A Simulation Study of Tooling Effects in Automated Manufacturing

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements for the
degree of Master of Applied Science in
Mechanical Engineering

By
Siros Amini

Ottawa-Carlton Institute for
Mechanical and Aeronautical Engineering
University of Ottawa
Ottawa, Ontario
Canada

ISBN 0-612-15585-4

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Abstract

With advent of complex and expensive automated manufacturing systems tool management is becoming an important issue both in design and implementation phases. The use of highly automated technology has resulted in an increasing need to integrate tool management more thoroughly into other planning and operational issues of these systems.

In this research effort a discrete events simulation model has been developed to analyze the system performance under different operational strategies. These strategies include three tool configurations; identical, nonidentical and hybrid, and two part loading patterns; batch and random loading. In addition, a new tool refilling concept, tool replenishing, is also examined. The replenishing concept consists of two policies; preventive and failure replenishing. The effects of shared and dedicated tooling policies have also been investigated. With the comparison of performance measures for each set of operational strategies, the mutual advantages and concerns of each of them are highlighted.

Within the simulation scope, the shared tooling policy is significantly superior to the dedicated tooling policy. It is also shown that identical tool configuration outperforms the other tool configurations. These have been confirmed by statistical tests performed on the simulation results.

Acknowledgement

I wish to express my appreciation and gratitude toward my supervisor, Professor Dr. Ming Liang. His valuable insight, advice and guidance greatly assisted me throughout my graduate studies. Also, I would like to thank the members of my defence committee, Dr. Dan Necsulescu and Dr. Geza Kardos, for their time and effort in reviewing and examining my thesis.

As well, I wish to thank all the members of the Faculty of Mechanical Engineering at the University of Ottawa and my graduate colleagues, especially Mr Saeed Zolfaghari, Farhad Kolahan and Han Zhixin for their useful suggestions and informative discussions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Dedication

to

My dear parents Manouchehr and Iran Amini

Chapter 1

INTRODUCTION

1.1 Background

The development in automated manufacturing has made it possible to satisfy the need for shorter production lead time, quicker response demand changes, and better control over the production processes. However, since automated manufacturing systems represent heavy capital investment and are much more complex than the conventional manufacturing systems, any misconceptions in design or mistakes in implementation may lead to a unreliable system, inadequate production efficiency and high operational cost. Due to the large amount of information involved and the dynamic nature, tooling problem has been recognized as one of the most critical issues among various aspects of design and operation in automated manufacturing (Gruver and Senninger 1990).

Most automated manufacturing systems consist of a number of general purpose machining centers. The number of operations that can be processed by a machining center depends on whether the required tools are allocated to the machining center. The productivity and the flexibility in handling product variety and volume changes of an automated manufacturing system therefore greatly depend on how the tools are distributed among the machining centers in the system. This issue is defined as *tool configuration problem.* For the same number of machining centers and the same number of tools, a system can be configured in many different patterns and behaves quite differently. It is therefore very important to analyze the effects of tool configuration on the system performance.

Once a system has been properly configured, the system can start the scheduled production. However, the tools allocated to the tool magazine of each machine may or may not be sufficient to complete the production requirement. If the tool magazine are not refilled, the production batch size will be restricted by the tool magazine capacity. Smaller batch size means more production setups which are not desirable. To reduce the number of setups, it may be beneficial to periodically refill the tool magazines with the new tools delivered from tool room. This is referred to as *tool replenishing problem.* Unlike the traditional tool replacement problem that focuses on a single tool-machine problem, tool replenishing deals with the tooling problem at the system level. The questions to answered include: how often should the replenishing be made? What tools should be replaced?

If all spares of one tool type on the tool magazine are exhausted before the scheduled review or replenishing, the required tools may be borrowed from a neighbour machine to prevent the unplanned machine shutdown. This is called *tool sharing.* Another advantage of tool sharing is that it can help reduce the number of tool spares in the system. Further the saved slots on a tool magazine can be used for other more critical tools and thus help improve the system flexibility. However, tool sharing does cause additional tool flow and human intervention, and may lead to the shutdown of the machine from which the tools are borrowed as it sometimes has to wait for the tools to be returned. Therefore, it is worthwhile analyzing both positive and negative effects of tool sharing before making any decision. These points are elaborated in detail in chapter 3.

All the three types of tooling problems are unique in the automated manufacturing systems. They are closely related to each other. For example, if tool sharing is allowed, less tool replenishing may be scheduled. Otherwise more frequent tool replenishing is required. Further, the appropriateness of a tooling strategy is also affected by other operational problems such as part loading, sequencing, and machine layout. A proper decision can be made only when these problems are collectively considered. This is, however, extremely difficult due to the large number of system variables. This is further complicated by a) the large number of tools in the automated manufacturing environment which can easily reaches thousands, b) the dynamic nature of the tooling problems, c) the probabilistic nature of tool lives, part arrival times, and material handling and loading/unloading times, and d) the need to keep track of the cumulative tool usage for each tool copy. With the existing techniques, it is almost impossible to develop an analytical model to incorporate all the above issues and to provide quick solutions to shop floor problems. This gives rise to the application of simulation approach as simulation can model almost any detail of a real system and provide the user with a good solution, though not optimum, or a close estimate of the system performance within a reasonable amount of time.

1.2 Purpose of the Research

The main goal of this research is to develop a simulation model to collectively analyze the effects of the following tooling problems on system performance under different part loading policies:

. tool configuration

. tool replenishing

. tool sharing

To assess the system performance several measurement criteria, depending upon the loading patterns, will be considered in this study. These are as follows:

. system utilization (for random loading)

. system throughput (for random loading)

. makespan (for batch loading)

. mean flow time (for batch loading)

The definitions of these performance measures are given in chapter 3.

1.3 Organization of the Research

Following this chapter the organization of the thesis is as follows: a brief review of the related literature is given in chapter 2. Chapter 3 is dedicated to the development of simulation model. The concepts of tool configuration, tool replenishing, tool sharing, part loading and operational strategies are also elaborated in this chapter. Simulation results and the discussion of the system behaviour under different policies are presented in chapter 4. Finally, in chapter 5 conclusions and recommendation for further researches are presented.

Chapter 2

TOOLING PROBLEMS -- A REVIEW

Tooling has been recognized as a critical issue in automated manufacturing (Gray et al. 1992, ElMaraghy 1985, and Tomek 1986). Industrial data indicates that (1) tooling accounts for 25-30% of the fixed and variable cost of production in an automated machining environment (Ayres 1988, and Tomek 1986), and (2) 16% of scheduled production cannot be met due to tooling problem (Mason 1986). Therefore, the way that tools are planned and used has a direct impact on the performance of an automated manufacturing system. However, since the number of cutting tools can easily run into thousands even in a medium sized automated machining firm (ElMaraghy 1985) and further because tool status, e.g., wear, breakage, number of copies in each tool magazine dynamically change, the tooling problem is among the most complex decision issues.

Considerable efforts have been made to investigate various tooling problems by many researchers. The tooling problems addressed in the literature can be divided into four categories: a) tool life and tool failure, b) tool replacement, c) tool sharing, and d) tool and part loading. The related studies are reviewed in the following sections.

2.1 Tool Life and Tool Failure

Tool life here refers to the "economic tool life" which is defined as the time duration over which a tool can be used to economically perform cutting functions. A tool is said to be failed when its economic life is reached. For example, a tool can be removed (replaced) from work either because it produces unsatisfactory parts due to gradual wear or because it has suffered a catastrophic injury (broken) before having a chance to be worn out. Therefore tool life is not a constant and is affected by many factors, such as cutting speed, feed rate, depth of cut, material of the workpiece. Depending on the cutting condition, it can be regarded as either a deterministic variable or a random variable following certain probability distribution. Accordingly, both deterministic and probabilistic approaches are used to estimate tool life.

With the deterministic approach, an empirical equation is usually used to predicate tool life. The parameters in the empirical equation are determined based on experiments. Taylor (1907) proposed the first empirical tool life equation which describes the relationship between tool life and cutting speed. Later, Taylor's tool life equation is extended by Cook (1973) to accommodate the effects of feed rate and depth of cut. The studies carried out by Malakooti et al (1990), Nagasaka and Hashimoto (1988) are also along this line.

Although the equations obtained based on the deterministic approach are easy to use, they fail to account for ageing, chatter, heterogeneity of the workpiece and other probabilistic characteristics. Therefore, these equations may not be reliable for decision making (Fenton and Joseph 1979). More accurate analysis of tool life can be obtained only when the probabilistic nature of tool life is taken into account (Commare et al. 1983).

Experimental results have shown that standard distribution such as Weibull, exponential, normal, log normal, and gamma distributions as well as their combinations can be fitted to describe the failure pattern of tools (Wagner and Barash, Ramalingam 1978, Ramalingam and Watson 1978a, 1978b). The type of distribution used to predict tool life largely depends upon the machining conditions and tool/part materials.

Ramalingam and Watson (1978a, 1978b) have reported that, in high speed cutting operations, for constant and time-independent hazard machining condition the probability density function of tool failure is exponentially distributed. They have also shown that, in the case of time dependent hazard rete, tool life distribution can be Weibull or a combination of Weibull and exponential distribution. Using an analytical approach, Ramalingam (1978) suggested that tool life follow a log-normal distribution when tool failure is mainly due to the crater wear.

In many machining operations tool quality decreases by use. In such a instance, tools fail more probably due to the gradual wear which is a complicated process involving shank wear, shipping and tempering. There are several tool life studies with the respect to wear mechanism. Wagner and Barash (1971), for instance, reported that for high-speed steel turning tool life values are subjected to a statistical distribution which can be approximated by a normal distribution. Hitomi et al (1979), based on experimental investigation, showed that the logarithmic normal distribution conforms for the tool life distribution. Ramalingam et al (1978) carried out a series of tests with brittle carbide tool inserts to demonstrate tool life distribution. Their results were in agreement with Weibull distribution.

2.2 Tool Replacement

One of the most important tooling problems is tool replacement problem which determines the proper time at which a tool in use or a failed tool on the machine is substituted by a new tool. Tool replacement decisions have a direct impact on the machining economics. If tools are replaced too early and too often, the remaining tool capacity is wasted and excessive machine idle time will be resulted. On the other hand, if the tools are replaced too late, the in-process tool failures may happen which generally cause serious quality problems -- usually the part is either scrapped or reworked. The three widely adopted tool replacement policies in industry are:

(a) scheduled tool replacement policy (STR);

(b) preventive planned tool replacement policy (PTR); and

(c) failure tool replacement policy (FTR).

Under scheduled replacement, the cutting tool is replaced in a specific scheduled time or upon failure, whichever occurs first. In the preventive replacement tool is replaced when a certain number of parts are completed or upon failure. In the last case a tool is replaced when it has failed.

To examine these policies, Commare et al (1983) presented an analytical model to determine optimum cutting conditions which minimize the unit production cost. They suggested that, if an in-process tool failure occurs, a penalty cost should be considered both for the workpiece affected and the time spent to rework the workpiece. They concluded that the scheduled tool replacement policy is always more convenient to use and gives the lowest production cost as compared to the other two policies. Their work was later extended to incorporate the effect of preliminary running-in of the cutting tools by Diega and Passannati (1984).

Zhou et al (1990) developed an optimal tool replacement model based on actual tool wear status. They assume that tool cutting capacity is directly proportional to the available particles on the cutting surface and can be related to tool wear measure. The objective is to find the optimal replacement point so that the unit production cost can be minimized, and the tool utilization and in-process tool failure are balanced.

Galante and Lombardo (1991) reported a new replacement approach which is able to update itself in process by means of an iterative procedure and converges to a conditioned optimum. The effectiveness of the approach is demonstrated analytically.

More recently, Agapiou (1992a, 1992b) studied the tool replacement problems for multifunctional part configurations. Both analytical models and experimental analysis have been conducted.

The other studies include the work of Ravignani et al 1978). But in their models the penalty cost is associated with the probability of tool failure at the scheduled tool replacement time. This implies that the influence of the probability of tool failure during the cut of the pieces before the scheduled tool replacement time is neglected.

As Gray et al (1993) pointed out, most of previous tool replacement studies have the following problems: (a) they consider only one machine in isolation, (b) the relationship between the processing rates and the tool replacement policy is ignored, and (c) they tend to overlook the impact of sharing tools on setup times.

2.3 Tool Sharing

Very often, in a cutting operation a required tool may not be available in the tool magazine of the machine at which the concerned operation is scheduled. This is mostly caused by: a) the required tools for this operation on this machine have been exhausted, or b) the required tools are not mounted on the tool magazine. The problem can be resolved by either sending the part to another machining centre where those needed tools are available or borrowing the required tools from tool a neighbour machining centre. In this research the second policy is referred to as *tool sharing*.

In an automated manufacturing system, tool sharing may or may not be applicable depending on many factors such as the availability of tool transportation facilities. Tool sharing, however, once implemented could significantly affect system performance and hence has been adopted as an important tooling strategy in practice (Gaalman et al, 1987). Han et al (1989) reported the following advantages for employing tool sharing:

- Because parts do not move, there is no need to reposition the workpiece or recalibrates the position of the tool head which results in a higher cutting precision;
- Since a part is processed by only one machining centre, a part is delivered into the shop only when a machining centre becomes available, thereby resulting in less work in process.

However, they also mentioned that this policy resulted in tool-move delay time, whicn would be increased when the tool to be borrowed was in use by other machining centre. They proposed a non-linear programming model for the loading of a set of tools to different machining centres, where each part visits only one of the machining centres for its entire processing. The objective was to minimize the amount of tool traffic among the machining centres and between a machining centre and tool crib under a reasonable work load balance. Nevertheless, the complexity of the model regarding the various flow-control strategies results in an insufficient interacting with complex systems. Apart from the study of Han et al (1989), the reported work on this topic is scant. Limited research is reviewed as follows.

Hankins and Rovito (1984) compared two tool allocations and distribution strategies through a case study by using computer simulation. Strictly speaking, however, this may not be considered as tool sharing problem.

Tang and Denardo (1988) have used a heuristic to minimize the total number of tool switches for a single machine where the requisite but unavailable tools must be brought to the machine and the time needed to switch tools is significant relative to the processing time. The impact on thc other machines are not considered.

Based on a Petri net approach, Reddy et al (1992) examined the effect of tool sharing on the machine utilization and presetting time in a flexible manufacturing system with only two machining centres. They concluded that the centralized tool management with tool sharing was advantageous. Furthermore, it was shown that by increasing the number of shared tools among the machines, the difference in production rates in the two cases (with and without tool sharing) became wider.

Kolahan (1993) approached the problem from different viewpoint. He evaluated the effect of tool sharing on system reliability and tool copies requirements and concluded that tool sharing resulted in a significant reduction in spares cost and required tool magazine capacity while maintaining the same level of system reliability throughout the production period.

2.4 Tool and Part Loading

The tool and part loading problem is specified as the decisions of assigning parts and required tools to the machines in the system subject to certain technological constraints. Two approaches have been used for loading tools and parts (Stecke and Kim 1986), namely:

- *Flexible Approach.* With this approach, once the processing of some part type(s) are finished, the space in tool magazine is freed up so that new part type(s) and the required tools can be introduced to the system.

- *Batching Approach.* This approach attempts to partition the part types and their required tools into separate batches. With this method, all of the production requirements of the selected part types are produced continuously in one batch.

Due to its complexity in analysis and implementation, the flexible approach has not been addressed adequately. Reported studies include the work by Stecke and Kim (1986, 1988). Their findings are that the flexible approach generally results in higher system utilization and needs less pallets as compared with the batching approach.

Though the flexible approach is promising in improving the system utilization, the complexity in planning and scheduling diminishes its practical value. Therefore the batching approach is widely used in reality and has attracted more attention in the literature.

Whiteny and Gual (1984) proposed a heuristic model to minimize the number of distinct batches and then balance workloads within each batch sequentially. Their heuristic is iterative and uses estimated performance indices chosen by the user.

Hwang (1986), and Hwang and Shogan (1989) pointed out that the one of the most critical restriction in the batch production was the tool magazine capacity and thus the tool magazine capacity should be considered in the batching decisions. They accordingly developed a simply integer programming model to solve the batching problem with a objective of maximizing the number of part types accommodated in a single batch. They further argued that in doing so the total number of batches and hence the over required setup time can be minimized.

A simulation approach was applied to the scheduling problem by Carrie and Perera (1986). The tool availability constraints were accommodated in their simulation model. However, they assume that there is no tool failure in the system and the rate of the tool changes due to the tool wear is almost constant. Their simulation results indicate that the minimization of tool changes using part launch sequence or by dynamically selecting jobs from queues is ineffective.

O'Grady and Menon (1987) developed a linear programming model which takes tools requirement, machine capacity, tool availability, and due dates into account. The objective function is a summation of the weighted deviations from the desired level of some system performance measures. However, it does not take into account the periodical nature of the problem. Besides, by varying the weighting factors a wide variety of solutions could be obtained, and there was no logic basis for establishing values of the weighting factors. Similar to the many other models, the data requirements are considerably high.

To address the combined problems of batching, loading and tool configuration in flexible manufacturing systems, Co et al (1990) developed a mixed-integer programming model. A four-pass heuristic algorithm is used to solve the problem. They indicate that the need for batching is primarily due to the tool magazine constraint. It follows that tool magazine must be reconfigured after each batch is processed.

Minis et al (1990) introduced a method considering setup and run times for the evaluation of the capacity requirements and pallets traffic as opposed to individual part traffic in the minimization criterion. Sundaram (1988) considered routing flexibility in the design of cellular manufacturing systems. The proposed method selects the machine with which the particular operation will be performed to minimize the total capital investment costs on machining centers. Part demands and available machining time are under consideration. The model, however, was applied to a system consisting of only two machining centers.

Liang and Dutta (1994) solved the combined problem of part selection, machine loading and tool configuration in a class of flexible manufacturing systems. They considered several system constraints such as the tool magazine capacity and available machining time to determine the optimum number of tool copies on each machine for a given production batch. However, it is noticed that their model is based on the basic assumptions that enough copies of each tool type are available, tool magazine refilling is not allowed, and tool lives are constant.

2.5 Limitations of Previous Studies

In summary of the literature reviewed, it is observed that the following problems exist in the previous studies and further investigations are required:

(I) Tool replenishing problem has not been addressed

Tool replenishing is a problem of determining how the tool magazine should be re-filled with new tools after a certain period of production. This is a unique tooling problem present in automated manufacturing. As reviewed earlier, most planning or loading decisions are made based on the tool magazine capacity. The number of different part types that can be simultaneously processed by a system and the production batch size are largely restricted by the tool magazine size. If the tool transportation facility is not available, such an approach may be understandable. However, as reported by Gray et al (1993), many automated manufacturing systems in use are equipped with sophisticated tool transportation facility. By properly using these tool transportation devices, the exhausted tools in the magazines can be easily refilled. In doing so, the production batch size can be significantly increased and more part types can be accommodated for simultaneous processing. Another expected benefit is the reduced machine downtime.

(2) Tool configuration problem has not been specifically investigated

This problem deals with the decisions of distributing tools among the machines. In automated manufacturing environment, the tool configuration problem is particularly important in that most machining centers are identical and the system performance greatly depends on how the machines are tooled. For example, if every machine in a machine cell is equipped with a identical set of tools, each machine cell can be regarded as a functional machine group and the system becomes a functional manufacturing system. The main advantage is the increased machine utilization due to the flexible routes within each machine cell. The drawback is the increased inter-cell material handling and waiting time.

Similarly, if each machine cell is tooled in such a way that a majority of the operations of a part can be processed within it, a group technology (GT) layout system can be achieved. The benefits include the reduced material handling, reduced setups and waiting time. The Disadvantage is the reduced part scheduling flexibility. Another interesting point is that different GT system can be achieved with the same set of machines by simply reconfiguring the tools for each machine cell. Further a mixed or hybrid system can be designed if some machine cells are identically tooled while the others are non-identically tooled.

However, a challenging issue is how to dynamically reconfigure the machine cells based on the product mixes that change frequently from period to period. Due to the large number of factors which affect decision process, the analytical models cannot be applied because they generally demand considerable amount of computational time and are not capable of respond the product mix change efficiently. A new approach therefore should be considered.

(3) Enough attention has not been paid to tool sharing problem

As indicated by Gray et al (1993), most tool replacement related studies focus on single-tool and single-machine problems. Based on such an approach, a failed tool is replaced with a tool from the tool magazine of the same machine. If the tool spares of this type are exhausted on this machine, then the machine has to stop operating even if several spare tools of this type are available in a neighbour machine. This will cause additional machine downtime. However, if tool sharing is allowed, additional tool flow and human intervention are inevitable. Further a potential problem is that the neighbour machine from which the tool(s) are borrowed may sometimes have to wait for the tool(s) to be returned before the next operation. It follows that a careful plan has to be made before determining whether tool sharing should be adopted.

(4) The collective study of the various tooling problems is not available in the literature

Due to the complexity of tooling problems, the previous studies tend to analyze each tooling problem in isolation with other tooling issues. However, different tooling issues such as tool configuration, tool sharing, tool replenishing, tool replacement, as well as the tool and part loading patterns have strong effects on each other and collectively influence the overall system performance. The best solution obtained by solving a particular tooling problem does not necessarily give an overall good solution and hence a collective study of various tooling problems is needed.

In view of the above, this thesis will be directed towards the development of an approach the joint study of the tool configuration, part loading, tool replenishing, and tool sharing problems. Considering the fact that tooling problems involve a large number of decision variables and are very dynamic in nature, analytical models may not be able to provide quick solutions for daily, sometimes hourly decision making in shop floor. Moreover, because simulation can accommodate almost any detail of a real problem, a simulation approach is employed in this study.

Chapter 3

MODEL DEVELOPMENT

In this chapter, a simulation model is developed to analyze the effects of tool sharing, tool replenishing, and tool configuration under different part loading patterns and loading levels. The background information of the automated manufacturing systems to be simulated is given in section 3.1. The operational issues and tooling policies to be examined are discussed in section 3.2. Finally, the simulation model is presented in section 3.3.

3.1 System Components and Layout Patterns

The automated manufacturing systems studied in this thesis consist of the following major components:

a. CNC (computer numerically controlled) machine tools

b. automated material handling and storage system (ASRS)

c. computer control system

d. loading/unloading facilities

These components and system layout patters are briefly introduced as follows.

CNC machine tools

CNC machine tools are typically machining centers that are capable of performing various machining operations for different parts if the required tools are available. A tool magazine with limited tool slots is generally provided for each machining center. Each machining center is also equipped with an automatic tool changer which can replace a tool within a few seconds. Most automated manufacturing systems in use comprise a number of general-purpose machining centers (vertical or/and horizontal machining centers) (Maleki 1991). In this study, all machining centers within a machine cell are assumed to be identical and general-purpose machines. However, the machining centers in different machine cells may be of different types. An input part buffer in front of each machine and an output part buffer at the back of each machine are also assumed. These locations are logical positions, although physically they are the same.

Automated material handling and storage system

Typical material handling equipment used in an automated manufacturing system includes automated guided vehicles (AGVs), conveyors, and robots. The AGVs and conveyors are used for the primary material handling activities, i.e., to transport material between machining centers and warehouse, as well as between machining centers. The robots are mostly employed for the secondary material handling tasks, e.g., machine loading and unloading. As compared with conveyors, AGVs are more flexible in path selection and therefore are most widely used. They can carry from 35 to 120,000 pounds at speeds ranging from 40 to 200 feet per minute. Many are guided by onboard computers and thus can automatically perform the material handling activities between warehouse and machines, and between machines. In automated manufacturing systems, an automated storage and retrieval system (ASRS) is usually used to store, track and retrieve the raw material, parts (finished parts and work in process), tools, fixtures and pallets. The part refixturing between successive operations is also performed in the ASRS. In this study, all the primary material handling activities are assumed to be carried out by AGVs and the secondary done by robots.

Computer control system

The operation of an automated manufacturing system is computer controlled. The functions of the computer control system include the following:

(1) Machining process control

This type of control is typically done by the local computers. Each machining center is generally equipped with its own computer for loading and editing NC programs, and performing the machining operations. During the machining process, the local computers control the tool path, cutting speeds, depth of cut and feed rate. In an integrated automated manufacturing system, the local computers are connected to the central computer so that the activities of the machining centers can be coordinated by the central computer.

(2) Production control

This function includes decisions on product mix and loading level (rate of input of the various part types into the system), operation sequencing, as well as fixture and pallet scheduling and control.

(3) Traffic control

This refers to the regulation of the material handling system, the material flow and tool flow. The computer will monitor the AGV and AGV path status, and schedule the traffic flow to avoid possible collisions and reduce waiting time.

(4) Tool control

The computer system monitors the tool status, i.e., keeps track of the tool life of each tool copy and updates the number of good tools and failed tools on each machining center. Based on the tool status the computer will accordingly make tool replacement, sharing or replenishing decisions.

Loading/unloading facilities

To reduce the setup time, the parts are mostly loaded/unloaded off-line. Therefore, the concept of palletized machining has been widely accepted (Luggen 1991). The common practice is that a part is fixtured onto a pallet which is designed to accommodate different part configurations in a central or local loading/unloading area before the next operation. If a machining center is available for the operation, both the part and pallet are loaded to the machining center. The loading or unloading of the part and its pallet typically requires less than one minute (Luggen 1991). The central loading/unloading station is usually located in the warehouse (ASRS) and the local loading/unloading stations could be located nearby the machine cells that they serve. To provide enough flexibility, both central and local loading/unloading facilities will be accommodated in the simulation model.

Layout patterns

The machining centers in an automated manufacturing system can be physically laid out in different patterns. The layout pattern is often affected by the shop floor dimensions, the material handling system, the warehouse (ASRS) and loading/unloading areas, and other planning considerations. The four commonly adopted layout configurations, in-line (Figure 3.1a), loop (Figure 3.1b), ladder (Figure 3.1c), and open-field (Figure 3.1d), are be incorporated in the simulation model.

3.2 Operational Scheme

Unlike analytic models, a simulation model can incorporate almost any level of detail about the real system. An automated manufacturing system can be operated using different part loading policies, tool configuration policies, tool sharing policies, and tool replenishing policies. All these policies will be modelled in this study. In this section these policies, operational strategies, and system performance measures are described. To facilitate discussion, the part flow in the system and the concepts of operation and processing steps are first introduced.

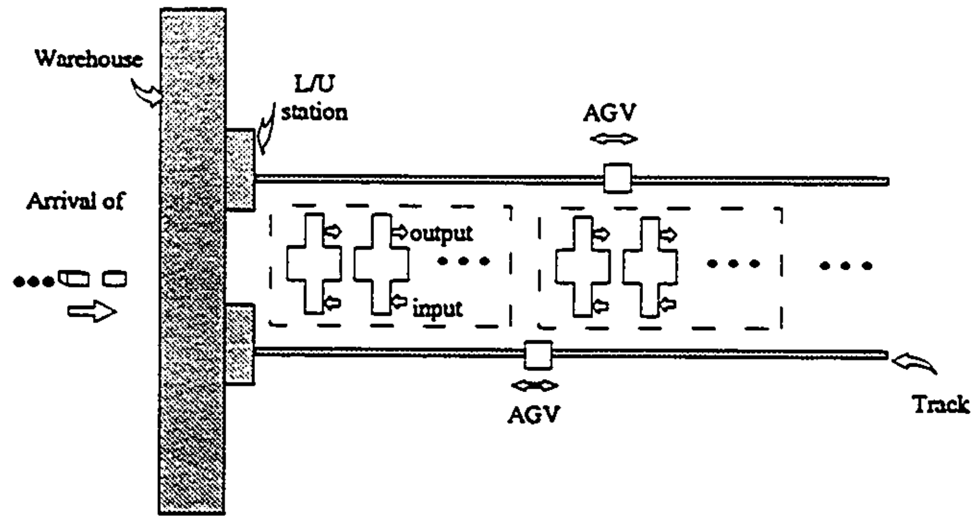


Figure 3.1a In-line layout

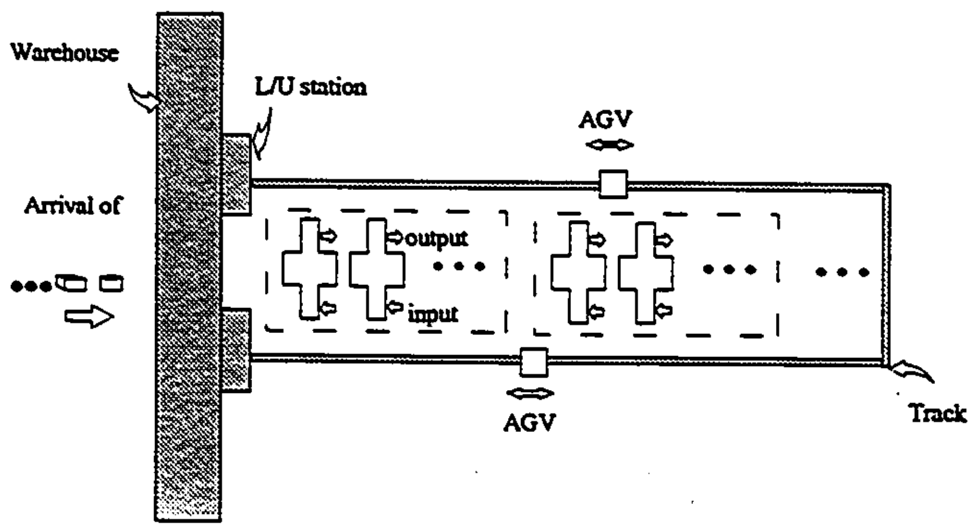


Figure 3.1b Loop layout

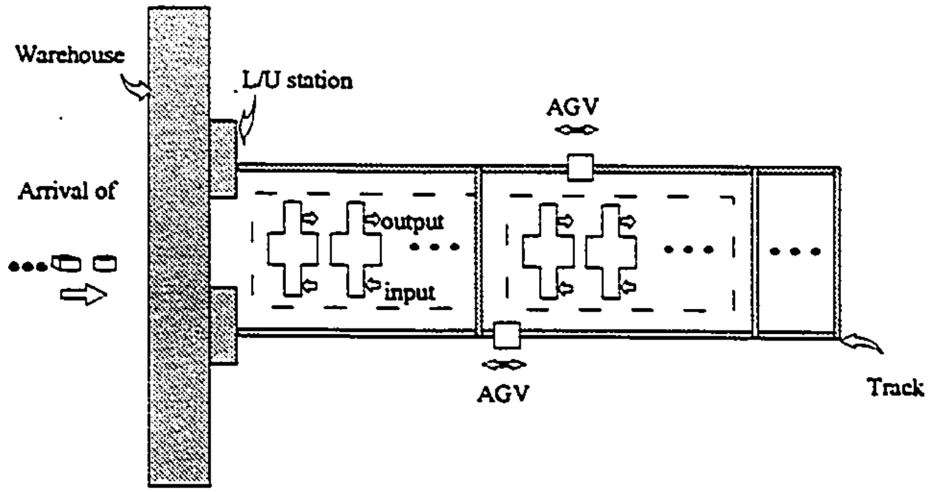


Figure 3.1c Ladder layout

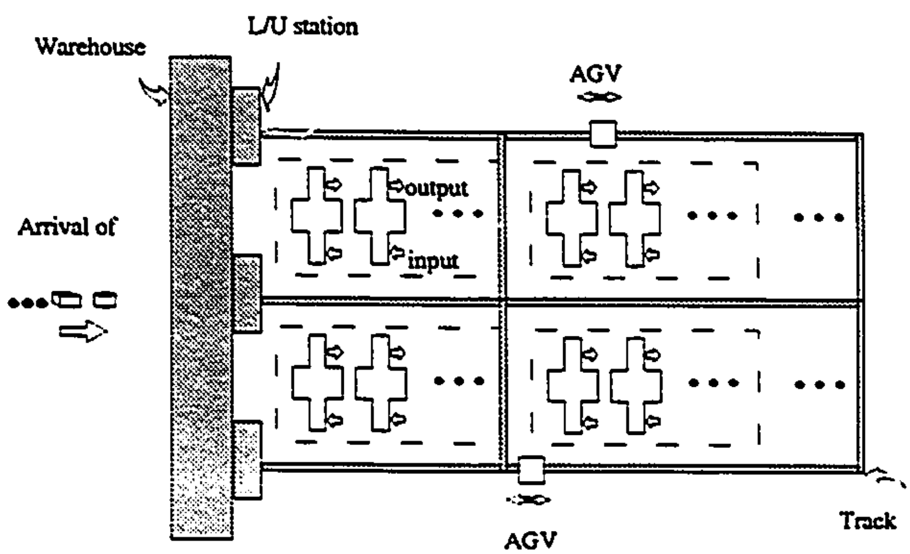


Figure 3.1d Open field layout

3.2.1 Operation, Processing Step, and Part Flow

Prior to the processing of a part, the required operations and process plan should be determined by the process planner or retrieved from the process plan database if available. Due to the flexibility of the machining centers, several processing steps can be carried out on a machining center with a single setup. A processing step is defined as the machining work performed by a single type of tool. A set of processing steps that can be completed on a single machine with a single setup is defined as an operation. Based on the geometry and physical characteristics of a part and availability of fixtures, processing steps may be performed following a specific sequence. The concepts of operation, processing step, and sequence are illustrated in Figure 3.2 and Table 3.1 for a casting case. The typical part flow in an automated manufacturing system is shown in Figure 3.3.

3.2.2 Tool Configuration Policies

In an automated manufacturing, many types of tools are used. Several copies of each tool type are available. The distribution of the available copies of these tool types among the machining centers is referred to as tool configuration (Lenz 1988, Liang and Dutta 1994). In this study the following commonly used tool configurations will be examined.

1. Identical tooling

Machines within the same cell are equipped with the same set of tool types. The number of copies of each tool type is also the same on each machine in the cell. The

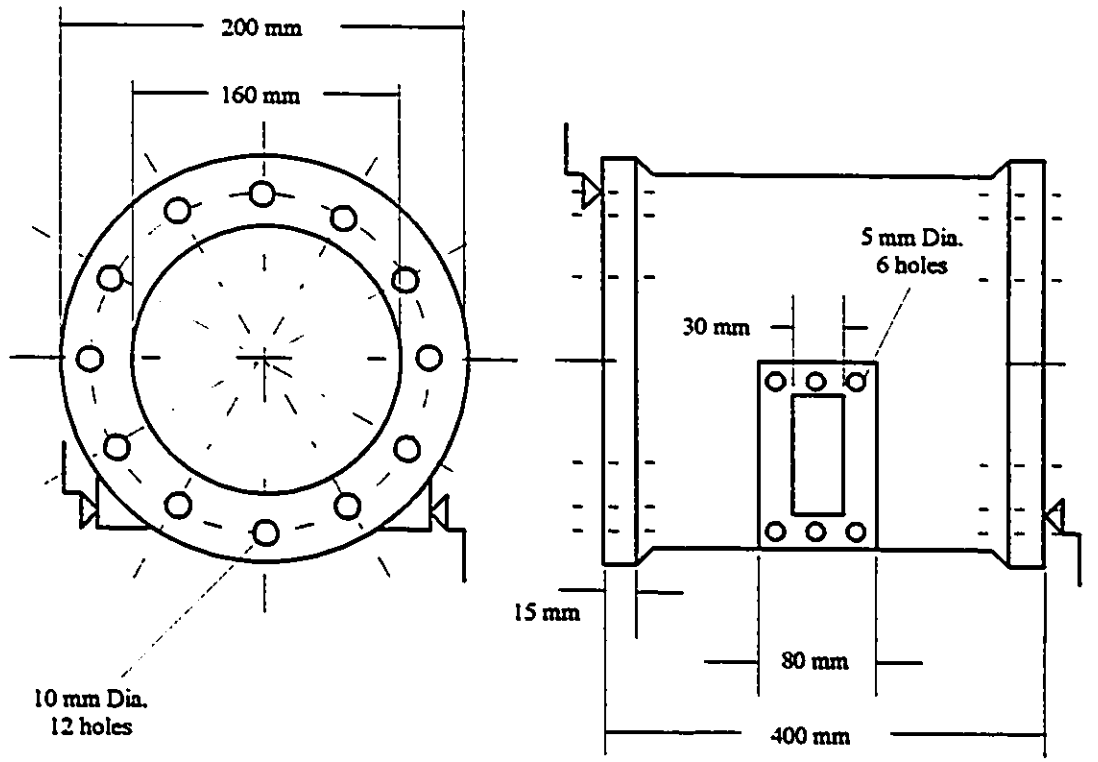


Figure 3.2 A casting case

Operation	Processing steps	Tool description	Processing Duration
1	1- Rough mill top face	Face milling cutter (M1)	8
	2- Finish mill top face	Face milling cutter (M2)	6
	3- Drill 5 dia. holes (12)	Drill - 5 mm diameter (D1)	12
	4- Ream 5 dia. holes (12)	Ream- 5 mm diameter (R1)	9
2	1- Rough mill bottom face	Face milling cutter (M1)	8
	2- Finish mill bottom face	Face milling cutter (M2)	6
	3- Drill 5 dia. holes (12)	Drill - 5 mm diameter (D1)	12
	4- Ream 5 dia. holes (12)	Ream- 5 mm diameter (R1)	9
3	1- Rough mill side face	Face milling cutter (M3)	12
	2- Finish mill side face	Face milling cutter (M4)	6
	3- Drill 10 dia. holes (6)	Drill - 10 mm diameter (D2)	8
	4- Ream 10 dia. holes (6)	Ream - 10mm diameter(R2)	7
4	1- Rough mill side face	Face milling cutter (M3)	12
	2- Finish mill side face	Face milling cutter (M4)	6
	3- Drill 10 dia. holes (6)	Drill - 10 mm diameter (D2)	8
	4- Ream 10 dia. holes (6)	Ream - 10mm diameter(R2)	7

Table 3 1 Process plan for the casting case shown in Figure 3.2

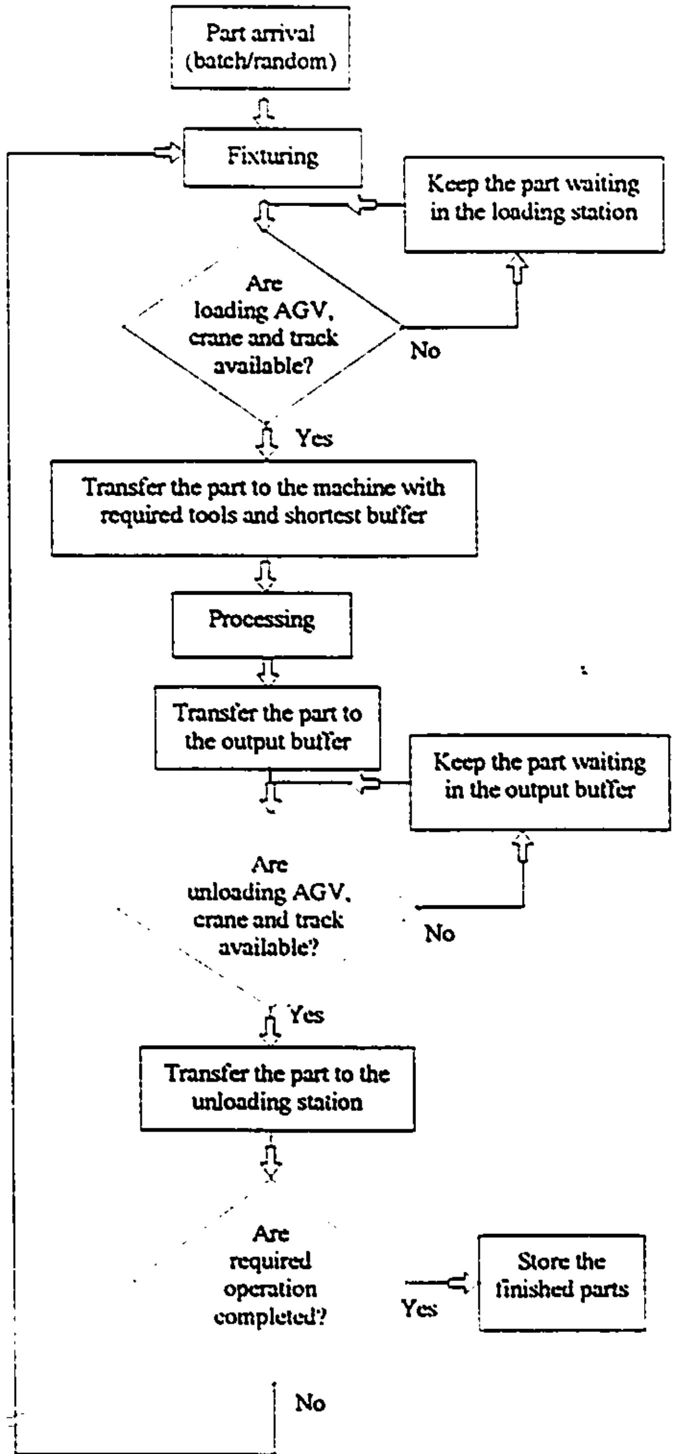


Figure 3.3 Part flow chart

machine cells configured in such a way may be considered as an extension of conventional functional layout system. The difference is that in a conventional functional layout system each machine is capable of performing a very limited number of operations whereas a machining center in an identically tooled machine cells can handle a wide range of operations if the required tools are available. The main advantage of this type of tool configuration is the increased machine utilization due to the load pooling between machines within each group. However, this is often achieved at the expenses of the increased material handling requirements.

