Fall 1-31-1995

Scheduling of flexible manufacturing systems with automated guided vehicles using petri net models

Hua-Sheng Chiu
New Jersey Institute of Technology

ABSTRACT

SCHEDULING OF FLEXIBLE MANUFACTURING SYSTEMS WITH AUTOMATED GUIDED VEHICLES USING PETRI NET MODELS

by
Hua-Sheng Chiu

In this thesis, Petri net models for Flexible Manufacturing Systems (FMS) are constructed. A firing sequence of the Petri net from the initial marking to the final marking can be seen as a schedule of the modeled FMS. By using the branch-and-bound algorithm, an optimal schedule of the FMS can be obtained.

Automated Guided Vehicle Systems (AGVS) are increasingly used for material handling in factories and warehouses. An AGVS can reduce labor costs and is ready to be integrated into an automated factory. This thesis presents two AGVS models (shared and duty) which integrate the control of AGVS with the control of part processing facilities. Both types of AGVS are modeled by Petri nets. We want to compare the two AGVS in terms of systems performance and discuss which application is more suitable for each AGVS type.

We also want to consider and solve AGV jam problems. The objective of the AGV jam-free control module is to guarantee a jam-free condition among AGVs in an FMS. Results have been obtained and analyzed.

SCHEDULING OF FLEXIBLE MANUFACTURING SYSTEMS WITH AUTOMATED GUIDED VEHICLES USING PETRI NET MODELS

by
Hua-Sheng Chiu

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

January 1995

APPROVAL PAGE

SCHEDULING OF FLEXIBLE MANUFACTURING SYSTEMS WITH AUTOMATED GUIDED VEHICLES USING PETRI NET MODELS

Hua-Sheng Chiu

[illegible]

Dr. MengChu Zhou, Thesis Advisor — Date
Assistant Professor
Department of Electrical and Computer Engineering, NJIT

[illegible]

Dr. Anthony Robbi, Committee Member — Date
Associate Professor
Department of Electrical and Computer Engineering, NJIT

Dr. Edwin Hou, Committee Member — Date
Assistant Professor
Department of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Hua-Sheng Chiu

Degree: Master of Science in Electrical and Computer Engineering

Date: January 1995

Undergraduate and Graduate Education:

- Master of Science in Electrical and Computer Engineering
  New Jersey Institute of Technology, Newark, NJ 1995

- Bachelor of Science in Automatic Control Engineering
  Feng Chia University, Taichung, Taiwan, R.O.C. 1989

Major: Electrical and Computer Engineering

This thesis is dedicated to
my parents, parents-in-law, and wife.

ACKNOWLEDGMENT

I wish to express my sincere gratitude to my thesis advisor Dr. MengChu Zhou for his ingenious guidance and encouragement.

Special thanks to Dr. Hou and Dr. Robbi serving as members of the committee.

I also thank Mr. H. Henry Xiong's help in programming the branch-and-bound algorithm.

I would like to thank my parents and parents-in-law, because who made it possible for me to continue my studies in the United States.

I also wish to thank my wife for her love and support.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

CHAPTER ONE

INTRODUCTION

1.1 Flexible Manufacturing Systems (FMS)

The word "flexible" is used to describe objects "capable of responding or conforming to changing or new situations." Flexibility is therefore the property that makes an object "flexible". In the context of manufacturing systems, flexibility is widely accepted to imply the "··· ability of a system to cope with changes". It is important for companies which compete in today's market.

Flexible manufacturing systems, having won recognition for their potential to revive ailing manufacturing industries, have been installed by many firms to produce such items as automobiles, aircrafts and machine tools. The growth of flexible automation has been propelled by advances in computerized manufacturing technology coupled with the need for shorter production runs, greater responsiveness to demand changes, customized production, and superior control of the production processes. Advocates claim that FMS offer a unique ability to simultaneously accomplish two hitherto irreconcilable objectives: high productivity and the ability to respond to changing markets. Thus, FMS deliver economies generally associated with the automation of mass production to small-batch manufacturing.

A key feature of FMS is their "flexibility". In fact, it is this quality that distinguishes them from the traditional high-volume, process-dedicated production systems like automated transfer line [Gupta 1989].

Manufacturing industries are under great pressure because of the rising costs of energy, material, labor, capital, and intensifying worldwide competition. While these trends will remain for a long time, the problems facing manufacturing today run much deeper. In many cases they stem from the very nature of the manufacturing process itself.

Flexible manufacturing systems are regarded as one of the most efficient methods used in reducing or eliminating problems in the manufacturing process. FMS is more than a technical solution; it is a business-driven solution leading to improved profitability through reduced lead times and inventory levels, rapid response to market changes, lower staffing levels, and improved manufacturing effectiveness through increased operational flexibility, predictability, and control.

FMS provides new hope to the manufacturing industries and will continue to be looked on not as an end in itself, but as a dynamic, evolving process to keep the manufacturing world alive and well in the years ahead.

1.2 What are Petri Nets ?

Consider an automated manufacturing system. The goal is to make a set of products from raw materials and purchased parts using resources such as machines, robots, Automated Guided Vehicles (AGVs), materials handling and storage devices. A complex set of activities must occur in order to meet this goal. The need exists in manufacturing to properly coordinate and synchronize these activities and resources which work concurrently to produce a set of products. This is the manufacturing control design problem.

Petri net theory was originally developed by Carl Adam Petri and presented in his doctoral dissertation in 1962. Petri nets are a graph theoretic as well as a visually graphical tool specifically designed for modeling, analysis, performance evaluation and control of interacting concurrent discrete event systems. These types of systems are characterized by a discrete state space and are event driven, as opposed to time dependent. Often, these events are asynchronous. They are sometimes called Discrete Event Systems (DES) or Discrete Event Dynamic Systems (DEDS). Examples of DEDS are manufacturing systems, communication networks, and computer-based systems.

The following will introduce Petri nets informally using the simple robotic example in Figure 1. In this example, we want to show the sequence of conditions and events that must occur for the robot to move two parts. We can describe a Petri net (PN) graphically as having several elements:

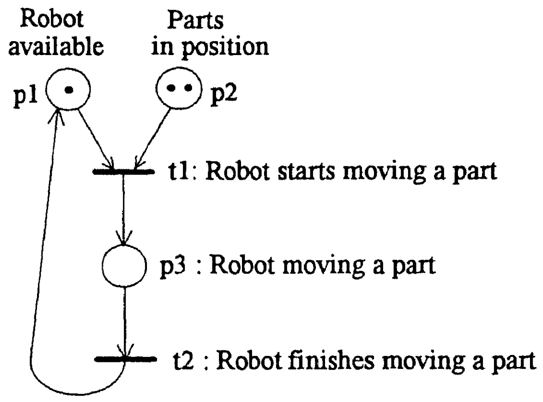

Robot available
Parts in position
p1
p2
t1: Robot starts moving a part
p3 : Robot moving a part
t2 : Robot finishes moving a part

Figure 1.1 A simple Petri net example

Places (circles) which represent conditions. In this thesis they represent resource availability or process status. In Figure 1.1, places p_1 and p_2 indicate the availability status of the robot and parts respectively, and p_3 indicates the operational status of the robot.

Transitions (bars) which represent events. In this thesis they represent the starting and stopping of processes. In Figure 1.1, t_1 represents the start of robot moving a part; t_2 the end of the robot moving the part.

Input functions that define arcs (arrows) from places to transitions. The arc from p_1 to t_1 defines p_1 as an input place to transition t_1; similarly the arc from p_2 to t_1 defines p_2 as an input place to t_1. The arc from p_3 to t_2 defines p_3 as an input place to t_2.

Output functions that define arcs (arrows) from places to transitions. The arc from t_1 to p_3 defines p_3 as an output place from t_1 and the arc from t_2 to p_1 defines p_1 as an output place from t_1.

These four elements define the structure of an ordinary Petri net. The state of a Petri net is indicated by it's

Marking, a vector whose component is the number of tokens (dots) in the corresponding place. In general, there could be more than one token per place. The initial marking for the PN in Figure 1.1 is one token in place p_1, two tokens in p_2, and zero token in p_3, so the marking vector is $mo=(1\ 2\ 0)^T$ [DiCesare 1991].

Petri nets are an approach for modeling, control, and performance analysis of automated manufacturing systems. This approach has become more important in recent years because it can solve problems that cannot be modeled. Petri nets are a very valuable performance analysis tool.

Petri nets were originally developed to model asynchronous, concurrent processes in communication and computer systems and have been used to model various types of discrete event systems. By introducing time delays associated with transitions and/or places in net models, they can capture the system actual temporal behavior. Petri nets are a promising tool for describing and studying flexible manufacturing systems. As a graphical tool, Petri nets can be used as a visual-communication aid similar to flow charts. As a mathematical tool, it is possible to set up mathematical models governing the behavior of systems and derive system performance indices [Xiong 1994].

Petri nets are useful tools for the modeling and analysis of a production system. It can provide accurate models of the precedence relations and concurrent, asynchronous events. In this thesis, a Petri net model is built to model the detailed behavior of an FMS and a schedule which is generated in a Petri net framework optimizes some a priori-assigned performance criterion [Sun 1993].

1.3 Problem Description

Production scheduling concerns the efficient allocation of resources over time for the manufacturing of goods. Scheduling problems arise whenever a common set of resources — material, machine, and robot — must be used to make a variety of different times under the same products. The objective of scheduling is to find a way to assign and sequence the use of these shared resources such that production constraints are satisfied and the time spent is minimized.

Interest in new approaches to production scheduling has been stimulated by a variety of pragmatic and theoretical consideration. First among these has been the increasingly competitive world markets for manufactured goods. Better production schedules provide a competitive advantage through gains in resource productivity and related efficiencies in operations management. Competition has also motivated the introduction of sophisticated and capital intensive new manufacturing systems made possible by the declining cost and increasing power of industrial computers and robots. Most notable among the new manufacturing technologies are systems for automated, flexible, agile, and computer-integrated manufacturing. These new systems have created a range of new operational problems, further speeding up the pace of scheduling research.

Among theorists the development of complexity theory and maturation of artificial intelligence have begun to redirect the body of scheduling research. Sequencing and scheduling theory long has been preoccupied with the design of constructive solutions and optimization algorithms for highly simplified machine-scheduling problems. These problems and procedures appear to be more robust than optimization-based machine scheduling and for this reason hold great promise for commercial adaptation. Taken as a whole, current market, technological, and theoretical developments have made solutions to both long-standing and newly emerging scheduling problems the subject of intense applied and theoretical research.

Recent advances in the theory and practice of production scheduling transcend traditional disciplinary boundaries. As a consequence, the scheduling literature has escaped its traditional locus in operations research, management science, and industrial engineering. Production research has recently been reported in proceedings and journals principally concerned with control theory, artificial intelligence, system simulation, and other branches of engineering and computer science. The sheer diversity and momentum of activity has made developments in production scheduling increasingly difficult to track and assimilate.

Production scheduling is the allocation of available production resources over time to meet some set of performance criteria. Typically, the scheduling problem involves a set of jobs to be completed, where each job comprises a set of operations to be performed. Operations require machines and material resources and must be performed according to some feasible technological sequence. Scheduling is influenced by such diverse factors as job properties, due-date requirements, release dates, cost restrictions, machine availabilities, machine capabilities, operation precedences, resource requirements, and resource availabilities. Performance criteria typically involve trade-offs between holding inventory for task, frequent production changeovers, satisfaction of production-level, and satisfaction of due dates.

Developing a production schedule involves selecting a sequence of operations (or process routing) that will result in the completion of a job, designating the resources needed to execute each operation in the routing, and assigning the times at which each operation in the routing will start and finish execution. Routings and resource assignments typically are the product of process planning. Scheduling generally refers to the activity of time tabling operations.

The formal job-shop or machine-scheduling problem may be stated as follows : N jobs are to be processed on M machines. Each job consists of a set of M operations, one operation uniquely associated with each of the M machines. The processing time for an operation cannot be split. Technological constraints demand that the operations within each job must be processed in a unique order. The scheduling problem involves determining the sequence and timing of each operation on each machine such that some given performance criterion is maximized or minimized. Typical performance criteria include minimizing the makespan (i.e., minimizing the time required to complete all of the jobs) [Rodammer 1988].

1.4 Motivation

Questions of schedule are perhaps not as vital as the decisions that determine what tasks are to be scheduled or how the task is to be performed once its turn has arrived. Nevertheless, if proper selection of schedule can yield some incremental improvement, it seems pointless to neglect the opportunity.

To illustrate the art of production scheduling, consider a specific example of the traditional, single-stage, job-shop, or machine scheduling problem [Rodammer 1988]. Example 1 is an FMS that has two machines M1 and M2. Two job types J1 and J2 are to be carried out and Table 1.1 shows the requirement for each jobs [Conway 1967]. The first operation of Job 1 can be carried out at Machine 1 and needs 4 unit time. The second operation of Job 1 can be carried out at Machine 2 and needs 1 unit time. The first operation of Job 2 can be carried out at Machine 1 and needs 1 unit time. The second operation of Job 2 can be carried out at Machine 2 and needs 4 unit time.

An operation $O_{i,j,k}$ represents the j-th operation of the i-th job type being performed at the k-th machine. Schedule 1 is shown in Figure 1.2(a). It means that Machine 1 processes the first operation of Job 1 (Machine 2 is idle), then Machine 1 processes the second operation of Job 1 and Machine 2 processes the first operation of Job 2 at the same time, then Machine 2 processes the second operation of Job 2 (machine 1 is idle). The makespan of schedule 1 is 9 Units. Schedule 2 is shown in Figure 1.2(b). It means that Machine 1 processes the first operation of Job 2 (Machine 2 is idle), then Machine 2 processes the second operation of Job 2 and Machine 1 processes the first operation of Job 1 at the same time, then Machine 2 processes the second operation of Job 1 (Machine 1 is idle). The time makespan of schedule 2 is 6 Units. There is a big difference between these results. Thus, you can know obviously how important the scheduling problems are.

Under the usual assumptions that all jobs are ready at the start and that all operations require the exclusive use of a machine and cannot be split. Schedule 2 represents the sequence of operations which minimizes the total time required to complete the processing of all jobs (the makespan).

Table 1.1 Job requirements of Example 1

Order	1 st	2 nd
Job 1	Machine 1 (4U)	Machine 2 (1U)
Job 2	Machine 1 (1U)	Machine 2 (4U)

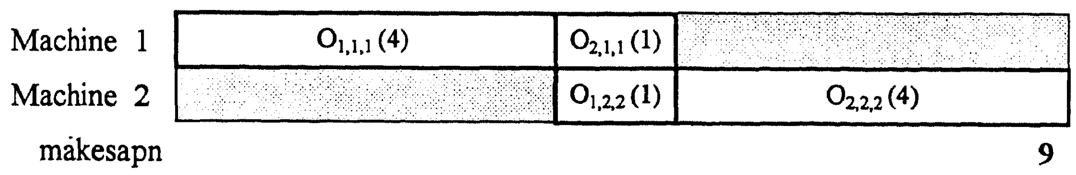


Figure 1.2(a) Schedule 1 of Example 1

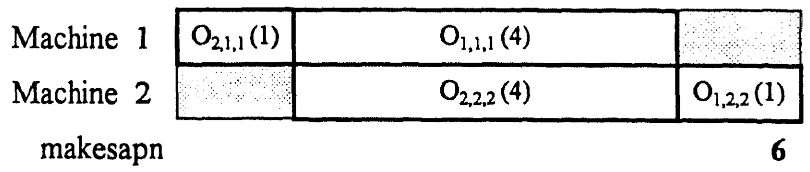


Figure 1.2(b) Schedule 2 of Example 1

Note that the required ordering of operations within each job (the technological sequence) is preserved and that the ordering of operations on the machines has been selected so as to achieve the desired objective [Rodammer 1988].

Automated Guided Vehicle Systems (AGVS) are increasingly used for material handling in factories and warehouses. An AGVS can replace conveyors and can accommodate new routes with relative ease. An AGVS can also reduce labor costs and is ready to be integrated into an automated factory. This thesis presents two AGVS models (shared and duty) which integrate the control of AGVS with the control of part processing facilities [Lee 1993]. These two types of AGVS are modeled by Petri nets. This thesis also compares the two AGVS and discusses which case is suitable for each one type.

1.5 Objectives

This thesis emphasizes the Petri net approach for modeling, control, and performance analysis of flexible manufacturing systems (FMS). This approach has become more important in recent years because it can solve problems that cannot be modeled using queuing theory, and avoid the time consuming, trial and error approach of simulation.

The mathematical model of an FMS cell has been synthesized using Petri nets. The concurrency, conflicts, resource-sharing, and sequential operations have been built in the net models. The resulting net models have the system properties such as liveness, safeness, and reversibility. Application of Petri net theory to practical FMS is a very important area in order to develop modern manufacturing control systems [Zhou 1993].

The purpose of this thesis is to demonstrate how Petri nets can be used in design of flexible manufacturing systems. Five objectives are as follows:

1) To present the branch-and-bound algorithm to solve the scheduling problem.
2) To discuss modeling, analysis, and control issues of FMS with Petri nets and summarize the Petri net design approaches proposed in the literature.
3) To analyze the system cycle time based on the resulting Petri net model.
4) To discuss the automated guided vehicle system (AGVS).
5) To solve the collision and traffic jam problems of vehicles.

CHAPTER TWO

DEVELOPMENT OF SCHEDULING ALGORITHM

2.1 Definition of Petri Nets

Mathematical modeling or synthesis of FMS is the first step to analyze, simulate, and control the operations of such systems. When concurrency and synchronization concepts become keys for study of modern manufacturing systems control, Petri nets provide a most straightforward tool to model these concepts. In addition, they can model conflicts, nondeterminism, time information, and resource-sharing environments.

Petri nets are a graphical tool where places, pictured as circles, are used to represent availability of resources, operation processes, or conditions and transitions, pictured as bars, model the events, start, or termination of operations. Arcs indicated the relationship between places and transitions. Tokens in places and their flow regulated by firing transitions add the dynamics to Petri nets. Thus a marked Petri net can be used to study dynamic behavior of the modeled discrete event systems [Zhou 1993].

For a fundamental knowledge of Petri net theory, a reader is referred to [Murata 1989]. Strictly, a Petri net may be defined as $Z = (P,T,I,O,m)$, where

(1) $P = \{p_1, p_2, \ldots, p_n\}$, $n > 0$;

(2) $T = \{t_1, t_2, \ldots, t_s\}$, $s > 0$;

(3) $I : P\times T \rightarrow \{0,1\}$;

(4) $O: P\times T \rightarrow \{0,1\}$;

(5) $m: P \rightarrow \{0,1,2,\ldots\}$;

In this definition, p_i is called a place, t_i a transition, I an input function defining the set of directed arcs from P to T, O an output function defining the set of directed arcs from T to P, and m is an n-dimensional marking whose ith component represents the number of tokens in place p_i. $m(p)$ denotes the number of tokens in place p.

