2-1997

Analysis of Aircraft Sortie Generation with Concurrent Maintenance and General Service Times

Daniel V. Hackman

ANALYSIS OF AIRCRAFT SORTIE GENERATION WITH CONCURRENT MAINTENANCE AND GENERAL SERVICE TIMES

THESIS

Daniel V. Hackman, Major, USAF

AFIT/GOR/ENS/97M-11

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/97M-11

ANALYSIS OF AIRCRAFT SORTIE GENERATION WITH CONCURRENT MAINTENANCE AND GENERAL SERVICE TIMES

THESIS

Daniel V. Hackman, Major, USAF

AFIT/GOR/ENS/97M-11

Disclaimer

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government

AFIT/GOR/ENS/97M-11

ANALYSIS OF AIRCRAFT SORTIE GENERATION

WITH CONCURRENT MAINTENANCE

AND GENERAL SERVICE TIMES

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Research

Daniel V. Hackman, B.S.

Major, USAF

February, 1997

Approved for public release; distribution unlimited

THESIS APPROVAL

Student: Daniel V. Hackman, Major, USAF Class: GOR-97M

Title: Analysis of Aircraft Sortie Generation With Concurrent Maintenance and General Service Times

Defense Date: 28 February 1997

Committee:	Name/Title/Department	Signature
Advisor	Dennis C. Dietz, Lt Col, USAF Assistant Professor of Operations Research Department of Operational Sciences	
Reader	W. Paul Murdock, Jr., Major, USAF Assistant Professor of Operations Research Department of Operational Sciences	

Acknowledgments

I want to thank my thesis advisor, Lieutenant Colonel Dennis C. Dietz, for his support and guidance. He allowed me the freedom to explore many different avenues in my attempt to define and complete this thesis project. When it came time to make the hard decisions, he was there to help keep me motivated and focused on my objectives. His work formed the starting point for my research and saved a lot of time when it came to programming the algorithm and validating the methodology.

I also wish to thank my fellow GOR-97M classmates for their support. Their enthusiasm and encouragement helped me keep the "big picture' in sight, and helped keep me going when the going got tough.

Finally, I must thank my wife, Rose, my daughter, Chrissy, and my son, Alex, for the sacrifices they made during our tour at AFIT. Their encouragement and understanding helped immeasurably. They did without, and pushed me onward, so I could finish what needed to be done. I will always treasure their love and support.

Daniel V. Hackman

Table of Contents

List of Figures

List of Tables

AFIT/GOR/ENS/97M-11

Abstract

The primary objective of this study was to develop an analytical methodology for evaluating an aircraft sortie generation process. The process is modeled as a closed network of general service queues with a fork-join node to model concurrent servicing. The model uses the Mean Value Analysis (MVA) algorithm and general queueing network analysis by decomposition to approximate network performance measures including resource utilization and the overall sortie generation rate.

The results of the study show that the analytical approximation's accuracy decreases as server utilization increases. However, when server utilization is kept in realistic ranges, the approximation is very accurate. When applied to a closed system of single server queues and delay stations, the approximation performs significantly better than a pure MVA-based approach. For closed or capacitated open systems with multiserver queues, the approximation can still be applied to provide upper and lower bounds on system performance.

ANALYSIS OF AIRCRAFT SORTIE GENERATION

WITH CONCURRENT MAINTENANCE

AND GENERAL SERVICE TIMES

I. Introduction

1.1 Motivation

The United States Air Force's ability to maximize aircraft employment effectiveness is strongly influenced by the aircraft sortie generation process. *Effectiveness* represents the combined effects of both qualitative measures such as *mission capability* (how well an aircraft can perform its designated mission) and quantitative measures such as *aircraft availability* (the portion of time aircraft are available for employment) and the *sortie generation rate* (the expected number of sorties available per aircraft per time period). Whether aircraft are used directly in combat or in any of a wide variety of support roles, many quantitative performance measures are determined by the sortie generation process. These parameters are often very difficult to estimate since they depend on both aircraft systems reliability and the supporting logistics system's resource availability [6]. A realistic analytical model would be extremely useful to decision makers by enabling rapid estimation of system performance measures, identification of bottlenecks and determination of the distribution of resources required to attain a target sortie generation rate or tempo of operations.

1.2 Background

The aircraft sortie generation process is fairly similar for all aircraft types. Generally, aircraft are readied for flight, fly a sortie, land, and prepare for their next sortie. The preparation process typically includes refueling, scheduled and unscheduled maintenance, cargo loading, and inspections. While cargo types and loading times may differ for each aircraft type (fighters and bombers load weapons, transports load pallets and vehicles, tankers load additional fuel, etc.), the basic flow remains the same.

The aircraft sortie generation process can be viewed as a system of entities (aircraft, personnel and equipment) which interact together. By partitioning the process into a set of mutually exclusive aircraft activities (taxi, fly, repair, etc.) and counting the number of aircraft occupied with each activity, the state of the system can be described at any given time. The state only changes at the finite times when aircraft move from one activity to another. This is known as a *discrete system* [3:9].

The aircraft sortie generation process can be modeled as a discrete queueing system where aircraft are the "customers" and the resources required to service them are the "servers." The system is represented by a set of N aircraft and M indexed stations in a closed queueing network. The state of the system at any time is defined by the number of aircraft at each station. Aircraft flow through the network by proceeding through a sequence of activities, or stations. For this study, activities are aggregated into six main functional areas: 1) *taxi* includes ground operations and inspections prior to flight; 2) *sortie* represents the actual flying time; 3) *troubleshooting* covers malfunction diagnosis and downloading weapons prior to repairs; 4) *repair* represents unscheduled maintenance on the aircraft; 5) *turn-around* includes refueling and scheduled maintenance activities between sorties; and 6) *munitions* includes weapon loading and arming prior to flight. After spending the required service or delay time (s_i) at station i, aircraft cycle through the network, proceeding from station i to station j with probability p_{ij}. Figure 1.1 diagrams the aircraft flow within the network.

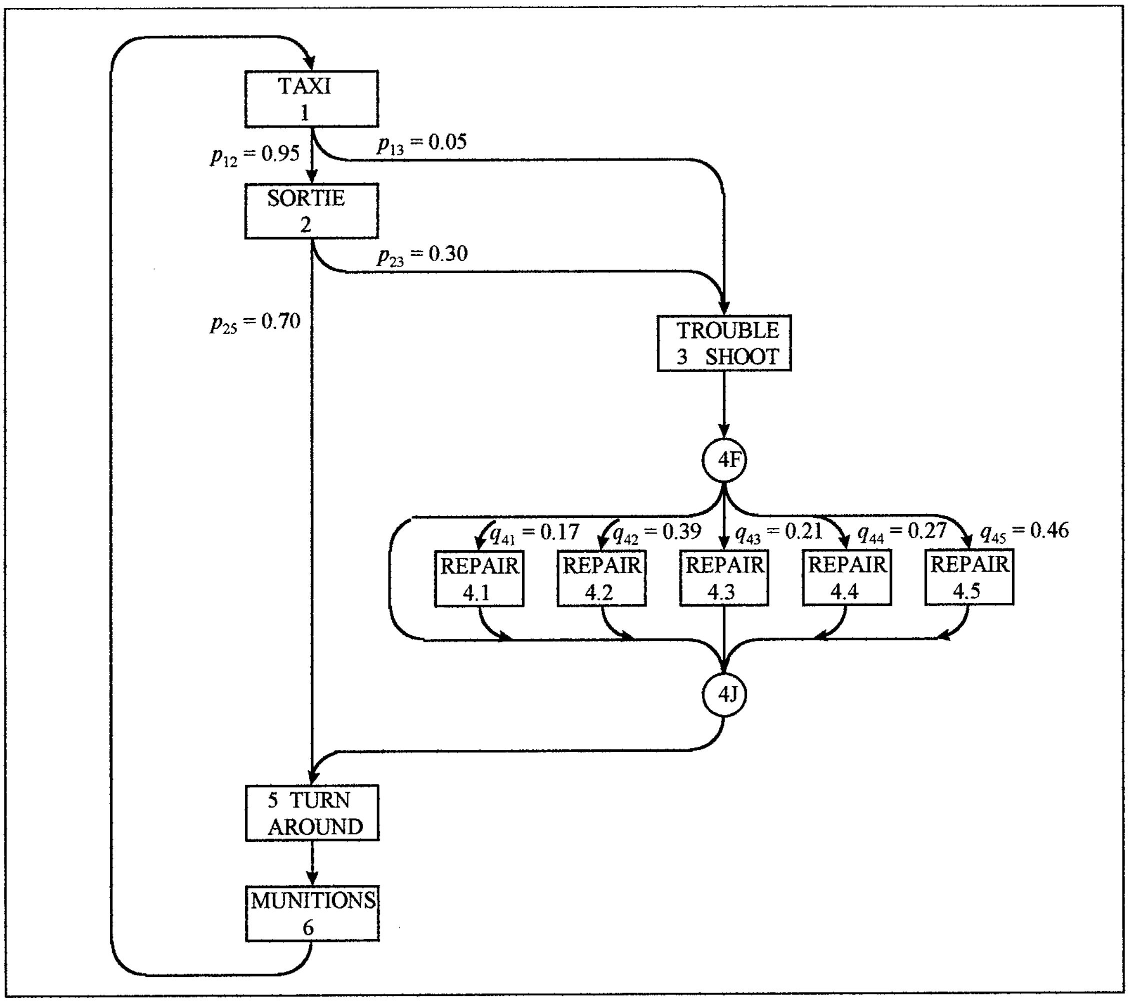

Figure 1.1: Aircraft Sortie Generation Process

Starting at station 1, *taxi*, the crew prepares the aircraft for a sortie. With probability $p_{13} = 0.05$ the aircraft is unable to fly the mission and aborts, proceeding to station 3, *troubleshoot*. Otherwise, it takes off and flies its *sortie* (delay at station 2) then returns to base. Upon landing, some aircraft require additional maintenance before their next flight and proceed to station 3 with probability $p_{23} = 0.30$ while the rest proceed directly to *turn-around* (station 5) to prepare for their next flight. *Troubleshooting* is represented by a delay at station 3 during which specific malfunctions are identified. Aircraft with malfunctions proceed to station 4, *repair*, where maintenance specialists repair the malfunctioning systems. Following repairs, aircraft proceed to *turn-around* (station 5), followed by *munitions* (station 6) to be readied for their next flight. Finally, readied aircraft arrive at the *taxi* station and begin the cycle again.

The repair station is the most interesting in the entire network. Here, aircraft may require several different types of maintenance for different malfunctioning systems. For the purposes of this study, aircraft systems are aggregated into five major categories: airframe, electrical/hydraulic, engine, avionics, and radar/weapons control. Because some repairs may be accomplished concurrently, a series of repair stations, one for each repair type, would overestimate the amount of time required to complete all repairs. Therefore, a fork-join structure is necessary. Here, aircraft arriving at the repair station "fork" node create "clones" of themselves, one for each required service type. The clones follow their own repair paths, then wait at the "join" node for their siblings to complete repairs. Each repair type ($k = 1,\ldots,5$) has its own unique mean service time (s_{4k}), resource level (m_{4k}), and conditional probability the repair will be required given a malfunction has occurred (q_{4k}). These probabilities are independent, so any malfunctioning aircraft may require any combination of the offered repair services. As a result, a false malfunction indication occurs with probability $q_{40} = \Pi_{k=1}^{5}(1-q_{4k})$, in which case the *repair* queues are bypassed and the aircraft proceeds directly to the *turn-around* station.

1.3 Problem

Discrete systems can be analyzed using analytical methods, such as queueing theory, or by empirical methods, such as discrete-event simulation [3:13]. Unfortunately, simulation models for complex systems, such as aircraft sortie generation, are expensive and time consuming, requiring careful analysis and multiple replications for each scenario just to provide acceptable confidence bounds on the results [3:5]. While analytical queueing network models may provide an alternative to simulation, they often require unrealistic simplifying assumptions to make them mathematically tractable. "Closed-form models are not able to analyze most of the complex systems that are encountered in practice" [3:6]. Realistic analytical models are useful to decision makers since they can rapidly estimate system performance measures, identify bottlenecks and determine the distribution of resources required to meet desired throughput goals.

Unfortunately, several aspects of the aircraft sortie generation process are difficult to model using queueing theory. These include:

1. Concurrent Maintenance: This problem arises when aircraft require two or more types of repair activity which may be performed simultaneously. A fork-join structure can be used to model concurrent maintenance as illustrated by station 4 of figure 1.1. However, queueing theory does not provide exact performance measures and a heuristic is necessary to approximate performance measures for

this type of system. Concurrent maintenance has been successfully modeled in exponential service networks [9, 14].

2. Maintenance Crew Size: Personnel in one maintenance specialty may be responsible for several different types of repairs, each requiring different amounts of time and resources. Queueing models may use average crew sizes based on service (resource) requirements for each maintenance specialty. The number of servers available is then defined as the total number of personnel divided by the average crew size. This aggregation is necessary for mathematical tractability, but fails to capture some of the variability of the service processes.

3. Machine-Operator Interference: Machine-operator interference describes the unnecessary idle time (waiting time) experienced by a machine (customer) when the operator (server) is busy with another machine [25:129]. When a resource performs more than one type of service or service at more than one station in a network, first-moment models such as MVA use only the weighted service time for the server. Thus, they cannot capture the variability of the aggregated service time distributions.

4. Batch Arrivals: While some support and special-purpose aircraft fly singly, most combat aircraft fly in formation as groups of two or more, resulting in the simultaneous arrival of multiple aircraft. Batch arrivals tend to cause more congestion, resulting in reduced system performance. Ignoring this situation could result in system performance measures being overstated.

5. Mission Scheduling: Once aircraft are readied for their mission, they usually wait until a scheduled launch time. Queueing network models typically release aircraft for launch immediately after service completion. Therefore, a model's performance measures, such as the throughput rate, should be viewed as an upper bound on system performance. Scheduling delays will result in lower throughput rates, hence lower server utilization.

1.4 Objective

The primary objective of this study is to develop an analytical methodology based on the MVA algorithm and general queueing network analysis by decomposition to evaluate an aircraft sortie generation process. The process is modeled as a closed network of general service queues with a fork-join node to model concurrent servicing. The model's purpose is to approximate network performance measures including waiting time, response time, queue length, resource utilization and the overall sortie generation rate. The queueing network to be solved is a modified version of the Aircraft Sortie Generation Model developed by Dietz and Jenkins [9].

1.4.1 Approach. The study objectives are accomplished as follows:

1. Develop an analytical methodology based on the MVA algorithm and general queueing network analysis by decomposition to approximate the performance of a network of general service queues and a fork-join queue with probabilistic branching.
2. Solve the queueing network model shown in figure 1.1.
3. Develop a simulation model to validate the analytical model results.

4. Compare the simulation model and analytical model results.

5. Repeat steps 2 through 4 at multiple levels of server utilization.

1.4.2 Scope

Due to the limited time available and problem complexity, this study considers single server (GI/G/1) and infinite server (GI/G/∞) queues only. Multiserver queues in the Aircraft Sortie Generation Model [9] are replaced with single server queues in this study. Extending the algorithm to accommodate multiserver (GI/G/m) queues is left for future research.

Concurrent maintenance is modeled using a fork-join queue. Variability resulting from aggregated crew sizes and service times (machine-operator interference) is captured by using the first two moments of the service distributions. Multiple classes of customers, batch arrivals, and mission scheduling are not modeled.

II. Previous Work

2.1 Introduction

Probably due to the complex nature to the aircraft sortie generation process, previous work in this area has primarily focused on simulation methods to gain insights into the nature of the system [1, 7, 11, 12, 21]. However, recent efforts are beginning to pay more attention to faster, more efficient analytical models [5, 6, 10, 22]. The first analytical models to address concurrent repair of aircraft subsystems use heuristics based on MVA and fork-join queues to approximate system performance [9, 14]. But these first moment models may not precisely capture the effects of known variability present in many real-world service processes. General queueing network analysis by decomposition can be applied to many systems to provide more accurate estimates of system performance measures [8, 16, 19, 27, 28]. This researcher's literature search did not find any example of the decomposition approach being applied to fork-join queues.

2.2 Simulation Models

Historically, analysts resorted to using simulation models such as the Logistics Composite Model (LCOM) [7, 12], the Sortie Generation Model (SGM) [1], Theater Simulation of Airbase Resources (TSAR) [11], and Dyna-Sim [21] to estimate system performance parameters. Jenkins describes each of these models in [14:2-1 - 2-4].

2.3 Analytical Models

The Fleet Maintenance System Design Model [5], B-52H/KC-135 Maintenance Model [22], Optimal Specialization of Maintenance Manpower Model [10] and the Small

Unit Maintenance Manpower Analyses (SUMMA) [6] are all analytical models used to estimate performance measures on vehicle maintenance issues. Jenkins describes each of these models in [14:2-5 - 2-7].

2.4 The Modified MVA Sortie Generation Model

The sortie generation process involves the concurrent repair of different aircraft subsystems. That is, when an aircraft lands, it may require several different types of service which may be performed at the same time. Dietz and Jenkins developed a new analytical model based on a modified MVA algorithm using fork-join queues to model concurrent maintenance, thus capturing more of the complexity of the sortie generation process [9, 14]. In the literature, fork-join queues are used to model throughput for a system of parallel processors where jobs arriving at the fork select one path [2:305] or are split into K siblings, sending one sibling down each of K paths [18:365]. Dietz and Jenkins modified this approach to allow selection of a variable number of paths corresponding to specific repair requirements [9].

These analytical models prove highly accurate when compared to simulation results for systems with exponential service times. However, their inherent exponential service time assumption prevents them from precisely capturing the effects of known variability present in many real-world service processes.

2.5 Stochastic Rendezvous Networks (SRVNs)

SRVNs are performance models designed for multitasking parallel software. In an SRVN, a *rendezvous* (RV) is a single client-server interaction where a client sends a request for service and then waits for the server's reply. When the server becomes available, the client and server are linked for service, blocking any further action by the client until service completion. Distributed software systems use RV for communication and synchronization. SRVNs can model multiple classes of customers with different service requirements, so aggregation of maintenance crew sizes and repair times would not be necessary. Unfortunately, since the customer is blocked during service, modeling concurrent maintenance is not possible [29:143].

2.6 The Queueing Network Analyzer (QNA)

The Queueing Network Analyzer is a software package developed at Bell Laboratories to estimate performance measures for open networks of queues with general service time and interarrival time distributions. The algorithm uses two parameters to describe the arrival and service processes, one for the rate and the other for the variability. Using a decomposition approach, the network's nodes are analyzed as GI/G/m queues using the first two moments of the inter-arrival time and service time distributions [27:2779].