2. Non-identical tooling

In this case, machines in the same cell are tooled differently in terms of both tool types and number of copies of each type. If properly planned, group technology (GT) cells may be formed with this type of configuration. The most claimed benefits of this type of tooling pattern include the reduced material handling and waiting time. However, since the production in the automated manufacturing environments is very dynamic in nature and the product mix and production volume may change significantly from one period to another, properly configuring tooling for each machining center could be a very challenging task. Wrongly configured tooling could significantly degrade the system performance.

3. Hybrid tooling

This configuration is a combination of the identical tooling and non-identical tooling. Namely, both identically configured machine cells and non-identically tooled machine cells exist in the same system. The main purpose of adopting this configuration is to achieve some preferred compromise between the identical configuration and the non-identical configuration. However, as many different hybrid combinations exist, the decision regarding which combination should be selected has to be carefully made.

3.2.3 Part Loading Policies

Batch loading

The production in this case is scheduled on a batch basis. The manufacturing system processes different batches of part types during different periods. At the beginning of each period, all the machines are in idle state and the raw material for the batch of parts to be machined during the period are available in the ASRS. The machines stop operating when the current batch has been completed. If the next batch is different from the previous one, the tools on each machine have to be re-configured. The next batch is then introduced into the system. Adopting this loading policy will significantly simplify planning and scheduling decisions, and is advantageous to the reduction of mean flow time.

Random loading

With this loading policy production orders arrive at the system randomly and are loaded to the system immediately whenever the required machines and tools are free. The interarrival times of the orders follow some probabilistic distribution. The required raw material for these orders are available in the warehouse upon or before the arrival of the orders. The system runs continuously unless the production orders are temporarily exhausted. This type of loading is also referred to as flexible loading in the literature (Stecke and Kim 1986). The expected benefit is the increased machine utilization. The obvious drawback is the complicated planning and scheduling.

3.2.4 Tool Replenishing Policies

As indicated in Chapter 2, both tool replacement and tool replenishing problems deal with tool change issues but they are different concepts. The term tool replenishing refers to refilling tool magazines with new tool copies delivered from the tool room. This is a unique tooling problem in automated manufacturing systems. Tool replacement means to substitute a failed tool on the machine spindle with a new tool. The focus of the tool replacement decision is on the tooling problem at single-tool-type level whereas the tool replenishing decision aims at the tooling problem at machine level involving several tool types. Since the tool replacement problem has been extensively investigated and the tool replenishing problem defined in this study has not drawn much attention in the literature, only the tool replenishing policies are examined for tool change related issues in this study.

As it is economically infeasible to replenish every tool immediately when it failed, tool replenishment on each machine is generally carried out periodically. The following tool replenishing policies will be examined:

(a) preventive replenishing

(b) failure tool replenishing

Under both policies, all the tools in the system are reviewed periodically with a fixed review interval. In the first policy, both the failed tools and the over-aged tools are replaced with new tools delivered from the tool room. The over-aged tools are those whose cumulative cutting times are greater than or equal to their expected tool lives. If a tool fails before the replenishing time, a spare in the tool magazine on the same machine will be used. If the tool spares are exhausted on this machine, a tool spare may be borrowed from a nearest neighbour machine. If sharing is not allowed or the tool spares are also exhausted on the neighbour machines in the machine cell, the concerned machine will stop cutting until the next tool replenishment. In the second case, only the failed tools are replaced after each scheduled review. If a tool fails before the scheduled replenishing, the same procedure as used in the first policy applies.

3.2.5 Tool Sharing

As mentioned above, a tool may fail before the scheduled review or replenishing. If the tool spares on the same machine have been exhausted, sharing tools with the neighbour machines in the same machine cell may be advantageous to the machine utilization. The detailed sharing process simulated in this study is shown in Figure 3.4. However, this will inevitably cause excessive cutting tool flow, human intervention, and possible low utilization of the machines from which the tool spares are borrowed. It is therefore very important to examine the overall effect of the tool sharing before production. If the

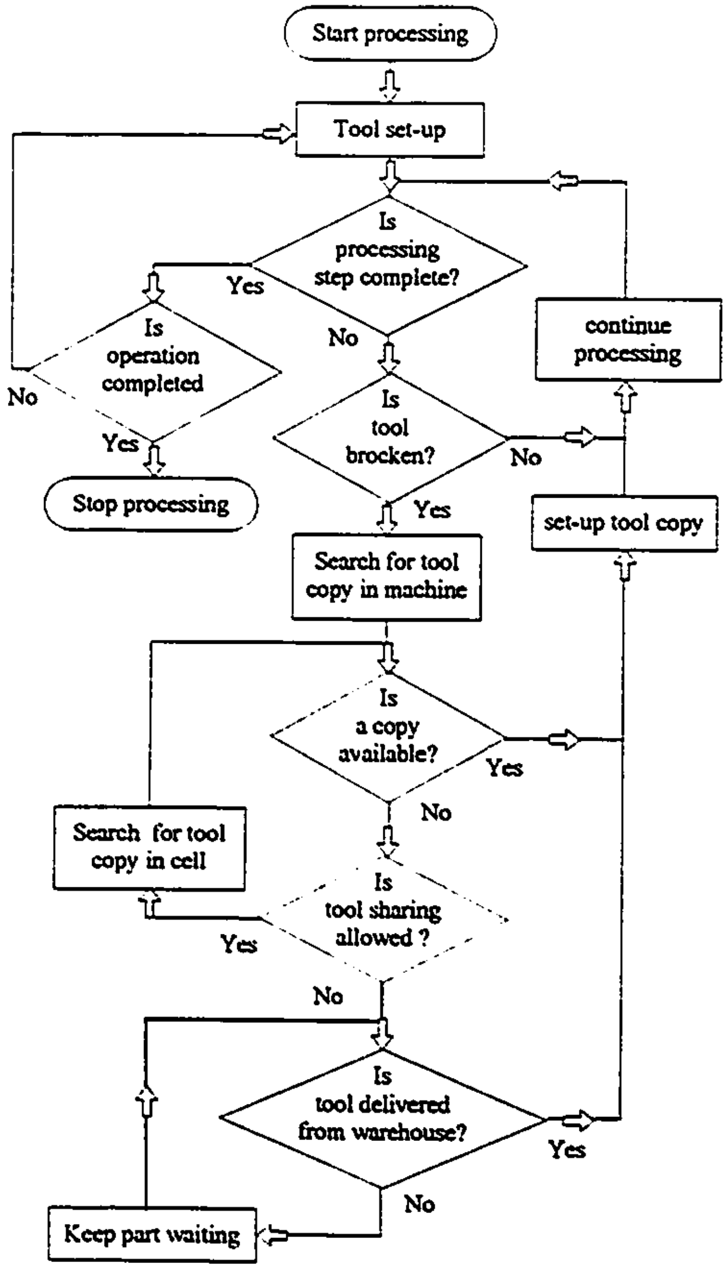


Figure 3.4 Tool replacement and sharing flow chart

overall benefit is significant and can overweigh the disturbing effects such as the cutting tool flow and human intervention caused by tool sharing, tool sharing can be recommended for a particular system. Otherwise, tool sharing should not be employed.

3.2.6 Operational Strategies and Simulation Scenarios

Based on the discussions in sections 3.2.2 to 3.2.5, an *operational strategy* is defined as a distinct combination of tool configuration, loading policy, tool replenishing policy, and tool sharing policy. For instance, the combination of identical tooling, batch loading, preventive tool replenishing, and shared tooling forms an operational strategy. For each strategy, there are many "WHAT IF" questions that should be answered before production. For example, what would be the system throughput or mean part flow time if the system is under loaded, moderately loaded, or over loaded? What if the tool review period is shorter or longer? What if the average tool life is shorter (ordinary tools) or longer (e.g., coated tools)? To answer these "WHAT IF" questions, each operational strategy has to be separately modelled and simulated before production. Such a separate simulation analysis for a specific operational strategy is called a *simulation scenario* in this study.

There exist 24 (3x2x2x2) operational strategies and hence 24 simulation scenarios for a manufacturing system. These strategies (scenarios) are summarized in Figure 3.5. For a given manufacturing system (e.g., a fixed number of machining centers, buffers, tool types, tool spares, AGVs, etc.) the performance of the system could be quite different if different operational strategies are applied. Therefore, a comparative study

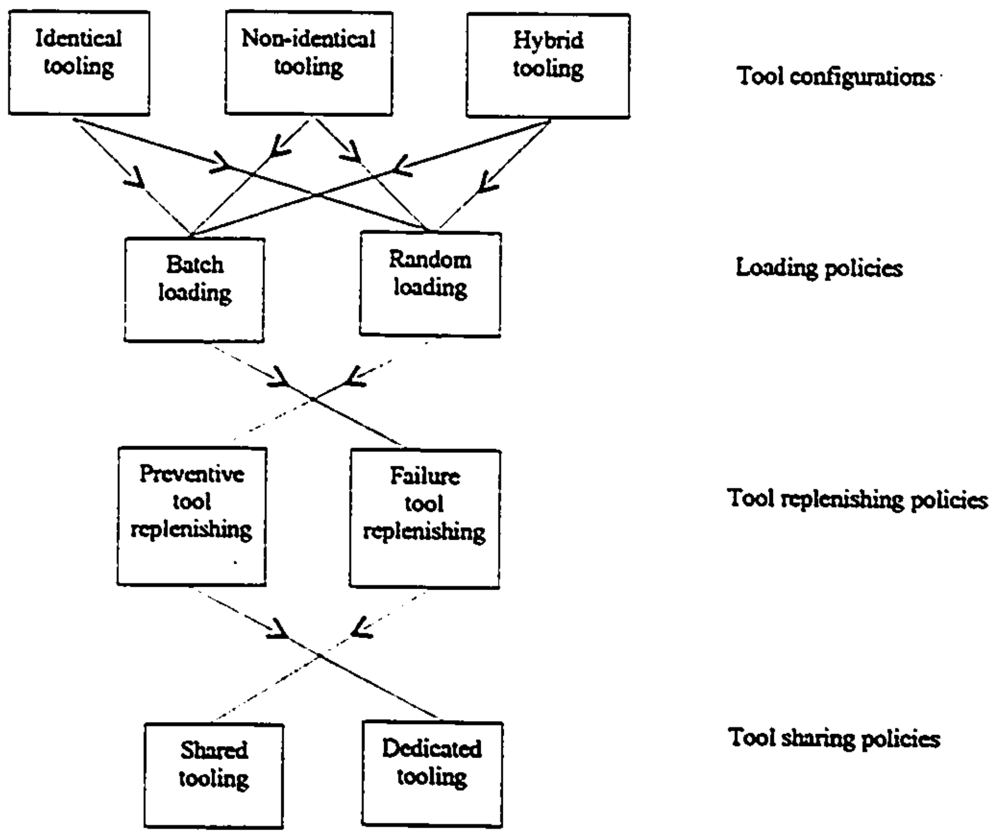


Figure 3.5 Operational strategies

of these strategies is highly desirable. However, to make meaningful comparison, different performance measures should be used for different loading patterns. The performance measures are discussed in the following section.

3.2.7 Performance Measures

1. Performance measures for batch loading: makespan (M) and mean flow time (F)

In this case the production volume of each part type is fixed. A logic aspiration is to complete each batch as early as possible. The associated performance measure is makespan (M) which is defined as the total time required to process a batch of parts.

In the meantime, it is also desirable to reduce the work-in-process (WIP) inventory cost. The WIP inventory cost of a part is directly affected by its flow time, i.e., the amount of time the part spends in the system. The total WIP inventory cost of this batch of parts is influenced by the mean flow time, the average of the flow times of all parts, which is given by

$$F = \frac{\sum_{i=1}^{I} \sum_{p=1}^{P_i} f_{ip}}{\sum_{i=1}^{I} P_i} \tag{3.1}$$

Where

i = part type index, i=1,...,I

p = workpiece index, p=1,...P_i

I = total number of part types to be processed in the current batch

P_i = total number of workpieces of part type i to be produced in the current batch

f_{ip} = the flow time required by the pth workpiece of part type i

In reality, different part types may have different priorities. For example, some part types may be more expensive and thus represent higher work-in-process inventory costs. In this case, the weighted mean flow time will be used:

$$F = \frac{\sum_{i=1}^{I} \sum_{p=1}^{P_i} w_i f_{ip}}{\sum_{i=1}^{I} P_i} \tag{3.1a}$$

where w_i is the weight of part type i and can be normalized by assuming

$$\sum_{i=1}^{I} w_i = I \tag{3.2}$$

The reason for considering both the makespan and the mean flow time is that a plan yielding a shorter makespan does not necessarily guarantee a shorter mean flow time and vice versa. If, however, both of the two measures serve the same goal: selecting the best operational strategy so that the required resource input (machining time or inventory cost) for achieving the fixed output is as low as possible. To provide the flexibility for the user in evaluating operational strategies, both measures will be accommodated in the simulation model.

2. Performance measures for random loading: system utilization (U) and throughput (H)

With this loading pattern the production requirements fluctuate from time to time. The production will continue with possible interruptions due to temporary shorts of orders. Under such circumstances, the production is an on-going process and the makespan will largely depends on when the production is stopped. In other words, the makespan is no longer a random variable and thus cannot be used as a system performance measure. Further, adopting random loading means that mean flow time may not be an important concern since otherwise batch loading policy will be used which generally yields shorter mean flow time. Actually, with the random loading policy some parts could wander in the system for a very long time if not properly scheduled. Therefore, the mean flow time in this case is mostly influenced by the sequencing rule, which is not the focus of this study, rather than the tooling policies. The PRIORITY statement in the SLAM II control statements can help specify any commonly used sequencing rules. Thus examining the effects of sequencing rules using SLAM II does not require much modelling effort and is virtually a trivial issue. As a result, the makespan and mean flow time cannot be used to examine the effects of different tooling policies for random loading. To prevent a part from staying in the system forever, the user can simply use the FIFO (first-in-first-out) sequencing rule.

In this case, to make better use of the heavily invested facilities, the user may be more concerned about system utilization (U) and throughput (H). In other words, the goal is to achieve as much system output as possible for the given system input (a fixed operating time horizon). Though, to certain extent, the two performance measures are equivalent, the users may have their own preferences. Therefore, the simulation model is designed to accommodate both of the two measures.

In SLAM II, the information about the resource utilization rate can be collected using COLCT nodes. However, the utilization rate collected in such a way may be misleading, especially for the complex problems. The reason is that the SLAM II COLCT nodes classify the total time into two categories: utilized time and idle time. As long as the resource, e.g., a machining center, is not freed by a FREE node, it is considered as utilized, regardless of the reason why the machining center is not freed. Very often, however, a machining center is not freed simply because: a) a tool is stuck on a machine due to in-process tool failure and the tool change may take considerably long time, or b) a finished part cannot be immediately unloaded since the output buffer is full. To circumvent this difficulty, the system utilization is calculated by tool utilization in this study. The logic is that the system can be utilized, namely, the parts can be machined, only when the tools are busy. The total machining time is then equal to the total tool busy time which can be easily collected using some user written programs. Therefore, the system utilization is indirectly calculated by

$$U = \frac{\sum_{k=1}^{K} \sum_{c=1}^{C_k} B_{kc}}{JT} \tag{3.3}$$

Where

k = tool type, $k=1,\dots,K$

c = the cth copy of a tool type, $c=1,\dots,C_k$

K = total number of tool types used in the production horizon

C_k = number of copies of tool type k used in the production horizon

J = total number of machining centers in the system

T = length of the production horizon

B_{kc} = cutting time of tool copy c of type k

The system throughput is measured by the total number of workpieces processed in the production horizon, which is self explanatory.

3.3 The Simulation Model

The simulation model for examining tooling effects in various automated manufacturing systems as described in section 3.1 has been developed using SLAM II and FORTRAN.

The manufacturing activities in the automated manufacturing systems can be modelled as discrete events since the state variables such as the number of parts in the system, the tool status, the number of tools on each machine, and machine status (busy or idle) change at discrete points in time. The following three approaches are generally used to model discrete changing systems (Pritsker 1986):

(a) process oriented approach

(b) event oriented approach

(c) combined process-event approach

The process oriented simulation approach models the flow of entities through a system. The process can be symbolically represented by a network consisting of a number of nodes or a series of blocks. A process oriented simulation language translates these nodes or blocks into statements which are then automatically executed as the entities pass through the process. Several process oriented simulation packages are commercially available. For example, GPSS is a process oriented language in which a number of blocks are developed in pseudo-code form. The simplicity of the process oriented approach lies in the fact that the event logic associated with the statement is contained within the symbols of the simulation language. However, although this approach is easy to understand, it is often lack of flexibility due to the limited standard symbols and thus is incapable of modelling complex problems.

The event-based method is another programming approach. With this approach, simulation time elapses and the state of the system remains constant until the next event occurs. This approach is generally more flexible and efficient (in terms of computing time). One disadvantage of this simulation structure is that it is difficult to understand and use, as the object cannot be traced through the system.

SLAM II is a simulation language developed based on the combined process-event approach which allows models to be developed using both process and event oriented approaches. The network part of SLAM II is similar to GPSS and permits the process oriented modelling. In the network part of SLAM II, the user written programs can be inserted using discrete event nodes. This allows the modeller to incorporate complex decision algorithms into the model. For simulating tooling problems, this feature is particularly important in that the tooling problems involve many logic decisions and tool usages (ages and spare levels) have to be dynamically updated. In this study, the overall system was modelled using a network model to simulate the manufacturing processes. The FORTRAN event nodes were used to enhance the decision ability and to provide bookkeeping feature.

The framework of the simulation model is outlined in Figure 3.6. The model basically consists of three modules:

(a) Pre-simulation module

(b) Simulation module

(c) Post-simulation module

These modules are described below.

3.3.1 Pre-simulation Module

This module selects simulation scenarios, initializes simulation variables, and documents various data that are used before and during each simulation run. The major components of this module include scenario selection procedure, control program, process plan files, and tool data files.

1. Scenario selection procedure

This is the first step of the simulation process. As discussed in section 3.2 and

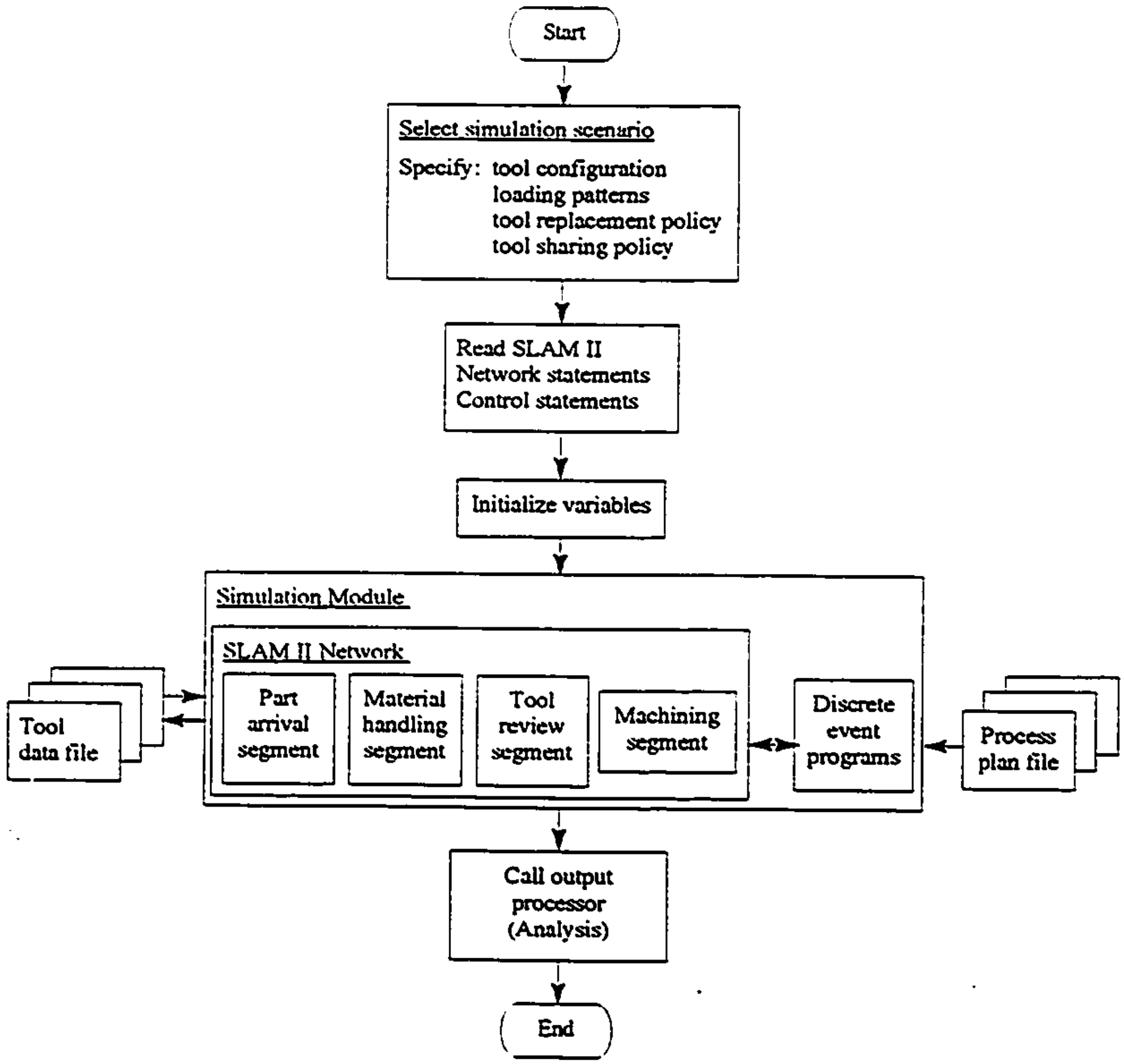


Figure 3.6 Framework of the simulation model

shown in Figure 3.5, for a given manufacturing system (i.e., fixed number of machines, buffers, material and storage systems, layout patterns, etc.) there exist 24 operational scenarios. In reality, however, the user may or may not simulate all the 24 scenarios. If some policies have been pre-specified based on preferences of the management, then only a subset of the scenarios are to be compared. For example, if the user has determined to use batch loading and identical tooling, there will be only four scenarios left for comparative analysis. In the extreme case where only one scenario is preferred, the comparative study will not be performed and only the "WHAT IF" questions concerned in this scenario will be answered by simulation. In any case the scenario(s) can be specified by the user following an interactive procedure.

2. *Control program*

This program is used to specify the following input information:

(a) System configuration

. number of machining centers

. machine types (vertical or horizontal)

. number of AGVs

. number of loading/unloading stations and their locations

. buffers and their sizes

. number of machine cells and the number of machines in each cell

. available tool types and the number of tools of each type

. AGV travel time from ASRS to each machining center

(b) Product mix and volume

. part types to be processed

. order size of each product

(c) Number of simulation runs, random number seeds, and termination rules

3. Process plan files

As illustrated earlier (Table 3.1), a process plan file for a part type specifies the processing steps, operations, the required machines and cutting tools for these operations, and the expected machining time for each processing step. If the process plan file of a part type is already available, it will be linked to the simulation module. The data contained in the process plans will be retrieved from time to time during the simulation process. If a new part type is introduced and its process plan does not exist in the module, the user will be prompted to provide associated information. The process plan files are then updated.

4. Tool data files

Each type of tools has a separate file. Each tool data file contains two types of data: temporary data and permanent data. The temporary data include the initial number of tool copies of each type on each machining center, the tool status (e.g., tool age--accumulative cutting time), and the actual number of copies of each tool type on each machine after a certain processing time. In the tool data files only the initial tool configuration on each machine is specified and recorded. The tool status and the actual number of copies of each tool type on each machine will be monitored by SLAM II global variables in the SLAM II network program. The permanent tool data include the expected life of each tool type and its probability distribution.

3.3.2 Simulation Module

This module consists of a SLAM II network program (Appendix 1), (4+2G+J) discrete event programs (G=number of machine cells, J=number of machines in the system), J USERF programs, an error warning program. The discrete event programs, USERF programs and the error warning program are coded in MS FORTRAN (Appendix 2). The SLAM II network program and these FORTRAN programs function interactively during the simulation process. For simplicity, the FORTRAN programs will be discussed following the SLAM II network flow. The network is divided into several segments. The relationship between the segments is shown in Figure 3.6 and the details of each segment is given in Figure 3.7.

1. Part arrival segment (Figure 3.7a)

In this segment each CREATE node is associated with a part type and simulates the part arrival pattern. For the random loading policy, the interarrival times of a type of parts follow a certain probability distribution which is also specified by the CREATE node. For the batch loading all parts to be produced in this batch are generated at time 0 by these CREATE nodes. Part type is then specified by an ASSIGN node for each newly created entity. The generated parts are routed to the loading segment.

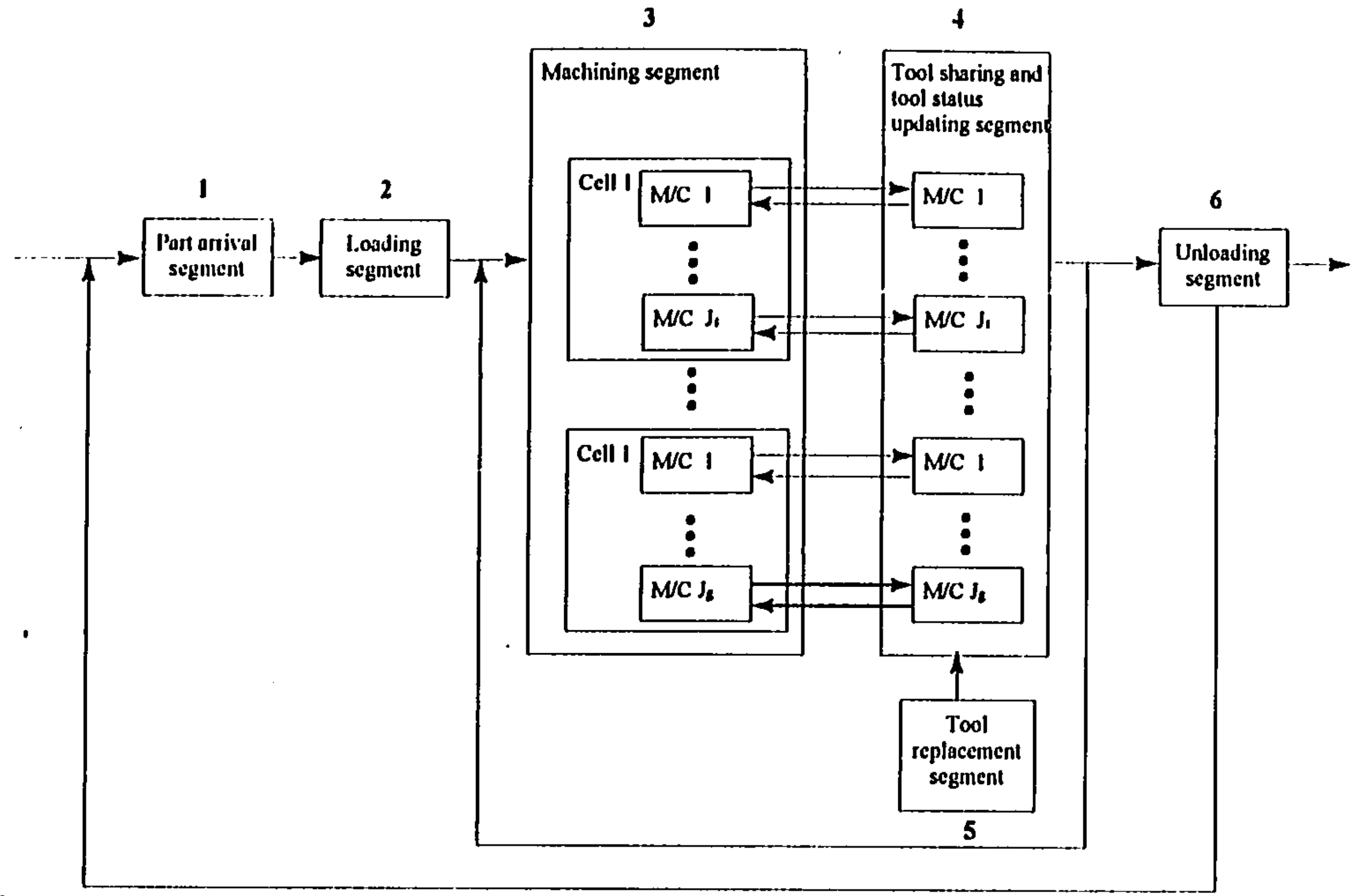


Figure 3.7 The relationship among the network segments

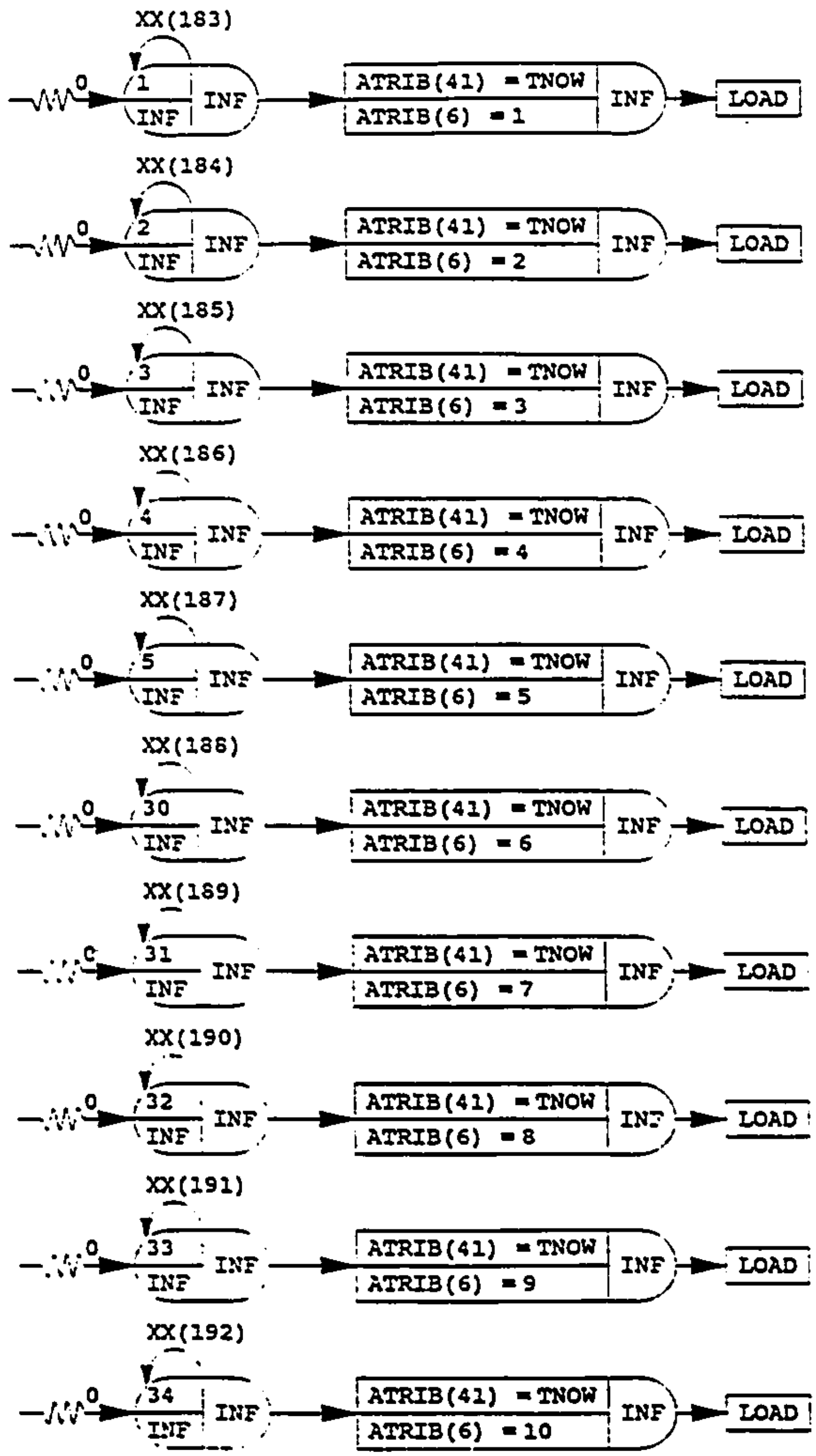


Figure 3.7a Part arrival segment

2. Loading segment (Figure 3.7b)

This segment is used to assign operations to machine cells and machines. At the beginning of the segment, the number of completed operations of each arrival entity (part) is updated by ASSIGN node ASOP. The updated operation number of each part is stored in a SLAM II variable ATRIB(7). Then the part will wait in the ASRS which is represented by a QUEUE node Q1. The EVENT node EVEA embodies a FORTRAN program which logically assigns a part to a feasible machine cell based on machining requirements. If all the required operations for the part have been processed, the EVEA node will send the part to the GO1 node in the unloading segment. The associated statistics are collected and the part leaves the system (network). Then the next part will be selected following certain priority rule which is specified by the user and is incorporated into the EVEA node. Once the EVEA node has decided to route a part to a feasible machine cell, the part will wait in an AWAIT node LOA for a loader and then wait in AGVL AWAIT node for an AGV. If both the loader and AGV are available, the part is loaded to the AGV. The loading time follows a given probability distribution. In the example network segment, a uniform distribution is assumed. After loading the loader is freed by a FREE node (FLOA). The loaded AGV then approaches to the feasible cell as specified by EVEA node if the AGV path is clear. Otherwise it has to wait. This is represented by TRACK AWAIT node.

After a part has been assigned to a feasible machine cell, a series of decisions has to be made in order to assign the part to a proper machine in the cell. The decisions are made in a FORTRAN program represented by the EVENT node following the TRACK

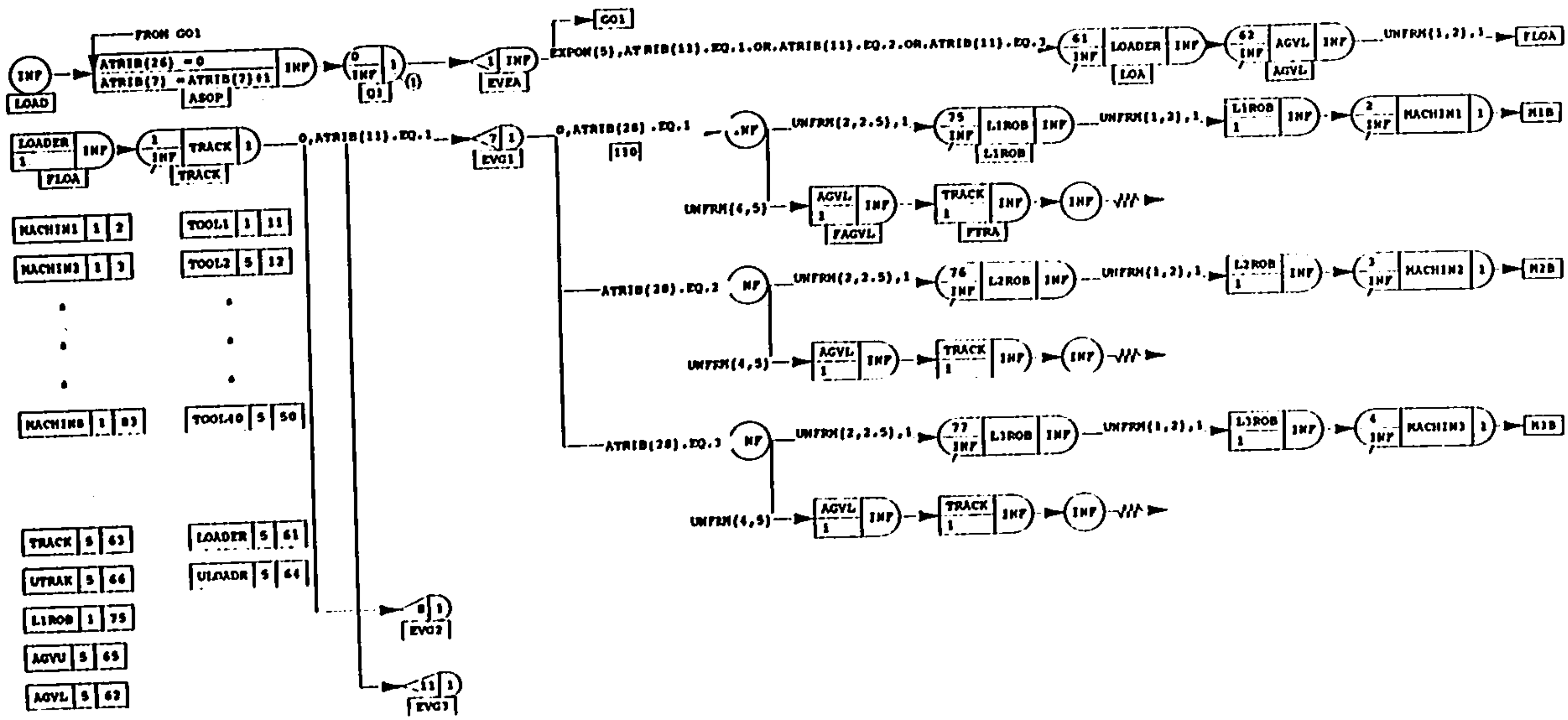


Figure 3.7b Loading segment

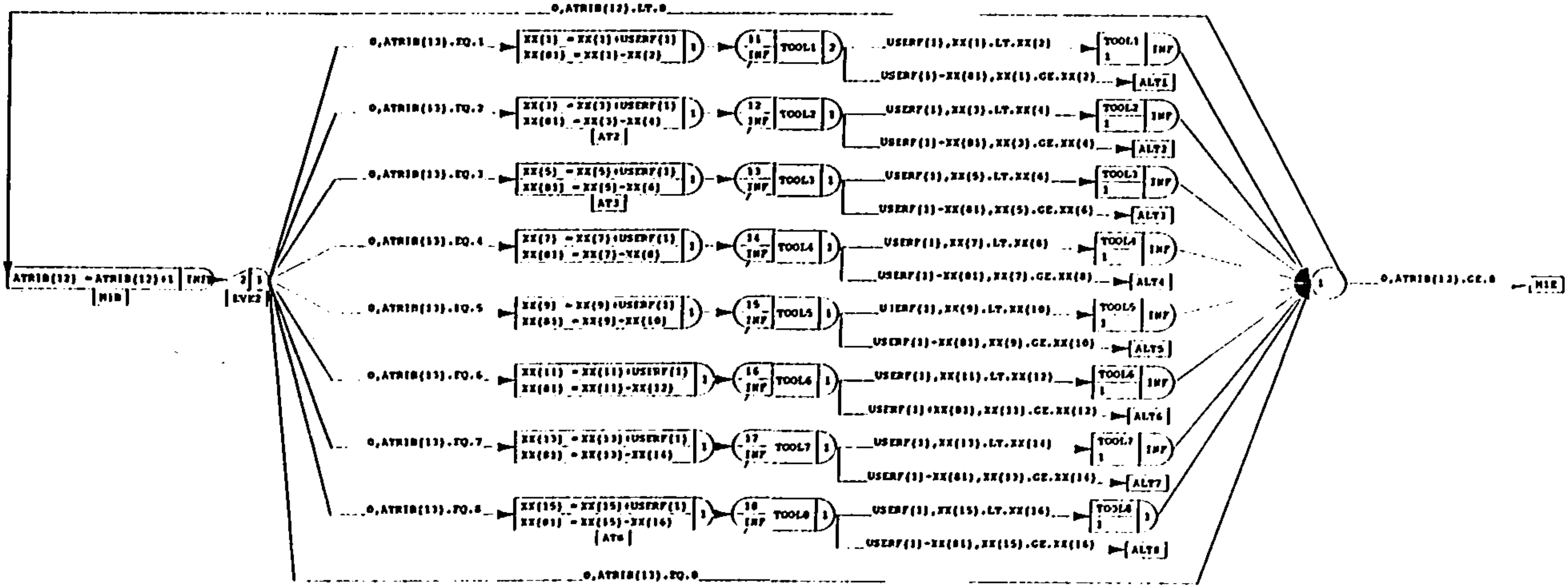


Figure 3.7c Machining segment

all the machines in the same cell are tooled identically, then the sub-segments in the same machine cell are the same. If the machines in a machine cell are tooled differently these sub-segments will be different accordingly but the structure is similar. Therefore, only one such sub-segment is included in Figure 3.7c.

If the machine is idle, the number of finished processing steps of a selected part from the input buffer will be increased by 1 (this is done in ASSIGN node M1B--for machine #1) and the part is loaded to the machine. The rule for selecting the next part from the input buffer is specified by the user. However, if it is desirable different selection rules can be simulated and the best one is employed. If the current job is the first one in the planning period, an EVENT node EVM1 (such a EVENT node is provide for every machine) specifies the tool configuration (tool types and number of copies of each tool type), dispatches the current processing step to its required tool type, and samples the tool life for the first copy of each tool type on the machine. It should be pointed out that the tool configuration has been specified in the tool data file, EVM1 node simply retrieves and "activates" the pre-specified tooling setup. As can be seen in Figure 3.7c, several branches emanate from EVM1 node. Each branch corresponds to a tool type. The dispatching result provided by the EVM1 is reflected by the ATRIB(13) value. For example, if the ATRIB(13) is 1 then tool type 1 is used for the current processing step and the consecutive activities will be performed in the first branch.

