

UNIVERSITY OF CALGARY

**Single-Stage Queueing Theory in Flexible
Manufacturing Systems (FMS)**

by

Jonathan Blankson

A THESIS

**SUBMITTED TO THE FACULTY OF GRADUATE STUDIES IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE**

DEPARTMENT OF MATHEMATICS AND STATISTICS

CALGARY, ALBERTA

APRIL, 2003

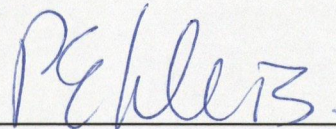
© Jonathan Blankson 2003

UNIVERSITY OF CALGARY
FACULTY OF GRADUATE STUDIES

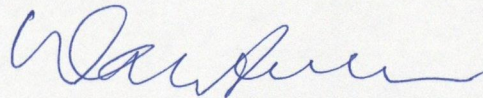
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Single-Stage Queueing Theory in Flexible Manufacturing Systems (FMS)" submitted by Jonathan Blankson in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE.



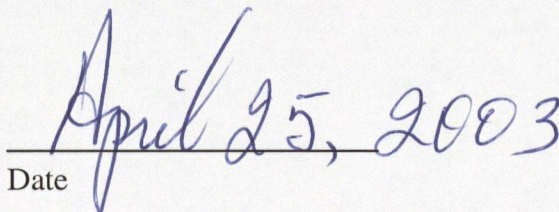
Supervisor, Dr. Ernest G. Enns
Department of Mathematics and Statistics



Dr. Peter F. Ehlers
Department of Mathematics and Statistics



Dr. Walter W. Zwirner
Department of Applied Psychology


Date

Abstract

The scope of this thesis is limited to analytical queueing theory models applied directly to discrete parts manufacturing systems. Furthermore, only closed queueing networks that can model flexible manufacturing systems of realistic size are considered.

The results of this work relate to the problem of determining the equilibrium distribution of parts (customers) in closed queueing systems composed of M interconnected stages of service centers. The number of parts, N , in a closed queueing system is fixed since parts pass repeatedly through the M stages with neither entrances nor exits permitted. At the i th stage there are c_i parallel exponential servers with mean service rate μ_i for each server. When service is completed at stage i , a part proceeds directly to stage j with probability p_{ij} . Such closed systems are said to be stochastically equivalent to open systems in which the number of parts cannot exceed N .

By means of numerical examples, performance measures of the analytical models developed are explored through a C++ program written for some algorithms. The algorithm values are compared with exact performance values and the results are presented. Also, this thesis presents flexibility measures through computational examples and definitions of single-machine and machine-group flexibility relative to task sets.

Acknowledgements

I give thanks to the Almighty God for the grace and mercy He granted me throughout my studies on U of C campus. To Him be the glory.

A special word of gratitude goes to my supervisor, Dr. E. G. Enns of the Department of Mathematics and Statistics, for his constant support, advice, and encouragement in bringing this thesis work to a successful completion. My thanks also go to the members of my committee. Their suggestions and comments helped improve the quality of this thesis.

My sincerest thanks go to my professors and all the staff of the Department of Mathematics and Statistics for their constant support and advice. Financial assistance provided by the University of Calgary is also well appreciated and thankfully acknowledged.

Special thanks go to my wife, Mary Blankson for the love, support and encouragement she gave. I thank all my family, colleagues, and friends, especially all the members of the North West Bible Study Group for their dedicated support during my graduate study.

Table of Contents

Approval Page	ii
Abstract	iii
Acknowledgements	iv
Table of Contents	v
List of Tables	vii
List of Figures	viii
1 Introduction	1
1.1 Thesis Objective	3
1.2 Thesis Outline	3
2 Characteristics of FMS	5
2.1 Introduction	5
2.2 What is an FMS?	5
2.3 System Machines	6
2.4 Material Handling System	6
2.5 Work Stations	7
2.6 Information and Control Systems	7
2.7 Flexibility	8
2.8 A Typical FMS Layout	9
3 Planning and Control of FMS	10
3.1 Introduction	10
3.2 System Design	10
3.2.1 Initial Specification Decisions	11
3.2.2 Subsequent Implementation Decisions	12
3.3 System Planning	13
3.4 Scheduling and Control	15

4	Exact FMS Models	18
4.1	Introduction	18
4.2	Single-Stage Closed Jackson Queueing Network Model	19
4.3	Mean Value Analysis	26
4.4	General Single-Stage Closed Jackson Queueing Network Model	27
5	Illustrative Numerical Examples	29
5.1	Introduction	29
5.2	Statement of Problem 1	29
5.3	Results of Problem 1	30
5.4	Statement of Problem 2	31
5.5	Results of Problem 2	32
5.6	Statement of Problem 3	33
5.7	Results of Problem 3	35
6	Measures of Flexibility in FMSs	37
6.1	Introduction	37
6.2	Measures of Single-Machine Flexibility Relative to a Finite Task Set	38
6.3	Measures of Group-Machine Flexibility Relative to a Finite Task Set	41
7	Summary and Discussion	47
Appendix A: The C++ program for Problem 1 using Algorithm 4.1 (MVA)		52
Appendix B: The C++ program for Problem 1 using Algorithm 4.2 (EMVA)		56
Appendix C: The C++ program for Problem 2 using Algorithm 4.1 (MVA)		60
Appendix D: The C++ program for Problem 2 using Algorithm 4.2 (EMVA)		64
Appendix E: The C++ program for Problem 3 using Algorithm 4.1 (MVA)		68
Appendix F: The C++ program for Problem 3 using Algorithm 4.2 (EMVA)		72
Bibliography		76

List of Tables

Table 1: A comparison of MVA approximation and exact performance – Problem 1	30
Table 2: A comparison of EMVA approximation and exact performance – Problem 1 ...	31
Table 3: Values we put in the C++ program found in appendix C	32
Table 4: A comparison of MVA approximation and exact performance – Problem 2	32
Table 5: Values we put in the C++ program found in appendix D	32
Table 6: A comparison of EMVA approximation and exact performance – Problem 2 ...	33
Table 7: Values we put in the C++ program found in appendix E	35
Table 8: A comparison of MVA approximation and exact performance – Problem 3	35
Table 9: Values we put in the C++ program found in appendix F	36
Table 10: A comparison of MVA approximation and exact performance – Problem 3 ...	36
Table 11: Machine flexibility	40
Table 12: Probabilities of p_{ij} of using m_i for task j	44
Table 13: Conditional probabilities	44
Table 14: Conditional expectation of machine-task efficiency	45
Table 15: Summary of the calculation of group flexibility	46

List of Figures

Figure 1: A Typical FMS Layout	9
Figure 2: A Closed Queueing Network	34

Chapter 1

Introduction

Queueing has become a way of modern life which we encounter at every step in our daily activities. This has made the study of queueing become more interesting and important as it provides both a theoretical background to the type of service that we may expect from a facility and the way in which the facility itself may be structured to respond to some specified quality of service to its users (customers). Application of queueing theory provides the theoretical framework for the design and study of flexible manufacturing systems (FMSs).

“A flexible manufacturing system (FMS) is a production unit capable of producing a range of discrete products with a minimum of manual intervention. It consists of production equipment workstations (machine tools or other equipment for fabrication, assembly, or treatment) linked by a materials-handling system to move parts from one workstation to another, and it operates as an integrated system under full programmable control.”¹

Generally, modeling approaches in flexible manufacturing systems can be categorized into two broad classes namely, analytical method and simulation method. This thesis only presents analytical modeling methods which have been used to study the behavior of queues in flexible manufacturing systems with emphasis on closed Jackson queueing

¹ M. E. Marchant, personal communication, Oct. 12, 1983. Adapted from a definition developed by the International Institution for Production Engineering Research.

networks. The earliest works in this area were by Jackson (1954)(1956) to evaluate service standards in aircraft maintenance. Gordon and Newell (1967) extended the work of Jackson to more general networks of closed queues where the customers, after finishing service at one station, could go to another station with a given probability. They showed that a product-form solution, obtained for open queueing networks, also holds for closed queueing networks, the difference being that, in this case, there is a normalization constant that is considered dependent on the entire network and the problem is, for large systems it becomes computationally expensive to obtain the normalization constant.

Buzacott and Shanthikumar (1993) reviewed queueing models that can be used to design flexible manufacturing systems. They devised efficient algorithms which reduced the computational difficulty of the normalization constant and relevant performance measures. Buzacott and Yao (1986) reviewed the development of analytical models for flexible manufacturing systems. They summarized the contributions of various groups at the time and outlined the directions in which the models needed extension.

As a result of increased automation and the trend towards an ever shorter life cycle for a product, it has become apparent that the flexibility of the machinery needed for complex production processes is now very important for long-term profitability. Brill and Mandelbaum (1987)(1989) defined measures of flexibility which take into account how well a machine or group of machines can perform relative to a background task set.

1.1 Thesis Objective

The objective of this thesis is to develop analytical queueing theory models applied directly to discrete parts manufacturing systems with emphasis on only single-class closed queueing networks that can model flexible manufacturing systems. Furthermore, both the performance measures and the flexibility measures will be explored through numerical examples and the results presented.

1.2 Thesis Outline

- 1) Chapter two deals with description of the fundamental components and characteristics of a flexible manufacturing system with a typical layout example.
- 2) Chapter three discusses some vital decisions which should be made in relation to design, planning, and scheduling and control before any flexible manufacturing system can be established and implemented.
- 3) Chapter four considers single class closed queueing network models that have a product-form solution and can be analyzed exactly, provided suitable simplifying assumptions are satisfied. Jackson's closed network theorem is introduced. This chapter also presents the mean value algorithm and the extended mean value algorithm as defined by Buzacott and Shanthikumar (1993) for the solution of simple closed queueing networks when there is only a single server at a service center.
- 4) Chapter five deals with computational examples to measure the performance of the analytical models developed in chapter four and the results presented. The C++ program codes for these examples can be found in the appendices.

- 5) Chapter six focuses on the flexible manufacturing system flexibility measures relative to a task set for single machine and group of machines as defined by Brill and Mandelbaum (1987)(1989) with computational examples.
- 6) Chapter seven is the summary and discussion.

Chapter 2

Characteristics of FMS

2.1 Introduction

In this section we will first define flexible manufacturing system (FMS) and then describe the fundamental components of FMS.

2.2 What is an FMS

Flexible manufacturing system is a computer-controlled group of numerically controlled machines (fully automated machines) with supporting work stations which are connected by automated material handling systems and are controlled by a central computer. For instance, if an FMS is designed to manufacture cars, then a production planner responsible for production planning gets information on market demands, production capacity, current production levels, raw materials needed, etc. to determine specification of the sequence of operations required for converting raw materials into parts and then assembling parts into products. The task plan is then divided into subtask plans to obtain short-term schedules showing, for each service center, its goals for the next point of action. The jobs assigned to the service center are then sequenced by the order in which they will be loaded onto the machines. The material handling system transports parts on pallets between work stations. The central computer (controller) keeps track of the system's status. The controller

downloads commands to the individual system components based on current status and production plans. The processed parts pass through the inspection unit for quality standard check before moving on to assembly work station to be assembled into finished products.

2.3 System Machines

A typical FMS consists of computer-controlled machines that are used for value-added production operations. Machine operations such as milling, shaping, turning, etc. are carried out to produce parts such as engine blocks, pumps, etc. (which have irregular shapes) and shafts, rings, etc. (which have regular shapes). The machines are provided with tool magazines (storages) and automatic tool changers to allow many operations to be performed on a part each time it is available to the machine. A particular work piece may have to visit more than one machine to have required operations performed. So, many parts can be machined without tooling changeovers.