2.7 The Manufacturing Workcell Machine/Operator Interference Model

This model estimates the performance of a manufacturing workcell consisting of *N* machines tended by a single operator. Using a decomposition approach and approximations of the first two moments of the service and interarrival time distributions, the model estimates operator-induced machine interference (the time machines must wait for the operator when the operator is busy with other machines) and the workstation utilization [8:576]. Since the arrival process is a function of the interference time and the interference time is a function of the arrival process, an iterative method is used to approximate these measures [8:583]. Desruelle and Steudel also develop an approximation for a conversion function that accounts for the transition from an infinite calling population in an open network to a finite calling population in a closed network [8:588-589].

2.8 Conclusion

The previous analytical aircraft sortie generation models [9, 14] assume all service and delay times are exponentially distributed in order to use the MVA algorithm, taking full advantage of the "memoryless" property of the exponential distribution to maintain mathematical tractability. Unfortunately, this assumption may not reflect real-world conditions, causing error in the results. An algorithm combining the features of [8] and [9] may permit an analyst to model concurrent maintenance and at the same time capture the effects of general service time distributions.

III. Methodology

3.1 Introduction

This research extends the aircraft sortie generation model developed by Dietz and Jenkins [9] to a system of GI/G/1 queues and delay stations where the interarrival and service time distributions are characterized by their first two moments. The method used in this study is based on the approach presented by Desruelle and Steudel to model machine/operator interference in a manufacturing workcell [8].

3.2 Overview

The method first solves the equivalent product-form network using the MVA algorithm [17, 23] and a heuristic developed by Dietz and Jenkins to accommodate fork-join queues [9]. Next, waiting times are adjusted using the current squared coefficients of variation (SCVs) of the general service and arrival processes [16:115; 17:664-665]. The revised waiting times are used to compute new cycle times, throughput rates, and station utilizations. The revised station utilizations are in turn used to update the SCVs for the arrival processes in an iterative process using the queuing network analysis traffic variability equations derived in [16, 27]. The updated arrival process SCVs are then used to update the waiting times and the process continues, iterating to convergence.

3.3 Model Design

3.3.1 Objective

The objective of this research is to develop an analytical methodology based on the MVA algorithm and general queueing network analysis by decomposition to approximate the performance of a network of general service queues and a fork-join queue with probabilistic branching. For demonstration purposes, the example system presented in figure 1-1 is modeled. Taxi times are assumed to be deterministic; sortie times are assumed to be uniformly distributed between 1.5 and 2.5 hours; and all other service times are assumed to follow lognormal distributions with SCV = 0.29. Table 3.1 summarizes the parameters used for the example network. Table 3.2 describes the notation used in this research. The subscript *i* always refers to node *i* of the network. For a fork-join node, the subscript *ik* represents substation *k* of node *i*.

Table 3.1. Sortie Generation Model Parameters

Activity	Service Time Mean	Service Time SCV	Repair Probability	Resource Level
Taxi	$s_1 = 0.25$	$scv_1 = 0.00$		$m_1 = \infty$
Sortie	$s_2 = 2.00$	$scv_2 = 0.02$		$m_2 = \infty$
Troubleshoot	$s_3 = 0.50$	$scv_3 = 0.29$		$m_3 = \infty$
Rpr1 (airframe)	$s_{41} = 2.20$	$scv_{41} = 0.29$	$q_{41} = 0.17$	$m_{41} = 1$
Rpr2 (electrical/hydraulic)	$s_{42} = 2.27$	$scv_{42} = 0.29$	$q_{42} = 0.39$	$m_{42} = 3$
Rpr3 (engine)	$s_{43} = 2.37$	$scv_{43} = 0.29$	$q_{43} = 0.21$	$m_{43} = 2$
Rpr4 (avionics)	$s_{44} = 1.50$	$scv_{44} = 0.29$	$q_{44} = 0.27$	$m_{44} = 1$
Rpr5 (radar/weapons control)	$s_{45} = 1.19$	$scv_{45} = 0.29$	$q_{45} = 0.46$	$m_{45} = 2$
Turn-around	$s_5 = 0.75$	$scv_5 = 0.29$		$m_5 = 6$
Munitions	$s_6 = 0.50$	$scv_6 = 0.29$		$m_6 = 4$

Table 3.2. Notation

c_{ai}^2	Squared coefficient of variation for the arrival process at node *i*.
c_{di}^2	Squared coefficient of variation for the departure process at node *i*.
c_{ij}^2	Squared coefficient of variation for the flow from node *i* to node *j*.
c_{si}^2	Squared coefficient of variation for the service process at node *i*.
$CT_1(N)$	Cycle time station 1.
M	Number of stations in the closed network.
m_i	Number of servers at station *i*.
N	Number of customers in the network, or calling population.
$\boldsymbol{P}$	Routing probability matrix.
p_{ij}	Routing probability that a customer completing service at station *i* next goes to station *j*.
$p_{ik}(N)$	Marginal probability that the last clone at fork-join node *i* finishes service at station *ik* when *N* customers are in the system.
q_{ik}	Conditional probability that a customer requires service type *ik* given it arrives at station *i*.
q_{i0}	Conditional probability that a customer requires no service given it arrives at station *i*.
$Q_i(N)$	Mean number of customers at station *i* (waiting or in service) with *N* customers in the network.
$R_i(N)$	Mean response time (waiting time and service time) at station *i* with *N* customers in the network.
$T_{ik}(N)$	Mean response time (waiting time and service time) at fork-join station *ik* with *N* customers in the network.
$\{S\}$	Set of services required at a fork-join queue.
s_i	Mean service time at station *i*.
v_i / v_1	Visit ratio, or average number of visits a customer makes to station *i* relative to station 1 in a closed network.
$W_i(N)_e$	Mean waiting time at station *i* with *N* customers in a closed network for an M/M/1 queue.
$W_i(N)$	Mean waiting time at station *i* with *N* customers in a closed network for a GI/G/1 queue.
$W_i(\infty)_e$	Mean waiting time at station *i* in an open network for an M/M/1 queue.
$W_i(\infty)$	Mean waiting time at station *i* in an open network for a GI/G/1 queue.
$\lambda_i(N)$	Throughput rate at station *i* with *N* customers in the system.
$\mu_i = 1/s_i$	Service rate at station *i*.
$\pi_i\{S\}$	Probability services in subset *S* of fork-join queue *i* are required.
ρ_i	Server utilization at station *i*.

3.3.2 The MVA Algorithm

The MVA algorithm can be applied to closed or capacitated open networks with N customers moving through an arbitrary but finite number M stations having symmetric service disciplines: first-come-first-serve (FCFS) exponential, processor sharing, infinite server, and preemptive-resume last-come-first-serve (LCFS) [4:250]. If concurrent repair activities (station 4) are removed from the example network and all service times are assumed to be exponentially distributed, performance measures for the remaining network can be computed using MVA. The *Arrival Theorem*, the foundation of MVA, states that, for a closed network with N customers, an arriving customer observes the same distribution of customers at a station as the stationary (random observer's) distribution for the same network with N-1 customers [17, 23]. This leads to the *Marginal Local Balance Theorem*, which states that for an M/M/1 queue

$$\mu_i P_i(n \mid N) = \lambda_i(N) P_i(n - 1 \mid N - 1) \tag{1}$$

where $P_i(n \mid N)$ is the conditional probability that n customers are at station i when N customers are in the network. The mean queue length at station i can be written as

$$Q_i(N) = \sum_{n=1}^{N} n\, P_i(n \mid N) = \sum_{n=1}^{N} \frac{n\lambda_i(N)}{\mu_i} P_i(n - 1 \mid N - 1) \tag{2}$$

The throughput rate $\lambda_i(N)$ is unknown, but application of Little's law provides

$$R_i(N) = s_i \sum_{n=1}^{N} n P_i(n - 1 \mid N - 1) = s_i[1 + Q_i(N-1)] \tag{3}$$

For a queue with an infinite number of servers there is no waiting, so $R_i(N) = s_i$ for all N.

Equation 3 establishes a recursive relationship between the response time of a station with N customers in the network and the distribution of customers at the station with N-1 customers in the network. Performance measures can be obtained by starting with $N = 1$ where $P_i(0 \mid N - 1) = 1$ and $Q_i(N - 1) = 0$ for all i, then incrementing N by 1 until the desired N is reached. Station throughput rates for each iteration are then

$$\lambda_i(N) = \frac{v_i}{v_1} \frac{N}{CT_1(N)} \tag{4}$$

where the cycle time at station 1 is given by

$$CT_1(N) = \sum_{i=1}^{M} \frac{v_i}{v_1} R_i(N) \tag{5}$$

and the visit ratios are calculated by solving the system of equations $\vec{v}\boldsymbol{P} = \vec{v}$. For the example system, the routing probability matrix is

$$\boldsymbol{P} = \begin{bmatrix} 0 & 0.95 & 0.05 & 0 & 0 & 0 \\ 0 & 0 & 0.30 & 0 & 0.70 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

By applying Little's law, queue length and utilization for each station are then

$$Q_i(N) = R_i(N)\lambda_i(N) \tag{7}$$

$$U_i(N) = s_i\lambda_i(N) \tag{8}$$

3.3.3 Modified MVA for Fork-Join Nodes

Since the fork-join structure for repair activities in the example system destroys the product-form nature of the network, MVA cannot be used directly and an approximation is necessary. Rao and Suri developed a heuristic based on MVA to analyze a single fork-join system consisting of single server queues for a closed fabrication/assembly system in [24]. Dietz and Jenkins extended the heuristic concepts to accommodate multiple fork-join nodes in a larger network, multiserver activities, and probabilistic service requirements [9, 14]. To apply the MVA algorithm to a network with fork-join nodes, two key approximations are needed [9]:

> *Approximation 1.* "For a network with N customers, a clone arriving at a substation sees the stationary (random observer's) distribution of clones at the substation for the same network with N - 1 customers."
>
> *Approximation 2.* "The response time experienced by a clone at a substation can be represented as an exponentially distributed random variable and is independent of the response time for clones at the other substations."

Based on approximation 2, the response time for any substation *i.k* is treated as an exponential random variable $T_{ik}(N)$ with rate parameter $\theta_{ik}(N) = 1/R_{ik}(N)$. For a fork-join queue i with K_i stations, there are 2^{Ki} possible combinations of service possibilities [9]. Let Ω_i be the union of all possible subsets for fork-join node i. The probability a customer arriving at node i requires set S is then

$$\pi_i(S) = \prod_{k \in S} q_{ik} \prod_{k \notin S} (1 - q_{ik}) \tag{9}$$

For each possible subset S, the expected maximum response time can be computed by

$$E\left[\max_{k\in S}\{T_{ik}(N)\}\right]=\sum_{k\in S}\frac{1}{\theta_{ik}(N)}-\sum_{k\in S}\sum_{\substack{l\in S\\ l<k}}\frac{1}{\theta_{ik}(N)+\theta_{il}(N)}$$

$$+\sum_{k\in S}\sum_{\substack{l\in S\\ l<k}}\sum_{\substack{m\in S\\ m<l,k}}\frac{1}{\theta_{ik}(N)+\theta_{il}(N)+\theta_{im}(N)}$$

$$-\cdots+(-1)^{K(S)+1}\frac{1}{\sum_{k\in S}\theta_{ik}(N)} \tag{10}$$

The MVA algorithm can now be modified to accommodate a network consisting of I simple service stations and J fork-join nodes. By approximation 1, response times at single server fork-join substations are

$$R_{ik}(N) = s_{ik}[1 + Q_{ik}(N\text{-}1)] \tag{11}$$

The substation response rates $\theta_{ik}(N) = 1/R_{ik}(N)$ are then input back into equation 10 to determine the conditional holding times for fork-join nodes which, in turn, is added to the cycle time calculation (equation 5) as follows:

$$CT_1(N)=\sum_{i\in I}\frac{v_i}{v_1}R_i(N)+\sum_{i\in J}\frac{v_i}{v_1}\sum_{S\subseteq\Omega_i}\pi_i(S)E\left[\max_{k\in S}\{T_{ik}(N)\}\right] \tag{12}$$

Throughput at a fork-join substation $i.k$ is given by

$$\lambda_i(N)=\frac{v_i}{v_1}\frac{q_{ik}N}{CT_1(N)} \tag{13}$$

Substation queue length and utilization are then obtained using Little's law as

$$Q_{ik}(N) = R_{ik}(N)\lambda_{ik}(N) \tag{14}$$

$$U_{ik}(N) = s_{ik}\lambda_{ik}(N) \tag{15}$$

If service times are exponentially distributed, performance measures calculated by the modified MVA heuristic give accurate approximate results [9]. However, since the service times given in the example network are not exponential and the heuristic does not accurately model the variability of the service times, the performance measure estimates will contain some error.

3.3.4 QNA Approximations for GI/G/1 Queues

The algorithms used in QNA provide one method to estimate performance measures for queues with general arrival and service processes based on the corresponding exact measures from exponential queues [27]. The expected waiting time for an M/M/1 queue is given by

$$W_i(\infty)_e = s_i \frac{\rho_i}{(1-\rho_i)} \tag{16}$$

where it is assumed that $0 \leq \rho < 1$ [15:191]. QNA uses an approximation for the mean waiting time of a GI/G/1 queue using the first two moments of the interarrival and service time distributions [27:2802]:

$$W_i(\infty) = g \frac{(c_{ai}^2 + c_{si}^2)}{2} s_i \frac{\rho_i}{(1-\rho_i)} = g \frac{(c_{ai}^2 + c_{si}^2)}{2} W_i(\infty)_e \tag{17}$$

where $g \equiv g(\rho_i, c_{ai}^2, c_{si}^2)$ is defined as

$$g(\rho_i, c_{ai}^2, c_{si}^2) = \begin{cases} \exp\left[-\dfrac{2(1-\rho_i)}{3\rho_i} \dfrac{\left(1-c_{ai}^2\right)^2}{\left(c_{ai}^2 + c_{si}^2\right)} \right], & c_{ai}^2 < 1 \\ 1, & c_{ai}^2 \geq 1 \end{cases} \tag{18}$$

This approximation works for open queues with infinite capacity and infinite calling population. However, the example network is closed with a finite calling population. Therefore, a conversion function that accounts for the transition from an infinite calling population in an open network to a finite calling population in a closed network is needed. Additionally, the first two moments of the arrival and service processes are required.

3.3.5 Open Network to Closed Network Conversion Function

Desruelle and Steudel developed the following approximation to estimate a Conversion function that accounts for the transition from an infinite calling population in an open network to a finite calling population in a closed network. Their method first computes a conversion function for mean waiting time in the equivalent product-form network, then applies the same conversion to the general service network [8:588-589]. From Little's law,

$$Q_i(N) = (W_i(N)_e + s_i)\lambda_i(N) \tag{19}$$

Using the arrival theorem and the memoryless property of the exponential distribution,

$$W_i(N)_e = s_i Q_i(N\text{-}1) \tag{20}$$

The Schweitzer-Bard heuristic, $\{Q(N\text{-}1)/(N\text{-}1)\}$ / $\{Q(N)/N\} = 1$ [26] is exact for closed symmetric exponential networks, but does not perform well for nonsymmetric networks. However, the ratio can still be used to provide a relation between $Q(N\text{-}1)$ and $Q(N)$. By applying the MVA algorithm, the ratio will yield a value, not necessarily equal to 1, for any N:

$$\frac{Q_i(N-1)/(N-1)}{Q_i(N)/N} = k \tag{21}$$

where k is a function of N, s_i, and $\lambda_i(N)$ in the product-form network. By combining equations 19-21 and since station utilization $\rho_i = s_i\lambda_i(N)$, mean waiting time at an exponential queue in a closed network can be expressed as

$$W_i(N)_e = \frac{\frac{Q_i(N-1)}{Q_i(N)}\rho_i s_i}{1-\rho_i \frac{Q_i(N-1)}{Q_i(N)}} \tag{22}$$

Using equations 16 and 22, the open network to closed network conversion function is

$$f(W_i(\infty)_e) = W_i(N)_e \,/\, W_i(\infty)_e = \frac{\frac{Q_i(N-1)}{Q_i(N)}}{\left(1+\left(\frac{W_i(\infty)_e}{s_i}\right)\left(1-\frac{Q_i(N-1)}{Q_i(N)}\right)\right)} \tag{23}$$

To extend this conversion factor to the nonproduct-form network, replace $W_i(\infty)_e$ in equation 23 with $W_i(\infty)$ from equation 17 and solve for $W_i(N)$:

$$W_i(N) = f(W_i(\infty))\, W_i(\infty) = \frac{\frac{Q_i(N-1)}{Q_i(N)}}{\left(1+\left(\frac{W_i(\infty)}{s_i}\right)\left(1-\frac{Q_i(N-1)}{Q_i(N)}\right)\right)} W_i(\infty) \tag{24}$$

With this approximation, the mean waiting time for a GI/G/1 queue in a closed system can be estimated as a function of the mean waiting time in the equivalent product-form network and the first two moments of the interarrival and service time distributions.

3.3.6 Traffic Variability Equations

For a closed system, the moments of the arrival process are a function of server utilization, which ultimately depends on the waiting times at the network's stations. For a given set of waiting times, the moments of the arrival processes can be determined through an iterative method using the queuing network analysis traffic variability equations derived in [16, 27]. For a GI/G/1 queue, the approximation for the SCV of the departure process is [16:116]:

$$c_{di}^2 = \rho_i^2 c_{si}^2 + (1-\rho_i^2)c_{ai}^2 \tag{25}$$

where $\rho_i = \lambda_i(N)s_i$ is the long-run portion of time the server is busy. This equation is used for stations 5, 6, and all of the substations in the fork-join node of the example network. For GI/G/∞ queues, the approximation for the SCV of the departure process is [8:582]:

$$c_{di}^2 = \rho_i^2 + (1-\rho_i^2)c_{ai}^2 \tag{26}$$

where $\rho_i = \lambda_i(N)s_i/N$ is the long-run portion of time a customer spends at delay station i. This equation is used for stations 1, 2, and 3 in the example network. When flows departing a station split, the approximation for the SCV of the flow on each branch is [27:2798]:

$$c_{ij}^2 = p_{ij}c_{di}^2 + (1-p_{ij}) \tag{27}$$

In the example network, this occurs following stations 1 and 2 and at the fork-join node. When flows merge at the next station, the approximation for the SCV of the arrival process is [8:582]:

$$c_{aj}^2 = \sum_i \frac{v_i}{v_j} p_{ij}\left(c_{ij}^2\right) \tag{28}$$

In the example network, this occurs when flows merge at stations 3 and 5.