In the ASSIGN node AT1, the tool age (after processing the current processing step) XX(1) and the sampled tool life XX(2) are compared. The difference between XX(1) and XX(2) is stored in another SLAM II global variable XX(81). If XX(1) is less than XX(2), the processing step will be performed by the current tool copy and then the tool copy will be freed by a FREE node FT1 (see the upper branch emanating from the AWAIT node T1). Otherwise, the tool copy will also be used until it fails (see the lower branch emanating from T1 node). In this case, the amount of work performed by the failed tool copy is USERF(1)-XX(1). USERF(1) is a user inserted FORTRAN code which retrieves, from the process plan data file, the required processing time for the current processing step and then retains this value till the next processing step entity triggers the ASSIGN node AT1. If a tool failure occurs, certain measures will be taken. This is illustrated in the tool status updating and tool sharing segment. It is worth pointing out that the above mentioned tool failure is in-process failure which is not desirable. In the simulation process this type failure is maintained because in reality the true life is unknown and thus the in-process failure is inevitable. The sampled tool life is simply used to simulate such in-process failures. This does not mean that we really know the true life. If the true tool life were known beforehand, the tool copy would be replaced before the current processing step.

Once the current processing step has been completed, the SLAM II network will check if all the processing steps have been performed. If not, the same procedure will be repeated for the next processing step; if yes, the part entity will be routed to the unloading segment where the part entity is either sent back to loading segment for the next operation (next set of processing steps) or destroyed if all the required operations have been processed.

4. Tool status updating and tool sharing segment (Figure 3.7d)

Each tool type has such a segment. This segment is mainly used to update the number of copies of this tool type on a machine, complete the remaining work left by a failed tool, and arrange tool sharing activities if allowed. Actually, each of this type of segments is embedded in a tool branch of the machining segment. To facilitate illustration, they are separated from the machining segment. Since all such segments have the similar structure, Figure 3.7d lists only two segments respectively belonging to tool types 1 and 2 on machine #1.

If an in-process tool failure occurs, the failed tool is removed from the machine. This is done by an ALTER node ALT1 which reduces the number of tool copies of this type (in this case tool type 1) by 1. Then a new copy of this type is loaded from the tool magazine. The new tool's life is sampled in an ASSIGN node TS1 and is recorded by XX(2). The remaining work left by the previous failed tool is performed by the new tool. If the remaining work is completed by the new tool, the new tool entity is routed back to the FREE node FT1 in the machining segment and the tool is freed. However, if the tool fails in process, the remaining work is updated and again recorded by XX(81). This is simulated by the lower branch emanating from the TS1 node. Then the ALT1 node removes a tool copy from the machine, i.e., NNRSC(6) (number of resource type 6, a global variable whose value equals the number of copies of tool type 1). This procedure is repeated by a SLAM II loop until the number of copies of this type has been exhausted, i.e., NNSRC(6) is 0. In this case, if tool sharing is not allowed (this logic is represented by XX(151).EQ.0 in the lower branch from the LBA node) the machine halts

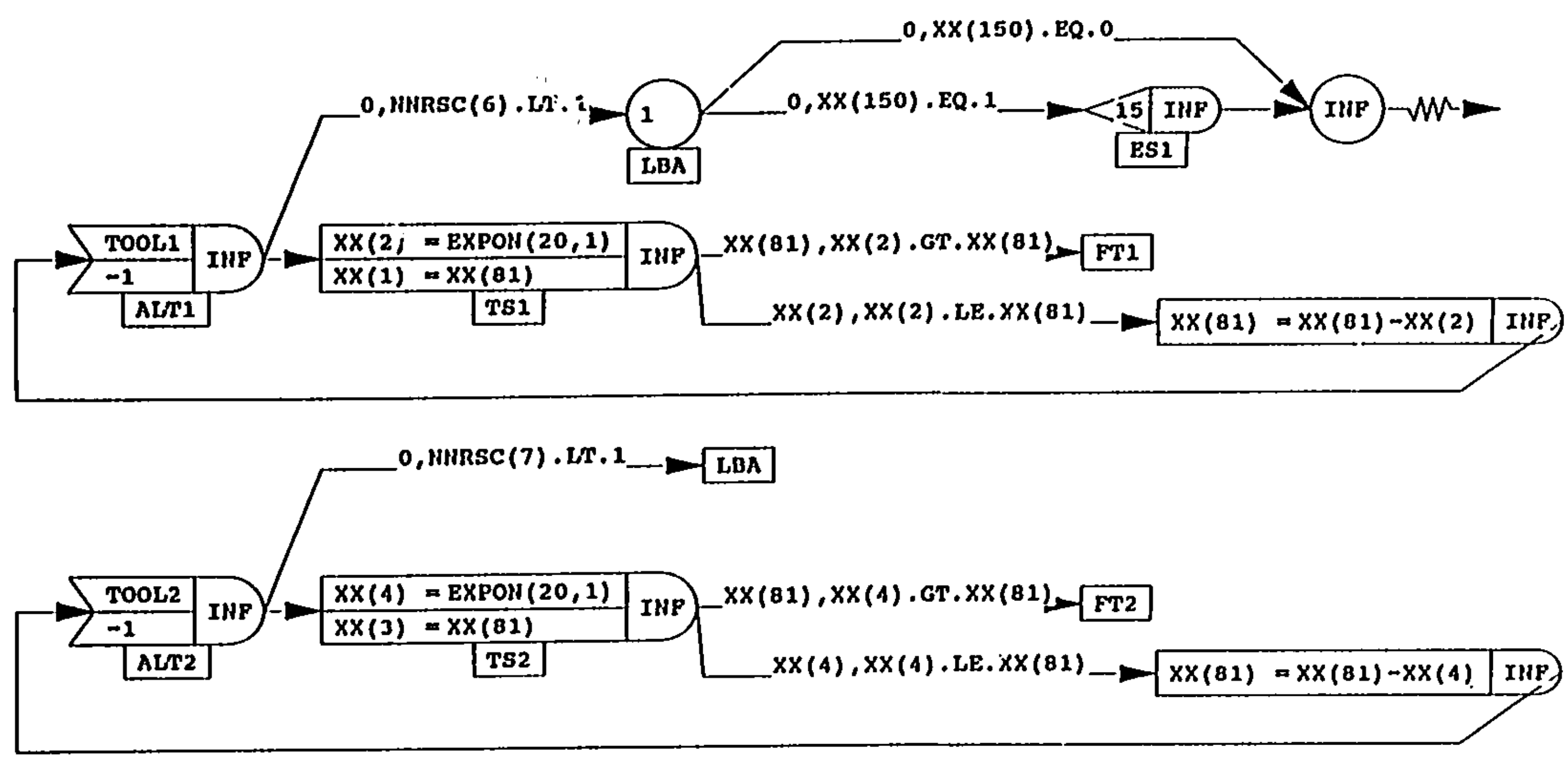


Figure 3.7d Tool status segment

and waits for tool replenishing which is scheduled by the tool replenishing segment. If tool sharing is allowed, i.e., if XX(151) equals 1, the EVENT node ES1 is activated to schedule the sharing activities. All the sharing activities in machine cell #1 are arranged by ES1 node. Such an EVENT node is provided for every machine cell.

5. Tool replenishing segment (Figure 3.7e)

This segment implements the two tool replenishing policies. However, it should be pointed out that the term "tool replenishing" here means to refill the tool magazines with the tools newly delivered from the tool room. The tools borrowed from another machine as described in the previous segment are not considered as replenished tools.

At the beginning of each simulation run, the CREATE node generates an entity which triggers the tool replenishing program represented by an EVENT node EVNR. Both of the two tool replenishing policies, i.e., preventive replenishing and failure replenishing are implemented by the EVNR node. A loop is used to provide the time delay and simulate replenishing periods. This segment takes care of all tool replenishing activities for all machines in the system and works interactively with the machining segment, and the tool status updating and tool sharing segment.

6. Unloading segment (Figure 3.7f)

This segment is used to reset the number of finished processing steps if a part needs more machining work, unload the part from the machine on which it was machined, route the part back to the ASRS if further operations are required, and collect

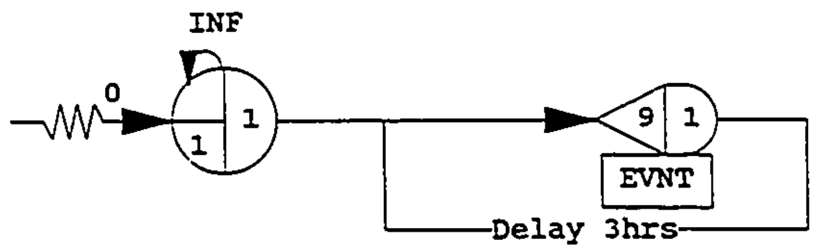


Figure 3.7e Tool replacement segment

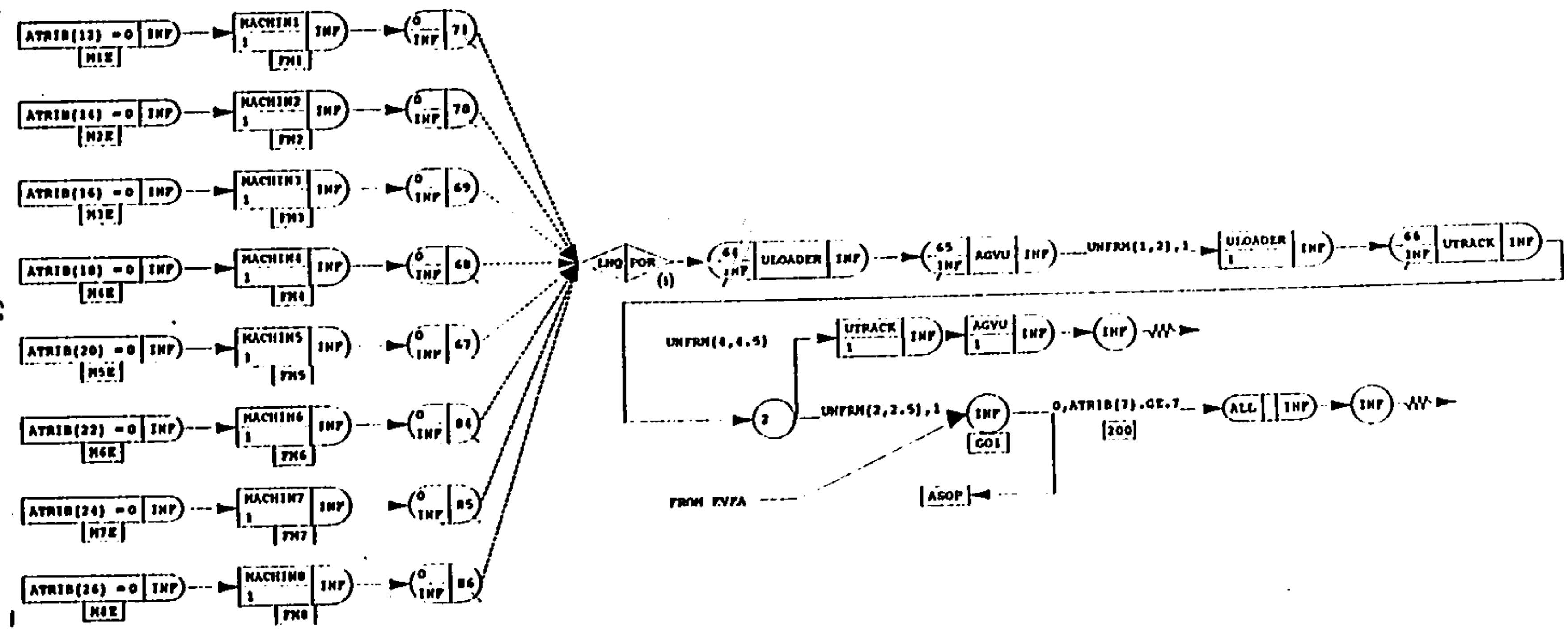


Figure 3.7f Unloading segment

all statistics for finished parts.

Once an operation (a set of processing steps) has been completed by the machining segment, the number of finished processing steps is reset to 0 by ATRIB(12) in the ASSIGN nodes, M1E, M2E, M3E, M4E, M5E, etc. Each machine has one such ASSIGN node. The part is then put to the output buffer symbolized by a QUEUE node and the machine is freed by a FREE node, e.g., FM1. If the AGVs and unloading robots are not available, the parts will wait in their own output buffers. If both the unloading robot and AGV are available and the AGV path is clear, the parts are removed from the output buffers based on a specified selection rule. This is implemented using a SELECT node. In Figure 3.7f, the LNQ selection rule is used. This rule gives the priority to the QUEUE node which has the current largest number of parts in the queue. The three AWAIT nodes and two FREE nodes following the SELECT node carry out the unloading tasks and update the status of the unloading resources (unloading robots, AGVs, and AGV paths). Once the part entity reaches a GOON node GO1, a logic check is made to determine if all the required operations have been processed. The number of operations finished so far is recorded by ATRIB(7). If the ATRIB(7) value of the part is equal to the required number of operations (in Figure 3.7f, this number is 7) the part entity is routed to a COLCT node to collect the statistics of interest. Depending on the purpose of simulation, different COLCT nodes may be used. In Figure 3.7f, the collection option is ALL which collects the information on the total time that each part entity has spent in the system. If the ATRIB(7) value of the concerned part is less than the required number of operations, the part entity is routed back to the ASOP node in the loading segment for refixturing.

3.3.3 Post-simulation Module

This module consists of a FORTRAN program for determining the number of runs, run length and calculating output intervals, and a C program for comparing different operational strategies. These programs are listed in Appendix 2. Their functions are briefly described below.

1. Calculating the number of replications, run length, and output confidence intervals

The output data from a simulation exhibits random variability when random number generators are used to produce the values of the input variables. For example, the simulations based on two different streams of random numbers will probably produce two sets of different outputs even if the other system parameters are the same. If the system performance is measured by a parameter Y (e.g., M, F, U, or H in this study), the result of a set of simulation experiments will be an estimate Y' of Y. The accuracy of estimator Y' can be measured by the variance (or standard deviation) of Y'. This variance is determined by the number of simulation runs and/or run length. More simulation runs and/or longer run length generally result in smaller variance, namely more accurate Y'. In reality, the desired accuracy level is generally pre-specified. To achieve such a accuracy level, the number of simulation replications and run length should be determined. The procedure is discussed separately for batch loading and random loading policies.

Batch loading

For this loading policy, the simulation stopping time T_E is determined by the nature of the problem, i.e., by the production batch size. The simulation will be terminated if the production batch has been completed. This type of simulation is called terminating (or transient) simulation (Banks and Carson 1984). Since the run length is determined by the production batch size, the only factor influencing the estimate accuracy is the number of simulation runs (R). According to Banks and Carson (1984), the relationship between R and the 100(1-α)% confidence interval for a performance measure Y is given by

$$Y' - \frac{t_{\alpha/2,R-1} S}{\sqrt{R}} \leq Y \leq Y' + \frac{t_{\alpha/2,R-1} S}{\sqrt{R}} \tag{3.4}$$

where

$t_{\alpha/2,R-1}$ = 100(1-α) percentage point of a t distribution with R-1 degrees of freedom

Y' = estimate of Y (it could be either M or F)

R = number of simulation replications

S = sample standard deviation.

Y' and S are respectively defined by

$$Y' = \frac{1}{R} \sum_{r=1}^{R} Y_r \tag{3.5}$$

$$S = \sqrt{\sum_{r=1}^{R} \frac{(Y_r - Y')^2}{R-1}} \tag{3.6}$$

where Y_r is the simulation output of Y from the rth replication.

For a desired accuracy ϵ the required number of runs can be determined by the following procedure.

Initially make R_0 simulation replications. R_0 is 2 or larger but at least 4 or 5 is recommended (Banks and Carson 1984). The outputs of the R_0 replications are used to calculate S_0 based equation (3.6). If

$$\frac{t_{\alpha/2, R_0-1} S_0}{\sqrt{R_0}} \leq \epsilon \tag{3.7}$$

additional replications are not needed. Otherwise, solve for R in inequality (3.7) and R is the smallest integer satisfying

$$R \geq \left(\frac{t_{\alpha/2, R_0-1} S_0}{\epsilon} \right)^2 \tag{3.8}$$

Then $R-R_0$ simulation runs are required. The R obtained based on this procedure will also be used to calculate the output confidence interval using equation (3.4).

Random loading

In this case, both the simulation run length T_E and the number of runs R should be determined in order to accurately estimate the system performance measures (U and H). The simulation interest now is to find the steady-state or the long-run properties of the system, i.e., the properties which should not be influenced by the initial conditions of the model. To reduce the bias caused by the initial system conditions, the simulation output obtained from the initial transient period should be deleted before specifying the run length.

An important issue is how to determine the transient period length (d). Unfortunately, this is still a research topic and there is no widely accepted technique to determine d. Here, the method suggested by Banks and Carson (1984) is applied. This method first examines the confidence interval of the output data by gradually increasing d and then chooses the smallest d that yields acceptable confidence interval. The details are described as follows.

First make R pilot simulation runs with a fixed run length T_E (sufficiently long). For each replication, divide T_E into N equal time slots and collect simulation data for each time slot. Let Y_{rn} (r=1,..., R; n=1,...,N) denote the observation (for a particular performance measure) of the nth time slot from replication r. If d ($1 \leq d < N$) time slots are deleted, then the associated 100(1-α)% confidence interval is given by

$$Y(d)_{..} - \frac{t_{\alpha/2, R-1} S(d)}{\sqrt{R}} \leq Y \leq Y(d)'_{..} + \frac{t_{\alpha/2, R-1} S(d)}{\sqrt{R}} \tag{3.9}$$

where

$$Y(d)_{..} = \frac{1}{R}\sum_{r=1}^{R} Y_{r.}(d) \tag{3.10}$$

$$Y_{r.}(d)_{..} = \frac{1}{R-d}\sum_{n=d+1}^{R} Y_{rn} \tag{3.11}$$

$$S = \sqrt{\frac{1}{R-1}\left(\sum_{r=1}^{R} Y^2_{r.} - RY^2_{..}\right)} \tag{3.12}$$

For the system utilization, Y is replaced by U, the calculations can be performed by changing d from 1 to a certain number d^* which satisfies

$$\frac{t_{\alpha/2,R-1} S(d^*)}{\sqrt{R}} \leq \epsilon_d \tag{3.13}$$

Where ϵ_d is the acceptable accuracy level. Then d^* time slots will be deleted and the remaining period (T_E-d^*T_E/N) will be used for data collection. For system throughput, Y is replaced by the throughput per time slot. The remaining procedure is the same as above. The number of simulation runs (R) and the output confidence intervals are determined following the same procedure as used for the batch loading policy.

All the computations for determining R, d^*, and the output confidence intervals are performed using a FORTRAN program OTPT.

2. Comparing operational strategies

The comparisons between the operational strategies outlined in section 3.2 are done using two types of statistical tests: ANOVA test and Tukey T test. A C program STATEST (Appendix 2) is developed to carry out all the tests. The ANOVA is used to test if several strategies are significantly different. The Tukey T test is then used to test which strategy is better. The details of the two types of tests are given in Appendix 3.

Chapter 4

MODEL APPLICATION

This chapter demonstrates the application of the proposed model. All the 24 operational strategies discussed in Chapter 3 will be simulated and compared. The simulation results will be analyzed using ANOVA and Tukey T tests.

4.1 Problem Description

1. The system

The automated manufacturing system to be simulated consists of three machine cells G1, G2 and G3. The number of machining centers in the three machine cells are 3,2, and 3 respectively. The machines within a machine cell are assumed to identical but the machines in different machine cells may be different. Each machining center is equipped with a tool magazine with a capacity of 40 tool slots. An automatic tool changer is also provided for each machine and, therefore, the tool change time can be neglected in the simulation study. The system has a in-line layout (Figure 4.1). As shown in Figure 4.1, each machining center has two buffers for work-in-process, one for input parts and the other for output parts. The buffer sizes are assumed to be sufficiently large but can be also specified at any different value by the user.

The part flow in the system starts at warehouse where the scheduled part is fixtured onto a pallet. After fixturing, a crane loads the part to an AGV if it is available. The AGV then moves the part to the input buffer of a machine to which the part is assigned and the part is loaded to the machine if it is idle or the input buffer of the machine if the machine is busy by a robot. After finishing all the processing steps on the machine, the part is unloaded to the output buffer and then to an AGV if available. The AGV returns the part back to the warehouse. If the required operations have not been completed, the part will be refixtured for the next operation. Then the above procedure is repeated until all the required operations have been performed. All the operational times for material handling and loading/unloading are randomly distributed. The detailed data are shown in Table 4.1.

Description		Part type									
		1	2	3	4	5	6	7	8	9	10
# of parts in a batch	Random loading	N/A									
	Batch loading	12	10	8	13	5	4	9	15	17	7
Mean inter-arrival time (exponential distribution)	Random loading	100	200	200	500	200	300	300	200	400	250
	Batch loading	N/A									

Table 4.1 The distributions of material handling and loading/unloading times

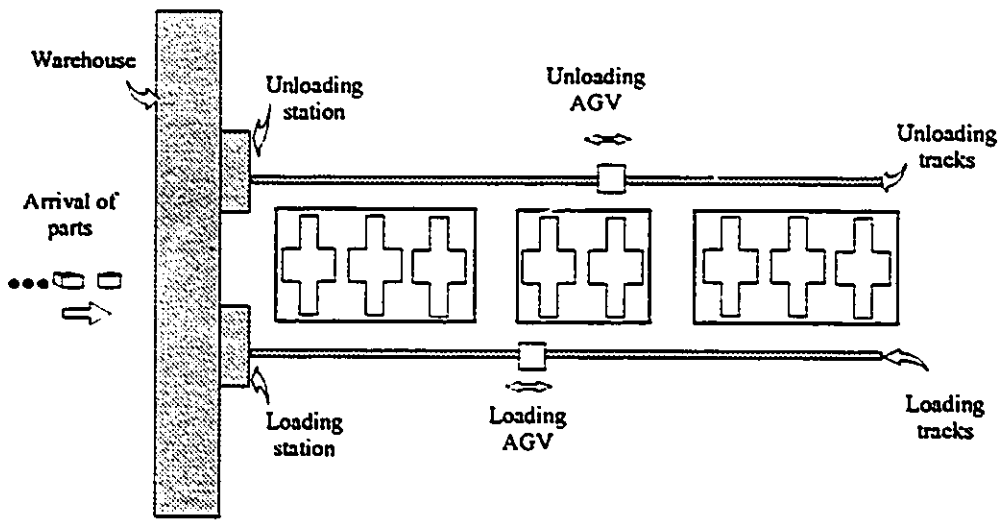


Figure 4.1 Layout of the simulated system

2. The parts

Ten different part types are to be machined by the system. Each part type needs a number of operations. Each operation further consists of several processing steps. Each processing step is defined as the machining work which can be processed using a single type of tool. However, the processing time required for the same type of processing step, e.g., a drilling step, may vary for different part types. Ideally, an operation, i.e., a set of processing steps can be processed on a single machining center if the tools on each machine are properly configured.

The input data regarding the number of operations, processing steps and the required machining time for each processing step of each part type are listed in Table 4.2.

3. The tools

In this example a total of 24 types of cutting tools, numbered from T1 to T24, are assumed. The available copies of each tool type are assumed to be sufficient to process the scheduled parts. This assumption is reasonable since otherwise the production cannot be pursued. However, if desirable, the limit on the number of copies of the each tool type can be easily accommodated by this simulation model. It is further assumed that all tool lives follow exponential distribution with a mean of 20 minutes.

Part type	1		2		3		
Operations required	7	6	14	9	10	11	12
Processing steps	1 8 5	10 16 10 9	3 2 8 6 1 8	4 6 4 2	8 2 1 2	4 8 1	4 6 4 5 4 6 7
Processing time(min)	13 1 5	4 12 7 1	2 9 9 2 7 2	6 8 9 2	8 7 7 2	3 2 3	6 8 9 2 2 3 6

Part type	4				5			
Operations required	1	6	8	2	15	1	16	4
Processing steps	7 5 1 3 8 5	10 16 10 9	3 2 8 6 1	12 15 11 14 9	5 4 6 1 7	7 5 1 3 8 5	14 13 9	15 14 10 12
Processing time(min)	1 1 6 8 6 8	4 12 5 5	2 9 9 3 7	24 3 5 4 6	4 2 3 6 6	1 1 6 8 6 8	3 1 2	6 6 4 1

Part type	6				7				8	
Operations required	2	17	5	18	23	24	25	26	23	19
Processing steps	12 15 11 14 9	13	9 14 9	2 5 6 3 1 3	19 18 24 19	20 23 24	17 19 20 21	18 19 24 23	19 18 24 19	8 6 7 8
Processing time(min)	24 3 5 4 6	3	7 3 13	4 3 5 3 2 6	5 7 2 1	6 4 7	4 7 2 3	5 4 3 2	5 7 2 1	2 6 4 1

Part type	9			10			
Operations required	20	23	22	27	13	21	3
Processing steps	8 7 8	19 18 24 19	16 9 16 13	17 22 17	23 22 18 24	24 17 23 20	21 22 17
Processing time(min)	4 5 2	5 7 2 1	1 5 5 4	7 3 13	6 6 4 5	7 4 2 5	8 7 5

Table 4.2 Machining time data

4. Operational strategies

All the 24 operational strategies will be simulated for this example problem. The details are described below.

(1) Tool configurations

For the identical configuration, all the machines in machine cell G1 are loaded with tool types T1 to T8 with five copies each type. In machine cell G2, tool types T9 to T16, five copies each, are identically loaded to the two machines in this cell. Similarly, the three machines in machine cell G3 are equipped with tool types T17 to T24, again five copies each tool type.

The tool distribution for a non-identical configuration is shown in Table 4.3.

	Tool type	1	2	3	4	5	6	7	8
	Machine 1	7*	0	5	7	7	0	7	7
G1	Machine 2	6	5	6	0	6	6	6	5
	Machine 3	5	6	6	5	6	6	6	0
	Tool type	9	10	11	12	13	14	15	16
	Machine 4	5	7	0	0	7	7	7	7
G2	Machine 5	6	5	6	6	0	6	6	5
	Tool type	17	18	19	20	21	22	23	24
	Machine 6	6	6	6	6	6	0	5	5
G3	Machine 7	6	6	7	7	7	7	0	0
	Machine 8	6	7	0	6	0	7	7	7

Table 4.3 Tool distribution of a non-identical tool configuration

Under hybrid configuration, machine cells G1 and G2 are configured in the same way as in non-identical configuration and the tool configuration of machine G3 is identical to that of the machine cell G3 in the identical configuration above.

(2) Loading policies

For batch loading, the production requirement of each part type is fixed in the current production period. The lot sizes for the 10 part types are 12,10,8,13,5,4,9,15,17, and 7 respectively. For the random loading policy, the production demand, i.e., the interarrival times of the part orders, follow exponential distribution. The mean interarrival times of the 10 part types are respectively specified as 54 min, 90 min, 102 min, 270 min, 102 min, 150 min, 120 min, 60 min, 228 min, and 90 min.

(3) Tool replenishing

Under the preventive tool replenishing policy, both the failed tools and the tools whose cumulative cutting times are greater than or equal to their expected lives are replaced with the new tools delivered from the tool room. In this example problem, the expected tool lives are 20 minutes. For the failure replenishing policy, only the failed tools are replaced.

(4) Tool sharing

Both dedicated tooling and shared tooling policies will be simulated. For the dedicated tooling policy, the tools allocated to a machining center can be used only by that machining center. If all the allocated tools are exhausted before the scheduled tool replenishing the machine will stop processing until the tools are refilled. Under the shared tooling policy, if the tools are exhausted on a particular machine, the required tools can be borrowed from a neighbour machine. However, to reduce excessive tool flow, it is specified in this problem that the tools can be shared only between the machines in the same machine cell.

4.2 Simulation

The number of replications for both batch and random loading policies were determined following the procedure suggested by Banks and Carson (1984) which is outlined in Section 3.3. The number of pilot replications R_0 value was chosen as 6 for all the 24 simulation scenarios. For each of the 24 simulation scenarios (for the 12 random loading scenarios the run length is chosen as 4400 minutes), six independent runs were made with different random number seeds. Based on the results of these simulation runs and the specified accuracy levels ($\acute{\epsilon}_u=0.01$ for system utilization, $\acute{\epsilon}_h=15$ workpieces for throughput, $\acute{\epsilon}_m=120$ min for makespan, and $\acute{\epsilon}_f=90$ min for mean flow time) the number of additional replications was calculated as between 2 and 4, depending on the performance measure used. Therefore, the number of replications R was selected as 10 for all scenarios.

For the random loading scenarios, following the procedure described in section 3.3, it was found that after truncating the data from the first 400 minutes the system performance measures (for both throughput and system utilization) reach steady-state.

Therefore, the initial data from the first 400 minutes are deleted for all the 12 random scenarios. The data are thus collected only over the remaining 4000 minutes.

To simulate the effects of loading level, i.e., the variations of part interarrival times, five different loading levels are considered for the random loading problems. It is assumed that all the interarrival times follow the exponential distribution. The mean interarrival times of the five loading levels for the ten part types are listed in Table 4.4. For both the random and batch loading problems, the tool replenishing time intervals range from 3 to 15 hours with the increment of 3 hours. Therefore, a total of 1800 (12x10x5=600 runs for batch loading and 12x10x5+12x10x5=1200 runs for random loading) simulation runs have been carried out. The simulations are performed on a 486 PC using the SLAMSYSTEM, TURBO C++, and Microsoft FORTRAN packages. The average computational times are about 3.5 minutes and 6 minutes for batch and random loading policies respectively.

	Loading level				
Part type	1	2	3	4	5
1	90	72	54	36	18
2	150	120	90	60	30
3	170	136	102	68	34
4	450	360	270	180	90
5	170	136	102	68	34
6	250	200	150	100	50
7	200	160	120	80	40
8	100	80	60	40	20
9	380	304	228	152	76
10	150	120	90	60	30

Table 4.4 Mean interarrival times (min) of the five loading levels

4.3 Results and Discussions

The simulation results have been summarized in Appendix 4, some typical samples of the simulation results are also included in Appendix 4. All the results are tested using a 2-step procedure. The purpose of the first step is to test whether the concerned strategies are significantly different. This is done using ANOVA. If the ANOVA test shows a significant difference, then the Tukey T test will be used to rank the strategies in terms of a particular performance measure. The simulation results and discussions are presented below.

4.3.1 Effects of Tool Configuration

The effects on mean flow time

As mentioned in Chapter 3, this performance measure is used only for batch loading policy. To examine the effects of tool configuration on mean flow time, simulations are carried for different tool replenishing and tool sharing policies with five different replenishing intervals. The related simulation results are plotted in Figure 4.2. The following observations can be made based on Figure 4.2:

(1) The mean flow time increases as the replenishing time interval increases for all tool replenishing and sharing policies. This is mainly caused by the tool shortage on the machines. The longer the replenishing interval is, the more serious the tool shortage will be, thereby rendering higher mean flow time.

(2) It appears that identical and hybrid tool configurations are superior to the non-identical configuration. It is also noticed that, for the combinations of preventive

replenishing + shared tooling, and failure replenishing + shared tooling, the performance of the non-identical configuration is similar to that of the identical and hybrid configurations if the replenishing interval is greater than or equal to nine hours. The statistical comparisons are listed in Table 4.5(a). As shown in Table 4.5(a), with a 90% confidence level, the identical configuration is ranked as the best for two combinations: preventive replenishing + shared tooling, and failure replenishing + dedicated tooling. The hybrid configuration has the best performance when failure replenishing and shared tooling are employed and ties the identical configuration for failure replenishing and dedicated tooling. Except the combination of preventive replenishing + shared tooling, for which the performance of three configurations have no significant difference, the non-identical tooling policy has the worst performance. However, this observation may not be generalized in that many different non-identical configurations can be suggested. The one simulated above is only one of them. This further reveals the difficulty in designing a good GT system (a special case of the non-identical configuration).

The effects on makespan

The related simulation outputs are summarized in Figure 4.3 and are tested in Table 4.5(a). It is noticed that the effects of tool configuration on the makespan is quite similar to the effects on the mean flow time. However, the non-identical configuration in this case shows a more apparent inferiority. This is also demonstrated by the statistical tests as shown in Table 4.5(a).

The effects on system utilization

System utilization is used as a performance criterion only for the random loading policy. The effects of tool configuration on system utilization are shown in Figures 4.4 and 4.5. Figure 4.4 shows such effects under different tool replenishing intervals (for loading level 3) and Figure 4.5 displays the effects under different loading levels (failure replenishing, 9-hour interval). By examining Figures 4.4, 4.5, and Table 4.5, it is observed:

(1) Tool configuration pattern has a strong impact on the system utilization. Figure 4.4 clearly shows such an effect. The system utilization can be as high as 90% if a 3-hour interval is applied (Figure 4.4 (b)) and as low as 37% with a 15-hour interval (Figure 4.4 (d)).

(2) The hybrid configuration is the best of the three tool configuration policies and the non-identical configuration has the poorest performance. This is observed by summarizing the statistical test results and the curves in Figure 4.4.

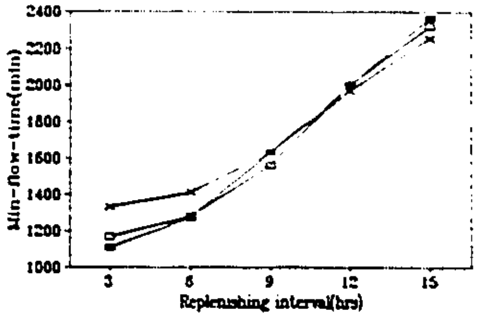
(3) Under different loading levels, there is no significant difference among the three tool configuration policies if tool sharing is allowed. However, if the loading level is increased to 5, i.e., the system is more heavily loaded, the tool configuration does show some effects. Such effects may be examined by additional simulations under higher loading levels. If tool sharing is not allowed, identical tooling shows better performance

in terms of system utilization. Significant difference is not found between non-identical and hybrid tool configurations (Table 4.5(b)).

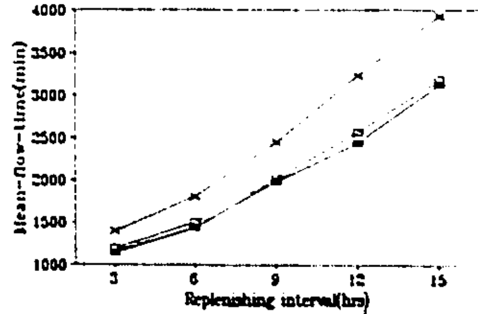
(4) System utilization level appears to have a saturation point above which the system utilization cannot be further increased by increasing the loading level. This is observed in Figure 4.5 (b). Figure 4.5 (b) clearly shows that the system utilization is stabilized when the loading level is above level 4. However, the saturation point is not seen in Figure 4.5 (a). An explanation is that the tool sharing has delayed the saturation point. If the loading level is further increased a similar saturation point can be expected. This indicates that tool sharing can help increase the maximum system utilization level.

The effects on throughput

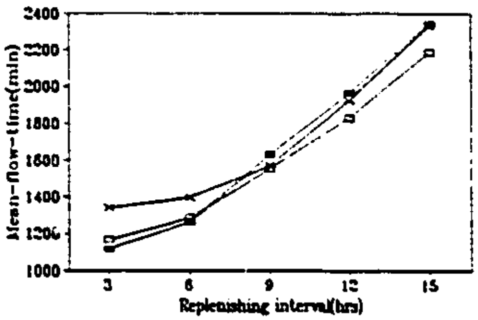
The statistical tests (Table 4.5(a)) and Figure 4.6 show that the effects of tool configuration are similar to its influences on system utilization. This is anticipated in that the system utilization and throughput are two similar criteria. If the system can be more efficiently utilized it is logical to expect more output from the system. The effects of tool configuration on throughput under various loading levels are displayed in (Figure 4.7) and tested in Table 4.5(b). Again, no significant difference was found among the three tool configurations under shared tooling. For the dedicated tooling, the identical tool configuration has the best performance. It is also noticed that the throughput has a slight drop as the loading level increases from level 4 to level 5. This is probably caused by the congestion of the shop floor due to the excessive work-in-process inventory.