Graphically, places are represented by circles and transitions by bars. If I(p,t) = 1, we include a directed arc from place p to transition t, and O(p,t) = 1, a directed arc from transition t to place p. A marking assigns to each place a nonnegative integer. If a marking assigns to place p a nonnegative integer k, we say that p has k tokens, and we put k black dots (tokens) in place p.

The behavior of many systems can be described in terms of systems states and their changes. In order to simulate the dynamic behavior of a system, a state or marking in a Petri net is changed according to the following transition (firing) rules:

(1) A transition is enabled if $m(p_i) \geq I(p_i,t_j)$ for any $p_i \in P$;

(2) An enabled transition t can fire at marking m', and its firing yields a new marking,

$m(p) = m'(p) + O(p,t) - I(p,t)$, for arbitrary p from P.

The marking m is said to be reachable from m'. Given Z and its initial marking m_0, the reachability set is the set of all marking reachable from m_0 through various sequences of transition firings and is denoted by $R(Z,m_0)$. Reachability set is a fundamental basis for studying the dynamic properties of a system.

A Petri net (Z,m_0) is said to be K-bounded or simply bounded if the number of tokens in each place does not exceed a finite number K for any marking reachable from m_0. A Petri net (Z,m_0) is said to be safe if it is 1-bounded. For bounded Petri net, from the initial marking m_0, we can obtain all possible reachable markings through firing all enabled transitions.

A Petri net (Z,m_0) is said to be live if, no matter what marking has been reached from m_0, it is possible to ultimately fire any transition of the net by progressing through some further firing sequence. A live Petri net guarantees deadlock-free operation, no matter what firing sequence is chosen. The application to manufacturing is direct because, in general, the discrete event control should not lead to deadlock or at least all possibilities for deadlock should be known.

A Petri net (Z,m_0) is said to be reversible if, for each marking m in $R(Z,m_0)$, m_0 is reachable from m. Therefore, in a reversible net one can always get back to the initial marking.

2.2 Petri Net Modeling

A certain order of activities needs to be followed by each job in FMS. For example, the activity sequence {operation 1, operation 2} should be followed by each job. Therefore, for Petri net modeling, the first important issue is the modeling of sequential activities for each job in the system.

The second modeling issue is synchronization. For example, Machine 1 will process material piece 1 only when it is present. It will never finish the process operation if the material is missing.

The third issue is modeling of concurrence. By concurrence we mean that there are parallel relationships among the concerned events. For example, two physical events 1) Machine 1 processes the first operation of Job 1, 2) Machine 2 processes the second operation of Job 2, are concurrent if both events may occur simultaneously. Two machines can operate concurrently if both can process tasks at the same time. High concurrency among system resources often implies high productivity.

The fourth modeling issue we are concerned with conflict, when the sharing of machines is encountered. The mutual exclusion concept is especially useful to model the characteristics of this situation.

Another modeling issue is priority. For example, the first operation of Job 1 is executed first, and the first operation of Job 2 is executed second.

The Petri net models must take the various issues as discussed above into consideration. The usual approach is to create a Petri net model, with which to analyze critical properties of interest. A more rigorous approach for Petri net modeling is to synthesize a Petri net of a system which has desirable properties such as boundedness and deadlock freeness.

Examples of Petri net models for linear sequence, synchronization, concurrency, and mutual exclusion are shown in Figure 2.1 [Xiong 1994].

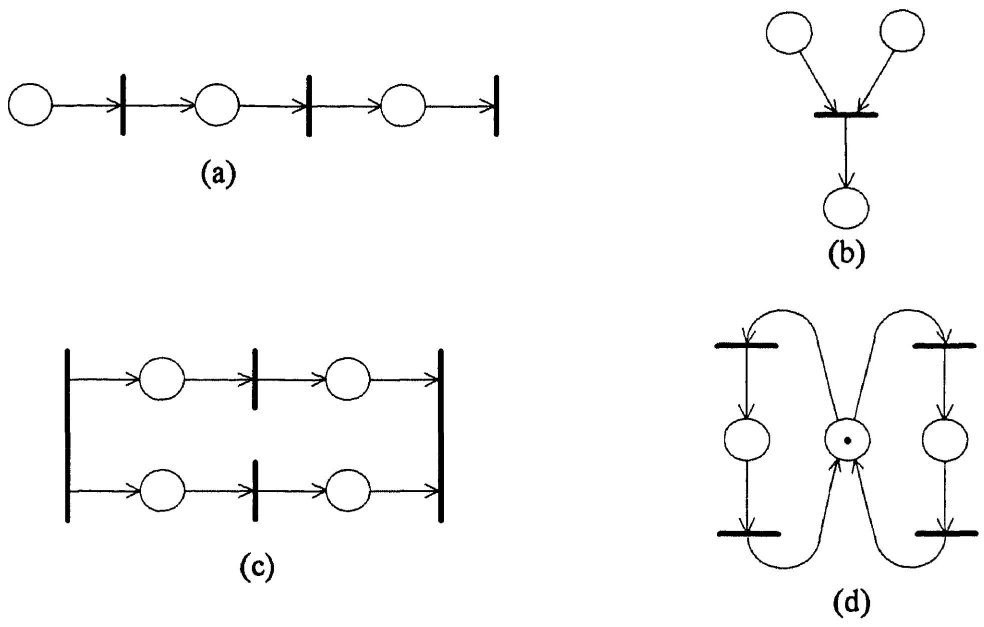


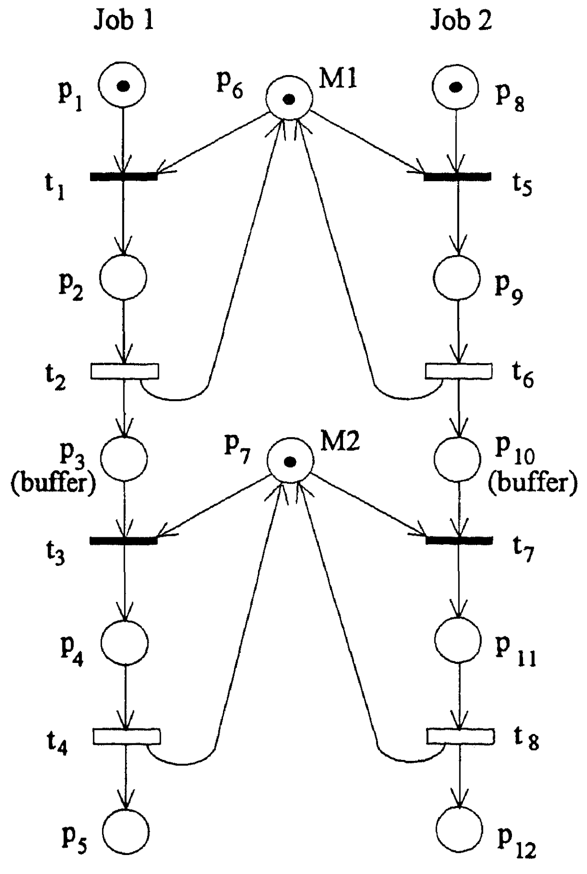
Figure 2.1 Examples of Petri net models for (a) linear sequence, (b) synchronization, (c) concurrency, and (d) mutual exclusion

Figure 2.2 is the Petri net model of Example 1. If we want to execute the first operation of Job 1, the Job 1 material and Machine 1 must be available. Thus, we put the arcs from p_1 and p_6 to t_1. If we want to execute the second operation of Job 1, the first operation of Job 1 must be finished and Machine 2 must be available. Thus, we put the arcs from p_3 and p_7 to t_3. The same method is applied to Job 2. Table 2.1 shows the description of places and transitions of Figure 2.2.

Because Job 1 and Job 2 share Machine 1 and Machine 2, so they are the mutual exclusive events. Two 2-parallel mutual exclusions (p_6, $\{t_1,t_2\} \cup \{t_5,t_6\}$) and ($p_7$, $\{t_3,t_4\} \cup \{t_7,t_8\}$) are used in this model by using the concepts presented in [Zhou 1991].

We picture the transitions t_1, t_3, t_5, and t_7 as solid bars because they are immediate transitions. The transitions t_2, t_4, t_6, and t_8 are pictured as open bars because they are timed transitions. Table 2.2 shows the operation times of transitions. Places p_3 and p_{10} model buffer spaces.

The initial marking $m_0=(1,0,0,0,0,1,1,1,0,0,0,0)^T$ where the values in the vector are associated with the series of places $p_1, p_2, \dots, p_{12}$. The value is 1 if the place is initially marked with a token, otherwise it is zero. The final marking is $(0,0,0,0,1,1,1,0,0,0,0,1)^T$. If a marking equals to the final marking, then we find a feasible solution.



initial marking : $(1,0,0,0,0,1,1,1,0,0,0,0)^T$
final marking : $(0,0,0,0,1,1,1,0,0,0,0,1)^T$

Figure 2.2 The Petri net model for Example 1

Table 2.1 Description of places and transitions of Figure 2.2

p_1	Job 1 material	t_1	The first operation of Job 1 can be processed
p_2	Job 1 material waiting the first operation	t_2	The first operation of Job 1 is processing
p_3	Job 1 material finished the first operation	t_3	The second operation of Job 1 can be processed
p_4	Job 1 material waiting the second operation	t_4	The second operation of Job 1 is processing
p_5	Job 1 material finished the second operation	t_5	The first operation of Job 2 can be processed
p_6	Machine 1 is available	t_6	The first operation of Job 2 is processing
p_7	Machine 2 is available	t_7	The second operation of Job 2 can be processed
p_8	Job 2 material	t_8	The second operation of Job 2 is processing
p_9	Job 2 material waiting the first operation		
p_{10}	Job 2 material finished the first operation		
p_{11}	Job 2 material waiting the second operation		
p_{12}	Job 2 material finished the second operation		

Table 2.2 Operation times of Figure 2.2

Operation	time
t_1	0
t_2	4
t_3	0
t_4	1
t_5	0
t_6	1
t_7	0
t_8	4

2.3 Branch and Bound Algorithm

Branch-and-bound methods have been developed in a variety of contexts, and under a variety of names, such as "backtrack programming" and "implicit enumeration". Essentially, the idea is to repeatedly break the set of feasible solutions into subsets, and to calculate bounds on the costs of the solutions contained within them. The bounds are used to discard entire subsets of solutions from further consideration. The simple but effective technique has scored a number of notable successes in practical computations.

Branch-and-bound algorithms are always 'primal' in the sense that they proceed from one feasible solution to another until optimality is verified. In fact they often find optimal or near optimal solutions early in the enumeration process and spend the majority of the time verifying optimality. Thus the user can be comforted by the expectation that termination before proof of optimality will likely yield a very good solution if not an optimal one [Christofides 1978].

In this thesis, timed Petri nets are employed to model the aforementioned scheduling problem, and solve for an optimum schedule. The algorithm presented in this thesis combines the execution of the Petri net with a modified branch-and-bound scheme. Once the Petri net model of system is constructed, the evolution of the system can be described by changes in the marking of the Petri net. In other words, all possible behaviors of the system can be completely tracked by the reachability graph of the Petri net. Theoretically, therefore, an optimal schedule can be obtained by generating the reachability graph and finding the optimal path from the initial marking to the final marking. The path is a fired sequence of the transitions of the Petri net model.

Since the precedence relationships and resource constraints are incorporated in the Petri net model, only the feasible schedules would be obtained by executing the Petri net. Thus the search space is limited to that of feasible schedules. It should be pointed out that firing a transition during a search implies to explore a new schedule [Shen 1992].

The branch-and-bound algorithm that finds an optimal or near optimal schedule is as follows. Given a Petri net model, it expands the reachability graph from the initial marking until the generated portion of the reachability graph touches the final marking. Once the final marking is reached, the path is constructed. Then the transition sequence of the path provides the order of starts of the activities.

The Algorithm :

Step 1: Find the initial marking (m_o) and final marking (m_f).

Step 2: Set the upper bound (τ_{ub}) of makespan.

Step 3: Let current marking $m_c = m_o$.

Step 4: If there is no enabled transition, then go to step 15.

Step 5: Fire any one and put the others in waiting-list.

Step 6: Record this fired transition to fired-sequence.

Step 7: Generate the new marking (m_{new}) for this fired transition. Let current marking (m_c)= m_{new}.

Step 8: Calculate the actual time (τ_c) from m_o to m_c.

Step 9: If $\tau_c > \tau_{ub}$, then go to Step 15.

Step 10: If m_c has occurred before and if it may not make a shorter time ($\tau_c > \tau_b$), then go to Step 15.

Step 11: If $m_c = m_f$, then go to Step 13.

Step 12: Go to Step 4.

Step 13: We find a feasible solution. Calculate the time makespan (τ_t).

Step 14: If $\tau_t > \tau_{ub}$, then reject this solution.
If $\tau_t < \tau_{ub}$, then this solution is selected. Let $\tau_{ub} = \tau_t$.

Step 15: Go backward and get a transition from waiting-list. If there are no transitions in waiting-list, then terminate. Otherwise, fire this selected transition and go to Step 6.

In Step 2, use a large number as the upper bound to ensure that reasonably short makespan is not left out. In Step 5, the waiting-list is a data structure used to record information of alternative transitions. In the waiting-list, items are stored and retrieved in Last-In-First-Out (LIFO). Figure 2.3 shows the flowchart of branch-and-bound algorithm.

The algorithm is implemented in Fortran. See Appendix A for the source code. In order to find enabled transitions in Step 4, the implementation identifies marked places. For each marked place, the program identifies its output transitions. For each output transition the program checks if all its input places are marked. If they are, the transition is enabled. In Step 5, recording "waiting-list" is actually combined with "current marking" and "current executing time" on stack in order to remember the condition of branch node.

2.4 The Execution Result

We can write the formal mathematical expression of Example 1 according to the definition of Petri net (section 2.1).

1) $P=\{p_i,\ 1\leq i \leq 12\}$;

2) $T=\{t_i,\ 1\leq i \leq 8\}$;

3) $$I = \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 1&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&1&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0 \end{pmatrix};$$

4) $$O = \begin{pmatrix} 0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&1&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&1 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{pmatrix};$$

5) $m_0 = (1,0,0,0,0,1,1,1,0,0,0,0)^T$;

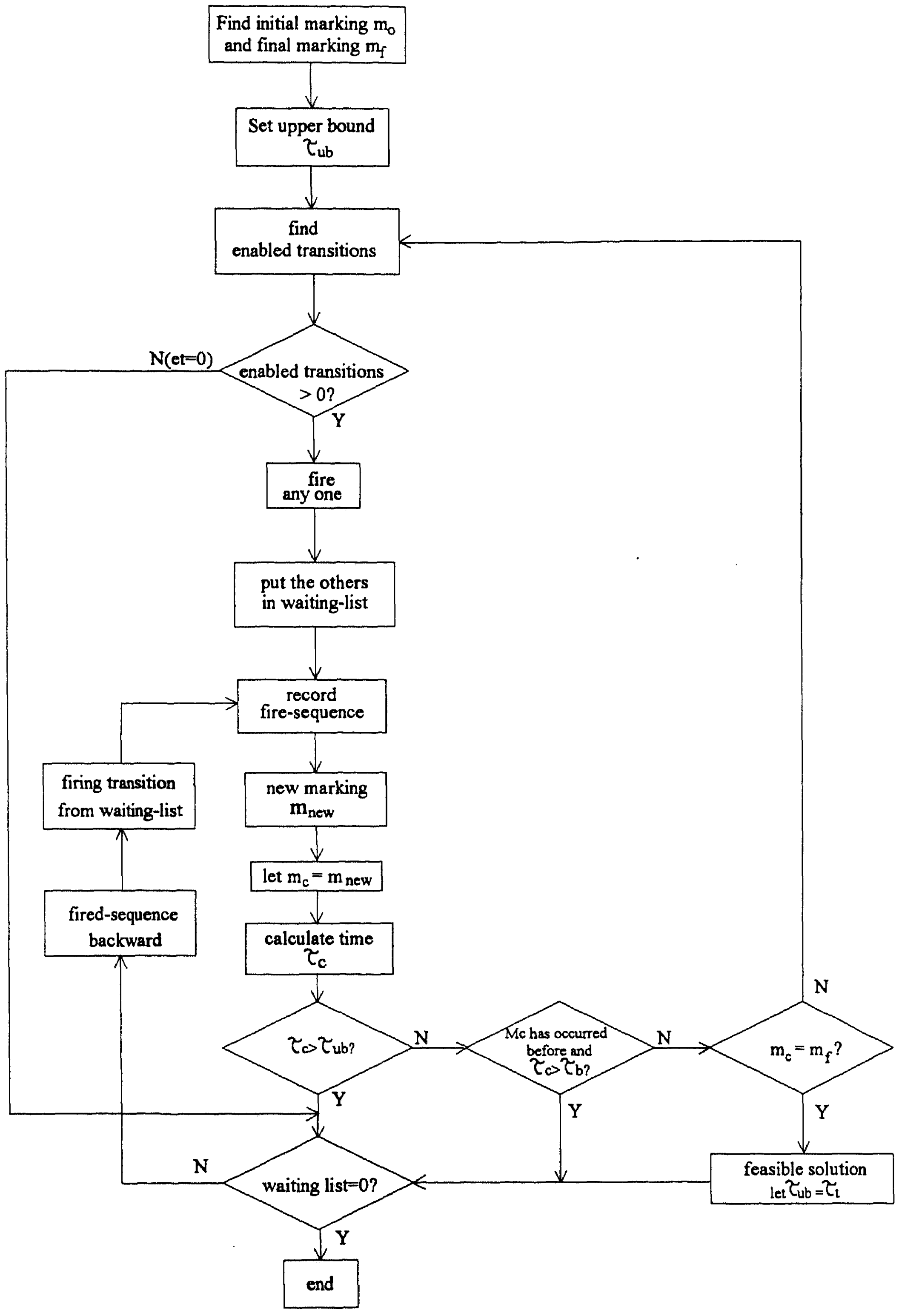


Figure 2.3 Flowchart of the branch-and-bound algorithm

We can easily find transitions t_1 and t_5 are enabled according to the transition firing rule (1): $m(p_i) \geq I(p_i,t_1)$ and $m(p_i) \geq I(p_i,t_5)$. Thus, we can fire either t_1 or t_5. If we choose to fire t_1, then we get the new marking = $(0,1,0,0,0,0,1,1,0,0,0,0)^T$ according to the transition firing rule (2) and put the transition t_5 to waiting-list. For example, $m(p_1) = m'(p_1) + O(p_1,t) - I(p_1,t) = 1 + 0 - 1 = 0$, $m(p_2) = m'(p_2) + O(p_2,t) - I(p_2,t) = 0 + 1 - 0 = 1$. $m(p_3)$ to $m(p_{12})$ can be obtained by using the same method. Therefore, we get the new marking=$(0,1,0,0,0,0,1,1,0,0,0,0)^T$. The operation time is zero (τ=0) because the operation time of t_1 is zero.