3.3.7 Traffic Variability Equations for a Fork-Join Node

The decomposition procedure must be modified for networks containing fork-join queues because the flow is not only split at the fork node, but additional congestion is created by the clones generated at that point. Because the MVA-based approach for analyzing fork-join queues with probabilistic branching is relatively new [9, 14], the literature does not provide any algorithm which may be used directly. The objective of this research is to develop a method to complete the analysis.

At the fork (splitting) node, p_{ij} in equation 27 must be replaced by q_{ik}, since this is the true portion of customers that visit station *ik*. The queue service equations (25, 26) remain unchanged since the flow into each queue is correct as adjusted in equation 27. The difficult part is combining the flows at the join (merging) node. Simply substituting q_{ik} for p_{ij} in equation 28 will not work because the extra traffic caused by the clones created at the fork node must be eliminated before the recombined flows depart the join node. What is needed is an accurate approximation for the departure process from the fork-join node which attributes the appropriate portion of the total flow to each substation in the fork-join node.

One possible technique is to normalize the total flow out of the fork channels and attribute the portion of c_{di}^2 to queue *ik* based on the ratio $q_{ik} / (q_{i0} + \Sigma_k q_{ik})$. While this technique estimates the *average* response time for a given set of service requirements, it fails to capture the effect of clones waiting for siblings to complete service.

The approximation sought should reflect the probability that the last clone to finish service comes from node *ik*. To meet this goal, an approximation is needed to estimate the probability that the *maximum* response time for a customer requiring S services comes from node ik, $k \in S$. A relatively simple approximation for this is described by Jenkins in [14:3-8 - 3-9]. By assuming response times are exponentially distributed and following a similar procedure to the one used to determine $E[\max_{k \in S}\{T_{ik}(N)\}]$ in equation 10, the conditional probabilities that the maximum response time occurs at substation ik can be estimated. For each possible subset S requiring two or more types of service, the probability that the maximum response time for a customer requiring S services comes from substation ik, $k \in S$, when N customers are in the system is estimated by

$$p_{ik}(N,S) = \sum_{\substack{l \neq k \\ l,k \in S}} \frac{\theta_{il}(N)}{\theta_{ik}(N) + \theta_{il}(N)} - \sum_{\substack{l \neq k \\ l \in S \\ k \in S}} \sum_{\substack{m \neq k \\ m < l \\ m \in S}} \frac{\theta_{il}(N) + \theta_{im}(N)}{\theta_{ik}(N) + \theta_{il}(N) + \theta_{im}(N)}$$

$$+ \sum_{\substack{l \neq k \\ l \in S \\ k \in S}} \sum_{\substack{m \neq k \\ m < l \\ m \in S}} \sum_{\substack{n \neq k \\ n < m \\ n \in S}} \frac{\theta_{il}(N) + \theta_{im}(N) + \theta_{in}(N)}{\theta_{ik}(N) + \theta_{il}(N) + \theta_{im}(N) + \theta_{in}(N)}$$

$$- \cdots + (-1)^{K(S)} \frac{\sum_{\substack{l \neq k \\ l,k \in S}} \theta_{il}(N)}{\sum_{l \in S} \theta_{il}(N)} \tag{29}$$

Obviously, $p_{ik}(N,S) = 0$ if $k \notin S$ and $p_{ik}(N,S) = 1$ if $k = S$. The marginal probability that the response time at substation ik is longer than that for any other substation is then

$$p_{ik}(N) = \sum_{S} \pi_i\{S\} p_{ik}(N,S) \tag{30}$$

Finally, by replacing p_{ij} in equation 28 with $p_{ik}(N)$ from equation 30 and letting $q_{i0} = \Pi_k(1\text{-}q_{ik})$ represent the probability that a customer requires no service (due to a false malfunction indication), the new approximation for the SCV of the departure process from the fork-join node becomes

$$c_{di}^2 = q_{i0}(q_{i0}c_{ai}^2 + 1 - q_{i0}) + \sum_k p_{ik}(N)c_{dik}^2 \tag{31}$$

With this approximation, it is now possible to analytically model concurrent maintenance and at the same time capture the effects of general service time distributions.

3.3.8 Solving the Example Network

Table 3.3. Example Network Traffic Variability Equations

$c_{a1}^2 = c_{d6}^2$	$c_{d1}^2 = \rho_1^2 + (1-\rho_1^2)c_{a1}^2$
$c_{a2}^2 = 0.95c_{d1}^2 + 0.05$	$c_{d2}^2 = \rho_2^2 + (1-\rho_2^2)c_{a2}^2$
$c_{a3}^2 = \frac{0.05}{0.335}\left(0.05c_{d1}^2 + 0.95\right) + \frac{0.285}{0.335}\left(0.30c_{d2}^2 + 0.70\right)$	$c_{d3}^2 = \rho_3^2 + (1-\rho_3^2)c_{a3}^2$
$c_{a4}^2 = c_{d3}^2$	
$c_{a41}^2 = 0.17c_{d3}^2 + 0.83$	$c_{d41}^2 = 0.29\rho_{41}^2 + (1-\rho_{41}^2)c_{a41}^2$
$c_{a42}^2 = 0.39c_{d3}^2 + 0.61$	$c_{d42}^2 = 0.29\rho_{42}^2 + (1-\rho_{42}^2)c_{a42}^2$
$c_{a43}^2 = 0.21c_{d3}^2 + 0.79$	$c_{d43}^2 = 0.29\rho_{43}^2 + (1-\rho_{43}^2)c_{a43}^2$
$c_{a44}^2 = 0.27c_{d3}^2 + 0.73$	$c_{d44}^2 = 0.29\rho_{44}^2 + (1-\rho_{44}^2)c_{a44}^2$
$c_{a45}^2 = 0.46c_{d3}^2 + 0.54$	$c_{d45}^2 = 0.29\rho_{45}^2 + (1-\rho_{45}^2)c_{a45}^2$
$c_{d4}^2 = p_{41}(N)c_{d41}^2 + p_{42}(N)c_{d42}^2 + p_{43}(N)c_{d43}^2 + p_{44}(N)c_{d44}^2 + p_{45}(N)c_{d45}^2 + q_{40}\left[q_{40}c_{a4}^2 + (1-q_{40})\right]$	
$c_{a5}^2 = 0.665\left(0.70c_{d2}^2 + 0.30\right) + 0.335c_{d4}^2$	$c_{d5}^2 = 0.29\rho_5^2 + (1-\rho_5^2)c_{a5}^2$
$c_{a6}^2 = c_{d5}^2$	$c_{d6}^2 = 0.29\rho_6^2 + (1-\rho_6^2)c_{a6}^2$

Table 3.3 shows the traffic variability equations used for the example network. By starting with an arbitrary value, say $c_{a1}^2 = 1$, and solving the rest of the network based on that value, a new value is computed ($c_{a1}^2 = c_{d6}^2$). The process is repeated using the new c_{a1}^2 and iterating to convergence. The waiting time calculations (equations 17 and

24) can then be updated with revised arrival process SCVs, in turn altering the response times (equations 3 and 11) and ultimately the utilization factors (equations 8 and 15), which were originally used in the traffic variability equations (25-28). Thus, the iterative process is repeated, alternately updating utilization factors and each c_{ai}^2 until reaching the desired tolerance. This methodology can now be applied to estimate the performance of a network of general service queues and a fork-join queue with probabilistic branching.

3.3.9 Computer Implementation

The heuristic is coded in a program called "GenQue," programmed in Borland's Turbo Pascal, version 6.0. This code includes a program called "ForQue," written by Dietz [9], which is used to estimate performance measures for the equivalent exponential server network. GenQue calculates and displays mean values for waiting time, response time, throughput, queue length, and server utilization at each station in the network. The source code and a sample input file are included in Appendix A. Appendix B shows sample program output.

IV. Findings and Analysis

4.1 Overview

This chapter details the results of the general queueing system heuristic when applied to the example network shown in figure 1.1. Due to the methodology's current inability to explicitly model multiserver queues (no open network to closed network conversion function for multiserver queues), two different approaches are used in an attempt to place bounds on actual system performance. Performance measure estimates for simulation, the exponential server model (ForQue), and the general server model (GenQue) are provided in a series of tables.

Simulation data is based on batch means analysis of 10^7 hours of simulated operating time with a 10^6 hour warmup. The range for each result is based on a 95 percent confidence interval. Model performance is evaluated in terms of absolute error (the difference between model and simulation results) and relative error (100 percent multiplied by the ratio between absolute error and the simulation result). The simulation used is a modified version of a program originally written by Dietz [9] for Pritsker's SLAM II, version 4.1. The simulation code is documented in Appendix C.

4.2 Single Server Model

4.2.1 Description

The first data set, the "Single Server Model," assumes only one server is available at each station, completely ignoring multiple servers. Because multiple servers perform at least as well as a single server of the same type, this set provides a lower bound on actual system performance. Table 4.1 shows the parameters used for this data set.

Table 4.1. Single Server Model Parameters

Activity	Service Time Mean	Service Time SCV	Repair Probability	Resource Level
Taxi	$s_1 = 0.25$	$scv_1 = 0.00$		$m_1 = \infty$
Sortie	$s_2 = 2.00$	$scv_2 = 0.02$		$m_2 = \infty$
Troubleshoot	$s_3 = 0.50$	$scv_3 = 0.29$		$m_3 = \infty$
Rpr1 (airframe)	$s_{41} = 2.20$	$scv_{41} = 0.29$	$q_{41} = 0.17$	$m_{41} = 1$
Rpr2 (electrical/hydraulic)	$s_{42} = 2.27$	$scv_{42} = 0.29$	$q_{42} = 0.39$	$m_{42} = 1$
Rpr3 (engine)	$s_{43} = 2.37$	$scv_{43} = 0.29$	$q_{43} = 0.21$	$m_{43} = 1$
Rpr4 (avionics)	$s_{44} = 1.50$	$scv_{44} = 0.29$	$q_{44} = 0.27$	$m_{44} = 1$
Rpr5 (radar/weapons control)	$s_{45} = 1.19$	$scv_{45} = 0.29$	$q_{45} = 0.46$	$m_{45} = 1$
Turn-around	$s_5 = 0.75$	$scv_5 = 0.29$		$m_5 = 1$
Munitions	$s_6 = 0.50$	$scv_6 = 0.29$		$m_6 = 1$

4.2.2 Results

The GenQue algorithm terminates when computed server utilization exceeds 1.0, implying that a bottleneck develops. For the single server system, the heuristic's results indicate that this occurs at station 5 when 8 or more aircraft are in the system. Tables 4.2 through 4.6 compare results for wait time, response time, throughput rate, queue length, and station utilization for simulation results and exponential and general service models.

The main difference between the exponential and general models is the method used to estimate wait time. Since the other performance measures are a function of wait time, any differences in estimated wait time are carried forward to the other performance measures. Using only the first moment and assuming exponential service times, the exponential model typically overestimates wait time for this system by 40 to 70 percent. By taking into account the second moment of the service and arrival processes, the general model usually provides significantly more accurate results. When server utilization within the network remains at realistic levels (up to 65 percent with 4 aircraft in the system), the maximum error observed is 0.024 hours (1.4 minutes) for the general model and 0.179 hours (10.7 minutes) for the exponential model. Even at higher utilization levels (up to 86 percent with 6 aircraft in the system), the maximum error observed is 0.112 hours (6.7 minutes) for the general model and 0.243 hours (14.6 minutes) for the exponential model.

In general, as the number of aircraft in the system and server utilization increase, accuracy in wait time estimation decreases for the general model while it increases in the exponential model. For the exponential model, higher utilization levels mean an arriving customer's wait time will be based more on the mean service times of the customers already in the queue and less on the arrival process. This is demonstrated in the N = 8 aircraft case where station 5 utilization is almost 97 percent and the wait time estimate is only 5.5 percent in error while all of the other stations have errors in excess of 40 percent. As utilization increases, the approximations used for the general model tend to increasingly underestimate actual waiting time. Similar results were also reported for a system consisting of a single GI/G/1 queue and a delay station [8: 584-585].

The general model's improvement in estimated waiting time is then carried forward to the other performance measures (response time, throughput rate, queue length, and utilization), reflecting a similar performance advantage for the general model over the exponential model. In most cases, the relative error for the general model is less than three percent for all of these performance measures.

Table 4.2. Single Server Model: Wait Time Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model			General Model		
				Result	Abs Error	% Error	Result	Abs Error	% Error
N=2									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	0.041 ± 0.002	0.065	0.023	56.64	0.042	0.001	-
4.2	1	2.270	0.100 ± 0.004	0.158	0.058	58.25	0.103	0.004	-
4.3	1	2.370	0.057 ± 0.004	0.093	0.036	63.64	0.060	0.004	-
4.4	1	1.500	0.029 ± 0.002	0.048	0.019	65.74	0.031	0.002	-
4.5	1	1.190	0.031 ± 0.001	0.051	0.020	66.61	0.033	0.002	7.71
5	1	0.750	0.078 ± 0.001	0.132	0.054	70.08	0.088	0.010	13.22
6	1	0.500	0.017 ± 0.000	0.059	0.042	247.16	0.034	0.018	103.79
N=4									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	0.125 ± 0.007	0.184	0.059	47.02	0.118	-0.007	-
4.2	1	2.270	0.330 ± 0.004	0.485	0.155	46.83	0.312	-0.019	-5.67
4.3	1	2.370	0.181 ± 0.006	0.269	0.088	48.43	0.173	-0.008	-4.69
4.4	1	1.500	0.089 ± 0.003	0.137	0.048	53.39	0.087	-0.002	-
4.5	1	1.190	0.097 ± 0.001	0.150	0.053	54.24	0.094	-0.003	-3.02
5	1	0.750	0.316 ± 0.002	0.495	0.179	56.52	0.318	0.002	-
6	1	0.500	0.060 ± 0.000	0.197	0.138	230.43	0.083	0.024	39.75
N=6									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	0.193 ± 0.007	0.282	0.089	46.02	0.180	-0.014	-7.09
4.2	1	2.270	0.563 ± 0.005	0.799	0.236	42.00	0.504	-0.058	-10.39
4.3	1	2.370	0.290 ± 0.011	0.419	0.128	44.24	0.266	-0.024	-8.35
4.4	1	1.500	0.138 ± 0.004	0.210	0.072	52.02	0.132	-0.006	-4.49
4.5	1	1.190	0.154 ± 0.002	0.235	0.081	52.52	0.144	-0.010	-6.19
5	1	0.750	0.789 ± 0.001	1.032	0.242	30.68	0.677	-0.112	-14.23
6	1	0.500	0.115 ± 0.000	0.358	0.243	211.14	0.108	-0.007	-5.98
N=8									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	Algorithm Terminates Station 5 Utilization Approaches 1.0		
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-			
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-			
4.1	1	2.200	0.234 ± 0.007	0.352	0.118	50.64			
4.2	1	2.270	0.737 ± 0.014	1.062	0.325	44.18			
4.3	1	2.370	0.357 ± 0.009	0.530	0.173	48.55			
4.4	1	1.500	0.167 ± 0.004	0.263	0.096	57.37			
4.5	1	1.190	0.188 ± 0.003	0.299	0.111	58.80			
5	1	0.750	1.689 ± 0.005	1.781	0.093	5.49			
6	1	0.500	0.167 ± 0.002	0.526	0.359	215.57			

Table 4.3. Single Server Model: Response Time Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model			General Model		
				Result	Abs Error	% Error	Result	Abs Error	% Error
N=2									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	2.240 ± 0.017	2.265	0.024	1.09	2.242	0.002	-
4.2	1	2.270	2.371 ± 0.009	2.428	0.057	2.41	2.373	0.003	-
4.3	1	2.370	2.432 ± 0.022	2.463	0.031	1.26	2.430	-0.002	-
4.4	1	1.500	1.521 ± 0.007	1.548	0.027	1.78	1.531	0.010	0.67
4.5	1	1.190	1.222 ± 0.004	1.241	0.019	1.58	1.223	0.001	-
5	1	0.750	0.828 ± 0.001	0.882	0.054	6.54	0.838	0.010	1.22
6	1	0.500	0.517 ± 0.001	0.559	0.042	8.13	0.534	0.018	3.45
N=4									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	2.325 ± 0.019	2.384	0.059	2.54	2.318	-0.007	-
4.2	1	2.270	2.603 ± 0.009	2.755	0.152	5.85	2.582	-0.021	-0.82
4.3	1	2.370	2.553 ± 0.012	2.639	0.086	3.38	2.543	-0.010	-
4.4	1	1.500	1.588 ± 0.007	1.637	0.048	3.05	1.587	-0.001	-
4.5	1	1.190	1.286 ± 0.003	1.340	0.054	4.20	1.284	-0.002	-
5	1	0.750	1.066 ± 0.002	1.245	0.179	16.79	1.068	0.002	-
6	1	0.500	0.560 ± 0.001	0.697	0.138	24.59	0.583	0.024	4.25
N=6									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	2.391 ± 0.015	2.482	0.091	3.81	2.380	-0.011	-
4.2	1	2.270	2.832 ± 0.009	3.069	0.238	8.39	2.774	-0.057	-2.02
4.3	1	2.370	2.663 ± 0.018	2.789	0.126	4.72	2.636	-0.027	-1.01
4.4	1	1.500	1.634 ± 0.006	1.710	0.077	4.69	1.632	-0.002	-
4.5	1	1.190	1.344 ± 0.002	1.425	0.081	6.00	1.334	-0.010	-0.71
5	1	0.750	1.539 ± 0.002	1.782	0.242	15.74	1.427	-0.112	-7.30
6	1	0.500	0.615 ± 0.001	0.858	0.244	39.62	0.608	-0.006	-1.06
N=8									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	Algorithm Terminates Station 5 Utilization Approaches 1.0		
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-			
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-			
4.1	1	2.200	2.434 ± 0.012	2.552	0.119	4.87			
4.2	1	2.270	3.007 ± 0.016	3.332	0.325	10.82			
4.3	1	2.370	2.734 ± 0.016	2.900	0.166	6.08			
4.4	1	1.500	1.668 ± 0.006	1.763	0.095	5.73			
4.5	1	1.190	1.377 ± 0.007	1.489	0.112	8.14			
5	1	0.750	2.438 ± 0.005	2.531	0.093	3.83			
6	1	0.500	0.667 ± 0.002	1.026	0.359	53.88			