2.4 Material Handling System

One important aspect of FMS is the material handling systems. They are automated to handle work pieces both within and between work stations. They act as the circulatory system of the whole plant system, distributing vital material to all of plant's cells. A manufacturing cell (plant's cell) is a cluster or collection of machines designed and arranged to produce a specific group of component parts. There are many kinds of material handling systems such as rail mounted carts, conveyors, tow carts and automatic guided vehicles (AGVs). But AGVs are now more popular than the others. An AGV is a self-powered electrically driven vehicle which transports a pallet containing one or more

fixtured parts for work piece to rest on. A work piece is one of the parts (jobs) to be processed which arrive in batches to a work station for processing. Work piece is transported on pallet to a fixture on the machine which is specially designed for the work piece. The fixture holds the work piece in a specific orientation for the machine to operate on it. Loading and unloading parts at machines are done automatically by pallet changers. To avoid blockage there may be the need for different types of pallets or pallets with fixtures of a given work piece to be set in place. The material handling system of FMSs permits work pieces to visit machines in any sequence and imposes no constraint on the number of visits of work pieces to machines.

2.5 Work Stations

Another vital component found in FMSs is a number of supporting work stations where operations such as milling, drilling, washing parts, inspecting parts, or measuring parts are performed. An automatic washer station is where all the cleaning of parts for machining is done. Inspection and measurement of parts are done at the coordinate measuring machine station. As loading and unloading station is used to enter and remove parts from the system, statistical quality control station is used to make sure that all manufactured products meet standard specifications.

2.6 Information and Control Systems

Information and control systems consist of either one central computer or several computers with programmed controllers that can exchange data and commands. The systems keep track of information on the status of jobs, machines operation, and material

handling systems, and also control the processing of jobs to and from the system. An attendant loads the computer with data and keeps track of problems to rectify or makes changes where necessary. The controller downloads commands to the individual system components according to production plans and current production status. The components respond and execute the command or show that they failed to execute. The controller controls loading and unloading of parts onto and moving between machines. It also controls tools path and automatic change of tools.

2.7 Flexibility

According to Askin and Standridge (1993), “flexibility refers to the ability to handle different product sizes, shapes, weights, paths, and volumes with the same equipment.” There are relatively many advantages of an FMS due to its flexibility nature. Some of the benefits of an FMS include the ability to respond to changing markets, and to quickly and efficiently incorporate design or process changes, or to use new materials. Flexible systems in an FMS permit multiple operations to be performed on a work piece. Machine tools of the system are computer controlled; hence the system is flexible to produce a variety of parts by a simple change of controller software. Also, with tool automated changing capability, setup time is eliminated, enhancing machine productivity and workflow characteristics. Furthermore, the system can respond flexibly to unforeseen activities, such as machine breakdown and temporary overloads, by dynamically rerouting work pieces to the closest available machine with the necessary tooling to alleviate potential bottlenecks. These flexibilities, among others, promise productivity improvements through increasing

machine utilization and, at the same time, reducing levels of work-in-process (WIP) and production cycle time.

2.8 A Typical FMS Layout

Generally, the number of pallets that move the parts in an FMS is assumed to be fixed, so the number of parts that the system can accommodate at a time is also fixed. To maximize the throughput (i.e., the number of parts processed per unit time), a part is replaced by a new part as soon as it finishes processing and leaves the system. This can be considered a closed system because the number of parts in the system can be assumed to be fixed. This characteristic allows these systems to be modeled as closed queueing networks. Figure 1 show a typical FMS layout which can be modeled by a closed queueing network.

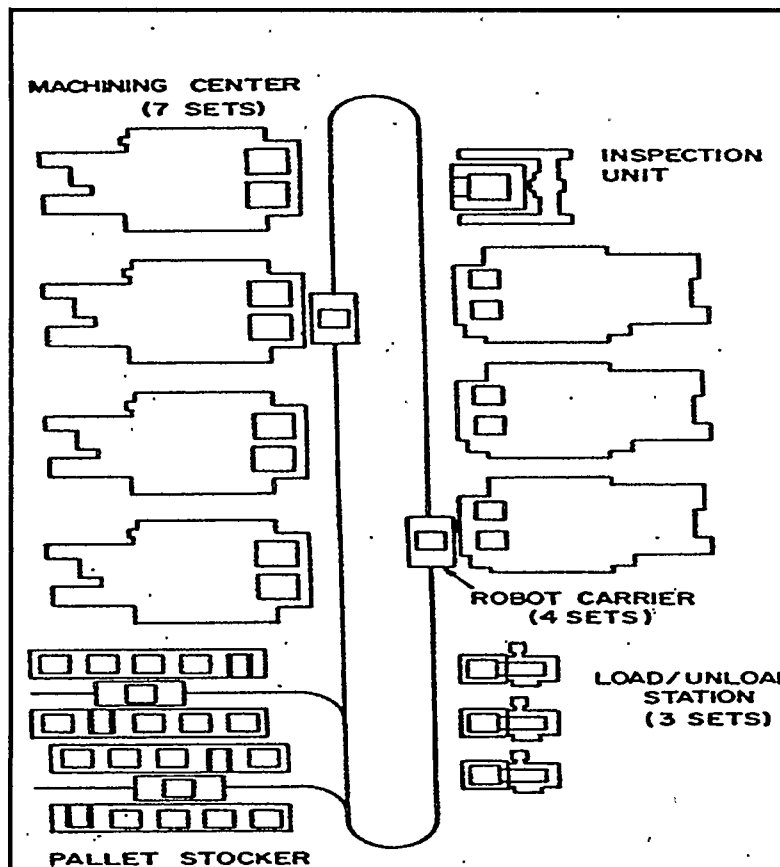


Figure 1: A Typical FMS Layout (Figure, courtesy of Arbel and Seidmann, 1984)

Chapter 3

Planning and Control of FMS

3.1 Introduction

For an FMS to work effectively to yield the desired result, some vital decisions should be made. In this section we will discuss some strategic decisions relative to design, planning, and scheduling and control.

3.2 System Design

In designing an FMS one starts with a specification having few or moderate details and creates certain design proposals. It is important to carry out some preliminary calculations to check the feasibility of the proposals against the specification. A design team must carefully ensure their proposals are technically sound, offer economic benefits, and are financially justifiable before establishing more details and implement. In light of these, some vital decisions have to be made and these have been divided into *initial specification decisions and subsequent implementation decisions* (Stecke, 1985).

3.2.1 Initial Specification Decisions

1. Determination of types of parts to be processed on the FMS

It must be decided which part types will be made on the system so that parts (identical part type) which share common tools, pallets, and fixtures or use the same materials can be grouped for appropriate machines to operate on them. This makes the system economical and also improves productivity.

2. Specification of the number and type of processing machines and their tools needed

Since the main aim is to make the FMS as flexible as possible the machine types and their tools ought to be selected in order to enable the system offer adequate flexibility in its operation. There should be a processing plan, which identifies the needed operations on each type of parts and the volume of parts required so that the number of machine types and tools needed to perform the operation can be determined. The number of machine types in the system normally depends on the capacity of the tool magazine.

3. Determination of the type of material handling system

The types of material handling system should be specified among rail-mounted carts, roller conveyors, tow carts, AGVs and so on. The choice is usually based on the methods through which the designer wants the system to handle materials efficiently and also the capital availability for its implementation. Some factors, which influence the choice, are the size and weight of the parts, and the volume of material handling.

4. Provision of tool and part system

Automatic tool and part changing facilities and the necessary standard interfaces to other materials handling system should be provided for efficient running of the system.

5. The materials storage system

This system could be a central storage within the system or/and a computer controlled warehouse. It should communicate with the material handling and data processing systems. It should be able to store rough materials, semi-finished and finished work pieces, tools, fixtures and other necessary components, together with spare parts.

6. Specification of computers and control system components

The type of computers with specific software/hardware to control the system should be determined. Normally this should consist of several computers, which keep track of information on jobs in process, machines operation, and material handling systems, and should be linked to a central computer, which controls the whole system. This computer network should be linked with sensory-based diagnostic feedback systems in operation in each cell to detect faults and errors in case of breakdowns and report them automatically.

3.2.2 Subsequent Implementation Decisions

1. Systematic Layout by which the FMS will be configured

The systems components such as machines, material handling systems, work stations, etc. should be properly laid out to avoid job delays which in effect reduces production cycle time.

2. Determination of number and type of pallets and fixtures

Decision on how many pallets and fixtures type should be in the system is crucial because it determines the number of work pieces and the types of work pieces which should be in operation at a time.

3. Specification of work methods

The overall procedure that involves a specific and well-defined method of work should be determined. This involves the description of the way in which a workshop and all its equipment to be used functions. In line of this comes feasibility study whose aims are to establish that the project is technically possible, to make the initial choices and to determine a financial justification envelope or overall plan with maximization of profit in mind.

4. Specification of information management system

Determination of software for production planning, job scheduling, data processing, data management, and control need to be specified. They should be compatible so that future developments would not be higher capital investment or be impossible.

3.3 System Planning

A good planning procedure must take account of possible alternative routes taken by the part through the workshop. It becomes an element of forecast planning for the part and thus a sub-system in itself for production management. The two main planning in FMS are process planning and production planning. Process planning in FMS involves determining the route and the plan of activities for manufacturing, inspecting, assembling, etc. a work piece. Production planning entails the implementation of a process plan in a defined

environment for all parts to be produced and scheduled according to the manufacturing facilities and other resources of the factory. Therefore, planning decisions must be made before work pieces are released to the system for production to commence.

1. Decision on the technology to be used

Technology is a factor that is often not clearly defined in the manufacturing plan. It is important that manufacturing have a clear understanding of the technologies that will be required and that new technologies be introduced carefully, based on well thought out planning. It is well known that a smaller FMS is easier to manage and control than a larger FMS, so the designer should consider such options. There should be a dynamic routing (i.e., deciding which operation should be performed next by which processor depending on current machine work loads) to improve performance. Also, more tools and programs should be assigned to the set of tasks that require higher processing times.

2. Selection of group of part types for simultaneous production

A collection of part types along with the machines and tooling required to produce them should be grouped. To the extent possible, each part should be made totally within its group. Each parts group visit different group of machine types for simultaneous production of all part types in order to avoid machine idleness (i.e. total machine utilization). Parts should be dispatched to the system to match demand while keeping in mind system capacity. When the system requires frequent changeovers, the dispatching problem dominates.

3. Sequencing of operations

Once in the system, a part must often visit a number of different machines before processing is complete. The sequence of the separate machine visits could be chosen to enhance the performance of the system.

4. Operation and frequency selection for quality check

There should be a decision on the type of measuring machine to monitor the quality of the parts being processed as well as the processes themselves. The intelligent selection of operations to measure and the frequency with which to measure them is required in order to ensure that quality standards are satisfied and that processing errors are quickly identified.

5. Decision on human resource

Implementation of any plan for automation is dependent on the availability of the technical skills and talents of people. It is necessary to have a clear understanding of the human resources, technical and non-technical, salaried and hourly that will be available to undertake implementation of the FMS.

3.4 Scheduling and Control

An FMS needs to perform operations under the control of a dynamic scheduling system. This means that decisions concerning what work piece is manufactured next on which cell, are made close to the operation currently being performed by the particular cell. In other words, a complete FMS schedule is not made in advance because it must be capable of responding to real-time decisions. The need to schedule the FMS for maximum effectiveness is great due to the high capital investment involve for such manufacturing processes. Since the objective of an FMS is to respond quickly to changes in customer

demand without carrying large finished goods inventory, dynamic scheduling decisions are necessary to maintain high system effectiveness. Also, there should be a control system which keeps track of production to make sure that all the production goals and targets are being met as scheduled.

1. Part sequence into FMS

Since an FMS can process a number of different parts and since these parts are required in certain ratios relative to one another, active control of the part input sequence is required. Workloads must be balanced so that all machines finish their work for each batch more or less together and new batch can start immediately.

2. Sequencing of fixturings

Many parts must make a number of passes through the system in order to process different sides. The sequence for these separate passes could be chosen to enhance the performance of the system.

3. Cart choice and movement

FMSs employ a number of separate carts for transporting parts from machine to machine. When the need arises for transporting a part, a choice among the carts of the system must be made. Carts are always moving, except while undergoing load/unload operation or while queueing at an occupied cell node. Shortest routes are chosen when there is a destination. Deadlocks are checked for periodically. Deadlocks refer to any blockages (congestion) which may occur from time to time.

