



**MODELS FOR SORTIE GENERATION WITH
AUTONOMIC LOGISTICS CAPABILITIES**

THESIS

Nicholaus A. Yager, First Lieutenant, USAF

AFIT/GOR/ENS/03-25

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

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense or the United States Government.

AFIT/GOR/ENS/03-25

MODELS FOR SORTIE GENERATION WITH AUTONOMIC LOGISTICS CAPABILITIES

THESIS

Presented to the Faculty
Department of Operational Sciences
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Operations Research

Nicholaus A. Yager, B.S.
First Lieutenant, USAF

March 2003

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT/GOR/ENS/03-25

**MODELS FOR SORTIE GENERATION WITH
AUTONOMIC LOGISTICS CAPABILITIES**

Nicholaus A. Yager, B.S.

First Lieutenant, USAF

Approved:

Dr. Jeffrey P. Kharoufeh
Thesis Advisor

Date

Dr. John O. Miller
Reader

Date

Acknowledgments

I would like to begin by thanking my thesis advisor, Dr. Jeff Kharoufeh, for his effort in helping me to complete this thesis. He pushed me when I needed it, encouraged me when appropriate, and tolerated my seemingly countless idiosyncrasies. Also, I would like to thank Dr. J.O. Miller for reading my thesis and ensuring everything contained within was accurate. I would like to thank Mark Goldschmidt, Frank Erdman, and Greg Dierker (ASC/ENMS) for helping me with the LCOM simulation, which is still a mystery to me. I also give my thanks to Joe Price, Don Hoffman, and Brad Beabout for helping me to understand Air Force maintenance.

In addition, I would be committing a great injustice by not thanking the following people: Brian Oppliger for reminding me there are worse things than writing a thesis; Kris Pruitt and Joe Price for the relaxing and traumatizing rock climbing trips; Chris McGee for bowling excellence (at getting us kicked out); Mo Azar for his Red Foreman imitation; Scott Percival for euchre breaks; and the Pepsi Corporation for Diet Mountain Dew. I must thank my parents for always being behind me and making me who I am. I need to thank my sister for her encouragement. Finally, I truly appreciate my niece for being unbearably cute.

God guides me through this wonderful life,
Over my shoulder through pain and strife,
Drives my happiness and calms my hate,
So I thank Him, 'cause no more thesis is truly great.

Nicholaus A. Yager

March 2003

Table of Contents

	Page
Acknowledgments	iv
List of Figures	vii
List of Tables	viii
Abstract	ix
1. Introduction	1-1
1.1 Background	1-1
1.2 Problem Definition and Methodology	1-3
1.3 Thesis Outline	1-4
2. Review of the Literature	2-1
2.1 Autonomic Logistics System (ALS)	2-1
2.2 Simulation Models for Sortie Generation	2-3
2.3 Analytical Models of Sortie Generation	2-5
2.3.1 Basics of Queueing Networks	2-6
2.3.2 Queueing Network Approach to Sortie Generation	2-10
3. Methodology	3-1
3.1 Review of Existing Model	3-1
3.2 Base Model for Sortie Generation	3-7
3.3 ALS Enhanced Maintenance Operations Model Description	3-8
3.4 Analytical Comparison of the Base and ALS Models	3-10

	Page
4. Numerical Results	4-1
4.1 Formal Description of Comparisons	4-1
5. Conclusions and Future Research	5-1
Bibliography	BIB-1
Appendix A. Base Model Matlab® Codes	A-1
Appendix B. ALS Model Matlab® Code	B-1
Appendix C. LCOM Input Files	C-1

List of Figures

Figure		Page
2.1.	Dietz and Jenkins [6] model for sortie generation.	2-10
3.1.	Dietz and Jenkins [6] model for sortie generation.	3-1
3.2.	Base model for sortie generation.	3-7
3.3.	ALS model for sortie generation.	3-9
3.4.	Differences between base and ALS models.	3-12
4.1.	Sortie generation rate as a function of K	4-3
4.2.	Sortie generation rate as time for troubleshoot decreases in ALS. .	4-8

List of Tables

Table		Page
2.1.	Definition of queueing network terminology.	2-7
2.2.	Description of BCMP nodes.	2-8
4.1.	Base and ALS sortie generation rate (sorties/hr) versus simulation.	4-2
4.2.	Descriptions of various network nodes.	4-2
4.3.	Definitions of output variables.	4-3
4.4.	Base model versus ALS model results.	4-4
4.5.	Base model versus ALS model results.	4-5
4.6.	Sortie generation rate (sortie/hr) as false alarm penalty increases.	4-6
4.7.	Base versus ALS sortie generation rate (sorties/hr) with fewer servers.	4-7
4.8.	Sortie generation rate (sorties/hr) when service time (T) reduced in troubleshoot.	4-7

Abstract

The primary objective of this research is to investigate the impact of an autonomic logistics system (ALS) on the sortie generation process for an individual airbase. As in some prior studies of this process, the methodology used to model the sortie generation process is a queueing network containing fork-join nodes for concurrent maintenance activities. The sortie generation rate is commonly regarded as the primary performance measure of the sortie generation process. This measure coincides with the throughput and is used to compare two models: i) pre-ALS operations and ii) ALS-enhanced airbase operations. Analysis of the models shows that the ALS model yields higher generation rates under a variety of scenarios resulting from the differences in the sortie generation process that are inherent when an ALS is implemented. These results demonstrate that implementation of an ALS will positively impact the sortie generation process by increased sortie generation rates with equivalent or reduced resource levels.

MODELS FOR SORTIE GENERATION WITH AUTONOMIC LOGISTICS CAPABILITIES

1. Introduction

1.1 Background

The United States Air Force is examining the possibility of using autonomic logistics for maintenance of aircraft. Autonomic logistics can be described as a process by which sensors on the aircraft send information to maintenance crews when an aircraft component has failed or is about to fail. This thesis is concerned with assessing the effects of an Autonomic Logistics System (ALS) on the ground operations of a given base. Air Force maintenance currently uses a two level approach to maintenance, one level on the base with the airframes themselves, and a second level called a depot detached from the bases where pieces removed from the airframes on base are sent to be repaired. Despite a possible impact on depot level maintenance, it is believed the greatest impact of the ALS will be on base level maintenance. The standard metric for determining the effectiveness of base level maintenance is the sortie generation rate, or the number of sorties a base can fly in a given time period [6].

The sortie generation process begins with pre-flight activities. Pre-flight consists of actions such as refuelling, munitions uploading, baggage and passenger loading, and the taxi onto the runway. Completion times of these tasks rarely vary from their average, with the exception of taxi which provides for the majority of interest in this activity. The next activity is the sortie. It is here that subsystems on the aircraft fail or malfunction, which will require maintenance for the aircraft to produce a following sortie. Once the aircraft lands, it must go through troubleshooting.

This activity contains a taxi to the aircraft's parking space, munitions download, baggage and passenger disembarkation, and the troubleshooting of the aircraft. The troubleshooting aspect requires maintainers to use diagnostic equipment to determine where the aircraft has experienced a failure, and ordering the appropriate part to fix the system. The final activity before repeating the process is maintenance. It is believed this is of great importance to the sortie generation process as it can cause the longest delays due to service, and the variability has the potential to be enormous. Here maintenance is assumed to encompass all aspects of repairing an aircraft so that it may perform another sortie. However, it is possible to complete multiple maintenance actions concurrently [6].

When autonomic logistics is incorporated into the sortie generation process, the process will change. The majority of the changes will occur in the troubleshoot node. Due to the prognostic capabilities of an ALS, the amount of diagnostic equipment needed to locate failures is greatly reduced. This also means that the amount of time required to complete diagnostics is decreased. Furthermore, due to the prognostic ability of an ALS, parts needed for maintenance are assumed to always be on hand when a maintenance action must be completed. The ALS allows maintainers to have prior knowledge of impending failures, and will have the needed lead-time to order a part that will soon fail. Finally, the ALS may be able to eliminate false alarms that can occur when performing diagnostics on an aircraft. The current assumption is that these differences will lead to an improved sortie generation rate [19].

Due to its great value to a military force, the sortie generation process has been studied in depth [6, 20]. Two main techniques have been used in such efforts. One of these techniques is simulation modeling. This method requires massive amounts of data and a significant length of time to perform a model run. However, simulations are very flexible and allow the user to closely mimic real world conditions. Once the simulation experiment has been conducted, a statistical analysis must be performed to properly interpret the results. A second technique that has been used to study

the sortie generation process is analytical models via queueing networks. These models, while often quite simple, can be used to model a complex system. Queueing network models run very quickly with minimal amounts of data. However, analyzing these networks requires intensive mathematical background. Moreover, such models can be significantly less flexible than simulation models; however, mean performance measures can be computed easily and without replication. With these characteristics in mind, one can see why queueing networks provide an attractive means to study the sortie generation process [6].

1.2 Problem Definition and Methodology

While the sortie generation process has been well studied by Jenkins [6] and Willits [20] using queueing networks and by many others using the Logistics Composite Model (LCOM) simulation, none of these studies have focused on the impact of autonomic logistics. Implementing an ALS on base-level maintenance will alter the activities required for the sortie generation process. While pre-flight and sortie activities will be unchanged, some post-flight activities will not require as much time, and some activities will be eliminated altogether.

In order to provide an understanding of the impact of autonomic logistics on sortie generation rates, the current sortie generation process will be modeled and compared with the proposed ALS process. In order to accomplish this, the proposed ALS must be well understood. Models can then be built combining the knowledge of the ALS with previous analytical models of the sortie generation process. The model results will be compared with those of computer simulation. The analysis of the models will be used to illustrate the effects of ALS on the sortie generation process.

Those currently involved in designing the ALS, such as the designers of the Joint Strike Fighter, believe that implementing autonomic logistics into maintenance

operations will reduce the resources required to maintain the aircraft supported by the ALS. This implies that, with equal resource levels, implementing an ALS will allow maintainers to maintain or exceed the current sortie generation rate. The previously discussed models will show that the sortie generation rate will in fact, be positively impacted by the presence of an ALS [19].

The main contributions of this thesis can be summarized as follows. One model will be created to allow analysis of the current sortie generation process and a second model will be created for analysis of a sortie generation process enhanced by an ALS. Analysis of these models will demonstrate that, by implementing an ALS, an airbase will experience a higher sortie generation rate confirming a key assumption of current ALS literature [19]. The models also provide a convenient means by which to perform a sensitivity analysis on model parameters.

1.3 Thesis Outline

The following chapter will review previous work in the areas of queueing networks, modeling the sortie generation process and autonomic logistics. Chapter 3 reviews a queueing network model for sortie generation and presents a revised version for current operations and a second revision for incorporating the ALS. Chapter 4 contains numerical analysis of the models compared to LCOM output, as well as sensitivity analysis of the two models to compare the results of each. Finally, Chapter 5 will provide conclusions, recommendations, and directions for future research.

2. Review of the Literature

In this thesis, we investigate the sortie generation process and the effect of an Autonomic Logistics System (ALS) upon this process. This will require the review of relevant work to clarify the sortie generation process and how the proposed ALS system will affect it. The following sections will review the ALS system, Logistics Composite Model (LCOM) simulation, and two previous analytical models.

2.1 Autonomic Logistics System (ALS)

This section will review current literature on the ALS. A brief discussion of the Prognostics and Health Management (PHM) and Joint Distributed Information System (JDIS) is followed by a synopsis of the ALS as a whole. Finally, discussion of the anticipated impact of an ALS on the current sortie generation process will be provided.

A key aspect of the ALS is the PHM. The goal of a PHM is to assess the current health of a component and predict when it will experience a failure and require maintenance actions. The ability to collect this data enables the ALS to give an adequate lead-time to order spare parts required to repair the aircraft. In addition, with such information the ALS will greatly reduce the need for diagnostic equipment and personnel and, in some cases, call for the elimination of these tools that are currently vital to maintenance activities [9]. Another improvement to current maintenance activities that will occur with prognostics is the enhanced capability of maintainers to know what personnel, equipment, and spare parts will be necessary to repair an aircraft while the aircraft is still in flight. This allows a maintenance crew to have everything in place to repair the aircraft upon sortie completion, and can give personnel an opportunity to review repair procedures for the aircraft as quickly as possible [19].

The Joint Distributed Information System (JDIS) of the ALS disseminates information gathered from the PHM to the people who need it such as commanders, maintainers and logisticians. The JDIS is the part of the ALS that automates the logistics system. This system is intended to be a single, all-knowing source for logistics concerns, knowing which aircraft need what actions, where spare parts are stored and where they are needed, and which personnel is needed to repair aircraft and where they located [19]. The JDIS automates the entire logistics system by analyzing all of the information and sortie generation rate targets to schedule the ordering of parts, manpower allocation, and maintenance actions [9].

In order to be successful, the ALS requires both the PHM and JDIS. The ALS will be comprised of an aircraft enabled by a PHM system, the JDIS, a responsive logistics infrastructure, and specialized maintainers to repair the aircraft. The following example shows how the four parts work together to enhance the sortie generation process. As an aircraft executes a mission, subsystems experience degradation over time that is detected by the optimally-located sensors. This information is then relayed to the JDIS while the aircraft is still on its mission. The JDIS determines which subsystems require repair and when the action must take place. The JDIS keeps track of impending failures, orders spare parts for upcoming maintenance action, and schedules personnel to repair an aircraft at the time most advantageous to the sortie generation process. To accomplish this, a flexible logistics system must be in place to handle orders, personnel and maintenance activity requests of the JDIS. Finally, an intelligent, well-equipped maintainer is needed to perform the repairs to the aircraft. Even though the ALS automates everything else, personnel are still needed to execute the work or the sortie generation process will collapse [9].

The implementation of an ALS provides advantages outside of the sortie generation process. Some of these include decreased responsibility of maintainers to locate failures, a proactive logistics systems as opposed to the current reactive system, and enhanced capability for predicting the sortie generation rate. However, since these

are not part of the process, they are not able to be modeled, or of concern to this thesis [9].