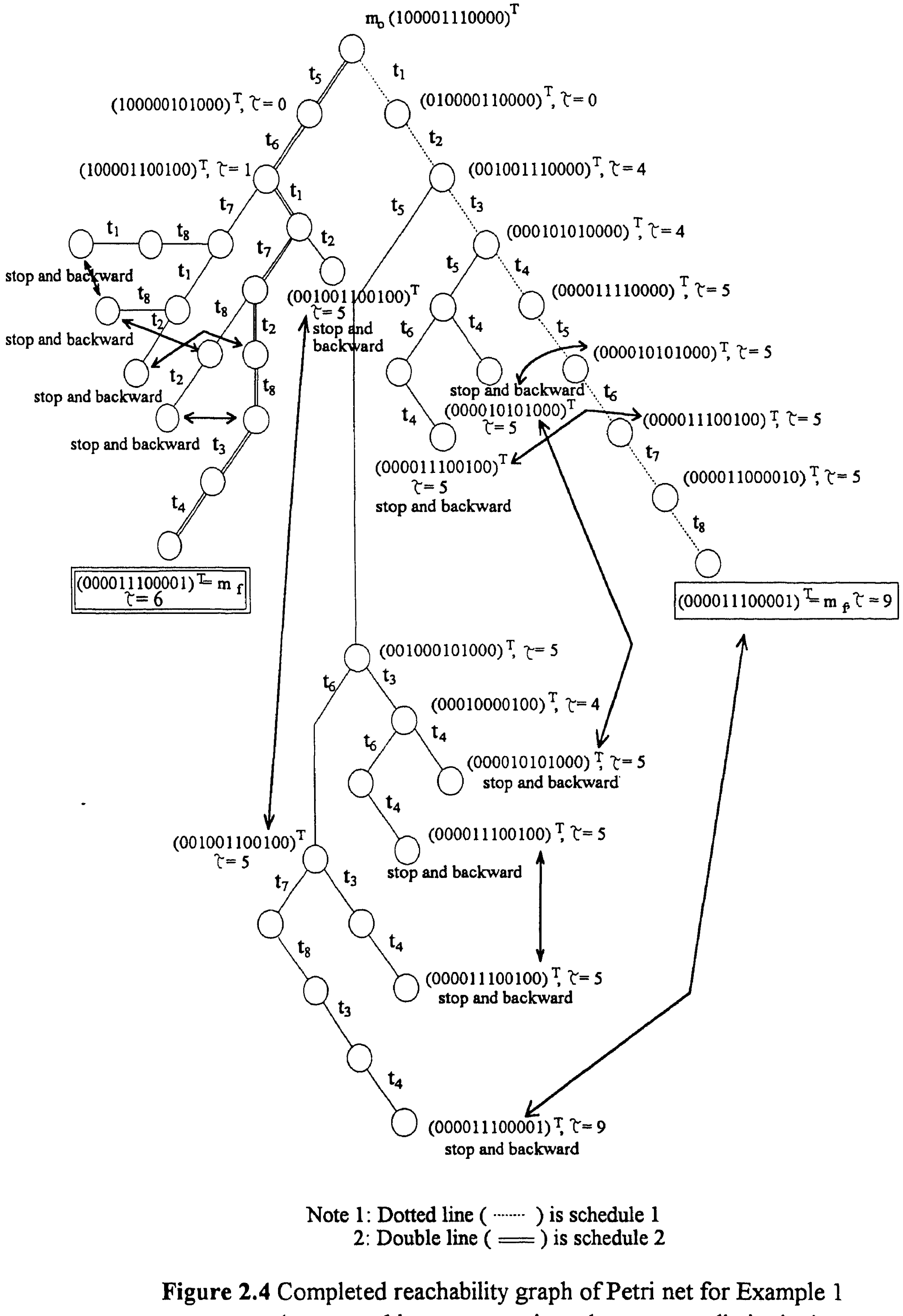
The new marking is $(0,1,0,0,0,0,1,1,0,0,0,0)^T$ after firing t_1. We can find only the transition t_2 is enabled, so we can only choose t_2 to fire. The new marking becomes $(0,0,1,0,0,1,1,1,0,0,0,0)^T$ and the operation time $\tau = 0 + 4 = 4$ because the operation time of t_2 is 4.

If a new marking has occurred before and it may not make a shorter operation time, then stop this branch and go backward to get another transition from waiting-list (as those bold lines shown in Figure 2.4).

We can get the execution results according to the Petri net and branch-and-bound algorithm. Figure 2.4 shows the completely reachability graph of Petri net for Example 1 (some markings are not given due to space limitation). The dotted line (…) is Schedule 1 (Figure 1.2(a)) and the double line (=) is Schedule 2 (Figure 1.2(b)) in Figure 2.4. It is obviously that Schedule 2 (makespan = 6U) is better than Schedule 1 (makespan = 9U). Thus, the optimal schedule is Schedule 2.

Table 2.3 The execution results of Example 1

	Makespan	Transition fired sequence
Schedule 1	9U	$t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8$
Schedule 2	6U	$t_5, t_6, t_1, t_7, t_2, t_8, t_3, t_4$



Note 1: Dotted line (········) is schedule 1
2: Double line (═══) is schedule 2

Figure 2.4 Completed reachability graph of Petri net for Example 1 (some markings are not given due to space limitation)

CHAPTER THREE

PERFORMANCE EVALUATION OF FMS USING MARKED GRAPHS

3.1 Marked Graphs

Marked graphs, a special class of timed Petri nets, are used for modeling and analyzing job-shop systems. The modeling allows for evaluating the steady-state performance of the system under a deterministic and cyclic production process. Given any fixed processing times, the productivity (i.e., production rate) of the system can be determined from the initial state. It is shown in particular that, given any desired product mix, it is possible to start the system with enough jobs in-process so that some machines will be fully utilized in the steady-state. These machines are called bottleneck machines, since they limit the throughput of the system. The system will work at the maximal rate and the productivity is optimum.

A job-shop system is a specific type of production system composed of a certain set of machines and a variety of jobs which must be produced using the machines. The manufacturing process of each job is specified as a sequence of machines to visit, i.e., as a routing into the system. Any routing is allowable but is defined uniquely for each job. Further, the time spent by the jobs on the machines is assumed to be fixed and deterministic. Finally, the sequencing of the jobs on the machines is also assumed to be given, as well as the order with which jobs are loaded into the system (input sequencing).

The model allows for evaluating such performance measure as the production rate in steady state. It is possible to fully utilize some machine in steady state with a finite number of jobs in-process. It means that the maximum production rate is obtained when those machines (called bottlenecks) are fully utilized, a condition that always can be satisfied.

In a timed marked graph, each transition takes a real time to fire. When a transition is enabled, a firing is initiated by moving one token from each of the transition input place.

The token remains in the transition during the time of the firing execution and then the firing terminates by adding one token in each of the transition output place.

Marked graphs (also called decision-free Petri nets) are Petri nets such that each place has exactly one input and one output transition as shown in Figure 3.1. In marked graph, any two transitions do not share the same input place. So, there is no conflict to simultaneously process several enabled transitions [Hillion 1989].

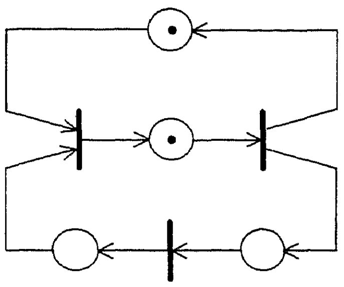


Figure 3.1 A example of marked graph

A marked graph has some properties as follows:

1) The token count in a direct circuit is invariant under any firing.
2) It is live iff there is at least one token on each direct circuit.
3) A live marked graph is safe if each place belongs to a directed circuit whose token count is one.
4) There is a live and safe marking in any marked graph if it is strongly connected.
5) The maximal number of tokens that a place can have is equal to the minimal number of tokens placed by initial marking on directed circuits containing the place.

For timed marked graphs, there exists already the formula to find the system cycle time [Freedman 1991], [Murata 1989], [Ramchandani 1974]. For a marked graph which has time delays in its transition and place, the system cycle time C is given by

$C = \text{Max}\{ T_i / N_i : i = 1,2,\ldots,n \}$ where

$T_i =$ Sum of the transition and place delays in circuit γ_i,

$N_i =$ Total number of tokens in the places in circuit γ_i, and

$n =$ Number of circuits in the marked graph.

Those circuits for which the cycle time is maximum are called bottleneck (critical) circuits. Indeed, those circuits are the ones that actually bound the throughput of the system. Two types of circuits in a marked graph are : 1) processing circuits — that model the manufacturing process of the sequence of each job. 2) command circuits — that model the sequencing of the jobs on the machines. If a circuit includes nodes of both processing and command circuit, then such circuit will be called mixed circuit [Hillion 1989].

3.2 Marked Graphs Modeling of FMS

Let us consider a job-shop system with 3 machines, donated by M1, M2, M3 and 4 different types of jobs, denoted by Job 1, Job 2, Job 3, Job 4. Table 3.1 shows the job requirements of this example [Baker 1974]. In this thesis, the batch size equals one for each job. We can consider it in future research when the batch size is greater than one.

Table 3.1 Job requirements of Example 2

Order	1st	2nd	3rd
Job 1	Machine 1 (4U)	Machine 2 (3U)	Machine 3 (2U)
Job 2	Machine 2 (1U)	Machine 1 (4U)	Machine 3 (4U)
Job 3	Machine 3 (3U)	Machine 2 (2U)	Machine 1 (3U)
Job 4	Machine 2 (3U)	Machine 3 (3U)	Machine 1 (1U)

As said in Section 2, we can obtain the Petri net model of Example 2 (Table 3.1) as Figure 3.2 shows. For the description of places and transitions of Figure 3.2, please refer to Table 3.2. Places p_{25}, p_{26}, and p_{27} represent availability of Machine 1, Machine 2, and Machine 3, respectively.

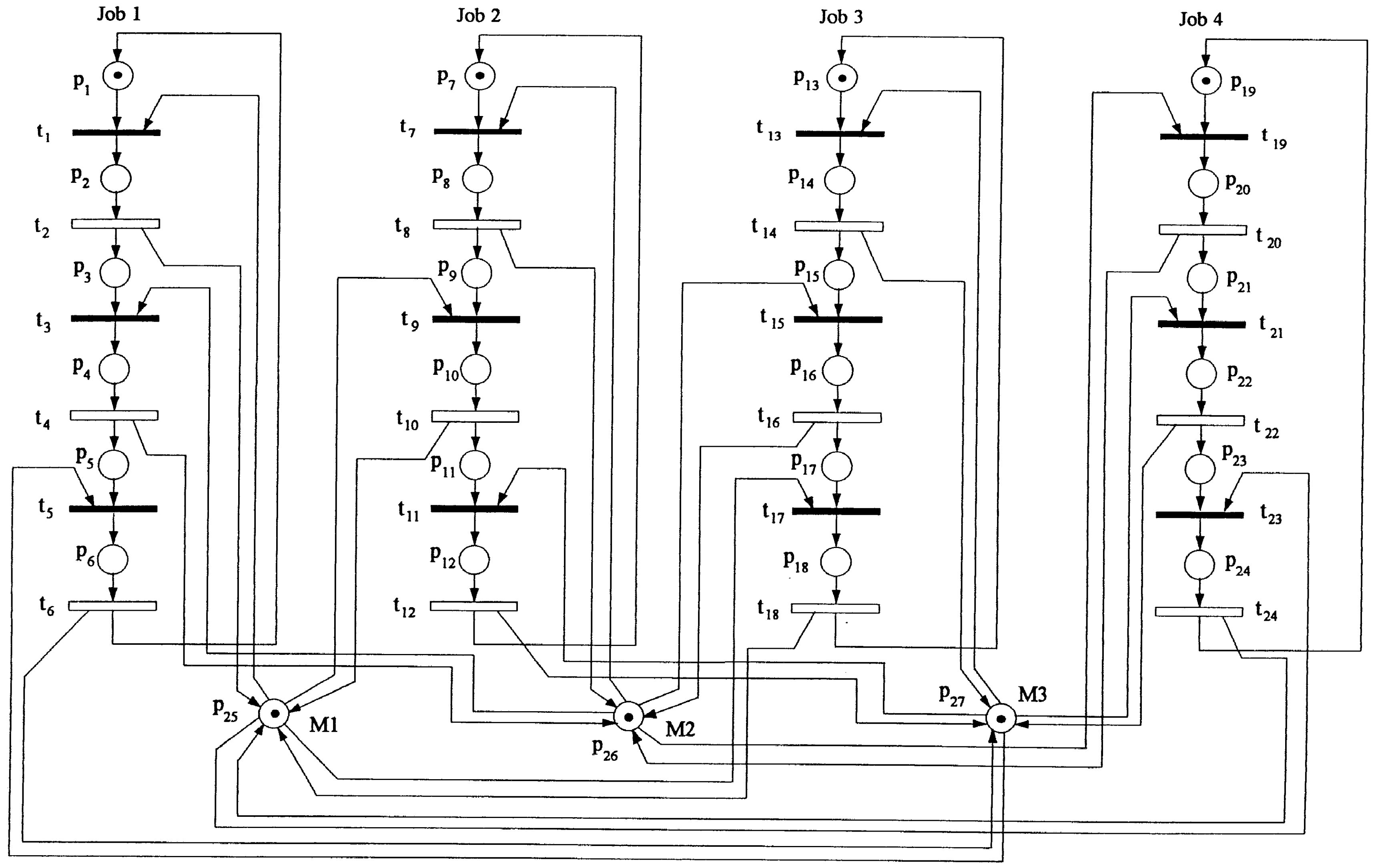


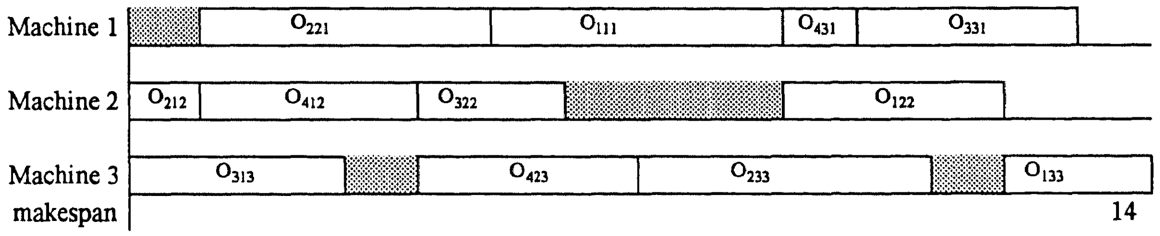
Figure 3.2 PN Model of the Example 2

Table 3.2 Description of places and transitions of Figure 3.2

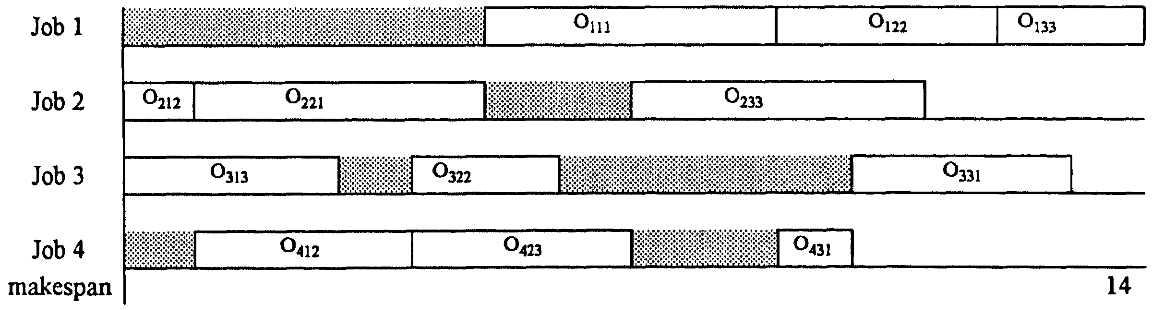
p_1	Job 1 material	t_1	The first operation of Job 1 can be processed
p_2	Job 1 material waiting the first operation	t_2	The first operation of Job 1 is processing
p_3	Job 1 material finished the first operation	t_3	The second operation of Job 1 can be processed
p_4	Job 1 material waiting the second operation	t_4	The second operation of Job 1 is processing
p_5	Job 1 material finished the second operation	t_5	The third operation of Job 1 can be processed
p_6	Job 1 material waiting the third operation	t_6	The third operation of Job 1 is processing
p_7	Job 2 material	t_7	The first operation of Job 2 can be processed
p_8	Job 2 material waiting the first operation	t_8	The first operation of Job 2 is processing
p_9	Job 2 material finished the first operation	t_9	The second operation of Job 2 can be processed
p_{10}	Job 2 material waiting the second operation	t_{10}	The second operation of Job 2 is processing
p_{11}	Job 2 material finished the second operation	t_{11}	The third operation of Job 2 can be processed
p_{12}	Job 2 material waiting the third operation	t_{12}	The third operation of Job 2 is processing
p_{13}	Job 3 material	t_{13}	The first operation of Job 3 can be processed
p_{14}	Job 3 material waiting the first operation	t_{14}	The first operation of Job 3 is processing
p_{15}	Job 3 material finished the first operation	t_{15}	The second operation of Job 3 can be processed
p_{16}	Job 3 material waiting the second operation	t_{16}	The second operation of Job 3 is processing
p_{17}	Job 3 material finished the second operation	t_{17}	The third operation of Job 3 can be processed
p_{18}	Job 3 material waiting the third operation	t_{18}	The third operation of Job 3 is processing
p_{19}	Job 4 material	t_{19}	The first operation of Job 4 can be processed
p_{20}	Job 4 material waiting the first operation	t_{20}	The first operation of Job 4 is processing
p_{21}	Job 4 material finished the first operation	t_{21}	The second operation of Job 4 can be processed
p_{22}	Job 4 material waiting the second operation	t_{22}	The second operation of Job 4 is processing
p_{23}	Job 4 material finished the second operation	t_{23}	The third operation of Job 4 can be processed
p_{24}	Job 4 material waiting the third operation	t_{24}	The third operation of Job 4 is processing
p_{25}	Machine 1 is available		
p_{26}	Machine 2 is available		
p_{27}	Machine 3 is available		

Since each place has exactly one input and output transition in a marked graph, we can easily find that the Petri net model of Figure 3.2 is not a marked graph because Places p_{25}, p_{26}, and p_{27} have more than one input and one output transition.