4. Total number of pallets and each pallet type in the system

Generally, increasing the supply of pallets will increase the rate at which parts flow through the system and vice versa. This is because having more pallets increases the probability that a part will always be ready for processing when a machine becomes idle. However, as more and more pallets are added to a system with limited storage capacity, the resulting congestion may actually reduce throughput. Most FMSs operate in a cycle mode; when a pallet comes out of the system, a part is chosen that can be fixtured on the pallet and the pallet is then sent back in. If more than one part type can use a given pallet type, the part that is most behind in production is usually chosen.

5. Part priorities

To make room for priority jobs, some control systems allow the operator to fine tune part priority above and beyond allocation of resources, i.e. machines, pallets, etc. The basic mechanism is to alter the processing order so that certain parts waiting to be processed by a particular machine can be processed first.

Chapter 4

Exact FMS Models

4.1 Introduction

Models are required to predict the performance of the system and also to give the designer insight into the technical issues related to the system. Analytical models can be developed for performance evaluation. In order to solve the models they require simplifying assumptions but they have the advantage that assumptions can be made explicit and it is not difficult for other analysts to check the development of the model solution and validate it. The existence of an analytical model enables simulation model validity to be checked. These analytical models also enable the effect of different assumptions to be found. In this section we shall describe the process of developing and using some analytical models by using closed Jackson queueing networks.

Assumptions:

1. The system has $M = (1, 2, \dots, m)$ distinct service centers with c_i identical servers at each service center.
2. The maximum number of parts allowed in the system simultaneously is N .
3. Parts that arrive when there are N parts already in the system wait in an external queue and are released to the system as soon as space is available.
4. The distribution of service time at server i is exponential; i.e. $F_i(t) = 1 - e^{-\mu_i t}$, where μ_i is the service rate of server i .

5. Parts are served according to a first come, first served (FCFS) service policy.
6. Parts arrive at the system as a Poisson process with parameter λ .
7. Parts arrive in batches; the batches of the r types of parts are pre-specified.
8. All these part types (the same class) can be loaded onto the same type of pallets.
9. The service center 0 acts as the load/unload server.

4.2 Single-Stage Closed Jackson Queueing Network Model

The analysis of FMS using closed queueing networks was the concept of cyclic networks. In a cyclic network, the number of parts and the machines in the system as well as the sequence in which the parts visit the machines for processing is fixed. After finishing processing on the last resource, the parts return to the first resource, forming a cycle.

Here we want the input process to maintain the number of parts at constant value of N . And each departing part is replaced immediately by a new part, thus we model this by a single-class closed queueing network with a total of N parts in it. The fraction of parts that will join machine center i on their arrival is γ_i , $i = 1, \dots, m$.

$$\left(\sum_{i=1}^m \gamma_i = 1 \right).$$

We must specify how parts (customers) determine the next queue they will visit when they have completed service at a particular queue. For all the models considered here, we assume that the selection of the next queue parts will visit is random and based on a discrete probability density function. The part that completes service at machine center i proceeds to station j ($\neq i$) with transfer probability matrix $\mathbf{P} = (p_{ij})_{i,j=0,\dots,m}$. Thus the part departs the system entirely with probability $p_{i0} = q_i = 1 - \sum_{j=1}^m p_{ij}$, $i = 1, \dots, m$. The

probability measure gives a great deal of control on how the parts move about the system. For the complete network of M queues, these probabilities can be seen as an $M \times M$ matrix $\mathbf{P} = [p_{ij}]$. Since a part cannot disappear after leaving a queue, the rows of this matrix must sum to 1 (i.e. the matrix is stochastic). The average number of times a part visits machine center i during its stay in the system,

$$v_i = \gamma_i + \sum_{j=1}^m v_j p_{ji}, \quad i = 1, \dots, m; \text{ where } v_0 = 1.$$

So the visit ratios are simply the solution to the matrix equation $\mathbf{v} = \boldsymbol{\gamma} + \mathbf{vP}$, where \mathbf{v} is a vector representing the visit ratios of parts to machine centers, $\boldsymbol{\gamma}$ is a vector representing the fraction of parts that will join machine centers on their arrival, and \mathbf{P} is a transfer (routing) probability matrix; it follows that $\mathbf{v} = (\mathbf{I} - \mathbf{P})^{-1} \boldsymbol{\gamma}$, where $v_0 = 1$. The square matrix \mathbf{P} is nonsingular since its rank is equal to the number of columns it has; that is, the columns of \mathbf{P} are linearly independent. Note that $\mathbf{vP} = v_1 p_1 + v_2 p_2 + \dots + v_m p_m$, where p_i is the i th column of \mathbf{P} , so that the condition of nonsingularity is just the statement that the columns of \mathbf{P} are linearly independent. Note also that for the condition of nonsingularity to hold there should be at least three machine/service centers (i.e., $m \geq 2$ for $i = 0, 1, \dots, m$) in the system.

We will assume that all self-transitions have been eliminated so that $p_{ii} = 0, i = 1, \dots, m$.

The service times of parts at machine center i are iid exponential random variables with mean $1/\mu_i$, $i = 0, 1, \dots, m$. All the service times and the arrival times are mutually independent. The rate at which a part is processed at machine center i when there are n parts is assumed to be $r_i(n)$, $n = 1, \dots, N$. This allows us present, as special cases, single or

multiple machines in parallel at the machine center i . Specifically if there are c_i machines in parallel at machine center i , we set

$$r_i(n) = \min\{n, c_i\}, \quad n = 1, \dots, N; \quad i = 0, 1, \dots, m.$$

Let us assume that at time t , machine center i , $i = 0, 1, \dots, m$, contains $A_i(t)$ individual parts, so that the state of the system may be represented as the vector

$\mathbf{A}(t) = (A_0(t), A_1(t), \dots, A_m(t))$. We assume for simplicity that the process $\mathbf{A}(t)$ is Markovian, taking values in the set $N_N = \{ \mathbf{n} = (n_0, n_1, \dots, n_m) : n_i = 1, \dots, N \text{ for } 0 \leq i \leq m \}$ of sequences of non-negative integers.

Define the stationary distribution of \mathbf{A} by $p(\mathbf{n}) = \lim_{t \rightarrow \infty} p\{\mathbf{A}(t) = \mathbf{n}\}$, $\mathbf{n} \in N_N$ assuming its existence. The inflow into state \mathbf{n} can occur from state $\mathbf{n} + \mathbf{e}_j - \mathbf{e}_i$ owing to a service completion of a part at machine center j ($j = 0, 1, \dots, m; j \neq i$) that joins machine center i directly; $i = 0, 1, \dots, m$. Here \mathbf{e}_i is the i th unit vector. Now, equating the rates of probability inflow and outflow of state \mathbf{n} , one gets the following balance equations:

$$\sum_{j=0}^m \sum_{i=0}^m \mu_j r_j(n_j + 1) p_{ji}(\mathbf{n} + \mathbf{e}_j - \mathbf{e}_i) = \sum_{i=0}^m \mu_i r_i(n_i) p(\mathbf{n}), \quad \mathbf{n} \in N_N \quad \dots\dots\dots(4.1)$$

We can get p using equation 4.1 along with the normalizing equation

$$\sum_{\mathbf{n} \in N_N} p(\mathbf{n}) = 1.$$

Theorem 4.1

The equilibrium distribution of an irreducible closed Jackson network with N jobs is

$$p(\mathbf{n}) = B_N \prod_{i=0}^m \frac{v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)}, \quad \mathbf{n} \in N_N, \quad \dots\dots\dots(4.2)$$

$$\text{where } B_N = \frac{1}{\sum_{n \in N_N} \prod_{i=0}^m \frac{v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)}} , \dots\dots\dots(4.3)$$

is the appropriate normalizing constant.

Proof of Theorem 4.1

The process has at most one equilibrium distribution, and any such distribution

$p = (p(n) : n \in N_N)$ satisfies the equations 4.1. We will try to see whether a product form

solution of the form $p(n) = B_N \prod_{i=0}^m p_i(n_i)$ will lead to a consistent solution for equation 4.1.

Substituting this product form in equation 4.1 and dividing by $p(n)$ we get

$$\sum_{j=0}^m \sum_{i=0}^m \mu_i r_j(n_j + 1) p_{ji} \frac{p_j(n_j + 1)}{p_j(n_j)} \frac{p_i(n_i - 1)}{p_i(n_i)} = \sum_{i=0}^m \mu_i r_i(n_i) , n \in N_N, \dots\dots\dots(4.4)$$

Equating term by term in equation 4.4 for each i we get

$$p_i(n_i - 1) \sum_{j=0}^m \mu_i r_j(n_j + 1) \frac{p_j(n_j + 1)}{p_j(n_j)} p_{ji} = \mu_i r_i(n_i) p_i(n_i), \quad i = 1, \dots, m; n \in N_N, \dots\dots\dots(4.5)$$

Therefore, if equation 4.5 has a consistent solution for $p_i, i = 0, 1, \dots, m$, it is also a solution to equation 4.4. For this to be true the term

$$\sum_{j=0}^m \mu_i r_j(n_j + 1) \frac{p_j(n_j + 1)}{p_j(n_j)} p_{ji} = \hat{v}_i \text{ say, should be independent of } n_j, \dots\dots\dots(4.6)$$

So, substituting \hat{v}_i in equation 4.5, we get

$$p_i(n_i - 1) \hat{v}_i = \mu_i r_i(n_i) p_i(n_i), \quad n_i = 1, \dots, N; i = 0, 1, \dots, m. \dots\dots\dots(4.7)$$

substituting equation 4.7 in equation 4.5, we get

$$p_i(n_i - 1) \sum_{j=0}^m \hat{v}_i p_{ji} = \mu_i r_i(n_i) p_i(n_i), \quad i = 0, 1, \dots, m. \dots\dots\dots(4.8)$$

Because $\hat{v}_i, i = 0, 1, \dots, m$ with $v_0 = 1$ defined earlier is the solution to

$\hat{v}_i = \sum_{j=0}^m \hat{v}_j p_{ji}, i = 0, 1, \dots, m$, setting $\hat{v}_i = v_i$ we see that the solution to

$$p_i(n_i - 1)v_i = \mu_i r_i(n_i)p_i(n_i), n_i = 1, \dots, N \dots \dots \dots (4.9)$$

is consistent with equations 4.7 and 4.8 and therefore

$p(n) = B_N \prod_{i=0}^m p_i(n_i), n \in N_N$ is a solution to equation 4.1.

$$\text{Because } p_i(n_i) = \frac{v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)}, n_i = 0, 1, \dots, N; i = 0, 1, \dots, m \dots \dots \dots (4.10)$$

satisfies equation 4.7 we see that

$$p(n) = B_N \prod_{i=0}^m \frac{v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)}, n \in N_N,$$

$$\text{where } B_N = \frac{1}{\sum_{n \in N_N} \prod_{i=0}^m \frac{v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)}}$$

Hence the proof is complete. (Buzacott and Shanthikumar (1993))

Now, we will treat the entire system as an $M/M(N)/I$ queueing system. Parts arrive at the rate $\lambda_i = \lambda v_i$. Parts are served (state-dependent) at the rate $\mu_i r_i(n_i)$ when there are n_i parts in the system ($n_i = 1, 2, \dots$). For this model, the level crossing rate balance equation for the stationary distribution $P\{X_i = n_i\}, n_i = 0, 1, \dots$, is

$$\lambda v_i p\{X_i = n_i\} = \mu_i r_i(n_i + 1) p\{X_i = n_i + 1\}, n_i = 0, 1, \dots \dots \dots (4.11)$$

where X_i is a random variable representing the stationary distribution of the number of parts in the $M/M(N)/I$ queueing system and $\mu_i r_i(n_i) = \min\{n_i, c_i\} \mu_i$ is the rate at which service completions occur when there are n_i parts in the system. That is, we can have c_i parallel

servers at service center i , such that $\lambda_i < \mu_i c_i$ or equivalently $\lambda < \mu_i c_i / v_i$. Solving equation

4.11 with the normalizing condition $\sum_{n_i=0}^{\infty} p(n_i) = 1$,

$$\text{one obtains } P\{X_i = n_i\} = \frac{\lambda^{n_i} v_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i r_i(j)} P\{X_i = 0\}, \quad n_i = 0, 1, \dots, \dots \dots \dots (4.12)$$

The distribution of X is the stationary distribution of an open Jackson queueing network with a set of service centers $\{0, 1, \dots, m\}$ and an arbitrary part visiting service center i on the average v_i number of times before it leaves the system ($i = 1, \dots, m$) and the load/unload center twice ($= 2v_0$). If we observe the open queueing network only when there are a total of N parts in it, then the distribution of the number of parts in the closed queueing network model is the same as that of the open queueing network. Now, we need to compute the convolution of the probability distributions,

$$P\{X_i = n_i\} = P\{X_i = 0\} (\lambda v_i / \mu_i)^{n_i} / \prod_{j=1}^{n_i} r_i(j), \quad i = 0, 1, \dots, m. \dots \dots \dots (4.13)$$

to get the probability distribution of the number of parts in the closed queueing network. The below algorithm by Buzacott and Shanthikumar (1993) will compute this convolution and the stationary distribution of the number of parts in the closed queueing network.