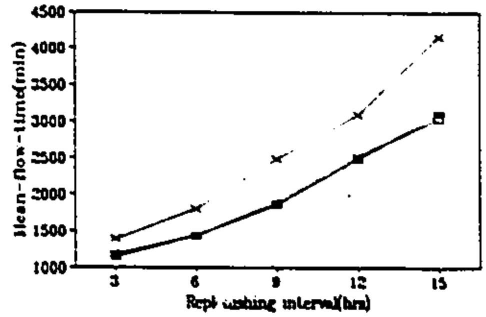
(a) Preventive replenishing, shared tooling



(b) Preventive replenishing, dedicated tooling



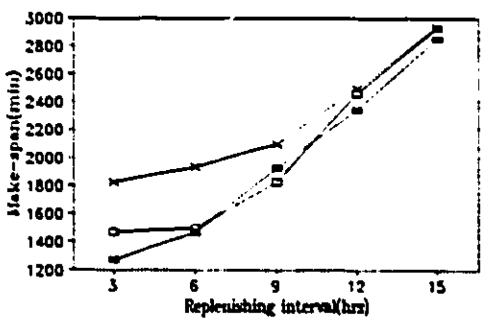
(c) Failure replenishing, shared tooling



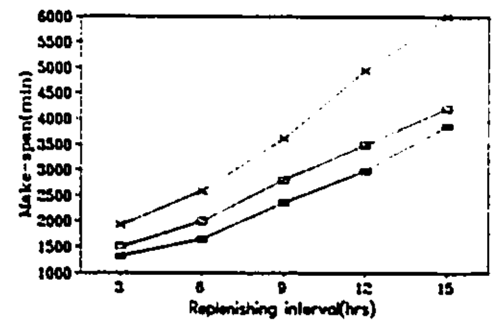
(d) Failure replenishing, dedicated tooling

(■ Identical × Non-identical □ Hybrid)

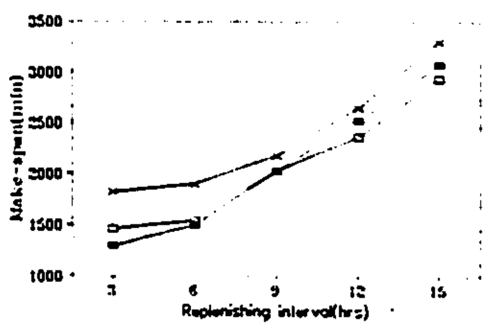
Figure 4.2 Effects of tool configuration on mean-flow-time(batch loading)



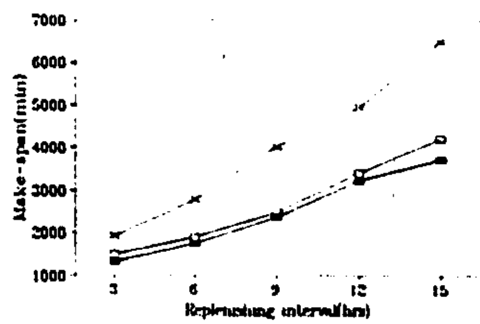
(a) Preventive replenishing, shared tooling



(b) Preventive replenishing, dedicated tooling



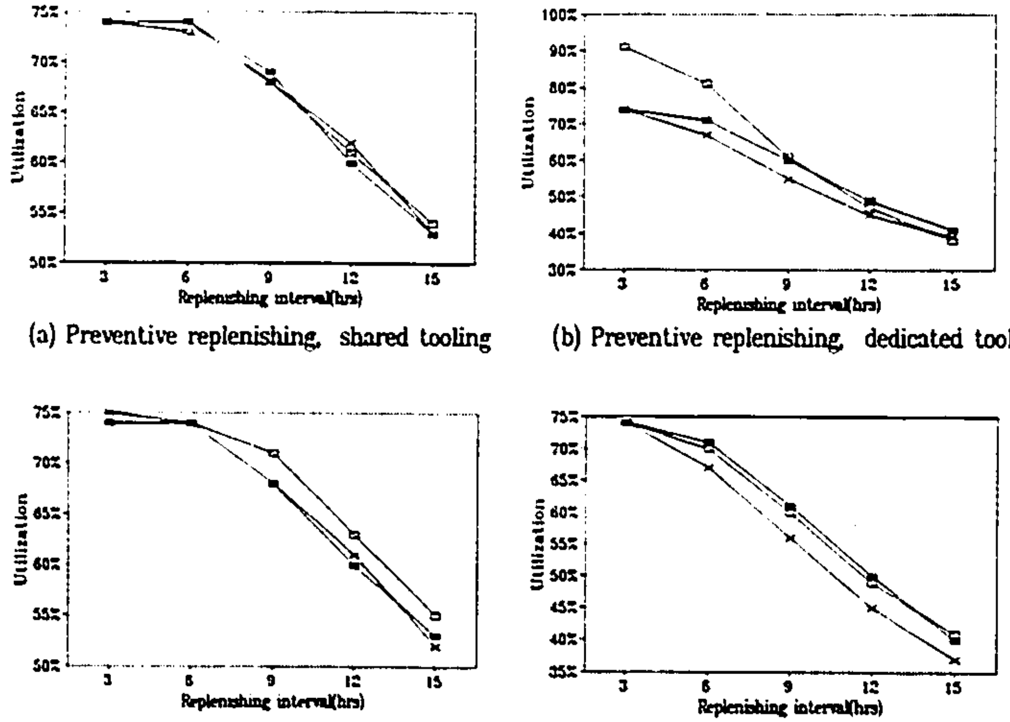
(c) Failure replenishing, shared tooling



(d) Failure replenishing, dedicated tooling

(■ Identical × Non-identical □ Hybrid)

Figure 4.3 Effects of tool configuration on make-span(batch loading)



(a) Preventive replenishing, shared tooling

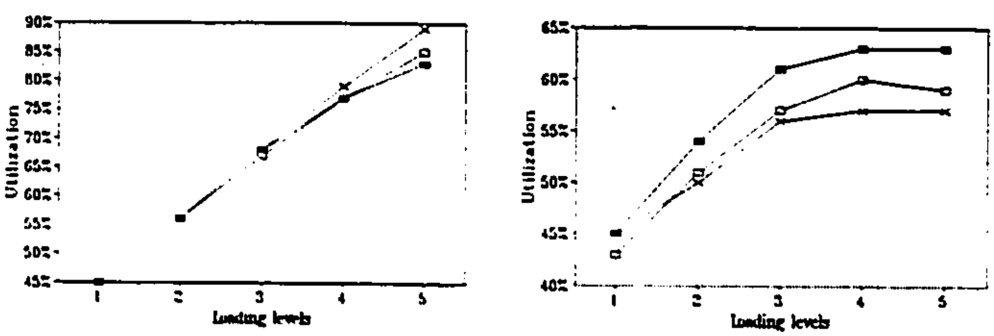
(b) Preventive replenishing, dedicated tooling

(c) Failure replenishing, shared tooling

(d) Failure replenishing, dedicated tooling

(■ Identical × Non-identical □ Hybrid)

Figure 4.4 Effects of tool configuration on system utilization under different tool replenishing intervals(random loading)

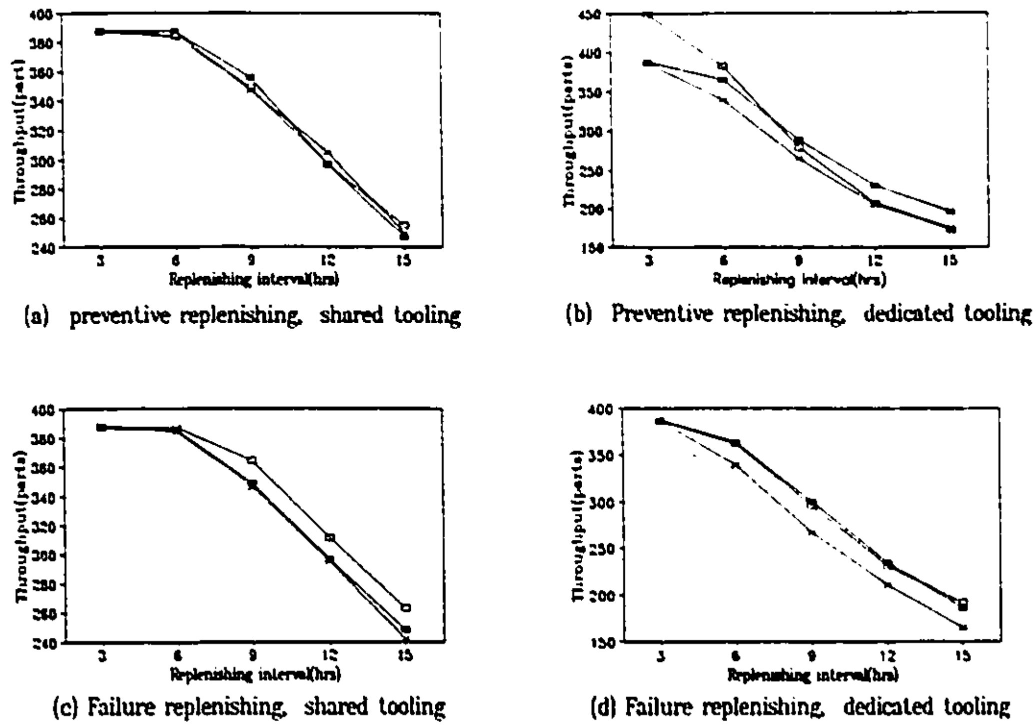


(a) Shared tooling

(b) Dedicated tooling

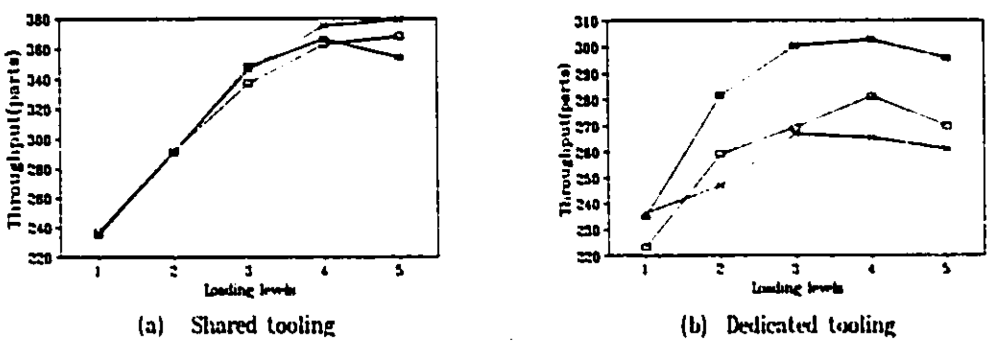
(■ Identical × Non-identical □ Hybrid)

Figure 4.5 Effects of tool configuration on system utilization under different loading levels(9-hr replenishing interval, failure replenishing, random loading)



(■ Identical × Non-identical □ Hybrid)

Figure 4.6 Effects of tool configuration on system throughput, under different tool replenishing intervals(random loading)



(■ Identical × Non-identical □ Hybrid)

Figure 4.7 Effects of tool configuration on system throughput, under different loading levels (9-hrs replenishing interval, failure replanishing, random loading)

Performance measure	Replenishing policy	Tool sharing policy	First test			Second test			Rank		
			$\mu_I \neq \mu_N$	$\mu_I \neq \mu_H$	$\mu_H \neq \mu_N$	$\mu_I^B < \mu_N^B$ $\mu_I^R > \mu_N^R$	$\mu_I^B < \mu_H^B$ $\mu_I^R > \mu_H^R$	$\mu_H^B < \mu_N^B$ $\mu_H^R > \mu_N^R$	I	N	H
Mean-flow-time (batch)	Preventive	Shared	No	No	No	–	–	–	–	–	–
		Dedicated	Yes	Yes	Yes	Yes	Yes	Yes	1	3	2
	Failure	Shared	No	No	Yes	–	–	Yes	–	2	1
		Dedicated	Yes	No	Yes	Yes	–	Yes	1	2	1
Make-span (batch)	Preventive	Shared	Yes	No	Yes	Yes	–	Yes	1	2	1
		Dedicated	Yes	Yes	Yes	Yes	Yes	Yes	1	3	2
	Failure	Shared	Yes	No	Yes	Yes	–	Yes	1	2	1
		Dedicated	Yes	Yes	Yes	Yes	Yes	Yes	1	3	2
Utilization (random)	Preventive	Shared	No	No	No	–	–	–	–	–	–
		Dedicated	Yes	No	Yes	Yes	–	Yes	1	2	1
	Failure	Shared	No	Yes	No	–	No	–	2	–	1
		Dedicated	Yes	No	Yes	Yes	–	Yes	1	2	1
Throughput (random)	Preventive	Shared	No	No	No	–	–	–	–	–	–
		Dedicated	Yes	No	No	Yes	–	–	1	2	–
	Failure	Shared	No	Yes	Yes	–	No	Yes	2	2	1
		Dedicated	Yes	No	Yes	Yes	–	Yes	1	2	1

(I = Identical, N = Non-identical, H = Hybrid, B = Batch, R = Random, Yes = Passed, No = Failed, – = Can not be determined based on this test)

Table 4.5(a) Statistical analysis results of tool configuration effects under different tool replenishing policies (90% confidence level)

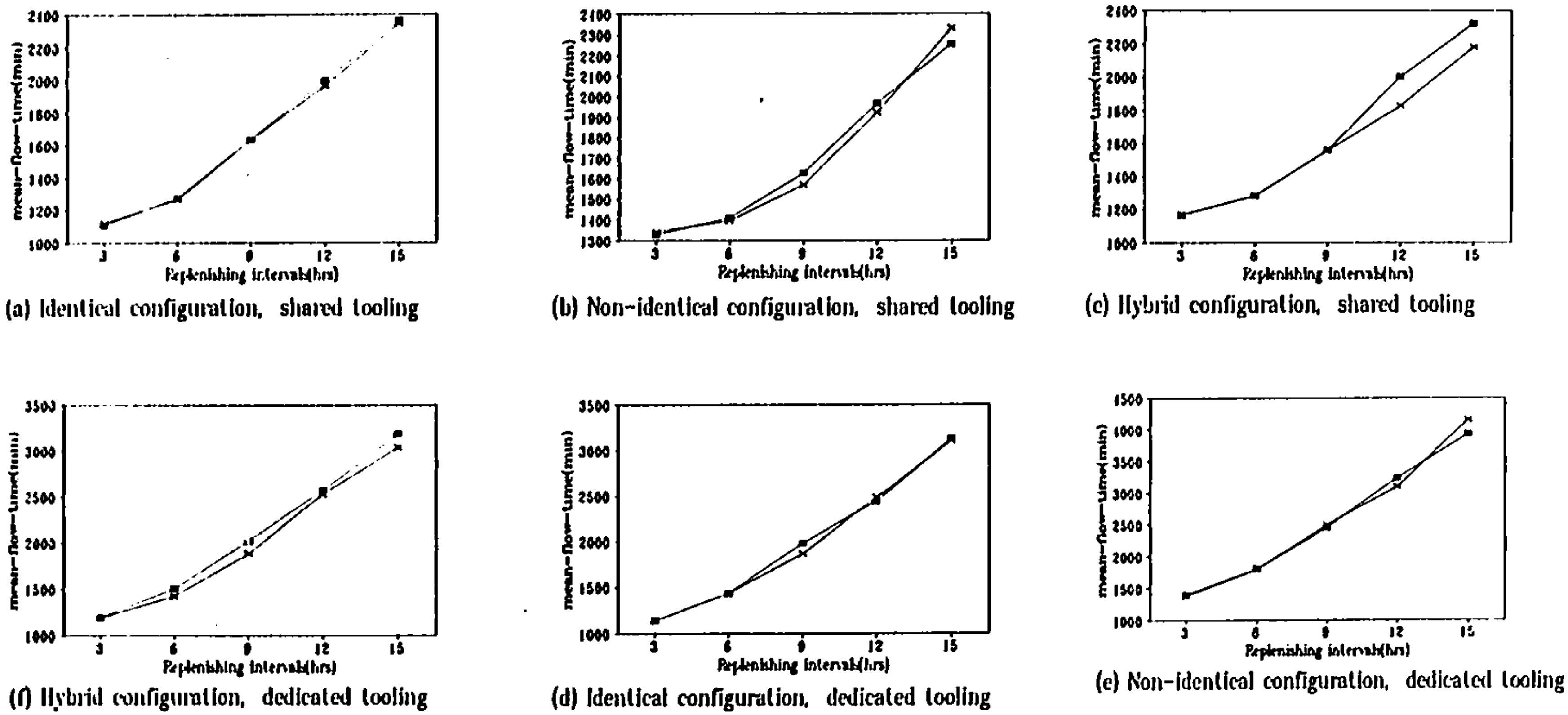
Performance measure	Tool sharing policy	First test			Second test			Rank		
		$\mu_I \neq \mu_N$	$\mu_I \neq \mu_H$	$\mu_H \neq \mu_N$	$\mu_I^B < \mu_N^B$ $\mu_I^R > \mu_N^R$	$\mu_I^B < \mu_H^B$ $\mu_I^R > \mu_H^R$	$\mu_H^B < \mu_N^B$ $\mu_H^R > \mu_N^R$	I	N	H
Utilization(random)	Shared	No	No	No	–	–	–	–	–	–
	Dedicated	Yes	Yes	No	Yes	Yes	–	1	2	2
Throughput(random)	Shared	No	No	No	–	–	–	–	–	–
	Dedicated	Yes	Yes	No	Yes	Yes	–	1	2	2

(I = Identical, N = Non-identical, H = Hybrid, B = Batch, R = Random, Yes = Passed, No = Failed, – = Can not be determined based on this test)

Table 4.5(b) Statistical analysis results of tool configuration effects under different loading levels and failure replenishing policy(90% confidence level)

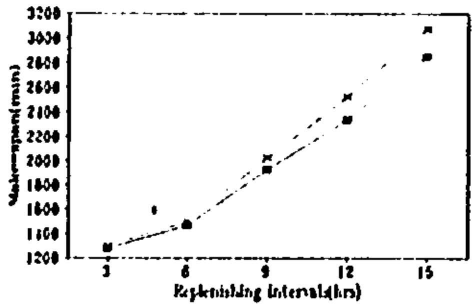
4.3.2 Effects of Tool Replenishing

The simulation results are summarized in Figures 4.8 to 4.11. The statistical test results are shown in Table 4.6. The majority of the results do not show significant difference between failure and preventive replenishing policies in terms of the four system performance measures. This is probably because there is a very limited number of over aged tools upon each scheduled replenishment. It is also noticed that preventive replenishing has lower makespan when policy combinations: identical configuration + shared tooling, and non-identical configuration + dedicated tooling are used. Failure replenishing has the lower mean flow time with hybrid configuration + dedicated tooling, and higher system utilization and higher throughput with hybrid configuration and shared tooling. However, based on the current simulation results it is difficult to conclude which replenishing policy is superior.

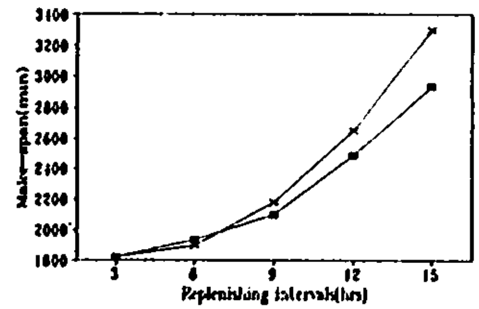


(■ Preventive × Failure)

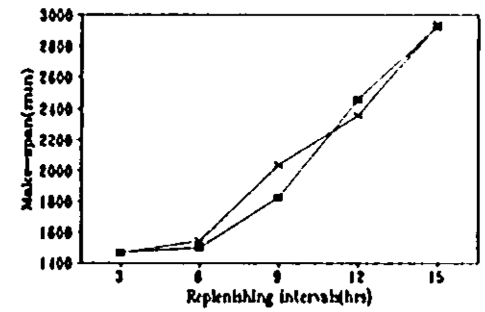
Figure 4.8 Effects of tool replenishing policy on mean-flow-time (batch loading)



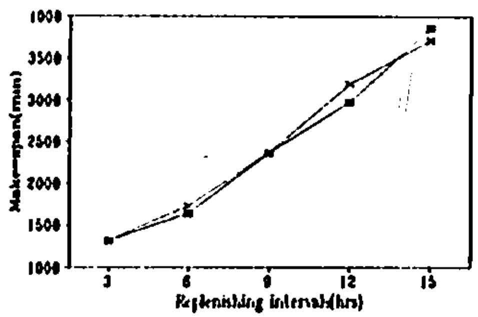
(a) Identical configuration, shared tooling



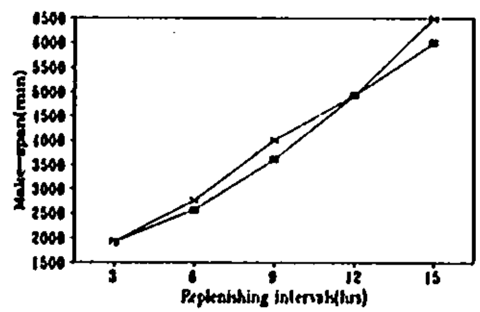
(b) Non-identical configuration, shared tooling



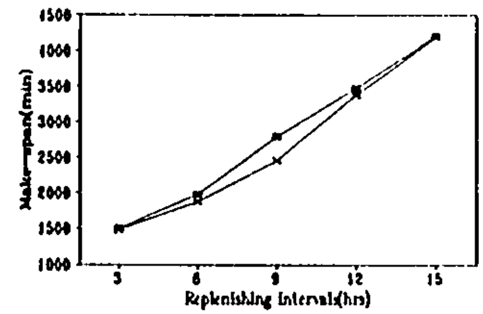
(c) Hybrid configuration, shared tooling



(d) Identical configuration, dedicated tooling



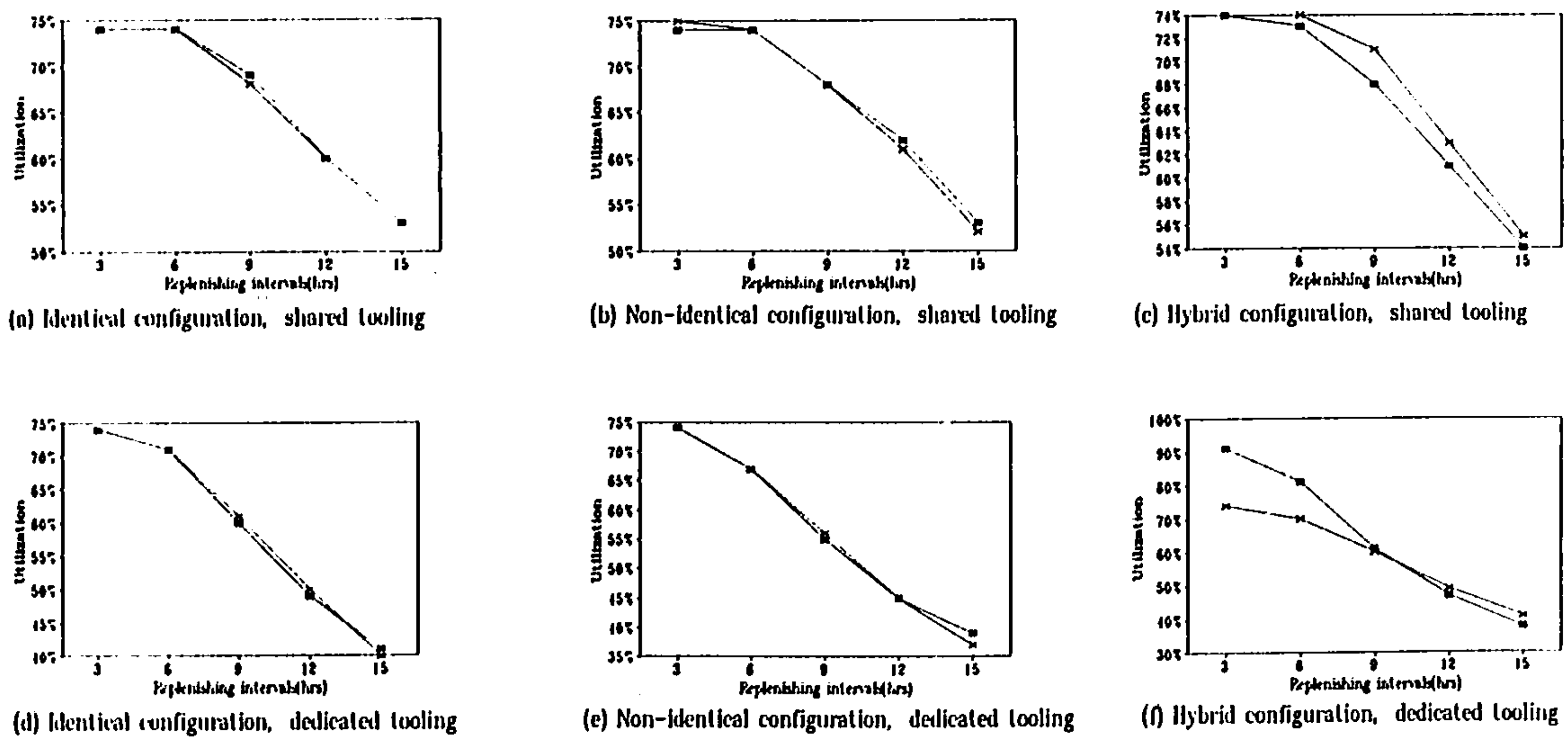
(e) Non-identical configuration, dedicated tooling



(f) hybrid configuration, dedicated tooling

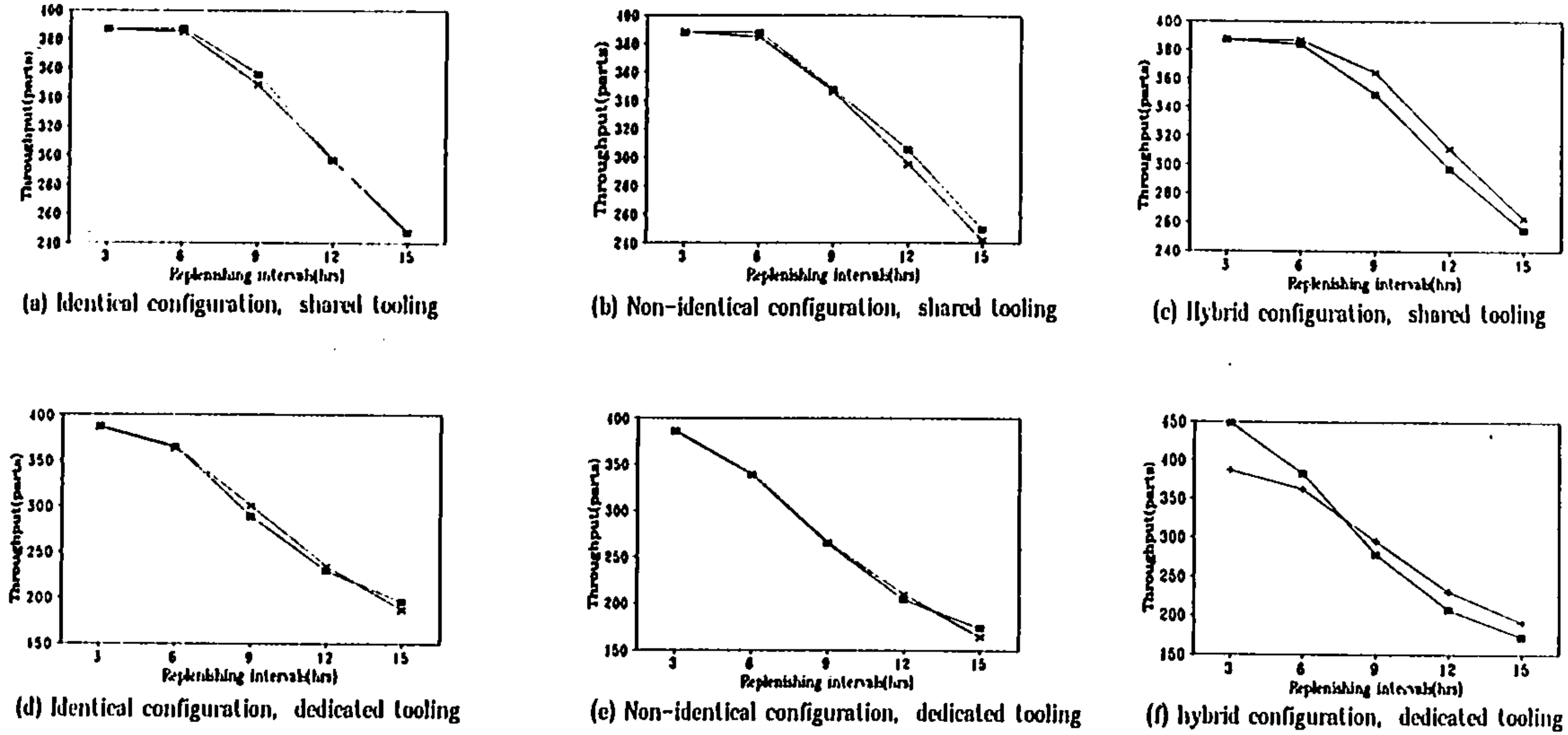
(■ Preventive × Failure)

Figure 4.9 Effects of tool replenishing policy on makespan(batch loading)



(■ Preventive × Failure)

Figure 4.10 Effects of tool replenishing policy on system utilization under different replenishing intervals(random loading)



(a) Identical configuration, shared tooling

(b) Non-identical configuration, shared tooling

(c) Hybrid configuration, shared tooling

(d) Identical configuration, dedicated tooling

(e) Non-identical configuration, dedicated tooling

(f) hybrid configuration, dedicated tooling

(■ Preventive × Failure)

Figure 4.11 Effects of tool replenishing policy on system throughput, under different replenishing intervals(random loading)

Performance measures	Tool configuration	Tool sharing policy	First test $\mu_P \neq \mu_F$	Second test $\mu_P^R > \mu_F^R$, $\mu_P^B < \mu_F^B$	Rank	
					Preventive	Failure
Mean-flow-time (batch)	Identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Non-identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Hybrid	Shared	No	–	–	–
		Dedicated	Yes	No	2	1
Make-span (batch)	Identical	Shared	Yes	Yes	1	2
		Dedicated	No	–	–	–
	Non-identical	Shared	No	–	–	–
		Dedicated	Yes	Yes	1	2
	Hybrid	Shared	No	–	–	–
		Dedicated	No	–	–	–
Utilization (Random)	Identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Non-identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Hybrid	Shared	Yes	No	2	1
		Dedicated	No	–	–	–
Throughput (Random)	Identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Non-identical	Shared	No	–	–	–
		Dedicated	No	–	–	–
	Hybrid	Shared	Yes	No	2	1
		Dedicated	No	–	–	–

(P = Preventive F = Failure yes = Passed No = Failed – = Can not be determined based on this test)

Table 4.6 Statistical analysis results of tool replenishing(90% confidence level)

4.3.3 Effects of Tool Sharing

The effects on mean flow time and makespan

The 12 graphs in Figures 4.12 and 4.13 show the related simulation results. All the graphs demonstrate a significant tool sharing effect. The significance has also been statistically tested (Table 4.7 (a)). Further it is seen that, with the shared tooling, the mean flow time and makespan have a very small increase when the tool replenishing interval increases from three hours to six hours. This implies that if the production batch size or order size is relatively small (in a more dynamic production environment), tool replenishing may not be necessary if tool sharing is allowed. In reality this means less tool inventory and less investment in the tool replenishing (equipment and labour costs).

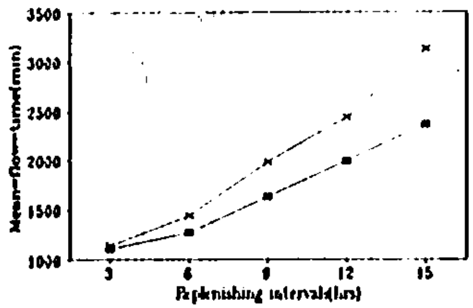
The effects on system utilization and throughput

The simulation results are plotted in Figures 4.14 and 4.15. The associated statistical analysis results are summarized in Table 4.7 (a). Again the shared tooling shows an overall superior behaviour. It is also noticed that the system utilization and throughput hardly change as the tool replenishing interval increases from 3 hours to 6 hours under the shared tooling policy. Therefore, the same implication as perceived above can be obtained: if tool sharing is adopted and production batch size is relatively small, it may be more preferred not to replenish tools.

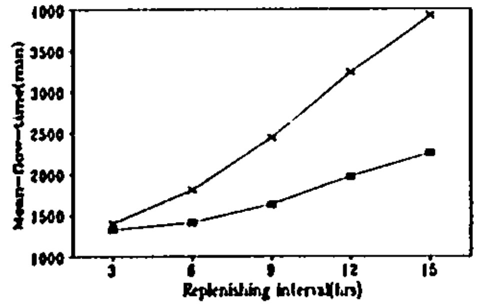
The effects on system utilization and throughput under different loading levels

Figures 4.16 and 4.17 show the simulation results (failure replenishing, 9-hour replenishing interval). The statistical analysis is given in Table 4.7 (b). As can be seen from Figures 4.16 and 4.17, and Table 4.7 (b), except the first loading level, shared tooling policy demonstrates a superior performance compared to the dedicated tooling policy for all the other loading levels.

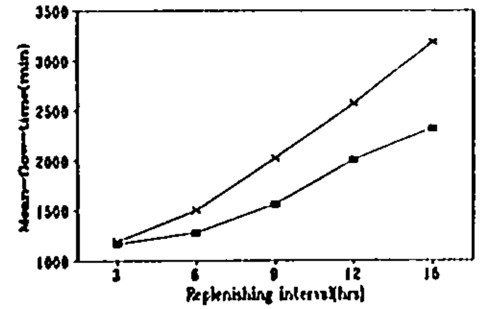
Similar to the finding in section 4.3.1, a saturation point is observed under the dedicated tooling policy. This implies that the system utilization and throughput cannot be further improved by increasing the loading level. The maximum system utilization rate under identical, non-identical, and hybrid configurations are respectively 0.63, 0.57, and 0.60, as compared to 0.83, 0.89, and 0.85 under shared tooling. In Figure 4.17, a dropping point is observed for all the three different tool configurations under dedicated tooling policy. This means that even if the system utilization is kept at the maximum level 0.63 and 0.57 for the identical and non-identical configurations (Figure 4.16 (a) and (b)), the system throughput may not maintain the maximum level. This phenomenon is more apparent for the identical configuration and dedicated tooling combination. As shown in Figure 4.16 (a), when the system utilization reaches the maximum (83%) from 77% by increasing the loading level from 4 to 5, the system throughput drops to 354 from the maximum value (366) (Figure 4.17 (a)). Possible causes of this phenomenon are a) the material handling and loading/unloading facilities are unable to move the finished parts out of the system; and b) due to the shop floor congestion, the material handling and loading/unloading facilities cannot effectively dispatch the operations so that a part can be completed at the earliest possible time. As a result, more parts are *partially* processed but fewer parts are *completely* processed. The consequence is the excessive work-in-process inventory and more congestion. This indicates that it may not a good idea to simply increase the system utilization (in this case, it means the utilization of the machining centers) without considering throughput.



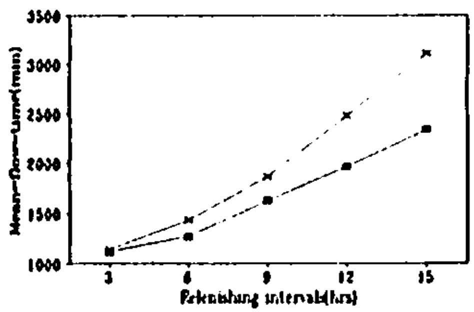
(a) Identical configuration, preventive replenishing



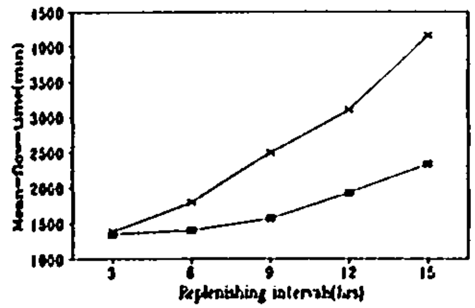
(b) Non-identical configuration, preventive replenishing



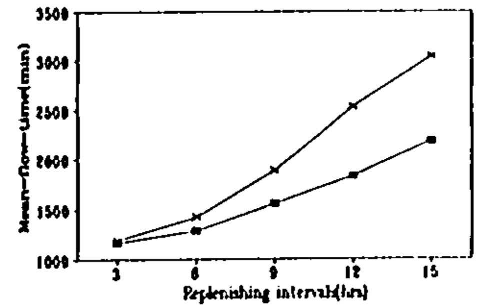
(c) Hybrid configuration, preventive failure



(d) Identical configuration, failure replenishing



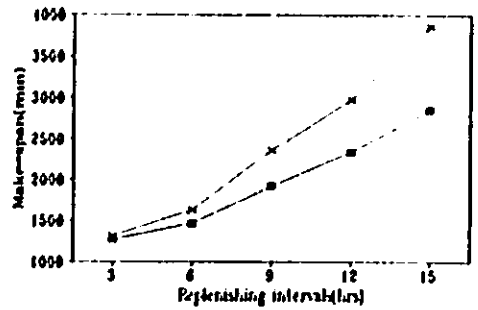
(e) Non-identical configuration, failure replenishing



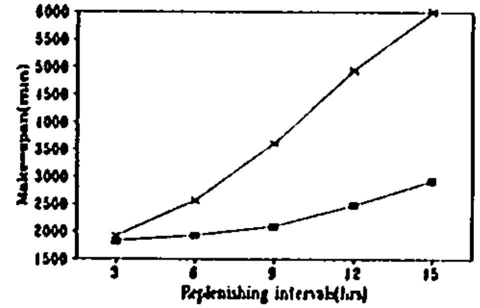
(f) Hybrid configuration, failure replenishing

(■ Shared × Dedicated)

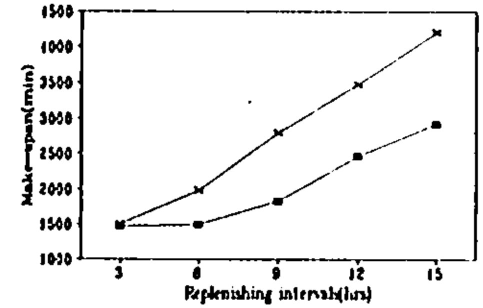
Figure 4.12 Effects of tool sharing on mean-flow-time (batch loading)



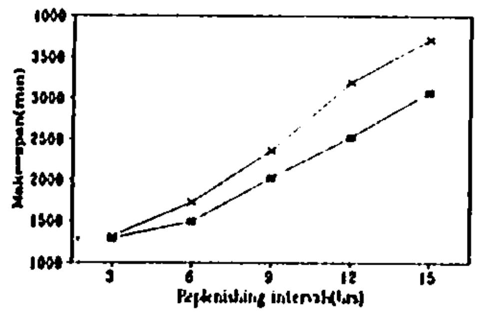
(a) Identical configuration, preventive replenishing



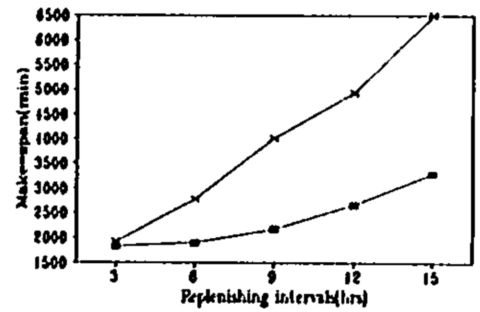
(b) Non-identical configuration, preventive replenishing



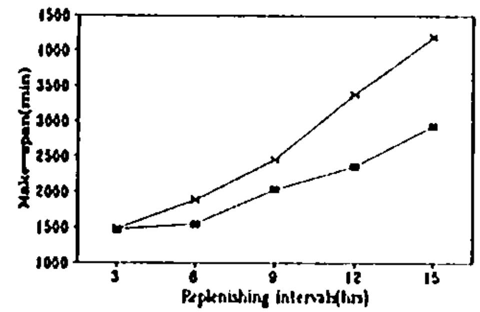
(c) Hybrid configuration, preventive replenishing



(d) Identical configuration, failure replenishing



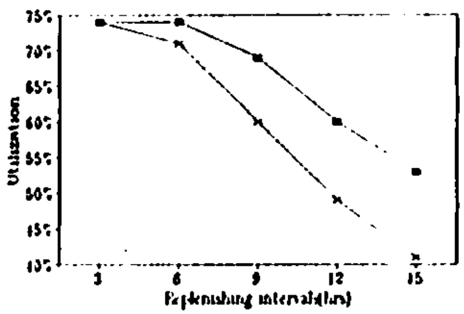
(e) Non-identical configuration, failure replenishing



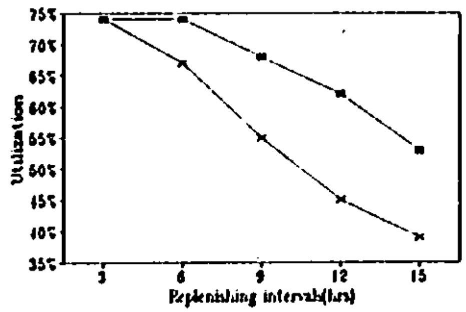
(f) Hybrid configuration, failure replenishing

(■ Shared × Dedicated)

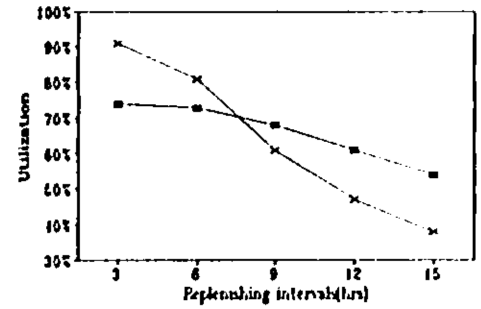
Figure 4.13 Effects of tool sharing on makespan(batch loading)



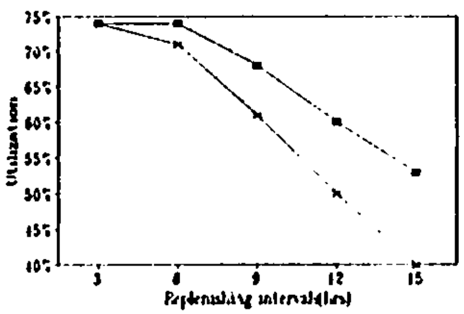
(a) Identical configuration, preventive replenishing



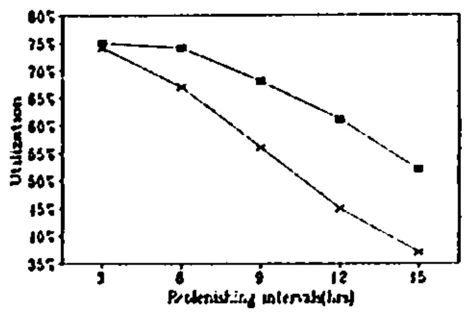
(b) Non-identical configuration, preventive replenishing



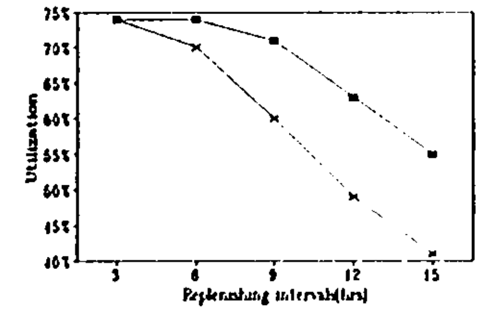
(c) Hybrid configuration, preventive replenishing



(d) Identical configuration, failure replenishing



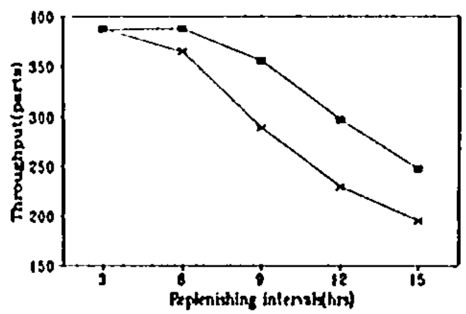
(e) Non-identical configuration, failure replenishing



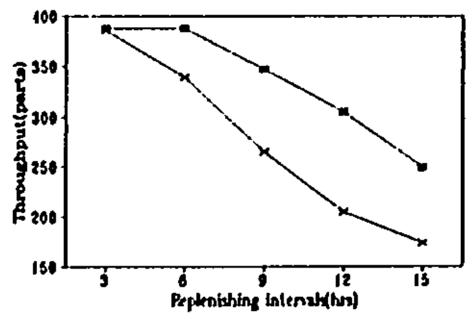
(f) Hybrid configuration, failure replenishing

(■ Shared × Dedicated)

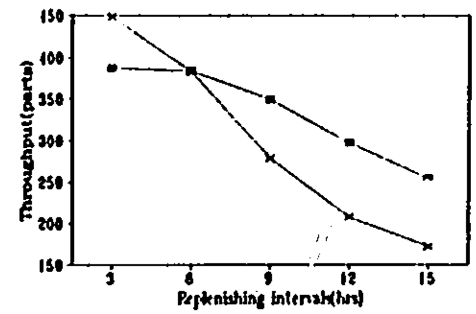
Figure 4.14 Effects of tool sharing on system utilization, under different replenishing intervals(random loading)



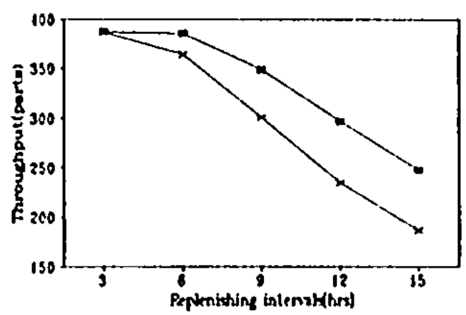
(a) Identical configuration, preventive replenishing



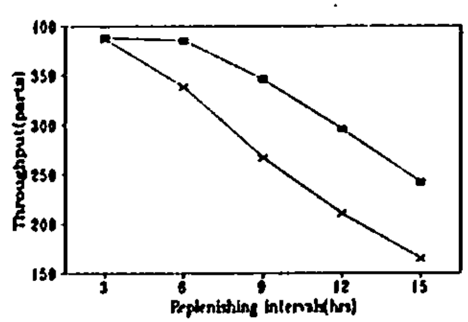
(b) Non-identical configuration, preventive replenishing