Since the scheduling problem exists in this example, we may use the branch-and-bound algorithm to obtain the optimal schedule. Figure 3.3 shows the optimal result for Example 2. In Figure 3.3(a), the manufacturing process of each job is supposed to be unique. That is to say, the executed sequences on each machine are fixed. Machine 1 processes Job 2 first, then Job 1 second, then Job 4 third, and Job 3 last. Machine 2 processes Job 2 first, then Job 4 second, then Job 3 third, and Job 1 last. Machine 3 processes Job 3 first, then Job 4 second, then Job 2 third, and Job 1 last.



(a) The executed sequences on each machine



(b) The job-by-job description

Figure 3.3 Two views of a optimal schedule of Example 2

For job shop systems, we can directly derive the marked graph from the initial Petri net model after the schedule is determined. A general procedure can be stated as follows:

(1) If a machine serves for i jobs, then change the number of places from 1 to i. Name these places as $p^1{}_k$, $p^2{}_k$, ..., $p^i{}_k$ where k is the initial symbol for that machine.

(2) According to the executed sequences on each machine, we draw an output arc from $p^1{}_k$ to the transition which represents the first job processed by the machine, then draw an input arc to $p^2{}_k$ from the transition which represents the first job finished by that machine. Based on the same method as above, we draw an output arc from $p^2{}_k$ to the transition which represents the second job processed by the machine, then draw an input arc to $p^3{}_k$ from the transition which represents the second job finished by that machine. Repeat the above procedure until we draw an output arc from $p^i{}_k$ to the last job processed by that machine. Then we draw a last arc from the last job finished by that machine to $p^1{}_k$.

(3) Put a token in $p^1{}_k$.

We can apply this general procedure to Example 2. Since Machine 1 processes 4 jobs, we draw 4 places ($p^1{}_{25}$, $p^2{}_{25}$, $p^3{}_{25}$, and $p^4{}_{25}$ where "25" represents Machine 1). Since it processes Job 2 first, we draw an output arc from $p^1{}_{25}$ to t_9 and draw an input arc to $p^2{}_{25}$ from t_{10}. Since Machine 1 processes Job 1 second, we draw an output arc from $p^2{}_{25}$ to t_1 and draw an input arc to $p^3{}_{25}$ from t_2. Because it processes Job 4 third, we draw an output arc from $p^3{}_{25}$ to t_{23} and draw an input arc to $p^4{}_{25}$ from t_{24}. Since it processes Job 3 last, we draw an output arc from $p^4{}_{25}$ to t_{17} and draw an input arc to $p^1{}_{25}$ from t_{18}. We do the same procedures for Machines 2 and 3. Finally, we put three tokens in places $p^1{}_{25}$, $p^1{}_{26}$, and $p^1{}_{27}$. According to the above procedure, we can obtain the marked graph of the FMS using the optimal scheduling of Example 2, as shown in Figure 3.4.

In Figure 3.4, each place has exactly one input and one output transition. Therefore, the net is a marked graph.

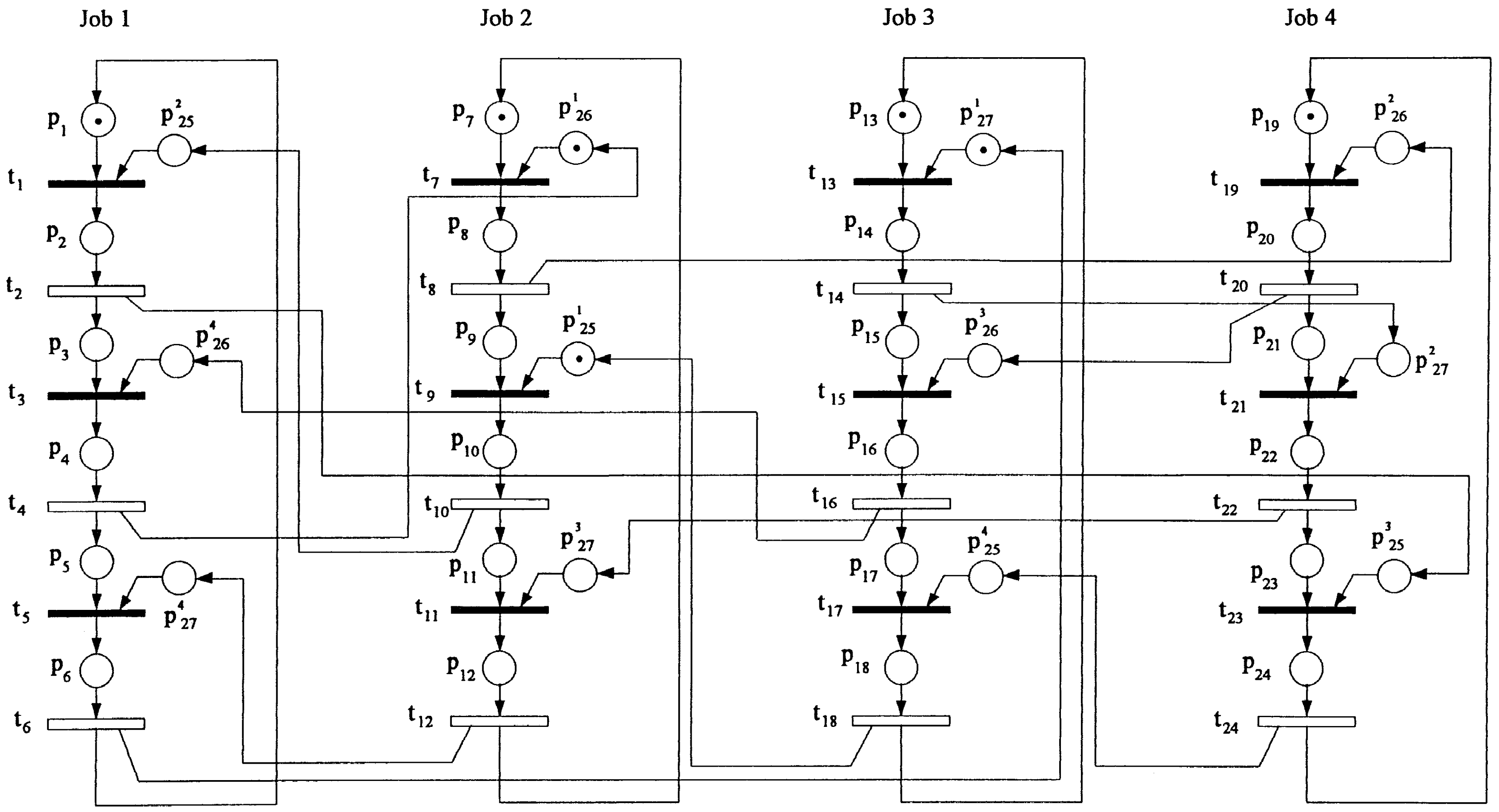


Figure 3.4 A marked graph of the optimal scheduling of Example 2

3.3 Performance Evaluation of Cyclic Systems

It should be clear that the Petri net model developed in Section 3.2 corresponds to a marked graph (see Figure 3.4). Furthermore, the net is strongly connected, thanks to the input command circuit that links the processing circuits all together. The firing times of transitions (as specified by the operation processing times — Table 3.3) make it finally a strongly connected timed Petri net.

Table 3.3 Operation times of Figure 3.4

Operation	time	Operation	time	Operation	time
t_1	0	t_9	0	t_{17}	0
t_2	4	t_{10}	4	t_{18}	1
t_3	0	t_{11}	0	t_{19}	0
t_4	3	t_{12}	4	t_{20}	3
t_5	0	t_{13}	0	t_{21}	0
t_6	2	t_{14}	3	t_{22}	3
t_7	0	t_{15}	0	t_{23}	0
t_8	1	t_{16}	2	t_{24}	1

Appendix B shows the elementary circuits and their cycle times of Figure 3.4. There are totally 49 elementary circuits (including the four processing circuits— γ_1, γ_2, γ_3, γ_4, and the three command circuits — γ_{14}, γ_{32}, γ_{47}). The $\max\{T_i/N_i\}$ is equal to 12 Units. Therefore, the maximum cycle time is 12 Units and the critical circuits are γ_6, γ_{32}, γ_{41}, and γ_{47}.

According to Figure 3.3, we can find that makespan of optimal scheduling of Example 2 is 14 Units. It means that it needs 14 Units when each job (Job 1, Job 2, Job 3, and Job 4) is executed exactly once.

Based on Figure 3.4, the cycle time of the system is 12 Units. It is reasonable, because it is a long-term run performance. In Figure 3.3(a), Machine 3 finished Job 1 (i.e., the first part of each job) at the end of 14-th Unit. But Machine 2 is idle in 12-th Unit, and Machine 2 can processes the first operation of the second part of the Job 2 (i.e., the second part of each job). Therefore, the cycle time (12 Units) < the makespan (14 Units) is reasonable. The cycle time is correct. If the cycle time is greater than the makespan, then the cycle time is incorrect.

CHAPTER FOUR

AUTOMATED GUIDED VEHICLE SYSTEMS (AGVS)

4.1 Two AGVS Models

Automated Guided Vehicle Systems (AGVS) are increasingly used for material handling in factories and warehouses. An AGVS can replace conveyors and can accommodate new routes with relative ease. The AGVS are flexible systems for transportation and suitable for simple transport operations with a small number of destinations as well as complex ones with many destinations [Muller 1983].

The advantages to use AGVS are as follows:

(1) Saving in costs and personnel.

(2) High performance capacity even in the area of high material flow densities.

(3) High transport safety.

(4) Automated transport, closing the gap between automated storage systems and manufacturing systems.

In this thesis, we want to discuss and compare two AGVS models: the shared AGVS and the duty AGVS through Petri net models [Lee 1993].

4.1.1 The Shared AGVS

In a shared AGVS, there are no firm objects served by the AGVS. Figure 4.1 shows an FMS with the shared AGVS. S_L and S_U represent the loading area and unloading area. $S1_i$ and $S1_o$ represent the input position and output position of Machine 1. $S2_i$, $S2_o$, $S3_i$, and $S3_o$ have the similar meanings as $S1_i$ and $S1_o$. r_1 represents the route from loading area to Machine 1. r_2 represents the route from loading area to Machine 2. $r_3 \sim r_{10}$ have the similar meanings. All AGVs can serve in the loading area, unloading area or any machine. Any AGV is devoted to a job until it is completed. The AGVs have an original position at the loading area. Raw material is loaded onto an empty AGV at the loading area and processed by a machine through the FMS to the unloading area. After the AGV delivers a part to the input position of a machine and unload a part to it, the AGV moves from input position to output position immediately. After the part is unloaded, the empty AGV returns to the original position at the loading area and waits for additional raw material.

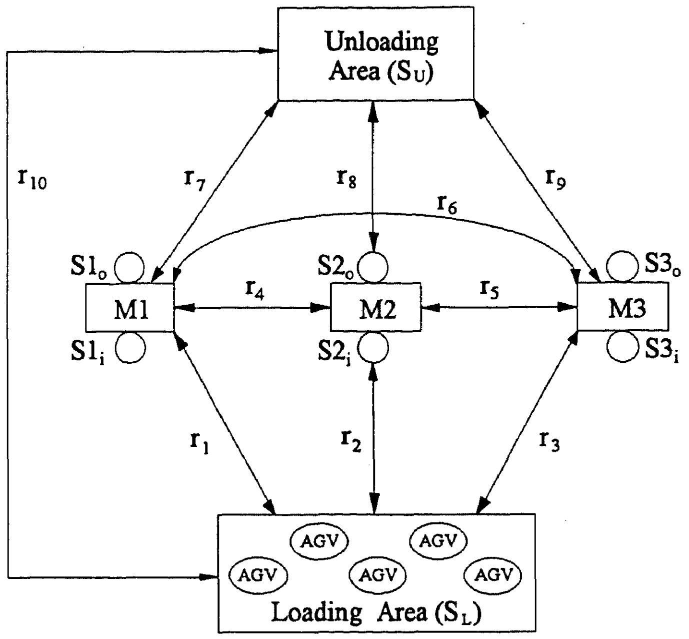


Figure 4.1 An FMS with the "shared" AGVS

4.1.2 The Duty AGVS

In a duty AGVS, there is a firm object served by the AGV. The loading area has some AGVs (the number is dependent on every different case) which are used to send parts to machines. Each machine Mj has its own AGV_{mj} which has an original position at the output of the machine ($S1_o$, $S2_o$, or $S3_o$ ...) and is used only to send a part away to the input of another machine ($S1_i$, $S2_i$, or $S3_i$ ...). After delivering a part to another machine, the AGV returns to its original position at the owner machine for next delivery. Figure 4.2 shows an FMS with the duty AGVS. There are no AGVs moving between loading area and unloading area, so we don't need to draw route "r_{10}".

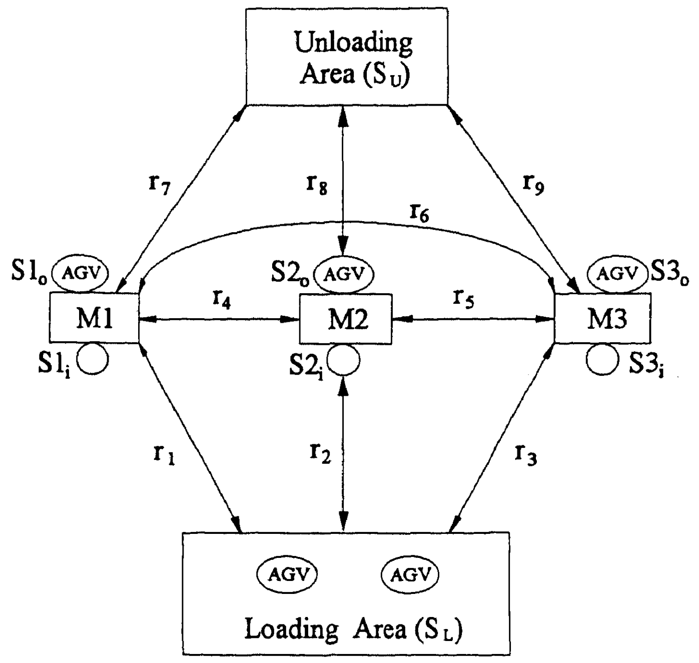


Figure 4.2 An FMS with the "duty" AGVS

4.2 Petri Net Models of AGVS-Based FMS

We have already discussed two AGVS models in Section 4.1. Furthermore, we want to discuss the suitable cases for these two AGVS models.

Table 4.1 shows the job requirements of Example 3. In Example 3, we have two jobs and three machines, and each job has the same processed sequence through machines.

Table 4.1 Job requirements of Example 3

Order	1 st	2 nd	3 rd
Job 1	Machine 1	Machine 2	Machine 3
Job 2	Machine 1	Machine 2	Machine 3

Table 4.2 shows the job requirements of Example 4. We have two jobs and three machines, and Job 2 has the reverse processed sequence through machines. Two AGVS will be applied to Examples 3 and 4 respectively.

Table 4.2 Job requirements of Example 4

Order	1 st	2 nd	3 rd
Job 1	Machine 1	Machine 2	Machine 3
Job 2	Machine 3	Machine 2	Machine 1

1) The shared AGVS: The FMS of Examples 3 and 4 along with 5 shared AGVs and their paths are depicted in Figures 4.3 and 4.5. p_{17} represents that AGVs are available at the loading area. Because all AGVs have an original position at the loading area, we put five tokens in p_{17}.

2) The duty AGVS: The FMS of Examples 3 and 4 along with 5 duty AGVs and their paths are depicted in Figures 4.4 and 4.6. p_{17} represents that AGVs are available on the loading area. These places (p_{22}, p_{27}, and p_{32}) represent that AGVs are available on the output of machines (Machines 1, 2, and 3). According to the definition of the duty AGVS, each machine M_j has its own AGV_{mj} which has an original position at the output of the machine. Thus, we put two tokens in p_{17} and one token in p_{22}, p_{27}, and p_{32}, respectively.

Now we discuss about the initial marking. p_{20}, p_{25}, and p_{30} represent Machines 1, 2, and 3, so we put one token in p_{20}, p_{25}, and p_{30}. p_{18}, p_{23}, p_{28}, p_{33}, p_{35}, p_{58}, and p_{59} represent paths, so we put one token in these places. p_{19}, p_{21}, p_{24}, p_{26}, p_{29}, p_{31}, and p_{34} represent the input or output positions of machines and unloading area. Because these are empty originally, we put one token in these places.

Tables 4.3(a) and 4.3(b) show the description of places and transitions of Figures 4.3, 4.4, 4.5, and 4.6. We assume that the time is negligible for the following operations: (1) a material is loaded to AGV, (2) a material is unloaded from AGV, and (3) AGV moves from input position (Sj_i) to output position (Sj_o) of the same machine. Therefore, we draw these transitions as immediate transitions, i.e. solid transitions.