Table 4.4. Single Server Model: Throughput Rate Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model Result	Exponential Model Abs Error	Exponential Model % Error	General Model Result	General Model Abs Error	General Model % Error
N=2									
1	Inf	0.250	0.464 ± 0.001	0.446	-0.019	-4.04	0.454	-0.011	-2.30
2	Inf	2.000	0.441 ± 0.001	0.423	-0.018	-4.04	0.431	-0.010	-2.30
3	Inf	0.500	0.156 ± 0.000	0.149	-0.006	-4.02	0.152	-0.004	-2.29
4.1	1	2.200	0.026 ± 0.000	0.025	-0.001	-4.04	0.026	-0.001	-2.53
4.2	1	2.270	0.061 ± 0.000	0.058	-0.002	-4.11	0.059	-0.001	-2.29
4.3	1	2.370	0.033 ± 0.000	0.031	-0.001	-4.26	0.032	-0.001	-2.42
4.4	1	1.500	0.042 ± 0.000	0.040	-0.002	-4.10	0.041	-0.001	-2.44
4.5	1	1.190	0.072 ± 0.000	0.069	-0.003	-3.99	0.070	-0.002	-2.32
5	1	0.750	0.464 ± 0.001	0.446	-0.019	-4.04	0.454	-0.011	-2.30
6	1	0.500	0.464 ± 0.001	0.446	-0.019	-4.04	0.454	-0.011	-2.30
N=4									
1	Inf	0.250	0.864 ± 0.001	0.791	-0.073	-8.42	0.846	-0.018	-2.08
2	Inf	2.000	0.821 ± 0.001	0.752	-0.069	-8.42	0.804	-0.017	-2.08
3	Inf	0.500	0.289 ± 0.000	0.265	-0.024	-8.42	0.283	-0.006	-2.09
4.1	1	2.200	0.049 ± 0.000	0.045	-0.004	-8.60	0.048	-0.001	-2.10
4.2	1	2.270	0.113 ± 0.000	0.103	-0.010	-8.50	0.111	-0.002	-2.12
4.3	1	2.370	0.061 ± 0.000	0.056	-0.005	-8.57	0.060	-0.001	-2.15
4.4	1	1.500	0.078 ± 0.000	0.072	-0.007	-8.53	0.077	-0.002	-2.14
4.5	1	1.190	0.133 ± 0.000	0.122	-0.011	-8.42	0.130	-0.003	-2.11
5	1	0.750	0.864 ± 0.001	0.791	-0.073	-8.42	0.846	-0.018	-2.08
6	1	0.500	0.864 ± 0.001	0.791	-0.073	-8.42	0.846	-0.018	-2.08
N=6									
1	Inf	0.250	1.153 ± 0.001	1.032	-0.121	-10.52	1.165	0.012	1.02
2	Inf	2.000	1.095 ± 0.001	0.980	-0.115	-10.53	1.107	0.011	1.02
3	Inf	0.500	0.386 ± 0.000	0.346	-0.041	-10.53	0.390	0.004	1.02
4.1	1	2.200	0.066 ± 0.000	0.059	-0.007	-10.53	0.066	0.001	0.88
4.2	1	2.270	0.151 ± 0.000	0.135	-0.016	-10.55	0.152	0.001	0.99
4.3	1	2.370	0.081 ± 0.000	0.073	-0.009	-10.56	0.082	0.001	0.89
4.4	1	1.500	0.104 ± 0.000	0.093	-0.011	-10.59	0.105	0.001	1.01
4.5	1	1.190	0.178 ± 0.000	0.159	-0.019	-10.52	0.180	0.002	1.02
5	1	0.750	1.153 ± 0.001	1.032	-0.121	-10.52	1.165	0.012	1.02
6	1	0.500	1.153 ± 0.001	1.032	-0.121	-10.52	1.165	0.012	1.02
N=8									
1	Inf	0.250	1.285 ± 0.016	1.180	-0.105	-8.20			
2	Inf	2.000	1.221 ± 0.016	1.121	-0.100	-8.19			
3	Inf	0.500	0.430 ± 0.006	0.395	-0.035	-8.19			
4.1	1	2.200	0.073 ± 0.001	0.067	-0.006	-8.25		Algorithm Terminates	
4.2	1	2.270	0.168 ± 0.002	0.154	-0.014	-8.24			
4.3	1	2.370	0.090 ± 0.001	0.083	-0.007	-8.24		Station 5 Utilization	
4.4	1	1.500	0.116 ± 0.001	0.107	-0.010	-8.24		Approaches 1.0	
4.5	1	1.190	0.198 ± 0.003	0.182	-0.016	-8.18			
5	1	0.750	1.285 ± 0.016	1.180	-0.105	-8.20			
6	1	0.500	1.285 ± 0.016	1.180	-0.105	-8.20			

Table 4.5. Single Server Model: Queue Length Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model Result	Exponential Model Abs Error	Exponential Model % Error	General Model Result	General Model Abs Error	General Model % Error
N=2									
1	Inf	0.250	0.116 ± 0.000	0.111	-0.005	-4.04	0.113	-0.003	-2.32
2	Inf	2.000	0.882 ± 0.001	0.847	-0.036	-4.06	0.862	-0.020	-2.32
3	Inf	0.500	0.078 ± 0.000	0.075	-0.003	-4.33	0.076	-0.002	-2.54
4.1	1	2.200	0.059 ± 0.001	0.058	-0.002	-3.00	0.058	-0.001	-
4.2	1	2.270	0.144 ± 0.001	0.141	-0.003	-1.84	0.141	-0.003	-2.26
4.3	1	2.370	0.080 ± 0.001	0.077	-0.002	-3.00	0.078	-0.002	-2.50
4.4	1	1.500	0.064 ± 0.001	0.062	-0.001	-1.95	0.063	-0.001	-
4.5	1	1.190	0.088 ± 0.001	0.085	-0.002	-2.68	0.086	-0.002	-2.34
5	1	0.750	0.384 ± 0.000	0.393	0.009	2.24	0.380	-0.004	-1.12
6	1	0.500	0.240 ± 0.000	0.249	0.009	3.76	0.243	0.003	1.09
N=4									
1	Inf	0.250	0.216 ± 0.000	0.198	-0.018	-8.40	0.211	-0.005	-2.11
2	Inf	2.000	1.641 ± 0.001	1.503	-0.138	-8.40	1.607	-0.034	-2.06
3	Inf	0.500	0.144 ± 0.000	0.133	-0.012	-8.26	0.142	-0.003	-1.89
4.1	1	2.200	0.114 ± 0.001	0.107	-0.007	-5.73	0.112	-0.002	-1.96
4.2	1	2.270	0.294 ± 0.002	0.285	-0.009	-3.15	0.285	-0.009	-2.98
4.3	1	2.370	0.154 ± 0.001	0.147	-0.008	-4.93	0.151	-0.003	-2.01
4.4	1	1.500	0.124 ± 0.001	0.117	-0.007	-5.37	0.121	-0.002	-1.90
4.5	1	1.190	0.171 ± 0.001	0.163	-0.007	-4.26	0.167	-0.003	-1.86
5	1	0.750	0.921 ± 0.001	0.985	0.064	6.95	0.903	-0.018	-1.93
6	1	0.500	0.483 ± 0.001	0.552	0.068	14.10	0.494	0.010	2.10
N=6									
1	Inf	0.250	0.288 ± 0.000	0.258	-0.030	-10.56	0.291	0.003	0.98
2	Inf	2.000	2.190 ± 0.002	1.960	-0.230	-10.49	2.213	0.023	1.06
3	Inf	0.500	0.193 ± 0.000	0.173	-0.021	-10.64	0.195	0.002	0.89
4.1	1	2.200	0.157 ± 0.001	0.146	-0.011	-7.02	0.158	0.001	-
4.2	1	2.270	0.427 ± 0.002	0.414	-0.014	-3.16	0.422	-0.005	-1.17
4.3	1	2.370	0.217 ± 0.003	0.202	-0.015	-6.75	0.216	-0.001	-
4.4	1	1.500	0.171 ± 0.001	0.160	-0.011	-6.49	0.172	0.001	-
4.5	1	1.190	0.238 ± 0.001	0.227	-0.012	-5.00	0.240	0.001	0.45
5	1	0.750	1.775 ± 0.002	1.838	0.063	3.57	1.662	-0.112	-6.34
6	1	0.500	0.709 ± 0.001	0.886	0.177	24.94	0.709	0.000	-
N=8									
1	Inf	0.250	0.323 ± 0.000	0.295	-0.029	-8.82			
2	Inf	2.000	2.457 ± 0.002	2.241	-0.216	-8.78			
3	Inf	0.500	0.217 ± 0.001	0.198	-0.019	-8.96			
4.1	1	2.200	0.179 ± 0.001	0.172	-0.007	-3.94	Algorithm Terminates		
4.2	1	2.270	0.508 ± 0.004	0.514	0.006	1.18			
4.3	1	2.370	0.249 ± 0.002	0.241	-0.008	-3.14	Station 5 Utilization		
4.4	1	1.500	0.196 ± 0.001	0.188	-0.008	-3.96	Approaches 1.0		
4.5	1	1.190	0.275 ± 0.001	0.271	-0.004	-1.43			
5	1	0.750	3.154 ± 0.005	2.986	-0.168	-5.32			
6	1	0.500	0.863 ± 0.003	1.210	0.348	40.32			

Table 4.6. Single Server Model: Station Utilization Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model Result	Exponential Model Abs Error	Exponential Model % Error	General Model Result	General Model Abs Error	General Model % Error
N=2									
1	Inf	0.250	0.116 ± 0.000	0.111	-0.005	-4.04	0.113	-0.003	-2.32
2	Inf	2.000	0.882 ± 0.001	0.847	-0.036	-4.06	0.862	-0.020	-2.32
3	Inf	0.500	0.078 ± 0.000	0.075	-0.003	-4.33	0.076	-0.002	-2.54
4.1	1	2.200	0.058 ± 0.001	0.056	-0.002	-4.09	0.057	-0.001	-2.37
4.2	1	2.270	0.138 ± 0.001	0.132	-0.006	-4.13	0.135	-0.003	-2.39
4.3	1	2.370	0.078 ± 0.001	0.074	-0.003	-4.43	0.076	-0.002	-2.75
4.4	1	1.500	0.062 ± 0.001	0.061	-0.002	-3.06	0.062	-0.001	-1.30
4.5	1	1.190	0.085 ± 0.000	0.082	-0.004	-4.22	0.083	-0.002	-2.46
5	1	0.750	0.348 ± 0.000	0.334	-0.014	-4.06	0.340	-0.008	-2.31
6	1	0.500	0.232 ± 0.000	0.223	-0.009	-3.98	0.227	-0.005	-2.26
N=4									
1	Inf	0.250	0.216 ± 0.000	0.198	-0.018	-8.40	0.211	-0.005	-2.11
2	Inf	2.000	1.641 ± 0.001	1.503	-0.138	-8.40	1.607	-0.034	-2.06
3	Inf	0.500	0.144 ± 0.000	0.133	-0.012	-8.26	0.142	-0.003	-1.89
4.1	1	2.200	0.108 ± 0.001	0.099	-0.009	-8.06	0.106	-0.002	-1.66
4.2	1	2.270	0.257 ± 0.001	0.235	-0.022	-8.62	0.251	-0.006	-2.31
4.3	1	2.370	0.143 ± 0.001	0.132	-0.012	-8.06	0.141	-0.002	-1.71
4.4	1	1.500	0.117 ± 0.001	0.107	-0.010	-8.15	0.115	-0.002	-1.73
4.5	1	1.190	0.158 ± 0.001	0.145	-0.013	-7.99	0.155	-0.003	-1.65
5	1	0.750	0.648 ± 0.001	0.593	-0.054	-8.39	0.634	-0.013	-2.06
6	1	0.500	0.432 ± 0.001	0.396	-0.036	-8.40	0.423	-0.009	-2.06
N=6									
1	Inf	0.250	0.288 ± 0.000	0.258	-0.030	-10.56	0.291	0.003	0.98
2	Inf	2.000	2.190 ± 0.002	1.960	-0.230	-10.49	2.213	0.023	1.06
3	Inf	0.500	0.193 ± 0.000	0.173	-0.021	-10.64	0.195	0.002	0.89
4.1	1	2.200	0.144 ± 0.001	0.129	-0.015	-10.30	0.146	0.002	1.22
4.2	1	2.270	0.342 ± 0.001	0.306	-0.036	-10.60	0.346	0.003	0.94
4.3	1	2.370	0.193 ± 0.002	0.172	-0.021	-11.05	0.194	0.001	0.43
4.4	1	1.500	0.156 ± 0.001	0.140	-0.016	-10.38	0.158	0.002	1.14
4.5	1	1.190	0.211 ± 0.001	0.189	-0.022	-10.39	0.214	0.002	1.17
5	1	0.750	0.864 ± 0.000	0.774	-0.091	-10.49	0.874	0.009	1.06
6	1	0.500	0.576 ± 0.001	0.516	-0.060	-10.45	0.582	0.006	1.11
N=8									
1	Inf	0.250	0.323 ± 0.000	0.295	-0.029	-8.82	Algorithm Terminates Station 5 Utilization Approaches 1.0		
2	Inf	2.000	2.457 ± 0.002	2.241	-0.216	-8.78			
3	Inf	0.500	0.217 ± 0.001	0.198	-0.019	-8.96			
4.1	1	2.200	0.161 ± 0.001	0.148	-0.014	-8.41			
4.2	1	2.270	0.383 ± 0.002	0.350	-0.033	-8.69			
4.3	1	2.370	0.216 ± 0.001	0.197	-0.019	-8.96			
4.4	1	1.500	0.176 ± 0.001	0.160	-0.016	-9.19			
4.5	1	1.190	0.237 ± 0.001	0.216	-0.021	-8.74			
5	1	0.750	0.969 ± 0.000	0.885	-0.085	-8.74			
6	1	0.500	0.647 ± 0.001	0.590	-0.057	-8.83			

4.3 Aggregated Server Model

4.3.1 Description

The second data set accounts for multiple servers by aggregating all of the servers at a single station and then increasing the service rate by a factor equal to the original number of servers, reducing the mean service time by the same factor. Since the single "super server" performs equal to or better than multiple servers, this set, the "Aggregated Server Model," provides an upper bound on actual system performance. Table 4.7 shows the parameters used for this data set.

Table 4.7. Aggregated Server Model Parameters

Activity	Service Time Mean	Service Time SCV	Repair Probability	Resource Level
Taxi	$s_1 = 0.250$	$scv_1 = 0.00$		$m_1 = \infty$
Sortie	$s_2 = 2.000$	$scv_2 = 0.02$		$m_2 = \infty$
Troubleshoot	$s_3 = 0.500$	$scv_3 = 0.29$		$m_3 = \infty$
Rpr1 (airframe)	$s_{41} = 2.200$	$scv_{41} = 0.29$	$q_{41} = 0.17$	$m_{41} = 1$
Rpr2 (electrical/hydraulic)	$s_{42} = 0.757$	$scv_{42} = 0.29$	$q_{42} = 0.39$	$m_{42} = 1$
Rpr3 (engine)	$s_{43} = 1.185$	$scv_{43} = 0.29$	$q_{43} = 0.21$	$m_{43} = 1$
Rpr4 (avionics)	$s_{44} = 1.500$	$scv_{44} = 0.29$	$q_{44} = 0.27$	$m_{44} = 1$
Rpr5 (radar/weapons control)	$s_{45} = 0.595$	$scv_{45} = 0.29$	$q_{45} = 0.46$	$m_{45} = 1$
Turn-around	$s_5 = 0.125$	$scv_5 = 0.29$		$m_5 = 1$
Munitions	$s_6 = 0.125$	$scv_6 = 0.29$		$m_6 = 1$

4.3.2 Results

The heuristic's results indicate that a bottleneck develops at substation 4.4 (server utilization approaches 1.0) when 30 or more aircraft are in the system. Tables 4.8 through 4.12 compare results for wait time, response time, throughput rate, queue length, and station utilization for simulation results and exponential and general service models.

In this data set, as in the Single Server set, the exponential model typically overestimates wait time for the system, this time by only 30 to 50 percent. Except at the highest levels of server utilization, the general model still provides significantly more accurate results. When server utilization within the network remains at lower levels (up to 51 percent with 12 aircraft in the system), the maximum error observed is 0.066 hours (4.0 minutes) for the general model and 0.419 hours (25.1 minutes) for the exponential model. Even at higher utilization levels (up to 71 percent with 18 aircraft in the system), the maximum error observed is only 0.294 hours (17.6 minutes) for the general model and 0.630 hours (37.8 minutes) for the exponential model. The only time the exponential model outperforms the general model for this system is when the number of aircraft in the system and server utilization are high, as shown in the results for substations 4.1 and 4.4 with 24 aircraft in the system.

As in the Single Server model, as the number of aircraft in the system and server utilization increase, accuracy in wait time estimation decreases for the general model while it increases in the exponential model. Again, the exponential model is most accurate when server utilization is highest. This is demonstrated in the N = 24 aircraft case where substation 4.4 utilization is 84 percent and the wait time estimate is less than 10 percent in error while the general service model's error is 2.5 times larger. Once again, as utilization increases, the general model increasingly underestimates actual waiting time.

A possible explanation for this phenomenon is suggested in [8:585]. The problem may be caused in part by violation of the renewal assumption for the aircraft interarrival times to the service queues. When a long service time occurs at one station, many or all of the other aircraft may join the queue during that service time. Then, when the service ends, a series of departures with short interdeparture times can occur (if service times are short), resulting in added congestion and increased waiting times at the stations downstream in the network (much like the batch arrival problem). To capture these effects, the model must be modified to account for this autocorrelation between service times and interarrival times.

Except for the extreme cases, the general model's more accurate waiting time estimates result in better estimates for the other performance measures (response time, throughput rate, queue length, and utilization). In most cases when server utilization is less than 70 percent, the relative error for the general model is less than five percent for all of these performance measures.