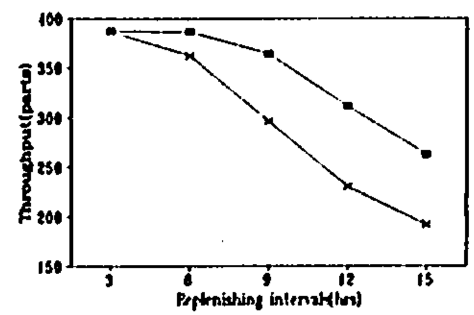
(c) Hybrid configuration, preventive replenishing



(d) Identical configuration, failure replenishing



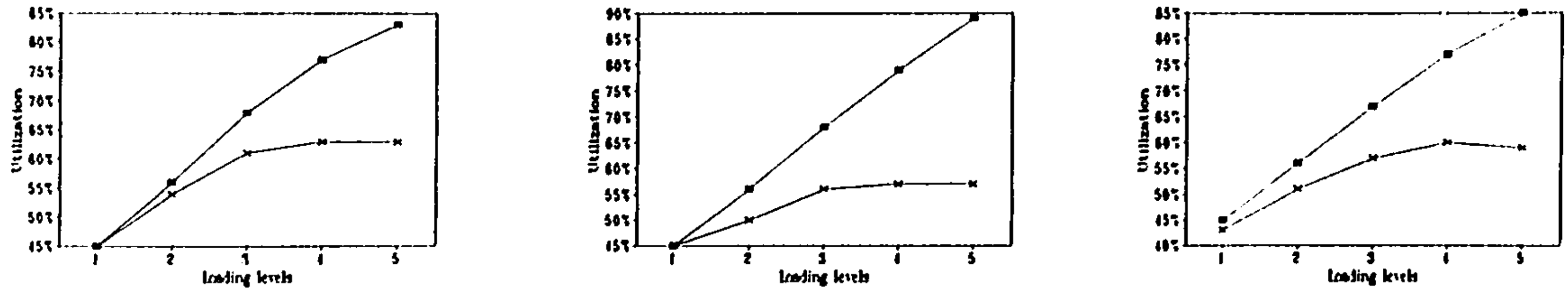
(e) Non-identical configuration, failure replenishing



(f) Hybrid configuration, failure replenishing

(■ Shared × Dedicated)

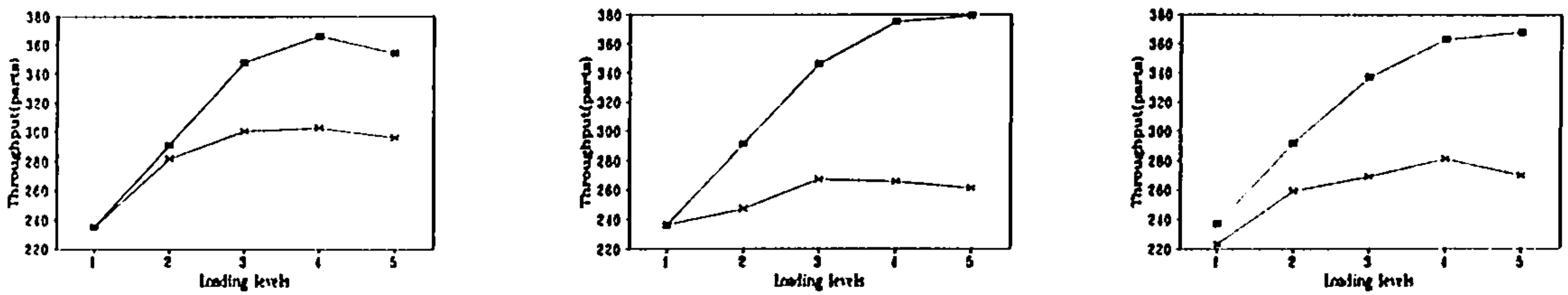
Figure 4.15 Effects of tool sharing on system throughput under different replenishing intervals(random loading)



(a) Identical configuration (b) Non-identical configuration (c) Hybrid configuration

(■ Shared × Dedicated)

Figure 4.16 Effects of tool sharing on system utilization under different loading levels(random loading, failure replenishing, 9-hr replenishing interval)



(a) Identical configuration (b) Non-identical configuration (c) Hybrid configuration

(■ Shared × Dedicated)

Figure 4.17 Effects of tool sharing on system throughput under different loading levels(random loading, failure replenishing, 9-hr replenishing interval)

Performance measure	Tool configuration	Replenishing policy	First test $\mu_S \neq \mu_D$	Second test $\mu_S^R > \mu_D^R$, $\mu_S^B < \mu_D^B$	Rank	
					Shared	Dedicated
Mean-flow-time (batch)	Identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Non-identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Hybrid	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
Make-span (batch)	Identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Non-identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Hybrid	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
Utilization (random)	Identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Non-identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Hybrid	Preventive	No	–	–	–
		Failure	Yes	Yes	1	2
Throughput (random)	Identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Non-identical	Preventive	Yes	Yes	1	2
		Failure	Yes	Yes	1	2
	Hybrid	Preventive	No	–	–	–
		Failure	Yes	Yes	1	2

(S = Shared tooling D = Dedicated tooling Yes = Passed No = Failed – = Can not be determined based on this test)

Table 4.7(a) Statistical analysis results of tool sharing(90% confidence level) under different tool replenishing intervals

Performance measure	Tool configuration	First test $\mu_S \neq \mu_D$	Second test $\mu_S^R > \mu_D^R$, $\mu_S^B < \mu_D^B$	Rank	
				Shared	Dedicated
Utilization (random)	Identical	Yes	Yes	1	2
	Non-identical	Yes	Yes	1	2
	Hybrid	Yes	Yes	1	2
Throughput (random)	Identical	Yes	Yes	1	2
	Non-identical	Yes	Yes	1	2
	Hybrid	Yes	Yes	1	2

(S = Shared tooling D = Dedicated tooling Yes = Passed No = Failed – = Can not be determined based on this test)

Table 4.7(b) Statistical analysis of tool sharing(90% confidence level) under different loading levels and failure replenishing policy

Chapter 5

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

5.1 Conclusions

Tooling problem has been recognized as a critical issue for the success of a manufacturing company. Important but not yet adequately addressed tooling issues include tool configuration, tool replenishing, and tool sharing. Extensive survey indicates that a collective study of the effects of these issues appears not available in the literature. To provide the user with a decision tool and enrich the literature in this area, the following have been accomplished:

1. A simulation model which is capable of simulate the various operational strategies defined by tool configurations, tool replenishing policies, tool sharing policies,

and loading patterns has been developed using SLAM II and FORTRAN. The post-simulation analysis module is developed using C. Its efficiency (in terms of computational time) and flexibility in handling a large number of different simulation scenarios and large problems have been demonstrated in an application example. Further, with minor modifications, the model can be used to examine the effects of different sequencing rules, tool spare constraint, and buffer capacity.

2. The main findings obtained within the simulation scope of this thesis include:

(1) Shared tooling policy shows a significant superior performance as compared with the dedicated tooling policy.

(2) For the random loading policy, system utilization and throughput can be improved by increasing loading level. However, there exists a saturation point above which the system utilization or throughput cannot be improved by increasing loading level. However, this saturation point can be significantly delayed by using the shared tooling policy.

(3) Higher system utilization does not necessarily imply a higher system throughput.

(4) Under the shared tooling policy, it may be unnecessary to replenish tools if the production batch size is relatively small.

(5) In general the identical tool configuration is the best and the non-identical configuration is the worst of the three simulated configurations.

(6) There is no significant difference between the failure and preventive tool

replenishing policies. This is probably because there are very limited over aged tools upon each replenishment which are not sufficient to show the difference between the two policies.

5.2 Future Research Directions

This simulation model has been used to evaluate the difference between different operational strategies in terms of mean flow time, makespan, system utilization and throughput. However, the cost trade-off has not been considered. For instance, the shared tooling policy may need more labour and equipment costs though it outperformed the dedicated tooling in terms of the four criteria used in this study. Therefore, the model could be extended to incorporate the cost considerations.

A comparative study could be carried out to investigate the joint effects of sequencing rules, tooling policies and loading patterns. This could be easily implemented by slightly modifying the developed simulation model.

The proposed model may be further used to determine a good GT system by dynamically re-configuring the tools in the system whenever the production demand changes. However, the required computational time could be unreasonably high.

It could be also interesting to develop an intelligent simulation framework which includes the proposed simulation model and a neural network model. Since all simulation models are memoryless, they cannot memorize or learn from the past simulations. If a neural network model can be developed and jointly used with the simulation model, the historical data and the simulation results can be used to train the network model.

Therefore, if the similar scenarios arise in later production periods, the neural network can be used to make necessary decisions and re-simulation would be unnecessary.

REFERENCES

Agapiou, J., 1992, "Cutting Tool Strategies for Multifunctional Part Configurations: Part I – Analytical Economic Models for Cutting Tools", *International Journal of Advanced Manufacturing Technology*, Vol.7, 59-69.

Agapiou, J., 1992, "Cutting Tool Strategies for Multifunctional Part Configurations: Part II – Discussion of Experimental and Analytical Results", *International Journal of Advanced Manufacturing Technology*, Vol.7, 70-81.

Ayres, R., 1988, "Future trends in Factory Automation", *Manufacturing review*, Vol. 1, No. 2, 93-103.

Banks, J. and Carson, J., 1984, Discrete-Event System Simulation, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Carrie, A.S. and Perera, D.T.S., 1986, "Work Scheduling in FMS Under tool Availability Constraints", *International Journal of Production Research*, Vol. 24, No. 6.

Co, H.C., Biermann, J.S., and Chen, S.K., 1990, "A Methodical Approach to the Flexible Manufacturing System Batching, Loading and Tool Configuration Problems", *International Journal of Production Research*, Vol.28, No.12, 2171-2186.

Commare, U.L., Diega N.L., and Passannanti, A., 1983, "Optimum Tool Replacement Policies with Penalty Cost for Unforseen Tool Failure", *Journal of Machine Tool Decision Research*, Vol.23, No.4, 237-243.

Cook, N.H., 1973, "Tool Wear and Tool Life", *ASME Journal of Engineering for Industry*, 931-938.

Diega, N.L.S. and Passannanti, A., 1984, "Optimum Tool Replacement Policies with Running-in of the Cutting-Edge", *Annals of the CIRP*, Vol.33, 1-3.

ElMaraghy, H.A., 1985, "Automated Tool Management in Flexible Manufacturing", *Journal of Manufacturing Systems*, Vol.4, No.1, 1-13.

Fenton, R.G. and Joseph, N.D., 1979, "The effects of Statistical Nature of Tool-Life on the Economics of Machining", *International Journal of Machine Tool Design Research*, Vol.19, 43-50.

Gaalman, G.L., Nawijn, W.M., and Platzer, L.W., 1987, "Tool Sharing in an FMS - A Feasibility Study", *Journal of Engineering Cost and Production Economics*, Vol.12, 107-115.

Galante, G. and Lombardo, A., 1991, "Tool Replacement with Adaptive Control in a Non-Stationary Non-Periodic Stochastic Process", *International Journal of Production Research*, Vol.29, No.11, 2365-2374.

Gray, A.E., Seidmann, A., and Stecke, K.E, 1993, "A Synthesis of Decision Models for Tool Management in Automated Manufacturing", *Management Science* Vol.39, No.5, 549-565.

Gruver, W.A., and Senninger, M.T., 1990, "Tooling Management in an FMS", *CIRP: Annals of the International Institution for Production Engineering Research*, March, 40-

45.

Han, M., Na, Y.K., and Hogg, G.L., 1989, "Real-time Control and Job Dispatching in Flexible Manufacturing Systems", *International Journal of Production Research*, Vol.27, No.8, 1257-1267.

Hankins, S.L. and Rovito, V.P., 1984, "A Comparison of Two Tool Allocation and Distribution Strategies for FMS", *Proceedings of the First ORSA/TIMS Conference on Flexible Manufacturing Systems*, Ann Arbor, MI, USA, 272-277.

Hitomi, K., Nakamura, N., and Inoue, S., 1979, "Reliability Analysis of Cutting Tools", *Journal of Engineering for Industry*, Vol.101, No.1, 185-190.

Hwang, S., 1986, "Part Selection Problems in FMSs Planning Stage", *Proceeding of the 2nd ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*, Ann Arbor MI, Elsevier Science Publishers B.V., Amsterdam, 297-309.

Hwang, S. and Shogan, A., 1989, "Modelling and Solving an FMS Part Selection Problem", *International Journal of Production Research*, Vol.27, No.8, 1349-1366.

Kolahan, F., 1993, "The Effect of Tool Sharing on Reliability of Flexible Manufacturing Systems", Master's Thesis, Department of Mechanical Engineering, Ottawa University, Ottawa, Canada.

Lenz, J.E., 1988, Flexible Manufacturing: Benefits for the Low-Inventory Factory, Marcel Dekker, Inc., New York.

Liang, M. and Dutta, S., 1994, "Solving a Combined Part-Selection, Machine-Loading and Tool-Configuration Problem in Flexible Manufacturing Systems," *Journal of Production and Operations Management* (to appear).

Luggen, W.W., 1991, Flexible Manufacturing Cells and Systems, Prentice-Hall, Englewood Cliffs, New Jersey.

Malakooti, B., Wang, J., and Tandler, E., 1990, "A Sensor-Based Accelerated Approach for Multi-Attribute Machinability and Tool Life Evaluation", *International Journal of Production Research,* Vol.28, No.12, 2373-2392.

Maleki, R.A., 1991, Flexible Manufacturing Systems: The Technology and Management, Prentice-Hall, Englewood Cliffs, New Jersey.

Mason, F., 1986, "Computerized Cutting-Tool Management", *American Machinists & Automated Manufacturing,* Vol.130, No.5, 105-132.

Minis, I., Harhalakis, G. and Jajadia, S., 1990, "Manufacturing Cell Formation with Multiple Functionally Identical Machines", *Manufacturing Review*, Vol.3, No.4, 252-261.

Nagasaka, K. and Hashimoto, F., 1988, "Tool Wear Prediction and Economics in Machining Stepped Parts", *International Journal of Machine Tools and Manufacture,* Vol.28, No.4, 569-576.

O'Grady, P. and Menon, U., 1987, "Loading a Flexible Manufacturing System", *International Journal of Production Research,* Vol.25, No.7, 1053-1068.

Pritsker, A.A.B., 1986, Introduction to Simulation and SLAM II, John Wiley & Sons, New York.

Ramalingam, S., 1978, "Tool life Distribution. Part 2: Multiple-Injury Tool-life Model", *Journal of Engineering for Industry*, Vol.99, No.3, 523-531.

Ramalingam, S., Peng, Y.I., and Wataon, J.D., 1978, "Tool life Distribution. Part 3: Mechanism of Single Injury Tool Failure and Tool Life Distribution in interrupted cutting", *ASME Journal of Engineering for Industry*, Vol.100, No.1, 193-200.

Ramalingam, S. and Watson, J.D., 1978a, "Tool life Distribution. Part 1: Single-Injury Tool-life Model", *ASME Journal of Engineering for Industry*, Vol.99, No.3, 519-522.

Ramalingam, S. and Watson, J.D., 1978b, "Tool life Distribution. Part 4: Minor Phases in Work Material and Multiple-Injury Tool Failure", *ASME Journal of Engineering for Industry*, Vol.100, No.1, 201-209.

Ravignani, G.L., Zompi, A., and Levi, R., 1978, "Optimal Replacement Policy", *American Society of Mechanical Engineers*, Paper No. 78-WA/PROD-25.

Reddy, C.E., Chetty, O.V.K., and Chaudhuri, D., 1992, "A Petri Net Based Approach for Analyzing Tool Management Issues in FMS", *International Journal of Production Research*, Vol.30, No.6, 1427-1446.

Stecke, K. and Kim, I., 1986, "A Flexible Approach to Implementing the Short-Term FMS Planning Function", *Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems*, Ann Arbor MI, Elsevier Science Publishers B.V., Amsterdam, 283-295, August, 12-15.

Stecke, K. and Kim, I., 1988, "Comparison of Various Approaches to Part Selection Problem in Flexible Manufacturing Systems", *Recent Developments in Production Research*, edited by A. Mital, Elsevier Science Publishers B.V., Amsterdam, Netherlands, 675-681.

Sundaram, R.M., 1989, "Cellular Manufacturing Systems Design with Alternate Routing Considerations", *Journal of Mechanical Working Technology*, Vol. 20, 425-432.

Tang, C.S. and Denardo, E.V., 1988, "Models Arising from a Flexible Manufacturing Machine; Part I: Minimization of the Number of Tool Switches", *Operations Research*, Vol.36, No.5, 355-362.

Taylor, F.W., 1907, "On the Art of Cutting Metals", ASME Transactions, Vol.28, 310-350.

Tomek, P., April 1986, "Tooling Strategies Related to FMS Management", *The FMS Magazine*, Vol.4, 102-107.

Wagner, J.G. and Barash, M.M., 1971, "Study of the Distribution of the life of HSS Tools", *ASME Journal of Engineering for Industry*, Vol.93, November, 212-218.

Whitney, C.K. and Gaul, T.S., 1984, "Sequential Decision Procedures for Batching and Balancing in FMSs", *Proceedings of the 1st ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*, Ann Arbor MI, Elsevier Science Publishers B.V., Amsterdam, 243-248.

Zhou, C., Chandra, J., and Wysk, R., 1990, "Optimal Cutting Tool Replacement Based on Tool Wear Status", *International Journal of production research*, Vol.28, No.7, 1357-1367.

APPENDIX 1

SLAM II Control and Network Statements

SLAM II Control Statements

```
AMINI1/1/1994,10,Y,N,Y/Y,Y,Y/10,132;
LIMITS,130,48,1000;
SEEDS;
INTLC,XX(182)=3,XX(181)=540; INTLC, XX(192)=90, XX(191)=228, XX(190)=60
XX(189)=120 ,XX(188)=150, XX(187)=102, XX(186)=270,
XX(185)=102 ,XX(184)=90, XX(183)=54;
PRIORITY/63,LVF(41)/62,LVF(41)/61,LVF(41)/6,LVF(41)/5,LVF(41)/4,LVF(41)/3,LVF(
41)/2,LVF(41)/1,LVF(41);
PRIORITY/83,LVF(41)/82,LVF(41)/81,LVF(41)/80,LVF(41)/79,LVF(41)/78,LVF(41)/73,
LVF(41)/72,LVF(41)/77,LVF(41)/76,LVF(41)/75,LVF(41)/66,LVF(41)/65,LVF(41)/64,
LVF(41);
ARRAY(1,47)/8.0,21,0.0,0.0,0.0,0.0,0.0,1.0,3.0,4.0,8.0,4.0,4.0,7.0,3.0,5.0,2.0,
8.0,8.0,4.0,10,16,10,12,14,15,13,9.0,11,16,19,18,17,23,24,17,18,21,20,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(2,47)/9.0,10,0.0,0.0,0.0,0.0,0.0,8.0,2.0,6.0,2.0,8.0,6.0,5.0,2.0,4.0,5.0,
6.0,7.0,7.0,16,9.0,16,15,13,14,0.0,14,10,9.0,18,24,22,22,17,19,19,22,23,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(3,47)/11,12,13,0.0,0.0,0.0,0.0,5.0,8.0,4.0,1.0,1.0,4.0,1.0,8.0,6.0,6.0,
7.0,8.0,6.0,10,15,10,11,9.0,10,0.0,9.0,16,16,24,18,17,18,23,20,24,17,24,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(4,47)/14,23,15,24,0.0,0.0,0.0,0.0,6.0,2.0,8.0,0.0,5.0,3.0,6.0,1.0,3.0,5.0,
0.0,0.0,9.0,12,9.0,14,13,16,0.0,0.0,11,13,19,17,0.0,24,20,21,23,18,21,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(5,47)/16,14,25,26,0.0,0.0,0.0,0.0,1.0,0.0,2.0,0.0,4.0,8.0,1.0,7.0,1.0,8.0,
0.0,0.0,0.0,0.0,0.0,9.0,0.0,12,0.0,0.0,0.0,16,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(6,47)/24,27,28,17,0.0,0.0,0.0,0.0,8.0,0.0,0.0,0.0,0.0,6.0,5.0,8.0,0.0,3.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(7,47)/31,39,36,37,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,7.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(8,47)/31,18,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(9,47)/19,31,30,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(10,47)/33,34,35,38,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
;LOADING TOOLS FOR MACHINE GROUP 1
ARRAY(11,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(12,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
```

```
ARRAY(13,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
;LOADING TOOLS FOR MACHINE GROUP 2
ARRAY(14,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(15,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
;LOADING TOOLS FOR MACHINE GROUP 3
ARRAY(16,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(17,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(18,47)/5.0,5.0,5.0,5.0,5.0,5.0,5.0,5.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
;TOOL REPLACEMENT POLICIES
;SHARING [19,1]=1 GR1,SHARING[19,2]=1 GR2,SHARING [19,3]=1 GR3
ARRAY(19,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,1.0,1.0,1.0,1.0,1.0;
;PROCESSING TIME(MIN.)
ARRAY(20,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,13,2.0,6.0,8.0,3.0,6.0,1.0,2.0,4.0,4.0,
2.0,4.0,2.0,4.0,7.0,4.0,24,3.0,6.0,3.0,7.0,5.0,1.0,5.0,4.0,7.0,6.0,7.0,4.0,5.0,
8.0,6.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(21,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,9.0,8.0,7.0,2.0,8.0,1.0,9.0,2.0,
3.0,6.0,5.0,1.0,12,4.0,12,3.0,1.0,6.0,16,3.0,7.0,5.0,7.0,12,3.0,6.0,4.0,7.0,4.0,
7.0,4.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(22,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,5.0,9.0,9.0,7.0,3.0,9.0,6.0,9.0,3.0,
5.0,4.0,2.0,5.0,7.0,2.0,5.0,5.0,2.0,4.0,0.0,13,2.0,5.0,2.0,5.0,13,4.0,2.0,2.0,
3.0,5.0,7.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(23,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,8.0,2.0,2.0,0.0,4.0,2.0,8.0,3.0,6.0,
3.0,0.0,1.0,1.0,1.0,1.0,5.0,4.0,0.0,0.0,5.0,6.0,0.0,4.0,1.0,2.0,3.0,5.0,5.0,3.0,
2.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(24,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,7.0,0.0,2.0,0.0,2.0,6.0,7.0,6.0,
2.0,1.0,0.0,0.0,0.0,6.0,0.0,6.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(25,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,2.0,0.0,0.0,0.0,3.0,8.0,0.0,0.0,
6.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
ARRAY(26,47)/0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,6.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
INTLC,XX(10)=0,XX(9)=0,XX(8)=0,XX(6)=0,XX(7)=0,XX(5)=0 ,XX(4)=0,
XX(3)=0,XX(1)=0,
XX(2)=0;
INTLC,XX(20)=0,XX(19)=0,XX(18)=0,XX(17)=0,XX(16)=0,XX(15)=0,XX(14)=0,XX(13)=0,
XX(12)=0,XX(11)=0;
```

```
100 INTLC,XX(30)=0,XX(29)=0,XX(28)=0,XX(27)=0,XX(26)=0,XX(25)=0, XX(24)=0
    ,XX(23)=0,
101 XX(22)=0,XX(21)=0;
102 INTLC,XX(40)=0,XX(39)=0,XX(38)=0,XX(37)=0,XX(36)=0,XX(35)=0,XX(34)=0,
    XX(33)=0,
103 XX(32)=0,XX(31)=0;
104 INTLC,XX(50)=0,XX(49)=0,XX(48)=0,XX(47)=0,XX(46)=0,XX(45)=0,XX(44)=0,
    XX(43)=0,
105 XX(42)=0,XX(41)=0;
106 INTLC,XX(60)=0,XX(59)=0,XX(58)=0,XX(57)=0,XX(56)=0,XX(55)=0,XX(54)=0,
    XX(53)=0,
107 XX(52)=0,XX(51)=0;
108 INTLC,XX(70)=0,XX(69)=0,XX(68)=0,XX(67)=0,XX(66)=0,XX(65)=0,XX(64)=0,
    XX(63)=0,
109 XX(62)=0,XX(61)=0;
110 INTLC,XX(80)=0,XX(79)=0,XX(78)=0,XX(77)=0,XX(76)=0,XX(75)=0,XX(74)=0,
    XX(73)=0,
111 XX(72)=0,XX(71)=0;
112 INTLC,XX(90)=0,XX(89)=0,XX(88)=0,XX(87)=0,XX(86)=0,XX(85)=0,XX(84)=0,
    XX(83)=0,
113 XX(82)=0,XX(81)=0;
114 INTLC,XX(100)=0,XX(99)=0,XX(98)=0,XX(97)=0,XX(96)=0,XX(95)=0,XX(94)=0,
    XX(93)=0,
115 XX(92)=0,XX(91)=0;
116 INTLC,XX(110)=0,XX(109)=0,XX(108)=0,XX(107)=0,XX(106)=0,XX(105)=0,XX(104)=0,
117 XX(103)=0,XX(102)=0,XX(101)=0;
118 INTLC,XX(120)=0,XX(119)=0,XX(118)=0,XX(117)=0,XX(116)=0,XX(115)=0,XX(114)=0,
119 XX(113)=0,XX(112)=0,XX(111)=0;
120 INTLC,XX(130)=0,XX(129)=0,XX(128)=0,XX(127)=0,XX(126)=0,XX(125)=0,XX(124)=0,
121 XX(123)=0,XX(122)=0,XX(121)=0;
122 INTLC,XX(200)=0,XX(140)=0,XX(139)=0,XX(138)=0,XX(137)=0,XX(136)=0,XX(135)=0,
123 XX(134)=0,XX(133)=0,XX(132)=0,XX(131)=0;
124 MONTR,CLEAR,400;
```