It is believed that the ALS will alter the current sortie generation process in three major ways: i) completely changing the ordering activity, ii) reduction of diagnostic equipment and time to detect failures, and iii) elimination (or at least reduction) of false alarms. As mentioned, the JDIS will order parts for scheduled maintenance activities to minimize the impact of maintenance actions on the sortie generation process. This means that when a part is needed for a maintenance activity, it will already be in place so that no ordering must occur. Secondly, the prognostic ability of the PHM, in theory, allows the JDIS to know the remaining useful life each subsystem has at a particular point in time. This means the diagnostics aspect of troubleshooting will be greatly reduced as the PHM will alert the JDIS of a failure or impending failure. This will at least reduce the need for diagnostics, and could in some instances eliminate the diagnostic actions altogether. Finally, the sensors of the PHM are installed as to be redundant so that the JDIS knows the actual state of the subsystems of the aircraft. This should ensure that when a system shows a failure, it has actually failed. It must be reiterated that failures will still occur with an ALS in place, but the makeup of the ALS will promptly correct these problems and allow the sortie generation process to function with enhanced performance as compared to the current state [9, 19].

2.2 Simulation Models for Sortie Generation

One method used for modeling real-world systems is discrete-event computer simulation. A stochastic simulation is a computer program used to mimic the processes of a system by simulating each step of the process using input data and internal probability distributions. The United States Air Force uses many simulations to study a variety of aspects unique to its operations, one of which is the sortie generation process. Of the simulations used for the sortie generation process,

Dyna-Sim [13] was designed to show the importance of automatic test equipment in the sortie generation process, the Sortie Generation Model (SGM) [1] was designed to evaluate varying levels of resources on the sortie generation process, and the Logistics Composite Model (LCOM) was designed to estimate manpower requirements for a given sortie generation rate. Throughout the years, LCOM has become the Air Force approved simulation model for estimating the sortie generation rate [4]. In addition, there exist two simulations that model the ALS. Rebulanan [17] modeled the function of the ALS within the Joint Strike Fighter's current proposal of automatic logistics, while Malley [12] examined the PHM component of the ALS. The following discussion will review the operation of LCOM and its use in the United States Air Force. Since LCOM is flexible in nature and is a validated tool of the Air Force, it will be used in this thesis as a benchmark for the queueing network models of Chapter 3.

LCOM is a simulation model used to define manpower requirements for a variety of aircraft currently in the United States Air Force inventory. The model was created in the 1960's by the Air Force Logistics Command and The Rand Corporation in an effort to create a data analysis tool for base-level maintenance operations and the sortie generation process [4]. LCOM has become the standard tool of the Air Force analyst for estimating manpower, spare parts requirements and sortie generation rate. It accomplishes this via a dynamic simulation meaning the simulation makes random draws from a population to determine how the modeled network will behave. LCOM uses these random draws for a variety of purposes, including sortie duration, repair duration, and chance of failure of a subsystem of an aircraft [4]. By running multiple replications of LCOM, a statistical analysis of the results should be performed to bound the actual performance measures of the process being modeled [5].

One of the advantages of LCOM is its extreme flexibility. This allows LCOM to model a vast array of networks and aircraft. However, this advantage can quickly

become disadvantageous. As process complexity increases, more closely reflecting real-world conditions, the amount of necessary input data explodes, making the model tedious. For instance, it has been reported in [5] that intensive LCOM studies can take as long as six months to gather the input data.

In general, the LCOM model runs in the following manner. The analyst inputs an operational requirement on a mission for a certain aircraft. This requirement is entered into the model as a scenario in which the specified aircraft will participate. Each time a sortie is demanded by the scenario, the simulation calls a suitable and available aircraft to perform the sortie. When the aircraft returns, it must follow a set of rules corresponding to base-level maintenance operations. The analyst running LCOM is responsible for entering the scenario, the number of aircraft, and the mean and variance of repair times, sortie times, and the probability that aircraft subsystems will experience failures [5].

Though LCOM is designed to set manpower requirements, it is also a useful tool for determining sortie generation rates for a given scenario. LCOM estimates manpower requirements by having the user input a desired sortie generation rate, and calculates the appropriate manpower level required to achieve that rate [4]. As such, one of the outputs of LCOM is the sortie generation rate. Sensitivity analysis on the input data provides commanders and decision makers a good idea of how the sortie generation process will react under various conditions which aides in determining how to conduct operations. Also, this makes LCOM a useful tool for benchmarking analytical models [5].

2.3 Analytical Models of Sortie Generation

This section reviews an alternative to simulation modeling in order to examine a real-world system, namely analytical queueing network models. The sortie generation process is well suited to be modeled as a queueing network. The mathematics

of queueing networks will be reviewed followed by a review of the work of Dietz and Jenkins [6], a pertinent queueing network model for the sortie generation process.

A queueing network is a connected graph in which nodes represent single station queues and arcs connect nodes between which entities flow. Individual queueing stations have a resource level, service time distribution, and capacity. These characteristics make queues a viable means for modeling each step of the sortie generation process. As an example of how the activities may be modeled as queues, we look at the pre-flight activity. Clearly, this step requires personnel, which are the resources, to accomplish the tasks. This work can be completed in a random amount of time that can be closely approximated by a service time distribution. The physical waiting area for aircraft corresponds to the capacity of the queueing station. The other activities of the sortie generation process follow a similar setup, which shows the process can be modeled as a queueing network.

2.3.1 Basics of Queueing Networks

This subsection reviews the basic concepts of analytical queueing networks and the mathematics used to evaluate the models for their steady-state performance. The properties of queueing networks easily lend themselves to modeling real-world process such as factory production lines, communication systems, and the sortie generation process. Chapter 3 of this thesis presents the sortie generation process modeled as a queueing network. Using an approximation algorithm to find the steady-state performance measures is appropriate due to the complexity of this model. The approximation algorithm used in Chapter 3 is known as the Mean Value Analysis (MVA) algorithm, and is reviewed at the end of this section.

Let $Z_i(t)$ be an integer-valued random variable denoting the current number of jobs at node i at time t , for $i = 1, 2, \dots, N$ and $t \geq 0$. The random variable $Z_i(t)$ is defined on the sample space $S = \{0, 1, 2, \dots\}$. Let $\{X(t) : t \geq 0\}$ be the current state of the system of nodes, $X(t) = (Z_1(t), Z_2(t), \dots, Z_N(t))$ where N is the number

Table 2.1 Definition of queueing network terminology.

Symbol	Description
K	Number of jobs in system
$\mu_i(k)$	Service rate at node i when there are k jobs present
R_i	Mean response time at node i
λ_i	Throughput
Q_i	Queue length
U_i	Server utilization
s_i	Mean service time at node i
q_i	Probability of repair needed at node i
r_i	Number of servers at node i
$P_i(k K)$	Probability of k customers at node i given K total aircraft
$\lambda_i(K)$	Throughput at node i when K jobs are in the system
$Q_i(K)$	Queue length at node i when K jobs are in the system
$R_i(K)$	Response time at node i when K jobs are in the system
$CT_1(K)$	Average cycle time at node 1 when K jobs are in the system
P	Matrix of routing probabilities
v_i/v_1	Mean number of visits to station i for every visit to node 1

of nodes in the system. The steady-state probability of having k jobs present at any node i is defined as

$$\pi_i(k) := \lim_{t \rightarrow \infty} P\{Z_i(t) = k\}. \quad (2.1)$$

Table 2.1 presents a list of terminology used throughout the discussion of queueing networks and the MVA algorithm.

The steady-state probability of k_i customers at node i represented as $\pi_i(k_i)$ can be found using rudimentary queueing theory. When the steady-state probability for a queueing network can be found from the steady-state probabilities of each station of the network, the network is known as a product-form network. This term was first introduced by Jackson [10] for a very basic network of exponentially distributed service times for each single server node in the system, now known as a Jackson network. In that case, the steady-state probability is given by

$$\pi(k_1, k_2, \dots, k_N) = \pi_1(k_1) \cdot \pi_2(k_2) \cdot \dots \cdot \pi_N(k_N). \quad (2.2)$$

Table 2.2 Description of BCMP nodes.

Type Number	Node type and service discipline
Type 1	M/M/m-FCFS (first come first served)
Type 2	M/G/1-PS (processor sharing)
Type 3	M/G/ ∞ (infinite server)
Type 4	M/G/1-LCFS PR (last come first serve, priority)

This is the simplest product-form network, where the normalization factor for each station is simply 1. For all networks that are not Jackson networks, the marginal probabilities must be multiplied by a normalization constant, not 1, to yield the steady-state probability of the entire network. Subsequent work by Baskett, *et al.* [2] extended product-form networks to a large number of networks including open, closed and mixed networks. Table 2.2 shows the four basic types of nodes for the so-called BCMP networks that will yield a product form solution. Product-form networks are extremely valuable to queueing networks by providing simplistic computation of the network steady-state probabilities using the marginal steady-state probabilities corresponding to each node.

Mean value analysis (MVA) was first introduced by Reiser and Lavenberg [18] to determine performance measures of a closed queueing network. This iterative approach was developed to avoid calculating the normalization constant, which is often very difficult and costly in both time and computing power. As the name suggests, this algorithm simplifies calculations by settling for mean values for random quantities in lieu of the probability distribution. The MVA for closed queueing networks is based on the arrival theorem and Little's Law. The arrival theorem states that in a closed network, a job arriving to node i with k jobs in the system sees the same probability mass function as it would at the same node as it would if there are $k - 1$ jobs in the system. Little's Law relates the basic performance measures to each other and is central to queueing theory. Little's Law states that

$$Q = \lambda R, \quad (2.3)$$

where Q is the mean number of jobs in system, λ is the system throughput, and R is the mean response time. This holds for each node and the system as a whole. The MVA algorithm is now presented for a single class closed network.

The first step of MVA is to initialize the terms,

$$Q_i(0) = 0, \quad \pi_i(0|0) = 1, \quad \pi_i(j|0) = 0.$$

The next step is to calculate the mean response time of each node, T_i . The following equation demonstrates how to compute the mean response time given a specific BCMP type of node, as defined in Table 2.2.

$$R_i(k) = \begin{cases} \mu_i^{-1} (1 + Q_i(k-1)) & \text{Type 1,2,4} \\ (\mu_i m_i)^{-1} \left(1 + Q_i(k-1) + \sum_{j=0}^{m_i-2} \frac{(m_i-j-1)}{\pi_i(j|k-1)^{-1}} \right) & \text{Type 1} \\ \mu_i^{-1} & \text{Type 3} \end{cases} \quad (2.4)$$

Using $T_i(k)$, the system throughput can now be calculated as such,

$$\lambda(k) = \frac{k}{\sum_{i=1}^{m_i-2} v_i Q_i(k)}. \quad (2.5)$$

The throughput is now used to calculate $Q_i(k)$,

$$Q_i(k) = v_i \lambda(k) R_i(k). \quad (2.6)$$

The MVA algorithm is executed by initializing the index k to the value 1 and then computing the results for Equations (2.4), (2.5), (2.6). This step is reiterated for $k = 2, 3, \dots, K$ to find the performance measures for the desired total number of jobs in the system [3].

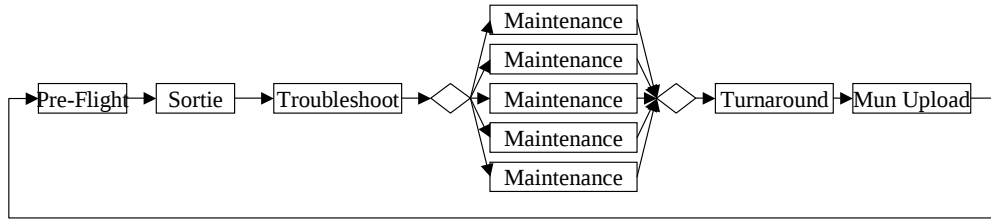


Figure 2.1 Dietz and Jenkins [6] model for sortie generation.

2.3.2 Queueing Network Approach to Sortie Generation

Dietz and Jenkins [6] proposed an analytical model to calculate performance measures of the sortie generation process. This work was later generalized by Dietz and Hackman [8]. The model developed by Dietz and Jenkins [6] is simplistic yet captures many of the important features of the sortie generation process. Since queueing networks provide quick results with minimal input, they are well suited for analyzing the affects of adding an ALS to base operations. The methods reviewed here provide a foundation for the models developed in this thesis.

Dietz and Jenkins [6] present an analytical method to modeling the sortie generation process via a closed queueing network. Previous analytical models had failed to properly account for maintenance operations, which are often done simultaneously in a real world environment. Their work models the maintenance activity using a fork-join node. A fork-join node is a queueing station where, upon arrival to the node, a customer is decomposed into in J distinct and identical clones and those clones begin service along the J paths concurrently. Dietz and Jenkins [6] also present a method that enhances previous work on fork-join nodes by allowing for probabilistic branching on the J paths. This allows an analyst to assign a probability that a clone will require service on each path. Their model, as shown in Figure 2.1, graphically depicts the sortie generation process, including the fork-join node for maintenance.

This closed queueing network model consists of six nodes: pre-flight, sortie, troubleshoot, a fork-join node, turnaround, and munitions upload. The fork-join node represents five critical systems of an aircraft that need maintenance. Each station is assumed to have an exponentially distributed service time. This distribution is used to ensure tractability of the queueing network. This restriction is not as constraining as it appears, as the performance measures of closed queueing networks are frequently unaffected by variances and higher-order moments [6]. In addition, the work of Dietz and Hackman [8] extended this network by relaxing the assumption of exponential service times. The mathematics of the network and MVA algorithm employed to determine the performance measures lays the foundation for the work of this thesis, and is presented in full detail in Section 3.1.

Dietz and Jenkins [6] present numerical comparisons for their modified MVA algorithm and a simulation of the same model. These results were presented for $K = 10, 30, 50, 70$ aircraft in the system. The response time $R_i(K)$, throughput $\lambda_i(K)$, queue length $Q_i(K)$, and utilization $U_i(K)$ were the main measures. Node 2 is the sortie node, and as such the throughput is equivalent to the sortie generation rate. The largest error percentage for the entire network as compared with computer simulation is less than 2.4%. With these excellent results, this model makes an intelligent choice for the foundation of a high-level examination of the sortie generation process, or possibly other similar processes requiring concurrent service.