Table 4.8. Aggregated Server Model: Wait Time Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model Result	Exponential Model Abs Error	Exponential Model % Error	General Model Result	General Model Abs Error	General Model % Error
N=6									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	0.358 ± 0.007	0.527	0.169	47.37	0.346	-0.011	-3.17
4.2	1	0.757	0.091 ± 0.002	0.138	0.046	50.76	0.090	-0.002	-1.83
4.3	1	1.185	0.117 ± 0.001	0.178	0.062	52.95	0.116	-0.001	-0.52
4.4	1	1.500	0.263 ± 0.005	0.395	0.132	49.97	0.260	-0.004	-1.35
4.5	1	0.595	0.065 ± 0.001	0.100	0.035	54.07	0.065	0.000	0.03
5	1	0.125	0.019 ± 0.000	0.030	0.011	57.62	0.019	0.000	0.16
6	1	0.125	0.012 ± 0.000	0.030	0.017	139.78	0.018	0.006	45.95
N=12									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	1.008 ± 0.011	1.427	0.419	41.56	0.942	-0.066	-6.59
4.2	1	0.757	0.239 ± 0.003	0.348	0.108	45.31	0.225	-0.015	-6.19
4.3	1	1.185	0.300 ± 0.005	0.432	0.133	44.34	0.280	-0.020	-6.51
4.4	1	1.500	0.790 ± 0.007	1.102	0.311	39.35	0.728	-0.062	-7.89
4.5	1	0.595	0.168 ± 0.001	0.247	0.078	46.42	0.158	-0.010	-6.03
5	1	0.125	0.053 ± 0.000	0.081	0.028	53.38	0.048	-0.005	-9.83
6	1	0.125	0.035 ± 0.000	0.081	0.046	132.79	0.041	0.006	18.41
N=18									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	2.081 ± 0.021	2.711	0.630	30.28	1.802	-0.279	-13.40
4.2	1	0.757	0.438 ± 0.003	0.603	0.165	37.77	0.386	-0.052	-11.81
4.3	1	1.185	0.522 ± 0.006	0.716	0.194	37.13	0.461	-0.061	-11.66
4.4	1	1.500	1.754 ± 0.022	2.183	0.429	24.42	1.460	-0.294	-16.77
4.5	1	0.595	0.300 ± 0.003	0.418	0.118	39.28	0.265	-0.035	-11.68
5	1	0.125	0.104 ± 0.000	0.153	0.049	47.59	0.083	-0.020	-19.71
6	1	0.125	0.069 ± 0.000	0.153	0.084	121.50	0.064	-0.005	-7.09
N=24									
1	Inf	0.250	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
2	Inf	2.000	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
3	Inf	0.500	0.000 ± 0.000	0.000	0.000	-	0.000	0.000	-
4.1	1	2.200	3.798 ± 0.063	4.419	0.621	16.36	2.963	-0.835	-21.98
4.2	1	0.757	0.674 ± 0.007	0.880	0.206	30.54	0.559	-0.115	-17.08
4.3	1	1.185	0.743 ± 0.010	0.996	0.253	34.09	0.638	-0.105	-14.10
4.4	1	1.500	3.437 ± 0.040	3.776	0.340	9.89	2.570	-0.867	-25.22
4.5	1	0.595	0.444 ± 0.003	0.596	0.152	34.18	0.374	-0.070	-15.78
5	1	0.125	0.179 ± 0.001	0.250	0.070	39.07	0.128	-0.051	-28.48
6	1	0.125	0.125 ± 0.001	0.250	0.124	98.98	0.091	-0.034	-27.19

Table 4.9. Aggregated Server Model: Response Time Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model Result	Exponential Model Abs Error	Exponential Model % Error	General Model Result	General Model Abs Error	General Model % Error
N=6									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	2.562 ± 0.013	2.727	0.165	6.45	2.546	-0.015	-0.60
4.2	1	0.757	0.847 ± 0.002	0.895	0.048	5.63	0.847	0.000	-
4.3	1	1.185	1.301 ± 0.004	1.363	0.062	4.78	1.301	0.000	-
4.4	1	1.500	1.760 ± 0.008	1.895	0.135	7.66	1.760	0.000	-
4.5	1	0.595	0.660 ± 0.001	0.695	0.035	5.24	0.660	0.000	-
5	1	0.125	0.144 ± 0.000	0.155	0.011	7.60	0.144	0.000	-
6	1	0.125	0.137 ± 0.000	0.155	0.017	12.70	0.143	0.006	4.18
N=12									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	3.203 ± 0.012	3.627	0.424	13.24	3.142	-0.061	-1.91
4.2	1	0.757	0.997 ± 0.004	1.105	0.108	10.85	0.982	-0.015	-1.52
4.3	1	1.185	1.487 ± 0.006	1.617	0.131	8.80	1.465	-0.022	-1.45
4.4	1	1.500	2.290 ± 0.009	2.602	0.311	13.59	2.228	-0.062	-2.72
4.5	1	0.595	0.763 ± 0.001	0.842	0.079	10.31	0.753	-0.010	-1.26
5	1	0.125	0.178 ± 0.000	0.206	0.028	15.83	0.173	-0.005	-2.91
6	1	0.125	0.160 ± 0.000	0.206	0.046	28.87	0.166	0.006	4.01
N=18									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	4.280 ± 0.025	4.911	0.631	14.74	4.002	-0.278	-6.49
4.2	1	0.757	1.194 ± 0.004	1.360	0.166	13.87	1.143	-0.051	-4.30
4.3	1	1.185	1.709 ± 0.008	1.901	0.192	11.26	1.646	-0.062	-3.65
4.4	1	1.500	3.254 ± 0.023	3.683	0.429	13.17	2.960	-0.294	-9.04
4.5	1	0.595	0.894 ± 0.004	1.013	0.119	13.27	0.860	-0.034	-3.83
5	1	0.125	0.229 ± 0.000	0.278	0.049	21.57	0.208	-0.021	-8.97
6	1	0.125	0.194 ± 0.000	0.278	0.084	43.31	0.189	-0.005	-2.52
N=24									
1	Inf	0.250	0.250 ± 0.000	0.250	0.000	-	0.250	0.000	-
2	Inf	2.000	2.000 ± 0.000	2.000	0.000	-	2.000	0.000	-
3	Inf	0.500	0.500 ± 0.000	0.500	0.000	-	0.500	0.000	-
4.1	1	2.200	5.994 ± 0.062	6.619	0.625	10.42	5.163	-0.832	-13.87
4.2	1	0.757	1.431 ± 0.007	1.637	0.206	14.39	1.316	-0.115	-8.04
4.3	1	1.185	1.927 ± 0.011	2.181	0.254	13.18	1.823	-0.104	-5.39
4.4	1	1.500	4.936 ± 0.042	5.266	0.331	6.70	4.070	-0.866	-17.55
4.5	1	0.595	1.039 ± 0.003	1.191	0.152	14.63	0.969	-0.070	-6.20
5	1	0.125	0.310 ± 0.011	0.375	0.065	20.90	0.253	-0.056	-18.22
6	1	0.125	0.250 ± 0.001	0.375	0.124	49.54	0.216	-0.034	-13.63

Table 4.10. Aggregated Server Model: Throughput Rate Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model			General Model		
				Result	Abs Error	% Error	Result	Abs Error	% Error
N=6									
1	Inf	0.250	1.975 ± 0.001	1.915	-0.061	-3.07	1.948	-0.027	-1.39
2	Inf	2.000	1.877 ± 0.001	1.819	-0.058	-3.06	1.851	-0.026	-1.39
3	Inf	0.500	0.662 ± 0.000	0.642	-0.020	-3.06	0.653	-0.009	-1.40
4.1	1	2.200	0.113 ± 0.000	0.109	-0.004	-3.19	0.111	-0.002	-1.51
4.2	1	0.757	0.258 ± 0.000	0.250	-0.008	-3.09	0.255	-0.004	-1.43
4.3	1	1.185	0.139 ± 0.000	0.135	-0.004	-3.14	0.137	-0.002	-1.49
4.4	1	1.500	0.179 ± 0.000	0.173	-0.006	-3.12	0.176	-0.003	-1.44
4.5	1	0.595	0.304 ± 0.000	0.295	-0.009	-3.06	0.300	-0.004	-1.38
5	1	0.125	1.975 ± 0.001	1.915	-0.061	-3.07	1.948	-0.027	-1.39
6	1	0.125	1.975 ± 0.001	1.915	-0.061	-3.07	1.948	-0.027	-1.39
N=12									
1	Inf	0.250	3.752 ± 0.002	3.547	-0.205	-5.46	3.717	-0.035	-0.94
2	Inf	2.000	3.565 ± 0.002	3.370	-0.195	-5.46	3.531	-0.033	-0.93
3	Inf	0.500	1.257 ± 0.001	1.188	-0.069	-5.46	1.245	-0.012	-0.94
4.1	1	2.200	0.214 ± 0.000	0.202	-0.012	-5.55	0.212	-0.002	-1.01
4.2	1	0.757	0.490 ± 0.000	0.463	-0.027	-5.51	0.486	-0.005	-0.98
4.3	1	1.185	0.264 ± 0.000	0.250	-0.015	-5.55	0.262	-0.003	-1.00
4.4	1	1.500	0.340 ± 0.000	0.321	-0.019	-5.53	0.336	-0.003	-0.99
4.5	1	0.595	0.578 ± 0.000	0.547	-0.032	-5.47	0.573	-0.005	-0.93
5	1	0.125	3.752 ± 0.002	3.547	-0.205	-5.46	3.717	-0.035	-0.94
6	1	0.125	3.752 ± 0.002	3.547	-0.205	-5.46	3.717	-0.035	-0.94
N=18									
1	Inf	0.250	5.208 ± 0.004	4.821	-0.387	-7.43	5.258	0.050	0.95
2	Inf	2.000	4.948 ± 0.004	4.580	-0.368	-7.43	4.995	0.047	0.95
3	Inf	0.500	1.745 ± 0.001	1.615	-0.130	-7.43	1.761	0.017	0.95
4.1	1	2.200	0.297 ± 0.000	0.275	-0.022	-7.50	0.299	0.003	0.85
4.2	1	0.757	0.681 ± 0.001	0.630	-0.051	-7.47	0.687	0.006	0.91
4.3	1	1.185	0.367 ± 0.000	0.339	-0.027	-7.49	0.370	0.003	0.88
4.4	1	1.500	0.471 ± 0.000	0.436	-0.035	-7.48	0.476	0.004	0.90
4.5	1	0.595	0.803 ± 0.001	0.743	-0.060	-7.43	0.810	0.008	0.95
5	1	0.125	5.208 ± 0.004	4.821	-0.387	-7.43	5.258	0.050	0.95
6	1	0.125	5.208 ± 0.004	4.821	-0.387	-7.43	5.258	0.050	0.95
N=24									
1	Inf	0.250	6.220 ± 0.010	5.718	-0.502	-8.07	6.523	0.304	4.88
2	Inf	2.000	5.909 ± 0.009	5.432	-0.477	-8.07	6.197	0.288	4.88
3	Inf	0.500	2.084 ± 0.003	1.916	-0.168	-8.07	2.185	0.102	4.88
4.1	1	2.200	0.355 ± 0.001	0.326	-0.029	-8.16	0.372	0.017	4.79
4.2	1	0.757	0.813 ± 0.001	0.747	-0.066	-8.11	0.852	0.039	4.84
4.3	1	1.185	0.438 ± 0.001	0.402	-0.036	-8.15	0.459	0.021	4.80
4.4	1	1.500	0.563 ± 0.001	0.517	-0.046	-8.12	0.590	0.027	4.82
4.5	1	0.595	0.958 ± 0.001	0.881	-0.077	-8.07	1.005	0.047	4.88
5	1	0.125	6.220 ± 0.010	5.718	-0.502	-8.07	6.523	0.304	4.88
6	1	0.125	6.220 ± 0.010	5.718	-0.502	-8.07	6.523	0.304	4.88

Table 4.11. Aggregated Server Model: Queue Length Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model			General Model		
				Result	Abs Error	% Error	Result	Abs Error	% Error
N=6									
1	Inf	0.250	0.494 ± 0.000	0.479	-0.015	-3.06	0.487	-0.007	-1.38
2	Inf	2.000	3.754 ± 0.003	3.638	-0.115	-3.08	3.701	-0.053	-1.40
3	Inf	0.500	0.331 ± 0.001	0.321	-0.010	-3.03	0.326	-0.004	-1.34
4.1	1	2.200	0.288 ± 0.002	0.297	0.009	3.21	0.283	-0.006	-1.96
4.2	1	0.757	0.219 ± 0.001	0.224	0.005	2.28	0.216	-0.003	-1.56
4.3	1	1.185	0.181 ± 0.001	0.184	0.002	1.31	0.178	-0.003	-1.62
4.4	1	1.500	0.314 ± 0.002	0.328	0.014	4.40	0.310	-0.004	-1.39
4.5	1	0.595	0.201 ± 0.001	0.205	0.004	1.95	0.198	-0.003	-1.49
5	1	0.125	0.284 ± 0.000	0.297	0.012	4.27	0.280	-0.004	-1.39
6	1	0.125	0.272 ± 0.000	0.297	0.025	9.20	0.279	0.007	2.75
N=12									
1	Inf	0.250	0.938 ± 0.000	0.887	-0.051	-5.47	0.929	-0.009	-0.94
2	Inf	2.000	7.132 ± 0.005	6.740	-0.392	-5.50	7.062	-0.069	-0.97
3	Inf	0.500	0.628 ± 0.001	0.594	-0.034	-5.43	0.623	-0.006	-0.91
4.1	1	2.200	0.683 ± 0.003	0.733	0.049	7.21	0.665	-0.018	-2.69
4.2	1	0.757	0.489 ± 0.003	0.512	0.024	4.73	0.477	-0.012	-2.42
4.3	1	1.185	0.393 ± 0.003	0.404	0.011	2.77	0.383	-0.010	-2.45
4.4	1	1.500	0.777 ± 0.004	0.835	0.057	7.37	0.749	-0.028	-3.64
4.5	1	0.595	0.439 ± 0.004	0.460	0.021	4.81	0.431	-0.007	-1.70
5	1	0.125	0.667 ± 0.001	0.730	0.063	9.48	0.641	-0.025	-3.82
6	1	0.125	0.599 ± 0.001	0.730	0.131	21.80	0.617	0.018	3.03
N=18									
1	Inf	0.250	1.302 ± 0.001	1.205	-0.097	-7.44	1.315	0.012	0.95
2	Inf	2.000	9.885 ± 0.008	9.160	-0.724	-7.33	9.990	0.105	1.07
3	Inf	0.500	0.873 ± 0.001	0.808	-0.066	-7.54	0.881	0.007	0.83
4.1	1	2.200	1.266 ± 0.008	1.348	0.082	6.52	1.198	-0.068	-5.33
4.2	1	0.757	0.813 ± 0.003	0.857	0.043	5.33	0.785	-0.028	-3.46
4.3	1	1.185	0.626 ± 0.004	0.645	0.019	3.04	0.609	-0.017	-2.70
4.4	1	1.500	1.536 ± 0.014	1.606	0.071	4.60	1.408	-0.128	-8.32
4.5	1	0.595	0.719 ± 0.004	0.753	0.034	4.66	0.697	-0.022	-3.10
5	1	0.125	1.192 ± 0.002	1.342	0.150	12.54	1.096	-0.097	-8.09
6	1	0.125	1.011 ± 0.002	1.342	0.330	32.66	0.995	-0.016	-1.61
N=24									
1	Inf	0.250	1.555 ± 0.002	1.429	-0.126	-8.08	1.631	0.076	4.87
2	Inf	2.000	11.807 ± 0.018	10.864	-0.943	-7.99	12.394	0.588	4.98
3	Inf	0.500	1.043 ± 0.001	0.958	-0.086	-8.21	1.093	0.049	4.72
4.1	1	2.200	2.128 ± 0.027	2.155	0.027	1.27	1.918	-0.210	-9.88
4.2	1	0.757	1.164 ± 0.007	1.223	0.059	5.07	1.122	-0.042	-3.64
4.3	1	1.185	0.845 ± 0.005	0.877	0.032	3.81	0.837	-0.008	-1.01
4.4	1	1.500	2.778 ± 0.025	2.724	-0.054	-1.95	2.401	-0.377	-13.55
4.5	1	0.595	0.996 ± 0.004	1.049	0.053	5.35	0.974	-0.022	-2.20
5	1	0.125	1.893 ± 0.007	2.142	0.248	13.11	1.653	-0.240	-12.72
6	1	0.125	1.558 ± 0.006	2.142	0.584	37.48	1.411	-0.147	-9.44

Table 4.12. Aggregated Server Model: Station Utilization Comparison

Station	Servers	Service Time	Simulation Results	Exponential Model			General Model		
				Result	Abs Error	% Error	Result	Abs Error	% Error
N=6									
1	Inf	0.250	0.494 ± 0.000	0.479	-0.015	-3.06	0.487	-0.007	-1.38
2	Inf	2.000	3.754 ± 0.003	3.638	-0.115	-3.08	3.701	-0.053	-1.40
3	Inf	0.500	0.331 ± 0.001	0.321	-0.010	-3.03	0.326	-0.004	-1.34
4.1	1	2.200	0.248 ± 0.002	0.240	-0.008	-3.24	0.244	-0.004	-1.55
4.2	1	0.757	0.195 ± 0.001	0.189	-0.006	-3.01	0.193	-0.003	-1.32
4.3	1	1.185	0.165 ± 0.001	0.160	-0.005	-3.28	0.162	-0.003	-1.59
4.4	1	1.500	0.267 ± 0.001	0.260	-0.008	-2.82	0.264	-0.003	-1.14
4.5	1	0.595	0.181 ± 0.001	0.176	-0.006	-3.09	0.179	-0.003	-1.43
5	1	0.125	0.247 ± 0.000	0.239	-0.007	-3.03	0.244	-0.003	-1.37
6	1	0.125	0.247 ± 0.000	0.239	-0.007	-3.03	0.244	-0.003	-1.37
N=12									
1	Inf	0.250	0.938 ± 0.000	0.887	-0.051	-5.47	0.929	-0.009	-0.94
2	Inf	2.000	7.132 ± 0.005	6.740	-0.392	-5.50	7.062	-0.069	-0.97
3	Inf	0.500	0.628 ± 0.001	0.594	-0.034	-5.43	0.623	-0.006	-0.91
4.1	1	2.200	0.468 ± 0.001	0.444	-0.024	-5.11	0.466	-0.003	-0.56
4.2	1	0.757	0.371 ± 0.002	0.351	-0.020	-5.49	0.368	-0.004	-0.96
4.3	1	1.185	0.314 ± 0.001	0.296	-0.018	-5.71	0.310	-0.004	-1.18
4.4	1	1.500	0.509 ± 0.001	0.481	-0.028	-5.46	0.504	-0.005	-0.95
4.5	1	0.595	0.343 ± 0.001	0.325	-0.018	-5.31	0.341	-0.003	-0.77
5	1	0.125	0.469 ± 0.000	0.443	-0.026	-5.46	0.465	-0.004	-0.94
6	1	0.125	0.469 ± 0.000	0.443	-0.026	-5.48	0.465	-0.004	-0.96
N=18									
1	Inf	0.250	1.302 ± 0.001	1.205	-0.097	-7.44	1.315	0.012	0.95
2	Inf	2.000	9.885 ± 0.008	9.160	-0.724	-7.33	9.990	0.105	1.07
3	Inf	0.500	0.873 ± 0.001	0.808	-0.066	-7.54	0.881	0.007	0.83
4.1	1	2.200	0.650 ± 0.002	0.604	-0.046	-7.12	0.659	0.008	1.29
4.2	1	0.757	0.515 ± 0.001	0.477	-0.038	-7.46	0.520	0.005	0.92
4.3	1	1.185	0.435 ± 0.002	0.402	-0.033	-7.51	0.438	0.004	0.86
4.4	1	1.500	0.708 ± 0.002	0.654	-0.054	-7.57	0.713	0.006	0.81
4.5	1	0.595	0.478 ± 0.001	0.442	-0.036	-7.50	0.482	0.004	0.87
5	1	0.125	0.651 ± 0.001	0.603	-0.049	-7.45	0.657	0.006	0.91
6	1	0.125	0.651 ± 0.000	0.603	-0.048	-7.40	0.657	0.006	0.98
N=24									
1	Inf	0.250	1.555 ± 0.002	1.429	-0.126	-8.08	1.631	0.076	4.87
2	Inf	2.000	11.807 ± 0.018	10.864	-0.943	-7.99	12.394	0.588	4.98
3	Inf	0.500	1.043 ± 0.001	0.958	-0.086	-8.21	1.093	0.049	4.72
4.1	1	2.200	0.780 ± 0.002	0.716	-0.063	-8.13	0.817	0.038	4.81
4.2	1	0.757	0.616 ± 0.001	0.566	-0.050	-8.14	0.645	0.030	4.81
4.3	1	1.185	0.519 ± 0.001	0.477	-0.043	-8.21	0.544	0.024	4.71
4.4	1	1.500	0.844 ± 0.002	0.776	-0.068	-8.04	0.885	0.041	4.91
4.5	1	0.595	0.570 ± 0.001	0.524	-0.046	-8.06	0.598	0.028	4.88
5	1	0.125	0.778 ± 0.001	0.715	-0.063	-8.09	0.815	0.038	4.86
6	1	0.125	0.778 ± 0.001	0.715	-0.063	-8.12	0.815	0.038	4.83

4.4 Sortie Generation Rate

For the aircraft sortie generation model [9], the aircraft sortie generation rate is defined as the throughput rate at station 2. This performance measure estimates the expected number of sorties the system can generate per hour. Figures 4.1 and 4.2 display the sortie generation rate estimates for the single server model and the aggregated server model, respectively. The simulation data is based on the system parameters given in tables 4-1 and 4-7.