Algorithm 4.0 Convolution Algorithm

Step 1: Set $p_i(0) = 1$, $i = 0, 1, \dots, m$

Step 2: For $i = 0, 1, \dots, m$

For $n = 0, 1, \dots, N-1$, set $p_i(n+1) = p_i(n) v_i / (\mu_i r_i(n+1))$

Step 3: Set $\hat{p}(n) = p_0(n)$, $n = 0, 1, \dots, N$

For $i = 1, \dots, m$

For $n = 0, 1, \dots, N$, set $\hat{q}(n) = \sum_{\ell=0}^n \hat{p}(\ell) p_i(n-\ell)$

For $n = 0, 1, \dots, N$, set $\hat{p}(n) = \hat{q}(n)$

Step 4: $p(n) = (1/\hat{q}(N)) \prod_{i=0}^m p_i(n_i)$, $n \in N_N$

Step 5: Stop.

Various performance measures can be computed from the model. For example, the expected number of parts at service center i , is

$$E[N_i(N)] = \text{Th}(N) \sum_{n_i=0}^{N-1} \frac{(n_i+1)v_i}{\mu_i r_i(n_i+1)} p_i(n_i; N-1), \quad i = 0, 1, \dots, m. \quad (4.14)$$

where $\text{Th}(N)$ is the throughput rate, thus $\text{Th}(N) = \mu_0 E[r_0(N_0(N))]$ is the rate at which parts are being loaded/unloaded at the load/unload service center.

The long-term expected time of an arbitrary part in service center i , is

$$E[T_i(N)] = \sum_{n_i=0}^{N-1} \frac{n_i+1}{\mu_i r_i(n_i+1)} p_i(n_i; N-1), \quad i = 0, 1, \dots, m. \quad (4.15)$$

$$\text{The throughput rate, } \text{Th}(N) = \frac{N}{\sum_{i=0}^m v_i E[T_i(N)]}, \quad i = 0, 1, \dots, m. \quad (4.16)$$

where $\sum_{i=0}^m E[N_i(N)] = N$.

4.3 Mean Value Analysis (MVA)

Mean value analysis of queues is an approach which can be used to study queueing networks with product-form solutions. With this procedure, operations are measured in terms of mean queue size, mean waiting time, and throughput. It is based on Little's formula, $L = \lambda W$ (i.e., the average number of customers in the system L , is equal to the average arrival rate λ , times the average amount of time spent by the customers in the system, W) and the following results:

In a closed queueing network with product-form solution, the probability that the system is in state N upon arrival of a part in the system with N parts is the same as the long term equilibrium probabilities of N in a system with $N - 1$ parts.

Mean value analysis (MVA) results in a simple recursive algorithm to determine measures of performance and can be used to analyze closed networks by applying an algorithm which has been found to be fairly accurate.

Now, if we observe that $p_i(0;0) = 1$, $i = 0,1,\dots,m$ and $Th(1) = \frac{1}{\sum_{i=0}^m \frac{v_i}{\mu_i r_i(1)}}$, then

Buzacott and Shanthikumar (1993) suggest that the following mean value analysis algorithm provides an efficient way to compute the system performance measures if each service center has only a single server (i.e., $c_i = 1$, $i = 0,1,\dots,m$).

Algorithm 4.1 Mean Value Analysis (MVA)

Step 1: Set $E[N_i(0)] = 0$, $i = 0,1,\dots,m$.

Step 2: For $\ell = 1,\dots,N$, compute

$$E[T_i(\ell)] = \{E[N_i(\ell-1)] + 1\} / \mu_i, \quad i = 0,1,\dots,m.$$

$$Th(\ell) = \ell / \left\{ \sum_{i=0}^m v_i E[T_i(\ell)] \right\}$$

$$E[N_i(\ell)] = v_i Th(\ell) E[T_i(\ell)] \quad , \quad i = 0, 1, \dots, m.$$

Step 3: Stop.

Note that the expected number of times a part visits service center i as a class ℓ part is v_i ,

the solution to the equation $v_i = \sum_{j=1}^m v_j p_{ji}, i = 0, 1, \dots, m$ (i.e. $\mathbf{v} = \mathbf{vP}$). Because each part

visits the load/unload station only once before it is replaced by a new (raw) part we also have $v_0 = 1$.

4.4 General Single-Stage Closed Queueing Network

It is important to account for deviations from the exponential service time assumptions. We will allow the service time distribution to be arbitrary and to be described by the mean and second moment of service time at each service center. We assume that there are c_i ($c_i \geq 1$) parallel servers at service center i , and the part service times at each service center form an iid sequence of random variables with distribution function Z_{s_i} with mean $E[S_i]$, second moment $E[S_i^2]$, $i = 0, 1, \dots, m$. Thus each service center acts as a $\bullet/G_i/c_i$ queueing system with the input process “ \bullet ” determined by the rest of the service centers.

We suppose that service center i can be modeled by a $M_i(N)/G_i/c_i$ queueing system with state-dependent arrival rates $\lambda(n_i)$, $n_i = 0, 1, \dots, N$ and the mean and squared coefficient of variation of the service times are $E[S_i] = 1/\mu_i$ and $C_{s_i}^2$, respectively. Expressions for the

$M(N)/G/c$ queue usually rely on C_s^2 ; which is assumed known for each service center (it depends on the service time distribution).

According to Buzacott and Shanthikumar (1993) when we have only a single server (i.e., $c_i = 1$) at each of the service centers, each arrival to service center i on its arrival will see the part in service, if any, requiring an average of $E[S_i^2]/2E[S_i]$ additional processing time to complete its service. This exhibits the property of an $M/G/1$ queueing system. They also said that, with this and assumption that an arrival with N parts in the system sees the time average behavior of a system with $N-1$ parts, we have

$$E[T_i(N)] = \{E[N_i(N-1)] - v_i Th(N-1)E[S_i] + 1\}E[S_i] + v_i Th(N-1)E[S_i^2]/2, \quad i = 0, 1, \dots, m. \quad (4.17)$$

So, approximate performance measures of the general closed queueing network are as follows;

Algorithm 4.2 Extended Mean Value Analysis (EMVA)

Step 1: $E[N_i(0)] = 0, i = 0, 1, \dots, m; Th(0) = 0$

Step 2: For $\ell = 1, \dots, N$, compute

$$E[T_i(\ell)] = \{E[N_i(\ell-1)] - v_i Th(\ell-1)E[S_i] + 1\}E[S_i] + v_i Th(\ell-1)E[S_i^2]/2, \quad i = 0, 1, \dots, m$$

$$Th(\ell) = \ell / \left\{ \sum_{i=0}^m v_i E[T_i(\ell)] \right\}$$

$$E[N_i(\ell)] = v_i Th(\ell) E[T_i(\ell)], \quad i = 0, 1, \dots, m.$$

Step 3: Stop.

Chapter 5

Illustrative Numerical Examples

5.1 Introduction

In this section we shall explore, by means of computational examples, performance measures of the analytical models developed in Chapter Four using Algorithms 4.1 and 4.2. All the results are from C++ programs written for Algorithms 4.1 and 4.2 for each example. The C++ programs for each example are shown in the appendices A-F.

In all cases considered we have only a single server (i.e., $c_i = 1, i = 1, \dots, m$) at each service center, and service center 0 is considered to be the load/unload server. The expected number of times a part visits service center i as a class ℓ part is v_i , the solution to the

equation $v_i = \sum_{j=1}^m v_j p_{ji}, i = 0, 1, \dots, m$ (i.e. $\mathbf{v} = \mathbf{vP}$). Because each part visits the

load/unload station only once before it is replaced by a new (raw) part we also have $v_0 = 1$.

The parts are serviced according to the FCFS service protocol.

5.2 Statement of Problem 1

The management of a company considered owning a new designed FMS. It is estimated that this new system can handle a total of six jobs. Three machine centers (0, 1, 2) (with a load/unload station being machine center 0) are to be formed to cover all the processing requirements of the six jobs. Each machine center is to be equipped with a single server and

their processing times are expected to be $E[S_0] = 0.7$, $E[S_1] = 1$, $E[S_2] = 0.3$ and their squared coefficient of variation of the service times are assumed known to be $C_{S_0}^2 = 0.25$, $C_{S_1}^2 = 1$, $C_{S_2}^2 = 0.5$. The routing probability matrix

$$P = \begin{bmatrix} 0 & 0.45 & 0.55 \\ 0.3 & 0 & 0.7 \\ 0.4 & 0.6 & 0 \end{bmatrix}$$

We want to calculate the performance measures.

5.3 Results of Problem 1

Using Algorithm 4.1 (MVA); we know that $\mu_i = 1/E[S_i]$, so we get $\mu_0 = 1.42857$, $\mu_1 = 1$,

and $\mu_2 = 3.33333$. Also, with $v_i = \sum_{j=1}^m v_j p_{ji}$, $i = 0, 1, \dots, m$ (i.e. $v = vP$), we get $v_0 = 1$, $v_1 =$

1.3448, and $v_2 = 1.4914$. We put these values in the C++ program found in appendix A and the following results were obtained.

MVA approximate throughput = 0.7308

Exact throughput = 0.7306

Table 1 gives the comparison of exact and MVA approximate average queue length and average waiting time for problem 1.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	MVA Approx.	Exact	MVA Approx.	Exact
0	0.9821	0.9818	1.3438	1.3435
1	4.5404	4.5416	4.6198	4.6202
2	0.4776	0.4772	0.4382	0.4376

Table 1: A comparison of MVA approximation and exact performance – Problem 1

Using Algorithm 4.2 (EMVA); with $E[S_i^2] = (E[S_i])^2 \times (1 + C_{S_i}^2)$ we get, $E[S_0^2] = 0.6125$,

$E[S_1^2] = 2$, and $E[S_2^2] = 0.135$. We put these values and the values of v_i and $E[S_i]$ in the

C++ program found in appendix B and the following results were obtained.

EMVA approximate throughput = 0.7359

Exact throughput = 0.7358

Table 2 gives the comparison of exact and EMVA approximate average queue length and average waiting time for problem 1.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	EMVA Approx.	Exact	EMVA Approx.	Exact
0	0.8192	0.8190	1.1132	1.1128
1	4.7356	4.7359	4.7851	4.7857
2	0.4453	0.4459	0.4057	0.4046

Table 2: A comparison of EMVA approximation and exact performance – Problem 1

5.4 Statement of Problem 2

An FMS consists of three machine centers, each with a single machine. A single type of part is loaded onto $N = 10$ pallets and processed by this FMS. Assume each part visits machine center $i = 0, 1, 2$ on the average $v_0 = 1$, $v_1 = 0.4$, $v_2 = 0.8$ times and the average processing requirements are $E[S_0] = 1.35$, $E[S_1] = 2$, $E[S_2] = 1.65$ respectively per part per visit to machine center i . If the squared coefficient of variation of the service times of each machine are known to be 1.5, 1.25, and 1 respectively; we want to calculate the performance measures.

5.5 Results of Problem 2

Using Algorithm 4.1 (MVA); from the problem we have

m	0	1	2
μ_i	0.7407	0.5	0.6061
v_i	1	0.4	0.8

Table 3: Values we put in the C++ program found in appendix C

We put these values (Table 3) in the C++ program found in appendix C and the following results were obtained.