Chapter 3 will present the methodology of Deitz and Jenkins [6] and the means by which their model is extended for the models in this thesis. Their model will first be modified to more closely reflect current operations. This model will then be extended to analyze the sortie generation process in the presence of an autonomic logistics system. Both models will be analyzed and compared to LCOM simulations for the purpose of validation.

3. Methodology

3.1 *Review of Existing Model*

This section will review the model and results developed by Dietz and Jenkins [6]. First a brief review of the model will be outlined. Then the mathematics of the queueing network and the results will be presented. This model builds the foundation for the two new models developed in this thesis whose details will be presented in Section 3.2.

As mentioned in Section 2.3.2, the approach of Deitz and Jenkins [6] to model the sortie generation process is that of queueing networks. This closed queueing network model consists of six nodes: pre-flight, sortie, troubleshoot, a fork-join node, turnaround, and munitions upload. The fork-join node consists of five substations that represent five critical systems of an aircraft that needs maintenance. Each station is assumed to have an exponentially distributed service time. Figure 3.1 gives a graphical depiction of the model.

The model was analyzed using the MVA heuristic developed by Rao and Suri [16] to deal with closed queueing networks with fork-join nodes, and enhanced to handle fork-join nodes with probabilistic branching by Jenkins [11]. While MVA will produce exact results for product-form networks, the heuristic will give only approximate results for any network with more than one customer in system. The heuristic will be used here since this network contains a fork-join node, and thus

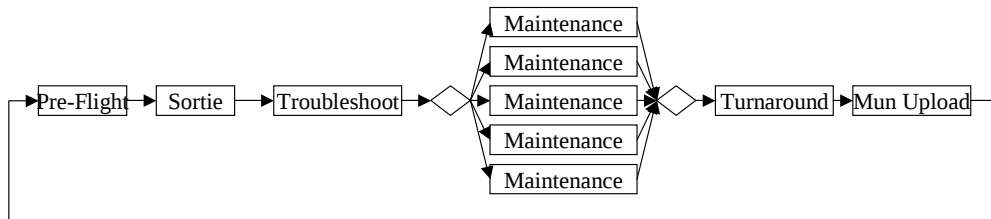


Figure 3.1 Dietz and Jenkins [6] model for sortie generation.

will not have a product-form solution. The first step in analyzing the network is to present the MVA algorithm.

Let $Z_i(t)$ be an integer-valued random variable denoting the current number of jobs at node i at time t , for $i = 1, 2, \dots, N$ and $t \geq 0$. $Z_i(t)$ is defined on the sample space $S = \{0, 1, 2, \dots\}$. Let $X(t)$ be the current state of the system of nodes, $X(t) = (Z_1(t), Z_2(t), \dots, Z_N(t))$ where N is the number of nodes in the system. The steady-state probability at any node i is defined as

$$\pi_i(k) := \lim_{t \rightarrow \infty} P\{Z_i(t) = k\}. \quad (3.1)$$

The MVA algorithm is a method used to calculate the $\pi_i(k)$ values for a given network. This allows for the computation of the mean performance measures for a closed queueing network. This algorithm depends on two major theorems: the arrival theorem and Little's Law. The arrival theorem states that in a closed network, a job arriving to node i with k jobs in the system sees the same probability mass function as it would at the same node if there are $k - 1$ jobs in the system. Little's Law relates the basic performance measures to each other and is central to queueing theory. Little's Law states that

$$Q = \lambda R, \quad (3.2)$$

where Q is the mean number of jobs in system, λ is the system throughput, and R is the mean response time. This holds for each node and the system as a whole.

The arrival theorem is now extended to the marginal local balance theorem. This states

$$\mu_i(k)P_i(k|K) = \lambda_i(K)P_i(k-1|K-1) \quad (3.3)$$

where $\mu_i(k)$ is the state-dependent service rate with k jobs at node i , $\lambda_i(K)$ is the arrival rate with K jobs at node i , and $P_i(k|K)$ is the probability of having k jobs at node i given there are K jobs in system. Using Little's Law along with the marginal

local balance theorem then yields the following equations for mean response time and queue length

$$R_i(K) = \sum_{k=1}^K \frac{k}{\mu_i(k)} P_i(k-1|K-1) \quad (3.4)$$

$$Q_i(K) = \sum_{K=1}^K \frac{k\lambda_i(K)}{\mu_i(k)} P_i(k-1|K-1). \quad (3.5)$$

The visit ratios to each node are now calculated by solving the system of equations resulting from $\vec{v}\mathbf{P} = \vec{v}$, and arbitrarily setting $v_1 = 1$. The visit ratios allow us to calculate the cycle time for a customer at node i for all nodes in the network.

$$CT_i(K) = \sum_{i=1}^M \frac{v_i R_i(K)}{v_1} \quad (3.6)$$

The cycle time calculations allow us to compute $\lambda_i(K)$ as

$$\lambda_i(K) = \frac{K v_i}{CT_i(K) v_1}. \quad (3.7)$$

With $\lambda_i(K)$ now calculated, the distribution of customers may now be calculated for $k+1$ jobs in the system. The equations are easily derived from the marginal local balance theorem.

$$P_i(k|K) = \frac{\lambda_i(k) P_i(k-1|K-1)}{\mu_i(k)} \quad 0 < n \leq K \quad (3.8)$$

$$P_i(0|K) = 1 - \sum_{k=1}^K P_i(k|K) \quad (3.9)$$

MVA is a recursive algorithm that begins with solving for the visit ratios and the following boundary conditions

$$P_i(0|0) = 1 \quad (3.10)$$

$$P_i(k|0) = 0, \quad k > 0. \quad (3.11)$$

Using the boundary conditions allow us to make calculations for $k = 1$ of the previous equations in the following order: $R_i(k)$, $CT_i(k)$, $\lambda_i(k)$, $P_i(k|K)$, $P_i(0|k)$, Equations (3.4), (3.6), (3.7), (3.8), (3.9). This order is then reiterated for $k = 2, 3, \dots, K$.

An MVA heuristic was developed by Rao and Suri [16] to handle the case when a network contains a fork-join node. A fork-join node allows for a customer in the system to receive service simultaneously for different activities. When a customer enters the fork, it is cloned and sent down the different paths of the node, and thus to the substations on each path. The customer departs the fork-join node only when all clones have completed service and meet at the join node. The heuristic of Rao and Suri [16] does not compute an exact solution as networks containing fork-join nodes do not have product-form solutions. However, the heuristic has been shown to approximate the solution very well [16].

The MVA heuristic requires two major assumptions, the previously presented arrival theorem and that the response time for each substation of the fork-join node is exponential and independent of all other substations of the fork-join node. For the purpose of the following discussion, all substations are assumed to have one server. This assumption is later relaxed.

The first step in analyzing the fork-join node is to find the mean response time for each substation. Similar to finding the response time in the MVA algorithm, the arrival theorem allows us to find the mean response time by the equation

$$R_j(K) = \sum_{k=1}^K \frac{k}{\mu_k(k)} P_k(k-1|K-1) \quad (3.12)$$

where $j = 1, 2, \dots, J$ and J is the number of paths in the fork-join node. Using $R_k(K)$ we can determine the mean holding time. For simplicity, let

$$\theta_j(K) = 1/R_j(K). \quad (3.13)$$

It is easily shown that a clone at substation j has a longer response time than all of the other clones in the fork-join node with probability

$$p_j(k) = P\{t_j(k) > \max[t_1(k), t_2(k), \dots, t_{j-1}(k)]\} \quad (3.14)$$

where t_j is the random time to complete service of a clone at substation j . Dietz and Jenkins [6] use the definition of a cumulative distribution function and the exponential service times of each substation to show that

$$\begin{aligned} F(t) &= \prod_{j \in K} P\{t_j(n) \leq t\} \\ &= \prod_{j \in K} (1 - \exp\{-\theta_j(k)t\}). \end{aligned} \quad (3.15)$$

Since the holding time of a customer at the fork-join node is the maximum time to completion of the substations, the expected value of this random variable is desired. The mathematical expectation of a non-negative continuous random variable X with cdf $F(\cdot)$ is given by

$$E[X] = \int_0^\infty \{1 - F(t)\} dt, \quad (3.16)$$

therefore

$$E[\max_{j \in J} \{t_j(n)\}] = \int_0^\infty \{1 - F(t)\} dt. \quad (3.17)$$

This equation can be simplified to (cf. Dietz and Jenkins [6])

$$\begin{aligned} E[\max_{j \in J} \{t_j(k)\}] &= \\ &= \sum_{j \in J} \frac{1}{\theta_j(k)} - \sum_{j \in J} \sum_{l > j \in J} \frac{1}{\theta_j(k) + \theta_l(k)} + \dots + (-1)^{J+1} \frac{1}{\sum_{m \in J} \theta_m(k)}. \end{aligned} \quad (3.18)$$

The cycle time for the system must be recalculated using the following equation when Y is the set of non-fork-join nodes and Z is the set of fork-join nodes

$$CT(K) = \sum_{i \in Y} v_i R_i(K) + \sum_{i \in Z} v_i E[\max_{j \in J} \{t_j(k)\}]. \quad (3.19)$$

This allows for the network to be analyzed as in the previous section. The following equations apply to performance measures for the fork-join substations but must be calculated for each iteration of the MVA heuristic algorithm.

$$\lambda_j(K) = \frac{K v_j}{CT_i(K) v_j} \quad (3.20)$$

$$Q_j(K) = R_j(K) \lambda_j(K) \quad (3.21)$$

$$U_j(K) = s_j \lambda_j(K) \quad (3.22)$$

$$P_j(0|K) = 1 - \sum_{k=1}^K P_j(k|K) \quad (3.23)$$

$$P_j(k|K) = \frac{\lambda_j(k) P_j(k-1|K-1)}{\mu_j(k)} \quad 0 < k \leq K \quad (3.24)$$

The results of the previous section are now extended to allow for probabilistic branching on the different paths of the fork-join node. This allows the model to accurately depict the real world conditions of base-level maintenance, as an aircraft does not always require maintenance on each critical system.

Let Ω be the set of possible combinations of paths of the fork-join node, including the occurrence of no paths (i.e., $\emptyset$). Also, let S be one subset of Ω and let q_i be the probability that path i will be visited. Thus, the probability of each S occurring can be shown as

$$\pi(S) = \prod_{k \in S} q_k \prod_{k \notin S} (1 - q_k) \quad (3.25)$$

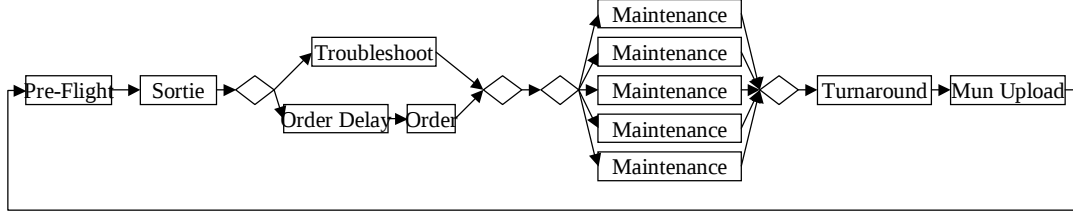


Figure 3.2 Base model for sortie generation.

when the probability of visiting a path is independent of visiting the other paths. Now, $E[\max_{j \in S}\{t_j(k)\}]$ is calculated for each S in Ω . The final changes to the heuristic are new cycle time and throughput calculations for fork-join substations,

$$CT(K) = \sum_{i \in Y} v_i R_i(K) + \sum_{i \in Z} v_i \sum_{S \in \Omega} E[\max_{j \in S}\{t_j(K)\}]. \quad (3.26)$$

$$\lambda_k(N) = \frac{K v_j q_j}{CT_i(K) v_j} \quad (3.27)$$

3.2 Base Model for Sortie Generation

The first step in examining the impact of an ALS on the sortie generation process is to determine a model that depicts the current sortie generation process. The network used for the purpose of examining performance measures as they exist in the current sortie generation process is a modified version of the network reviewed in the previous section. This new network, henceforth referred to as the *base model*, consists of the following nodes: pre-flight, sortie, troubleshoot, order delay, order, maintenance, turnaround, and munitions upload. The maintenance node is a fork-join node to allow simultaneous service repairs, as maintenance operations are done in the real world. Figure 3.2 graphically depicts the model.

The base model has a station for ordering the parts needed for maintenance. This is done simply by adding a station the aircraft must visit before visiting the maintenance fork-join node. The ordering action is approximated by an exponential

infinite server node as the exponential distribution is a suitable approximation for ordering actions and orders for multiple jobs can be accomplished at the same time [6]. To further expand upon the base model, the ordering and troubleshoot nodes are converted to a fork-join nodes to allow for analysis of ordering during the troubleshooting phase of the sortie generation process. An order delay node is added before the order node so that orders do not begin at the exact same time as the troubleshoot activity.

The base model also differs from that of Dietz and Jenkins [6] in the maintenance fork-join node. In this base model, a mean response time is added to the node in the case where no paths (i.e. $\emptyset$) are taken. The case when an aircraft visits the maintenance node but no paths are actually visited equates to a false alarm. This false alarm is from the diagnostic equipment used during the troubleshooting phase to determine which subsystem or subsystems have experienced a failure. The base model incurs a time penalty when a false alarm occurs. This penalty is added because a false alarm requires maintainers to work on the aircraft to actually determine that a false alarm has occurred as opposed to an actual failure. These alterations to the Dietz and Jenkins [6] model are necessary for this thesis. Their model makes the assumption that any part needed for a maintenance action is already in place for the maintainers. This is not a realistic assumption in the current sortie generation process, as parts must be ordered when a failure is determined during the troubleshooting phase. There are instances when the parts are on hand, but this is not always the case. The model also assumed that no time elapses to determine a false alarm has occurred. This assumption is inaccurate as maintainers must perform work before finding out that the detected failure is actually a false alarm.