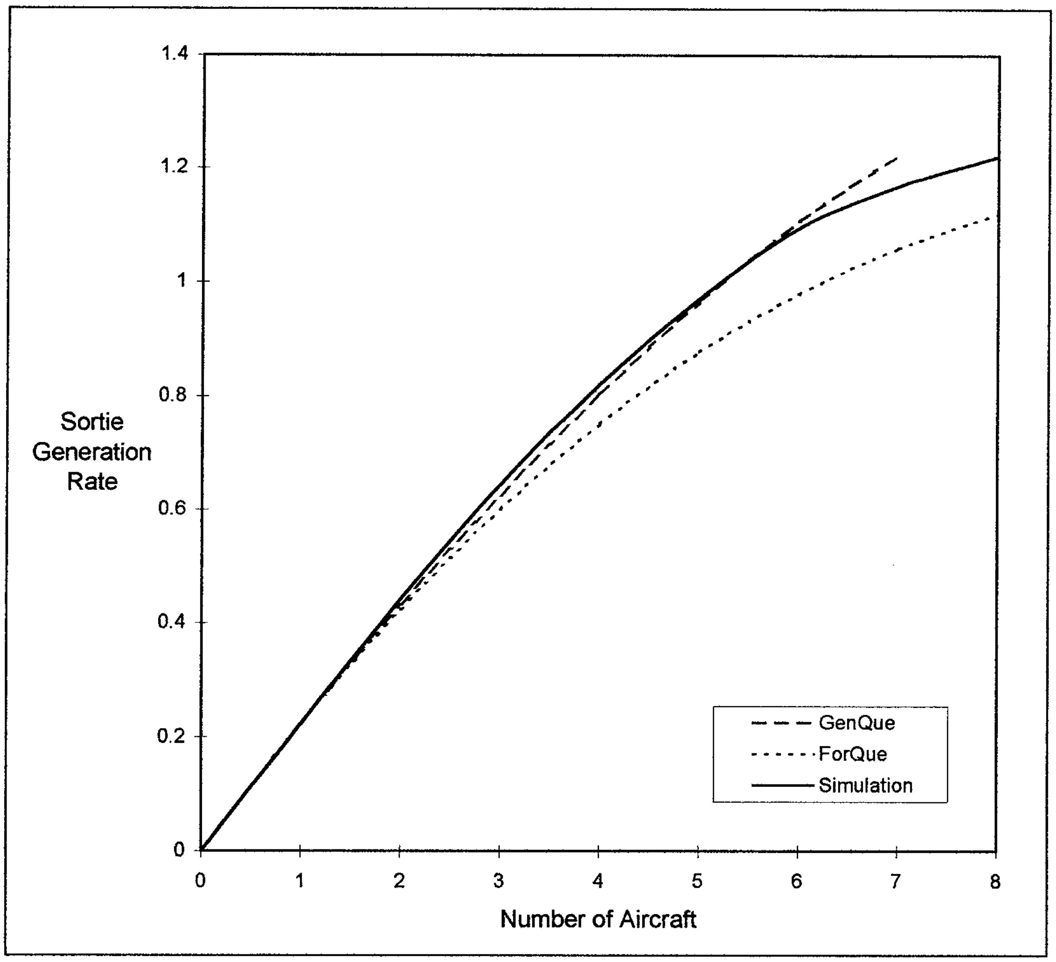

Figure 4.1 Sortie Generation Rate (Single Server Model)

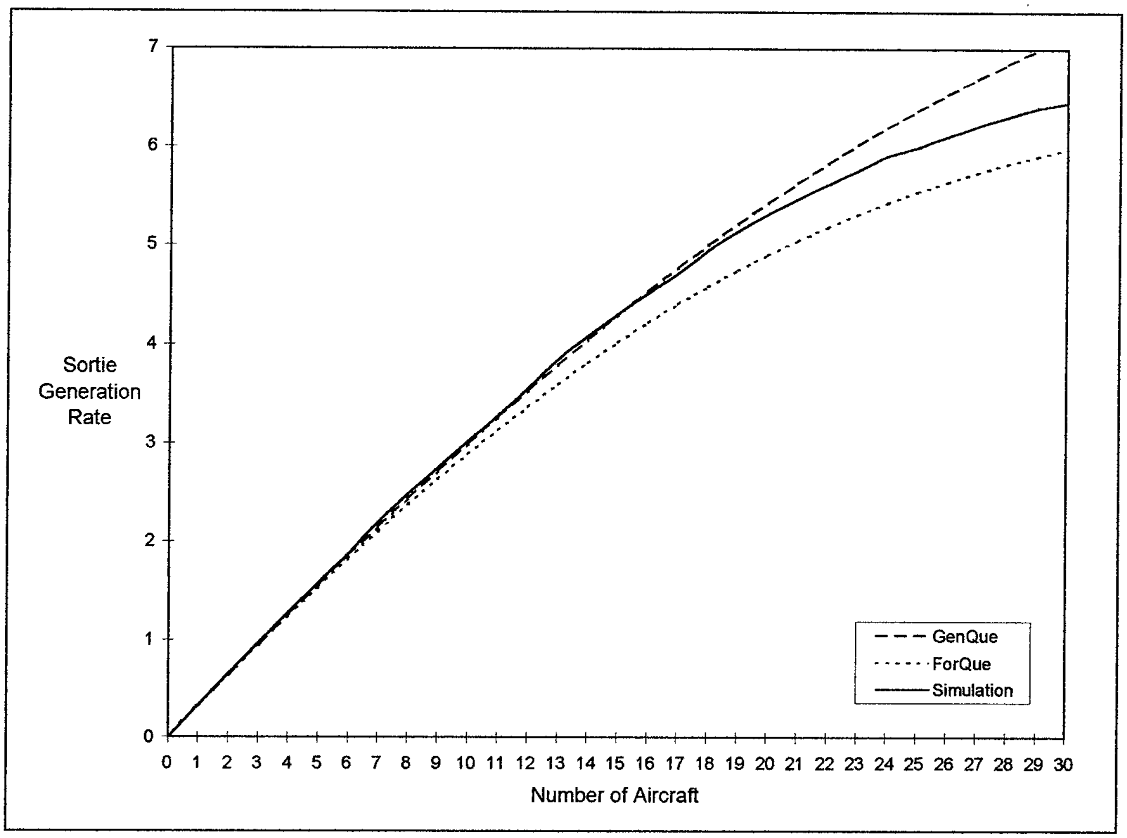

Figure 4.2 Sortie Generation Rate (Aggregated Server Model)

In both cases, the general model error is less than three percent at the lower utilization rates while the exponential model error is between 3 and 11 percent. The graphs illustrate the tendency for the general model's performance to decrease as the number of aircraft and station utilization increase.

While the general model is very accurate for systems composed of single server queues, the original example network contains several multiserver queues. In this case, the exponential model's capability to model multiserver queues makes it a better estimator of system performance. Figure 4.3 plots simulation and exponential model results for the example system with multiserver parameters given in table 3.1. The graph illustrates how the general model can only bound multiserver system performance.

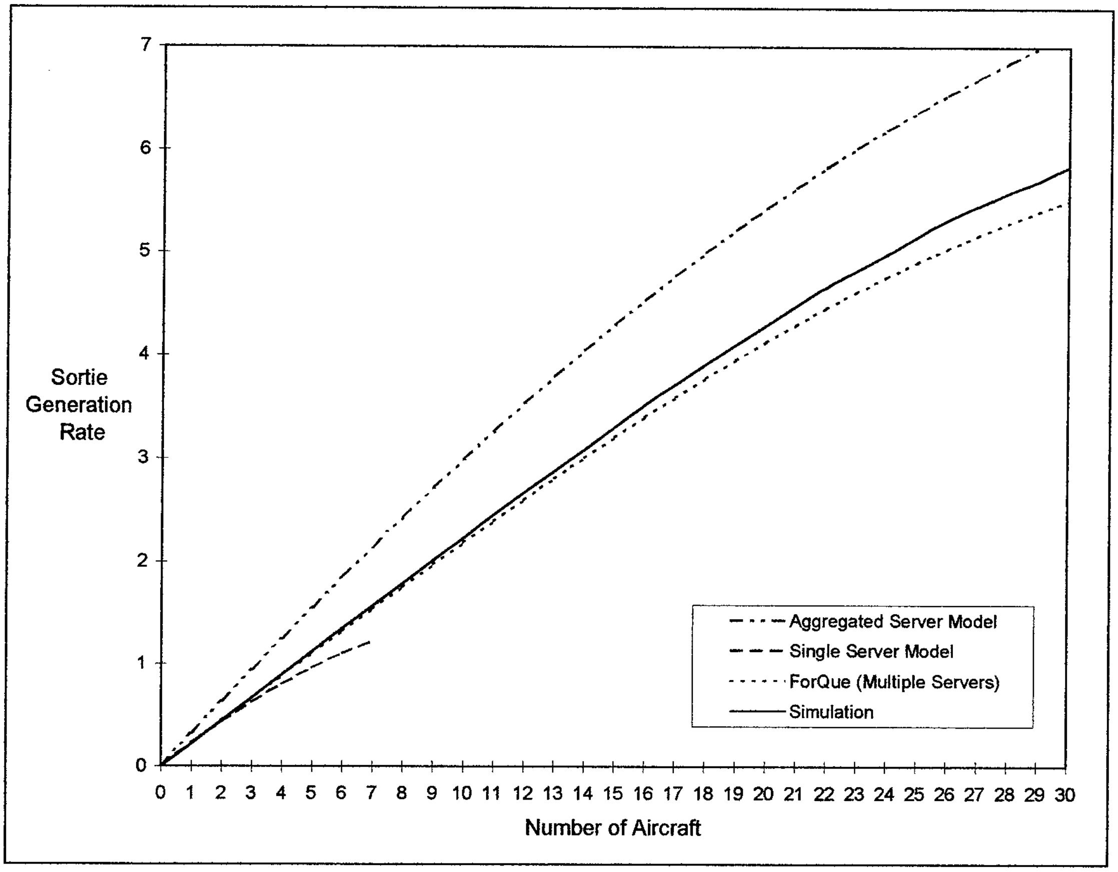

Figure 4.3 Sortie Generation Rate (Multiple Server Model)

4.5 Summary

The general service model accurately estimates system performance measures such as resource utilization and the aircraft sortie generation rate, for a system composed of GI/G/1 queues and delay stations, especially for lower levels of server utilization. The heuristic can also be used to identify when and where bottlenecks may develop, implying that system performance can be improved by increasing resource levels at the bottleneck station. For closed systems with multiserver stations, the general model can still be used to determine upper and lower bounds on system performance, however, the exponential model will probably provide better estimates of actual system performance.

V. Conclusions

5.1 General Observations

This methodology can be applied directly to analyze systems which can be modeled as open or closed networks of single server queues, delay stations, and fork-join nodes. For open networks, the method is applied for given arrival rates, omitting the open-to-closed network conversion.

The fork-join approximation (equation 31) depends on the substation response times and the given conditional repair probabilities. Response time approximations for open multiserver queues with infinite capacity already exist [27: 2806-2807]. Therefore, the methodology can also be extended to estimate system performance measures for open, uncapacitated networks with multiserver queues.

5.2 Methodology Limitations

The most significant limitation of this methodology is its current inability to explicitly model multiserver queues for closed networks. Waiting time approximations for GI/G/m queues already exist [27: 2806-2807], and the MVA algorithm [9] and QNA traffic variability equations [27:2799] can also be applied to multiserver queues. However, an open network to closed network conversion function, such as the one derived for GI/G/1 queues (equation 24), is still required before the method can be applied to capacitated open networks or closed networks with multiserver queues.

5.3 *Recommendations for Future Research*

The analytical model developed in this study can only provide upper and lower bounds for the aircraft sortie generation model under examination. To increase the model's fidelity, an open network to closed network conversion function for multiserver queues must be developed.

The example system uses low-variance service time distributions which are typical for the repair processes they represent [7, 12]. The model accurately estimates performance measures for a network of GI/G/1 queues and delay stations using those service time distributions. The robustness of the model can be tested by evaluating systems stations with high variance service time distributions, and a combination of stations with high and low variance service time distributions.

The fork-join approximation (equation 31) uses weights based on the assumption that response times at fork-join substations are exponentially distributed. A different weighting scheme to estimate $p_{ik}(N,S)$ in equation 29 may provide more accurate results.

5.4 *Summary*

The methodology accurately estimates system performance measures for a closed system composed of GI/G/1 queues, delay stations, and fork-join nodes, especially for lower levels of station utilization. When a closed system contains multiserver stations, the model can still be used to determine upper and lower bounds on system performance. The model may also be extended to examine uncapacitated, open networks with multiserver queues and fork-join nodes. Other potential applications include automated assembly systems and communications processes with synchronization requirements.

A.1 Description

The heuristic for The GenQue program uses the ForQue program from [9] to estimate performance measures for an equivalent exponential server network. The procedures in ForQue are modified to capture values required for the general queueing network analysis. Figure A.1 shows the relationship between the program's components.

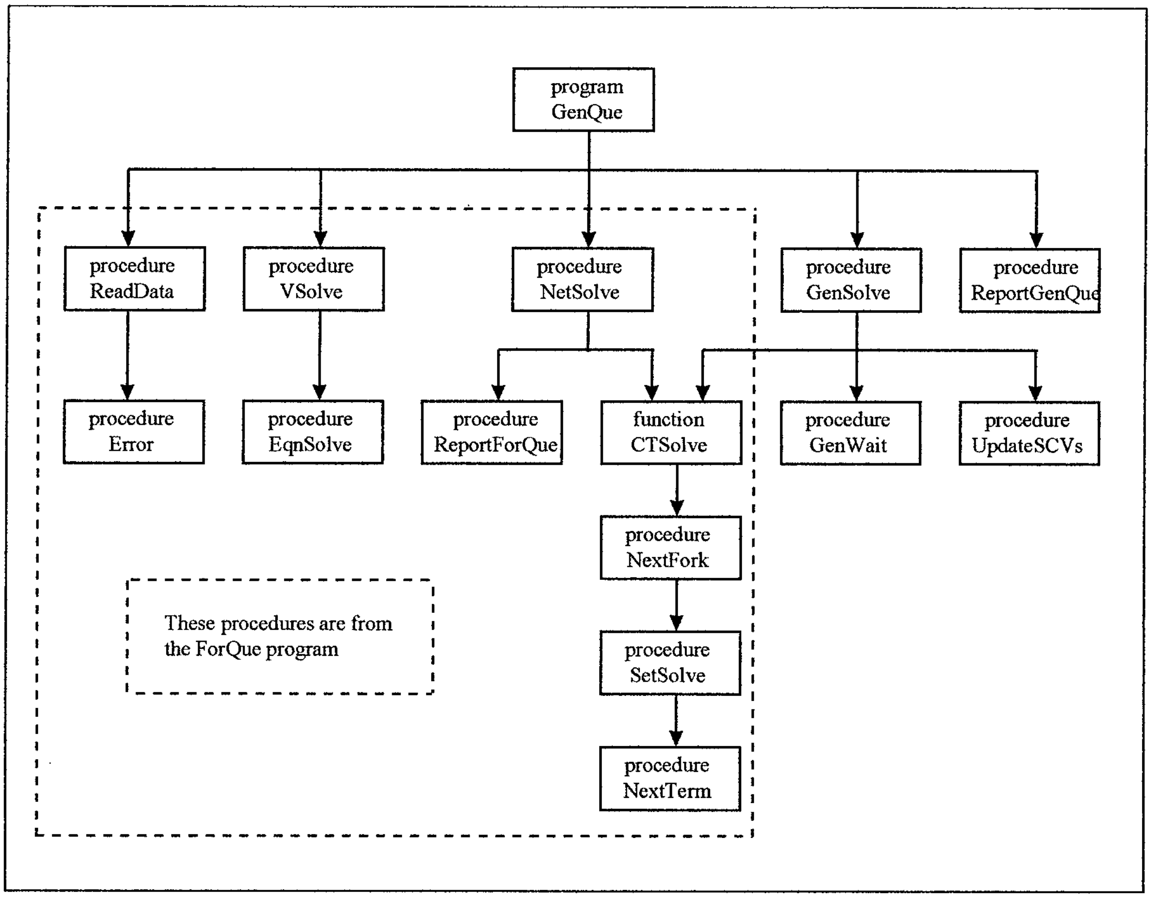

Figure A.1 Relationship Between GenQue Program Components

A.2 Input Data Files

To use the GenQue program, data must first be entered into a text file in the format displayed in figure A.2. Descriptions of each data entry are provided in the table. The table shows the input data file for the example network with 24 aircraft using the aggregated service rates from section 4.2.