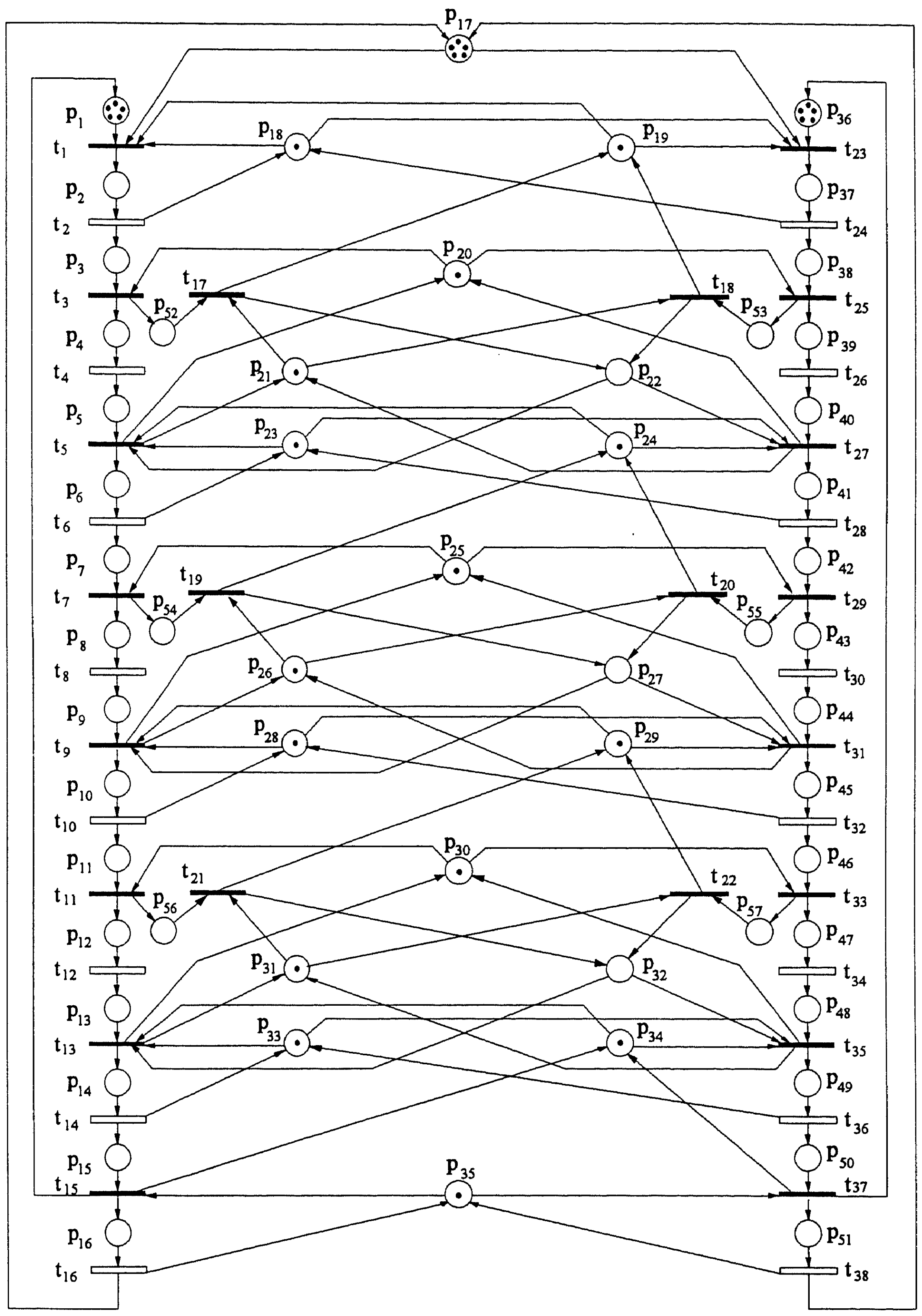


Figure 4.3 The Petri net model for Example 3 with the shared AGVS

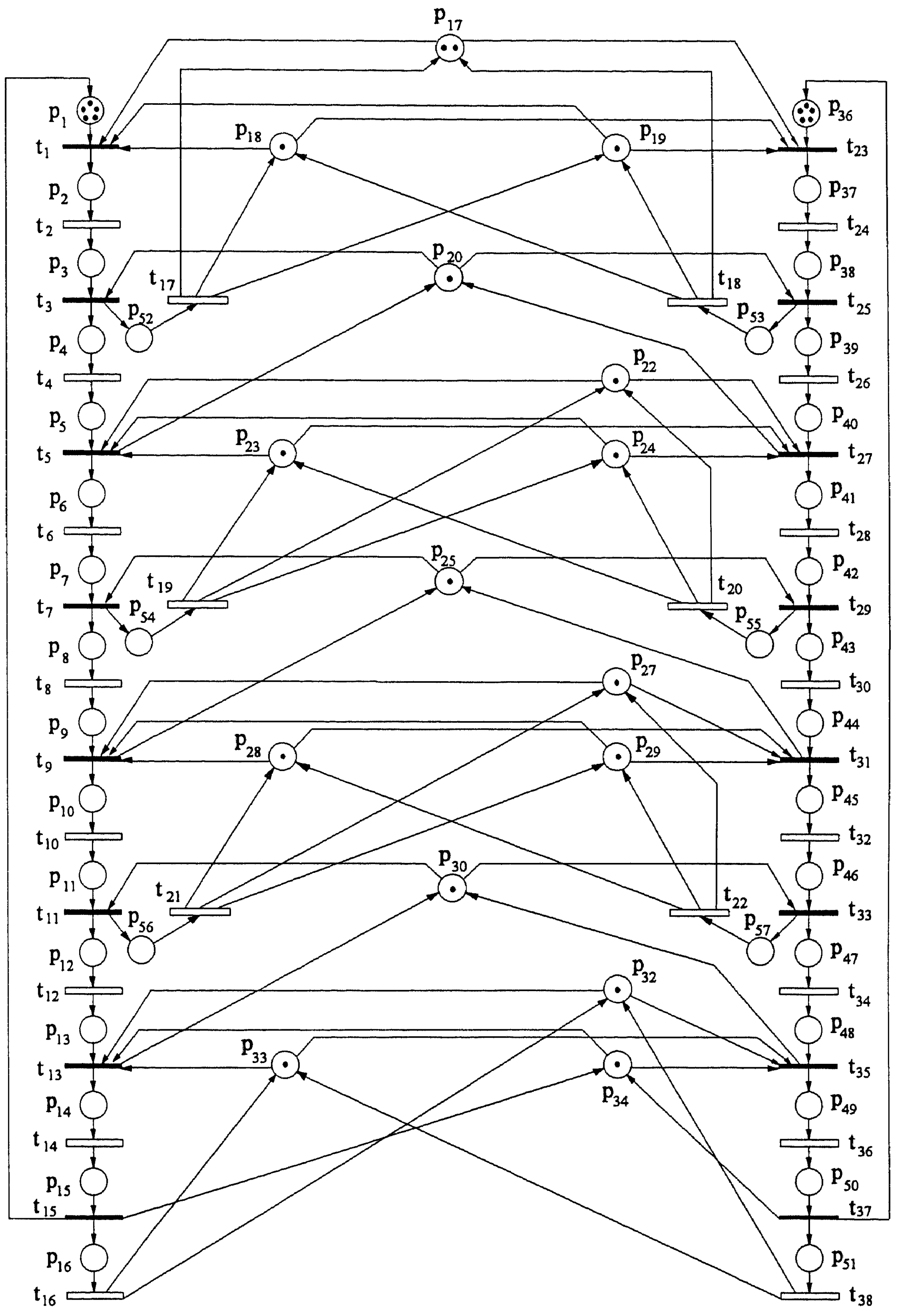


Figure 4.4 The Petri net model for Example 3 with the duty AGVS

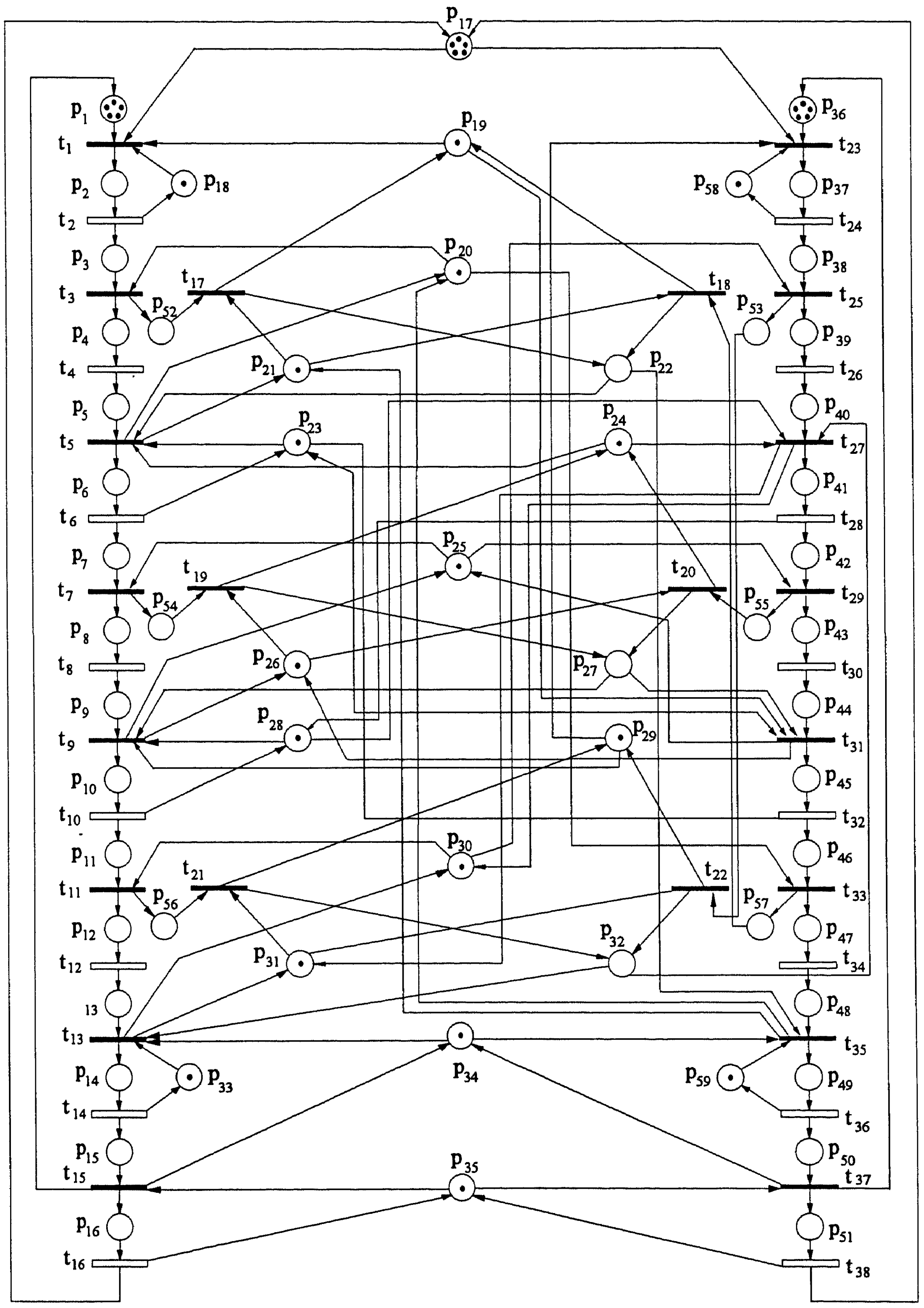


Figure 4.5 The Petri net model for Example 4 with the shared AGVS

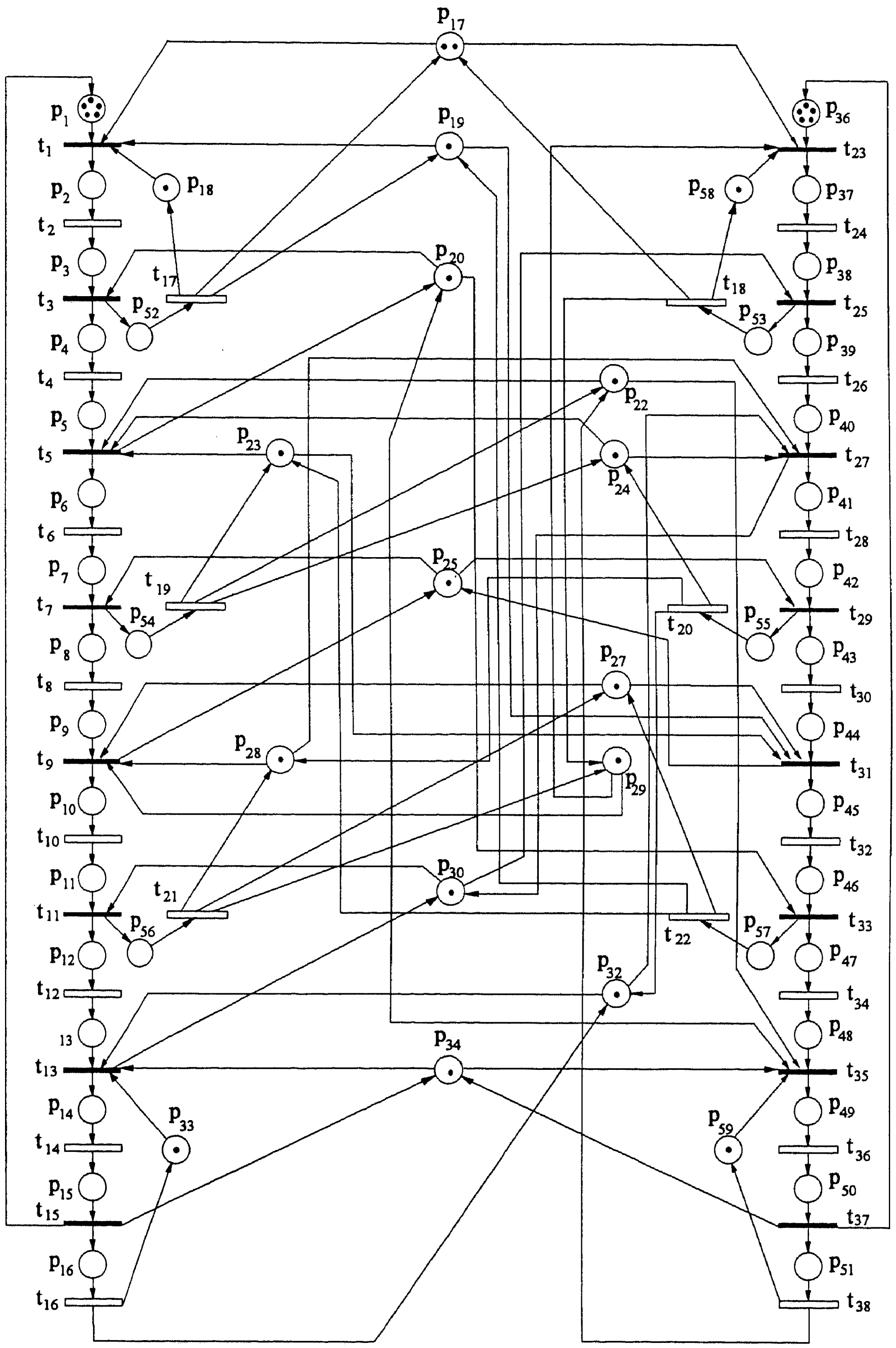


Figure 4.6 The Petri net model for Example 4 with the duty AGVS

Table 4.3(a) Description of places of Figures 4.3, 4.4, 4.5, and 4.6

p_1	Job 1 material is available	p_{34}	S_U is available
p_2	Job 1 material loaded to AGV	p_{35}	Path r_{10} is available
p_3	Material 1 on $S1_i$	p_{36}	Job 2 material is available
p_4	Material 1 waiting first operation	p_{37}	Job 2 material loaded to AGV
p_5	Material 1 finished first operation	p_{38}	Material 2 on $S1_i$ (for Fig. 4-3, 4-4)
p_6	Material 1 on $S1_o$	p_{38}	Material 2 on $S3_i$ (for Fig. 4-5, 4-6)
p_7	Material 1 on $S2_i$	p_{39}	Material 2 waiting first operation
p_8	Material 1 waiting second operation	p_{40}	Material 2 finished first operation
p_9	Material 1 finished second operation	p_{41}	Material 2 on $S1_o$ (for Fig. 4-3, 4-4)
p_{10}	Material 1 on $S2_o$	p_{41}	Material 2 on $S3_o$ (for Fig. 4-5, 4-6)
p_{11}	Material 1 on $S3_i$	p_{42}	Material 2 on $S2_i$
p_{12}	Material 1 waiting third operation	p_{43}	Material 2 waiting second operation
p_{13}	Material 1 finished third operation	p_{44}	Material 2 finished second operation
p_{14}	Material 1 on $S3_o$	p_{45}	Material 2 on $S2_o$
p_{15}	Material 1 on S_U	p_{46}	Material 2 on $S3_i$ (for Fig. 4-3, 4-4)
p_{16}	AGV waiting to return to S_L	p_{46}	Material 2 on $S1_i$ (for Fig. 4-5, 4-6)
p_{17}	AGVs on S_L are available	p_{47}	Material 2 waiting third operation
p_{18}	Path r_1 is available	p_{48}	Material 2 finished third operation
p_{19}	$S1_i$ is available	p_{49}	Material 2 on $S3_o$ (for Fig. 4-3, 4-4)
p_{20}	Machine 1 is available	p_{49}	Material 2 on $S3_o$ (for Fig. 4-5, 4-6)
p_{21}	$S1_o$ is available	p_{50}	Material 2 on S_U
p_{22}	AGV on $S1_o$	p_{51}	AGV waiting to return to S_L
p_{23}	Path r_4 is available	p_{52}	AGV on $S1_i$
p_{24}	$S2_i$ is available	p_{53}	AGV on $S1_i$ (for Fig. 4-3, 4-4)
p_{25}	Machine 2 is available	p_{53}	AGV on $S1_i$ (for Fig. 4-5, 4-6)
p_{26}	$S2_o$ is available	p_{54}	AGV on $S2_i$
p_{27}	AGV on $S2_o$	p_{55}	AGV on $S2_i$
p_{28}	Path r_5 is available	p_{56}	AGV on $S3_i$
p_{29}	$S3_i$ is available	p_{57}	AGV on $S3_i$ (for Fig. 4-3, 4-4)
p_{30}	Machine 3 is available	p_{57}	AGV on $S3_i$ (for Fig. 4-5, 4-6)
p_{31}	$S3_o$ is available	p_{58}	Path r_3 is available (for Fig. 4-5, 4-6)
p_{32}	AGV on $S3_o$	p_{59}	Path r_7 is available (for Fig. 4-5, 4-6)
p_{33}	Path r_9 is available		