3.3 ALS Enhanced Maintenance Operations Model Description

The main purpose of this thesis is to examine the effects of the ALS on the sortie generation process. As discussed in Section 2.1, the ALS equips each aircraft

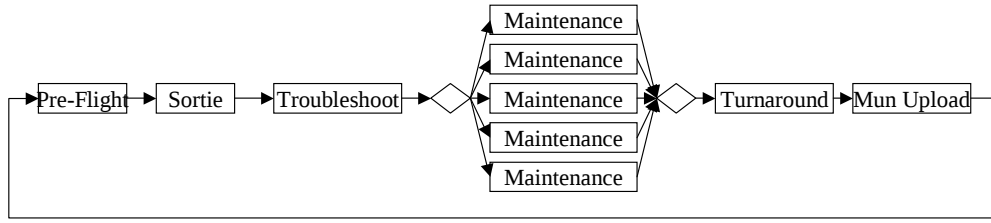


Figure 3.3 ALS model for sortie generation.

with numerous sensors that detect the current state of critical systems of the aircraft. The sensors of the plane are programmed to send a signal to ground maintenance informing them of systems with impending failures. This allows for the parts necessary for maintenance to be ordered well before the actual maintenance action must occur. Therefore, a new model is needed to describe the ALS sortie generation process. This model is now presented in full detail.

The ALS model consists of six nodes: pre-flight, sortie, troubleshoot, a fork-join node, turnaround, and munitions upload. The fork-join node consists of five substations that denote five critical systems of an aircraft that needs maintenance. Figure 3.3 shows the model.

At first glance, the ALS and base model appear to be extremely similar. This is in large part due to rigid aspects of the sortie generation process. In fact, the pre-flight, sortie, turnaround, and munitions upload nodes are unchanged from the base model. The ALS will not impact how these activities occur within the sortie generation process.

One of the noticeable differences is the absence of the order delay and order nodes. As previously discussed, the ALS allows maintainers to know of impending failures to critical systems. The ALS uses the JDIS, discussed in Section 2.1, to automatically order the parts required for maintenance action with the appropriate lead-time so that all parts are on hand when the action must take place. This means

that maintainers will no longer need to order parts to complete maintenance on an aircraft, so both of the nodes are not included in the ALS model.

In addition, the response time for the troubleshoot node is decreased from the base model. This is done to account for the decreased amount of time required to perform diagnostics when an ALS is in place. Much of the diagnostic testing that occurs in the current sortie generation process is automated by the prognostic sensors included in the ALS. This reduces the amount of resources and time required to complete the troubleshooting activities. Due to a reduced service time, the mean time for the troubleshoot node is decreased. Since it is unknown how drastically this time will be reduced, numerical comparisons are presented in Chapter 4 which vary the percentage of time that the ALS decreases troubleshooting activities. This analysis will include the case when the service time is zero, as would be the case for an ALS that has perfect prognostics.

Finally, the maintenance fork-join node has a minor difference in the ALS model as opposed to the base model. The sensors of the ALS and moreover the PHM, previously discussed in Section 2.1, may be able to eliminate false alarms [9]. This allows the penalty incurred when a false alarm occurs to be zero in the ALS model. The penalty is set to zero as the probabilistic branching method for fork-join nodes must include the possibility of no paths being visited. An analysis of this penalty is presented in Chapter 4.

3.4 Analytical Comparison of the Base and ALS Models

This section provides a simple analytical proof that the sortie generation process will be positively impacted when an ALS is incorporated. Doing so requires close examination of the differences between the base model and the ALS model. The differences in modeling are considered in the context of the MVA heuristic to

determine the analytical difference between the two models. This difference will show the advantage of incorporating autonomic logistics in airbase operations.

The base model and ALS model differ in the manner by which ordering is accomplished. The base model requires the customer to go through the pre-flight node, then the sortie node, then enters a fork-join node where it is processed simultaneously in the troubleshoot and order nodes. The ALS model does not contain an order node. This is one of the major advantages of an ALS. With the prognostic capability of autonomic logistics, maintainers always have an appropriate lead-time to have needed spare parts on hand. This provides an advantage to completing the sortie generation process since it eliminates a node that the aircraft would have to visit without an ALS present.

The models also differ in two other minor aspects; the troubleshoot and maintenance nodes. Since the ALS relies on prognostic equipment to perform the majority of system checks, it is anticipated that the amount of time spent troubleshooting the aircraft on the ground will be drastically reduced. Also, the ALS is intended to eliminate false alarms caused by identifying a needed repair that does not exist [19]. For this reason, a penalty is added to the maintenance fork-join node when a false alarm occurs. This penalty mimics the fact that time is spent determining that a false alarm has occurred as opposed to an actual failure. The penalty is zero for the ALS model, since theoretically a false alarm will never occur.

With the remainder of the network being the same, analysis of the difference between the networks reduces to examination of these four nodes: troubleshoot, order delay, order, and maintenance. Figure 3.4 shows how these nodes are configured in the two models.

Finding the response time of each node is the first step in performing the MVA algorithm. The response time for a node or substation is given by

$$R_i(k) = \frac{k}{\mu_i(k)} p_i(k|K). \quad (3.28)$$

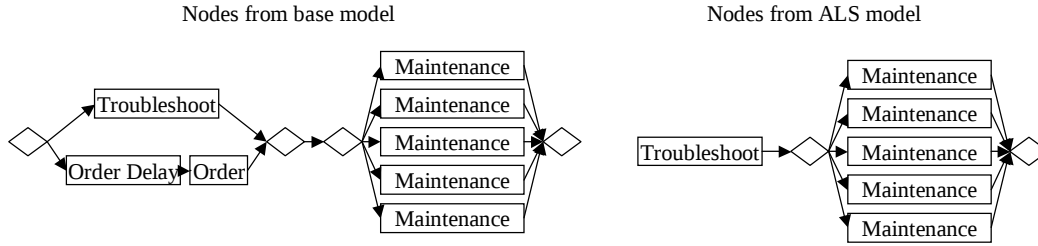


Figure 3.4 Differences between base and ALS models.

This is suitable for regular nodes, but more work is required for finding the total response time for a fork-join node. Letting $\theta_i(k) = 1/R_i(k)$, the equation is

$$E[\max_{j \in J} \{t_j(k)\}] = \sum_{j \in J} \frac{1}{\theta_j(k)} - \sum_{j \in J} \sum_{l > j \in J} \frac{1}{\theta_j(k) + \theta_l(k)} + \cdots + (-1)^{J+1} \frac{1}{\sum_{m \in J} \theta_m(k)}. \quad (3.29)$$

First, we will examine the difference in the maintenance node. As previously described, the difference in the maintenance nodes of the two models lies in the penalty, defined as $R_p(n)$, of the base model. The models for this thesis have five paths in the maintenance fork-join node. This means there are $2^5 = 32$ distinct paths that are possible when an aircraft arrives at this node. Both models compute response times exactly the same for thirty-one of these combinations. The lone exception is when no paths are taken. For the base model, this is the event of a false alarm, where diagnostic equipment reported a failure where one does not exist. The ALS model assumes that no false alarms occur thanks to its prognostic approach. To account for this difference, $R_p(n) = c$ where c is set to 1 (hr) account for time spent working maintenance when no maintenance occurs for the base model. For the ALS model, $c = 0$ since it is assumed that no false alarms occur. Anytime the base model has a non-zero c value, this will obviously result in a higher response time for the base model. Now we examine the other three nodes that vary between the two models.

As seen in Figure 3.4, the fork-join node in the base model containing the troubleshoot, order delay, and order nodes is replaced in the ALS model by a troubleshoot node. The reason for this is that the ALS gives maintainers prior knowledge of impending maintenance actions, and therefore have all parts on hand when maintenance must be performed on an aircraft. Another bonus of the ALS is that the troubleshoot response time is reduced since much of the base model diagnostic work is replaced by the prognostic capabilities of the ALS.

For the comparison of the nodes differing between the models, assume the response time for the troubleshoot node is the same for both models. Let R_1 be the mean response time for the troubleshoot node, R_2 be the mean response time for the order delay node, and R_3 be the mean response time for the order node. This allows us to represent the total response time for the set of nodes being examined for the base model R_B , as

$$R_B = R_1 + \left[R_2 + R_3 - \frac{1}{\frac{1}{R_2} + \frac{1}{R_3}} \right], \quad (3.30)$$

and for the ALS model as

$$R_A = R_1. \quad (3.31)$$

The conjecture is that the total response time for the set of nodes in ALS model will be smaller than that for the base model.

Proposition 3.1 *The ALS-enhanced sortie generation process will always yield a higher sortie generation rate than a non-ALS-enhanced system, i.e., $\lambda_A > \lambda_B$*

Proof. We seek to show that $\lambda_A > \lambda_B$ will always be true. We start by showing $R_B > R_A$.

$$\begin{aligned}
R_2 + R_3 - \frac{1}{\frac{1}{R_1} + \frac{1}{R_2+R_3}} &= R_2 + R_3 - \frac{R_1(R_2 + R_3)}{R_1 + R_2 + R_3} \\
&= \frac{(R_1 + R_2 + R_3)^2 - R_1(R_2 + R_3)}{R_1 + R_2 + R_3} \\
&= \frac{R_1^2 + 2R_1(R_2 + R_3) + (R_2 + R_3)^2 - R_1(R_2 + R_3)}{R_1 + R_2 + R_3} \\
&= \frac{R_1^2 + R_1(R_2 + R_3) + (R_2 + R_3)^2}{R_1 + R_2 + R_3} \\
&> 0,
\end{aligned}$$

which implies

$$\begin{aligned}
R_B &= R_1 + R_2 + R_3 - \frac{1}{\frac{1}{R_1} + \frac{1}{R_2+R_3}} \\
&> R_1 \\
&= R_A.
\end{aligned} \tag{3.32}$$

Equation (3.32) clearly shows that for non-negative mean response times for the order delay and order nodes, the base model will yield higher response times. Since all mean response times will be greater than zero, the ALS model will have a lower response time for the set of nodes. Therefore, both differences between the models result in lower mean response times for the nodes in question.

Now it is necessary to examine the impact on the remainder of the network. Let $v_r R_r$ be the response times multiplied by their respective visit ratios for all other nodes in the system, and let $v_A R_A$ be this calculation for the ALS node set previously examined and $v_B R_B$ for the calculation for the base model node set. The cycle times

for the models is then

$$CT_B = v_r R_r + v_B R_B \quad (3.33)$$

$$CT_A = v_r R_r + v_A R_A \quad (3.34)$$

Since $v_B = v_A$ we know $CT_A < CT_B$ since $R_A < R_B$. Examining the throughput equations, we see that

$$\lambda_B = \frac{K v_B}{CT_B} \quad (3.35)$$

$$\lambda_A = \frac{K v_A}{CT_A} \quad (3.36)$$

which implies

$$\lambda_A > \lambda_B. \quad (3.37)$$

■

The value λ_A is the throughput of the nodes for an ALS-enhanced sortie generation process. The sortie generation rate is a measure of how many sorties can be flown during a specific time period. The throughput is a measure of how many jobs pass through a node of the network. Thus, the throughput of the sortie node is equivalent to the the sortie generation rate. This means the sortie generation rate for a base that uses the ALS will be higher than that for the same base without the ALS. This result is significant for autonomic logistics as it proves the underlying assumption that an ALS will positively impact the sortie generation rate. A main goal of implementing autonomic logistics is to provide commanders with a tool so that they can reach desired sortie generation rates. Current sortie generation rates are heavily reliant on stockpiled spare parts, skilled maintainers, and the quantity of maintenance personnel and diagnostic equipment. The ALS eliminates the need to stockpile spare parts and may reduce resource level requirements. It eliminates the need to stockpile parts with its built-in ordering system to provide the needed lead time to have only those parts necessary for maintenance actions to be completed in

the near future. The diagnostic equipment requirements are reduced by the prognostic sensors added to the plane as part of the ALS. While the result shows that sustaining the level of resources will induce a higher sortie generation rate with an ALS, it also implies that the sortie generation rate of current operations is attainable with fewer resources. This will be demonstrated numerically in the next chapter.

4. Numerical Results

4.1 Formal Description of Comparisons

The purpose of this chapter is to numerically compare the analytical ALS and Base model, as well as the LCOM simulations for each of the models. These numerical results provide a better understanding of the impact of ALS on the sortie generation process. The performance measure that was examined is the throughput of the sortie node which corresponds to the sortie generation rate measured in sorties flown per hour (sorties/hr). The comparisons show how the sortie generation rate behaves for both models when we vary the total number of aircraft in system, the number of servers in the maintenance nodes, and the mean service times of the troubleshoot and order nodes.

The ALS and base models were coded in MATLAB[®] using the MVA algorithm of Dietz and Jenkins [6] presented in Section 3.1. The LCOM simulations were conducted using WinLCOM 2000.D. These simulations were designed to operate in the same manner as the analytical models, using exponential distributions, fly-when-ready schedules, and other similar aspects from the analytical models. Sample input files of LCOM and the MATLAB[®] codes may be found in the appendix of this thesis.

An initial run of each model was made with fixed parameter values, with mean service times given in hours. These results were compared directly with the LCOM simulation runs to assess the adequacy of the queueing network models. The values used in this initial case are the control values for the remainder of the comparisons so that the sensitivity of the models to perturbations made be assessed.

Table 4.1 shows how the sortie generation rates of the base model and ALS model compared with the LCOM results. The percentage error ranges from 0.01% to 6.3%. The results indicate that the analytical models closely approximate the results of the simulation run in LCOM.

Table 4.1 Base and ALS sortie generation rate (sorties/hr) versus simulation.

K	Base	LCOM	ALS	LCOM
10	1.974	1.953	2.227	2.210
20	3.791	3.792	4.195	4.254
30	5.206	5.340	5.558	5.810
40	6.069	6.377	6.275	6.594
50	6.510	6.828	6.618	6.943

The first comparison was designed to investigate how sortie generation rates react and compare when adjusting the total number of aircraft in the system. The results are given for $K = 10, 20, 30, 40, 50, 60$ in Tables 4.4 and 4.5. These values are chosen to demonstrate how the model reacts to this parameter. However, the majority of airbases will not contain more than 30 aircraft. Exceptions to this do occur, and provide another reason for examining these values. The sortie generation rates for both models were graphed to demonstrate the trends in each model. Table 4.2 provides definitions for each station's designator and the different outputs reported.

Table 4.2 Descriptions of various network nodes.