N (customers) M (main stations)	24	6				
s_i (mean service times: 0 = fork-join node)	0.25	2.00	0.50	0.0	0.125	0.125
NC (number of channels: 0 = delay station 1 = single server)	0	0	0	5	1	1
$SCVS$ (SCV for service distribution)	0.00	0.02	0.29	0.00	0.29	0.29
$\boldsymbol{P}$ (transition probability matrix)	0	0.95	0.05	0	0	0
	0	0	0.3	0	0.7	0
	0	0	0	1	0	0
	0	0	0	0	1	0
	0	0	0	0	0	1
	1	0	0	0	0	0
s_{ik} (fork-join substation mean service times)	2.20	0.757	1.185	1.50	0.595	
NC (number of servers)	1	1	1	1	1	
$SCVS$ (SCV for service distribution)	0.29	0.29	0.29	0.29	0.29	
q_{ik} (conditional prob that service k is required)	0.17	0.39	0.21	0.27	0.46	

Figure A.2 Sample Input Data File

A.3 GenQue Program

```
program GenQue(input,output,Dat,Out);

{*********************************************************************}
{*                                                                   *}
{*  Estimates performance measures for closed networks of GI/G/1,    *}
{*  queues, delay stations, and a fork/join node with general service *}
{*  time distributions characterized by the first two moments.  This *}
{*  program uses the "ForQue" program written by D. C. Dietz to      *}
{*  estimate performance measures for the equivalent exponential     *}
{*  service network                                                  *}
{*                                                                   *}
{*  Language: Turbo-Pascal 6.0                                       *}
{*                                                                   *}
{*  Source: Major Daniel V. Hackman, dhackman@afit.af.mil            *}
{*          AFIT/EN117, 2950 P St, Wright-Patterson AFB OH 45433-7765 *}
{*                                                                   *}
{*********************************************************************}

const NMax=100; {maximum number of customers in network}
      MMax=20; {maximum number of stations in network}

type MIntArray=array[1..MMax] of integer;
     MRealArray=array[1..MMax] of real;
     PMatrix=array[1..MMax,1..MMax+1] of real;
     PPMatrix=array[1..MMax,0..NMax] of real;

var I,J,K,M,MM,N: integer;
    CT1: real;
    NC,Link: MIntArray;
    Q,QM,QN,Pik,R,RHO,S,SCV,SCVA,SCVD,SCVS,T,U,V,W,WOld: MRealArray;
    P: PMatrix;
    OutCode: char;
    FileName: string[24];
    Dat,Out: text;
```

```
{*********************************************************************}

procedure Error(ErrCode,I: integer);

{Reports errors}

begin
    writeln;
    case ErrCode of
        1: writeln('ERROR: more than ',NMax:2,' customers');
        2: writeln('ERROR: more than ',MMax:2,' stations');
        3: writeln('ERROR: routing probs from station ',I:2,
              ' do not sum to 1.0000');
        4: writeln('ERROR: no multi-server queues');
        5: writeln('ERROR: only one fork-join queue');
        6: writeln('ERROR: station ',I,' utilization greater than 1');
      end; {case}
    writeln('Program terminated; press <enter> to exit');
    readln;
    halt;
  end; {Error}
```

```
{*********************************************************************}

procedure ReadData(var N,M,MM: integer;
                   var NC: MIntArray;
                   var S,SCVS: MRealArray;
                   var P: PMatrix;
                   var Link: MIntArray;
                   var SCVA: MRealArray;
                   var Dat: text);

{Reads input data}

var I,J,K: integer;
    PSum: real;

begin
    read(Dat,N);
    if not (N in [1..NMax]) then Error(1,0);
    read(Dat,M);
    if not (M in [1..MMax]) then Error(2,0);
    for J:=1 to M do read(Dat,S[J]);
    for J:=1 to M do begin
        read(Dat,NC[J]);
        if (NC[J]>1) and (S[J]>0) then Error(4,0); {No multiserver
queues}
      end; {for}
    for J:=1 to M do read(Dat,SCVS[J]);
    for I:=1 to M do begin
        PSum:=0;
        for J:=1 to M do begin
            read(Dat,P[I,J]);
            PSum:=PSum+P[I,J];
          end; {for}
        if (PSum<0.9999) or (PSum>1.0001) then Error(3,I);
      end; {for}
    MM:=M;
    for I:=1 to M do begin
        if (S[I]>0) then Link[I]:=0 else begin
            if (MM>M) then Error(5,0);      {only 1 fork-join queue}
            Link[I]:=MM+1;
            for J:=MM+1 to MM+NC[I] do begin
                Link[J]:=I;
                read(Dat,S[J]);
                for K:=1 to M do P[K,J]:=0;
              end; {for}
            for J:=MM+1 to MM+NC[I] do read(Dat,NC[J]);
            for J:=MM+1 to MM+NC[I] do read(Dat,SCVS[J]);
            for J:=MM+1 to MM+NC[I] do read(Dat,P[I,J]);
            MM:=MM+NC[I];
            if (MM>MMax) then Error(2,0);
          end; {else}
      end; {for}
    SCVA[1]:=1;
    for I:=1 to MM do WOld[I]:=0;
  end; {ReadData}
```

```
{************************************************************************}

procedure EqnSolve(var M: integer;
                   var X: PMatrix;
                   var V: MRealArray);

{Solves simultaneous equations}

var I,J,K,IC,KK,MM,IT,IS: integer;
    B,W,C: real;
    ID: MIntArray;
    Y: MRealArray;

begin
    MM:=M+1;
    for I:=1 to M do ID[I]:=I;
    K:=1;
    repeat
        KK:=K+1;
        IS:=K;
        IT:=K;
        B:=abs(X[K,K]);
        for I:=K to M do for J:=K to M do
          if (abs(X[I,J])>B) then begin
              IS:=I;
              IT:=J;
              B:=abs(X[I,J]);
            end; {if}
        if (IS>K) then for J:=K to MM do begin
            C:=X[IS,J];
            X[IS,J]:=X[K,J];
            X[K,J]:=C;
          end; {for}
        if (IT>K) then begin
            IC:=ID[K];
            ID[K]:=ID[IT];
            ID[IT]:=IC;
            for I:=1 to M do begin
                C:=X[I,IT];
                X[I,IT]:=X[I,K];
                X[I,K]:=C;
              end; {for}
          end; {if}
        for J:=KK to MM do begin
            X[K,J]:=X[K,J]/X[K,K];
            for I:=KK to M do begin
                W:=X[I,K]*X[K,J];
                X[I,J]:=X[I,J]-W;
                if (abs(X[I,J])<0.00001*abs(W)) then X[I,J]:=0;
              end; {for}
          end; {for}
        K:=KK;
      until (K=M);
    Y[M]:=X[M,MM]/X[M,M];
```

```
  for I:=1 to M-1 do begin
      K:=M-I;
      KK:=K+1;
      Y[K]:=X[K,MM];
      for J:=KK to M do Y[K]:=Y[K]-X[K,J]*Y[J];
    end; {for}
  for I:=1 to M do for J:=1 to M do
    if (ID[J]=I) then V[I]:=Y[J];
end; {EqnSolve}
```

```
{*********************************************************************}

procedure VSolve(var M,MM: integer;
                 var P: PMatrix;
                 var NC,Link: MIntArray;
                 var V: MRealArray);

{Calculates visit ratios}

var I,J,K: integer;
    X: PMatrix;

begin
    X[1,1]:=1;
    X[1,M+1]:=1;
    for I:=2 to M do begin
        X[1,I]:=0;
        for J:=1 to M do X[I,J]:=P[J,I];
        X[I,M+1]:=0;
      end; {for}
    for I:=2 to M do X[I,I]:=X[I,I]-1;
    EqnSolve(M,X,V);
    for J:=M+1 to MM do V[J]:=V[Link[J]]*P[Link[J],J];
  end; {VSolve}
```

```
{*********************************************************************}

procedure NextTerm(K,Sign: integer;
                   Num,Den: real;
                   var J: integer;
                   var Pwait: real;
                   var Rate: MRealArray);

var L: integer;

begin
    Num:=Num+Rate[K];
    Den:=Den+Rate[K];
    Sign:=-Sign;
    PWait:=PWait+Sign*Num/Den;
    if (K<J-1) then
      for L:=K+1 to J-1 do NextTerm(L,Sign,Num,Den,J,PWait,Rate);
  end; {NextTerm}
```

```
{*********************************************************************}

procedure SetSolve(var I,NSet: integer;
                   var PSet,CT: real;
                   var V,Pik,Rate: MRealArray;
                   var ID: MIntArray);

{Updates cycle time for a particular set of active forks}

var J,JJ,K,KK,Sign,TempID: integer;
    Num,Den,PWait,Time,TempRate: real;

begin
    if (NSet>0) then begin
        Time:=1/Rate[1];
        for JJ:=1 to NSet do begin
            if (JJ>1) then TempRate:=Rate[1];
            if (JJ>1) then TempID:=ID[1];
            if (JJ>1) then for KK:=1 to NSet-1 do begin
                Rate[KK]:=Rate[KK+1];
                ID[KK]:=ID[KK+1];
              end; {for}
            if (JJ>1) then Rate[NSet]:=TempRate;
            if (JJ>1) then ID[NSet]:=TempID;
            for J:=2 to NSet do begin
                PWait:=0;
                Num:=0;
                Den:=Rate[J];
                for K:=1 to J-1 do begin
                    Sign:=-1;
                    NextTerm(K,Sign,Num,Den,J,PWait,Rate);
                  end; {for}
                Time:=Time+PWait/Rate[J];
              end; {for}
            if (NSet=1) then Pik[ID[1]]:=Pik[ID[1]]+PSet
              else Pik[ID[NSet]]:=Pik[ID[NSet]]+PSet*PWait;
          if (JJ=1) then CT:=CT+V[I]*PSet*Time;
          end; {for}
      end; {if}
  end; {SetSolve}
```

```
{*********************************************************************}

procedure NextFork(Next,NSet: integer;
                   PSet: real;
                   Rate: MRealArray;
                   ID: MIntArray;
                   var I,High: integer;
                   var CT: real;
                   var V,R: MRealArray);

begin
    if (Next=High) then SetSolve(I,NSet,PSet,CT,V,Pik,Rate,ID) else
        begin
        Next:=Next+1;
        if (P[I,Next]<1) then
          NextFork(Next,NSet,PSet*(1-P[I,Next]),Rate,ID,I,High,CT,V,R);
        NSet:=NSet+1;
        Rate[NSet]:=1/R[Next];
        ID[NSet]:=Next;
        NextFork(Next,NSet,PSet*P[I,Next],Rate,ID,I,High,CT,V,R);
      end; {else}
  end; {NextFork}
```

```
{*******************************************************************}

function CTSolve(var M: integer;
                 var P: PMatrix;
                 var Link: MIntArray;
                 var Pik,S,V,R: MRealArray): real;

{Returns network cycle time}

var I,Next,NSet,High: integer;
    CT,PSet: real;
    Rate: MRealArray;
    ID: MIntArray;

begin
    CT:=0;
    for I:=M+1 to MM do Pik[I]:=0;
    for I:=1 to M do if (S[I]>0) then CT:=CT+V[I]*R[I] else begin
        Next:=Link[I];
        High:=Link[I]+NC[I]-1;
        NSet:=0;
        PSet:=1;
        if (P[I,Next]<1) then begin
            PSet:=1-P[I,Next];
            NextFork(Next,NSet,PSet,Rate,ID,I,High,CT,V,R);
          end; {if}
        NSet:=1;
        PSet:=P[I,Next];
        Rate[1]:=1/R[Next];
        ID[1]:=Next;
        NextFork(Next,NSet,PSet,Rate,ID,I,High,CT,V,R);
      end; {for}
    CTSolve:=CT;
  end; {CTSolve}
```

```
{************************************************************************}

procedure ReportForQue(var K,M,MM: integer;
                       var NC: MIntArray;
                       var S,V,T,Q,R,W,U: MRealArray;
                       var Out: text);

{Writes ForQue performance report}

var I,J: integer;

begin
    writeln(Out,'Number of Customers =',K:3);
    writeln(Out);
    writeln(Out,'Sta-  Nmbr  Average    Visit  Through ',
      '   Queue  Respons     Wait   Utiliz');
    writeln(Out,'tion  Chls   Svc Tm     Ratio     -put ',
      '  Length     Time     Time   -ation');
    writeln(Out,'====  ====  =======  =======  ======= ',
      ' =======  =======  =======  =======');
    for I:=1 to M do
      writeln(Out,I:4,NC[I]:6,S[I]:9:4,V[I]:9:4,
        T[I]:9:4,Q[I]:9:4,R[I]:9:4,W[I]:9:4,U[I]:9:4);
    J:=0;
    for I:=M+1 to MM do begin
        if (J=0) then writeln(Out,'-----------------------',
          '-----------------------------------------------');
        J:=J+1;
        writeln(Out,Link[I]:2,'-',J:1,NC[I]:6,S[I]:9:4,
        V[I]:9:4,T[I]:9:4,Q[I]:9:4,R[I]:9:4,W[I]:9:4,U[I]:9:4);
        if (Link[I+1]>Link[I]) then J:=0;
      end; {for}
    writeln(Out);
    writeln(Out);
  end; {ReportForQue}
```

```
{*****************************************************************}

procedure NetSolve(var N,M,MM: integer;
                   var CT1: real;
                   var NC: MIntArray;
                   var P: PMatrix;
                   var RHO,Q,R,S,T,U,V,W: MRealArray;
                   var OutCode: char;
                   var Out: text);

{Calculates performance measures for the exponential queueing network}

var I,J,K: integer;
    Cap: real;
    PP: PPMatrix;

begin
    for I:=1 to MM do begin
        PP[I,0]:=1;
        Q[I]:=0;
      end; {for}
    for K:=1 to N do begin
        for I:=M+1 to MM do Pik[I]:=0;
        for I:=1 to MM do
          if (NC[I]=0) or (NC[I]>=N) or (S[I]=0) then R[I]:=S[I]
            else if (NC[I]=1) then R[I]:=S[I]*(1+Q[I]) else begin
                R[I]:=0;
                for J:=1 to K do begin
                    if (J<NC[I]) then Cap:=J else Cap:=NC[I];
                    R[I]:=R[I]+J*PP[I,J-1]/Cap;
                  end; {for}
                R[I]:=R[I]*S[I];
              end; {else}
        CT1:=CTSolve(M,P,Link,Pik,S,V,R);
        for I:=1 to MM do begin
            T[I]:=K*V[I]/CT1;
            Q[I]:=R[I]*T[I];
            W[I]:=R[I]-S[I];
            U[I]:=S[I]*T[I];
            if (K=N-1) then QM[I]:=Q[I];                    {Store Q(N-1)}
            if (K=N) then QN[I]:=Q[I];                        {Store Q(N)}
            if (NC[I]>0) and (NC[I]<N) and (S[I]>0) then begin
                for J:=K downto 1 do begin
                    if (J<NC[I]) then Cap:=J else Cap:=NC[I];
                    PP[I,J]:=U[I]*PP[I,J-1]/Cap;
                  end; {for}
                PP[I,0]:=1;
                for J:=1 to K do PP[I,0]:=PP[I,0]-PP[I,J];
              end; {if}
          end; {for}
        if (OutCode in ['y','Y']) or (K=N) then
          ReportForQue(K,M,MM,NC,S,V,T,Q,R,W,U,Out);
      end; {for}
  end; {NetSolve}
```

```
{***********************************************************************}

procedure ReportGenQue(var M,MM: integer;
                       var NC: MIntArray;
                       var SCVS,SCVA,SCVD,T,Q,R,W,U: MRealArray;
                       var Out: text);

{Writes GenQue performance report}

var I,J: integer;

begin
    writeln(Out,'Number of Customers =',N:3);
    writeln(Out);
    writeln(Out,'Sta-  Service  Arrival   Depart  Through ',
      '   Queue  Respons     Wait   Utiliz');
    writeln(Out,'tion      SCV      SCV      SCV     -put ',
      ' Length     Time     Time   -ation');
    writeln(Out,'==== ======= ======= ======= ======= ',
      ' ======= ======= ======= =======');
    for I:=1 to M do
      writeln(Out,I:4,SCVS[I]:9:4,SCVA[I]:9:4,SCVD[I]:9:4,
        T[I]:9:4,Q[I]:9:4,R[I]:9:4,W[I]:9:4,U[I]:9:4);
    J:=0;
    for I:=M+1 to MM do begin
        if (J=0) then writeln(Out,'-----------------------',
          '------------------------------------------------------');
        J:=J+1;
        writeln(Out,Link[I]:2,'-',J:1,SCVS[I]:9:4,SCVA[I]:9:4,
        SCVD[I]:9:4,T[I]:9:4,Q[I]:9:4,R[I]:9:4,W[I]:9:4,U[I]:9:4);
        if (Link[I+1]>Link[I]) then J:=0;
      end; {for}
    writeln(Out);
    writeln(Out);
  end; {ReportGenQue}
```

```
{*********************************************************************}

procedure GenWait(MM: integer;
                  NC: MIntArray;
                  var SCVA,SCVS,RHO,QM,QN,R,W: MRealArray);

{Computes waiting time for GI/G/1 queues using current arrival process
SCV}

var I: integer;
    F,G,WI: MRealArray;

begin
    for I:=1 to MM do if (NC[I]<>1) then W[I]:=0 else begin
        if (SCVA[I]>=1.0) then G[I]:=1
          else G[I]:=Exp(-(2*(1-RHO[I])*(1-SCVA[I])*(1-SCVA[I]))
                          /(3*RHO[I]*(SCVA[I]+SCVS[I])));
        WI[I]:=G[I]*(SCVA[I]+SCVS[I])/2*S[I]*U[I]/(1-U[I]);
        F[I]:=(QM[I]/QN[I])/(1+(WI[I]/S[I])*(1-(QM[I]/QN[I])));
        W[I]:=F[I]*WI[I];
      end; {for}
    for I:=1 to MM do R[I]:=S[I]+W[I];
  end; {GenWait}
```

```
{********************************************************************}

procedure UpdateSCVs (var SCV,SCVA,SCVD: MRealArray;
                      M: integer;
                      NC: MIntArray;
                      S,RHO,SCVS: MRealArray);

var I,J,K: integer;
    PikTot: real;
    AGAIN: string[1];

{Iteration loop to update SCVs given new RHO values}

begin
    for I:=1 to M do begin                                {departure SCVs}
        if (S[I]=0) then begin                                 {fork-join}
            SCVD[I]:=0;
            PikTot:=0;
            for K:=M+1 to M+NC[I] do begin
                PikTot:=PikTot+Pik[K];
                SCVD[K]:=RHO[K]*RHO[K]*SCVS[K]
                         +(1-RHO[K]*RHO[K])*SCVA[K];
                SCVD[I]:=SCVD[I]+Pik[K]*SCVD[K];
              end; {for}
            SCVD[I]:=SCVD[I]+(1-PikTot)*((1-PikTot)*SCVA[I]+PikTot);
          end; {if}
        if (NC[I]=1) then                                        {GI/G/1}
          SCVD[I]:=RHO[I]*RHO[I]*SCVS[I]+(1-RHO[I]*RHO[I])*SCVA[I];
        if (NC[I]=0) and (S[I]>0) then                    {delay station}
          SCVD[I]:=RHO[I]*RHO[I]+(1-RHO[I]*RHO[I])*SCVA[I];
                                                            {arrival SCVs}
        if (I=M) then begin                                {last station}
            SCVA[1]:=0;
            for J:=2 to M do begin
                SCVA[1]:=SCVA[1]+V[J]*P[J,1]*(P[J,1]*SCVD[J]+1-P[J,1]);
                if (S[1]=0) then for K:=M+1 to M+NC[1] do    {fork-join}
                    SCVA[K]:=P[1,K]*SCVA[1]+1-P[1,K];
              end; {for}
          end; {if}
        if (I<M) then begin                            {not last station}
            SCVA[I+1]:=0;
            for J:=1 to M do if (J<>I+1) then SCVA[I+1]:=SCVA[I+1]
                    +V[J]/V[I+1]*P[J,I+1]*(P[J,I+1]*SCVD[J]+1-P[J,I+1]);
            if (S[I+1]=0) then for K:=M+1 to M+NC[I+1] do    {fork-join}
              SCVA[K]:=P[I+1,K]*SCVA[I+1]+1-P[I+1,K];
          end; {if}
      end; {for}
    AGAIN:='N';
    for I:=1 to MM do begin                     {check for convergence}
        if abs(SCV[I]-SCVA[I])>0.001 then AGAIN:='Y';
        SCV[I]:=SCVA[I];                                {store old SCVAs}
      end; {for}
    if (AGAIN='Y') then UpdateSCVs(SCV,SCVA,SCVD,M,NC,S,RHO,SCVS);
  end; {UpdateSCVs}
```

```
{*********************************************************************}

procedure GenSolve(var CT1: real;
                   var RHO,WOld: MRealArray;
                   N,MM: integer;
                   NC: MIntArray);

{Calculates performance measures for the general queueing network}

var AGAIN: string[1];

begin
    for I:=1 to MM do begin
        if (NC[I]=1) then RHO[I]:=S[I]*N*V[I]/CT1
          else if (NC[I]=0) and (S[I]>0) then RHO[I]:=S[I]*V[I]/CT1
          else RHO[I]:=0;
      if (RHO[I]>1) then Error(6,I);
      end; {for}
    UpdateSCVs(SCV,SCVA,SCVD,M,NC,S,RHO,SCVS);
    GenWait(MM,NC,SCVA,SCVS,RHO,QM,QN,R,W);
    CT1:=CTSolve(M,P,Link,Pik,S,V,R);
    AGAIN:='N';
    for I:=1 to MM do begin                      {check for convergence}
        if abs(W[I]-WOld[I])>0.001 then AGAIN:='Y';
        WOld[I]:=W[I];                            {store old wait times}
      end; {for}
    if (AGAIN='Y') then GenSolve(CT1,RHO,WOld,N,MM,NC);
    for I:=1 to MM do begin
        T[I]:=N*V[I]/CT1;
        Q[I]:=R[I]*T[I];
        U[I]:=S[I]*T[I];
      end; {for}
  end; {GenSolve}
```

```
{*********************************************************************}

begin
    writeln;
    writeln('> Running GenQue');
    writeln;
    write('Enter Input Filename:  ');
    readln(FileName);
    assign(Dat,FileName);
    reset(Dat);
    write('Enter Output Filename (printer=PRN):  ');
    readln(FileName);
    assign(Out,FileName);
    rewrite(Out);
    write('Intermediate output? (Y/N):  ');
    readln(OutCode);
    writeln;
    writeln('> Reading data ...');
    ReadData(N,M,MM,NC,S,SCVS,P,Link,SCVA,Dat);
    writeln('> Calculating visit ratios ...');
    VSolve(M,MM,P,NC,Link,V);
    writeln('> Calculating performance measures ...');
    writeln;
    writeln(Out,'*** ForQue Performance Report ***');
    writeln(Out);
    NetSolve(N,M,MM,CT1,NC,P,RHO,Q,R,S,T,U,V,W,OutCode,Out);
    writeln(Out,'*** GenQue Performance Report ***');
    GenSolve(CT1,RHO,WOld,N,MM,NC);
    writeln(Out);
    ReportGenQue(M,MM,NC,SCVS,SCVA,SCVD,T,Q,R,W,U,Out);
    close(Dat);
    close(Out);
    writeln('> Program complete.');
    writeln;
  end. {GenQue}
```