MVA approximate throughput = 0.6704

Exact throughput = 0.6711

Table 4 gives the comparison of exact and MVA approximate average queue length and average waiting time for problem 2.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	MVA Approx.	Exact	MVA Approx.	Exact
0	4.1170	4.2314	6.9083	6.9267
1	1.0696	1.0256	4.1393	4.1105
2	3.8133	3.6840	7.9415	7.9113

Table 4: A comparison of MVA approximation and exact performance – Problem 2

Using Algorithm 4.2 (EMVA); from the problem we get

m	0	1	2
$E[S_i]$	1.35	2	1.65
v_i	1	0.4	0.8
$C_{S_i}^2$	1.5	1.25	1
$E[S_i^2]$	4.55625	9	5.445

Table 5: Values we put in the C++ program found in appendix D

We put these values (Table 5) in the C++ program found in appendix D and the following results were obtained.

EMVA approximation throughput = 0.6549

Exact throughput = 0.6558

Table 6 gives the comparison of exact and EMVA approximate average queue length and average waiting time for problem 2.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	EMVA Approx.	Exact	EMVA Approx.	Exact
0	5.0121	5.1243	7.6527	7.6942
1	1.1225	1.0441	4.2847	4.2645
2	3.8655	3.7451	7.3775	7.3567

Table 6: A comparison of EMVA approximation and exact performance – Problem 2

5.6 Statement of Problem 3

Consider a closed queueing network model of an FMS with a load/unload station 0 and three service centers (1, 2, 3). Each center is known to have a single machine and their processing rates are $\mu_0 = 2$, $\mu_1 = 0.6897$, $\mu_2 = 1$, $\mu_3 = 0.5$. Parts are transported between service centers by a closed-loop conveyor that connects all machines. The conveyor has ample capacity and small delay time relative to machine processing time and will be excluded from our analysis. The system is shown in Figure 2; with inter-machine transfer probabilities: $P_{00} = P_{11} = P_{22} = P_{33} = P_{30} = P_{12} = P_{13} = 0$, $P_{01} = 0.2$, $P_{02} = P_{20} = P_{23} = 0.3$, $P_{03} = P_{31} = P_{32} = 0.5$, $P_{10} = 1$, $P_{21} = 0.4$. The squared coefficient of variation of the service

times of each machine are fixed to be 0.5, 2, 1, and 0.5 respectively. $N = 15$ parts are kept in process. We want to examine the performance measures of the system.

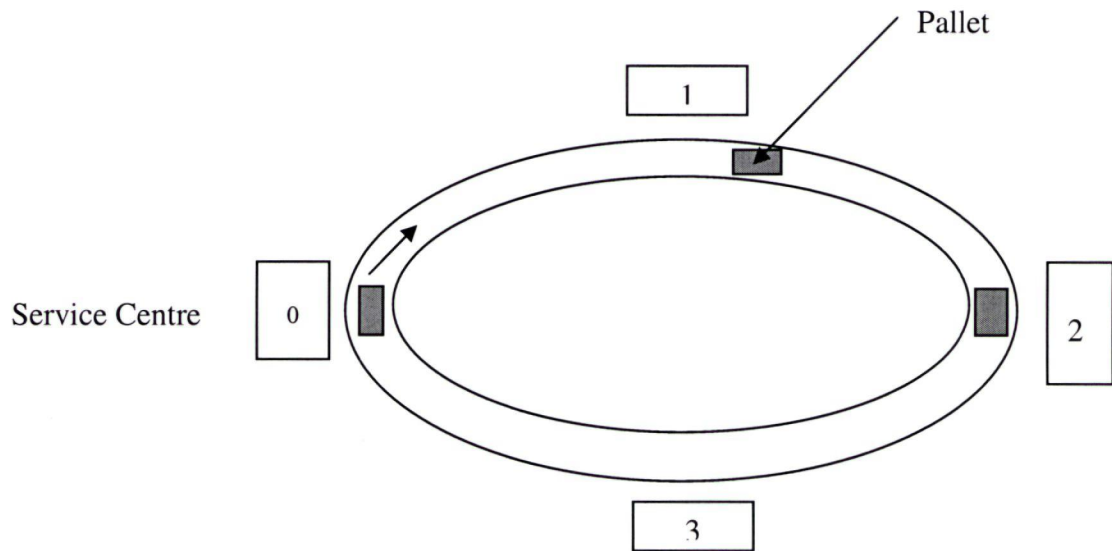


Figure 2: A Closed Queueing Network

5.7 Results of Problem 3

Using Algorithm 4.1 (MVA); from the problem we have

m	0	1	2	3
μ_i	2	0.6897	1	0.5
v_i	1	0.1059	0.6471	0.6941

Table 7: Values we put in the C++ program found in appendix E

We put these values (Table7) in the C++ program found in appendix E and the following results were obtained.

MVA approximation throughput = 0.7203

Exact throughput = 0.7214

Table 8 gives the comparison of exact and MVA approximate average queue length and average waiting time for problem 3.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	MVA Approx.	Exact	MVA Approx.	Exact
0	0.5629	0.5604	0.7814	0.7762
1	0.1244	0.1097	1.6302	1.6274
2	0.8730	0.8776	1.8728	1.8842
3	13.4398	13.4122	26.8801	26.8432

Table 8: A comparison of MVA approximation and exact performance – Problem 3

Using Algorithm 4.2 (EMVA); from the problem we get

m	0	1	2	3
$E[S_i]$	0.5	1.45	1	2
v_i	1	0.1059	0.6471	0.6941
$C_{S_i}^2$	0.5	2	1	0.5
$E[S_i^2]$	0.375	6.3075	2	6

Table 9: Values we put in the C++ program found in appendix F

We put these values (Table 9) in the C++ program found in appendix F and the following results were obtained.

EMVA approximation throughput = 0.7346

Exact throughput = 0.7362

Table 10 gives the comparison of exact and EMVA approximate average queue length and average waiting time for problem 3.

Machine center i	Average number of parts $E[N_i(\ell)]$		Average waiting time $E[T_i(\ell)]$	
	EMVA Approx.	Exact	EMVA Approx.	Exact
0	0.5380	0.5298	0.7188	0.7121
1	0.1344	0.1307	1.7272	1.6952
2	0.9093	0.9244	1.9129	1.9223
3	13.4283	13.4013	26.3351	26.3321

Table 10: A comparison of EMVA approximation and exact performance – Problem 3

The results from all the three illustrative examples indicate that both Algorithm 4.1 (MVA) and Algorithm 4.2 (EMVA) accurately predict the exact performance measures. In practical application, the designer should make sure that any changes in the assumptions go with changes in the model. The exact performance measure results were obtained from queueing network analysis package software.

Chapter 6

Measures of Flexibility in FMSs

6.1 Introduction

A company's ultimate success depends on its ability to utilize resources and meet the needs of the market. These internal factors steer demand and in turn the volume of business and the price of the commodity. Drastic changes in market demands and rapid technological development have created a need for more flexible production systems and more complex products with a larger degree of variation.

There is strong pressure towards the use of more and more mechanized and automated equipment from single numerical computerized-machines to complete manufacturing systems. At the same time, there is a need for flexibility towards changes in the products. These changes have to be made in a limited time and without the need for large reinvestments in the production system. Thus, production analysts are concerned with how well systems perform under a variety of conditions, and the ability of machines or groups of machines to adapt to change. FMSs have generated interest in flexibility, how to achieve it and how to measure it. In this section we shall explore measures of flexibility of machines and groups of machines relative to sets of required tasks, as defined by Brill and Mandelbaum (1987)(1989) by giving examples of their computation.

6.2 Measures of Single-Machine Flexibility Relative to Finite Task Set

The flexibility measures depend on the set of tasks to be done relative to a background task set, the relative importance of the tasks and the efficiency of machines in doing them.

Let a_{ij} denote the effectiveness of machine i for doing task j , where $0 \leq a_{ij} \leq 1$. The effectiveness measure will reflect the machine characteristics such as set up time, speed, quality, cost of doing the task, etc. For instance, the effectiveness measure might depend inversely on cost but directly on speed. That is, if the cost of a machine in doing a task is denoted by C and the speed of the machine in doing the task is denoted by S , then the effectiveness of the machine, $a = k / C$ and $a = kS$ respectively, where k is a constant, which is chosen such that $0 \leq a \leq 1$. It follows that $S = 1 / C$, thus as the cost of doing a task increases, the speed of doing the task decreases. Also, let $\omega_j, 0 \leq \omega_j \leq 1$, denote the weight of importance of task j . If the task set consists of all tasks under consideration, denoted by task set S_T , then

$$\sum_{j \in S_T} \omega_j = 1.$$

If the task set S_T is a subset of all tasks, i.e. $S_T \subseteq S_T$, then

$$\sum_{j \in S_T} \omega_j \leq 1.$$

Thus the measure of flexibility for machine i relative to a task set S_T is defined as

$$F_{i,S_T} = \frac{\sum_{j \in S_T} a_{ij} \omega_j}{\sum_{j \in S_T} \omega_j} \dots\dots\dots (6.1)$$

provided $\sum_{j \in S_T} \omega_j > 0$.

Example 6.1

A task set consisting of five tasks is to be performed, and six different machines are being considered to do them (Table 11). To each machine-task pair corresponds a number between 0 and 1 which indicates how effective the machine is in doing the task. An effectiveness rating of 1 indicates that the machine can perform the task most effectively, while a rating of 0 indicates that the machine cannot do that task at all. Also, to each task corresponds a non-negative weight of importance such that the overall weight for the whole task set is 1. Therefore, a machine is more flexible than another if its weighted effectiveness in performing tasks in the set is greater. Thus, a measure of flexibility for a machine is its weighted effectiveness over all the tasks in the set.

For instance, the measure of flexibility of machine 3 is given by

$$F_{m_3} = \frac{0.0 \times 0 + 0.1 \times 0.7 + 0.2 \times 0.3 + 0.3 \times 0.6 + 0.4 \times 0.5}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.51$$

where the denominator is the sum of all the importance weights.

The flexibility measure for each machine are computed in the same way and shown in Table 11. Machine 5, m_5 , has a greater flexibility measure than machine 2, m_2 , although it can do fewer tasks to some degree, and machine 1, m_1 , has a greater flexibility measure than machine 4, m_4 , although both can do all five tasks to some degree.

Machine-task efficiencies						
Task	1	2	3	4	5	
Weight of importance	0.0	0.1	0.2	0.3	0.4	Machine flexibility F_{m_i}
Group 1 machines m_1	1	1	1	1	1	1.00
Group 2 machines m_2	0	0.8	0.4	0.9	0	0.43
m_3	0	0.7	0.3	0.6	0.5	0.51
Group 3 machines m_4	0.3	0.8	0.7	0.4	0.9	0.70
m_5	0	0	0	0.7	0.6	0.45
m_6	1	0	0	0	0	0.00

Table 11: Machine flexibility

Example 6.2

In most practical applications only a subset of the tasks may be relevant. Referring to Table 11, assume only tasks 3, 4, and 5 are applicable. Using the same importance weights (0.2, 0.3, 0.4) for tasks 3, 4, and 5, respectively, we can compute flexibility measures between 0 and 1 relative to this new task subset. For instance, the flexibility of machine 3, m_3 , relative to the task subset $S_T = \{3, 4, 5\}$ is given by

$$F_{m_3, S_T} = \frac{(0.2 \times 0.3 + 0.3 \times 0.6 + 0.4 \times 0.5)}{(0.2 + 0.3 + 0.4)} = 0.49$$

where the denominator is the sum of the importance weights for the subset.

The flexibility measures of machines 1 – 6 relative to the task set $S_T = \{3, 4, 5\}$ are respectively 1.00, 0.39, 0.49, 0.69, 0.50, 0.00.

The ordering of the flexibility measures of the machines depends on the background reference subset of tasks. If the task set of five tasks is used, then an ordering of the machines in ascending order by their flexibility measures is $(m_6, m_2, m_5, m_3, m_4, m_1)$.