Table 4.3(b) Description of transitions of Figures 4.3, 4.4, 4.5, and 4.6

t_1	Material 1 loaded to AGV	t_{22}	AGV moving from $S3_i$ to $S3_o$ (for 4-3)
t_2	Material 1 moved from S_L to $S1_i$	t_{22}	AGV moving from $S3_i$ to $S2_o$ (for 4-4)
t_3	Material 1 moved from $S1_i$ to Machine 1	t_{22}	AGV moving from $S1_i$ to $S1_o$ (for 4-5)
t_4	Material 1 processed first operation	t_{22}	AGV moving from $S1_i$ to $S2_o$ (for 4-6)
t_5	Material 1 moved from Machine 1 to $S1_o$	t_{23}	Material 2 loaded to AGV
t_6	Material 1 moved from $S1_o$ to $S2_i$	t_{24}	Material 2 moved from S_L to $S1_i$(for 4-3,4-4)
t_7	Material 1 moved from $S2_i$ to Machine 2	t_{24}	Material 2 moved from S_L to $S3_i$(for 4-5,4-6)
t_8	Material 1 processed second operation	t_{25}	Material 2 moved from $S1_i$ to Machine 1(4-3,4-4)
t_9	Material 1 moved from Machine 2 to $S2_o$	t_{25}	Material 2 moved from $S3_i$ to Machine 3(4-5,4-6)
t_{10}	Material 1 moved from $S2_o$ to $S3_i$	t_{26}	Material 2 processed first operation
t_{11}	Material 1 moved from $S3_i$ to Machine 3	t_{27}	Material 2 moved from Machine 1 to $S1_o$(4-3,4-4)
t_{12}	Material 1 processed third operation	t_{27}	Material 2 moved from Machine 3 to $S3_o$(4-5,4-6)
t_{13}	Material 1 moved from Machine 3 to $S3_o$	t_{28}	Material 2 moved from $S1_o$ to $S2_i$ (4-3,4-4)
t_{14}	Material 1 moved from $S3_o$ to S_U	t_{28}	Material 2 moved from $S3_o$ to $S2_i$ (4-5,4-6)
t_{15}	Material 1 moved from S_U to unloading area	t_{29}	Material 2 moved from $S2_i$ to Machine 2
t_{16}	AGV moving from S_U to S_L (for 4-3, 4-5)	t_{30}	Material 2 processed second operation
t_{16}	AGV moving from S_U to $S3_o$ (for 4-4, 4-6)	t_{31}	Material 2 moved from Machine 2 to $S2_o$
t_{17}	AGV moving from $S1_i$ to $S1_o$ (for 4-3,4-5)	t_{32}	Material 2 moved from $S2_o$ to $S3_i$ (4-3,4-4)
t_{17}	AGV moving from $S1_i$ to S_L (for 4-4, 4-6)	t_{32}	Material 2 moved from $S2_o$ to $S1_i$ (4-5,4-6)
t_{18}	AGV moving from $S1_i$ to $S1_o$ (for 4-3)	t_{33}	Material 2 moved from $S3_i$ to Machine 3(4-3,4-4)
t_{18}	AGV moving from $S1_i$ to S_L (for 4-4)	t_{33}	Material 2 moved from $S1_i$ to Machine 1(4-5,4-6)
t_{18}	AGV moving from $S3_i$ to $S3_o$ (for 4-5)	t_{34}	Material 2 processed third operation
t_{18}	AGV moving from $S3_i$ to S_L (for 4-6)	t_{35}	Material 2 moved from Machine 3 to $S3_o$(4-3,4-4)
t_{19}	AGV moving from $S2_i$ to $S2_o$ (for 4-3,4-5)	t_{35}	Material 2 moved from Machine 1 to $S1_o$(4-5,4-6)
t_{19}	AGV moving from $S2_i$ to S_L (for 4-4, 4-6)	t_{36}	Material 2 moved from $S3_o$ to S_U (4-3,4-4)
t_{20}	AGV moving from $S2_i$ to $S2_o$ (for 4-3,4-5)	t_{36}	Material 2 moved from $S1_o$ to S_U (4-5,4-6)
t_{20}	AGV moving from $S2_i$ to $S1_o$ (for 4-4)	t_{37}	Material 2 moved from S_U to unloading area
t_{20}	AGV moving from $S2_i$ to $S3_o$ (for 4-6)	t_{38}	AGV moving from S_U to S_L (for Fig. 4-3,4-5)
t_{21}	AGV moving from $S3_i$ to $S3_o$ (for 4-3,4-5)	t_{38}	AGV moving from S_U to $S1_o$ (for Fig. 4-6)
t_{21}	AGV moving from $S3_i$ to S_L (for 4-4, 4-6)	t_{38}	AGV moving from S_U to $S3_o$ (for Fig. 4-4)

4.3 Stochastic Petri Net Package (SPNP)

Developed in the early 80's, Stochastic Petri Net Package (SPNP) has become one of the most commonly used Petri net tools. The package can deal with Petri net text file inputs which define places, transitions, their input and output relations, and transitions' random firing delay times with exponential distributions. In SPNP, Petri nets with immediate transitions and priorities of enabled transitions are allowed.

Given a stochastic Petri net described as a Petri net text file, the package first generates its reachability graph. Then the Markov chain solution is derived. The performance indices such as the steady-state transition firing frequency and probability of a place with token presence can be computed. Based on these, one can calculate system throughput and transient solution for stochastic Petri net.

SPNP runs under the UNIX system on a wide array of platforms such as 5.AX, SUN, and Convex, and under the VMS system such as VAX [Ciardo 90].

4.4 Performance Analysis

In this section we will use the SPNP software to evaluate four cases (Example 3 with the shared AGVS, Example 3 with the duty AGVS, Example 4 with the shared AGVS, and Example 4 with the duty AGVS). We assume that all the time delays are associated with transitions instead of places in order that we can directly use the SPNP package. The firing rates assumed for the transitions in Figures 4.3-6 are summarized in Table 4.4. When an immediate transition is enabled, it can fire instantaneously. Suppose that v denotes the AGV speed rate, a variable. Appendices C-F show the codes of Figures 4.3-6 for SPNP.

Table 4.5(a) shows the system throughputs of Example 3 (the same sequence) in Figures 4.3 and 4.4. We can see that the duty AGVS is better than the shared AGVS. The throughput difference is approximately 11% when the AGV speed rate equals to 0.75. Table 4.5(b) shows the system throughputs of Example 4 (the reverse sequence) in the

Figures 4.5 and 4.6. We can see that the shared AGVS is better than the duty AGVS. The throughput difference is approximately 10% when the AGV speed rate equals to 0.75.

Table 4.4 Firing rates of transitions of Figures 4.3, 4.4, 4.5, and 4.6

Transition	Firing Rate	Transition	Firing Rate	Transition	Firing Rate
t_2	4v	t_{24}	4v	t_{17}(for Fig 4-3,5)	∞
t_4	1.0	t_{26}	4.0(4-3,4-4)	t_{17}(for Fig 4-4,6)	4v
t_6	4v	t_{26}	1.0(4-5,4-6)	t_{18}(for Fig 4-3,5)	∞
t_8	4.0	t_{28}	4v	t_{18}(for Fig 4-4,6)	4v
t_{10}	4v	t_{30}	1.0	t_{19}(for Fig 4-3,5)	∞
t_{12}	4.0	t_{32}	4v	t_{19}(for Fig 4-4,6)	4v
t_{14}	4v	t_{34}	1.0(4-3,4-4)	t_{20}(for Fig 4-3,5)	∞
t_{16}	v (4-3,4-5)	t_{34}	4.0(4-5,4-6)	t_{20}(for Fig 4-4,6)	4v
t_{16}	4v (4-4,4-6)	t_{36}	4v	t_{21}(for Fig 4-3,5)	∞
		t_{38}	v (4-3,4-5)	t_{21}(for Fig 4-4,6)	4v
		t_{38}	4v (4-4,4-6)	t_{22}(for Fig 4-3,5)	∞
				t_{22}(for Fig 4-4,6)	4v
The firing rates of other transitions are infinite (immediate transitions).					

Table 4.5(a) The system throughputs of Figures 4.3 and 4.4 for the same sequence

AGV Speed Rate	0	0.25	0.5	0.75	1.0
Duty AGVS	0	0.254	0.402	0.488	0.542
Shared AGVS	0	0.198	0.343	0.440	0.506

Table 4.5(b) The system throughputs of Figures 4.5 and 4.6 for the reverse sequence

AGV Speed Rate	0	0.25	0.5	0.75	1.0
Duty AGVS	0	0.135	0.284	0.396	0.468
Shared AGVS	0	0.194	0.338	0.436	0.503

The results of performance analysis are shown in Figure 4.7 and 4.8 for the four Petri net models shown in Figures 4.3, 4.4, 4.5, and 4.6. They show the changes in system throughput and AGV speed rate when the AGV speed rate increases from 0 to 1.0. The system throughput decreases as the AGV speed rate decreases.

In Figure 4.7, we compare these two systems (1) An FMS whose every job has the same processed sequence through machines combined with the shared AGVS; and (2) An FMS whose every job has the same processed sequence through machines combined with the duty AGVS. We can see that the system throughput of using the duty AGVS is greater than the system throughput of using the shared AGVS. Thus, the duty AGVS is more suitable to Example 3.

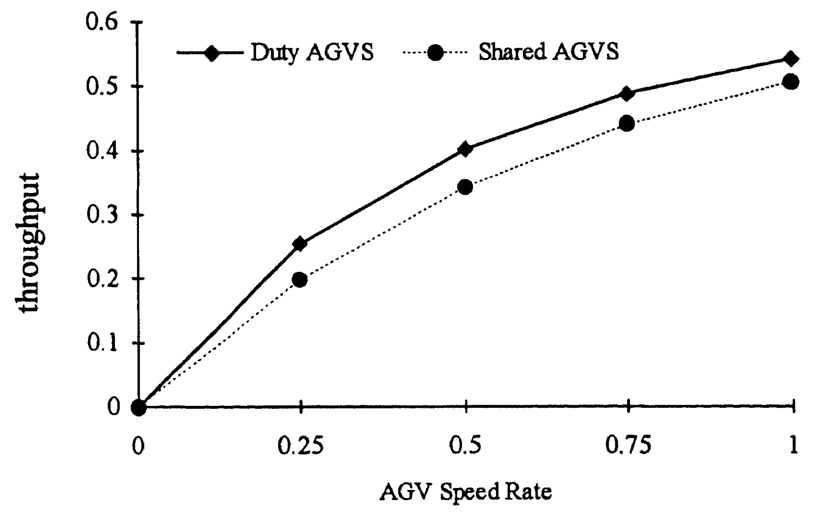


Figure 4.7 The system throughput vs. AGV speed rate for Example 3

In Figure 4.8, we compare these two systems (1) An FMS whose every job has the reverse processed sequence through machines combined with the shared AGVS; and (2) An FMS whose every job has the reverse processed sequence through machines combined with the duty AGVS. We can see that the system throughput of using the shared AGVS is greater than the system throughput of using the duty AGVS. Thus, the shared AGVS is more suitable to Example 4.

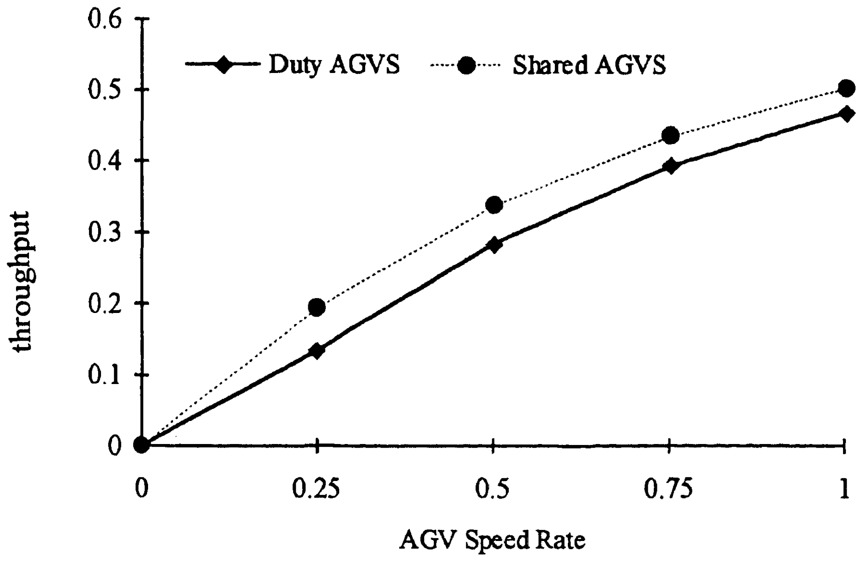


Figure 4.8 The system throughput vs. AGV speed rate for Example 4

Therefore, the result shows that:

If each job has the same processed sequence through machines, then it is better to use the duty AGVS. If each job has the reverse processed sequence through machines, then it is better to use the shared AGVS.

The intuitive reasons for the results are given below:

(1) If each job has the same processed sequence through machines, the system is simple and regular for AGVs' transportation because the parts always come from same position and will be delivered to same position for machines. Thus, we use the duty AGVS (every machine has its own AGV) to this case. The duty AGVS will be better for this case.

(2) If each job has the reverse processed sequence through machines, the system is complex and irregular for AGVs' transportation because the parts always come from different positions and will be delivered to different positions for machines. Thus, we use the shared AGVS (any AGV is devoted to a job (part) until it is completed) to this case. The shared AGVS will be better for this case.

CHAPTER FIVE

AGV JAM-FREE CONTROL

In an FMS, the transporting system may have more than one AGV (for example: the FMS has 5 AGVs in Figure 4.1) which transport materials among different workstations. Thus, the traffic jam problems must be considered and solved. The objective of the AGV jam-free control module is to guarantee jam-free condition among AGVs.

The basic idea of the AGV jam-free control is described as follows. When a missioned AGV found a station on its traveling route occupied by another AGV, the control center will send a command to that AGV to leave to an adjacent station. When that station is released, the missioned AGV can resume its traveling on the same route path.

In this thesis, we want to discuss two different cases:

1) The distance between two stations is short.

2) The distance between two stations is long.

If the time that AGV moves between two stations is less than one minute, we can classify it to short distance case. Otherwise, it belongs to long distance case. However, it depends on physical requirements.

5.1 Short Distance Case

Figure 5.1 shows the jam-free control unit module when the distance between stations is short. In Figure 5.1, AGV_x stops at Station i and AGV_y stops at Station j. AGV_x want to move from Station i to Station j. Because Station j is occupied by AGV_y, the control center sends a command to AGV_y to leave for Station k. Then AGV_x can start to move from Station i to Station j. Places p_i and p_j represent that AGVs stop at Station i and Station j. Place p_{ij} represents AGV got the order to move from Station i to Station j. Place

p_{je} represents Station j is empty since its marking is zero. Transition t_{ij} represents AGV moves from Station i to Station j.

AGV_X and AGV_Y stop at Station i and j since each of p_i and p_j has a token. AGV_X wants to move from Station i to Station j due to a token in p_{ij}. Because there are no tokens in p_{je} (Station j is not empty), AGV_X cannot start to move. Control center sends an order (p_{jk}) to move AGV_Y. AGV_Y then moves from Station j to Station k assuming Station k is available as shown in Figure 5.1. Thus, Station j becomes empty (p_{ej} has a token). Transition t_{ij} can be enabled, and AGV_X can start to move from Station i to Station j. The traffic problem is solved.

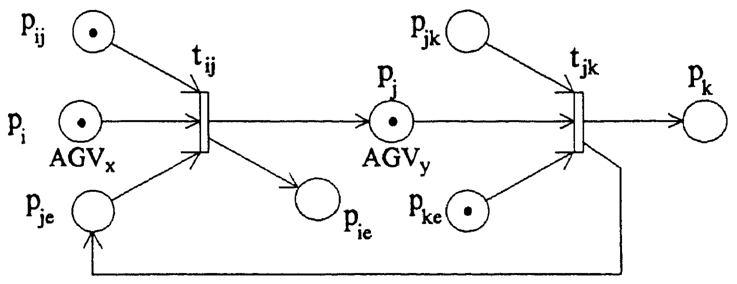


Figure 5.1 Jam-free control unit module for short distance case

Table 5.1 Description of places and transitions of Figure 5.1

p_i	AGV stays at station i	p_{ie}	Station i is empty
p_j	AGV stays at station j	p_{je}	Station j is empty
p_k	AGV stays at station k	p_{ke}	Station k is empty
p_{ij}	AGV can move from station i to j	t_{ij}	AGV moves from station i to j
p_{jk}	AGV can move from station j to k	t_{jk}	AGV moves from station j to k

5.2 Long Distance Case

In Section 5.1, AGV_X cannot move until AGV_Y leaves Station j. If the distance between stations is short, then the jam-free control unit module (Figure 5.1) can be accepted (because the waiting time of AGV_X is relatively short). If the distance between stations is long, then the jam-free control unit module (Figure 5.1) must be modified because otherwise the waiting time of AGV_X is too long.

We add a limit switch at route r_{jk} but it is near Station j as Figure 5.2 shows. When AGV_Y leaves Station j and touches the limit switch, AGV_X at Station i can start to move. This method can save much time, because AGV_X does not need to wait too long.

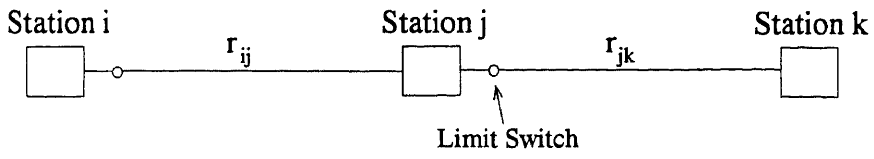


Figure 5.2 The path added limit switch

Figure 5.3 shows the jam-free control unit module when the distance between stations is long. Transition $t_{jk\text{-}l}$ represents AGV_Y moves from Station j to limit switch and touches the limit switch. Because it will spend very short time, we draw the transition $t_{jk\text{-}l}$ as an immediate transition. We can regard Station j as empty when AGV_Y touches the limit switch of r_{jk}. Thus, transition $t_{ij\text{-}l}$ is enabled when p_{je} has a token (AGV_Y touches the limit switch). It means that AGV_X can start to move from Station i to Station j. It will save the waiting time of AGV_X .

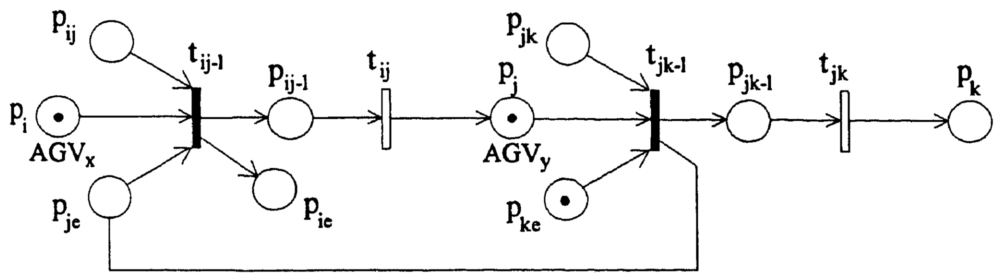


Figure 5.3 Jam-free control unit module for long distance case

Table 5.2 Description of places and transitions of Figure 5.3

p_i	AGV stays at station i	p_{ie}	Station i is empty
p_j	AGV stays at station j	p_{je}	Station j is empty
p_k	AGV stays at station k	p_{ke}	Station k is empty
p_{ij}	AGV can move from station i to j	$t_{ij\text{-}l}$	AGV moves from station i to limit switch of r_{ij}
p_{jk}	AGV can move from station j to k	$t_{jk\text{-}l}$	AGV moves from station j to limit switch of r_{jk}
$p_{ij\text{-}l}$	AGV touches the limit switch of r_{ij}	t_{ij}	AGV moves from station i to j
$p_{jk\text{-}l}$	AGV touches the limit switch of r_{jk}	t_{jk}	AGV moves from station j to k

5.3 Jam-Free Petri Net Models

Figure 5.4 shows an example of system layouts. r_{ij} represents the routes of AGVs. Arrows represent the moving directions of an AGV. An AGV can move from Station 1 to Station 2 or Station 3. An AGV can move from Station 2 to Station 3 or Station 4. If an AGV stays at Station 3, then it can only move to Station 4. If an AGV stays at Station 4, then it can only move to Station 1. In short distance case, we can ignore the limit switches.