Acronym	Node Description
PF	Pre-Flight
Sortie	Sortie
TS	Troubleshoot
O D	Order Delay
O	Order
M1	Maintenance substation 1 (airframe)
M2	Maintenance substation 2 (electrical/hydraulic)
M3	Maintenance substation 3 (engine/propulsion)
M4	Maintenance substation 4 (avionics)
M5	Maintenance substation 5 (radar/weapons control)
TA	Turnaround
MU	Munitions Upload

Table 4.3 Definitions of output variables.

Variable	Definition
R	Response time (service time + waiting time)
λ	Throughput - service completions per hour
λ_{sortie}	Sortie generation rate (sorties/hour)
Q	Average number of customers in station
U	Single-server: average percent of time busy
U	Multi-server: average no of servers busy

Tables 4.4 and 4.5 show that the ALS model provides improvements in all aspects over the base model. The sortie generation rate continued to improve as aircraft are added to the system. Figure 4.1 graphically depicts the sortie generation rate as a function of the total number of aircraft in system.

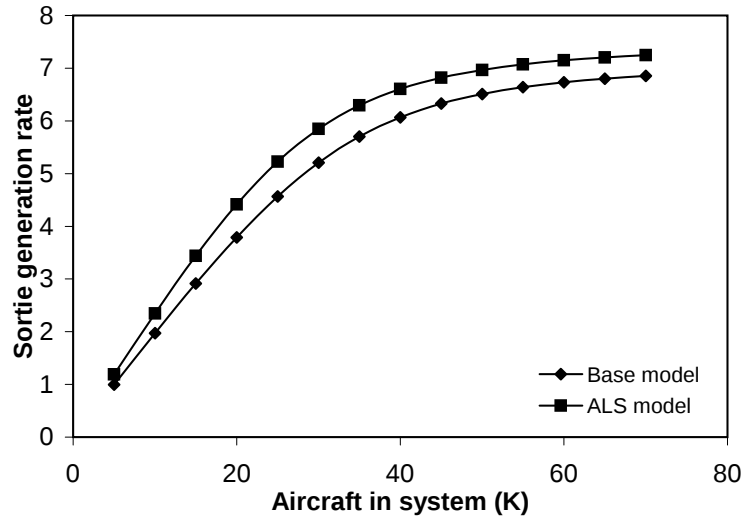


Figure 4.1 Sortie generation rate as a function of K .

Next, the false alarm penalty was examined. The probability of the fork-join node experiencing the occurrence of no paths is used to model the real-world occurrence of false alarms. The false alarm penalty is adjusted by adding a mean service time. The false alarm penalty can be added to the model and the ALS still experienced higher sortie generation rates. With 10 aircraft in system, the sortie generation rate for the base model was 1.9744 and for the ALS model was 2.2008.

Table 4.4 Base model versus ALS model results.

		Base Model				ALS model			
	Station	R	λ	Q	U	R	λ	Q	U
$K=10$	PF	0.250	2.078	0.520	0.520	0.250	2.345	0.586	0.586
	Sortie	2.000	1.974	3.949	3.949	2.000	2.227	4.455	4.455
	TS	0.500	0.696	0.348	0.348	0.250	0.785	0.196	0.196
	O D	0.050	0.197	0.010	0.010	—	—	—	—
	O	2.000	0.197	0.393	0.393	—	—	—	—
	M1	2.848	0.118	0.337	0.260	2.956	0.134	0.395	0.294
	M2	2.284	0.272	0.620	0.616	2.289	0.306	0.701	0.695
	M3	2.422	0.146	0.354	0.347	2.437	0.165	0.402	0.391
	M4	1.989	0.188	0.374	0.282	2.072	0.212	0.439	0.318
	M5	1.222	0.320	0.391	0.381	1.231	0.361	0.445	0.430
	TA	0.750	2.078	1.559	1.559	0.750	2.345	1.759	1.758
	MU	0.502	2.078	1.042	1.039	0.502	2.345	1.178	1.172
$K=20$	PF	0.250	3.990	0.998	0.998	0.250	4.415	1.104	1.104
	Sortie	2.000	3.791	7.581	7.581	2.000	4.195	8.389	8.389
	TS	0.500	1.337	0.668	0.668	0.250	1.479	0.370	0.370
	O D	0.050	0.377	0.019	0.019	—	—	—	—
	O	2.000	0.377	0.754	0.754	—	—	—	—
	M1	4.070	0.227	0.925	0.500	4.463	0.252	1.122	0.553
	M2	2.398	0.521	1.250	1.183	2.444	0.577	1.410	1.310
	M3	2.617	0.281	0.735	0.665	2.681	0.311	0.833	0.736
	M4	2.969	0.361	1.072	0.541	3.299	0.399	1.318	0.599
	M5	1.343	0.615	0.826	0.732	1.383	0.680	0.941	0.810
	TA	0.762	3.990	3.041	2.993	0.771	4.415	3.402	3.312
	MU	0.528	3.990	2.106	1.995	0.541	4.415	2.389	2.208
$K=30$	PF	0.250	5.480	1.370	1.370	0.250	5.850	1.463	1.463
	Sortie	2.000	5.206	10.412	10.412	2.000	5.558	11.116	11.116
	TS	0.500	1.836	0.918	0.918	0.250	1.960	0.490	0.490
	O D	0.050	0.518	0.026	0.026	—	—	—	—
	O	2.000	0.518	1.036	1.036	—	—	—	—
	M1	6.166	0.312	1.924	0.687	7.046	0.333	2.347	0.733
	M2	2.657	0.716	1.902	1.625	2.761	0.764	2.111	1.735
	M3	2.927	0.386	1.128	0.914	3.035	0.412	1.249	0.975
	M4	4.843	0.496	2.401	0.744	5.698	0.529	3.015	0.794
	M5	1.543	0.845	1.303	1.005	1.615	0.902	1.456	1.073
	TA	0.830	5.480	4.546	4.110	0.867	5.850	5.071	4.388
	MU	0.615	5.480	3.370	2.740	0.657	5.850	3.845	2.925

Table 4.5 Base model versus ALS model results.

		Base Model				ALS model			
	Station	R	λ	Q	U	R	λ	Q	U
$K=40$	PF	0.250	6.388	1.597	1.597	0.250	6.605	1.651	1.651
	Sortie	2.000	6.069	12.137	12.137	2.000	6.275	12.550	12.550
	TS	0.500	2.140	1.070	1.070	0.250	2.213	0.553	0.553
	O D	0.050	0.604	0.030	0.030	—	—	—	—
	O	2.000	0.604	1.208	1.208	—	—	—	—
	M1	9.260	0.364	3.369	0.800	10.565	0.376	3.974	0.828
	M2	2.988	0.835	2.494	1.895	3.104	0.863	2.679	1.959
	M3	3.245	0.449	1.458	1.065	3.343	0.465	1.554	1.101
	M4	8.121	0.578	4.692	0.867	9.702	0.598	5.797	0.896
	M5	1.761	0.984	1.734	1.171	1.832	1.018	1.865	1.211
	TA	0.973	6.388	6.217	4.791	1.039	6.605	6.866	4.954
	MU	0.772	6.388	4.928	3.194	0.841	6.605	5.553	3.303
$K=50$	PF	0.250	6.853	1.713	1.713	0.250	6.966	1.742	1.742
	Sortie	2.000	6.510	13.020	13.020	2.000	6.618	13.235	13.235
	TS	0.500	2.296	1.148	1.148	0.250	2.334	0.583	0.583
	O D	0.050	0.648	0.032	0.032	—	—	—	—
	O	2.000	0.648	1.296	1.296	—	—	—	—
	M1	12.920	0.390	5.042	0.859	14.365	0.397	5.699	0.873
	M2	3.272	0.895	2.929	2.032	3.359	0.910	3.057	2.066
	M3	3.475	0.482	1.675	1.143	3.540	0.490	1.735	1.161
	M4	13.044	0.620	8.085	0.930	15.371	0.630	9.685	0.945
	M5	1.929	1.056	2.037	1.257	1.978	1.074	2.124	1.277
	TA	1.163	6.853	7.969	5.140	1.240	6.966	8.636	5.225
	MU	0.968	6.853	6.631	3.426	1.046	6.966	7.287	3.483
$K=60$	PF	0.250	7.087	1.772	1.772	0.250	7.148	1.787	1.787
	Sortie	2.000	6.733	13.465	13.465	2.000	6.790	13.581	13.581
	TS	0.500	2.374	1.187	1.187	0.250	2.395	0.599	0.599
	O D	0.050	0.670	0.034	0.034	—	—	—	—
	O	2.000	0.670	1.340	1.340	—	—	—	—
	M1	16.551	0.404	6.680	0.888	17.912	0.407	7.291	0.896
	M2	3.464	0.926	3.208	2.102	3.521	0.934	3.288	2.120
	M3	3.615	0.499	1.802	1.182	3.653	0.503	1.837	1.192
	M4	19.610	0.641	12.570	0.962	22.651	0.647	14.644	0.970
	M5	2.036	1.092	2.224	1.300	2.066	1.102	2.276	1.311
	TA	1.357	7.087	9.617	5.315	1.430	7.148	10.224	5.361
	MU	1.165	7.087	8.256	3.544	1.239	7.148	8.856	3.574

With 50 aircraft in the system, the sortie generation rate for the base model was 6.5100 and for the ALS model was 6.6086. This trend held even in the unrealistic case when the penalty for a false alarm was set to 10 hours.

The next comparison examined the impact of reduced resource level for the maintenance node in the ALS model as compared to the base model with a full complement of maintenance servers. This was done by trial and error to find the cases where the ALS model outperforms the base model via sortie generation rate. The results are depicted in Table 4.7.

The results demonstrate the capability of the ALS model with fewer maintenance servers to achieve the sortie generation rate of the base model with a full complement of servers. For the case of 10 aircraft in system, the extreme case of single servers for each maintenance substation in the ALS model still exceeded that of the base model. When $K = 30$, the ALS model produced a higher sortie generation rate with one fewer resource at maintenance substation 2. While unlikely, when more than 30 aircraft are in the system, the ALS required all servers to outperform the base model. This is due to the large number of aircraft severely constraining the full complement of servers. When more servers were added to the base model, the ALS model outperformed that result with fewer servers. Allowing the base model to have 3 servers at all maintenance substations gave a sortie generation rate of 7.1278 with 50 aircraft in the system. With this number of aircraft in system, the ALS model with fewer servers (2,3,3,2,2) gave a sortie generation rate of 7.1855. The results

Table 4.6 Sortie generation rate (sortie/hr) as false alarm penalty increases.

Penalty	ALS	Base
0	2.227	1.996
1	2.201	1.974
2	2.175	1.953
10	1.987	1.800

Table 4.7 Base versus ALS sortie generation rate (sorties/hr) with fewer servers.

No. of Servers	$K = 10$	$K = 30$	$K = 50$
Base - 1,3,2,1,2	1.974	5.206	6.510
ALS - 1,3,2,1,2	2.186	5.507	6.603
ALS - 1,2,2,1,2	2.176	5.306	6.211
ALS - 1,1,1,1,1	2.041	3.203	3.203

of this comparison show that ALS-enhanced operations may achieve a comparable sortie generation rate of the current operations despite having fewer resources.

The final comparison was to adjust the mean service time at the troubleshoot node for the ALS model. Since there is debate as to how much the ALS will reduce the time required for troubleshoot activities, this was examined by scaling the mean troubleshoot duration by 0%, 20%, 40%, 60%, 80%, and 100% of the base model control values. The results in Table 4.8 show that ALS-enhanced operations will outperform current operations for any reduction of time, even in the case where the mean service time for the ALS troubleshoot node is 100% of the base model.

Table 4.8 Sortie generation rate (sorties/hr) when service time (T) reduced in troubleshoot.

	Base	ALS					
K	T	T	0.8T	0.6T	0.4T	0.2T	0T
5	0.997	1.107	1.115	1.124	1.133	1.142	1.151
20	3.791	4.131	4.156	4.182	4.208	4.234	4.260
35	5.704	5.943	5.959	5.974	5.989	6.005	6.020
50	6.510	6.603	6.609	6.614	6.621	6.626	6.632

The graph of the results for $K = 35$ and $K = 50$ can be found in Figure 4.2. This result demonstrates that ALS-enhanced operations provide an airbase with an increase in sortie generation rate. It is interesting to note that, for the analytical models presented in this thesis, the ALS model outperformed the base model regardless of the percentage of the mean service time for troubleshoot activities were used for the ALS model. This implies that, despite the current unknown amount of reduction

for troubleshoot service time, ALS-enhanced operations are still expected to yield higher sortie generation rates than current operations.

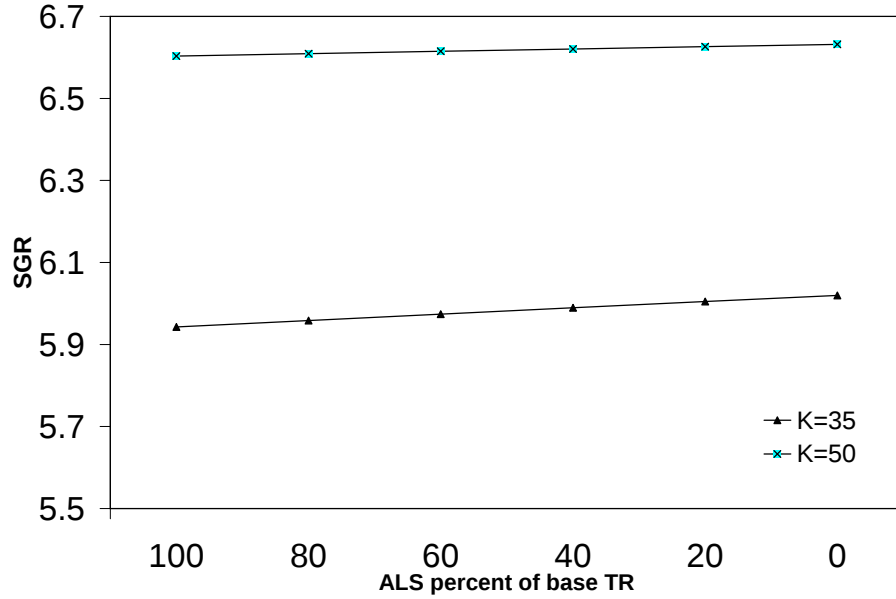


Figure 4.2 Sortie generation rate as time for troubleshoot decreases in ALS.