On the other hand, if the machine flexibilities are measured relative to the task subset

$S_T = \{3, 4, 5\}$, then the ordering is $(m_6, m_2, m_3, m_5, m_4, m_1)$.

6.3 Measures of Machine-Group Flexibility Relative to a Finite Task Set

Production analysts or decision-makers may be interested in the flexibility of a group of machines rather than that of an individual machine.

We consider three groups of machines from Table 11:

Group 1 = $\{m_1\}$, Group 2 = $\{m_2, m_3\}$, Group 3 = $\{m_4, m_5, m_6\}$. There are many ways flexibility of a group of machines relative to a task set S_T can be defined.

1. An optimistic measure of machine-group flexibility

An optimistic measure of group flexibility is given by

$$F_{G, S_T}^{(1)} = \frac{\sum_{j \in S_T, i \in G} \omega_j \max\{a_{ij}\}}{\sum_{j \in S_T} \omega_j} \dots\dots\dots(6.2)$$

where G denotes the group of machines. The machine-group effectiveness is computed by taking the maximum value of effectiveness (or best machine) for each task. For instance, the group 2 effectiveness measures are 0, 0.8, 0.4, 0.9, 0.5 for the five tasks respectively. Thus, the flexibility measure is calculated similarly to that of a single machine (that is, as the inner product of the task importance weights and the group effectiveness measures). The flexibility measures with respect to $S_T = S_T$ for group 1, group 2, and group 3 are as follows:

$$F_{G_1, S_T}^{(1)} = \frac{0.0 \times 1 + 0.1 \times 1 + 0.2 \times 1 + 0.3 \times 1 + 0.4 \times 1}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 1$$

$$F_{G_2, S_T}^{(1)} = \frac{0.0 \times 0 + 0.1 \times 0.8 + 0.2 \times 0.4 + 0.3 \times 0.9 + 0.4 \times 0.5}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.63$$

$$F_{G_3, S_T}^{(1)} = \frac{0.0 \times 1 + 0.1 \times 0.8 + 0.2 \times 0.7 + 0.3 \times 0.7 + 0.4 \times 0.9}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.79$$

respectively. Note that the group flexibility measure obtained by using the maximum effectiveness will always be greater than or equal to the measure of flexibility of a single machine in the group. This optimistic measure supposes the best allocation of machines to jobs.

2. A pessimistic measure of machine-group flexibility

A pessimistic measure of flexibility of the machine group with respect to a subset S_T would be obtained by assuming the least effective assignable machine to do the task. In this case the measure would be

$$F_{G, S_T}^{(2)} = \frac{\sum_{j \in S_T, i \in G} \omega_j \min^+ \{a_{ij}\}}{\sum_{j \in S_T} \omega_j} \dots\dots\dots(6.3)$$

where, for each $j \in S_T$

$$\min^+ (a_{ij}) = \begin{cases} \min_i \{a_{ij} | a_{ij} > 0\}, & \text{if there exists } a_{ij} > 0 \\ 0, & \text{otherwise.} \end{cases}$$

$i \in G$

Under this criterion, the lowest assignable value of effectiveness for each task is used. Effectiveness measures of zero are excluded since corresponding machines cannot be assigned to that task. The pessimistic flexibility for group 1, group 2, and group 3 with respect to $S_T = S_T$ are as follows:

$$F_{G_1, S_T}^{(2)} = \frac{0.0 \times 1 + 0.1 \times 1 + 0.2 \times 1 + 0.3 \times 1 + 0.4 \times 1}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 1$$

$$F_{G_2, S_T}^{(2)} = \frac{0.0 \times 0 + 0.1 \times 0.7 + 0.2 \times 0.3 + 0.3 \times 0.6 + 0.4 \times 0.5}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.51$$

$$F_{G_3, S_T}^{(2)} = \frac{0.0 \times 0.3 + 0.1 \times 0.8 + 0.2 \times 0.7 + 0.3 \times 0.4 + 0.4 \times 0.6}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.58$$

respectively.

3. A probabilistic measure of machine-group flexibility

A probabilistic measure of machine-group flexibility could be obtained by assuming that

machine i has a likelihood of being used for task j with probability p_{ij} , where $\sum_{i=1}^m p_{ij} = 1$,

and $p_{ij} = 0$ if $a_{ij} = 0$. The group flexibility measure with respect to subset S_T would be

$$F_{G, S_T}^{(3)} = \frac{\sum_{j \in S_T} \omega_j \sum_{i \in G} q_{ij} a_{ij}}{\sum_{j \in S_T} \omega_j} \dots\dots\dots(6.4)$$

$$= \frac{\sum_{j \in S_T} \omega_j E[\text{efficiency, conditional on doing task } j]}{\sum_{j \in S_T} \omega_j},$$

where $q_{ij} = \frac{p_{ij}}{\left(\sum_{i \in G} p_{ij} \right)}$ and $E[\cdot]$ stand for ‘expected value’ and $0 \leq E[\cdot] \leq 1$.

If we consider the weight of importance of task j , ω_j , as the probability of doing task j , then this group flexibility measure with respect to subset S_T can be interpreted as the expected value of the machine-task efficiency ratings for the machines in the group. For an individual machine, the probabilistic flexibility measure is the same as the machine’s usual flexibility measure.

Example 6.3

Table 12 below shows probabilities assigned to each of the six machines and tasks, the sum in each column being 1.

Task	1	2	3	4	5
m_1	–	0.3	0.5	0.2	0.1
m_2	–	0.2	0.3	0.4	–
m_3	0.4	–	0.1	0.2	0.4
m_4	–	0.5	0.1	0.1	0.3
m_5	–	–	–	0.1	0.2
m_6	0.6	–	–	–	–

Table 12: Probabilities of p_{ij} of using m_i for task j .

Consider the calculation of the flexibility with respect to $S_T = S_T$ for group 1, group 2,

and group 3 in the probabilities case . The conditional probabilities for group 1, group 2,

and group 3 are calculated using $q_{ij} = \frac{p_{ij}}{\sum_{i \in G} p_{ij}}$ and shown in Table 13.

Task	1	2	3	4	5
Group 1 machines m_1	-	0.3 / 0.3=1	0.5 / 0.5=1	0.2 / 0.2=1	0.1 / 0.1=1
Group 2 machines m_2	-	0.2 / 0.2=1	0.3 / 0.4=0.75	0.4 / 0.6=0.67	-
m_3	0.4 / 0.4=1	-	0.1 / 0.4=0.25	0.2 / 0.6=0.33	0.4 / 0.4=1
Group 3 machines m_4	-	0.5 / 0.5=1	0.1 / 0.1=1	0.1 / 0.2=0.5	0.3 / 0.5=0.6
m_5	-	-	-	0.1 / 0.2=0.5	0.2 / 0.5=0.4
m_6	0.6 / 0.6=1	-	-	-	-

Table 13: Conditional probabilities

The conditional expectation of machine-task efficiency is computed for each task using

$\sum_{i \in G} q_{ij} a_{ij}$ and shown in Table 14.

Task	1	2	3	4	5
Group 1 machines m_1	-	1(1)=1	1(1)=1	1(1)=1	1(1)=1
Group efficiencies	-	1	1	1	1
Group 2 machines m_2	-	1(0.8)=0.8	0.75(0.4)=0.3	0.67(0.9)=0.603	-
m_3	1(0)=0	-	0.25(0.3)=0.075	0.33(0.6)=0.198	1(0.5)=0.5
Group efficiencies	0	0.8	0.375	0.801	0.5
Group 3 machines m_4	-	1(0.8)=0.8	1(0.7)=0.7	0.5(0.4)=0.20	0.6(0.9)=0.54
m_5	-	-	-	0.5(0.7)=0.35	0.4(0.6)=0.24
m_6	1(1)=1	-	-	-	-
Group efficiencies	1	0.8	0.7	0.55	0.78

Table 14: The conditional expectation of machine-task efficiency

The probabilistic flexibility for group 1, group 2, and group 3 are calculated using

equation 6.4 as follows:

$$F_{G_1, S_T}^{(3)} = \frac{0.1 \times 1 + 0.2 \times 1 + 0.3 \times 1 + 0.4 \times 1}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 1$$

$$F_{G_2, S_T}^{(3)} = \frac{0.0 \times 0 + 0.1 \times 0.8 + 0.2 \times 0.375 + 0.3 \times 0.801 + 0.4 \times 0.5}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.5953$$

$$F_{G_3, S_T}^{(3)} = \frac{0.0 \times 1 + 0.1 \times 0.8 + 0.2 \times 0.7 + 0.3 \times 0.55 + 0.4 \times 0.78}{0.0 + 0.1 + 0.2 + 0.3 + 0.4} = 0.697$$

Table 15 shows the summary of the calculation of group flexibility from Table 11 – Table 14. In Table 15, conditional probabilities for each group from Table 13 have been shown, and the machine task efficiencies from Table 11 are shown in parentheses. The conditional expectation of machine-task efficiency computed for each task from Table 14 has been shown in the last row of each group. Each group measure of flexibility is equal to the

weighted sum of the conditional expectations, utilizing the task weights of importance, thus we have 1 for group 1, 0.5953 for group 2, and 0.697 for group 3.

Task	1	2	3	4	5
Weight of importance	0.0	0.1	0.2	0.3	0.4
m_1 (Group 1) Group efficiencies Flexibility =	- - 1	1(1) 1	1(1) 1	1(1) 1	1(1) 1
m_2 (Group 2) m_3 Group efficiencies Flexibility =	- 1(0) 0 0.5953	1(0.8) - 0.8	0.75(0.4) 0.25(0.3) 0.375	0.67(0.9) 0.33(0.6) 0.801	- 1(0.5) 0.5
m_4 (Group 3) m_5 m_6 Group efficiencies Flexibility =	- - 1(1) 1 0.697	1(0.8) - - 0.8	1(0.7) - - 0.7	0.5(0.4) 0.5(0.7) - 0.55	0.6(0.9) 0.4(0.6) - 0.78

Table 15: Summary of the calculation of group flexibility

From the calculated values obtained for the flexibility measures for machine groups 1 – 3 in all the three cases i.e., optimistic measure, pessimistic measure, and probabilistic measure, indicate that group 1 machines are more flexible, followed by group 3 machines and group 2 machines in that order. These computations are unique for this particular problem. This is so because a change in the background task sets changes the values of the flexibility measures. In an application, the background task sets should be defined to suit the particular problem at hand since the flexibility measures depend on the background task sets, and their values in general are sensitive to changes in the task sets.

Chapter 7

Summary and Discussion

Queueing theory models have found widespread use in the analysis of service provider facilities such as manufacturing systems, telecommunication systems, and many other situations where congestion or competition for scarce resources may occur. This thesis is intended to apply single-stage queueing theory to flexible manufacturing systems. Before we can apply queueing theory to flexible manufacturing systems, one has to know and also understand what a flexible manufacturing system is, and how it works. In light of these, we have defined a flexible manufacturing system (which is viewed as a fully automated system of service-providing work stations where parts (customers) enter the system of servers, visit their required servers in turn, and then leave the system).

Any good and efficient flexible manufacturing system starts with strategic decisions in line with design, planning, and scheduling and control. We have outlined a detailed decision strategy in relation to these issues which are intended to be used by a management group (decision makers) in support of choosing a flexible manufacturing system, its set of number and type of machines, type of parts (products) to produce, designing a machine for particular assignment, specification of computers and control system components, etc.

The management ultimate goal is to make sure that the system's machines are able to utilize all resources to boost production, so for productivity improvements through

increasing machine utilization, the system machines should be provided with tool magazines (storages) and automatic tool changers to allow many operations to be performed on a part each time it is available to the machine. Also, the material handling system of an FMS should be carefully selected such that it permits parts to visit machines in any sequence and imposes no constraint on the number of visits of parts to machines. Some factors that influence the choice of material handling system are the size and weight of the parts, the volume of material handling, and the capital availability for its implementation. Furthermore, the system should be able to respond flexibly to unforeseen activities, such as machine breakdown and temporary overloads, by dynamically rerouting parts to the nearest available machine with the necessary tooling to alleviate potential bottlenecks.

