in Section IV.

Figure 19 shows the histogram of congestion delays for the most delayed flight (Flight ID 49) whose expected value of delay is 97 seconds. This figure shows that the distribution of congestion delays for this flight over 10,000 Monte Carlo simulations is bimodal. For about 3500 iterations out of the total 10,000 Monte Carlo iterations, the flight experiences delays in the range of 0-10 seconds. For the remaining 6500 iterations, the flight experiences delays of about 150 seconds. This bimodal behavior can be explained by the fact that the flight switches positions in the landing sequence due to the perturbations in the various iterations. These position switches can happen at entry points or merges in the arrival queuing network. Also, note that this position switching and the resultant delays are revealed only in the complete histogram and not in the expected value of the delays. This also shows the advantages of the simulation based approach as opposed to an analytical queuing theory approach that is based on unimodal distributions such as Gaussian or Erlang.

Figure 19. Histogram of Congestion Delay for the Most Delayed Flight

Figure 20 shows that the most delayed server has a mean value of delay of 87 seconds but the delays can be in the 30-140 second range for an individual Monte Carlo iteration.
Figure 20. Histogram of Congestion Delay for the Most Delayed Server

Figure 21 shows the histogram of congestion delays for all flights in the simulation. Note that this histogram is plotted by considering the average delay for every flight over all 10,000 Monte Carlo iterations. This figure reveals that 70 flights experience 0 congestion delays and delays for most of the flights are below 30 seconds. Figure 22 shows the histogram of congestion delays for all servers. It shows that congestion delays are not experienced at most servers (~280).

Figure 23 shows the histogram of percentage utilization for runway RW 24R, which ranges between 39.5% and 43.5%. Note that the definition of runway utilization (Section IV.B) does not include the scheduling buffer used at the runway. The metric will give 100% efficiency only if all flights land with the exact mandated 3 nmi or wake vortex separation, without any buffers to manage uncertainties.

Figure 24 shows the histogram of average efficiency for all flights in the simulation. The flight efficiency is 100% if the flight does not experience any congestion delays. For a flight with 95% efficiency, 95% of the flight time is unimpeded while 5% of the flight time is due to congestion delays. This figure shows that most flights have efficiencies in excess of 95% with the efficiency of the least efficient flight being 90%.

17

American Institute of Aeronautics and Astronautics
VI. Summary and Concluding Remarks

This paper presented two software tools for stochastic analysis of terminal airspace operations. The first tool called Queue Generator (Q-Gen) develops a route network model of the terminal airspace under study and segments the routes into smaller servers based on the inter-aircraft separation criteria. The second tool called Discrete Event Queuing Simulator (DEQS) performs a queuing simulation by propagating flights along the servers of the route network, while ensuring that the capacity of each server is not exceeded. The output from this queuing simulation can be used to generate the queuing metrics such as delay, throughput and traffic flow efficiency. Monte Carlo simulation can be performed to determine the effect of uncertainty in meeting the scheduled arrival times and the wind prediction uncertainty. DEQS can perform 10,000 Monte Carlo iterations of a 6-hour terminal area queuing simulation in less than 90 seconds.

Q-Gen allows the modeling of the terminal airspace routes with high fidelity. DEQS allows the simulation of a planned schedule and propagation of flights, where each flight can be assigned distinct transit times, without any aggregation of flights into flows as is commonly done in queuing analysis. The simulation enforces separation at every node of the server network (every 1-nmi interval) and not only at the entry fixes and merges points. Thus, Q-Gen and DEQS can perform a high-fidelity queuing simulation of a terminal airspace. Also the extremely fast processing speed (10,000 Monte Carlo iterations of a 6-hour terminal area queuing simulation in less than 90 seconds) makes real-time stochastic analysis feasible. This opens up the possibility of the use of DEQS as an operational tool in strategic planning of terminal area operations.

Acknowledgments

This research was supported under NASA Phase II SBIR Contract No. NNX11CA08C monitored out of the NASA Ames Research Center. The authors thank Douglas Isaacson, Alexander Sadovsky and John Robinson of the NASA Ames Research Center for many fruitful discussions and feedback which helped improve the quality of this work.

References

4 Rapid Update Cycle (RUC): http://ruc.noaa.gov/, accessed on 07/22/2013
8 FAA webpage on JO 7110.65s: Air Traffic Control: www.faa.gov/documentlibrary/media/order/7110.65s.pdf