In this chapter, the ALS was shown to achieve higher sortie generation rates with any number of aircraft in system, a false alarm penalty, and any reduction in mean service time for the troubleshoot activity. Moreover, the number of servers at the maintenance substations can be reduced in ALS-enhanced operations and achieve the sortie generation rate of current operations. These results have illustrated that the ALS will positively impact the sortie generation process with equivalent or reduced resource levels.

5. Conclusions and Future Research

The primary objective of this thesis was to examine the sortie generation process and the changes it would experience when autonomic logistics are implemented. This work led to the development of two models, one for current airbase operations and one for ALS-enhanced operations. The base and ALS models are closed queueing networks containing a fork-join node. Modeling the sortie generation process in this manner required assumptions such as a fly-when-ready schedule, a perfect performing PHM, and exponentially distributed service times. For the given assumptions the ALS model was analytically shown to always produce higher sortie generation rates than the base model.

The output of the queueing network models were then compared to the simulated results of the Logistics Composite Model (LCOM), the Air Force tool for determining sortie generation rate. The queueing network models were then used to numerically examine the sortie generation rate under several scenarios. These model comparisons, with the given problem assumptions, demonstrated that the ALS-model will yield a higher generation rate than the base model.

The first numerical comparison performed between the two models was designed to investigate how the models compared as the number of aircraft in the system increases. This comparison numerically demonstrated the analytical result that the ALS model will always outperform the base model, with both models subjected to the underlying assumptions. This result is important as the reason for implementing autonomic logistics is to positively impact the sortie generation process for any number of aircraft at an airbase.

A major advantage of autonomic logistics that is assumed in the current literature is that implementing an ALS into the sortie generation process would allow a base to achieve the sortie generation rate of current operations with fewer resources [9, 19]. In the case where current operations are compared to the ALS model with

single-server maintenance nodes and 10 aircraft, the sortie generation rate is higher for the ALS model. This provides evidence that ALS-enhanced operations with fewer resources can meet the sortie generation rate of current operations with a full complement of resources. As the number of aircraft increases, more resources are required for the ALS to outperform the base model. At 30 aircraft a server can be removed from maintenance substation 2 and the sortie generation rate is higher for the ALS model than the base model. For more than 30 aircraft, the ALS requires an equal number of resources to outperform the base model. These results, all of which are subjected to the assumptions of the models, show that the ALS using fewer resources can indeed achieve the sortie generation rate of current operations.

The two models were also compared for differing mean service times in the troubleshoot node of the ALS model. The results demonstrated that ALS-enhanced operations will yield a higher sortie generation rate even if the mean service time for the ALS model is the same as in the base model where both models are subjected to the assumptions of the models. Although the duration of the troubleshoot activity with an implemented ALS is not well understood, this result shows that autonomic logistics will positively impact the sortie generation process despite the reduction in mean service time for troubleshooting activities.

These combined results provide the military operations research community with analytical and numerical evidence that implementing autonomic logistics will positively impact the sortie generation process. Current literature on the ALS assumes that a positive impact will occur, however, this was based only on speculation on the advantages of autonomic logistics. Modeling the basics of pre-ALS and ALS airbase operations allowed for comparisons that clearly illustrate the improved sortie generation rates for the ALS model. In addition, the numerical results confirm the assumption that an airbase will be able to maintain current sortie generation rates using reduced resource levels for maintenance activities. This conclusion ap-

pears to be heavily influenced by the presupposition of a perfect prognostics health management system.

The comparison of the models presented in this thesis will prove valuable once the ALS has been implemented and actual performance data can be collected. One major consideration is the handling of maintenance actions in an ALS sortie generation process. In theory, all maintenance actions will become scheduled as maintainers will have information regarding impending failures. It is not possible to definitively know that current probabilities of required maintenance actions will remain the same under the ALS. Should repairs be made to avoid a failure, probabilities could increase, but if subsystems experience fewer failures due to a better maintained aircraft, these probabilities could decrease. These items and aspects yet uncovered could further impact future models of the ALS sortie generation process.

The work of this thesis would more accurately reflect real-world conditions if the service time distributions were allowed to be a distribution other than the exponential. This does not present a problem for this thesis as the intent was to examine differences between current operations and ALS-enhanced operations. However, when implemented ALS data on service time distributions exist, it would be of interest to remove this assumption and subsequently apply the more general results of Hackman and Dietz [8].

A possible extension to this thesis would be to analytically model the prognostics and health management (PHM) system of the ALS. The main interest here would be to examine how the Joint Distributed Information System (JDIS) orders the spare parts required for future maintenance actions. Since this part of the network is somewhat disparate from the remainder of the sortie generation process, an approach to modeling the ordering activity of JDIS would be the use of queueing networks with signals. This methodology is relatively novel; therefore this approach would require extension of known queueing network results to incorporate signals.

The sortie generation rates produced by the models in this thesis assume a fly-when-ready concept which means that, as soon as an aircraft is available to fly a sortie, it will fly a sortie. This assumption could closely emulate war-time operations, but it is not consistent with the peace-time environment. Due to the nature of queueing networks, it would be difficult to relax this assumption. It is conjectured that a simulation study would allow for a mission schedule, and thus, be more appropriate for this sort of analysis.

Bibliography

1. Abell, J., *The Sortie Generation Model System, Volume I: Executive Summary*. Contract MDA903-80-C-0554, Logistics Management Institute, September 1981.
2. Baskett, F., K. Chandy, R. Muntz, and F. Palacios (1975), Open, closed, and mixed networks of queues with different classes of customers, *Journal of the ACM*, **22** 248-260.
3. Bolch, G., S. Greiner, H. de Meer, and K. Trivedi (1998), *Queueing Networks and Markov Chains*. John Wiley & Sons, Inc., New York.
4. Boyle, E., *LCOM explained*. Technical Report AFHRKL-TP-90-58, Wright-Patterson AFB, OH: Logistics and Human Factors Division, Human Resources Laboratory, July 1990.
5. Dahlman, C., R. Kerchner, and D. Thaler (2002), *Setting Requirements for Maintenance Manpower in the U.S. Air Force*. RAND, Santa Monica.
6. Dietz, D., and R. Jenkins (1997), Analysis of aircraft sortie generation with the use of a fork-join queueing network mode, *Naval Research Logistics* **44** , 153-164.
7. Gross, D. and C. Harris (1998), *Fundamentals of Queueing Theory*. John Wiley & Sons, Inc., New York.
8. Hackman, D., and D. Dietz (1998), Analytical modeling of aircraft sortie generation with concurrent maintenance and general service times, *MOR Journal* **3**, 61-80.
9. Henley, S., R. Curren, B. Scheuren, A. Hess, and G. Goodman (2000), Autonomous logistics-support concept for the 21st century. *IEEE Aerospace Conference Proceedings* 417-421.
10. Jackson, J. (1963), Jobshop-like queueing systems, *Management Science*, **10** 131-142.
11. Jenkins, R., *A Mean Value Analysis Heuristic for Analysis of Aircraft Sortie Generation*. MS Thesis AFIT/GOR/ENS/94M-07 , Air Force Institute of Technology(AU), Wright-Patterson AFB OH, March 1994.
12. Malley, M., *A Methodology for Simulating the Joint Strike Fighter's (JSF) Prognostics and Health Management System*. MS Thesis AFIT/GOR/ENS/01M-11 , Air Force Institute of Technology(AU), Wright-Patterson AFB OH, March 2001.
13. Miller, L.W., R. Stanton, and G. Crawford, *Dyna-Sim: A Non-stationary Queueing Simulation with Application to the Automated Test Equipment Problem*. Technical Report N-2087-AF, The Rand Corporation, July 1984.

14. Onvural, R. (1990), Survey of closed queueing networks with blocking, *ACM Computing Surveys* **22**, 83-121.
15. Perros, H. (1994), *Queueing Networks with Blocking*. Oxford University Press, New York.
16. Rao, P.C., and R. Suri, Analytical models for closed systems - Part II, unpublished, September 1993.
17. Rebulanan, R., *Simulation of the Joint Strike Fighter's (JSF) Autonomic Logistics System (ALS) Using the Java[®] Programming Language*. MS Thesis AFIT/GOR/ENS/00M-19, Air Force Institute of Technology(AU), Wright-Patterson AFB OH, March 2000.
18. Reiser, M., and S. Lavenberg (1980), Mean value analysis of closed multichain queueing networks, *Journal of the ACM*, **27**, 313-322.
19. Smith, G., J. B. Schroeder, and B. Masquelier (1999), Logistics for the Joint Strike Fighter - It ain't business as usual, *Air Force Journal of Logistics* **23**, 12-17.
20. Willits, C., *Nested Fork-Join Queueing Networks and Their Application to Mobility Airfield Operations Analysis*. Ph.D. Dissertation, Air Force Institute of Technology(AU), Wright-Patterson AFB OH, March 1997(ADA323251).

Appendix A. Base Model Matlab[®] Codes

```

%*****
% Program phase16
%
% The purpose of this MATLAB program is to compute the performance
% measures of the sortie generation process as described in Chapter 3.
% The process is a 6 node queueing network where the third and fourth
% nodes are fork-join nodes, allowing simultaneous service of
% troubleshoot, order delay and order activities for node three and
% for node four up to five maintenance activities.
%       Author: Lt. Nicholas Yager
% Last Revision: 1/28/03
%
%*****
clear;
%number of jobs in system
K = 10;
%service times for normal nodes
s = [.25 2 .6 2 .75 .5];
%service times for fork-join substations
s(41:45) = [2.2 2.27 2.37 1.5 1.19]; s(31:33) = [.5 .05 2];
%number of servers for normal nodes
r = [K K K K 6 4];
%number of servers for fork-join substations
r(41:45) = [1 3 2 1 2]; r(31:33) = [K K K];
%visit ratios
v = [1 .95 .335 .335 1 1];
%probability of needing service at a fork-join substations
Q(41:45)= [.17 .39 .21 .27 .46]; Q(31:33)= [1 .28218 .28218];
%Lines 33-44 initialize the MVA algorithm
%pi(a,1,1) is probability of zero jobs at node a
%with zero total jobs in system
for a = 1:6
    pi(a,1,1) = 1;
end clear a; for a = 41:45
    pi(a,1,1) = 1;
end clear a; for a = 31:33
    pi(a,1,1) = 1;
end
%n loops the algorithm for 1 to K, the iterative part of MVA
for n = 2:K+1
    %Calculates response time for maintenance substation
    for e = 41:45
        clear f;

```

```

    for f = 2:n
        sumRfj(f) = ((f-1)/min((f-1)/s(e),r(e)/s(e)))*pi(e,f-1,n-1);
    end
    %response time calcs for fork-join substations
    Rfj(e,n) = sum(sumRfj);
    theta(e,n) = 1/Rfj(e,n);
end
%get_EMaxT1 is a function that calculates the response time for the
%entire maintenance node
sumEmxTtimespi1(n) = get_EmaxT1(theta(:,n));
clear theta2; clear e;
%Calculates response time for maintenance substation
for e = 31:33
    clear f;
    for f = 2:n
        sumRfj(f) = ((f-1)/min((f-1)/s(e),r(e)/s(e)))*pi(e,f-1,n-1);
    end
    %response time calcs for fork-join substations
    Rfj(e,n) = sum(sumRfj);
    theta3(e,n) = 1/Rfj(e,n);
end
%get_EmaxT3 calculates the response time for the fork-join node
%containing troubleshoot, order delay and order
sumEmxTtimespi2(n) = get_EmaxT3(theta3(:,n));
for i = 1:6
    for j = 2:n
        sumR(j) = ((j-1)/min((j-1)/s(i),r(i)/s(i)))*pi(i,j-1,n-1);
    end
    %response time calc for normal nodes
    R(i,n) = sum(sumR);
end
%needed summation for cycle time calculation
for viloop = 1:6
    viRsum(viloop)=v(viloop)*R(viloop,n);
end
viR(n)=sum(viRsum(1:2))+sum(viRsum(5:6));
%calculates cycle time for a network containing a fork-join node
newCT1(n)=viR(n)+v(4)*sumEmxTtimespi1(n)+v(3)*sumEmxTtimespi2(n);
for fj = 41:45
    %throughput calculation for fork-join substations
    lambdafj(fj,n)=v(4)*(n-1)*Q(fj)/(newCT1(n));
    %number of jobs at fork-join substations
    Q1(fj,n)=Rfj(fj,n)*lambdafj(fj,n);
    %utilization at fork-join substations
    U(fj,n)=s(fj)*lambdafj(fj,n);

```

```

end
for fj = 31:33
    %throughput calculation for fork-join substations
    lambdafj(fj,n)=v(4)*(n-1)*Q(fj)/(newCT1(n));
    %number of jobs at fork-join substations
    Ql(fj,n)=Rfj(fj,n)*lambdafj(fj,n);
    %utilization at fork-join substations
    U(fj,n)=s(fj)*lambdafj(fj,n);
end
%throughput calculation for normal nodes
for d = 1:6
    lambda(d,n) = v(d)*(n-1)/(newCT1(n));
end
clear i;
%lines 111-123: recalculation of fork-join(4) pi values
for i = 41:45
    for c = 2:n
        pi(i,c,n)=(lambdafj(i,n)/min((c-1)/s(i),r(i)/s(i)))*
            pi(i,c-1,n-1);
    end
    %vector of p(i,n,N) values where 1>n>=N
    for l = 2:n
        sumpi1(l)=pi(i,l,n);
    end
    sumpi=sum(sumpi1);
    %finds p(i,0,N)
    pi(i,1,n)=1 - sumpi;
end
%lines 124-138: recalculation of fork-join(3) pi values
clear i; clear j;
for i = 31:33
    %finds p(i,n,N) values for n>=1 given N
    for c = 2:n
        pi(i,c,n)=(lambdafj(i,n)/min((c-1)/s(i),r(i)/s(i)))*
            pi(i,c-1,n-1);
    end
    %vector of p(i,n,N) values where 1>n>=N
    for l = 2:n
        sumpi1(l) = pi(i,l,n);
    end
    sumpi = sum(sumpi1);
    %finds p(i,0,N)
    pi(i,1,n) = 1 - sumpi;
end
%lines 139-153: recalculation of standard pi values