To design an FMS there is the need for proposals and the designers must carefully ensure that their proposals meet certain criteria, such as their design is technically sound, offer economic benefits, and are financially justifiable before establishing more details and then implement. It is important for an FMS to have compatible information management system software so that future developments would not be higher capital investment or be impossible. Also, the designers and the planners should make it clear for the manufacturers to understand the technologies that will be required and that new technologies be introduced carefully, based on well thought out planning. There should be a measuring machine (quality control) to monitor the quality of the parts being processed as well as the processes themselves. The intelligent selection of operations to measure and the frequency with which to measure them is required in order to ensure that quality standards are met and that processing errors are quickly identified and corrected.

Decisions concerning what work piece is manufactured next on which work station, are made close to the operation currently being performed by the particular work station. That is, a complete FMS schedule is not made in advance because it must be capable of responding to real-time decisions – respond quickly to changes in customer demand.

Generally, increasing the number of pallets in the system increases the rate at which parts flow through the system and vice versa. This is due to the fact that having more pallets increases the probability that a part will always be ready for processing when a machine becomes idle. But increasing the number of pallets should go along with increasing the storage capacity to avoid congestion that affects the throughput.

We have developed analytical models to predict the performance of the flexible manufacturing system and also to give the designer insight into the technical issues related to the system. We considered only closed queueing networks that can model flexible manufacturing systems but we observed that there are two basic types of queueing networks distinguished by their nature of arrivals and departures. If a queueing network system has no potential for arrivals from the outside and at the same time does not allow parts (customers) to leave, we say that the queueing network system is closed. On the other hand, if arrivals and departures are allowed, the queueing network system is said to be open. Clearly, the number of parts in a closed queueing network system is fixed, while the number of parts in an open queueing network system can be any value from zero to infinity. So, closed queueing network system is said to be stochastically equivalent to open queueing network system if we observe the open queueing network system only when there is a fixed number of parts (a total of N parts) in it.

In order to solve the models they require simplifying assumptions. Although the assumptions of such models may not be met exactly in a practical application, these models can still be useful in providing a rough estimate of a performance measure. In designing an FMS, if the number of servers needed at each work station or service center is not known but the arrival rate λ_i and the service rate μ_i are known or can be estimated, then the inequality $\lambda_i > \mu_i c_i$ can be used to provide an initial estimate for the number of servers, c_i , at service center i .

Exact performance measure results were used to compare the predictive ability of the Algorithm 4.1 and Algorithm 4.2 and it was found that the algorithms provided performance measure estimates that were as accurate as the exact performance measures. The exact performance measure results were obtained from queueing network analysis package software. It should be noted that any changes in the assumptions should go with changes in the models. Thus it is important for the designer to have available resources (software) that implement the algorithms presented in this thesis so that he can try out a wide range of different system conditions to understand the influence of various design factors on system performance.

There is strong pressure towards the use of more and more automated machines due to overwhelming changes in market demands, ever shorter life cycle for products, and rapid technological advancement. As a result of this, it is now very important that flexible manufacturing systems become more flexible for complex production processes which are the real situation on the ground of the manufacturing environment. We have defined flexibility measures of single machine and group of machines relative to finite task set. The flexibility measures depend on the set of tasks to be done relative to a background task set,

the relative weights of importance of the tasks and the efficiencies of the machines in doing them.

In a practical application, decision makers must make sure that the background task sets are detailed defined to incorporate how important each task is to the particular problem in the production environment. This is so because of the sensitivity nature of the flexibility measures to the background task sets. In general, the values of the flexibility measures are sensitive to changes in the task sets because the flexibility measures depend on them. Static situations flexibility measures are considered and the simple numerical illustrative examples to explore the flexibility measures presented consider discrete task sets. However, the definitions of flexibility measures can be extended to cover continuous task sets.

Appendix A

The C++ program for Problem 1 using Algorithm 4.1 (MVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$\mu[i] = \mu_i$ = mean part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 2$), number of parts ($N = 6$), mean part service times, $\mu[3] = \{1.42857, 1, 3.33333\}$, and expected number of times a part visits service center i , $v[3] = \{1, 1.3448, 1.4914\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
// MVA Problem 1

#include <stdio.h>
#include <iostream.h>

#define N 6
#define m 2

//Assigns values for number of machine centers(m) and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ; //initialize sum

    double ET[m+1][N+1],Th[N+1],EN[m+1][N+1] , mu[3]={1.42857,1,3.33333},
    v[3]={1,1.3448,1.4914};

    //Assign known values to Mean service time and mean number of times a
    //part visits service centers for various machine centers

    for (i = 0;i<=m;i++){

        EN[i][0] = 0;

        //Assigns 0 to first column of matrix EN

    }

    j=1; Th[0]=0;

    //Starting a while loop to calculate elements of ET, EN and Th

    while (j<=N){//while loop begins...

        for( i = 0;i<=m; i++){ //Start 1 ...for loop

            //Finding column j elements for ET matrix (Mean waiting times)

            ET[i][j]= (EN[i][j-1] +1) /mu[i];
```

```

} //End 1 ...for loop

sum=0;

for (k = 0; k<=m; k++){ //Start 2 ...for loop
    sum = sum + v[k]*ET[k][j];
} //End 2 ...for loop

Th[j] = j/sum; //calculating throughput rate

for( i = 0; i<=m; i++){ //Start 3 ...for loop
    EN[i][j]= v[i] * Th[j] * ET[i][j];
} // End 3...forloop

j++; //increasing count for j

} //End of while loop for j

//Printing results...
//#####

cout<< "Here are the elements of the Th vector \n ";
cout<< "\n ";

for( j = 1; j<=N; j++){
    cout<< "Th["<<j<<"] = "<< Th[j]<< "\n ";
}

cout<< "\n ";

cout<< "Here are the elements of the EN matrix ... \n ";

cout<< "\n ";

for ( j=1; j<=N; j++){
    for( i = 0; i<=m; i++){
        cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<< " ";
    }
}

```

```
        cout<< "\n ";
    }

    cout<< "\n ";
    cout<< "Here are the contents of the ET matrix \n ";

    cout<< "\n ";
    for ( j=1; j<=N; j++){
        for( i = 0; i<=m; i++){
            cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";
        }
        cout<< "\n ";
    }

    return ;
} //End of main...
```

Appendix B

The C++ program for Problem 1 using Algorithm 4.2 (EMVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$ES[i] = E[S_i]$ = mean part service time at service center i

$ESsq[i] = E[S_i^2]$ = second moment part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 2$), number of parts ($N = 6$), mean part service times, $ES[3] = \{0.7, 1, 0.3\}$, second moment part service times,

$ESsq[3] = \{0.6125, 2, 0.135\}$, and expected number of times a part visits service center i , $v[3] = \{1, 1.3448, 1.4914\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
//EMVA Problem 1

#include <stdio.h>
#include <iostream.h>

#define N 6
#define m 2

//Assigns values for number of machine centers(m) and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ; //initialize sum

    double          ES[3]={0.7,1,0.3},          ET[m+1][N+1],Th[N+1],EN[m+1][N+1]
    ,ESsq[3]={0.6125,2,0.135}, v[3]={1,1.3448,1.4914};

    //Assign known values to Mean service time, second moment of service
    //time and mean number of times a part visits service centers for
    //various machine centers

    for (i = 0;i<=m;i++){

        EN[i][0] = 0;

//Assigns 0 to first column of matrix EN

    }

    j=1;  Th[0]=0;

// Starting a while loop to calculate elements of ET, EN and Th
```

```

while (j<=N){ //while loop begins...

    for( i = 0;i<=m; i++){ //Start 1 ...for loop

//Finding column j elements for ET matrix (Mean waiting times)

        ET[i][j]= (EN[i][j-1] -v[i]*Th[j-1] * ES[i] +1)*ES[i] +
v[i]*Th[j-1]*ESsq[i]/2;

    }//End 1 ...for loop

    sum=0;

    for (k = 0; k<=m; k++){ //Start 2 ...for loop

        sum = sum + v[k]*ET[k][j];

    }//End 2 ...for loop

    Th[j] = j/sum; //Calculating throughput rate

    for( i = 0;i<=m; i++){ //Start 3 ...for loop

        EN[i][j]= v[i] * Th[j] * ET[i][j];

    }// End 3...for loop

    j++; //increasing count for j

} //End of while loop for j

//Printing results...
//#####

cout<< "Here are the elements of the Th vector \n ";
cout<< "\n ";

for( j = 1;j<=N; j++){

    cout<< "Th["<<j<<"] = "<< Th[j]<<"\n ";

}

cout<< "\n ";

cout<< "Here are the elements of the EN matrix ... \n ";

```

```

cout<< "\n ";
for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){
    cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<<" ";
    }
    cout<< "\n ";
}

cout<< "\n ";
cout<< "Here are the contents of the ET matrix \n ";
cout<< "\n ";
for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){
    cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";
    }
    cout<< "\n ";
}

return ;
} //End of main...

```

Appendix C

The C++ program for Problem 2 using Algorithm 4.1 (MVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$\mu[i] = \mu_i$ = mean part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 2$), number of parts ($N = 10$), mean part service times, $\mu[3] = \{0.7407, 0.5, 0.6061\}$, and expected number of times a part visits service center i , $v[3] = \{1, 0.4, 0.8\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
//MVA Problem 2

#include <stdio.h>
#include <iostream.h>

#define N 10
#define m 2

//Assigns values for number of machine centers(m) and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ; //initialize sum

    double ET[m+1][N+1], Th[N+1], EN[m+1][N+1] , mu[3]={0.7407,0.5,0.6061},
    v[3]={1,0.4,0.8};

    //Assign known values to Mean service time and mean number of times a
    //part visits service centers for various machine centers

    for (i = 0; i<=m; i++){

        EN[i][0] = 0;

        //Assigns 0 to first column of matrix EN
    }

    j=1; Th[0]=0;

    //Starting a while loop to calculate elements of ET, EN and Th

    while (j<=N){ //while loop begins...

        for( i = 0; i<=m; i++){ //Start 1 ...for loop

            //Finding column j elements for ET matrix (Mean waiting times)

            ET[i][j]= (EN[i][j-1] +1) /mu[i];

        } //End 1 ...for loop
    }
}
```

```

sum=0;

for (k = 0; k<=m; k++){ //Start 2 ...for loop

    sum = sum + v[k]*ET[k][j];

} //End 2...for loop

Th[j] = j/sum; //calculating throughput rate


for( i = 0;i<=m; i++){ //Start 3 ...for loop

    EN[i][j]= v[i] * Th[j] * ET[i][j];

} // End 3 ...for loop

j++; //increasing count for j

} //End of while loop for j

    //Printing results...
    //#####

cout<< "Here are the elements of the Th vector \n ";
cout<< "\n ";

for( j = 1;j<=N; j++){

    cout<< "Th["<<j<<"] = "<< Th[j]<<"\n ";

}

cout<< "\n ";

cout<< "Here are the elements of the EN matrix ... \n ";

cout<< "\n ";

for ( j=1; j<=N; j++){

for( i = 0;i<=m; i++){

    cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<<" ";

    }

}

```

```

        cout<< "\n ";

    }

    cout<< "\n ";

    cout<< "Here are the contents of the ET matrix \n ";

    cout<< "\n ";

    for ( j=1; j<=N; j++){
    for( i = 0;i<=m; i++){

        cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";

        }

        cout<< "\n ";

    }

    return ;