Appendix B. Sample Output Data

*** ForQue Performance Report ***

Number of Customers = 24

Sta-tion	Nmbr Chls	Average Svc Tm	Visit Ratio	Through-put	Queue Length	Respons Time	Wait Time	Utiliz-ation
1	0	0.2500	1.0000	5.7178	1.4294	0.2500	0.0000	1.4294
2	0	2.0000	0.9500	5.4319	10.8638	2.0000	0.0000	10.8638
3	0	0.5000	0.3350	1.9155	0.9577	0.5000	0.0000	0.9577
4	5	0.0000	0.3350	1.9155	0.0000	0.0000	0.0000	0.0000
5	1	0.1250	1.0000	5.7178	2.1416	0.3745	0.2495	0.7147
6	1	0.1250	1.0000	5.7178	2.1416	0.3745	0.2495	0.7147
4-1	1	2.2000	0.0570	0.3256	2.1553	6.6190	4.4190	0.7164
4-2	1	0.7570	0.1307	0.7470	1.2228	1.6369	0.8799	0.5655
4-3	1	1.1850	0.0704	0.4022	0.8773	2.1810	0.9960	0.4767
4-4	1	1.5000	0.0905	0.5172	2.7236	5.2663	3.7663	0.7758
4-5	1	0.5950	0.1541	0.8811	1.0492	1.1907	0.5957	0.5243

*** GenQue Performance Report ***

Number of Customers = 24

Sta-tion	Service SCV	Arrival SCV	Depart SCV	Through-put	Queue Length	Respons Time	Wait Time	Utiliz-ation
1	0.0000	0.3333	0.3365	6.5233	1.6308	0.2500	0.0000	1.6308
2	0.0200	0.3696	0.5377	6.1972	12.3943	2.0000	0.0000	12.3943
3	0.2900	0.8771	0.8773	2.1853	1.0927	0.5000	0.0000	1.0927
4	0.0000	0.8773	0.6749	2.1853	0.0000	0.0000	0.0000	0.0000
5	0.2900	0.6759	0.4193	6.5233	1.6526	0.2533	0.1283	0.8154
6	0.2900	0.4193	0.3333	6.5233	1.4107	0.2163	0.0913	0.8154
4-1	0.2900	0.9791	0.5188	0.3715	1.9180	5.1629	2.9629	0.8173
4-2	0.2900	0.9522	0.6766	0.8523	1.1215	1.3159	0.5589	0.6452
4-3	0.2900	0.9742	0.7719	0.4589	0.8366	1.8231	0.6381	0.5438
4-4	0.2900	0.9669	0.4367	0.5900	2.4014	4.0698	2.5698	0.8851
4-5	0.2900	0.9436	0.7098	1.0052	0.9740	0.9689	0.3739	0.5981

Appendix C. Sortie Generation Simulation Program

```
GEN,HACKMAN,GENERAL MODEL,2/19/97,11,Y,N,Y/Y,N,,132;
LIMITS,12,4,400;
INTLC, XX(1)=0.17,XX(2)=0.39,XX(3)=0.21,XX(4)=0.27,
       XX(5)=0.46;  mx routing probabilities
;
NETWORK;
;
      RESOURCE/CREWCHF(1),1;
      RESOURCE/MUNCREW(1),2;
      RESOURCE/AFSC1(1),3;
      RESOURCE/AFSC2(1),4;
      RESOURCE/AFSC3(1),5;
      RESOURCE/AFSC4(1),6;
      RESOURCE/AFSC5(1),7;
;
      CREATE,0,,3,24;
      ASSIGN,II=10;
;
TAXI  ASSIGN,ATRIB(2)=TNOW;
      ACT/1,0.25;
      COLCT(1),TNOW-ATRIB(2),TAXI TIME;
      COLCT(11),TNOW-ATRIB(3),CYCLE TIME; ,40/0/1;
      ASSIGN,ATRIB(1)=0,ATRIB(3)=TNOW,1;
      ACT,,0.05,MX;
      ACT;
;
FLY   ASSIGN,ATRIB(2)=TNOW;
      ACT/2,UNFRM(1.5,2.5,1);
      COLCT(2),TNOW-ATRIB(2),FLY TIME;
      ASSIGN,ATRIB(1)=1,1;
      ACT,,0.30,MX;
      ACT;
;
TURN  ASSIGN,ATRIB(2)=TNOW;
      AWAIT,CREWCHF/1;
      ACT/4,RLOGN(0.125,0.067,1);
      FREE,CREWCHF/1;
      COLCT(4),TNOW-ATRIB(2),TURN TIME;
;
ARM   ASSIGN,ATRIB(2)=TNOW;
      AWAIT,MUNCREW/1;
      ACT/5,RLOGN(0.125,0.067,1);
      FREE,MUNCREW;
      COLCT(5),TNOW-ATRIB(2),ARM TIME,,1;
      ACT,,,TAXI;
;
MX    ASSIGN,ATRIB(2)=TNOW;
      ACT/3,RLOGN(0.5,0.269,1);
      COLCT(3),TNOW-ATRIB(2),TSHOOT TIME,,5;
      ACT,,,MX1;
      ACT,,,MX2;
      ACT,,,MX3;
      ACT,,,MX4;
      ACT,,,MX5;
;
```

```
MX1     ASSIGN,ATRIB(2)=TNOW,1;
        ACT,,1-XX(1),Q1;
        ACT;
        AWAIT,AFSC1/1;
        ACT/6,RLOGN(2.20,1.185,1);
        FREE,AFSC1/1;
        COLCT(6),TNOW-ATRIB(2),MX1 TIME; ,40/0/1;
Q1      QUE(8),,,,JOIN;
;
MX2     ASSIGN,ATRIB(2)=TNOW,1;
        ACT,,1-XX(2),Q2;
        ACT;
        AWAIT,AFSC2/1;
        ACT/7,RLOGN(0.757,0.408,1);
        FREE,AFSC2/1;
        COLCT(7),TNOW-ATRIB(2),MX2 TIME; ,40/0/1;
Q2      QUE(9),,,,JOIN;
;
MX3     ASSIGN,ATRIB(2)=TNOW,1;
        ACT,,1-XX(3),Q3;
        ACT;
        ASSIGN,ATRIB(2)=TNOW;
        AWAIT,AFSC3/1;
        ACT/8,RLOGN(1.185,0.638,1);
        FREE,AFSC3/1;
        COLCT(8),TNOW-ATRIB(2),MX3 TIME; ,40/0/1;
Q3      QUE(10),,,,JOIN;
;
MX4     ASSIGN,ATRIB(2)=TNOW,1;
        ACT,,1-XX(4),Q4;
        ACT;
        AWAIT,AFSC4/1;
        ACT/9,RLOGN(1.50,0.808,1);
        FREE,AFSC4/1;
        COLCT(9),TNOW-ATRIB(2),MX4 TIME; ,40/0/1;
Q4      QUE(11),,,,JOIN;
;
MX5     ASSIGN,ATRIB(2)=TNOW,1;
        ACT,,1-XX(5),Q5;
        ACT;
        AWAIT,AFSC5/1;
        ACT/10,RLOGN(0.595,0.320,1);
        FREE,AFSC5/1;
        COLCT(10),TNOW-ATRIB(2),MX5 TIME; ,40/0/1;
Q5      QUE(12),,,,JOIN;
;
JOIN    MATCH,3,Q1/LOOP,Q2,Q3,Q4,Q5;
LOOP    GOON,1;
;       ACT,,ATRIB(2).EQ.TNOW,MX;  recycle
        ACT,,,TURN;
;
        ENDNETWORK;
;
INITIALIZE,0,100000,Y,,Y,N;
FIN;
```

Bibliography

[1] Abell, John B. *The Sortie Generation Model System.* Logistics Management Institute, Washington, DC, September 1981.

[2] Balsamo, S. and I. Mura. "Approximate Response Time Distribution in Fork and Join Systems," *Performance Evaluation Review, 23*: 305-306 (1995).

[3] Banks, Jerry, et al. *Discrete-Event System Simulation.* Upper Saddle River, New Jersey: Prentice Hall, 1996.

[4] Baskett, Forest, et al. "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," *Journal of the A.C.M., 22*: 248-260 (1975).

[5] Boling, Ronald W. and Frederick S. Hillier. *A Fleet Maintenance System Design Model.* Technical Report 214, Stanford, CA: Stanford University, December 1984.

[6] Boyle, Edward et al. *Small Unit Maintenance Manpower Analyses (SUMMA).* Technical Report AFHRL-TR-89-26, Wright-Patterson AFB, OH: Logistics and Human Factors Division, March 1990.

[7] Cronk, Richard and Alan J. Wallace. *LCOM User Manual, Version 94.B.* Air Force Aeronautical Systems Center, Wright-Patterson AFB, OH, March 1994.

[8] Desruelle, P. and H. J. Steudel. "A Queueing Network Model of a Single-Operator Manufacturing Workcell with Machine/Operator Interference," *Management Science, 42*: 576-590 (April 1996).

[9] Dietz, Dennis C. and Richard C. Jenkins. "Analysis of Aircraft Sortie Generation with the Use of a Fork-Join Queueing Network Model," *Naval Research Logistics, 44*: 153-164 (1997).

[10] Dietz, Dennis C. and Matthew Rosenshine. "Optimal Specialization of Maintenance Manpower." to appear in IIE Transactions 97.

[11] Emerson, Donald E. *An Introduction to the TSAR Simulation Program.* Technical Report R-2584-AF, The Rand Corporation, February 1982.

[12] Fisher, R. R., et al. *The Logistics Composite Model: An Overall View.* Technical Report RM-5544-PR, The Rand Corporation, Santa Monica, CA, May 1968.

[13] Gelenbe, E. and G. Pujolle. *Introduction to Queueing Networks.* New York: John Wiley & Sons, 1987.

[14] Jenkins, Richard C. *A Mean Value Analysis Heuristic for Analysis of Aircraft Sortie Generation*, MS thesis, AFIT/GOR/ENS/94M-07. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 1994 .

[15] Kleinrock, Leonard. *Queueing Systems Volume I: Theory.* New York: John Wiley & Sons, 1975.

[16] Kuehn, Paul J. "Approximate Analysis of General Queuing Networks by Decomposition," *IEEE Transactions on Communications, COM-27*: 113-126 (January 1979).

[17] Lavenberg, S. S. and M. Reiser. "Stationary State Probabilities at Arrival Instants for Closed Queueing Networks with Multiple Types of Customers," *Journal of Applied Probability, 17*: 1048-1061 (1980).

[18] Liu, Y. C. and H. G. Perros. "A Decomposition Procedure for the Analysis of a Closed Fork/Join Queueing System," *IEEE Transactions on Computers, 40*: 365-370 (March 1991).

[19] Marshall, K. T. "Some Inequalities in Queuing," *Operations Research, 16*: 651-665 (1968).

[20] Mendenhall, William, et al. *Mathematical Statistics with Applications.* Belmont, California: Duxbury Press, 1990.

[21] Miller, L. W., et al. Dyna-Sim*: A Nonstationary Queuing Simulation with Application to the Automated Test Equipment Problem.* Technical Report N-2087-AF, The Rand Corporation, Santa Monica, CA, Jul 1987.

[22] Presutti, Jr, Victor, et al. *Maintenance Manpower, Spares Costs, and Repair Costs Impacts of Going to Two Levels of Maintenance for the B52H and the KC135.* Technical Report 89-213, Wright-Patterson AFB, OH: Concept Development Division, Headquarters, Air Force Logistics Command, December 1989.

[23] Reiser, M. and S. S. Lavenberg. "Mean-Value Analysis of a Closed Multi-chain Queueing Network," *Journal of the A.C.M., 27*: 313-332 (1980).

[24] Rao, P. Chandrasekhar and Rajan Suri. "Approximate Queueing Network Models for Closed Fabrication/Assembly Systems. Part I: Single Level Systems," *Production and Operations Management, 3*: 244-275 (Fall 1994).

[25] Stecke, Kathryn E. "Review of Operator/Machine Interference Models," *International Journal of Production Research, 23*: 129-151 (January-February 1985).

[26] Suri, Rajan and R. R. Hildebrant. "Modeling Flexible Manufacturing Systems Using Mean-Value Analysis," *Journal of Manufacturing Systems, 3*: 27-38 (1984).

[27] Whitt, Ward. "The Queueing Network Analyzer," *The Bell System Technical Journal, 62*: 2779-2815 (November 1983).

[28] Whitt, Ward. "Performance of the Queueing Network Analyzer," *The Bell System Technical Journal*, 62: 2817-2843 (November 1983).

[29] Woodside, C. Murray. "Throughtput Calculation for Basic Stochastic Rendezvous Networks," *Performance Evaluation, 9*: 143-160 (1989).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 1997	Master's Thesis

4. TITLE AND SUBTITLE
ANALYSIS OF AIRCRAFT SORTIE GENERATION WITH CONCURRENT MAINTENANCE AND GENERAL SERVICE TIMES

5. FUNDING NUMBERS

6. AUTHOR(S)
Daniel V. Hackman, Major, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology/ENS
2750 P Street
Wright-Patterson AFB, Ohio 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/97M-11

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Armstrong Laboratories
Air Force Materiel Command
Brooks Air Force Base, Texas 78235

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The primary objective of this study was to develop an analytical methodology for evaluating an aircraft sortie generation process. The process is modeled as a closed network of general service queues with a fork-join node to model concurrent servicing. The model uses the Mean Value Analysis (MVA) algorithm and general queueing network analysis by decomposition to approximate network performance measures including resource utilization and the overall sortie generation rate.

The results of the study show that the analytical approximation's accuracy decreases as server utilization increases. However, when server utilization is kept in realistic ranges, the approximation is very accurate. When applied to a closed system of single server queues and delay stations, the approximation performs significantly better than a pure MVA-based approach. For closed or capacitated open systems with multiserver queues, the approximation can still be applied to provide upper and lower bounds on system performance.

14. SUBJECT TERMS
Queueing Theory, Queueing Network Model, Fork-Join Queue
Aircraft Sortie Generation, Maintenance Manpower

15. NUMBER OF PAGES
84

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	PR	- Project
G	- Grant	TA	- Task
PE	- Program Element	WU	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.