```

```

clear i; clear j;
for i = 1:6
    %finds p(i,n,N) values for n>=1 given N
    for j = 2:n
        pi(i,j,n)=(lambda(i,n)/min((j-1)/s(i),r(i)/s(i)))*
            pi(i,j-1,n-1);
    end
    %vector of p(i,n,N) values where 1>n>=N
    for l = 2:n
        sumpi12(l) = pi(i,l,n);
    end
    sumpi2 = sum(sumpi12);
    %finds p(i,0,N)
    pi(i,1,n) = 1 - sumpi2;
end
for z = 1:6
    %number of jobs at node i
    Ql(z,n)=R(z,n)*lambda(z,n);
    %utilization at node i
    U(z,n)=s(z)*lambda(z,n);
end
end
%lines 162-170 display a table of the results
rshow=[R(1,n); R(2,n); Rfj(31,n); Rfj(32,n); Rfj(33,n); Rfj(41,n);
Rfj(42,n);
Rfj(43,n); Rfj(44,n); Rfj(45,n); R(5,n); R(6,n)];
lambdashow=[lambda(1,n); lambda(2,n); lambdaj(31,n);
lambdaj(32,n);
lambdaj(33,n); lambdaj(41,n); lambdaj(42,n); lambdaj(43,n);
lambdaj(44,n); lambdaj(45,n); lambda(5,n); lambda(6,n)];
qlength = [Ql(1,n); Ql(2,n); Ql(31,n); Ql(32,n); Ql(33,n);
Ql(41,n);
Ql(42,n); Ql(43,n); Ql(44,n); Ql(45,n); Ql(5,n); Ql(6,n)];
ushow = [U(1,n); U(2,n); U(31,n); U(32,n); U(33,n); U(41,n);
U(42,n);
U(43,n); U(44,n); U(45,n); U(5,n); U(6,n)];
table = [rshow lambdashow qlength ushow]

```

```

%*****
% Program get_EmaxT
%
% The purpose of this MATLAB program is to compute expected maximum
% of the time required to complete service in the maintenance
% fork-join node.
%       Author: Lt. Nicholas Yager
% Last Revision: 1/28/03
%
%*****
function EmaxT = get_EmaxT(theta2);
%omega defines the state space for the fork-join node
omega = [1 1 1 1 1;41 1 1 1 1;42 1 1 1 1; 43 1 1 1 1; 44 1 1 1 1;
        45 1 1 1 1; 41 42 1 1 1; 41 43 1 1 1; 41 44 1 1 1; 41 45 1 1 1;
        42 43 1 1 1; 42 44 1 1 1; 42 45 1 1 1; 43 44 1 1 1; 43 45 1 1 1;
        44 45 1 1 1; 41 42 43 1 1; 41 42 44 1 1; 41 42 45 1 1;
        41 43 44 1 1; 41 43 45 1 1; 41 44 45 1 1; 42 43 44 1 1;
        42 43 45 1 1; 42 44 45 1 1; 43 44 45 1 1; 41 42 43 44 1;
        41 42 43 45 1; 41 42 44 45 1; 41 43 44 45 1; 42 43 44 45 1;
        41 42 43 44 45];
%piofS is the probability of each omega row occurring
piofS = [0.157670933; 0.032294047; 0.100806007; 0.041912527;
        0.058316647; 0.134312277; 0.020647013; 0.008584493; 0.011944373;
        0.027509743; 0.026796533; 0.037284413; 0.085871783; 0.015501893;
        0.035703263; 0.049677143; 0.005488447; 0.007636567; 0.017588197;
        0.003175087; 0.007312717; 0.010174837; 0.009911047; 0.022826677;
        0.031760797; 0.013205317; 0.002029973; 0.004675343; 0.006505223;
        0.002704703; 0.008442743; 0.001729237];
%this loop calculates E[T] for each row of omega
for Sloop = 1:32
    [w,v] = size(theta2);
    if w == 45
        S = omega(Sloop,:);
    end
    for checkloop = 1:5
        if S(6-checkloop) == 1
            S = S(1:5-checkloop);
        end
    end
    clear x; clear sum1; clear sum2; clear sum3; clear sum4;
    clear lastsum; clear sum11; clear lastsum1;
    [w,x] = size(S);
    %x=0 sets the false-alarm penalty
    if x==0
        EmaxT(Sloop)=1;
    end
end

```

```

end
if x >= 1
    for g = 1:x
        sum11(g)=1/theta2(S(g));
    end
    sum1=sum(sum11);
    %the following 3 if statements find sums required to calculate E[T]
    if x>=3
        sum2=(1/(theta2(S(1))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(3))))+
            (1/(theta2(S(3))+theta2(S(2))));
    end
    if x >=4
        sum3=(-1)*(1/(theta2(S(1))+theta2(S(2))))-
            (1/(theta2(S(1))+theta2(S(3))))-
            (1/(theta2(S(3))+theta2(S(2))))-
            (1/(theta2(S(1))+theta2(S(4))))-
            (1/(theta2(S(4))+theta2(S(3))))-
            (1/(theta2(S(4))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(4))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(4))))+
            (1/(theta2(S(4))+theta2(S(2))+theta2(S(3))));
    end
    if x >=5
        sum4=(-1)*(1/(theta2(S(1))+theta2(S(2))))-
            (1/(theta2(S(4))+theta2(S(5))))-
            (1/(theta2(S(3))+theta2(S(5))))-
            (1/(theta2(S(2))+theta2(S(5))))-
            (1/(theta2(S(1))+theta2(S(5))))-
            (1/(theta2(S(1))+theta2(S(3))))-
            (1/(theta2(S(3))+theta2(S(2))))-
            (1/(theta2(S(1))+theta2(S(4))))-
            (1/(theta2(S(4))+theta2(S(3))))-
            (1/(theta2(S(4))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(4))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(5))))+
            (1/(theta2(S(1))+theta2(S(3))+theta2(S(4))))+
            (1/(theta2(S(1))+theta2(S(3))+theta2(S(5))))+
            (1/(theta2(S(1))+theta2(S(4))+theta2(S(5))))+
            (1/(theta2(S(2))+theta2(S(3))+theta2(S(4))))+
            (1/(theta2(S(2))+theta2(S(3))+theta2(S(5))))+
            (1/(theta2(S(2))+theta2(S(4))+theta2(S(5))))+
            (1/(theta2(S(3))+theta2(S(4))+theta2(S(5))))-

```

```

        (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))+theta2(S(4))))-
        (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))+theta2(S(5))))-
        (1/(theta2(S(1))+theta2(S(2))+theta2(S(5))+theta2(S(4))))-
        (1/(theta2(S(1))+theta2(S(5))+theta2(S(3))+theta2(S(4))))-
        (1/(theta2(S(5))+theta2(S(2))+theta2(S(3))+theta2(S(4))));
    end
    %finds the sum of the substation response times
    for p = 1:x
        lastsum1(p)=theta2(S(p));
    end
    lastsum=sum(lastsum1);
    %the following 5 if statements find E[T] as
    %defined in Ch3
    if x==1
        EmxT(Sloop)=sum1;
    end
    if x==2
        EmxT(Sloop)=sum1-(1/lastsum);
    end
    if x==3
        EmxT(Sloop)=sum1-sum2+(1/lastsum);
    end
    if x==4
        EmxT(Sloop)=sum1+sum3-(1/lastsum);
    end
    if x==5
        EmxT(Sloop)=sum1+sum4+(1/lastsum);
    end
end end
%calculates the 32 E[T]
%by multiplying by the probability of occurrence
for t = 1:32
    EmxTtimespi(t) = EmxT(t)*piofS(t);
end
%sums the terms, giving the expected maximum of T
sumEmxTtimespi=sum(EmxTtimespi); EmaxT = sumEmxTtimespi; clear
theta2;

```

```

%*****
% Program get_EmaxT3
%
% The purpose of this MATLAB program is to compute expected maximum
% of the time required to complete service in the troubleshoot,
% order delay, and order fork-join node.
%       Author: Lt. Nicholas Yager
% Last Revision: 1/28/03
%
%*****
function EmaxT = get_EmaxT(theta4);
%omega is the state space of S and piofS is the probability of each state
piofS = [0; 0.15767; 0.842328]; omega = [1 1 1; 31 1 1; 31 32 33];
for Sloop = 1:3
    S = omega(Sloop,:);
    %the following removes the 1's from S
    for checkloop = 1:3
        if S(4-checkloop) == 1
            S = S(1:3-checkloop);
        end
    end
    clear x; clear sum1; clear sum2; clear sum3; clear sum4;
    clear lastsum; clear sum11; clear lastsum1;
    [w,x] = size(S);
    %EmxT = E[maxT] = expected value of the maximum of T
    if x == 1
        EmxT(Sloop)=1/(theta4(S(1)));
    end
    if x == 2
        EmxT(Sloop)=1/theta4(S(1))+1/theta4(S(2))-
            1/(theta4(S(1))+theta4(S(2)));
    end
    if x == 3
        EmxT(Sloop)=1/(theta4(S(1))+theta4(S(2)))+1/theta4(S(3))-
            1/(theta4(S(1))+theta4(S(2))+theta4(S(3)));
    end
end
%calculates the 3 E[T], and multiplies by the probability of occurrence
for t = 1:3
    EmxTtimespi(t) = EmxT(t)*piofS(t);
end
%sums the terms, giving the expected maximum of T
sumEmxTtimespi=sum(EmxTtimespi); EmaxT = sumEmxTtimespi; clear
theta4;

```

Appendix B. ALS Model Matlab[®] Code

```
%*****
% Program phase23
%
% The purpose of this MATLAB program is to compute the performance
% measures of the sortie generation process as described by Dietz
% and Jenkins. The process is a 6 node queueing network where the
% fourth node is a fork-join node, allowing simlutaneous service up
% to five maintenance activities.
% Author: Lt. Nicholas Yager
% Last Revision: 1/28/03
%
%*****
clear;
%number of jobs in sysytem
K = 50;
%service times for normal nodes
s = [.25 2 .25 2 .75 .5];
%service times for fork-join substations
s(41:45) = [2.2 2.27 2.37 1.5 1.19];
%number of servers for normal nodes
r = [K K K K 6 4];
%number of servers for fork-join substations
r(41:45) = [1 3 2 1 2];
%visit ratios
v = [1 .95 .335 .335 1 1];
%probability of needing service at a fork-join substations
Q(41:45)= [.17 .39 .21 .27 .46];
%lines 32-38 initialize the MVA algorithm
%pi(a,1,1) is probability of zero jobs at node a
%with zero total jobs in system
for a = 1:6
    pi(a,1,1) = 1;
end
clear a;
for a = 41:45
    pi(a,1,1) = 1;
end
%n loops the algorithm for 1 to K, the iterive part of MVA
for n = 2:K+1
    for e = 41:45
        clear f;
        for f = 2:n
            sumRfj(f) = ((f-1)/min((f-1)/s(e),r(e)/s(e)))*pi(e,f-1,n-1);
        end
        %response time calcs for fork-join substations
    end
end
```

```

        Rfj(e,n) = sum(sumRfj);
        theta(e,n) = 1/Rfj(e,n);
    end
    %get_EmaxT1 is a function that calculates the expected maximum of T
    sumEmxTtimespi1(n) = get_EmaxT1(theta(:,n));
    clear theta2; clear e;
    for i = 1:6
        for j = 2:n
            sumR(j) = ((j-1)/min((j-1)/s(i),r(i)/s(i)))*pi(i,j-1,n-1);
        end
        %response time calc for normal nodes
        R(i,n) = sum(sumR);
    end
    %needed summation for cycle time calculation
    for viloop = 1:6
        viRsum(viloop)=v(viloop)*R(viloop,n);
    end
    viR(n)=sum(viRsum(1:3))+sum(viRsum(5:6));
    %calculates cycle time for a network containing a fork-join node
    newCT1(n)=viR(n)+v(4)*sumEmxTtimespi1(n);
    for fj = 41:45
        %throughput calculation for fork-join substations
        lambdafj(fj,n)=v(4)*(n-1)*Q(fj)/(newCT1(n));
        %number of jobs at fork-join substations
        Ql(fj,n)=Rfj(fj,n)*lambdafj(fj,n);
        %utilization at fork-join substations
        U(fj,n)=s(fj)*lambdafj(fj,n);
    end
    %throughput calculation for normal nodes
    for d = 1:6
        lambda(d,n) = v(d)*(n-1)/(newCT1(n));
    end
    clear i;
    %recalculation of fork-join pi values
    for i = 41:45
        %finds p(i,n,N) values for n>=1 given N
        for c = 2:n
            pi(i,c,n)=(lambdafj(i,n)/min((c-1)/s(i),r(i)/s(i)))*
                pi(i,c-1,n-1);
        end
        %vector of p(i,n,N) values where 1>n>=N
        for l = 2:n
            sumpi1(l) = pi(i,l,n);
        end
        sumpi = sum(sumpi1);
    end

```

```

        %finds p(i,0,N)
        pi(i,1,n) = 1 - sumpi;
    end
    clear i; clear j;
    %begin pi calcs for normal nodes
    for i = 1:6
        %finds p(i,n,N) values for n>=1 given N
        for j = 2:n
            pi(i,j,n)=(lambda(i,n)/min((j-1)/s(i),r(i)/s(i)))*
                pi(i,j-1,n-1);
        end
        %vector of p(i,n,N) values where 1>n>=N
        for l = 2:n
            sumpi12(l) = pi(i,l,n);
        end
        sumpi2 = sum(sumpi12);
        %finds p(i,0,N)
        pi(i,1,n) = 1 - sumpi2;
    end
    for z = 1:6
        %number of jobs at node i
        Ql(z,n)=R(z,n)*lambda(z,n);
        %utilization at node i
        U(z,n)=s(z)*lambda(z,n);
    end
end
%the remainder of the code prints the table of results
rshow=[R(1,n); R(2,n); R(3,n); Rfj(41,n); Rfj(42,n); Rfj(43,n);
Rfj(44,n);
        Rfj(45,n); R(5,n); R(6,n)];
lambdashow=[lambda(1,n); lambda(2,n); lambda(3,n); lambdafj(41,n);
        lambdafj(42,n); lambdafj(43,n); lambdafj(44,n); lambdafj(45,n);
        lambda(5,n); lambda(6,n)];
qlength = [Ql(1,n); Ql(2,n); Ql(3,n); Ql(41,n); Ql(42,n);
Ql(43,n); Ql(44,n);
        Ql(45,n); Ql(5,n); Ql(6,n)];
ushow = [U(1,n); U(2,n); U(3,n); U(41,n); U(42,n); U(43,n);
U(44,n); U(45,n);
        U(5,n); U(6,n)];
table = [rshow lambdashow qlength ushow]