SLAM II Network Statements

```
NETWORK;
;FILE ARRIVAL.NET, NODE LABEL SEED
;FILE GRR1.NET, NODE LABEL SEED
;FILE GRR2.NET, NODE LABEL SEED
;FILE GRR3.NET, NODE LABEL SEED
;FILE RPERREP.NET, NODE LABEL SEED ZAAA
     RESOURCE/1,MACHIN1,2;
     RESOURCE/2,MACHIN2,3;
     RESOURCE/3,MACHIN3,4;
     RESOURCE/4,MACHIN4,5;
     RESOURCE/5,MACHIN5,6;
     RESOURCE/6,TOOL1,11;
     RESOURCE/7,TOOL2(5),12;
     RESOURCE/8,TOOL3(5),13;
     RESOURCE/9,TOOL4(5),14;
     RESOURCE/10,TOOL5(5),15;
     RESOURCE/11,TOOL6(5),16;
     RESOURCE/12,TOOL7(5),17;
     RESOURCE/13,TOOL8(5),18;
     RESOURCE/14,TOOL9(5),19;
     RESOURCE/15,TOOL10(5),20;
     RESOURCE/16,TOOL11(5),21;
     RESOURCE/17,TOOL12(5),22;
     RESOURCE/18,TOOL13(5),23;
     RESOURCE/19,TOOL14(5),24;
     RESOURCE/20,TOOL15(5),25;
     RESOURCE/21,TOOL16(5),26;
     RESOURCE/22,TOOL17(5),27;
     RESOURCE/23,TOOL18(5),28;
     RESOURCE/24,TOOL19(5),29;
     RESOURCE/25,TOOL20(5),30;
     RESOURCE/26,TOOL21(5),31;
     RESOURCE/27,TOOL22(5),32;
     RESOURCE/28,TOOL23(5),33;
     RESOURCE/29,TOOL24(5),34;
     RESOURCE/30,TOOL25(5),35;
     RESOURCE/31,TOOL26(5),36;
     RESOURCE/32,TOOL27(5),37;
     RESOURCE/33,TOOL28(5),38;
     RESOURCE/34,TOOL29(5),39;
     RESOURCE/35,TOOL30(5),40;
     RESOURCE/36,TOOL31(5),41;
     RESOURCE/37,TOOL32(5),42;
     RESOURCE/38,TOOL33(5),43;
     RESOURCE/39,TOOL34(5),44;
     RESOURCE/40,TOOL35(5),45;
     RESOURCE/41,TOOL36(5),46;
     RESOURCE/42,TOOL37(5),47;
     RESOURCE/43,TOOL38(5),48;
```

```
      RESOURCE/44,TOOL39(5),49;
      RESOURCE/45,TOOL40(5),50;
      RESOURCE/56,LOADER(5),61;
      RESOURCE/57,AGVL(5),62;
      RESOURCE/58,TRACK(5),63;
      RESOURCE/59,ULOADER(5),64;
      RESOURCE/60,AGVU(5),65;
      RESOURCE/61,UTRACK(5),66;
      RESOURCE/62,L1ROB,75;
      RESOURCE/63,L2ROB,76;
      RESOURCE/64,L3ROB,77;
      RESOURCE/65,L4ROB,72;
      RESOURCE/66,L5ROB,73;
      RESOURCE/67,L6ROB,78;
      RESOURCE/68,L7ROB,79;
      RESOURCE/69,L8ROB,80;
      RESOURCE/70,MACHIN6,81;
      RESOURCE/71,MACHIN7,82;
      RESOURCE/72,MACHIN8,83;
      RESOURCE/101,TOOL41(5),101;
      RESOURCE/102,TOOL42(5),102;
      RESOURCE/103,TOOL43(5),103;
      RESOURCE/104,TOOL44(5),104;
      RESOURCE/105,TOOL45(5),105;
      RESOURCE/106,TOOL46(5),106;
      RESOURCE/107,TOOL47(5),107;
      RESOURCE/108,TOOL48(5),108;
      RESOURCE/109,TOOL49(5),109;
      RESOURCE/110,TOOL50(5),110;
      RESOURCE/111,TOOL51(5),111;
      RESOURCE/112,TOOL52(5),112;
      RESOURCE/113,TOOL53(5),113;
      RESOURCE/114,TOOL54(5),114;
      RESOURCE/115,TOOL55(5),115;
      RESOURCE/116,TOOL56(5),116;
      RESOURCE/117,TOOL57(5),117;
      RESOURCE/118,TOOL58(5),118;
      RESOURCE/119,TOOL59(5),119;
      RESOURCE/120,TOOL60(5),120;
      RESOURCE/121,TOOL61(5),121;
      RESOURCE/122,TOOL62(5),122;
      RESOURCE/123,TOOL63(5),123;
      RESOURCE/124,TOOL64(5),124;
      RESOURCE/125,TCRIB,125;
;FILE TOOL2.NET, NODE LABEL SEED
;FILE ARRIVAL.NET, NODE LABEL SEED
;
CR1   CREATE,XX(183),,1;
      ACTIVITY;
AS1   ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=1;
      ACTIVITY,,,G1;
;
```

```
CR2   CREATE,XX(184),,2;
      ACTIVITY;
AS2   ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=2;
      ACTIVITY,,,G1;
;
CR3   CREATE,XX(185),,3;
      ACTIVITY;
AS3   ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=3;
      ACTIVITY,,,G1;
;
CR4   CREATE,XX(186),,4;
      ACTIVITY;
AS4   ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=4;
      ACTIVITY,,,G1;
;
CR5   CREATE,XX(187),,5;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=5;
      ACTIVITY,,,G1;
;
CR6   CREATE,XX(188),,30;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=6;
      ACTIVITY,,,G1;
;
      CREATE,XX(189),,31;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=7;
      ACTIVITY,,,G1;
;
      CREATE,XX(190),,32;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=8;
      ACTIVITY,,,G1;
;
      CREATE,XX(191),,33;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=9;
      ACTIVITY,,,G1;
;
      CREATE,XX(192),,34;
      ACTIVITY;
      ASSIGN,ATRIB(41)=TNOW,ATRIB(6)=10;
      ACTIVITY,,,G1;
;FILE GRR1.NET, NODE LABEL SEED
YAAB  GOON,1;
      ACTIVITY,,ATRIB(12).LT.8;
      ACTIVITY,,ATRIB(12).GE.8,M1E;
M1B   ASSIGN,ATRIB(12)=ATRIB(12)+1;
      ACTIVITY;
EVE2  EVENT,2,1;
      ACTIVITY,,ATRIB(13).EQ.1;
```

```
     ACTIVITY,,ATRIB(13).EQ.2,AT2;
     ACTIVITY,,ATRIB(13).EQ.3,AT3;
     ACTIVITY,,ATRIB(13).EQ.4,AT4;
     ACTIVITY,,ATRIB(13).EQ.5,AT5;
     ACTIVITY,,ATRIB(13).EQ.6,AT6;
     ACTIVITY,,ATRIB(13).EQ.7,AT7;
     ACTIVITY,,ATRIB(13).EQ.8,AT8;
     ACTIVITY,,ATRIB(13).EQ.0,YAAB;
AT1  ASSIGN,XX(1)=XX(1)+USERF(1),XX(81)=XX(1)-XX(2),1;
     ACTIVITY;
T1   AWAIT(11),TOOL1,,2;
     ACTIVITY,USERF(1),XX(1).LT.XX(2);
     ACTIVITY,USERF(1)-XX(81),XX(1).GE.XX(2),ALT1;
FT1  FREE,TOOL1;
     ACTIVITY,,,YAAB;
AT2  ASSIGN,XX(3)=XX(3)+USERF(1),XX(81)=XX(3)-XX(4),1;
     ACTIVITY;
T2   AWAIT(12),TOOL2,,1;
     ACTIVITY,USERF(1),XX(3).LT.XX(4);
     ACTIVITY,USERF(1)-XX(81),XX(3).GE.XX(4),ALT2;
FT2  FREE,TOOL2;
     ACTIVITY,,,YAAB;
AT3  ASSIGN,XX(5)=XX(5)+USERF(1),XX(81)=XX(5)-XX(6),1;
     ACTIVITY;
T3   AWAIT(13),TOOL3,,1;
     ACTIVITY,USERF(1),XX(5).LT.XX(6);
     ACTIVITY,USERF(1)-XX(81),XX(5).GE.XX(6),ALT3;
FT3  FREE,TOOL3;
     ACTIVITY,,,YAAB;
AT4  ASSIGN,XX(7)=XX(7)+USERF(1),XX(81)=XX(7)-XX(8),1;
     ACTIVITY;
T4   AWAIT(14),TOOL4,,1;
     ACTIVITY,USERF(1),XX(7).LT.XX(8);
     ACTIVITY,USERF(1)-XX(81),XX(7).GE.XX(8),ALT4;
FT4  FREE,TOOL4;
     ACTIVITY,,,YAAB;
AT5  ASSIGN,XX(9)=XX(9)+USERF(1),XX(81)=XX(9)-XX(10),1;
     ACTIVITY;
T5   AWAIT(15),TOOL5,,1;
     ACTIVITY,USERF(1),XX(9).LT.XX(10);
     ACTIVITY,USERF(1)-XX(81),XX(9).GE.XX(10),ALT5;
FT5  FREE,TOOL5;
     ACTIVITY,,,YAAB;
AT6  ASSIGN,XX(11)=XX(11)+USERF(1),XX(81)=XX(11)-XX(12),1;
     ACTIVITY;
T6   AWAIT(16),TOOL6,,1;
     ACTIVITY,USERF(1),XX(11).LT.XX(12);
     ACTIVITY,USERF(1)+XX(81),XX(11).GE.XX(12),ALT6;
FT6  FREE,TOOL6;
     ACTIVITY,,,YAAB;
AT7  ASSIGN,XX(13)=XX(13)+USERF(1),XX(81)=XX(13)-XX(14),1;
     ACTIVITY;
```

```
T7   AWAIT(17),TOOL7,,1;
     ACTIVITY,USERF(1),XX(13).LT.XX(14);
     ACTIVITY,USERF(1)-XX(81),XX(13).GE.XX(14),ALT7;
FT7  FREE,TOOL7;
     ACTIVITY,,,YAAB;
AT8  ASSIGN,XX(15)=XX(15)+USERF(1),XX(81)=XX(15)-XX(16),1;
     ACTIVITY;
T8   AWAIT(18),TOOL8,,1;
     ACTIVITY,USERF(1),XX(15).LT.XX(16);
     ACTIVITY,USERF(1)-XX(81),XX(15).GE.XX(16),ALT8;
FT8  FREE,TOOL8,1;
     ACTIVITY,,,YAAB;
M2B  ASSIGN,ATRIB(14)=ATRIB(14) + 1;
     ACTIVITY;
EVE3 EVENT,3,1;
     ACTIVITY,,ATRIB(15).EQ.1;
     ACTIVITY,,ATRIB(15).EQ.2,YAAD;
     ACTIVITY,,ATRIB(15).EQ.3,YAAE;
     ACTIVITY,,ATRIB(15).EQ.4,YAAF;
     ACTIVITY,,ATRIB(15).EQ.5,YAAG;
     ACTIVITY,,ATRIB(15).EQ.7,YAAH;
     ACTIVITY,,ATRIB(15).EQ.6,YAAI;
     ACTIVITY,,ATRIB(15).EQ.0,YAAC;
     ACTIVITY,,ATRIB(15).EQ.8,YAAJ;
     ASSIGN,XX(17)=XX(17) + USERF(2),XX(82)=XX(17)-XX(18);
     ACTIVITY;
T9   AWAIT(19),TOOL9,,1;
     ACTIVITY,USERF(2),XX(17).LT.XX(18);
     ACTIVITY,USERF(2)-XX(82),XX(17).GE.XX(18),CT9;
FT9  FREE,TOOL9;
     ACTIVITY;
YAAC GOON,1;
     ACTIVITY,,ATRIB(14).LT.8,M2B;
     ACTIVITY,,ATRIB(14).GE.8,M2E;
YAAD ASSIGN,XX(19)=XX(19) + USERF(2),XX(82)=XX(19)-XX(20);
     ACTIVITY;
T10  AWAIT(20),TOOL10,,1;
     ACTIVITY,USERF(2),XX(19).LT.XX(20);
     ACTIVITY,USERF(2)-XX(82),XX(19).GE.XX(20),CT10;
FT10 FREE,TOOL10;
     ACTIVITY,,,YAAC;
YAAE ASSIGN,XX(21)=XX(21) + USERF(2),XX(82)=XX(21)-XX(22);
     ACTIVITY;
T11  AWAIT(21),TOOL11,,1;
     ACTIVITY,USERF(2),XX(21).LT.XX(22);
     ACTIVITY,USERF(2)-XX(82),XX(21).GE.XX(22),CT11;
FT11 FREE,TOOL11;
     ACTIVITY,,,YAAC;
YAAF ASSIGN,XX(23)=XX(23) + USERF(2),XX(82)=XX(23)-XX(24);
     ACTIVITY;
T12  AWAIT(22),TOOL12,,1;
     ACTIVITY,USERF(2),XX(23).LT.XX(24);
```

```
     ACTIVITY,USERF(2)-XX(82),XX(23).GE.XX(24),CT12;
FT12 FREE,TOOL12;
     ACTIVITY,,,YAAC;
YAAG ASSIGN,XX(25)=XX(25) + USERF(2),XX(82)=XX(25)-XX(26);
     ACTIVITY;
T13  AWAIT(23),TOOL13,,1;
     ACTIVITY,USERF(2),XX(25).LT.XX(26);
     ACTIVITY,USERF(2)-XX(82),XX(25).GE.XX(26),CT13;
FT13 FREE,TOOL13;
     ACTIVITY,,,YAAC;
YAAH ASSIGN,XX(29)=XX(29) + USERF(2),XX(82)=XX(29)-XX(30);
     ACTIVITY;
T15  AWAIT(25),TOOL15,,1;
     ACTIVITY,USERF(2),XX(29).LT.XX(30);
     ACTIVITY,USERF(2)-XX(82),XX(29).GE.XX(30),CT15;
FT15 FREE,TOOL15;
     ACTIVITY,,,YAAC;
YAAI ASSIGN,XX(27)=XX(27) + USERF(2),XX(82)=XX(27)-XX(28);
     ACTIVITY;
T14  AWAIT(24),TOOL14,,1;
     ACTIVITY,USERF(2),XX(27).LT.XX(28);
     ACTIVITY,USERF(2)-XX(82),XX(27).GE.XX(28),CT14;
FT14 FREE,TOOL14;
     ACTIVITY,,,YAAC;
YAAJ ASSIGN,XX(31)=XX(31) + USERF(2),XX(82)=XX(31)-XX(32);
     ACTIVITY;
T16  AWAIT(26),TOOL16,,1;
     ACTIVITY,USERF(2),XX(31).LT.XX(32);
     ACTIVITY,USERF(2)-XX(82),XX(31).GE.XX(32),CT16;
FT16 FREE,TOOL16;
     ACTIVITY,,,YAAC;
M3B  ASSIGN,ATRIB(16)=ATRIB(16) + 1,1;
     ACTIVITY;
EVE4 EVENT,4,1;
     ACTIVITY,,ATRIB(17).EQ.1;
     ACTIVITY,,ATRIB(17).EQ.2,YAAL;
     ACTIVITY,,ATRIB(17).EQ.3,YAAM;
     ACTIVITY,,ATRIB(17).EQ.4,YAAN;
     ACTIVITY,,ATRIB(17).EQ.5,YAAO;
     ACTIVITY,,ATRIB(17).EQ.0,YAAK;
     ACTIVITY,,ATRIB(17).EQ.6,YAAP;
     ACTIVITY,,ATRIB(17).EQ.7,YAAQ;
     ACTIVITY,,ATRIB(17).EQ.8,YAAR;
     ASSIGN,XX(33)=XX(33) + USERF(3),XX(83)=XX(33)-XX(34);
     ACTIVITY;
T17  AWAIT(27),TOOL17,,1;
     ACTIVITY,USERF(3),XX(33).LT.XX(34);
     ACTIVITY,USERF(3)-XX(83),XX(33).GE.XX(34),CT17;
FT17 FREE,TOOL17;
     ACTIVITY;
YAAK GOON,1;
     ACTIVITY,,ATRIB(16).LT.8,M3B;
```

```
     ACTIVITY,,ATRIB(16).GE.8,M3E;
YAAL ASSIGN,XX(35)=XX(35) + USERF(3),XX(83)=XX(35)-XX(36);
     ACTIVITY;
T18  AWAIT(28),TOOL18,,1;
     ACTIVITY,USERF(3),XX(35).LT.XX(36);
     ACTIVITY,USERF(3)-XX(83),XX(35).GE.XX(36),CT18;
FT18 FREE,TOOL18;
     ACTIVITY,,,YAAK;
YAAM ASSIGN,XX(37)=XX(37) + USERF(3),XX(83)=XX(37)-XX(38);
     ACTIVITY;
T19  AWAIT(29),TOOL19,,1;
     ACTIVITY,USERF(3),XX(37).LT.XX(38);
     ACTIVITY,USERF(3)-XX(83),XX(37).GE.XX(38),CT19;
FT19 FREE,TOOL19;
     ACTIVITY,,,YAAK;
YAAN ASSIGN,XX(39)=XX(39) + USERF(3),XX(83)=XX(39)-XX(40);
     ACTIVITY;
T20  AWAIT(30),TOOL20,,1;
     ACTIVITY,USERF(3),XX(39).LT.XX(40);
     ACTIVITY,USERF(3)-XX(83),XX(39).GE.XX(40),CT20;
FT20 FREE,TOOL20;
     ACTIVITY,,,YAAK;
YAAO ASSIGN,XX(41)=XX(41) + USERF(3),XX(83)=XX(41)-XX(42);
     ACTIVITY;
T21  AWAIT(31),TOOL21,,1;
     ACTIVITY,USERF(3),XX(41).LT.XX(42);
     ACTIVITY,USERF(3)-XX(83),XX(41).GE.XX(42),CT21;
FT21 FREE,TOOL21;
     ACTIVITY,,,YAAK;
YAAP ASSIGN,XX(43)=XX(43) + USERF(3),XX(83)=XX(43)-XX(44);
     ACTIVITY;
T22  AWAIT(32),TOOL22,,1;
     ACTIVITY,USERF(3),XX(43).LT.XX(44);
     ACTIVITY,USERF(3)-XX(83),XX(43).GE.XX(44),CT22;
FT22 FREE,TOOL22;
     ACTIVITY,,,YAAK;
YAAQ ASSIGN,XX(45)=XX(45) + USERF(3),XX(83)=XX(45)-XX(46);
     ACTIVITY;
T23  AWAIT(33),TOOL23,,1;
     ACTIVITY,USERF(3),XX(45).LT.XX(46);
     ACTIVITY,USERF(3)-XX(83),XX(45).GE.XX(46),CT23;
FT23 FREE,TOOL23;
     ACTIVITY,,,YAAK;
YAAR ASSIGN,XX(47)=XX(47) + USERF(3),XX(83)=XX(47)-XX(48);
     ACTIVITY;
T24  AWAIT(34),TOOL24,,1;
     ACTIVITY,USERF(3),XX(47).LT.XX(48);
     ACTIVITY,USERF(3)-XX(83),XX(47).GE.XX(48),CT24;
FT24 FREE,TOOL24;
     ACTIVITY,,,YAAK;
;FILE GRR2.NET, NODE LABEL SEED
M4B  ASSIGN,ATRIB(19)=ATRIB(19)+1,1;
```

```
     ACTIVITY;
EVES EVENT,5,1;
     ACTIVITY,,ATRIB(18).EQ.9;
     ACTIVITY,,ATRIB(18).EQ.10,XAAC;
     ACTIVITY,,ATRIB(18).EQ.11,XAAD;
     ACTIVITY,,ATRIB(18).EQ.12,XAAE;
     ACTIVITY,,ATRIB(18).EQ.13,XAAF;
     ACTIVITY,,ATRIB(18).EQ.14,XAAG;
     ACTIVITY,,ATRIB(18).EQ.15,XAAH;
     ACTIVITY,,ATRIB(18).EQ.16,XAAI;
     ACTIVITY,,ATRIB(18).EQ.0,XAAB;
     ASSIGN,XX(49)=XX(49) + USERF(4),XX(84)=XX(49)-XX(50);
     ACTIVITY;
T25  AWAIT(35),TOOL25,,1;
     ACTIVITY/25,USERF(4),XX(49).LT.XX(50);
     ACTIVITY/75,USERF(4)-XX(84),XX(49).GE.XX(50),CT25;
FT25 FREE,TOOL25;
     ACTIVITY;
XAAB GOON,1;
     ACTIVITY,,ATRIB(19).LT.8,M4B;
     ACTIVITY,,ATRIB(19).GE.8,M4E;
XAAC ASSIGN,XX(51)=XX(51) + USERF(4),XX(84)=XX(51)-XX(52);
     ACTIVITY;
T26  AWAIT(36),TOOL26,,1;
     ACTIVITY/26,USERF(4),XX(51).LT.XX(52);
     ACTIVITY/76,USERF(4)-XX(84),XX(51).GE.XX(52),CT26;
FT26 FREE,TOOL26;
     ACTIVITY,,,XAAB;
XAAD ASSIGN,XX(53)=XX(53) + USERF(4),XX(84)=XX(53)-XX(54);
     ACTIVITY;
T27  AWAIT(37),TOOL27,,1;
     ACTIVITY/27,USERF(4),XX(53).LT.XX(54);
     ACTIVITY/77,USERF(4)-XX(84),XX(53).GE.XX(54),CT27;
FT27 FREE,TOOL27;
     ACTIVITY,,,XAAB;
XAAE ASSIGN,XX(55)=XX(55) + USERF(4),XX(84)=XX(55)-XX(56);
     ACTIVITY;
T28  AWAIT(38),TOOL28,,1;
     ACTIVITY/28,USERF(4),XX(55).LT.XX(56);
     ACTIVITY/78,USERF(4)-XX(84),XX(55).GE.XX(56),CT28;
FT28 FREE,TOOL28;
     ACTIVITY,,,XAAB;
XAAF ASSIGN,XX(57)=XX(57) + USERF(4),XX(84)=XX(57)-XX(58);
     ACTIVITY;
T29  AWAIT(39),TOOL29,,1;
     ACTIVITY/29,USERF(4),XX(57).LT.XX(58);
     ACTIVITY/79,USERF(4)-XX(84),XX(57).GE.XX(58),CT29;
FT29 FREE,TOOL29,1;
     ACTIVITY,,,XAAB;
XAAG ASSIGN,XX(59)=XX(59) + USERF(4),XX(84)=XX(59)-XX(60);
     ACTIVITY;
T30  AWAIT(40),TOOL30,,1;
```

```
      ACTIVITY/30,USERF(4),XX(59).LT.XX(60);
      ACTIVITY/80,USERF(4)-XX(84),XX(59).GE.XX(60),CT30;
FT30  FREE,TOOL30,1;
      ACTIVITY,,,XAAB;
XAAH  ASSIGN,XX(61)=XX(61) + USERF(4),XX(84)=XX(61)-XX(62);
      ACTIVITY;
T31   AWAIT(41),TOOL31,,1;
      ACTIVITY/31,USERF(4),XX(61).LT.XX(62);
      ACTIVITY/81,USERF(4)-XX(84),XX(61).GE.XX(62),CT31;
FT31  FREE,TOOL31,1;
      ACTIVITY,,,XAAB;
XAAI  ASSIGN,XX(63)=XX(63) + USERF(4),XX(84)=XX(63)-XX(64);
      ACTIVITY;
T32   AWAIT(42),TOOL32,,1;
      ACTIVITY/32,USERF(4),XX(63).LT.XX(64);
      ACTIVITY/82,USERF(4)-XX(84),XX(63).GE.XX(64),CT32;
FT32  FREE,TOOL32,1;
      ACTIVITY,,,XAAB;
M5B   ASSIGN,ATRIB(20)=ATRIB(20) + 1;
      ACTIVITY;
EVE6  EVENT,6,1;
      ACTIVITY,,ATRIB(21).EQ.9;
      ACTIVITY,,ATRIB(21).EQ.10,XAAK;
      ACTIVITY,,ATRIB(21).EQ.11,XAAL;
      ACTIVITY,,ATRIB(21).EQ.12,XAAM;
      ACTIVITY,,ATRIB(21).EQ.0,XAAJ;
      ACTIVITY,,ATRIB(21).EQ.13,XAAN;
      ACTIVITY,,ATRIB(21).EQ.15,XAAO;
      ACTIVITY,,ATRIB(21).EQ.14,XAAP;
      ACTIVITY,,ATRIB(21).EQ.16,XAAQ;
      ASSIGN,XX(65)=XX(65) + USERF(5),XX(85)=XX(65)-XX(66);
      ACTIVITY;
T33   AWAIT(43),TOOL33,,1;
      ACTIVITY/33,USERF(5),XX(65).LT.XX(66);
      ACTIVITY/83,USERF(5)-XX(85),XX(65).GE.XX(66),CT33;
FT33  FREE,TOOL33;
      ACTIVITY;
XAAJ  GOON,1;
      ACTIVITY,,ATRIB(20).GE.8,M5E;
      ACTIVITY,,ATRIB(20).LT.8,M5B;
XAAK  ASSIGN,XX(67)=XX(67) + USERF(5),XX(85)=XX(67)-XX(68);
      ACTIVITY;
T34   AWAIT(44),TOOL34,,1;
      ACTIVITY/34,USERF(5),XX(67).LT.XX(68);
      ACTIVITY/84,USERF(5)-XX(85),XX(67).GE.XX(68),CT34;
FT34  FREE,TOOL34;
      ACTIVITY,,,XAAJ;
XAAL  ASSIGN,XX(69)=XX(69) + USERF(5),XX(85)=XX(69)-XX(70);
      ACTIVITY;
T35   AWAIT(45),TOOL35,,1;
      ACTIVITY/35,USERF(5),XX(69).LT.XX(70);
      ACTIVITY/85,USERF(5)-XX(85),XX(69).GE.XX(70),CT35;
```

```
FT35  FREE,TOOL35;
      ACTIVITY,,,XAAJ;
XAAM  ASSIGN,XX(71)=XX(71) + USERF(5),XX(85)=XX(71)-XX(72);
      ACTIVITY;
T36   AWAIT(46),TOOL36,,1;
      ACTIVITY/36,USERF(5),XX(71).LT.XX(72);
      ACTIVITY/86,USERF(5)-XX(85),XX(71).GE.XX(72),CT36;
FT36  FREE,TOOL36;
      ACTIVITY,,,XAAJ;
XAAN  ASSIGN,XX(73)=XX(73) + USERF(5),XX(85)=XX(73)-XX(74);
      ACTIVITY;
T37   AWAIT(47),TOOL37,,1;
      ACTIVITY/37,USERF(5),XX(73).LT.XX(74);
      ACTIVITY/87,USERF(5)-XX(85),XX(73).GE.XX(74),CT37;
FT37  FREE,TOOL37;
      ACTIVITY,,,XAAJ;
XAAO  ASSIGN,XX(77)=XX(77) + USERF(5),XX(85)=XX(77)-XX(78);
      ACTIVITY;
T39   AWAIT(49),TOOL39,,1;
      ACTIVITY/39,USERF(5),XX(77).LT.XX(78);
      ACTIVITY/89,USERF(5)-XX(85),XX(77).GE.XX(78),CT39;
FT39  FREE,TOOL39;
      ACTIVITY,,,XAAJ;
XAAP  ASSIGN,XX(75)=XX(75) + USERF(5),XX(85)=XX(75)-XX(76);
      ACTIVITY;
T38   AWAIT(48),TOOL38,,1;
      ACTIVITY/38,USERF(5),XX(75).LT.XX(76);
      ACTIVITY/88,USERF(5)-XX(85),XX(75).GE.XX(76),CT38;
FT38  FREE,TOOL38;
      ACTIVITY,,,XAAJ;
XAAQ  ASSIGN,XX(79)=XX(79) + USERF(5),XX(85)=XX(79)-XX(80);
      ACTIVITY;
T40   AWAIT(50),TOOL40,,1;
      ACTIVITY/40,USERF(5),XX(79).LT.XX(80);
      ACTIVITY/90,USERF(5)-XX(85),XX(79).GE.XX(80),CT40;
FT40  FREE,TOOL40;
      ACTIVITY,,,XAAJ;
;FILE GRR3.NET, NODE LABEL SEED
M6B   ASSIGN,ATRIB(22)=ATRIB(22) + 1;
      ACTIVITY;
EV12  EVENT,12,1;
      ACTIVITY,,ATRIB(23).EQ.17;
      ACTIVITY,,ATRIB(23).EQ.19,AON;
      ACTIVITY,,ATRIB(23).EQ.18,WAAC;
      ACTIVITY,,ATRIB(23).EQ.20,AOM;
      ACTIVITY,,ATRIB(23).EQ.0,WAAB;
      ACTIVITY,,ATRIB(23).EQ.23,WAAD;
      ACTIVITY,,ATRIB(23).EQ.24,WAAE;
      ACTIVITY,,ATRIB(23).EQ.22,WAAF;
      ACTIVITY,,ATRIB(23).EQ.21,WAAG;
AOO   ASSIGN,XX(101)=XX(101) + USERF(6),XX(86)=XX(101)-XX(102),1;
      ACTIVITY;
```

```
T41   AWAIT(101),TOOL41,,1;
      ACTIVITY,USERF(6),XX(101).LT.XX(102);
      ACTIVITY,USERF(6)-XX(86),XX(101).GE.XX(102),CT41;
FT41  FREE,TOOL41;
      ACTIVITY;
WAAB  GOON,1;
      ACTIVITY,,ATRIB(22).LT.8,M6B;
      ACTIVITY,,ATRIB(22).GE.8,M6E;
AON   ASSIGN,XX(105)=XX(105) + USERF(6),XX(86)=XX(105)-XX(106),1;
      ACTIVITY;
T43   AWAIT(103),TOOL43,,1;
      ACTIVITY,USERF(6),XX(105).LT.XX(106);
      ACTIVITY,USERF(6)-XX(86),XX(105).GE.XX(106),CT43;
FT43  FREE,TOOL43;
      ACTIVITY,,,WAAB;
WAAC  ASSIGN,XX(103)=XX(103) + USERF(6),XX(86)=XX(103)-XX(104),1;
      ACTIVITY;
T42   AWAIT(102),TOOL42,,1;
      ACTIVITY,USERF(6),XX(103).LT.XX(104);
      ACTIVITY,USERF(6)-XX(86),XX(103).GE.XX(104),CT42;
FT42  FREE,TOOL42;
      ACTIVITY,,,WAAB;
AOM   ASSIGN,XX(107)=XX(107) + USERF(6),XX(86)=XX(107)-XX(108),1;
      ACTIVITY;
T44   AWAIT(104),TOOL44,,1;
      ACTIVITY,USERF(6),XX(107).LT.XX(108);
      ACTIVITY,USERF(6)-XX(86),XX(107).GE.XX(108),CT44;
FT44  FREE,TOOL44;
      ACTIVITY,,,WAAB;
WAAD  ASSIGN,XX(113)=XX(113) + USERF(6),XX(86)=XX(113)-XX(114),1;
      ACTIVITY;
T47   AWAIT(107),TOOL47,,1;
      ACTIVITY,USERF(6),XX(113).LT.XX(114);
      ACTIVITY,USERF(6)-XX(86),XX(113).GE.XX(114),CT47;
FT47  FREE,TOOL47;
      ACTIVITY,,,WAAB;
WAAE  ASSIGN,XX(115)=XX(115) + USERF(6),XX(86)=XX(115)-XX(115),1;
      ACTIVITY;
T48   AWAIT(108),TOOL48,,1;
      ACTIVITY,USERF(6),XX(115).LT.XX(116);
      ACTIVITY,USERF(6)-XX(86),XX(115).GE.XX(116),CT48;
FT48  FREE,TOOL48;
      ACTIVITY,,,WAAB;
WAAF  ASSIGN,XX(111)=XX(111) + USERF(6),XX(86)=XX(111)-XX(112),1;
      ACTIVITY;
T46   AWAIT(106),TOOL46,,1;
      ACTIVITY,USERF(6),XX(111).LT.XX(112);
      ACTIVITY,USERF(6)-XX(86),XX(111).GE.XX(112),CT46;
FT46  FREE,TOOL46;
      ACTIVITY,,,WAAB;
WAAG  ASSIGN,XX(109)=XX(109) + USERF(6),XX(86)=XX(109)-XX(110),1;
      ACTIVITY;
```

```
T45  AWAIT(105),TOOL45,,1;
     ACTIVITY,USERF(6),XX(109).LT.XX(110);
     ACTIVITY,USERF(6)-XX(86),XX(109).GE.XX(110),CT45;
FT45 FREE,TOOL45;
     ACTIVITY,,,WAAB;
M7B  ASSIGN,ATRIB(24)=ATRIB(24) + 1;
     ACTIVITY;
EV13 EVENT,13,1;
     ACTIVITY,,ATRIB(25).EQ.17;
     ACTIVITY,,ATRIB(25).EQ.18,WAAI;
     ACTIVITY,,ATRIB(25).EQ.19,WAAJ;
     ACTIVITY,,ATRIB(25).EQ.20,WAAK;
     ACTIVITY,,ATRIB(25).EQ.0,WAAH;
     ACTIVITY,,ATRIB(25).EQ.21,WAAL;
     ACTIVITY,,ATRIB(25).EQ.23,WAAM;
     ACTIVITY,,ATRIB(25).EQ.22,WAAN;
     ACTIVITY,,ATRIB(25).EQ.24,WAAO;
     ASSIGN,XX(117)=XX(117) + USERF(7),XX(87)=XX(117)-XX(118),1;
     ACTIVITY;
T49  AWAIT(109),TOOL49,,1;
     ACTIVITY,USERF(7),XX(117).LT.XX(118);
     ACTIVITY,USERF(7)-XX(87),XX(117).GE.XX(118),CT49;
FT49 FREE,TOOL49;
     ACTIVITY;
WAAH GOON,1;
     ACTIVITY,,ATRIB(24).LT.8,M7B;
     ACTIVITY,,ATRIB(24).GE.8,M7E;
WAAI ASSIGN,XX(119)=XX(119) + USERF(7),XX(87)=XX(119)-XX(120),1;
     ACTIVITY;
T50  AWAIT(110),TOOL50,,1;
     ACTIVITY,USERF(7),XX(119).LT.XX(120);
     ACTIVITY,USERF(7)-XX(87),XX(119).GE.XX(120),CT50;
FT50 FREE,TOOL50;
     ACTIVITY,,,WAAH;
WAAJ ASSIGN,XX(121)=XX(121) + USERF(7),XX(87)=XX(121)-XX(122),1;
     ACTIVITY;
T51  AWAIT(111),TOOL51,,1;
     ACTIVITY,USERF(7),XX(121).LT.XX(122);
     ACTIVITY,USERF(7)-XX(121),XX(121).GE.XX(122),CT51;
FT51 FREE,TOOL51;
     ACTIVITY,,,WAAH;
WAAK ASSIGN,XX(123)=XX(123) + USERF(7),XX(87)=XX(123)-XX(124),1;
     ACTIVITY;
T52  AWAIT(112),TOOL52,,1;
     ACTIVITY,USERF(7),XX(123).LT.XX(124);
     ACTIVITY,USERF(7)-XX(87),XX(123).GE.XX(124),CT52;
FT52 FREE,TOOL52;
     ACTIVITY,,,WAAH;
WAAL ASSIGN,XX(125)=XX(125) + USERF(7),XX(87)=XX(125)-XX(126),1;
     ACTIVITY;
T53  AWAIT(113),TOOL53,,1;
     ACTIVITY,USERF(7),XX(125).LT.XX(126);
```

```
     ACTIVITY,USERF(7)-XX(87),XX(125).GE.XX(126),CT53;
FT53 FREE,TOOL53;
     ACTIVITY,,,WAAH;
WAAM ASSIGN,XX(129)=XX(129) + USERF(7),XX(87)=XX(129)-XX(130),1;
     ACTIVITY;
T55  AWAIT(115),TOOL55,,1;
     ACTIVITY,USERF(7),XX(129).LT.XX(130);
     ACTIVITY,USERF(7)-XX(87),XX(129).GE.XX(130),CT55;
FT55 FREE,TOOL55;
     ACTIVITY,,,WAAH;
WAAN ASSIGN,XX(127)=XX(127) + USERF(7),XX(87)=XX(127)-XX(128),1;
     ACTIVITY;
T54  AWAIT(114),TOOL54,,1;
     ACTIVITY,USERF(7),XX(127).LT.XX(128);
     ACTIVITY,USERF(7)-XX(87),XX(127).GE.XX(128),CT54;
FT54 FREE,TOOL54;
     ACTIVITY,,,WAAH;
WAAO ASSIGN,XX(131)=XX(131) + USERF(7),XX(87)=XX(131)-XX(132),1;
     ACTIVITY;
T56  AWAIT(116),TOOL56,,1;
     ACTIVITY,USERF(7),XX(131).LT.XX(132);
     ACTIVITY,USERF(7)-XX(87),XX(131).GE.XX(132),CT56;
FT56 FREE,TOOL56;
     ACTIVITY,,,WAAH;
M8B  ASSIGN,ATRIB(26)=ATRIB(26) + 1,1;
     ACTIVITY;
EV14 EVENT,14,1;
     ACTIVITY,,ATRIB(27).EQ.17;
     ACTIVITY,,ATRIB(27).EQ.18,WAAQ;
     ACTIVITY,,ATRIB(27).EQ.19,WAAR;
     ACTIVITY,,ATRIB(27).EQ.20,WAAS;
     ACTIVITY,,ATRIB(27).EQ.0,WAAP;
     ACTIVITY,,ATRIB(27).EQ.21,WAAT;
     ACTIVITY,,ATRIB(27).EQ.23,WAAU;
     ACTIVITY,,ATRIB(27).EQ.22,WAAV;
     ACTIVITY,,ATRIB(27).EQ.24,WAAW;
     ASSIGN,XX(133)=XX(133) + USERF(8),XX(88)=XX(133)-XX(134),1;
     ACTIVITY;
T57  AWAIT(117),TOOL57,,1;
     ACTIVITY,USERF(8),XX(133).LT.XX(134);
     ACTIVITY,USERF(7)-XX(88),XX(133).GE.XX(134),CT57;
FT57 FREE,TOOL57;
     ACTIVITY;
WAAP GOON,1;
     ACTIVITY,,ATRIB(26).LT.8,M8B;
     ACTIVITY,,ATRIB(26).GE.8,M8E;
WAAQ ASSIGN,XX(135)=XX(135) + USERF(8),XX(88)=XX(135)-XX(136),1;
     ACTIVITY;
T58  AWAIT(118),TOOL58,,1;
     ACTIVITY,USERF(8),XX(135).LT.XX(136);
     ACTIVITY,USERF(8)-XX(88),XX(135).GE.XX(136),CT58;
FT58 FREE,TOOL58;
```

```
       ACTIVITY,,,WAAP;
WAAR  ASSIGN,XX(137)=XX(137) + USERF(8),XX(88)=XX(137)-XX(138),1;
       ACTIVITY;
T59   AWAIT(119),TOOL59,,1;
       ACTIVITY,USERF(8),XX(137).LT.XX(138);
       ACTIVITY,USERF(8)-XX(88),XX(137).GE.XX(138),CT59;
FT59  FREE,TOOL59;
       ACTIVITY,,,WAAP;
WAAS  ASSIGN,XX(139)=XX(139) + USERF(8),XX(88)=XX(139)-XX(140),1;
       ACTIVITY;
T60   AWAIT(120),TOOL60,,1;
       ACTIVITY,USERF(8),XX(139).LT.XX(140);
       ACTIVITY,USERF(8)-XX(88),XX(139).GE.XX(140),CT60;
FT60  FREE,TOOL60;
       ACTIVITY,,,WAAP;
WAAT  ASSIGN,XX(141)=XX(141) + USERF(8),XX(88)=XX(141)-XX(142),1;
       ACTIVITY;
T61   AWAIT(121),TOOL61,,1;
       ACTIVITY,USERF(8),XX(141).LT.XX(142);
       ACTIVITY,USERF(8)-XX(88),XX(141).GE.XX(142),CT61;
FT61  FREE,TOOL61;
       ACTIVITY,,,WAAP;
WAAU  ASSIGN,XX(145)=XX(145) + USERF(8),XX(88)=XX(145)-XX(146),1;
       ACTIVITY;
T63   AWAIT(123),TOOL63,,1;
       ACTIVITY,USERF(8),XX(145).LT.XX(146);
       ACTIVITY,USERF(8)-XX(88),XX(145).GE.XX(146),CT63;
FT63  FREE,TOOL63;
       ACTIVITY,,,WAAP;
WAAV  ASSIGN,XX(143)=XX(143) + USERF(8),XX(88)=XX(143)-XX(144),1;
       ACTIVITY;
T62   AWAIT(122),TOOL62,,1;
       ACTIVITY,USERF(8),XX(143).LT.XX(144);
       ACTIVITY,USERF(8)-XX(88),XX(143).GE.XX(144),CT62;
FT62  FREE,TOOL62;
       ACTIVITY,,,WAAP;
WAAW  ASSIGN,XX(147)=XX(147) + USERF(8),XX(88)=XX(147)-XX(148),1;
       ACTIVITY;
T64   AWAIT(124),TOOL64,,1;
       ACTIVITY,USERF(8),XX(147).LT.XX(148);
       ACTIVITY,USERF(8)-XX(88),XX(147).GE.XX(148),CT64;
FT64  FREE,TOOL64;
       ACTIVITY,,,WAAP;
;FILE RPERREP.NET, NODE LABEL SEED ZAAA
;
ZACA  ASSIGN,XX(84)=XX(84)-XX(54);
       ACTIVITY;
CT27  ALTER,TOOL27,-1,2;
       ACTIVITY;
       ACTIVITY,,NNRSC(32).LT.1,EV16;
TS27  ASSIGN,XX(54)=EXPON(20,1),XX(53)=XX(84),1;
       ACTIVITY,XX(54),XX(54).LE.XX(84),ZACA;
```

```
     ACTIVITY,XX(84),XX(54).GT.XX(84),FT27;
;
ZACB ASSIGN,XX(84)=XX(84)-XX(56);
     ACTIVITY;
CT28 ALTER,TOOL28,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(33).LT.1,EV16;
TS28 ASSIGN,XX(56)=EXPON(20,1),XX(55)=XX(84),1;
     ACTIVITY,XX(56),XX(56).LE.XX(84),ZACB;
     ACTIVITY,XX(84),XX(56).GT.XX(84),FT28;
;
ZACC ASSIGN,XX(84)=XX(84)-XX(58);
     ACTIVITY;
CT29 ALTER,TOOL29,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(34).LT.1,EV16;
TS29 ASSIGN,XX(58)=EXPON(20,1),XX(57)=XX(84),1;
     ACTIVITY,XX(58),XX(58).LE.XX(84),ZACC;
     ACTIVITY,XX(84),XX(58).GT.XX(84),FT29;
;
ZACD ASSIGN,XX(84)=XX(84)-XX(60);
     ACTIVITY;
CT30 ALTER,TOOL30,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(35).LT.1,EV16;
TS30 ASSIGN,XX(60)=EXPON(20,1),XX(59)=XX(84),1;
     ACTIVITY,XX(60),XX(60).LE.XX(84),ZACD;
     ACTIVITY,XX(84),XX(60).GT.XX(84),FT30;
;
ZACE ASSIGN,XX(84)=XX(84)-XX(62);
     ACTIVITY;
CT31 ALTER,TOOL31,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(36).LT.1,EV16;
TS31 ASSIGN,XX(62)=EXPON(20,1),XX(61)=XX(84),1;
     ACTIVITY,XX(62),XX(62).LE.XX(84),ZACE;
     ACTIVITY,XX(84),XX(62).GT.XX(84),FT31;
;
ZACF ASSIGN,XX(84)=XX(84)-XX(64);
     ACTIVITY;
CT32 ALTER,TOOL32,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(37).LT.1,EV16;
TS32 ASSIGN,XX(64)=EXPON(20,1),XX(63)=XX(84),1;
     ACTIVITY,XX(64),XX(64).LE.XX(84),ZACF;
     ACTIVITY,XX(84),XX(64).GT.XX(84),FT32;
;
ZACG ASSIGN,XX(85)=XX(85)-XX(66);
     ACTIVITY;
CT33 ALTER,TOOL33,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(38).LT.1,EV16;
```

```
TS33  ASSIGN,XX(66)=EXPON(20,1),XX(65)=XX(85),1;
      ACTIVITY,XX(66),XX(66).LE.XX(85),ZACG;
      ACTIVITY,XX(85),XX(66).GT.XX(85),FT33;
;
ZACH  ASSIGN,XX(85)=XX(85)-XX(68);
      ACTIVITY;
CT34  ALTER,TOOL34,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(39).LT.1,EV16;
TS34  ASSIGN,XX(68)=EXPON(20,1),XX(67)=XX(85),1;
      ACTIVITY,XX(68),XX(68).LE.XX(85),ZACH;
      ACTIVITY,XX(85),XX(68).GT.XX(85),FT34;
;
ZACI  ASSIGN,XX(85)=XX(85)-XX(70);
      ACTIVITY;
CT35  ALTER,TOOL35,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(40).LT.1,EV16;
TS35  ASSIGN,XX(70)=EXPON(20,1),XX(69)=XX(85),1;
      ACTIVITY,XX(70),XX(70).LE.XX(85),ZACI;
      ACTIVITY,XX(85),XX(70).GT.XX(85),FT35;
;
ZACJ  ASSIGN,XX(85)=XX(85)-XX(72);
      ACTIVITY;
CT36  ALTER,TOOL36,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(41).LT.1,EV16;
TS36  ASSIGN,XX(72)=EXPON(20,1),XX(71)=XX(85),1;
      ACTIVITY,XX(72),XX(72).LE.XX(85),ZACJ;
      ACTIVITY,XX(85),XX(72).GT.XX(85),FT36;
;
ZACK  ASSIGN,XX(85)=XX(85)-XX(74);
      ACTIVITY;
CT37  ALTER,TOOL37,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(42).LT.1,EV16;
TS37  ASSIGN,XX(74)=EXPON(20,1),XX(73)=XX(85),1;
      ACTIVITY,XX(74),XX(74).LE.XX(85),ZACK;
      ACTIVITY,XX(85),XX(74).GT.XX(85),FT37;
;
ZACL  ASSIGN,XX(85)=XX(85)-XX(76);
      ACTIVITY;
CT38  ALTER,TOOL38,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(43).LT.1,EV16;
TS38  ASSIGN,XX(76)=EXPON(20,1),XX(75)=XX(85),1;
      ACTIVITY,XX(76),XX(76).LE.XX(85),ZACL;
      ACTIVITY,XX(85),XX(76).GT.XX(85),FT38;
;
ZACM  ASSIGN,XX(85)=XX(85)-XX(78);
      ACTIVITY;
CT39  ALTER,TOOL39,-1,2;
```

```
     ACTIVITY;
     ACTIVITY,,NNRSC(44).LT.1,EV16;
TS39 ASSIGN,XX(78)=EXPON(20,1),XX(77)=XX(85),1;
     ACTIVITY,XX(78),XX(78).LE.XX(85),ZACM;
     ACTIVITY,XX(85),XX(78).GT.XX(85),FT39;
;
ZACN ASSIGN,XX(85)=XX(85)-XX(80);
     ACTIVITY;
CT40 ALTER,TOOL40,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(45).LT.1,EV16;
TS40 ASSIGN,XX(80)=EXPON(20,1),XX(79)=XX(85),1;
     ACTIVITY,XX(80),XX(80).LE.XX(85),ZACN;
     ACTIVITY,XX(85),XX(80).GT.XX(85),FT40;
;
ZACO ASSIGN,XX(86)=XX(86)-XX(102);
     ACTIVITY;
CT41 ALTER,TOOL41,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(101).LT.1,EV17;
TS41 ASSIGN,XX(102)=EXPON(20,1),XX(101)=XX(86),1;
     ACTIVITY,XX(102),XX(102).LE.XX(86),ZACO;
     ACTIVITY,XX(86),XX(102).GT.XX(86),FT41;
;
G1   GOON;
     ACTIVITY;
ASOP ASSIGN,ATRIB(26)=0,ATRIB(7)=ATRIB(7)+1;
     ACTIVITY/100;
Q1   QUEUE(1),,,;
     ACTIVITY(1);
EVE1 EVENT,1;
     ACTIVITY,EXPON(5),ATRIB(11).EQ.1.OR.ATRIB(11).EQ.2.OR.ATRIB(11).EQ.3;
     ACTIVITY,,ATRIB(11).EQ.4,ZAAN;
LOA  AWAIT(61),LOADER;
     ACTIVITY;
AGVL AWAIT(62),AGVL;
     ACTIVITY,UNFRM(1,2);
FLOA FREE,LOADER;
     ACTIVITY;
TRACK AWAIT(63),TRACK;
     ACTIVITY,,ATRIB(11).EQ.1;
     ACTIVITY/101,,ATRIB(11).EQ.3,EVE9;
     ACTIVITY,,ATRIB(11).EQ.2,EVE8;
EVE7 EVENT,7,1;
     ACTIVITY/110,,ATRIB(28).EQ.1;
     ACTIVITY/111,,ATRIB(28).EQ.2,ZAAC;
     ACTIVITY/112,,ATRIB(28).EQ.3,ZAAE;
     GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,FAGV;
L1ROB AWAIT(75),L1ROB;
     ACTIVITY,UNFRM(1,2);
```

```
     FREE,L1ROB;
     ACTIVITY;
MAC1 AWAIT(2),MACHIN1,,1;
     ACTIVITY,,,M1B;
FAGVL FREE,AGVL;
     ACTIVITY;
FTRA FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAC GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAB;
     AWAIT(76),L2ROB;
     ACTIVITY,UNFRM(1,2);
     FREE,L2ROB;
     ACTIVITY;
MAC2 AWAIT(3),MACHIN2;
     ACTIVITY,,,M2B;
ZAAB FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAE GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAD;
     AWAIT(77),L3ROB;
     ACTIVITY,UNFRM(1,2);
     FREE,L3ROB;
     ACTIVITY;
MAC3 AWAIT(4),MACHIN3;
     ACTIVITY,,,M3B;
ZAAD FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
EVE9 EVENT,11,1;
     ACTIVITY/102,,ATRIB(28).EQ.6;
     ACTIVITY/103,,ATRIB(28).EQ.7,ZAAH;
     ACTIVITY/104,,ATRIB(28).EQ.8,ZAAJ;
     GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAF;
ZAFB AWAIT(78),L6ROB,,1;
     ACTIVITY,UNFRM(1,2);
     FREE,L6ROB,1;
     ACTIVITY;
ZAFA AWAIT(81),MACHIN6,,1;
     ACTIVITY,,,M6B;
ZAAF FREE,AGVL;
     ACTIVITY;
```

```
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAH GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAG;
     AWAIT(79),L7ROB,,1;
     ACTIVITY,UNFRM(1,2);
     FREE,L7ROB,1;
     ACTIVITY;
ZAFC AWAIT(82),MACHIN7,,1;
     ACTIVITY,,,M7B;
ZAAG FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAJ GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAI;
ZAFF AWAIT(80),L8ROB,,1;
     ACTIVITY,UNFRM(1,2);
     FREE,L8ROB,1;
     ACTIVITY;
ZAFE AWAIT(83),MACHIN8,,1;
     ACTIVITY,,,M8B;
ZAAI FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
EVE8 EVENT,8,1;
     ACTIVITY/113,,ATRIB(28).EQ.4;
     ACTIVITY/114,,ATRIB(28).EQ.5,ZAAM;
     GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAK;
     AWAIT(72),L4ROB;
     ACTIVITY,UNFRM(1,2);
     FREE,L4ROB,1;
     ACTIVITY;
MAC4 AWAIT(5),MACHIN4,,1;
     ACTIVITY,,,M4B;
ZAAK FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAM GOON;
     ACTIVITY,UNFRM(2,2.5);
     ACTIVITY,UNFRM(4,5),,ZAAL;
     AWAIT(73),L5ROB,,1;
```

```
     ACTIVITY,UNFRM(1,2);
     FREE,L5ROB,1;
     ACTIVITY;
MAC5 AWAIT(6),MACHIN5,,1;
     ACTIVITY,,,M5B;
ZAAL FREE,AGVL;
     ACTIVITY;
     FREE,TRACK;
     ACTIVITY;
     TERMINATE;
ZAAN GOON;
     ACTIVITY,,ATRIB(7).LT.7,ASOP;
     ACTIVITY/200,,ATRIB(7).GE.7;
     COLCT,ALL;
     ACTIVITY;
     TERMINATE;
;
ZACP ASSIGN,XX(86)=XX(86)-XX(104);
     ACTIVITY;
CT42 ALTER,TOOL42,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(102).LT.1,EV17;
TS42 ASSIGN,XX(104)=EXPON(20,1),XX(103)=XX(86),1;
     ACTIVITY,XX(104),XX(104).LE.XX(86),ZACP;
     ACTIVITY,XX(86),XX(104).GT.XX(86),FT42;
;
ZACQ ASSIGN,XX(86)=XX(86)-XX(106);
     ACTIVITY;
CT43 ALTER,TOOL43,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(103).LT.1,EV17;
TS43 ASSIGN,XX(106)=EXPON(20,1),XX(105)=XX(86),1;
     ACTIVITY,XX(106),XX(106).LE.XX(86),ZACQ;
     ACTIVITY,XX(86),XX(106).GT.XX(86),FT43;
;
ZACR ASSIGN,XX(86)=XX(86)-XX(108);
     ACTIVITY;
CT44 ALTER,TOOL44,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(104).LT.1,EV17;
TS44 ASSIGN,XX(108)=EXPON(20,1),XX(107)=XX(86),1;
     ACTIVITY,XX(108),XX(108).LE.XX(86),ZACR;
     ACTIVITY,XX(86),XX(108).GT.XX(86),FT44;
;
ZACS ASSIGN,XX(86)=XX(86)-XX(110);
     ACTIVITY;
CT45 ALTER,TOOL45,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(105).LT.1,EV17;
TS45 ASSIGN,XX(110)=EXPON(20,1),XX(109)=XX(86),1;
     ACTIVITY,XX(110),XX(110).LE.XX(86),ZACS;
     ACTIVITY,XX(86),XX(110).GT.XX(86),FT45;
```

```
;
ZACT  ASSIGN,XX(86)=XX(86)-XX(112);
      ACTIVITY;
CT46  ALTER,TOOL46,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(106).LT.1,EV17;
TS46  ASSIGN,XX(112)=EXPON(20,1),XX(111)=XX(86),1;
      ACTIVITY,XX(112),XX(112).LE.XX(86),ZACT;
      ACTIVITY,XX(86),XX(112).GT.XX(86),FT46;
;
ZACU  ASSIGN,XX(86)=XX(86)-XX(114);
      ACTIVITY;
CT47  ALTER,TOOL47,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(107).LT.1,EV17;
TS47  ASSIGN,XX(114)=EXPON(20,1),XX(113)=XX(86),1;
      ACTIVITY,XX(114),XX(114).LE.XX(86),ZACU;
      ACTIVITY,XX(86),XX(114).GT.XX(86),FT47;
;
M1E   ASSIGN,ATRIB(12)=0;
      ACTIVITY;
FM1   FREE,MACHIN1;
      ACTIVITY;
ZAAP  QUEUE(71),,,,ZAAO ;
ZAAO  SELECT,,,,ZAAP,ZAAQ,ZAAR,ZAAS,ZAAT,ZAAU,ZAAV,ZAAW;
      ACTIVITY(1);
      AWAIT(64),ULOADER;
      ACTIVITY;
      AWAIT(65),AGVU;
      ACTIVITY,UNFRM(1,2);
      FREE,ULOADER;
      ACTIVITY;
      AWAIT(66),UTRACK;
      ACTIVITY;
      GOON,2;
      ACTIVITY,UNFRM(2,2.5),,ZAAN;
      ACTIVITY,UNFRM(4,5);
      FREE,UTRACK;
      ACTIVITY;
      FREE,AGVU;
      ACTIVITY;
      TERMINATE;
;
ZAAQ  QUEUE(70),,,,ZAAO ;
;
ZAAR  QUEUE(69),,,,ZAAO ;
;
ZACV  ASSIGN,XX(86)=XX(86)-XX(116);
      ACTIVITY;
CT48  ALTER,TOOL48,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(108).LT.1,EV17;
```

```
TS48  ASSIGN,XX(116)=EXPON(20,1),XX(115)=XX(86),1;
      ACTIVITY,XX(116),XX(116).LE.XX(86),ZACV;
      ACTIVITY,XX(86),XX(116).GT.XX(86),FT48;
;
ZAAS  QUEUE(68),,,,ZAAO ;
;
ZAAT  QUEUE(67),,,,ZAAO ;
;
ZAAU  QUEUE(86),,,,ZAAO ;
;
ZACW  ASSIGN,XX(87)=XX(87)-XX(120);
      ACTIVITY;
CT50  ALTER,TOOL50,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(110).LT.1,EV17;
TS50  ASSIGN,XX(120)=EXPON(20,1),XX(119)=XX(87),1;
      ACTIVITY,XX(120),XX(120).LE.XX(87),ZACW;
      ACTIVITY,XX(87),XX(120).GT.XX(87),FT50;
;
ZAAV  QUEUE(85),,,,ZAAO ;
;
ZAAW  QUEUE(84),,,,ZAAO ;
;
ZACX  ASSIGN,XX(87)=XX(87)-XX(122);
      ACTIVITY;
CT51  ALTER,TOOL51,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(111).LT.1,EV17;
TS51  ASSIGN,XX(122)=EXPON(20,1),XX(121)=XX(87),1;
      ACTIVITY,XX(122),XX(122).LE.XX(87),ZACX;
      ACTIVITY,XX(87),XX(122).GT.XX(87),FT51;
;
M2E   ASSIGN,ATRIB(14)=0;
      ACTIVITY;
FM2   FREE,MACHIN2;
      ACTIVITY,,,ZAAQ;
;
M3E   ASSIGN,ATRIB(16)=0;
      ACTIVITY;
FM3   FREE,MACHIN3;
      ACTIVITY,,,ZAAR;
;
M4E   ASSIGN,ATRIB(19)=0;
      ACTIVITY;
FM4   FREE,MACHIN4;
      ACTIVITY,,,ZAAS;
;
ZACY  ASSIGN,XX(87)=XX(87)-XX(124);
      ACTIVITY;
CT52  ALTER,TOOL52,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(112).LT.1,EV17;
```

```
TS52  ASSIGN,XX(124)=EXPON(20,1),XX(123)=XX(87),1;
      ACTIVITY,XX(124),XX(124).LE.XX(87),ZACY;
      ACTIVITY,XX(87),XX(124).GT.XX(87),FT52;
;
M5E   ASSIGN,ATRIB(20)=0;
      ACTIVITY;
FM5   FREE,MACHIN5;
      ACTIVITY,,,ZAAT;
;
M6E   ASSIGN,ATRIB(22)=0;
      ACTIVITY;
ZAFG  FREE,MACHIN6;
      ACTIVITY,,,ZAAW;
;
M7E   ASSIGN,ATRIB(24)=0;
      ACTIVITY;
ZAFH  FREE,MACHIN7;
      ACTIVITY,,,ZAAV;
;
ZACZ  ASSIGN,XX(87)=XX(87)-XX(126);
      ACTIVITY;
CT53  ALTER,TOOL53,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(113).LT.1,EV17;
TS53  ASSIGN,XX(126)=EXPON(20,1),XX(125)=XX(87),1;
      ACTIVITY,XX(126),XX(126).LE.XX(87),ZACZ;
      ACTIVITY,XX(87),XX(126).GT.XX(87),FT53;
;
M8E   ASSIGN,ATRIB(26)=0;
      ACTIVITY;
ZAFI  FREE,MACHIN8;
      ACTIVITY,,,ZAAU;
;
      CREATE,,,,1,2;
      ACTIVITY;
      ACTIVITY/500,,,ZAAY;
ZAAX  EVENT,18,1;
      ACTIVITY,1,,ZAAX;
ZAAY  EVENT,9,1;
      ACTIVITY/400,XX(181),,ZAAY;
;
ZADA  ASSIGN,XX(87)=XX(87)-XX(118);
      ACTIVITY;
CT49  ALTER,TOOL49,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(109).LT.1,EV17;
TS49  ASSIGN,XX(118)=EXPON(20,1),XX(117)=XX(87),1;
      ACTIVITY,XX(118),XX(118).LE.XX(87),ZADA;
      ACTIVITY,XX(87),XX(118).GT.XX(87),FT49;
;
ZADB  ASSIGN,XX(87)=XX(87)-XX(130);
      ACTIVITY;
```

```
CT55  ALTER,TOOL55,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(115).LT.1,EV17;
TS55  ASSIGN,XX(130)=EXPON(20,1),XX(129)=XX(87),1;
      ACTIVITY,XX(130),XX(130).LE.XX(87),ZADB;
      ACTIVITY,XX(87),XX(130).GT.XX(87),FT55;
;
ZABA  ASSIGN,XX(81)=XX(81)-XX(2);
      ACTIVITY;
ALT1  ALTER,TOOL1,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(6).LT.1,EV15;
TS1   ASSIGN,XX(2)=EXPON(20,1),XX(1)=XX(81),1;
      ACTIVITY,XX(2),XX(2).LE.XX(81),ZABA;
      ACTIVITY,XX(81),XX(2).GT.XX(81),FT1;
;
ZADC  ASSIGN,XX(87)=XX(87)-XX(132);
      ACTIVITY;
CT56  ALTER,TOOL56,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(116).LT.1,EV17;
TS56  ASSIGN,XX(132)=EXPON(20,1),XX(131)=XX(87),1;
      ACTIVITY,XX(132),XX(132).LE.XX(87),ZADC;
      ACTIVITY,XX(87),XX(132).GT.XX(87),FT56;
;
ZABB  ASSIGN,XX(81)=XX(81)-XX(4);
      ACTIVITY;
ALT2  ALTER,TOOL2,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(7).LT.1,EV15;
TS2   ASSIGN,XX(4)=EXPON(20,1),XX(3)=XX(81),1;
      ACTIVITY,XX(4),XX(4).LE.XX(81),ZABB;
      ACTIVITY,XX(81),XX(4).GT.XX(81),FT2;
;
ZADD  ASSIGN,XX(87)=XX(87)-XX(128);
      ACTIVITY;
CT54  ALTER,TOOL54,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(14).LT.1,EV17;
TS54  ASSIGN,XX(128)=EXPON(20,1),XX(127)=XX(87),1;
      ACTIVITY,XX(128),XX(128).LE.XX(87),ZADD;
      ACTIVITY,XX(87),XX(128).GT.XX(87),FT54;
;
ZABC  ASSIGN,XX(81)=XX(81)-XX(6);
      ACTIVITY;
ALT3  ALTER,TOOL3,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(8).LT.1,EV15;
TS3   ASSIGN,XX(6)=EXPON(20,1),XX(5)=XX(81),1;
      ACTIVITY,XX(6),XX(6).LE.XX(81),ZABC;
      ACTIVITY,XX(81),XX(6).GT.XX(81),FT3;
;
```

```
ZADE  ASSIGN,XX(88)=XX(88)-XX(134);
      ACTIVITY;
CT57  ALTER,TOOL57,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(117).LT.1,EV17;
TS57  ASSIGN,XX(134)=EXPON(20,1),XX(133)=XX(88),1;
      ACTIVITY,XX(134),XX(134).LE.XX(88),ZADE;
      ACTIVITY,XX(88),XX(134).GT.XX(88),FT57;
;
ZABD  ASSIGN,XX(81)=XX(81)-XX(8);
      ACTIVITY;
ALT4  ALTER,TOOL4,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(9).LT.1,EV15;
TS4   ASSIGN,XX(8)=EXPON(20,1),XX(7)=XX(81),1;
      ACTIVITY,XX(8),XX(8).LE.XX(81),ZABD;
      ACTIVITY,XX(81),XX(8).GT.XX(81),FT4;
;
ZADF  ASSIGN,XX(88)=XX(88)-XX(136);
      ACTIVITY;
CT58  ALTER,TOOL58,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(118).LT.1,EV17;
TS58  ASSIGN,XX(136)=EXPON(20,1),XX(135)=XX(88),1;
      ACTIVITY,XX(136),XX(136).LE.XX(88),ZADF;
      ACTIVITY,XX(88),XX(136).GT.XX(88),FT58;
;
ZABE  ASSIGN,XX(81)=XX(81)-XX(10);
      ACTIVITY;
ALT5  ALTER,TOOL5,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(10).LT.1,EV15;
TS5   ASSIGN,XX(10)=EXPON(20,1),XX(9)=XX(81),1;
      ACTIVITY,XX(10),XX(10).LE.XX(81),ZABE;
      ACTIVITY,XX(81),XX(10).GT.XX(81),FT5;
;
ZADG  ASSIGN,XX(88)=XX(88)-XX(138);
      ACTIVITY;
CT59  ALTER,TOOL59,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(119).LT.1,EV17;
TS59  ASSIGN,XX(138)=EXPON(20,1),XX(137)=XX(88),1;
      ACTIVITY,XX(138),XX(138).LE.XX(88),ZADG;
      ACTIVITY,XX(88),XX(138).GT.XX(88),FT59;
;
ZABF  ASSIGN,XX(81)=XX(81)-XX(12);
      ACTIVITY;
ALT6  ALTER,TOOL6,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(11).LT.1,EV15;
TS6   ASSIGN,XX(12)=EXPON(20,1),XX(11)=XX(81),1;
      ACTIVITY,XX(12),XX(12).LE.XX(81),ZABF;
```

```
     ACTIVITY,XX(81),XX(12).GT.XX(81),FT6;
;
ZADH  ASSIGN,XX(88)=XX(88)-XX(140);
     ACTIVITY;
CT60  ALTER,TOOL60,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(120).LT.1,EV17;
TS60  ASSIGN,XX(140)=EXPON(20,1),XX(139)=XX(88),1;
     ACTIVITY,XX(140),XX(140).LE.XX(88),ZADH;
     ACTIVITY,XX(88),XX(140).GT.XX(88),FT60;
;
ZABG  ASSIGN,XX(81)=XX(81)-XX(14);
     ACTIVITY;
ALT7  ALTER,TOOL7,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(12).LT.1,EV15;
TS7   ASSIGN,XX(14)=EXPON(20,1),XX(13)=XX(81),1;
     ACTIVITY,XX(14),XX(14).LE.XX(81),ZABG;
     ACTIVITY,XX(81),XX(14).GT.XX(81),FT7;
;
ZADI  ASSIGN,XX(88)=XX(88)-XX(142);
     ACTIVITY;
CT61  ALTER,TOOL61,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(121).LT.1,EV17;
TS61  ASSIGN,XX(142)=EXPON(20,1),XX(141)=XX(88),1;
     ACTIVITY,XX(142),XX(142).LE.XX(88),ZADI;
     ACTIVITY,XX(88),XX(142).GT.XX(88),FT61;
;
ZABH  ASSIGN,XX(81)=XX(81)-XX(16);
     ACTIVITY;
ALT8  ALTER,TOOL8,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(13).LT.1,EV15;
TS8   ASSIGN,XX(16)=EXPON(20,1),XX(15)=XX(81),1;
     ACTIVITY,XX(16),XX(16).LE.XX(81),ZABH;
     ACTIVITY,XX(81),XX(16).GT.XX(81),FT8;
;
ZADJ  ASSIGN,XX(88)=XX(88)-XX(144);
     ACTIVITY;
CT62  ALTER,TOOL62,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(122).LT.1,EV17;
TS62  ASSIGN,XX(144)=EXPON(20,1),XX(143)=XX(88),1;
     ACTIVITY,XX(144),XX(144).LE.XX(88),ZADJ;
     ACTIVITY,XX(144),XX(144).GT.XX(88),FT62;
;
ZABI  ASSIGN,XX(82)=XX(82)-XX(18);
     ACTIVITY;
CT9   ALTER,TOOL9,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(14).LT.1,EV15;
```

```
TS9  ASSIGN,XX(18)=EXPON(20,1),XX(17)=XX(82),1;
     ACTIVITY,XX(18),XX(18).LE.XX(82),ZABI;
     ACTIVITY,XX(82),XX(18).GT.XX(82),FT9;
;
ZADK  ASSIGN,XX(88)=XX(88)-XX(146);
     ACTIVITY;
CT63  ALTER,TOOL63,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(123).LT.1,EV17;
TS63  ASSIGN,XX(146)=EXPON(20,1),XX(145)=XX(88),1;
     ACTIVITY,XX(146),XX(146).LE.XX(88),ZADK;
     ACTIVITY,XX(88),XX(146).GT.XX(88),FT63;
;
ZABJ  ASSIGN,XX(82)=XX(82)-XX(20);
     ACTIVITY;
CT10  ALTER,TOOL10,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(15).LT.1,EV15;
TS10  ASSIGN,XX(20)=EXPON(20,1),XX(19)=XX(82),1;
     ACTIVITY,XX(20),XX(20).LE.XX(82),ZABJ;
     ACTIVITY,XX(82),XX(20).GT.XX(82),FT10;
;
ZADL  ASSIGN,XX(88)=XX(88)-XX(148);
     ACTIVITY;
CT64  ALTER,TOOL64,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(124).LT.1,EV17;
TS64  ASSIGN,XX(148)=EXPON(20,1),XX(147)=XX(88),1;
     ACTIVITY,XX(148),XX(148).LE.XX(88),ZADL;
     ACTIVITY,XX(88),XX(148).GT.XX(88),FT64;
;
ZABK  ASSIGN,XX(82)=XX(82)-XX(22);
     ACTIVITY;
CT11  ALTER,TOOL11,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(16).LT.1,EV15;
TS11  ASSIGN,XX(22)=EXPON(20,1),XX(21)=XX(82),1;
     ACTIVITY,XX(22),XX(22).LE.XX(82),ZABK;
     ACTIVITY,XX(82),XX(22).GT.XX(82),FT11;
;
ZABL  ASSIGN,XX(82)=XX(82)-XX(24);
     ACTIVITY;
CT12  ALTER,TOOL12,-1,2;
     ACTIVITY;
     ACTIVITY,,NNRSC(17).LT.1,EV15;
TS12  ASSIGN,XX(24)=EXPON(20,1),XX(23)=XX(82),1;
     ACTIVITY,XX(24),XX(24).LE.XX(82),ZABL;
     ACTIVITY,XX(82),XX(24).GT.XX(82),FT12;
;
ZABM  ASSIGN,XX(82)=XX(82)-XX(26);
     ACTIVITY;
CT13  ALTER,TOOL13,-1,2;
```

```
      ACTIVITY;
      ACTIVITY,,NNRSC(18).LT.1,EV15;
TS13  ASSIGN,XX(26)=EXPON(20,1),XX(25)=XX(82),1;
      ACTIVITY,XX(26),XX(26).LE.XX(82),ZABM;
      ACTIVITY,XX(82),XX(26).GT.XX(82),FT13;
;
ZABN  ASSIGN,XX(82)=XX(82)-XX(28);
      ACTIVITY;
CT14  ALTER,TOOL14,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(19).LT.1,EV15;
TS14  ASSIGN,XX(28)=EXPON(20,1),XX(27)=XX(82),1;
      ACTIVITY,XX(28),XX(28).LE.XX(82),ZABN;
      ACTIVITY,XX(82),XX(28).GT.XX(82),FT14;
;
ZABO  ASSIGN,XX(82)=XX(82)-XX(30);
      ACTIVITY;
CT15  ALTER,TOOL15,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(20).LT.1,EV15;
TS15  ASSIGN,XX(30)=EXPON(20,1),XX(29)=XX(81),1;
      ACTIVITY,XX(30),XX(30).LE.XX(82),ZABO;
      ACTIVITY,XX(82),XX(30).GT.XX(82),FT15;
;
ZABP  ASSIGN,XX(82)=XX(82)-XX(32);
      ACTIVITY;
CT16  ALTER,TOOL16,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(21).LT.1,EV15;
TS16  ASSIGN,XX(32)=EXPON(20,1),XX(31)=XX(81),1;
      ACTIVITY,XX(32),XX(32).LE.XX(82),ZABP;
      ACTIVITY,XX(82),XX(32).GT.XX(82),FT16;
;
ZABQ  ASSIGN,XX(83)=XX(83)-XX(34);
      ACTIVITY;
CT17  ALTER,TOOL17,-1,2;
      ACTIVITY,,,N17;
      ACTIVITY,,NNRSC(22).LT.1,EV15;
N17   ASSIGN,XX(34)=EXPON(20,1),XX(33)=XX(83),1;
      ACTIVITY,XX(34),XX(34).LE.XX(83),ZABQ;
      ACTIVITY,XX(83),XX(34).GT.XX(83),FT17;
;
ZABR  ASSIGN,XX(83)=XX(83)-XX(36);
      ACTIVITY;
CT18  ALTER,TOOL18,-1,2;
      ACTIVITY,,,N18;
      ACTIVITY,,NNRSC(23).LT.1,EV15;
N18   ASSIGN,XX(36)=EXPON(20,1),XX(35)=XX(83),1;
      ACTIVITY,XX(36),XX(36).LE.XX(83),ZABR;
      ACTIVITY,XX(83),XX(36).GT.XX(83),FT18;
;
ZABS  ASSIGN,XX(83)=XX(83)-XX(38);
```

```
      ACTIVITY;
CT19  ALTER,TOOL19,-1,2;
      ACTIVITY,,,N19;
      ACTIVITY,,NNRSC(24).LT.1,EV15;
N19   ASSIGN,XX(38)=EXPON(20,1),XX(37)=XX(83),1;
      ACTIVITY,XX(38),XX(38).LE.XX(83),ZABS;
      ACTIVITY,XX(83),XX(38).GT.XX(83),FT19;
;
ZABT  ASSIGN,XX(83)=XX(83)-XX(40);
      ACTIVITY;
CT20  ALTER,TOOL20,-1,2;
      ACTIVITY,,,N20;
      ACTIVITY,,NNRSC(25).LT.1,EV15;
N20   ASSIGN,XX(40)=EXPON(20,1),XX(39)=XX(83),1;
      ACTIVITY,XX(40),XX(40).LE.XX(83),ZABT;
      ACTIVITY,XX(83),XX(40).GT.XX(83),FT20;
;
ZABU  ASSIGN,XX(83)=XX(83)-XX(42);
      ACTIVITY;
CT21  ALTER,TOOL21,-1,2;
      ACTIVITY,,,N21;
      ACTIVITY,,NNRSC(26).LT.1,EV15;
N21   ASSIGN,XX(42)=EXPON(20,1),XX(41)=XX(83),1;
      ACTIVITY,XX(42),XX(42).LE.XX(83),ZABU;
      ACTIVITY,XX(83),XX(42).GT.XX(83),FT21;
;
ZABV  ASSIGN,XX(83)=XX(83)-XX(44);
      ACTIVITY;
CT22  ALTER,TOOL22,-1,2;
      ACTIVITY,,,N22;
      ACTIVITY,,NNRSC(27).LT.1,EV15;
N22   ASSIGN,XX(44)=EXPON(20,1),XX(43)=XX(83),1;
      ACTIVITY,XX(44),XX(44).LE.XX(83),ZABV;
      ACTIVITY,XX(83),XX(44).GT.XX(83),FT22;
;
ZABW  ASSIGN,XX(83)=XX(83)-XX(46);
      ACTIVITY;
CT23  ALTER,TOOL23,-1,2;
      ACTIVITY,,,N23;
      ACTIVITY,,NNRSC(28).LT.1,EV15;
N23   ASSIGN,XX(46)=EXPON(20,1),XX(45)=XX(83),1;
      ACTIVITY,XX(46),XX(46).LE.XX(83),ZABW;
      ACTIVITY,XX(83),XX(46).GT.XX(83),FT23;
;
ZABX  ASSIGN,XX(83)=XX(83)-XX(48);
      ACTIVITY;
CT24  ALTER,TOOL24,-1,2;
      ACTIVITY,,,N24;
      ACTIVITY,,NNRSC(29).LT.1,EV15;
N24   ASSIGN,XX(48)=EXPON(20,1),XX(47)=XX(83),1;
      ACTIVITY,XX(48),XX(48).LE.XX(83),ZABX;
      ACTIVITY,XX(83),XX(48).GT.XX(83),FT24;
```

```
;
ZABY  ASSIGN,XX(84)=XX(84)-XX(50);
      ACTIVITY;
CT25  ALTER,TOOL25,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(30).LT.1,EV16;
S25   ASSIGN,XX(50)=EXPON(20,1),XX(49)=XX(84),1;
      ACTIVITY,XX(50),XX(50).LE.XX(84),ZABY;
      ACTIVITY,XX(84),XX(50).GT.XX(84),FT25;
;
ZABZ  ASSIGN,XX(84)=XX(84)-XX(52);
      ACTIVITY;
CT26  ALTER,TOOL26,-1,2;
      ACTIVITY;
      ACTIVITY,,NNRSC(31).LT.1,EV16;
TS26  ASSIGN,XX(52)=EXPON(20,1),XX(51)=XX(84),1;
      ACTIVITY,XX(52),XX(52).LE.XX(84),ZABZ;
      ACTIVITY,XX(84),XX(52).GT.XX(84),FT26;
;FILE TOOL2.NET, NODE LABEL SEED
;
EV15  GOON,1;
      ACTIVITY,,XX(150).EQ.1;
      ACTIVITY,,XX(150).EQ.0,VAAB;
      EVENT,15;
      ACTIVITY;
VAAB  TERMINATE;
;
EV16  GOON,1;
      ACTIVITY,,XX(151).EQ.1;
      ACTIVITY,,XX(151).EQ.0,VAAC;
      EVENT,16;
      ACTIVITY;
VAAC  TERMINATE;
;
EV17  GOON,1;
      ACTIVITY,,XX(152).EQ.1;
      ACTIVITY,,XX(152).EQ.0,VAAD;
      EVENT,17;
      ACTIVITY;
VAAD  TERMINATE;
      END;
INITIALIZE,,,Y;
FIN;
```