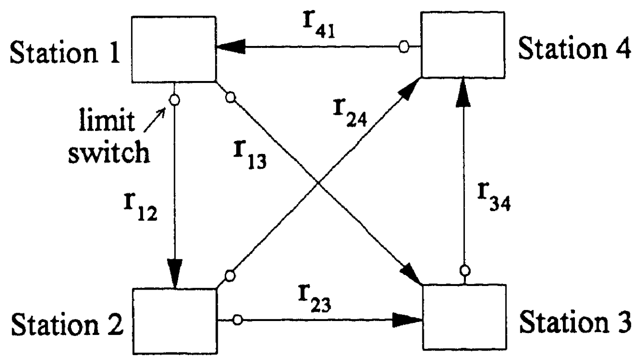


Figure 5.4 System layout directed graph

Figure 5.5 shows the jam-free Petri net model for short distance case. Figure 5.6 shows the jam-free Petri net model for long distance case. They can guarantee the jam-free condition among AGVs.

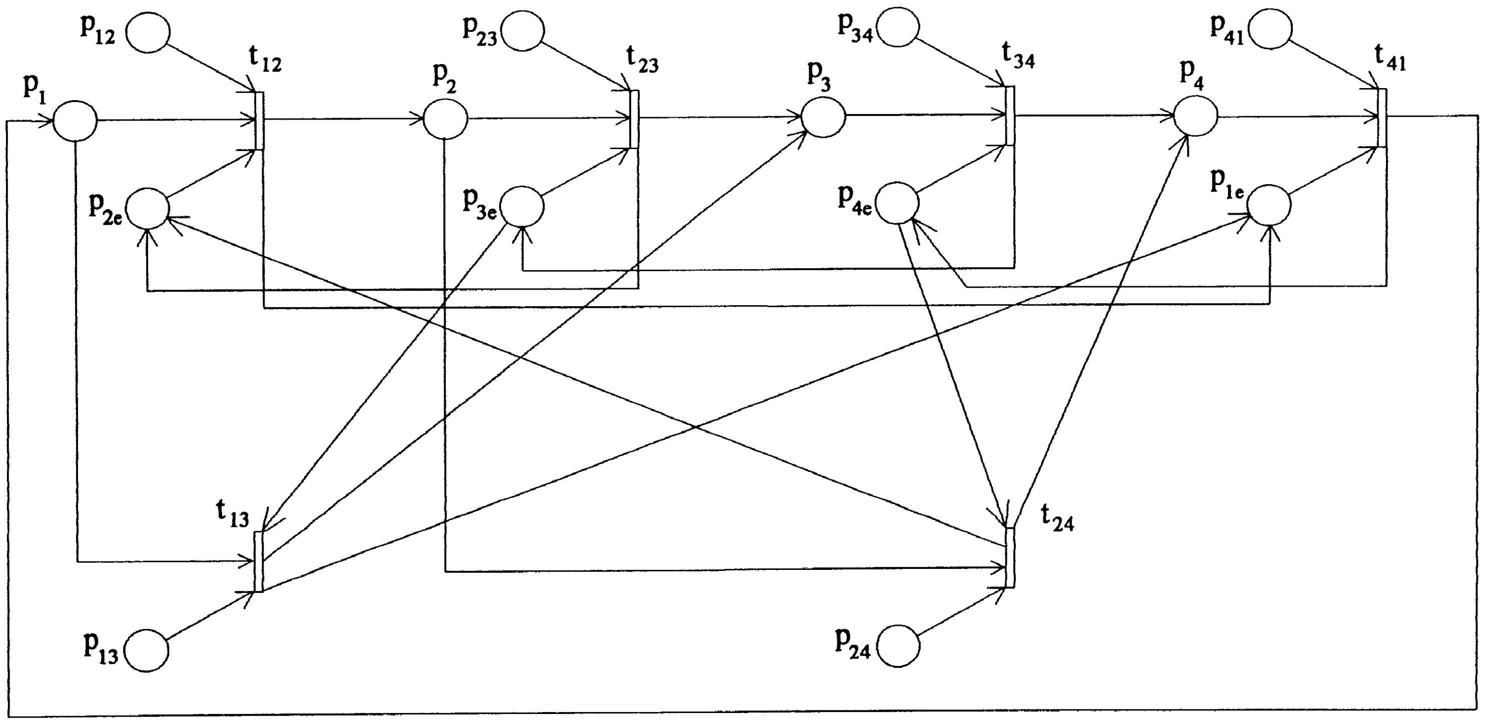


Figure 5.5 Jam-free Petri net model for short distance case

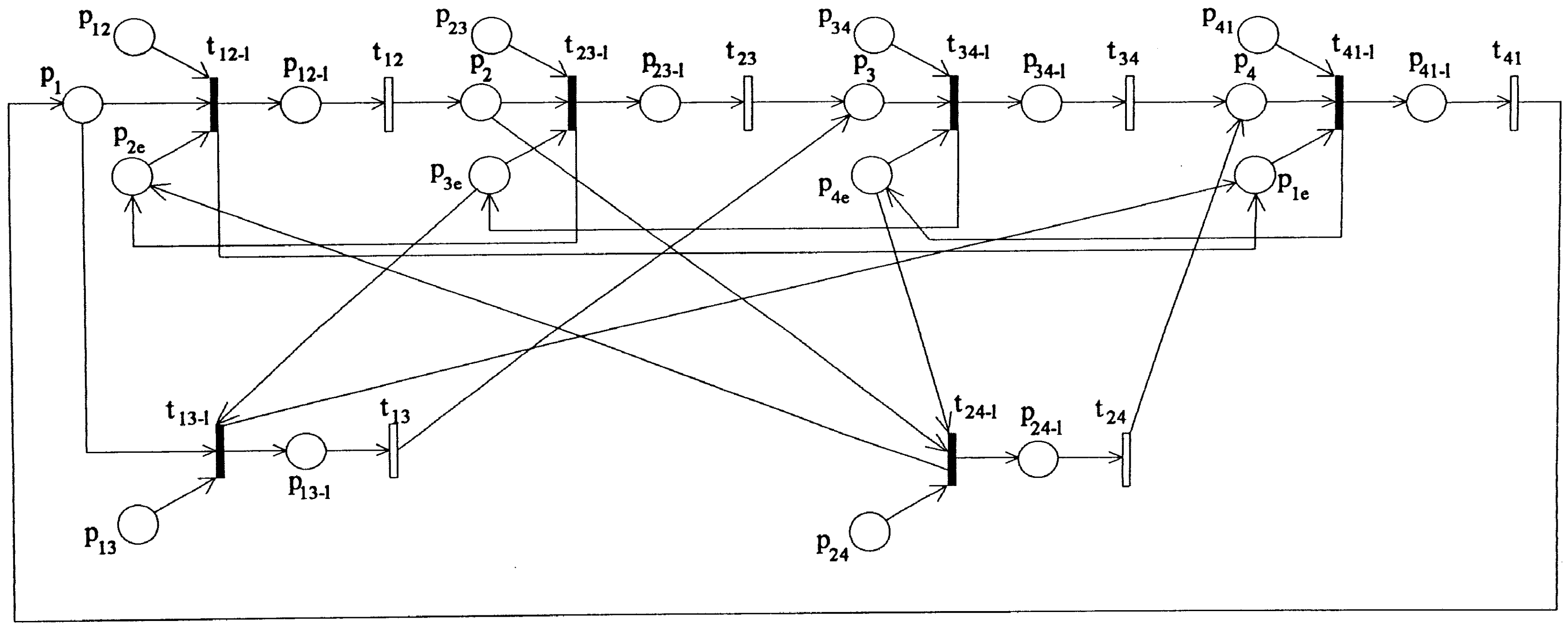


Figure 5.6 Jam-free Petri net model for long distance case

Table 5.3 Description of places and transitions of Figure 5.5

p_1	AGV stays at station 1	p_{1e}	Station 1 is empty
p_2	AGV stays at station 2	p_{2e}	Station 2 is empty
p_3	AGV stays at station 3	p_{3e}	Station 3 is empty
p_4	AGV stays at station 4	p_{4e}	Station 4 is empty
p_{12}	AGV can move from station 1 to 2	t_{12}	AGV moves from station 1 to 2
p_{13}	AGV can move from station 1 to 3	t_{13}	AGV moves from station 1 to 3
p_{23}	AGV can move from station 2 to 3	t_{23}	AGV moves from station 2 to 3
p_{24}	AGV can move from station 2 to 4	t_{24}	AGV moves from station 2 to 4
p_{34}	AGV can move from station 3 to 4	t_{34}	AGV moves from station 3 to 4
p_{41}	AGV can move from station 4 to 1	t_{41}	AGV moves from station 4 to 1

Table 5.4 Description of places and transitions of Figure 5.6

p_1	AGV stays at station 1	p_{1e}	Station 1 is empty
p_2	AGV stays at station 2	p_{2e}	Station 2 is empty
p_3	AGV stays at station 3	p_{3e}	Station 3 is empty
p_4	AGV stays at station 4	p_{4e}	Station 4 is empty
p_{12}	AGV can move from station 1 to 2	$t_{12\text{-}1}$	AGV moves from station 1 to limit switch of r_{12}
p_{13}	AGV can move from station 1 to 3	$t_{13\text{-}1}$	AGV moves from station 1 to limit switch of r_{13}
p_{23}	AGV can move from station 2 to 3	$t_{23\text{-}1}$	AGV moves from station 2 to limit switch of r_{23}
p_{24}	AGV can move from station 2 to 4	$t_{24\text{-}1}$	AGV moves from station 2 to limit switch of r_{24}
p_{34}	AGV can move from station 3 to 4	$t_{34\text{-}1}$	AGV moves from station 3 to limit switch of r_{34}
p_{41}	AGV can move from station 4 to 1	$t_{41\text{-}1}$	AGV moves from station 4 to limit switch of r_{41}
$p_{12\text{-}1}$	AGV touches the limit switch of r_{12}	t_{12}	AGV moves from station 1 to 2
$p_{13\text{-}1}$	AGV touches the limit switch of r_{13}	t_{13}	AGV moves from station 1 to 3
$p_{23\text{-}1}$	AGV touches the limit switch of r_{23}	t_{23}	AGV moves from station 2 to 3
$p_{24\text{-}1}$	AGV touches the limit switch of r_{24}	t_{24}	AGV moves from station 2 to 4
$p_{34\text{-}1}$	AGV touches the limit switch of r_{34}	t_{34}	AGV moves from station 3 to 4
$p_{41\text{-}1}$	AGV touches the limit switch of r_{41}	t_{41}	AGV moves from station 4 to 1

CHAPTER SIX

CONCLUSION AND FUTURE RESEARCH

6.1 Conclusion

This thesis discusses the scheduling problem of an FMS. The complete Petri net model is constructed for an FMS. A firing sequence of the Petri net from the initial marking to the final marking can be seen as a schedule of the modeled FMS. Once the final marking is reached, the path is constructed. Then the transition sequence of the path provides the order of starts of the activities. In order to obtain an optimal schedule and alleviate the computing complexity, the branch-and-bound algorithm is used. This method can give not only an optimal solution, but also can easily be implemented on computers, since the memory requirement is bounded and adjustable.

In order to get the long-term production rate of a system given the schedule marked graphs are applied. They are a special class of timed Petri nets used for modeling and analyzing job-shop systems. The modeling allows for evaluating the long-term periodic performance of the system under a deterministic and cyclic production process. Given any fixed processing times, the productivity (i.e., production rate) of the system can be determined from the initial state. The system will work at the maximal rate and the productivity is optimum.

Two AGVS models (shared and duty) are presented in this thesis. The shared model means that there are no firm objects served by an individual AGV. An AGV is devoted to a job until the job is completed. The duty model signifies that there is a firm object served by each AGV, i. e., each area or machine has its own AGV(s). This thesis takes two examples and discusses which case is more suitable for each AGVS type.

This thesis demonstrates the use of stochastic Petri nets for performance modeling and evaluation of AGVS. The results can help designers construct more efficient manufacturing systems, and engineers run factories more efficiently. Systematic discrete-event modeling tools and methods such as Petri nets shall play a significant role in design and implementation of intelligent manufacturing systems which contains flexible machines, material handling systems, intelligent robots and AGVs. It's very helpful for designers to use the Petri net method to model and evaluate different design alternatives before full implementation of an AGVS.

In an FMS, the transporting system may have more than one AGV which transports materials among different workstations. Thus, traffic jam problems must be considered and solved. The objective of the AGV jam-free control module is to guarantee the jam-free condition among AGVs. Petri net modeling of both short and long distance cases is given for the first time. The work will help design better AGVs.

6.2 Future Research

There are four areas which need to be explored in the future study:

1) The failure rates of machines and AGVs are not considered in this thesis. Therefore, an FMS that includes the failure rates of machines and AGVs requires future research.
2) The paths of AGVs and the structures of machines (input position and output position) in this thesis are just one of many types. The control of AGVs need future research when the structure is different, especially in connection with a particular physical layout of the paths.
3) To add the jam-free control module into the Petri net model of an FMS with AGVS. This thesis considers only the jam-free control module and the Petri net model of an FMS with AGVS separately.
4) To explore the scheduling applications to real industrial problems.
5) To find how to convert the initial Petri net model to the marked graph when the batch size is greater than one for each job.

APPENDIX A

Source Code for Branch and Bound Algorithm

```
        PROGRAM PetriNet

        parameter(nplace=12,ntran=8,nconfl=2,ktop=150)
        integer mark(nplace),Ma(nplace),Mu(nplace,2)
        integer ptime(nplace),fmark(nplace)
        integer iarc(nplace,ntran), oarc(nplace,ntran)
        integer incid(nplace,ntran)
        integer choice(nconfl,ntran)
        integer tenabl(ntran), tfire(ntran)
        integer clock, tseque(100)
        real dueinf(ntran,3)

        integer top,stack(ktop,nplace)
        integer wlist(ktop,ntran),tstack(ktop,1)
        common /one/ptime,iarc,oarc,choice,dueinf

        open(11,file='netinput',status='old')
        open(12,file='netout',status='new')

        write(*,*) 'Input the initial marking'
        read(11,*) (mark(i),i=1,nplace)
        write(*,*) 'Input the time associated with places'
        read(11,*) (ptime(i),i=1,nplace)
        write(*,*) 'Input the final marking'
        read(11,*) (fmark(i),i=1,nplace)

        do 5 i=1,nplace
           read(11,*) (iarc(i,j),j=1,ntran)
5       continue

        do 10 i=1,nplace
           read(11,*) (oarc(i,j),j=1,ntran)
10      continue

        close(11)

        do 15 i=1,nplace
           do 15 j=1,ntran
           incid(i,j)=oarc(i,j)-iarc(i,j)
15      continue

        do 20 i=1,nplace
           Mu(i,1)=0
           Mu(i,2)=10000000
20      continue

        clock=0
```

```
      do 25 i=1,nplace
         Ma(i)=mark(i)
25    continue

      write(12,30) (mark(i),i=1,nplace)
30       format(1x,'Initial marking',1x,20i3)
      write(12,35) (fmark(i),i=1,nplace)
35       format(1x,'Final marking',3x,20i3)

      top=ktop+1
      go to 85

40    do 45 i=1,nplace
         if (mark(i).NE.fmark(i)) go to 70
45    continue

50    top=top+1
      if (top .GT. ktop) go to 170

      do 55 j=1,ntran
         tfire(j)=0
55    continue

      do 65 j=1,ntran
         if (wlist(top,j) .EQ. 1) then
            wlist(top,j)=0
            tfire(j)=1
            do 60 i=1,nplace
               Ma(i)=stack(top,i)
               Mu(i,1)=0
60          continue
            clock=tstack(top,1)
            go to 120
         end if
65    continue

      go to 50

70    clock=Mu(1,2)

      do 75 i=2,nplace
         if (Mu(i,2) .LT. clock) then
            clock=Mu(i,2)
         end if
75    continue

      do 80 i=1,nplace
         if ((Mu(i,1) .GT. 0).AND.(Mu(i,2) .EQ. clock))
         then
            Ma(i)=Ma(i)+Mu(i,1)
            Mu(i,1)=0
            Mu(i,2)=10000000
         end if
80    continue
```

```
85      top=top-1

        do 90 i=1,nplace
           stack(top,i)=Ma(i)
90      continue

        tstack(top,1)=clock

        do 95 j=1,ntran
           tenabl(j)=0
           tfire(j)=0
95      continue

        kempty=0
        do 105 j=1,ntran
           do 100 i=1,nplace
              if (Ma(i) .LT. iarc(i,j)) goto 105
100        continue
           kempty=1
           tenabl(j)=1
105     continue

        do 110 j=1,ntran
           wlist(top,j)=tenabl(j)
110     continue

        do 115 j=1,ntran
           if (tenabl(j).eq.1) then
              tenabl(j)=0
              tfire(j)=1
              wlist(top,j)=0
              go to 120
           end if
115     continue

120     do 130 i=1,nplace
           ksum=0
           do 125 j=1,ntran
              ksum=ksum+incid(i,j)*tfire(j)
125        continue

           if (ksum .GT. 0) then
              Mu(i,1)=Mu(i,1)+ksum
              Mu(i,2)=clock+ptime(i)
           else
              Ma(i)=Ma(i)+ksum
           end if
           mark(i)=Ma(i)+Mu(i,1)
130     continue

        write(12,135) clock
135        format(/,1x,'Clock',4x,1i3)
        write(12,140) (wlist(top,j),j=1,ntran)
140        format(1x,'Wlist',4x,20i3)
```

```
        write(12,145) (tfire(j),j=1,ntran)
145        format(1x,'Firing',3x,20i3)
        write(12,150) (mark(i),i=1,nplace)
150        format(1x,'Marking',2x,20i3)

        do 155 j=1,ntran
           tenabl(j)=0
           tfire(j)=0
155     continue

        k0=0
        do 165 j=1,ntran
           do 160 i=1,nplace
              if (Ma(i) .LT. iarc(i,j)) goto 165
160        continue
           k0=1
           tfire(j)=1
           if (k0 .GT. 0) goto 120
165     continue

        go to 40

170     stop

        end
```