} //End of main...

```

Appendix D

The C++ program for Problem 2 using Algorithm 4.2 (EMVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$ES[i] = E[S_i]$ = mean part service time at service center i

$ESsq[i] = E[S_i^2]$ = second moment part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 2$), number of parts ($N = 10$), mean part service times, $ES[3] = \{1.35, 2, 1.65\}$, second moment part service

times, $ESsq[3] = \{4.55625, 9, 5.445\}$, and expected number of times a part visits service center i , $v[3] = \{1, 0.4, 0.8\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
//EMVA Problem 2

#include <stdio.h>
#include <iostream.h>

#define N 10
#define m 2

//Assigns values for number of machine centers(m)and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ;//initialize sum

    double      ES[3]={1.35,2,1.65},      ET[m+1][N+1],Th[N+1],EN[m+1][N+1]
,ESsq[3]={4.55625,9,5.445}, v[3]={1,0.4,0.8};

    //Assign known values to Mean service time, second moment of service
    //time and mean number of times a part visits service centers for
    //various machine centers

    for (i = 0;i<=m;i++){

        EN[i][0] = 0;

        //Assigns 0 to first column of matrix EN

    }

    j=1; Th[0]=0;

    //Starting a while loop to calculate elements of ET, EN and Th

    while (j<=N){ //while loop begins...
```

```

        for( i = 0;i<=m; i++){ //Start 1...for loop

//Finding column j elements for ET matrix (Mean waiting times)

        ET[i][j]= (EN[i][j-1] -v[i]*Th[j-1] * ES[i] +1)*ES[i] +
v[i]*Th[j-1]*ESsq[i]/2;

        }//End 1...for loop

        sum=0;

        for (k = 0; k<=m; k++){ //Start 2...for loop

                sum = sum + v[k]*ET[k][j];

        }//End 2 ...for loop

        Th[j] = j/sum; //Calculating throughput rate


        for( i = 0;i<=m; i++){ //Start 3...for loop

                EN[i][j]= v[i] * Th[j] * ET[i][j];

        }// End 3...for loop

        j++; //increasing count for j

    } //End of while loop for j


//Printing results...
//#####

        cout<< "Here are the elements of the Th vector \n ";
        cout<< "\n ";

        for( j = 1;j<=N; j++){

                cout<< "Th["<<j<<"] = "<< Th[j]<<"\n ";

        }

        cout<< "\n ";

```

```

cout<< "Here are the elements of the EN matrix ...\n ";

cout<< "\n ";
for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){

    cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<<" ";

    }

    cout<< "\n ";

}

cout<< "\n ";

cout<< "Here are the contents of the ET matrix \n ";

cout<< "\n ";
for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){

    cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";

    }

    cout<< "\n ";

}

return ;

} //End of main...

```

Appendix E

The C++ program for Problem 3 using Algorithm 4.1 (MVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$\mu[i] = \mu_i$ = mean part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 3$), number of parts ($N = 15$), mean part service times, $\mu[4] = \{2, 0.6897, 1, 0.5\}$, and expected number of times a part visits service center i , $v[4] = \{1, 0.1059, 0.6471, 0.6941\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
//MVA Problem 3

#include <stdio.h>
#include <iostream.h>

#define N 15
#define m 3

//Assigns values for number of machine centers(m) and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ; //initialize sum

    double ET[m+1][N+1],Th[N+1],EN[m+1][N+1] , mu[4]={2,0.6897,1,0.5},
v[4]={1,0.1059,0.6471,0.6941};

    //Assign known values to Mean service time and mean number of times a
    //part visits service centers for various machine centers

    for (i = 0;i<=m;i++){

        EN[i][0] = 0;

        //Assigns 0 to first column of matrix EN
    }

    j=1; Th[0]=0;

    //Starting a while loop to calculate elements of ET, EN and Th

    while (j<=N){ //while loop begins...

        for( i = 0;i<=m; i++){ //Start 1 ...for loop

            //Finding column j elements for ET matrix (Mean waiting times)

            ET[i][j]= (EN[i][j-1] +1) /mu[i];

        } //End 1 ...for loop
    }
}
```

```

sum=0;

for (k = 0; k<=m; k++){ //Start 2 ...forloop
    sum = sum + v[k]*ET[k][j];
} //End 2 ...for loop

Th[j] = j/sum; //calculating throughput rate


for( i = 0;i<=m; i++){ //Start 3 ...for loop
    EN[i][j]= v[i] * Th[j] * ET[i][j];

} // End 3 ...for loop

j++; //increasing count for j

} //End of while loop for j

    //Printing results...
    //#####

cout<< "Here are the elements of the Th vector \n ";
cout<< "\n ";

for( j = 1;j<=N; j++){

    cout<< "Th["<<j<<"] = "<< Th[j]<<"\n ";

}

cout<< "\n ";

cout<< "Here are the elements of the EN matrix ... \n ";

cout<< "\n ";

for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){

    cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<<" ";

    }

    cout<< "\n ";

```

```

    }

    cout<< "\n ";

    cout<< "Here are the contents of the ET matrix \n ";

    cout<< "\n ";

    for ( j=1; j<=N; j++){
    for( i = 0;i<=m; i++){

        cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";

        }

        cout<< "\n ";

    }

    return ;

} //End of main...

```

Appendix F

The C++ program for Problem 3 using Algorithm 4.2 (EMVA).

Notation

$i = k = 0, \dots, m$ = machine centers

$j = \ell = 1, \dots, N$ = class ℓ parts

m = number of machine centers

N = number of parts

$ES[i] = E[S_i]$ = mean part service time at service center i

$ESsq[i] = E[S_i^2]$ = second moment part service time at service center i

$V[i] = v_i$ = expected number of times a part visits service center i

$EN[i][j] = E[N_i(\ell)]$ = average number of parts (ℓ) at service center i (Note: $\ell = j$)

$ET[i][j] = E[T_i(\ell)]$ = average waiting time of an arbitrary part (ℓ) in service center i

(Note: $\ell = j$)

$Th[j] = Th[\ell]$ = throughput rate (Note: $\ell = j$)

$ET[m+1][N+1]$ = ET matrix of dimension $m+1 \times N+1$

$EN[m+1][N+1]$ = EN matrix of dimension $m+1 \times N+1$

$Th[N+1]$ = Th vector of $N+1$ elements

Input

The main function “void main()” takes number of machine centers ($m = 3$), number of parts ($N = 15$), mean part service times, $ES[4] = \{0.5, 1.45, 1, 2\}$, second moment part service

times, $ESsq[4] = \{0.375, 6.3075, 2, 6\}$, and expected number of times a part visits service center i , $v[4] = \{1, 0.1059, 0.6471, 0.6941\}$ as an input.

Output

It outputs the throughput rates in a vector form, followed by the average number of parts in a matrix form, and followed by the average waiting time of a part in a matrix form.

Program

```
//EMVA Problem 3

#include <stdio.h>
#include <iostream.h>

#define N 15
#define m 3

//Assigns values for number of machine centers(m) and number of
//parts(l=j)

void main()
{
    int i, j, k;

    double sum=0.0 ; //initialize sum

    double      ES[4]={0.5,1.45,1,2},      ET[m+1][N+1],Th[N+1],EN[m+1][N+1]
    ,ESsq[4]={0.375,6.3075,2,6}, v[4]={1,0.1059,0.6471,0.6941};

    //Assign known values to Mean service time, second moment of service
    //time and mean number of times a part visits service centers for
    //various machine centers

    for (i = 0;i<=m;i++){

        EN[i][0] = 0;

        //Assigns 0 to first column of matrix EN

    }

    j=1; Th[0]=0;

    //Starting a while loop to calculate elements of ET, EN and Th

        while (j<=N){ //while loop begins...

            for( i = 0;i<=m; i++){ //Start 1...forloop
```

```

//Finding column j elements for ET matrix (Mean waiting times)

        ET[i][j]= (EN[i][j-1] -v[i]*Th[j-1] * ES[i] +1)*ES[i] +
v[i]*Th[j-1]*ESsq[i]/2;

    }//End 1 ...for loop

    sum=0;

    for (k = 0; k<=m; k++){ //Start 2 ...for loop
        sum = sum + v[k]*ET[k][j];
    }//End 2 ...for loop

    Th[j] = j/sum; //Calculating throughput rate

    for( i = 0;i<=m; i++){ //Start 3 ...for loop
        EN[i][j]= v[i] * Th[j] * ET[i][j];
    }// End 3 ...for loop

    j++; //increasing count for j

} //End of while loop for j

        //Printing results...
    //#####

    cout<< "Here are the elements of the Th vector \n ";
    cout<< "\n ";

    for( j = 1;j<=N; j++){

        cout<< "Th["<<j<<"] = "<< Th[j]<<"\n ";

    }

    cout<< "\n ";

    cout<< "Here are the elements of the EN matrix ... \n ";

    cout<< "\n ";

```

```

for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){
    cout<< "EN["<<i<<","<<j<<"] = "<< EN[i][j]<<" ";

    }
    cout<< "\n ";
}
cout<< "\n ";
cout<< "Here are the contents of the ET matrix \n ";

cout<< "\n ";
for ( j=1; j<=N; j++){
for( i = 0;i<=m; i++){
    cout<< "ET["<<i<<","<<j<<"] = "<< ET[i][j]<<" ";

    }
    cout<< "\n ";
}

return ;
} //End of main...

```

Bibliography

- [1] Altiok, T., "Performance Analysis of Manufacturing Systems".(Springer Series in Operations Research),Springer-Verlag, New York, (1997).
- [2] Arbel, A. and Seidmann, A., "Performance Evaluation of Flexible Manufacturing Systems". IEEE Transactions on Systems, Man, and Cyber netics, Vol. SMC-14 No. 4, pp. 606-617, July-August (1984).
- [3] Askin, R. G. and Standridge, C. R., "Modeling and Analysis of Manufacturing Systems". John Wiley & Sons, Inc., New York, (1993).
- [4] Bonetto, R., "Flexible Manufacturing Systems in Practice". Hemisphere Publishing Corporation, U.S.A., (1988).
- [5] Brill, P. H. and Mandelbaum, M., "Measures of Flexibility for Production Systems". In Proceedings of the Ninth International Production Research Conference, Cincinnati, Ohio, USA, pp2474-2481, (1987).
- [6] Brill, P. H. and Mandelbaum, M., "On Measures of Flexibility in Manufacturing Systems". International Journal of Production Research, Vol. 27, No. 5, pp747-756, (1989).
- [7] Buzacott, J. A., "Modeling Automated Manufacturing Systems". Flexible Manufacturing Systems: Current Issues and Models. Edited by Choobineh, F. and Suri, R., pp. 130-137, (1986).
- [8] Buzacott, J. A. and Shanthikumar, G. J., "Stochastic Models of Manufacturing Systems". Prentice-Hall, Inc., New Jersey, (1993).

- [9] Buzacott, J. A. and Yao, D. D., "Flexible Manufacturing Systems: A Review of Analytical Models". *Management Science* Vol. 32, No. 7, pp890-905, July (1986).
- [10] Gershwin, S. B., "Manufacturing Systems Engineering". PTR Prentice Hall, Inc., New Jersey, (1994).
- [11] Gershwin, S. B., Hildebrant, R. R., Suri, R., and Mitter, S. K., "A Control Perspective on Recent Trends in Manufacturing Systems". *IEEE Control Systems Magazine*, Vol. 6, No. 2, pp. 3-15, April (1986).
- [12] Gordon, W. J. and Newell, G. E. , "Closed Queueing Systems with Exponential Servers". *Operations Research*, Vol. 15, No. 2, pp254-265, (1967).
- [13] Gordon, W. J. and Newell, G. E. , "Cyclic Queueing Systems with Restricted Length Queues". *Operations Research*, Vol. 15, No. 2, pp266-277, (1967).
- [14] Govil, M. K. and Fu, M. C., "Queueing Theory in Manufacturing: A Survey". *Journal of Manufacturing Systems; Dearborn*; Vol. 18, pp. 214-240, (1999).
- [15] Grimmett, G. R. and Stirzaker, D. R., "Probability and Random Processes". Third Edition, Oxford University Press Inc., New York, (2001).
- [16] Jackson, R. R. P., "Queueing Systems with Phase-Type Service". *Operations Research Quarterly*, Vol. 5, pp109-120, (1954).
- [17] Jackson, R. R. P., "Random Queueing Processes with Phase-Type Service". *Journal of the Royal Statistical Society, B* Vol. 18, No. 1 pp129-132, (1956).
- [18] Kusiak, A., editor, "Flexible Manufacturing Systems: Methods and Studies". Vol. 12, North Holland, (1986).
- [19] Randy, P., "The Design and Operation of FMS". IFS (Publications) Ltd., UK, (1983).

- [20] Stecké, K. E., "Design, Planning, Scheduling, and Control Problems of Flexible Manufacturing Systems". Annals of Operations Research, Vol. 3, pp. 3-12, (1985).