APPENDIX 2

Discrete Event, USERF FORTRAN Programs, and Post-Simulation Test C^{++} program

Discrete Event Program

```
      SUBROUTINE EVENT(I)
$INCLUDE:'PARAM.INC'
$INCLUDE:'SCOM1.COM'
      TL=0
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20),I
    1 J1=ATRIB(6)
      J2=ATRIB(7)
      ATRIB(10)= GETARY(J1,J2)
      IF (ATRIB(10).GE.8).AND.(ATRIB(10).LE.20) THEN
      ATRIB(11)=1
      ELSE IF (ATRIB(10).GE.21).AND.(ATRIB(10).LE.30) THEN
      ATRIB(11)=2
      ELSE IF (ATRIB(10).GE.31).AND.(ATRIB(10).LE.47) THEN
      ATRIB(11)=3
      ELSE IF (ATRIB(10).EQ.0) THEN
      ATRIB(11)=4
      END IF
      CONTINUE
      RETURN
    2 K1=ATRIB(10)
      K2=ATRIB(12)
      ATRIB(13)=GETARY(K2,K1)
      IF (NNCNT(100).NE.0) GO TO 50
      XX(2)=EXPON(20,1)
      XX(4)=EXPON(20,1)
      XX(6)=EXPON(20,1)
      XX(8)=EXPON(20,1)
      XX(10)=EXPON(20,1)
      XX(12)=EXPON(20,1)
      XX(14)=EXPON(20,1)
      XX(16)=EXPON(20,1)
   50 CONTINUE
      RETURN
    3 K3=ATRIB(14)
      K4=ATRIB(10)
      ATRIB(15)=GETARY(K3,K4)
      IF (NNCNT(100).NE.0) GO TO 100
      XX(18)=EXPON(20,1)
      XX(20)=EXPON(20,1)
      XX(22)=EXPON(20,1)
      XX(24)=EXPON(20,1)
      XX(26)=EXPON(20,1)
      XX(28)=EXPON(20,1)
      XX(30)=EXPON(20,1)
      XX(32)=EXPON(20,1)
  100 CONTINUE
      'RETURN
    4 K5=ATRIB(16)
      K6=ATRIB(10)
```

```
      ATRIB(17)=GETARY(K5,K6)
      IF (NNCNT(100).NE.0) GO TO 150
      XX(34)=EXPON(20,1)
      XX(36)=EXPON(20,1)
      XX(38)=EXPON(20,1)
      XX(40)=EXPON(20,1)
      XX(42)=EXPON(20,1)
      XX(44)=EXPON(20,1)
      XX(46)=EXPON(20,1)
      XX(48)=EXPON(20,1)
 150  CONTINUE
      RETURN
   5  K7=ATRIB(19)
      K8=ATRIB(10)
      ATRIB(18)=GETARY(K7,K8)
      IF (NNCNT(100).NE.0) GO TO 200
      XX(50)=EXPON(20,1)
      XX(52)=EXPON(20,1)
      XX(54)=EXPON(20,1)
      XX(56)=EXPON(20,1)
      XX(58)=EXPON(20,1)
      XX(60)=EXPON(20,1)
      XX(62)=EXPON(20,1)
      XX(64)=EXPON(20,1)
 200  CONTINUE
      RETURN
   6  K9=ATRIB(20)
      K10=ATRIB(10)
      ATRIB(21)=GETARY(K9,K10)
      IF (NNCNT(100).NE.0) GO TO 250
      XX(66)=EXPON(20,1)
      XX(68)=EXPON(20,1)
      XX(70)=EXPON(20,1)
      XX(72)=EXPON(20,1)
      XX(74)=EXPON(20,1)
      XX(76)=EXPON(20,1)
      XX(78)=EXPON(20,1)
      XX(80)=EXPON(20,1)
 250  CONTINUE
      RETURN
   7  J7=0
      J8=0
      J9=0
      DO 300 J3=11,13,1
        DO 302  J4=1,8,1
         J5=ATRIB(10)
         ATRIB(30)=GETARY(J4,J5)
         IF (ATRIB(30).NE.0) GO TO 301
         IF (J3.NE.11) GO TO 303
         J7=1
         GO TO 300
 303      IF (J3.NE.12) GO TO 304
```

```
      J8=1
      GO TO 300
304   IF (J3.NE.13) GO TO 300
      J9=1
      GO TO 300
301   J6=ATRIB(30)
      ATRIB(27)=GETARY(J3,J6)
      IF (ATRIB(27).EQ.0) GO TO 300
302   CONTINUE
300 CONTINUE
    IF ((J7.NE.0).OR.(J8.NE.0).OR.(J9.NE.0)) GO TO 347
    CALL UERR(3)
    CALL ERROR(2)
    GO TO 349
347 XX(98)=NNQ(75)+NNQ(2)
    XX(99)=NNQ(76)+NNQ(3)
    XX(100)=NNQ(77)+NNQ(4)
    IF ((J7.NE.1).or.(J8.NE.1).or.(J9.NE.1)) GO TO 316
305 IF (XX(98).LT.XX(99)) GO TO 308
306 IF (XX(98).EQ.XX(99)) GO TO 309
307 IF (XX(99).LT.XX(100)) GO TO 310
    IF (XX(99).EQ.XX(100)) GO TO 311
    ATRIB(28)=3
    CONTINUE
    GO TO 349
308 IF (XX(98).LT.XX(100)) GO TO 312
    IF (XX(98).EQ.XX(100)) GO TO 313
    CONTINUE
    GO TO 306
309 IF (XX(98).LT.XX(100)) GO TO 314
    IF (XX(98).EQ.XX(100)) GO TO 315
    CONTINUE
    GO TO 307
310 ATRIB(28)=2
    CONTINUE
    GO TO 349
311 ATRIB(28)=2
    CONTINUE
    GO TO 349
312 ATRIB(28)=1
    CONTINUE
    GO TO 349
313 ATRIB(28)=1
    CONTINUE
    GO TO 349
314 ATRIB(28)=1
    CONTINUE
    GO TO 349
315 ATRIB(28)=1
    CONTINUE
    GO TO 349
316 IF ((J7.NE.1).or.(J8.NE.1).or.(j9.ne.0)) GO TO 322
```

```
317  IF (XX(98).LT.XX(99)) GO TO 320
318  IF (XX(98).EQ.XX(99)) GO TO 320
319  IF (XX(98).GT.XX(99)) GO TO 321
320  ATRIB(28)=1
     GO TO 349
321  ATRIB(28)=2
     GO TO 349
322  IF ((J8.NE.1).or.(J9.NE.1).or.(j7.ne.0)) GO TO 328
323  IF (XX(99).LT.XX(100)) GO TO 326
324  IF (XX(99).EQ.XX(100)) GO TO 326
325  IF (XX(99).GT.XX(100)) GO TO 327
326  ATRIB(28)=2
     GO TO 349
327  ATRIB(28)=3
     GO TO 349
328  IF ((J7.NE.1).or.(J9.NE.1).or.(j8.ne.0)) GO TO 334
329  IF (XX(98).LT.XX(100)) GO TO 332
330  IF (XX(98).EQ.XX(100)) GO TO 332
331  IF (XX(98).GT.XX(100)) GO TO 333
332  ATRIB(28)=1
     GO TO 349
333  ATRIB(28)=3
     GO TO 349
334  IF ((J7.NE.1).or.(j8.ne.0).or.(J9.NE.0)) GO TO 335
     ATRIB(28)=1
     GO TO 349
335  IF ((J8.NE.1).OR.(J7.NE.0).OR.(J9.NE.0)) GO TO 336
     ATRIB(28)=2
     GO TO 349
336  IF ((J9.NE.1).OR.(J7.NE.0).OR.(J8.NE.0)) GO TO 349
     ATRIB(28)=3
     GO TO 349
349  CONTINUE
     RETURN
8    JJJ7=0
     JJJ8=0
     JJJ9=0
     DO 360 JJJ3=14,15,1
     DO 362  JJJ4=1,8,1
     JJJ5=ATRIB(10)
     ATRIB(30)=GETARY(JJJ4,JJJ5)
     IF (ATRIB(30).NE.0) GO TO 361
     IF (JJJ3.NE.14) GO TO 363
     JJJ7=1
     GO TO 360
363 IF (JJJ3.NE.15) GO TO 364
     JJJ8=1
364 GO TO 360
361 JJ06=ATRIB(30)
     JJJ6=JJ06-8
     ATRIB(27)=GETARY(JJJ3,JJJ6)
     IF (ATRIB(27).EQ.0) GO TO 360
```

```
 362 CONTINUE
 360 CONTINUE
     IF ((JJJ7.NE.0).OR.(JJJ8.NE.0)) GO TO 353
     CALL UERR(2)
     CALL ERROR(1)
     GO TO 399
 353 XX(96)=NNQ(72)+NNQ(5)
     XX(97)=NNQ(73)+NNQ(6)
     IF ((JJJ7.NE.1).OR.(JJJ8.NE.1)) GO TO 351
     IF (XX(96).EQ.XX(97)) GO TO 350
     IF (XX(96).LT.XX(97)) GO TO 350
     CONTINUE
     ATRIB(28)=5
     GO TO 399
 350 ATRIB(28)=4
     GO TO 399
 351 IF ((JJJ7.NE.1).OR.(JJJ8.NE.0)) GO TO 352
     ATRIB(28)=4
     GO TO 399
 352 IF ((JJJ8.NE.1).OR.(JJJ7.NE.0)) GO TO 399
     ATRIB(28)=5
     GO TO 399
 399 CONTINUE
     RETURN

C****REPLENISHING TOOLS BASED ON FILURE REPLENISGING POLICY*****
   9 IS=5
     IH=0
     DO 401 IF=11,13,1
     DO 400 IG=1,8,1
     ITU=ITU+2
     ATRIB(29)=GETARY(IF,IG)
     IS=IS+1
     IH=ATRIB(29)-NNRSC(IS)
 410 CONTINUE
     CALL ALTER(IS,IH)
 400 CONTINUE
 401 CONTINUE
     ISSS=29
     IHHH=0
     DO 405 IFFF=14,15,1
     DO 404 IGGG=1,8,1
     ATRIB(29)=GETARY(IFFF,IGGG)
     ISSS=ISSS+1
     IHHH=ATRIB(29)-NNRSC(ISSS)
     CALL ALTER(ISSS,IHHH)
 404 CONTINUE
 405 CONTINUE
     ISS=100
     IHH=0
     DO 402 IFF=16,18,1
     DO 403 IGG=1,8,1
```

```
      ATRIB(29)=GETARY(IFF,IGG)
      ISS=ISS+1
      IHH=ATRIB(29)-NNRSC(ISS)
      CALL ALTER(ISS,IHH)
  403 CONTINUE
  402 CONTINUE
      XX(150)=GETARY(19,1)
      XX(151)=GETARY(19,2)
      XX(152)=GETARY(19,3)
      XX(153)=GETARY(19,4)
  407 CONTINUE
      RETURN

C****REPLENISHING TOOLS BASED ON PREVENTIVE REPLENISGING POLICY****
    9 IS=5
      IH=0
      IIG=1
      DO 401 IF=11,13,1
        DO 400 IG=1,8,1
         ATRIB(29)=GETARY(IF,IG)
         IS=IS+1
         IH=ATRIB(29)-NNRSC(IS)
         IF (XX(IIG).LT.20) GO TO 410
         XX(IIG)=0
  410     CONTINUE
         CALL ALTER(IS,IH)
         IIG=IIG+2
  400   CONTINUE
  401 CONTINUE
      ISSS=29
      IHHH=0
      IIIG=49
      DO 405 IFFF=14,15,1
        DO 404 IGGG=1,8,1
         ATRIB(29)=GETARY(IFFF,IGGG)
         ISSS=ISSS+1
         IHHH=ATRIB(29)-NNRSC(ISSS)
         IF (XX(IIIG).LT.20) GO TO 411
         XX(IIIG)=0
  411     CONTINUE
         CALL ALTER(ISSS,IHHH)
         IIIG=IIIG+2
  404   CONTINUE
  405 CONTINUE

      ISS=100
      IHH=0
      IIIIG=101
      DO 402 IFF=16,18,1
        DO 403 IGG=1,8,1
         ATRIB(29)=GETARY(IFF,IGG)
         ISS=ISS+1
```

```
      IHH=ATRIB(29)-NNRSC(ISS)
      IF (XX(IIIIG).LT.20) GO TO 412
      XX(IIIIG)=0
412    CONTINUE
      CALL ALTER(ISS,IHH)
      IIIIG=IIIIG+2
403   CONTINUE
402  CONTINUE
    XX(150)=GETARY(19,1)
    XX(151)=GETARY(19,2)
    XX(152)=GETARY(19,3)
    XX(153)=GETARY(19,4)
    CONTINUE
    RETURN
10  RETURN
11  JJ7=0
    JJ8=0
    JJ9=0
    DO 500 JJ3=16,18,1
    DO 502  JJ4=1,8,1
    JJ5=ATRIB(10)
    ATRIB(34)=GETARY(JJ4,JJ5)
    IF (ATRIB(34).NE.0) GO TO 501
    IF (JJ3.NE.16) GO TO 503
    JJ7=1
    GO TO 500
503 IF (JJ3.NE.17) GO TO 504
     JJ8=1
     GO TO 500
504  IF (JJ3.NE.18) GO TO 500
     JJ9=1
     GO TO 500
501  J6=ATRIB(34)
     J66=J6-16
     ATRIB(35)=GETARY(JJ3,J66)
     IF (ATRIB(35).EQ.0) GO TO 500
502  CONTINUE
500  CONTINUE
     IF ((JJ7.NE.0).OR.(JJ8.NE.0).OR.(JJ9.NE.0)) GO TO 537
     CALL UERR(4)
     CALL ERROR(3)
     GO TO 549
537  xx(160)=NNQ(78)+NNQ(81)
    XX(161)=NNQ(79)+NNQ(82)
    XX(162)=NNQ(80)+NNQ(83)
    IF ((JJ7.NE.1).OR.(JJ8.NE.1).OR.(JJ9.NE.1)) GO TO 516
505 IF (xx(160).LT.XX(161)) GO TO 508
506 IF (xx(160).EQ.XX(161)) GO TO 509
507 IF (XX(161).LT.XX(162)) GO TO 510
    IF (XX(161).EQ.XX(162)) GO TO 511
    ATRIB(28)=8
    CONTINUE
```

```
      GO TO 549
  508 IF (xx(160).LT.XX(162)) GO TO 512
      IF (xx(160).EQ.XX(162)) GO TO 513
      CONTINUE
      GO TO 506
  509 IF (xx(160).LT.XX(162)) GO TO 514
      IF (xx(160).EQ.XX(162)) GO TO 515
      CONTINUE
      GO TO 507
  510 ATRIB(28)=7
      CONTINUE
      GO TO 549
  511 ATRIB(28)=7
      CONTINUE
      GO TO 349
  512 ATRIB(28)=6
      CONTINUE
      GO TO 549
  513 ATRIB(28)=6
      CONTINUE
      GO TO 549
  514 ATRIB(28)=6
      CONTINUE
      GO TO 549
  515 ATRIB(28)=6
      CONTINUE
      GO TO 549
  516 IF ((JJ7.NE.1).OR.(JJ8.NE.1).OR.(JJ9.NE.0)) GO TO 522
  517 IF (xx(160).LT.XX(161)) GO TO 520
  518 IF (xx(160).EQ.XX(161)) GO TO 520
  519 IF (xx(160).GT.XX(161)) GO TO 521
  520 ATRIB(28)=6
      GO TO 549
  521 ATRIB(28)=7
      GO TO 549
  522 IF ((JJ8.NE.1).OR.(JJ9.NE.1).OR.(JJ7.NE.0)) GO TO 528
  523 IF (XX(161).LT.XX(162)) GO TO 526
  524 IF (XX(161).EQ.XX(162)) GO TO 526
  525 IF (XX(161).GT.XX(162)) GO TO 527
  526 ATRIB(28)=7
      GO TO 549
  527 ATRIB(28)=8
      GO TO 549
  528 IF ((JJ7.NE.1).OR.(JJ9.NE.1).OR.(JJ8.NE.0)) GO TO 534
  529 IF (xx(160).LT.XX(162)) GO TO 532
  530 IF (xx(160).EQ.XX(162)) GO TO 532
  531 IF (xx(160).GT.XX(162)) GO TO 533
  532 ATRIB(28)=6
      GO TO 549
  533 ATRIB(28)=8
      GO TO 549
  534 IF ((JJ7.NE.1).OR.(JJ8.NE.0).OR.(JJ9.NE.0)) GO TO 535
```

```
      ATRIB(28)=6
      GO TO 549
535   IF ((JJ8.NE.1).OR.(JJ9.NE.0).OR.(JJ7.NE.0)) GO TO 536
      ATRIB(28)=7
      GO TO 549
536   IF ((JJ9.NE.1).OR.(JJ7.NE.0).OR.(JJ8.NE.0)) GO TO 549
      ATRIB(28)=8
      GO TO 549
549   CONTINUE
      RETURN
12    K11=ATRIB(22)
      K12=ATRIB(10)
      ATRIB(23)=GETARY(K11,K12)
      IF (NNCNT(100).NE.0) GO TO 550
      XX(102)=EXPON(20,1)
      XX(104)=EXPON(20,1)
      XX(106)=EXPON(20,1)
      XX(108)=EXPON(20,1)
      XX(110)=EXPON(20,1)
      XX(112)=EXPON(20,1)
      XX(114)=EXPON(20,1)
      XX(116)=EXPON(20,1)
550   CONTINUE
      RETURN
13    K13=ATRIB(24)
      K14=ATRIB(10)
      ATRIB(25)=GETARY(K13,K14)
      IF (NNCNT(100).NE.0) GO TO 600
      XX(118)=EXPON(20,1)
      XX(120)=EXPON(20,1)
      XX(122)=EXPON(20,1)
      XX(124)=EXPON(20,1)
      XX(126)=EXPON(20,1)
      XX(128)=EXPON(20,1)
      XX(130)=EXPON(20,1)
      XX(132)=EXPON(20,1)
600   CONTINUE
      RETURN
14    K15=ATRIB(26)
      K16=ATRIB(10)
      ATRIB(27)=GETARY(K15,K16)
      IF (NNCNT(100).NE.0) GO TO 650
      XX(134)=EXPON(20,1)
      XX(136)=EXPON(20,1)
      XX(138)=EXPON(20,1)
      XX(140)=EXPON(20,1)
      XX(142)=EXPON(20,1)
      XX(144)=EXPON(20,1)
      XX(146)=EXPON(20,1)
      XX(148)=EXPON(20,1)
650   CONTINUE
      RETURN
```

```
 15 IF (ATRIB(28).EQ.1) GO TO 651
    IF (ATRIB(28).EQ.2) GO TO 701
    IF (ATRIB(28).EQ.3) GO TO 751
    GO TO 799
651 IZ=0
    IZZ=0
    IB=0
    IBBB=0
    IBBBB=0
    IB=ATRIB(13)
    IBB=IB+13
    IBBB=IB+5
    IBBBB=NNRSC(IBB)
    IF (IBBBB.NE.1) GO TO 652
    IF (NRUSE(IBB).EQ.1) GO TO 653
    IZ=((2*IB)-1)
    IZZ=IZ+16
    XX(IZZ)=XX(IZ)
    CALL ALTER(IBB,-1)
    CALL ALTER(IBBB,1)
    GO TO 799
652 IF (IBBBB.LT.1) GO TO 653
    CALL ALTER(IBB,-1)
    CALL ALTER(IBBB,1)
    GO TO 799
653 IZA=0
    IZZA=0
    IC=0
    ICCC=0
    ICCCC=0
    IC=ATRIB(15)
    ICC=IC+21
    ICCC=IC+5
    ICCCC=NNRSC(ICC)
    IF (ICCC.NE.1) GO TO 654
    IF (NRUSE(ICC).EQ.1) GO TO 799
    IZA=(2*IB)-1
    IZZA=IZA+32
    XX(IZZA)=XX(IZA)
    CALL ALTER(ICC,-1)
    CALL ALTER(ICCC,1)
    GO TO 799
654 IF (ICCCC.LT.1) GO TO 799
    CALL ALTER(ICC,-1)
    CALL ALTER(ICCC,1)
    GO TO 799
701 IZZB=0
    IZB=0
    ID=0
    IDDD=0
    IDDDD=0
    ID=ATRIB(15)
```

```
      IDD=ID+5
      IDDD=ID+13
      IDDDD=NNRSC(IDD)
      IF (IDDDD.NE.1) GO TO 702
      IF (NRUSE(IDD).EQ.1) GO TO 703
      IZB=((2*ID)+16)-1
      IZZB=IZB-16
      XX(IZZB)=XX(IZB)
      CALL ALTER(IDD,-1)
      CALL ALTER(IDDD,1)
      GO TO 799
702 IF (IDDDD.LT.1) GO TO 703
      CALL ALTER(IDD,-1)
      CALL ALTER(IDDD,1)
      GO TO 799
703 IZC=0
      IZZC=0
      IE=0
      IEEE=0
      IEEEE=0
      IE=ATRIB(15)
      IEE=IE+21
      IEEE=IE+13
      IEEEE=NNRSC(IEE)
      IF (IEEEE.NE.1) GO TO 705
      IF (NRUSE(IEE).EQ.1) GO TO 799
      IC=((2*IE)+16)-1
      ICC=IC+16
      XX(ICC)=XX(IC)
      CALL ALTER(IEE,-1)
      CALL ALTER(IEEE,1)
      GO TO 799
705 IF (IEEEE.LT.1) GO TO 799
      CALL ALTER(IEE,-1)
      CALL ALTER(IEEE,1)
      GO TO 799
751  IZD=0
      IZZD=0
      IF=0
      IFFF=0
      IFFFF=0
      IF=ATRIB(17)
      IFF=IF+5
      IFFF=IF+21
      IFFFF=NNRSC(IFF)
      IF (IFFFF.NE.1) GO TO 752
      IF (NRUSE(IFF).EQ.1) GO TO 753
      IZD=((2*IF)+32)-1
      IZZD=IZD-32
      XX(IZZ)=XX(IZ)
      CALL ALTER(IFF,-1)
      CALL ALTER(IFFF,1)
```

```
      GO TO 799
 752 IF (IFFFF.LT.1) GO TO 753
      CALL ALTER(IFF,-1)
      CALL ALTER(IFFF,1)
      GO TO 799
 753 IZZE=0
      IZE=0
      IG=0
      IGGG=0
      IGGGG=0
      IG=ATRIB(17)
      IGG=IG+13
      IGGG=IG+21
      IGGGG=NNRSC(IGG)
      IF (IGGGG.NE.1) GO TO 754
      IF (NRUSE(IGG).EQ.1) GO TO 799
      IZE=((2*IG)+32)-1
      IZZE=IZE-16
      XX(IZZE)=XX(IZE)
      CALL ALTER(IGG,-1)
      CALL ALTER(IGGG,1)
      GO TO 799
 754  IF (IGGGG.LT.1) GO TO 799
      CALL ALTER(IGG,-1)
      CALL ALTER(IGGG,1)
      GO TO 799
 799 CONTINUE
      RETURN
 16   IF (ATRIB(28).EQ.4) GO TO 801
      IF (ATRIB(28).EQ.5) GO TO 802
      GO TO 849
 801 IZF=0
      IZZF=0
      IJ=0
      IJJJ=0
      IJJJJ=0
      IJ=ATRIB(18)
      IJJ=IJ+29
      IJJJ=IJ+21
      IJJJJ=NNRSC(IJJ)
      IF (IJJJ.NE.1) GO TO 803
      IF (NRUSE(IJJ).EQ.1) GO TO 849
      IJ=IJ-8
      IZF=((2*IJ)+48)-1
      IZZF=IZF+16
      XX(IZZF)=XX(IZF)
      CALL ALTER(IJJ,-1)
      CALL ALTER(IJJJ,1)
      GO TO 849
 803  IF (IJJJJ.LT.1) GO TO 849
      CALL ALTER(IJJ,-1)
      CALL ALTER(IJJJ,1)
```

```
      GO TO 849
802 IK=0
      IKKK=0
      IKKKK=0
      IK=ATRIB(21)
      IKK=IK+21
      IKKK=IK+29
      IKKKK=NNRSC(IKK)
      IF (IKKK.NE.1) GO TO 805
      IF (NRUSE(IKK).EQ.1) GO TO 849
      IK=IK-8
      IZG=((2*IK)+64)-1
      IZZG=IZG-16
      XX(IZZG)=XX(IZG)
      CALL ALTER(IKK,-1)
      CALL ALTER(IKKK,1)
      GO TO 849
805 IF (IKKK.LT.1) GO TO 849
      CALL ALTER(IKK,-1)
      CALL ALTER(IKKK,1)
      GO TO 849
849 CONTINUE
      RETURN
 17 IF (ATRIB(28).EQ.6) GO TO 851
      IF (ATRIB(28).EQ.7) GO TO 901
      IF (ATRIB(28).EQ.8) GO TO 951
      GO TO 999
851 IZH=C
      IZZH=0
      IL=0
      ILLL=0
      ILLLL=0
      IL=ATRIB(23)
      ILL=IL+92
      ILLL=IL+84
      ILLLL=NNRSC(ILL)
      IF (ILLL.NE.1) GO TO 852
      IF (NRUSE(ILL).EQ.1) GO TO 853
      IL=IL-16
      IZH=((2*IL)+100)-1
      IZZH=IZH+16
      XX(IZZH)=XX(IZH)
      CALL ALTER(ILL,-1)
      CALL ALTER(ILLL,1)
      GO TO 999
852  IF (ILLLL.LT.1) GO TO 853
      CALL ALTER(ILL,-1)
      CALL ALTER(ILLL,1)
      GO TO 999
853  IZJ=0
      IZZJ=0
      IM=0
```

```
      IMMM=0
      IMMMM=0
      IM=ATRIB(23)
      IMM=IM+100
      IMMM=IM+84
      IMMMM=NNRSC(IMM)
      IF (IMMM.NE.1) GO TO 854
      IF (NRUSE(IMM).EQ.1) GO TO 999
      IM=IM-16
      IZJ=((2*IM)+100)-1
      IZZJ=IZJ+32
      XX(IZZH)=XX(IZH)
      CALL ALTER(ILL,-1)
      CALL ALTER(ILLL,1)
      GO TO 999
  854 IF (IMMMM.LT.1) GO TO 999
      CALL ALTER(IMM,-1)
      CALL ALTER(IMMM,1)
      GO TO 999
  901 IZK=0
      IZZK=0
      IN=0
      INNN=0
      INNNN=0
      IN=ATRIB(25)
      INN=IN+84
      INNN=IN+92
      INNNN=NNRSC(INN)
      IF (INNN.NE.1) GO TO 902
      IF (NRUSE(INN).EQ.1) GO TO 903
      IN=IN-16
      IZK=((2*IN)+116)-1
      IZZK=IZK-16
      XX(IZZK)=XX(IZK)
      CALL ALTER(INN,-1)
      CALL ALTER(INNN,1)
      GO TO 999
  902 IF (INNNN.LT.1) GO TO 903
      CALL ALTER(INN,-1)
      CALL ALTER(INNN,1)
      GO TO 999
  903 IZL=0
      IZZL=0
      IO=0
      IOOO=0
      IOOOO=0
      IO=ATRIB(25)
      IOO=IO+100
      IOOO=IO+92
      IOOOO=NNRSC(IOO)
      IF (IOOO.NE.1) GO TO 904
      IF (NRUSE(IOO).EQ.1) GO TO 999
```

```
      IO=IO-16
      IZL=((2*IO)+116)-1
      IZZL=IZL+16
      XX(IZZL)=XX(IZL)
      CALL ALTER(IOO,-1)
      CALL ALTER(IOOO,1)
      GO TO 999
  904 IF (IOOOO.LT.1) GO TO 999
      CALL ALTER(IOO,-1)
      CALL ALTER(IOOO,1)
      GO TO 999
  951 IZM=0
      IZZM=0
      IP=0
      IPPP=0
      IPPPP=0
      IP=ATRIB(27)
      IPP=IP+84
      IPPP=IP+100
      IPPPP=NNRSC(IPP)
      IF (IPPP.NE.1) GO TO 954
      IF (NRUSE(IPP).EQ.1) GO TO 999
      IO=IO-16
      IZM=((2*IP)+132)-1
      IZZM=IZM-32
      XX(IZZM)=XX(IZM)
      CALL ALTER(IPP,-1)
      CALL ALTER(IPPP,1)
      GO TO 999
  954 IF (IPPPP.LT.1) GO TO 953
      CALL ALTER(IPP,-1)
      CALL ALTER(IPPP,1)
      GO TO 999
  953 IZN=0
      IZZN=0
      IQ=0
      IQQQ=0
      IQQQQ=0
      IQ=ATRIB(27)
      IQQ=IQ+92
      IQQQ=IQ+100
      IQQQQ=NNRSC(IQQ)
      IF (IQQQ.NE.1) GO TO 955
      IF (NRUSE(IQQ).EQ.1) GO TO 999
      IQ=IQ-16
      IZN=((2*IQ)+132)-1
      IZZN=IZN-16
      XX(IZZN)=XX(IZN)
      CALL ALTER(IQQ,-1)
      CALL ALTER(IQQQ,1)
      GO TO 999
  955 IF (IQQQQ.LT.1) GO TO 999
```

```
      CALL ALTER(IQQ,-1)
      CALL ALTER(IQQQ,1)
      GO TO 999
  999 CONTINUE
      RETURN

C****CALL STAT  SUBROUTINE  IN THE CASE OF RANDOM LOADING*****
 18   IF(TNOW.LT.4000) GO TO 954
      call uerr(1)
      CALL STAT(1)
 954  CONTINUE
      RETURN

C****CALL STAT SUBROUTINE IN THE CASE OF BATCH LOADING*****
 18   B=0
      B=TNOW
      XX(200)=XX(200)+B
      IF(NNCNT(333).LT.130) GO TO 954
      call uerr(1)
      CALL STAT(1)
 954  CONTINUE
      RETURN
19    XX(154)=0
      ISI=5
      IHI=0
      DO 1001 IFI=11,13,1
      DO 1000 IGI=1,8,1
      ATRIB(29)=GETARY(IFI,IGI)
      ISI=ISI+1
      IHI=ATRIB(29)-NNRSC(ISI)
      IF(IHI.EQ.0) GO TO 1000
      XX(154)=1
      GO TO 1049
 1000 COTINUE
 1001 CONTINUE
      ISSSI=29
      IHHHI=0
      DO 1005 IFFFI=14,15,1
      DO 1004 IGGGI=1,8,1
      ATRIB(29)=GETARY(IFFFI,IGGGI)
      ISSSI=ISSSI+1
      IHHHI=ATRIB(29)-NNRSC(ISSSI)
      IF(IHHHI.EQ.0) GO TO 1005
      XX(154)=1
      GO TO 1049
 1004 CONTINUE
 1005 CONTINUE
      ISSI=100
      IHHI=0
      DO 1002 IFFI=16,18,1
      DO 1003 IGGI=1,8,1
      ATRIB(29)=GETARY(IFFI,IGGI)
```

```
      ISSI=ISSI+1
      IHHI=ATRIB(29)-NNRSC(ISSI)
      IF(IHHI.EQ.0) GO TO 1002
      XX(154)=1
       GO TO 1049
1003 CONTINUE
1002 CONTINUE
1049 RETURN
  20 ISII=5
      IHII=0
      DO 2001 IFII=11,13,1
      DO 2000 IGII=1,8,1
      ATRIB(29)=GETARY(IFII,IGII)
      ISII=ISII+1
       IHII=ATRIB(29)-NNRSC(ISII)
       CALL ALTER(ISII,IHII).
2000 CONTINUE
2001 CONTINUE
      ISSSII=29
      IHHHII=0
      DO 2005 IFFFII=14,15,1
      DO 2004 IGGGII=1,8,1
      ATRIB(29)=GETARY(IFFFII,IGGGII)
      ISSSII=ISSSII+1
      IHHHII=ATRIB(29)-NNRSC(ISSSII)
      CALL ALTER(ISSSII,IHHHII)
2004 CONTINUE
2005 CONTINUE
      ISSII=100
      IHHII=0
      DO 2002 IFFII=16,18,1
      DO 2003 IGGII=1,8,1
      ATRIB(29)=GETARY(IFFII,IGGII)
      ISSII=ISSII+1
      IHHII=ATRIB(29)-NNRSC(ISSII)
      CALL ALTER(ISSII,IHHII)
2003 CONTINUE
2002 CONTINUE
      RETURN
      END
```