```

```

%*****
% Program get_EmaxT
%
% The purpose of this MATLAB program is to compute expected maximum
% of the time required to complete service in the maintenance
% fork-join node.
%       Author: Lt. Nicholas Yager
% Last Revision: 1/28/03
%
%*****
function EmaxT = get_EmaxT(theta2);
%omega defines the state space for the fork-join node
omega = [1 1 1 1 1;41 1 1 1 1;42 1 1 1 1; 43 1 1 1 1; 44 1 1 1 1;
        45 1 1 1 1; 41 42 1 1 1; 41 43 1 1 1; 41 44 1 1 1; 41 45 1 1 1;
        42 43 1 1 1; 42 44 1 1 1; 42 45 1 1 1; 43 44 1 1 1; 43 45 1 1 1;
        44 45 1 1 1; 41 42 43 1 1; 41 42 44 1 1; 41 42 45 1 1;
        41 43 44 1 1; 41 43 45 1 1; 41 44 45 1 1; 42 43 44 1 1;
        42 43 45 1 1; 42 44 45 1 1; 43 44 45 1 1; 41 42 43 44 1;
        41 42 43 45 1; 41 42 44 45 1; 41 43 44 45 1; 42 43 44 45 1;
        41 42 43 44 45];
%piofS is the probability of each omega row occurring
piofS = [0.157670933; 0.032294047; 0.100806007; 0.041912527;
0.058316647;
        0.134312277; 0.020647013; 0.008584493; 0.011944373; 0.027509743;
        0.026796533; 0.037284413; 0.085871783; 0.015501893; 0.035703263;
        0.049677143; 0.005488447; 0.007636567; 0.017588197; 0.003175087;
        0.007312717; 0.010174837; 0.009911047; 0.022826677; 0.031760797;
        0.013205317; 0.002029973; 0.004675343; 0.006505223; 0.002704703;
        0.008442743; 0.001729237];
%this loop calculates E[T] for each row of omega
for Sloop = 1:32
    [w,v] = size(theta2);
    if w == 45
        S = omega(Sloop,:);
    end
    %the following removes the 1's from S
    for checkloop = 1:5
        if S(6-checkloop) == 1
            S = S(1:5-checkloop);
        end
    end
    clear x; clear sum1; clear sum2; clear sum3; clear sum4;
    clear lastsum; clear sum11; clear lastsum1;
    [w,x] = size(S);
    %x=0 sets the false-alarm penalty

```

```

if x==0
    EmxT(Sloop)=1;
end
if x >= 1
    for g = 1:x
        sum11(g)=1/theta2(S(g));
    end
    sum1=sum(sum11);
    %the following 3 if statements find sums required to calculate E[T]
    if x>=3
        sum2=(1/(theta2(S(1))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(3))))+(1/(theta2(S(3))+theta2(S(2))));
    end
    if x >=4
        sum3=(-1)*(1/(theta2(S(1))+theta2(S(2))))-
            (1/(theta2(S(1))+theta2(S(3))))-(1/(theta2(S(3))+theta2(S(2))))-
            (1/(theta2(S(1))+theta2(S(4))))-(1/(theta2(S(4))+theta2(S(3))))-
            (1/(theta2(S(4))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(4))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(4))))+
            (1/(theta2(S(4))+theta2(S(2))+theta2(S(3))));
    end
    if x >=5
        sum4=(-1)*(1/(theta2(S(1))+theta2(S(2))))-
            (1/(theta2(S(4))+theta2(S(5))))-
            (1/(theta2(S(3))+theta2(S(5))))-(1/(theta2(S(2))+theta2(S(5))))-
            (1/(theta2(S(1))+theta2(S(5))))-(1/(theta2(S(1))+theta2(S(3))))-
            (1/(theta2(S(3))+theta2(S(2))))-(1/(theta2(S(1))+theta2(S(4))))-
            (1/(theta2(S(4))+theta2(S(3))))-(1/(theta2(S(4))+theta2(S(2))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(4))))+
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(5))))+
            (1/(theta2(S(1))+theta2(S(3))+theta2(S(4))))+
            (1/(theta2(S(1))+theta2(S(3))+theta2(S(5))))+
            (1/(theta2(S(1))+theta2(S(4))+theta2(S(5))))+
            (1/(theta2(S(2))+theta2(S(3))+theta2(S(4))))+
            (1/(theta2(S(2))+theta2(S(3))+theta2(S(5))))+
            (1/(theta2(S(2))+theta2(S(4))+theta2(S(5))))+
            (1/(theta2(S(3))+theta2(S(4))+theta2(S(5))))-
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))+theta2(S(4))))-
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(3))+theta2(S(5))))-
            (1/(theta2(S(1))+theta2(S(2))+theta2(S(5))+theta2(S(4))))-
            (1/(theta2(S(1))+theta2(S(5))+theta2(S(3))+theta2(S(4))))-
            (1/(theta2(S(5))+theta2(S(2))+theta2(S(3))+theta2(S(4))));
    end
end

```

```

end
%finds the sum of the substation response times
for p = 1:x
    lastsum1(p)=theta2(S(p));
end
lastsum=sum(lastsum1);
%the following 5 if statements find E[T] as
%defined in Ch3
if x==1
    EmxT(Sloop)=sum1;
end
if x==2
    EmxT(Sloop)=sum1-(1/lastsum);
end
if x==3
    EmxT(Sloop)=sum1-sum2+(1/lastsum);
end
if x==4
    EmxT(Sloop)=sum1+sum3-(1/lastsum);
end
if x==5
    EmxT(Sloop)=sum1+sum4+(1/lastsum);
end
end end
%calculates the 32 E[T], and multiplies by the probability of occurrence
for t = 1:32
    EmxTtimespi(t) = EmxT(t)*piofS(t);
end
%sums the terms, giving the expected maximum of T
sumEmxTtimespi=sum(EmxTtimespi); EmaxT = sumEmxTtimespi;
clear theta2;

```

Appendix C. LCOM Input Files

```
*****
* The following is the input file for the LCOM run of the base
* model. This input data was created with the aid of Mark
* Goldschmidt, ASC/ANMS WPAFB OH.
*****

15
15 BASE      A      30
15 DS        M      999
15 NS        M      999
15 M1        M      999
15 M3        M      999
15 M41       M      1
15 M42       M      3
15 M43       M      2
15 M44       M      1
15 M45       M      2
15 M5        M      6
15 M6        M      4

20
20 SYSTEM    NO-SRT  I  0
20 SYSTEM    DO-SRT  I  0

25
25 DO_SORTIE      31 2.00H OH  X      DS      1
25 NO_SORTIE      31      C      NS      1
25 PF             31 0.25H OH  X      M1      1
25 TS             21 0.50H OH  X      M3      1
25 DUMSRT         11
25 SORTIE         11
25 R41            23 2.2H  OH  X      M41     1
25 R42            23 2.27H OH  X      M42     1
25 R43            23 2.37H OH  X      M43     1
25 R44            23 1.5H  OH  X      M44     1
25 R45            23 1.19H OH  X      M45     1
25 FA            23 2.00H OH  X      M1      1
25 OR            23 2.00H OH  X      M1      1
25 OD            23 0.05H OH  X      M1      1
25 TURN_AROUND    31 .75H  OH  X      M5      1
25 MUNITIONS      31 0.50H OH  X      M6      1

30
30 MN00001  PF      N132948 D
30 N132948      N133056 E .05
30 N132948      N133531 E .95
30 N133056      ADNO-SRT  1
```

30	N133056	DUMSRT		N133057	S	
30	N133057	NO_SORTIE		N133058	D	
30	N133058	N133068		N134715	C	
30	N133068	TS			A 1.00	
30	N133068	OD		N133060	A .842	
30	N133060	OR			D	
30	N134715	N134717		N135359	C	
30	N134717	R41			A 0.17	
30	N134717	R42			A 0.39	
30	N134717	R43			A 0.21	
30	N134717	R44			A 0.27	
30	N134717	R45			A 0.46	
30	N134717	FA			A 0.16	
30	N134717				A 1.00	
30	N135359	TURN_AROUND		N135553	D	
30	N135553	MUNITIONS			D	
30	N133531				ADDO-SRT	1
30	N133531	SORTIE		N142146	S	
30	N142146	DO_SORTIE		N142147	D	
30	N142147			N135555	E 0.70	
30	N142147			N133048	E 0.30	
30	N133048	N133068		N134715	C	
30	N135555			N135359	D	
45						
45	*	12 12				
45	R	7				
45	DS	200 200				
45	NS	200 200				
45	M1	200 200				
45	M3	200 200				
45	M41	200 200				
45	M42	200 200				
45	M43	200 200				
45	M44	200 200				
45	M45	200 200				
45	M5	200 200				
45	M6	200 200				
55						
55	LANT	MN00001	NORMAL	NORMAL	SP1	BASE
60						
60	SP1	C	NORMAL			0.0
60		C A	NORMAL			0.0
75						
75F1	1	0310	BASE	LANT	1 30 0 .01M	C3.0 4.0 1

```

*****
* The following is the input file for the LCOM run of the ALS
* model. This input data was created with the aid of Mark
* Goldschmidt, ASC/ANMS WPAFB OH.
*****
15
15 ALS      A      30
15 DS       M      999
15 NS       M      999
15 M1       M      999
15 M3       M      999
15 M41      M      1
15 M42      M      3
15 M43      M      2
15 M44      M      1
15 M45      M      2
15 M5       M      6
15 M6       M      4
20
20 SYSTEM   NO-SRT   I 0
20 SYSTEM   DO-SRT   I 0
25
25 DO_SORTIE      31 2.00H OH   X      DS      1
25 NO_SORTIE      31          C      NS      1
25 PF             31 0.25H OH   X      M1      1
25 TS             21 0.50H OH   X      M3      1
25 DUMSRT         11
25 SORTIE         11
25 R41            23 2.2H  OH   X      M41     1
25 R42            23 2.27H OH   X      M42     1
25 R43            23 2.37H OH   X      M43     1
25 R44            23 1.5H  OH   X      M44     1
25 R45            23 1.19H OH   X      M45     1
25 TURN_AROUND    31 .75H  OH   X      M5      1
25 MUNITIONS      31 0.50H OH   X      M6      1
30
30 MN00001  PF      N132948 D
30 N132948      N133056 E .05
30 N132948      N133531 E .95
30 N133056      ADNO-SRT  1
30 N133056  DUMSRT  N133057 S
30 N133057  NO_SORTIE N133058 D
30 N133058  TS      N134715 D
30 N134715  N134717  N135359 C
30 N134717  R41      A 0.17

```

30	N134717	R42			A	0.39	
30	N134717	R43			A	0.21	
30	N134717	R44			A	0.27	
30	N134717	R45			A	0.46	
30	N134717				A	1.00	
30	N135359	TURN_AROUND	N135553	D			
30	N135553	MUNITIONS		D			
30	N133531			ADD0-SRT		1	
30	N133531	SORTIE	N142146	S			
30	N142146	DO_SORTIE	N142147	D			
30	N142147		N135555	E	0.70		
30	N142147		N133058	E	0.30		
30	N135555		N135359	D			
45							
45	*	12	12				
45	R		7				
45	DS	200	200				
45	NS	200	200				
45	M1	200	200				
45	M3	200	200				
45	M41	200	200				
45	M42	200	200				
45	M43	200	200				
45	M44	200	200				
45	M45	200	200				
45	M5	200	200				
45	M6	200	200				
55							
55	LANT	MN00001	NORMAL	NORMAL	SP1		ALS
55	DUMSRT	MN00001	NORMAL	NORMAL	SP1		ALS
60							
60	SP1	C	NORMAL				0.0
60		C A	NORMAL				0.0
75							
75F1	1	0310	ALS	LANT	1 50	0 TASKTIME	3.0 4.0 1
75F1	1	0310	ALS	DUMSRT	1 50	0 TASKTIME	3.0 4.0 1

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 074-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 25-03-2003		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From – To) Mar 2002 – Mar 2003	
4. TITLE AND SUBTITLE MODELS FOR SORTIE GENERATION WITH AUTONOMIC LOGSTICS CAPABILITIES				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Yager, Nicholas, A., 1 st Lieutenant, USAF				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 P Street, Building 640 WPAFB OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GOR/ENS/03-25	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT The primary objective of this research is to investigate the impact of an autonomic logistics system (ALS) on the sortie generation process for an individual airbase. As in some prior studies of this process, the methodology used to model the sortie generation process is a queueing network containing fork-join nodes for concurrent maintenance activities. The sortie generation rate is commonly regarded as the primary performance measure of the sortie generation process. This measure coincides with the throughput and is used to compare two models: i) pre-ALS operations and ii) ALS-enhanced airbase operations. Analysis of the models shows that the ALS model yields higher sortie generation rates under a variety of scenarios resulting from the differences in the sortie generation process that are inherent when an ALS is implemented. These results demonstrate that implementation of an ALS will positively impact the sortie generation process by increased sortie generation rates with equivalent or reduced resource levels.					
15. SUBJECT TERMS Queueing Theory, Aircraft Maintenance, Logistics, Operations Research					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Jeffrey P. Kharoufeh, Ph.D. (ENS)
U	U	U	UU	74	19b. TELEPHONE NUMBER (Include area code) (937) 255-6565, ext 4336; e-mail: Jeffrey.Kharoufeh@afit.edu