USERF FORTRAN Programs

```
C****COLECTS STATISTICS IN THE CASE OF RANDOM LOADING****
      SUBROUTINE STAT(I)
      DIMENSION F(20), E(20)
$INCLUDE:'PARAM.INC'
$INCLUDE:'SCOM1.COM'
      GO TO (1),I

    1 A=0. A1=0. A2=0. A3=0. A4=0. A5=0. A6=0. A7=0. A8=0. A9=0. A10=0
      IRR=XX(182)
      IF (IRR.NE.1) GO TO 960
      OPEN(9,FILE="C:\NEW\LOAD\NLL1HYC.DAT")
      GO TO 959
  960 IF (IRR.NE.2) GO TO 961
      OPEN(9,FILE="C:\NEW\LOAD\NLL2HYC.DAT")
      GO TO 959
  961 IF (IRR.NE.3) GO TO 962
      OPEN(9,FILE="C:\NEW\LOAD\NLL3HYC.DAT")
      GO TO 959
  962 IF (IRR.NE.4) GO TO 963
      OPEN(9,FILE="C:\NEW\LOAD\NLL4HYC.DAT")
      GO TO 959
  963 OPEN(9,FILE="C:\NEW\LOAD\NLL5HYC.DAT")
  959 A=RRAVG(6)+RRAVG(7)+RRAVG(8)+RRAVG(9)
      A=A+RRAVG(10)+RRAVG(11)+RRAVG(12)+RRAVG(13)
      A1=RRAVG(14)+RRAVG(15)+RRAVG(16)+RRAVG(17)
      A1=A1+RRAVG(18)+RRAVG(19)+RRAVG(20)+RRAVG(21)
      A2=RRAVG(22)+RRAVG(23)+RRAVG(24)+RRAVG(25)
      A2=A2+RRAVG(26)+RRAVG(27)+RRAVG(28)+RRAVG(29)
      A3=RRAVG(30)+RRAVG(31)+RRAVG(32)+RRAVG(33)
      A3=A3+RRAVG(34)+RRAVG(35)+RRAVG(36)+RRAVG(37)
      A4=RRAVG(38)+RRAVG(39)+RRAVG(40)+RRAVG(41)
      A4=A4+RRAVG(42)+RRAVG(43)+RRAVG(44)+RRAVG(45)
      A5=RRAVG(101)+RRAVG(102)+RRAVG(103)+RRAVG(104)
      A5=A5+RRAVG(105)+RRAVG(106)+RRAVG(107)+RRAVG(108)
      A6=RRAVG(109)+RRAVG(110)+RRAVG(111)+RRAVG(112)
      A6=A6+RRAVG(113)+RRAVG(114)+RRAVG(115)+RRAVG(116)
      A7=RRAVG(117)+RRAVG(118)+RRAVG(119)+RRAVG(120)
      A7=A7+RRAVG(121)+RRAVG(122)+RRAVG(123)+RRAVG(124)
      MI=MI+1
      WRITE(9,957)MI
  957 FORMAT(1X,'RUN NUMBER = ',I10)
      WRITE(9,955)A,A1,A2,A3,A4,A5,A6,A7
  955 FORMAT(1X,'MACH.(1,2,3,4,5,6,7,8) UTIL.=',8F4.2)
      A8=A+A1+A2+A3+A4+A5+A6+A7
      A8=A8/8
      A9=NNCNT(200)
      WRITE(9,956)A8,A9
  956 FORMAT(1X,'SYS. UTIL. =',F6.3,5X,'THROUGHPUT =',F10.2)
      WRITE(9,958)A9
```

```
      FORMAT(1X,'THROUGHPUT =',F10.2)
      E(MI)=A8
      F(MI)=A9
      IF (MI.NE.10) GO TO 50
      DO 55 I=1,10
      ABM=ABM+E(I)
   55 ACM=ACM+F(I)
      ABM=ABM/10.
      ACM=ACM/10.
      WRITE(9,60)ABM,ACM
   60 FORMAT(1X,'MEAN UTILIZ. =',F10.2,5X,'MEAN THROUGHPUT =',F10.2)
      WRITE(9,61)ACM
      ASUMC=0
      ASUMB=0
      ASUMCC=0
      ASUMBB=0
      DO 20 I=1,10
      ASUMC=ASUMC + (ABS((F(I)-ACM)**2))
      ASUMCC=ASUMCC + (F(I)**2)
      ASUMBB=ASUMBB + (E(I)**2)
      ASUMB=ASUMB + (ABS((E(I)-ABM)**2))
   20 CONTINUE
      WRITE(9,62)ASUMBB
   62 FORMAT(1X,'SUM X*X FOR TOOLS UTILIZATION =',F20.3)
      WRITE(9,63)ASUMCC
   63 FORMAT(1X,'SUM X*X FOR THROGHPUT =',F20.2)
      ASC=ASUMC/9
      ASB=ASUMB/9
      ASCC=SQRT(ASC)
      ASBB=SQRT(ASB)
      AACH=(ACM + (1.83*(ASCC/SQRT(10))))
      AACL=(ACM - (1.83*(ASCC/SQRT(10))))
      AABH=(ABM + (1.83*(ASBB/SQRT(10))))
      AABL=(ABM - (1.83*(ASBB/SQRT(10))))
      WRITE(9,30)ASBB,ASCC
   30 FORMAT(1X,'S  UTILIZATION =',F10.4,5X,'S THROUGHPUT =',F10.2)
      WRITE(9,32)AABH,AACH
   32 FORMAT(1X,'HIEGH UTILIZ. =',F10.2,5X,'HIGH THROUGH. =',F10.2)
      WRITE(9,33)AABL,AACL
   33 FORMAT(1X,'LOWER UTILIZ. =',F10.2,5X,'LOWER THROUGH =',F10.2)
   50 RETURN
      END


C*****COLECTS STATISTICS IN THE CASE OF BATCH LOADING*****
      SUBROUTINE STAT(I)
      DIMENSION F(20), E(20)
$INCLUDE:'PARAM.INC'
$INCLUDE:'SCOM1.COM'
      GO TO (1),I
    1 A=0, A1=0, A2=0, A3=0, A4=0, A5=0, A6=0, A7=0, A8=0, A9=0, A10=0
      IRR=XX(182)
```

```
      IF (IRR.NE.1) GO TO 960
      OPEN(9,FILE="C:\REW\PRE\BATCH\RPR3HYB.DAT")
      GO TO 959

960   IF (IRR.NE.2) GO TO 961
      OPEN(9,FILE="C:\REW\PRE\BATCH\RPR6HYB.DAT")
      GO TO 959
961   IF (IRR.NE.3) GO TO 962
      OPEN(9,FILE="C:\REW\PRE\BATCH\RPR9HYB.DAT")
      GO TO 959
962   IF (IRR.NE.4) GO TO 963
      OPEN(9,FILE="C:\REW\PRE\BATCH\RPR12HB.DAT")
      GO TO 959
963   OPEN(9,FILE="C:\REW\PRE\BATCH\RPR15HY.DAT")
959   A9=TNOW
      MI=MI+1
      WRITE(9,957)MI
957   FORMAT(1X,'RUN NUMBER = ',I10)
      A10=XX(200)/100
      WRITE(9,958)A9,A10
958   FORMAT(1X,'MAKESPAN =',E9.4,5x,'MFT =',E10.4)
      E(MI)=A10
      F(MI)=A9
      IF (MI.NE.10) GO TO 50
      DO 55 I=1,10
      ABM=ABM+F(I)
 55   ACM=ACM+E(I)
      ABM=ABM/10.
      ACM=ACM/10.
      WRITE(9,60)ABM,ACM
 60   FORMAT(1X,'MEAN FOR MAKESPAN =',E9.4,5x,'MEAN FOR MFT =',E9.4)
      ASUMC=0
      ASUMB=0
      ASUMCC=0
      ASUMBB=0
      DO 20 I=1,10
      ASUMC=ASUMC + (ABS((E(I)-ACM)**2))
      ASUMCC=ASUMCC + (E(I)**2)
      ASUMBB=ASUMBB + (F(I)**2)
      ASUMB=ASUMB + (ABS((F(I)-ABM)**2))
 20   CONTINUE
      WRITE(9,62)ASUMBB
 62   FORMAT(1X,'SUM X*X FOR MAKESPAN =',F20.2)
      WRITE(9,63)ASUMCC
 63   FORMAT(1X,'SUM X*X FOR MTTF =',F20.2)
      ASC=ASUMC/9
      ASB=ASUMB/9
      ASCC=SQRT(ASC)
      ASBB=SQRT(ASB)
      AACH=(ACM + (1.83*(ASCC/SQRT(10))))
      AACL=(ACM - (1.83*(ASCC/SQRT(10))))
      AABH=(ABM + (1.83*(ASBB/SQRT(10))))
```

```
      AABL=(ABM - (1.83*(ASBB/SQRT(10))))
      WRITE(9,30)ASBB,ASCC
   30 FORMAT(1X,'S FOR MAKESPAN =',E9.4,5X,'S FOR MFT',E9.4)
      WRITE(9,32)AABH,AACH
   32 FORMAT(1X,'HIEG FOR MAKESPAN =',E9.4,9X,'HIEG FOR MFT =',E9.4)
      WRITE(9,33)AABL,AACL
   33 FORMAT(1X,'LOW FOR MAKESPAM =',E9.4,9X,'LOW FOR MFT =',E9.4)
   50 RETURN
      END

      FUNCTION USERF (I)
$INCLUDE:'PARAM.INC'
$INCLUDE:'SCOM1.COM'
      GO TO (1,2,3,4,5,6,7,8),I
    1 L1=0,L2=0
      L1=ATRIB(10)
      L2=ATRIB(12)+19
      LL=GETARY(L2,L1)
      USERF = LL
      RETURN
    2 L3=0,L4=0
      L3=ATRIB(14)+19
      L4=ATRIB(10)
      LL=GETARY(L3,L4)
      USERF = LL
      RETURN
    3 L5=0, L6=0
      L5=ATRIB(16)+19
      L6=ATRIB(10)
      LL=GETARY(L5,L6)
      USERF = LL
      RETURN
    4 L7=0, L8=0
      L7=ATRIB(19) +19
      L8=ATRIB(10)
      LL=GETARY(L7,L8)
      USERF = LL
      RETURN
    5 L9=0, L10=0
      L9=ATRIB(20) + 19
      L10=ATRIB(10)
      LL=GETARY(L9,L10)
      USERF = LL
      RETURN
    6 L11=0, L12=0
      L11=ATRIB(22)+19
      L12=ATRIB(10)
      LL=GETARY(L11,L12)
      USERF = LL
      RETURN
    7 L13=0, L14=0
      L13=ATRIB(24)+19
```

```
      L14=ATRIB(10)
      LL=GETARY(L13,L14)
      USERF - LL
      RETURN
    8 L15=0, L16=0
      L15=ATRIB(26)+19
      L16=ATRIB(10)
      LL=GETARY(L15,L16)
      USERF = LL
      RETURN
      END

      SUBROUTINE UERR(I)
$INCLUDE:'PARAM.INC'
$INCLUDE:'SCOM1.COM'
      GO TO (1,2,3,4),I
    1 WRITE(6,10)
   10 FORMAT(//)
      WRITE(6,11)
   11 FORMAT(5X,13(2H *),'INFORMATION',13(2H* ))
      WRITE(6,12)
   12 FORMAT(5X,'AUTOMATED MANUFACTURING SIMULATION MODEL')
      WRITE(6,13)
   13 FORMAT(5X,'BY SIROS AMINI, MECHANICAL ENGINEERING DEPARTMENT')
      WRITE(6,14)
   14 FORMAT(5X,'FACULTY OF ENGINEERING,UNIVERSITY OF OTTAWA')
      RETURN
    2 WRITE(6,15)
   15 FORMAT(//)
      WRITE(6,16)
   16 FORMAT(5X,13(2H *),'00PS!?',13(2H* ))
      WRITE(6,17)
   17 FORMAT(5X,'***ERROR***')
      WRITE(6,18)
   18 FORMAT (5X,'CHECK PROCESSING STEPS AND TOOLS FOR THE  MACHINE GROUP 2')
      RETURN
    3 WRITE(6,19)
   19 FORMAT(//)
      WRITE(6,20)
   20 FORMAT(5X,13(2H *),'OOPS!?',13(2H* ))
      WRITE(6,21)
   21 FORMAT(5X,'***ERROR***')
      WRITE(6,22)
   22 FORMAT (5X,'CHECK PROCESSING STEPS AND TOOLS FOR THE MACHINE GROUP 1')
      RETURN
    4 WRITE(6,23)
   23 FORMAT(//)
      WRITE(6,24)
   24 FORMAT(5X,13(2H *),'OOPS!?',13(2H* ))
      WRITE(6,25)
   25 FORMAT(5X,'***ERROR***')
      WRITE(6,26)
```

```
26 FORMAT (5X,'CHECK PROCESSING STEPS AND TOOLLS FOR THE MACHINE GROUP 3')
   RETURN
   END
```

Post-Simulation Test C++ Program

```
/***************** ANOVA & TUKEY T TEST ****************/
/******* SHARED TOOLING VS DEDICATED TOOLING ******/
#INCLUDE <STDIO.H>
#INCLUDE <CONIO.H>
#INCLUDE <MATH.H>
#INCLUDE <STRING.H>
FILE *FP1;
FILE *FP2;
MAIN()
{
INT I,J,K;
INT R=5,C=2,KMAX=30;
FLOAT X[2][5],UID[5],UDJ[5];
FLOAT MU,SSA,SSE,MSA,MSE,FA,F,DIF,CR,Q;
CHAR PDIR,RESULT1,RESULT2;
CHAR TIT1[80];
FP2=FOPEN("STATEST.OUT","W");
FPRINTF(FP2,"ANALYSIS OF VARIANCE ( @ F%90 = 4.54  &  Q%90 = 2.85 )\N");
FPRINTF(FP2,"\NSHARED TOOLING VS DEDICATED TOOLING\N\N\N");
FCLOSE(FP2);
FP1=FOPEN("STATEST.DAT","R");
FP2=FOPEN("STATEST.OUT","A");
FOR (K=1; K<=KMAX; ++K)
{

/*********** INITIALIZE VARIABLES **************/
FOR (I=0; I<R; ++I)
{
   FOR (J=0; J<C; ++J)
   {
        X[J][I]=0;
        UDJ[J]=0;
   }
   UID[I]=0;
}
MU=0;
SSA=0;
SSE=0;
MSA=0;
MSE=0;
FA=0;
F=4.54; /* F VALUE FOR %90 */
Q=2.85; /* Q VALUE FOR %90 */

/************* READ POINTS FROM DATA FILE *************/
FGETS(TIT1,80,FP1);
FSCANF(FP1,"%C\N",&PDIR);
FOR (J=0; J<C; ++J)
```

```
{
   FOR (I=0; I<R; ++I)
   {
        FSCANF(FP1,"%F,",&X[J][I]);
   }
   FSCANF(FP1,"\N");
}

/*************** CALCULATING THE AVERAGE VALUES ***************/
FOR (I=0; I<R; ++I)
{
   FOR (J=0; J<C; ++J)
   {
        UID[I]+=X[J][I];
   }
   UID[I]=UID[I]/C;
}
FOR (J=0; J<C; ++J)
{
   FOR (I=0; I<R; ++I)
   {
        UDJ[J]+=X[J][I];
   }
   UDJ[J]=UDJ[J]/R;
   MU+=UDJ[J];
}
MU=MU/C;

IF (X[0][0]==0) GOTO STEP1;

/*****************STATISTICAL ANALYSIS ****************/
FOR (J=0; J<C; ++J)
{
   FOR (I=0; I<R; ++I)
   {
        SSA+=POW((UDJ[J]-MU),2);
        SSE+=POW((X[J][I]-UDJ[J]-UID[I]+MU),2);
   }
}
MSA=SSA/(C-1);
MSE=SSE/((R-1)*(C-1));

IF (MSE==0) MSE=.000001;
FA=MSA/MSE;
IF (FA>=F) RESULT1='R';
ELSE RESULT1='A';

DIF=UDJ[0]-UDJ[1];
IF (PDIR=='N') DIF=-DIF;
CR=Q*SQRT(MSE/R);
IF (DIF>=CR) RESULT2='A';
ELSE RESULT2='R';
```

```
STEP1::

/*************** PRINTING THE RESULTS *****************/

FPRINTF(FP2,"\N*********************************************\N");
FPRINTF(FP2,"%S",TIT1);
FPRINTF(FP2,"\N\N");
FOR (J=0; J<C; ++J)
{
   IF (J==0) FPRINTF(FP2,"SHARED:      ");
   ELSE FPRINTF(FP2,"DEDICATED:   ");
   FOR (I=0; I<R; ++I)
   {
       FPRINTF(FP2,"%.2F  ",X[J][I]);
   }
   FPRINTF(FP2,"    %.2F\N",UDJ[J]);
}
FPRINTF(FP2,"\N            ");
FOR (I=0; I<R; ++I)
{
   FPRINTF(FP2,"%.2F  ",UID[I]);
}
FPRINTF(FP2,"    %.2F\N\N\N",MU);
FPRINTF(FP2,"\NSSA=%F\NSSE=%F\NMSA=%F\NMSE=%F\NFA=%F
%C\N",SSA,SSE,MSA,MSE,FA,RESULT1);
IF (RESULT1=='R')
{
  FPRINTF(FP2,"\NCR = %F\NDIFFERENCE = %F      %C",CR,DIF,RESULT2);
  IF (RESULT2=='A') FPRINTF(FP2,"CCEPTED");
  ELSE FPRINTF(FP2,"EJECTED");
}
FPRINTF(FP2,"\N*********************************************\N\N");
}
FCLOSE(FP1);
FCLOSE(FP2);
RETURN;
}
```

APPENDIX 3

ANOVA and Turkey T Test

Statistical tests: ANOVA and Tukey T test

The ANOVA approach attempts to analyze the variation of a response to each independent variables. In an experiment of measuring performance through various factor levels, several blocks of performance may be achieved. In this thesis , replenishing intervals represent the blocks, and operational strategies(e.g., shared and dedicated) represent the factor levels. The objective of ANOVA is to show whether assigning the obtained results to different factor levels is significant. Figure A3.1 illustrates an example of factor levels and blocks in a performance measurement.

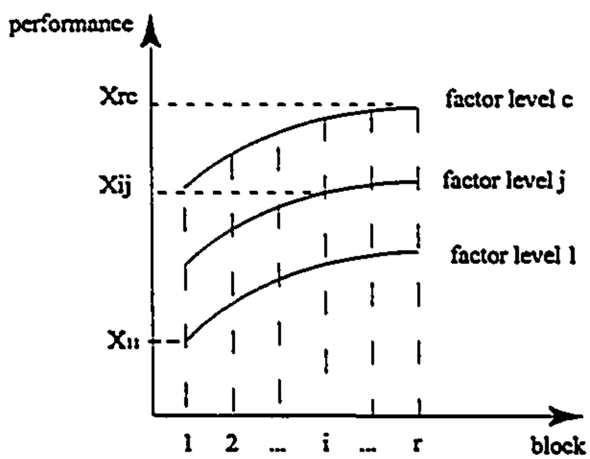


Figure A3.1 Different factor levels of performance measurement

As it is shown in the above figure, every X_{ij} represents the performance measure of factor level j at block i. Let:

r = number of blocks

c = number of factor levels

$$\mu_{i\cdot} = \text{the average of the } i\text{th block} = \frac{\sum_{j=1}^{c} X_{ij}}{c}$$

$$\mu_{\cdot j} = \text{the average of the } j\text{th factor level} = \frac{\sum_{i=1}^{c} X_{ij}}{r}$$

$$\mu = \text{the overall mean} = \frac{\sum_{j=1}^{c} \mu_{\cdot j}}{c} = \frac{\sum_{i=1}^{c} \mu_{i\cdot}}{r} = \frac{\sum_{i=1}^{r}\sum_{j=1}^{c} X_{ij}}{rc}$$

The objective is to show that for every j the $\mu_{\cdot j}$ is significantly greater than $\mu_{\cdot j-1}$. For this purpose two tests will be applied. In first attempt, the null hypothesis of no treatment effects against the opposite alternative will be tested. That is:

H_0: $\mu_{\cdot 1} = \mu_{\cdot 2} = \ldots = \mu_{\cdot j} = \ldots = \mu_{\cdot c}$

H_1: not all $\mu_{\cdot j}$ are equal

Once the equality of factor levels is rejected, the second test will be applied. This time the objective is to test whether the differences among all $\mu_{\cdot j}$ are significant or not. For either of these two tests the following computation formulas are needed:

SSA = variation among factor levels

SSE = experimental error variation

MSA = mean square among factor levels

MSE = mean square error

$$SSA = \sum_{i=1}^{r}\sum_{j=1}^{c}(\mu_{.j} - \mu)^2 = r\sum_{j=1}^{c}\mu_{.j}^2 - n\mu^2$$

$$SSE = \sum_{i=1}^{r}\sum_{j=1}^{c}(X_{ij} - \mu_{.j} - \mu_{i.} + \mu)^2$$

$$MSA = \frac{SSA}{c-1}$$

$$MSE = \frac{SSE}{(r-1)(c-1)}$$

a) Test of equality(ANOVA test)

As it is mentioned earlier, the objective of this test is to show that the factor levels are different. So following hypotheses can be established.

H_0: $\mu_{.1} = \mu_{.2} = \ldots = \mu_{.j} = \ldots = \mu_{.c}$

H_1: not all $\mu_{.j}$ are equal

In this thesis, maximum value of c is equal to 3(i.e., three tool configurations) the rejection of H_0 justifies the goal in this test. So, the rejection in this test is desired. The F statistic would be formed as follows:

$$F_A = \frac{MSA}{MSE} \sim F_{c-1,\,(r-1)(c-1)}$$

The null hypothesis would be rejected at the $\alpha\%$ level of significance if the computed F_A takes place in the critical region shown in Figure A3.2

H_0 rejection condition:

$F_A \geq F_{1-\alpha;\,c-1,\,(r-1)(c-1)}$

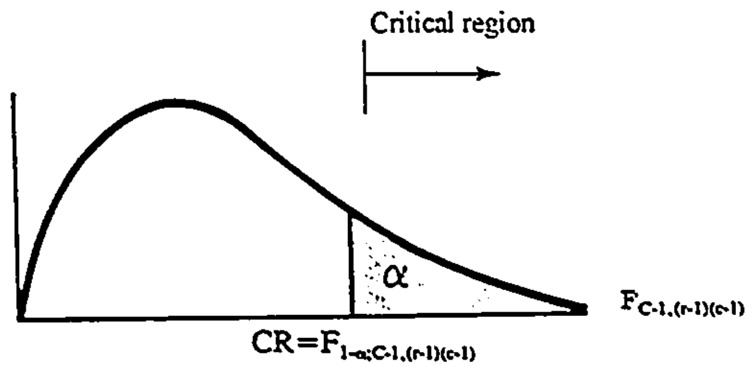


Figure A3.2 The critical region of null hypothesis

Note that $F_{1-\alpha;\,c-1,\,(r-1)(c-1)}$ is the tabulated upper-tail critical value from the F distribution having c-1 and (r-1)(c-1) degrees of freedom (reference A1).

b) Test of significance(Tukey T test)

After applying the above test, those results whose equality among the factor levels can not be rejected, are not valuable, however, some results show inequality among the levels by rejection the above test. In the recent case, in order to determine whether all $\mu_{.j}$

are significantly different from each other, the Tukey T method or the Scheffé S method can be used, depending on pairwise comparisons or all possible comparisons, respectively. In this study since there are only two factor levels(i.e., shared and dedicated), either of these two methods can be applied. Here the Tukey T method will be discussed. In Tukey T method the critical region is computed from

$$CR = \text{critical Region} = Q_{1-\alpha;\, c,\, (r-1)(cI)} \sqrt[2]{\frac{MSE}{r}}$$

where $Q_{1-\alpha;\, c,\, (r-1)(c-1)}$ is the tabulated upper-tail critical value from the Studentized Range distribution having c and $(r\text{-}1)(c\text{-}1)$ degree of freedom(reference A2).

The factor level j would be declared significantly greater than factor level j' if $\mu_{.j} - \mu_{.j'}$ takes place in the critical region.

References:

A1 E.S. Pearson, H.O. Hartley, "Biometric Tables for Statistician", Vol. 1, (1966)

A2 M. Merrington, C.M. Thompson"Tables of percentage points of the interval beta (F) distribution", Vol. 33 (1943).

E.S. Pearson and H.O. Hartley, "Biometrika tables for statisticians", Vol.1 (1954)

APPENDIX 4

Summary of simulation results

Tool configuration	Loading policy	Replenishing policy	Performance measure	Tooling policy	Means of replications for replenishing intervals(hrs)				
					3	6	9	12	15
Identical	Batch	Preventive	Mean-flow-time	Shared	1109	1272	1634	1996	2366
				Dedicated	1142	1439	1966	2440	3134
			Makespan	Shared	1275	1471	1926	2341	2651
				Dedicated	1317	1637	2361	2975	3658
		Failure	Mean-flow-time	Shared	1118	1265	1833	1964	2345
				Dedicated	1141	1436	1865	2486	3111
			Makespan	Shared	1296	1496	2028	2525	3075
				Dedicated	1317	1731	2361	3200	3711
	Random	Preventive	Utilization	Shared	0.74	0.74	0.69	0.6	0.53
				Dedicated	0.74	0.71	0.6	0.49	0.41
			Throughput	Shared	387.1	387.4	355.8	298.8	247
				Dedicated	387.3	365.2	288.7	229.9	195.6
		Failure	Utilization	Shared	0.74	0.74	0.68	0.6	0.53
				Dedicated	0.74	0.71	0.61	0.5	0.4
			Throughput	Shared	386.8	365.2	348.7	296.4	248
				Dedicated	386.6	363.7	300.5	234.5	186.5
Non-Identical	Batch	Preventive	Mean-flow-time	Shared	1331	1411	1630	1968	2256
				Dedicated	1396	1807	2447	3235	3927
			Makespan	Shared	1823	1935	2100	2492	2935
				Dedicated	1917	2577	3615	4940	5699
		Failure	Mean-flow-time	Shared	1343	1366	1571	1926	2331
				Dedicated	1363	1800	2469	3096	4160
			Makespan	Shared	1824	1900	2181	2655	3301
				Dedicated	1922	2775	4016	4941	6490
	Random	Preventive	Utilization	Shared	0.74	0.74	0.68	0.62	0.53
				Dedicated	0.74	0.87	0.55	0.45	0.39
			Throughput	Shared	387.4	387.4	347.3	305.2	249.5
				Dedicated	386	339	265	295	174
		Failure	Utilization	Shared	0.75	0.74	0.64	0.61	0.52
				Dedicated	0.74	0.87	0.56	0.45	0.37
			Throughput	Shared	388.2	384.5	346.1	295.1	242.2
				Dedicated	386.5	338.7	267.1	210.3	164.8

Table A4.1 Tabulated simulation results

Tool configuration	Loading policy	Replenishing policy	Performance measure	Tooling policy	Means of replications for replenishing intervals(hrs)				
					3	6	9	12	15
Hybrid	Batch	Preventive	Mean-flow-time	Dedicated	1166	1280	1560	2004	2322
				Shared	1191	1502	2022	2569	3188
			Makespan	Shared	1469	1498	1825	2460	2926
				Dedicated	1503	1965	2797	3480	4205
		Failure	Mean-flow-time	Shared	1167	1285	1555	1829	2185
				Dedicated	1186	1425	1885	2529	3039
			Makespan	Shared	1465	1544	2035	2358	2940
				Dedicated	1480	1884	2462	3388	4193
	Random	Preventive	Utilization	Shared	0.74	0.73	0.68	0.61	0.54
				Dedicated	0.91	0.81	0.61	0.47	0.38
			Throughput	Shared	387.6	384	349.1	297.3	254.7
				Dedicated	446	382.6	278.8	207.8	172.3
		Failure	Utilization	Shared	0.74	0.74	0.71	0.63	0.55
				Dedicated	0.74	0.7	0.6	0.49	0.41
			Throughput	Shared	387	388.6	364.2	311.5	283
				Dedicated	388.7	362.3	295.9	230.8	192

Table A4.1 Continue

Tool configuration	Loading policy	Replenishing policy	Performance measure		Means of replications for five loading levels				
					1	2	3	4	5
Identical	Random	Failure	Utilization	Shared	0.45	0.56	0.68	0.77	0.83
				Dedicated	0.45	0.54	0.61	0.63	0.63
			Throughput	Shared	235	291	348	366	354.3
				Dedicated	235	281.7	300.5	302.7	295.8
Non-Identicl	Random	Failure	Utilization	Shared	0.45	0.56	0.68	0.79	0.89
				Dedicated	0.45	0.5	0.56	0.57	0.57
			Throughput	Shared	236	291	346	375	379
				Dedicated	236.6	247.1	267.1	265.5	261.2
Hybrid	Random	Failure	Utilization	Shared	0.45	0.56	0.67	0.77	0.85
				Dedicated	0.43	0.51	0.57	0.6	0.59
			Throughput	Shared	237.3	292	337	363	368
				Dedicated	223.3	259.2	269.5	281.4	269.9

Table A4.1 Continue