APPENDIX B

Elementary Circuits and Cycle Times of Figure 3.4

	Elementary Circuits(γ_i)	T_i (s)	N_i	$C_i = T_i / N_i$
$\gamma_1 =$	$(p_1, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p_1)$	9	1	9.0
$\gamma_2 =$	$(p_7, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p_7)$	9	1	9.0
$\gamma_3 =$	$(p_{13}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p_{13})$	8	1	8.0
$\gamma_4 =$	$(p_{19}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p_{19})$	7	1	7.0
$\gamma_5 =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_5, p_6, t_6, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, p_3, t_3, p_4, t_4, p^{1}_{26})$	19	2	9.5
$\gamma_6 =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p^{1}_{25}, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p^{1}_{26})$	12	1	12.0
$\gamma_7 =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p^{2}_{25}, t_1, p_2, t_2, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p_{13}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_3, p_4, t_4, p^{1}_{26})$	21	2	10.5
$\gamma_8 =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_5, p_6, t_6, p_1, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p^{1}_{26})$	18	2	9.0
$\gamma_9 =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_5, p_6, t_6, p_1, t_1, p_2, t_2, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p_{13}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_3, p_4, t_4, p^{1}_{26})$	27	3	9.0
$\gamma_{10} =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_5, p_6, t_6, p_1, t_1, p_2, t_2, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p^{1}_{26}, t_3, p_4, t_4, p^{1}_{26})$	24	3	8.0
$\gamma_{11} =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p^{2}_{25}, t_1, p_2, t_2, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_5, p_6, t_6, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_3, p_4, t_4, p^{1}_{26})$	30	3	10.0
$\gamma_{12} =$	$(p^{1}_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p^{2}_{25}, t_1, p_2, t_2, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_3, p_4, t_4, p^{1}_{26})$	18	2	9.0
$\gamma_{13} =$	$(p^{1}_{26}, t_7, p_8, t_8, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_9, p_{10}, t_{10}, p^{2}_{25}, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p^{1}_{26})$	20	2	10.0
$\gamma_{14} =$	$(p^{1}_{26}, t_7, p_8, t_8, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_3, p_4, t_4, p^{1}_{26})$	9	1	9.0

$\gamma_{15} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	29	3	9.7
$\gamma_{16} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	22	2	11.0
$\gamma_{17} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	21	2	10.5
$\gamma_{18} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	32	3	10.7
$\gamma_{19} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{6}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{13}, p_{14}, t_{14}, p^{2}_{27}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	28	3	9.3
$\gamma_{20} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	26	3	8.7
$\gamma_{21} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	28	3	9.3
$\gamma_{22} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p_{13}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	19	2	9.5
$\gamma_{23} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p_{13}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	20	2	10.0
$\gamma_{24} =$	$(p^{1}_{26}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{1}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p^{1}_{26})$	29	3	9.7
$\gamma_{25} =$	$(p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p_{5}, t_{5}, p_{6}, t_{6}, p^{1}_{27})$	10	1	10.0
$\gamma_{26} =$	$(p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p_{5}, t_{5}, p_{6}, t_{6}, p^{1}_{27})$	21	2	10.5
$\gamma_{27} =$	$(p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{1}_{27}, t_{5}, p_{6}, t_{6}, p^{1}_{27})$	18	2	9.0

$\gamma_{28} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p^2_{25}, t_1, p_2, t_2, p^3_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^3_{27}, t_{11}, p_{12}, t_{12}, p^4_{27}, t_5, p_6, t_6, p^1_{27})$	30	3	10.0
$\gamma_{29} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p^4_{26}, t_3, p_4, t_4, p^1_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p^2_{25}, t_1, p_2, t_2, p^3_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^3_{27}, t_{11}, p_{12}, t_{12}, p^4_{27}, t_5, p_6, t_6, p^1_{27})$	29	3	9.7
$\gamma_{30} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^4_{25}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^1_{27}, t_5, p_6, t_6, p^1_{27})$	20	2	10.0
$\gamma_{31} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^4_{25}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p^2_{25}, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	23	2	11.5
$\gamma_{32} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p^3_{27}, t_{11}, p_{11}, t_{12}, p^4_{27}, t_5, p_6, t_6, p^1_{27})$	12	1	12.0
$\gamma_{33} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^4_{25}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p_7, t_7, p_8, t_8, p^2_{26}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p^4_{26}, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	29	3	9.7
$\gamma_{34} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p^2_{25}, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	28	3	9.3
$\gamma_{35} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^1_{27}, t_5, p_6, t_6, p^1_{27})$	25	3	8.3
$\gamma_{36} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p^4_{26}, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	17	2	8.5
$\gamma_{37} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p^4_{26}, t_3, p_4, t_4, p^1_{26}, t_7, p_8, t_8, p_9, t_9, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^4_{27}, t_5, p_6, t_6, p^1_{27})$	26	3	8.7
$\gamma_{38} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p^3_{27}, t_{11}, p_{12}, t_{12}, p_7, t_7, p_8, t_8, p^2_{26}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^1_{25}, t_9, p_{10}, t_{10}, p^2_{25}, t_1, p_2, t_2, p_3, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	32	3	10.7
$\gamma_{39} =$	$(p^1_{27}, t_{13}, p_{14}, t_{14}, p^2_{27}, t_{21}, p_{22}, t_{22}, p^3_{27}, t_{11}, p_{12}, t_{12}, p_7, t_7, p_8, t_8, p^2_{26}, t_{19}, p_{20}, t_{20}, p^3_{26}, t_{15}, p_{16}, t_{16}, p^4_{26}, t_3, p_4, t_4, p_5, t_5, p_6, t_6, p^1_{27})$	21	2	10.5

$\gamma_{40} =$	$(p^{1}_{27}, t_{13}, p_{14}, t_{14}, p^{2}_{27}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p_{7}, t_{7}, p_{8}, t_{8}, p_{9}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p_{5}, t_{5}, p_{6}, t_{6}, p^{1}_{27})$	30	3	10.0
$\gamma_{41} =$	$(p^{1}_{27}, t_{13}, p_{14}, t_{14}, p^{2}_{27}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p_{7}, t_{7}, p_{8}, t_{8}, p_{9}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p_{3}, t_{3}, p_{4}, t_{4}, p_{5}, t_{5}, p_{6}, t_{6}, p^{1}_{27})$	24	2	12.0
$\gamma_{42} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p_{7}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	17	2	8.5
$\gamma_{43} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p_{7}, t_{7}, p_{8}, t_{8}, p^{2}_{26}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p^{4}_{26}, t_{3}, p_{4}, t_{4}, p_{5}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	27	3	9.0
$\gamma_{44} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p_{7}, t_{7}, p_{8}, t_{8}, p^{3}_{25}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	19	2	9.5
$\gamma_{45} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{1}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	18	2	9.0
$\gamma_{46} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p_{11}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p_{1}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	23	3	7.7
$\gamma_{47} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p^{4}_{25}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	12	1	12.0
$\gamma_{48} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p^{3}_{26}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	17	2	8.5
$\gamma_{49} =$	$(p^{1}_{25}, t_{9}, p_{10}, t_{10}, p^{2}_{25}, t_{1}, p_{2}, t_{2}, p^{3}_{25}, t_{23}, p_{24}, t_{24}, p_{19}, t_{19}, p_{20}, t_{20}, p_{21}, t_{21}, p_{22}, t_{22}, p^{3}_{27}, t_{11}, p_{12}, t_{12}, p^{4}_{27}, t_{5}, p_{6}, t_{6}, p^{1}_{27}, t_{13}, p_{14}, t_{14}, p_{15}, t_{15}, p_{16}, t_{16}, p_{17}, t_{17}, p_{18}, t_{18}, p^{1}_{25})$	29	3	9.7
	Maximum cycle time = Max $\{ T_i / N_i \}$ = 12s			

APPENDIX C

The Code of Figure 4-3 for SPNP

```
#include "user.h"
float x,y1,y2,z,v;

parameters() {
     iopt(IOP_PR_FULL_MARK,VAL_YES);
     iopt(IOP_PR_MC,VAL_YES);
     iopt(IOP_PR_RGRAPH,VAL_YES);
     iopt(IOP_PR_PROB,VAL_YES);
     v=input("AGV Speed Rate(value from 0.1 to 1.0):");
     x=4.0;
     y1=4.0*v;
     y2=1.0*v;
     z=1.0;
}

net() {
     place("p1");   init("p1",5);
     place("p2");
     place("p3");
     place("p4");
     place("p5");
     place("p6");
     place("p7");
     place("p8");
     place("p9");
     place("p10");
     place("p11");
     place("p12");
     place("p13");
     place("p14");
     place("p15");
     place("p16");
     place("p17");  init("p17",5);
     place("p18");  init("p18",1);
     place("p19");  init("p19",1);
     place("p20");  init("p20",1);
     place("p21");  init("p21",1);
     place("p22");
     place("p23");  init("p23",1);
     place("p24");  init("p24",1);
     place("p25");  init("p25",1);
     place("p26");  init("p26",1);
     place("p27");
     place("p28");  init("p28",1);
     place("p29");  init("p29",1);
     place("p30");  init("p30",1);
     place("p31");  init("p31",1);
     place("p32");
     place("p33");  init("p33",1);
place("p34");  init("p34",1);
place("p35");  init("p35",1);
place("p36");  init("p36",5);
place("p37");
place("p38");
place("p39");
place("p40");
place("p41");
place("p42");
place("p43");
place("p44");
place("p45");
place("p46");
place("p47");
place("p48");
place("p49");
place("p50");
place("p51");
place("p52");
place("p53");
place("p54");
place("p55");
place("p56");
place("p57");

trans("t1");   rateval("t1",100.0);
trans("t2");   rateval("t2",y1);
trans("t3");   rateval("t3",100.0);
trans("t4");   rateval("t4",z);
trans("t5");   rateval("t5",100.0);
trans("t6");   rateval("t6",y1);
trans("t7");   rateval("t7",100.0);
trans("t8");   rateval("t8",x);
trans("t9");   rateval("t9",100.0);
trans("t10");  rateval("t10",y1);
trans("t11");  rateval("t11",100.0);
trans("t12");  rateval("t12",x);
trans("t13");  rateval("t13",100.0);
trans("t14");  rateval("t14",y1);
trans("t15");  rateval("t15",100.0);
trans("t16");  rateval("t16",y2);
trans("t17");  rateval("t17",100.0);
trans("t18");  rateval("t18",100.0);
trans("t19");  rateval("t19",100.0);
trans("t20");  rateval("t20",100.0);
trans("t21");  rateval("t21",100.0);
trans("t22");  rateval("t22",100.0);
trans("t23");  rateval("t23",100.0);
trans("t24");  rateval("t24",y1);
trans("t25");  rateval("t25",100.0);
trans("t26");  rateval("t26",x);
trans("t27");  rateval("t27",100.0);
trans("t28");  rateval("t28",y1);
trans("t29");  rateval("t29",100.0);
trans("t30");  rateval("t30",z);
trans("t31");  rateval("t31",100.0);
trans("t32");  rateval("t32",y1);
trans("t33");  rateval("t33",100.0);
trans("t34");  rateval("t34",z);
trans("t35");  rateval("t35",100.0);
trans("t36");  rateval("t36",y1);
trans("t37");  rateval("t37",100.0);
trans("t38");  rateval("t38",y2);

iarc("t1","p1");iarc("t1","p17");iarc("t1","p18");
iarc("t1","p19");
oarc("t1","p2");

iarc("t2","p2");
oarc("t2","p3");oarc("t2","p18");

iarc("t3","p3");iarc("t3","p20");
oarc("t3","p4");oarc("t3","p52");

iarc("t4","p4");
oarc("t4","p5");

iarc("t5","p5");iarc("t5","p22");iarc("t5","p24");
iarc("t5","p23");
oarc("t5","p6");oarc("t5","p20");oarc("t5","p21");

iarc("t6","p6");
oarc("t6","p7");oarc("t6","p23");

iarc("t7","p7");iarc("t7","p25");
oarc("t7","p8");oarc("t7","p54");

iarc("t8","p8");
oarc("t8","p9");

iarc("t9","p9");iarc("t9","p27");iarc("t9","p29");
iarc("t9","p28");
oarc("t9","p10");oarc("t9","p25");oarc("t9","p26");

iarc("t10","p10");
oarc("t10","p11");oarc("t10","p28");

iarc("t11","p11");iarc("t11","p30");
oarc("t11","p12");oarc("t11","p56");

iarc("t12","p12");
oarc("t12","p13");

iarc("t13","p13");iarc("t13","p32");iarc("t13","p34");
iarc("t13","p33");
oarc("t13","p14");oarc("t13","p30");oarc("t13","p31");
```

```
iarc("t14","p14");
oarc("t14","p15");oarc("t14","p33");

iarc("t15","p15");iarc("t15","p35");
oarc("t15","p16");oarc("t15","p34");oarc("t15","p1");

iarc("t16","p16");
oarc("t16","p17");oarc("t16","p35");

iarc("t17","p52");iarc("t17","p21");
oarc("t17","p19");oarc("t17","p22");

iarc("t18","p21");iarc("t18","p53");
oarc("t18","p19");oarc("t18","p22");

iarc("t19","p54");iarc("t19","p26");
oarc("t19","p24");oarc("t19","p27");

iarc("t20","p26");iarc("t20","p55");
oarc("t20","p24");oarc("t20","p27");

iarc("t21","p56");iarc("t21","p31");
oarc("t21","p29");oarc("t21","p32");

iarc("t22","p31");iarc("t22","p57");
oarc("t22","p29");oarc("t22","p32");

iarc("t23","p17");iarc("t23","p18");iarc("t23","p19");
iarc("t23","p36");
oarc("t23","p37");

iarc("t24","p37");
oarc("t24","p18");oarc("t24","p38");

iarc("t25","p20");iarc("t25","p38");
oarc("t25","p39");oarc("t25","p53");

iarc("t26","p39");
oarc("t26","p40");

iarc("t27","p22");iarc("t27","p23");iarc("t27","p24");
iarc("t27","p40");
oarc("t27","p20");oarc("t27","p21");oarc("t27","p41");

iarc("t28","p41");
oarc("t28","p23");oarc("t28","p42");

iarc("t29","p25");iarc("t29","p42");
oarc("t29","p43");oarc("t29","p55");

iarc("t30","p43");
oarc("t30","p44");
```

```
        iarc("t31","p27");iarc("t31","p28");iarc("t31","p29");
        iarc("t31","p44");
        oarc("t31","p25");oarc("t31","p26");oarc("t31","p45");

        iarc("t32","p45");
        oarc("t32","p28");oarc("t32","p46");

        iarc("t33","p30");iarc("t33","p46");
        oarc("t33","p47");oarc("t33","p57");

        iarc("t34","p47");
        oarc("t34","p48");

        iarc("t35","p32");iarc("t35","p33");iarc("t35","p34");
        iarc("t35","p48");
        oarc("t35","p49");oarc("t35","p30");oarc("t35","p31");

        iarc("t36","p49");
        oarc("t36","p33");oarc("t36","p50");

        iarc("t37","p35");iarc("t37","p50");
        oarc("t37","p34");oarc("t37","p51");
        oarc("t37","p36");

        iarc("t38","p51");
        oarc("t38","p35");oarc("t38","p17");
}

assert() {return(RES_NOERR);}
ac_init() {}
ac_reach() {}
reward_type ef0() {return(rate("t15")+rate("t37"));}
ac_final() {pr_expected("throughout=",ef0);
            pr_std_average();}
```

APPENDIX D

The Code of Figure 4-4 for SPNP

```
#include "user.h"
float x,y1,y2,z,v;

parameters() {
     iopt(IOP_PR_FULL_MARK,VAL_YES);
     iopt(IOP_PR_MC,VAL_YES);
     iopt(IOP_PR_RGRAPH,VAL_YES);
     iopt(IOP_PR_PROB,VAL_YES);
     v=input("AGV Speed Rate(value from 0.1 to 1.0):");
     x=4.0;
     y1=4.0*v;
     y2=1.0*v;
     z=1.0;
}

net() {
     place("p1");   init("p1",4);
     place("p2");
     place("p3");
     place("p4");
     place("p5");
     place("p6");
     place("p7");
     place("p8");
     place("p9");
     place("p10");
     place("p11");
     place("p12");
     place("p13");
     place("p14");
     place("p15");
     place("p16");
     place("p17");  init("p17",2);
     place("p18");  init("p18",1);
     place("p19");  init("p19",1);
     place("p20");  init("p20",1);
     place("p22");  init("p22",1);
     place("p23");  init("p23",1);
     place("p24");  init("p24",1);
     place("p25");  init("p25",1);
     place("p27");  init("p27",1);
     place("p28");  init("p28",1);
     place("p29");  init("p29",1);
     place("p30");  init("p30",1);
     place("p32");  init("p32",1);
     place("p33");  init("p33",1);
     place("p34");  init("p34",1);
     place("p36");  init("p36",4);
     place("p37");
place("p38");
place("p39");
place("p40");
place("p41");
place("p42");
place("p43");
place("p44");
place("p45");
place("p46");
place("p47");
place("p48");
place("p49");
place("p50");
place("p51");
place("p52");
place("p53");
place("p54");
place("p55");
place("p56");
place("p57");

trans("t1");   rateval("t1",100.0);
trans("t2");   rateval("t2",y1);
trans("t3");   rateval("t3",100.0);
trans("t4");   rateval("t4",z);
trans("t5");   rateval("t5",100.0);
trans("t6");   rateval("t6",y1);
trans("t7");   rateval("t7",100.0);
trans("t8");   rateval("t8",x);
trans("t9");   rateval("t9",100.0);
trans("t10");  rateval("t10",y1);
trans("t11");  rateval("t11",100.0);
trans("t12");  rateval("t12",x);
trans("t13");  rateval("t13",100.0);
trans("t14");  rateval("t14",y1);
trans("t15");  rateval("t15",100.0);
trans("t16");  rateval("t16",y1);
trans("t17");  rateval("t17",y1);
trans("t18");  rateval("t18",y1);
trans("t19");  rateval("t19",y1);
trans("t20");  rateval("t20",y1);
trans("t21");  rateval("t21",y1);
trans("t22");  rateval("t22",y1);
trans("t23");  rateval("t23",100.0);
trans("t24");  rateval("t24",y1);
trans("t25");  rateval("t25",100.0);
trans("t26");  rateval("t26",x);
trans("t27");  rateval("t27",100.0);
trans("t28");  rateval("t28",y1);
trans("t29");  rateval("t29",100.0);
trans("t30");  rateval("t30",z);
trans("t31");  rateval("t31",100.0);
trans("t32");  rateval("t32",y1);
trans("t33");  rateval("t33",100.0);
trans("t34");  rateval("t34",z);
trans("t35");  rateval("t35",100.0);
trans("t36");  rateval("t36",y1);
trans("t37");  rateval("t37",100.0);
trans("t38");  rateval("t38",y1);

iarc("t1","p1");iarc("t1","p17");iarc("t1","p18");
iarc("t1","p19");
oarc("t1","p2");

iarc("t2","p2");
oarc("t2","p3");

iarc("t3","p3");iarc("t3","p20");
oarc("t3","p4");oarc("t3","p52");

iarc("t4","p4");
oarc("t4","p5");

iarc("t5","p5");iarc("t5","p22");iarc("t5","p24");
iarc("t5","p23");
oarc("t5","p6");oarc("t5","p20");

iarc("t6","p6");
oarc("t6","p7");

iarc("t7","p7");iarc("t7","p25");
oarc("t7","p8");oarc("t7","p54");

iarc("t8","p8");
oarc("t8","p9");

iarc("t9","p9");iarc("t9","p27");iarc("t9","p29");
iarc("t9","p28");
oarc("t9","p10");oarc("t9","p25");

iarc("t10","p10");
oarc("t10","p11");

iarc("t11","p11");iarc("t11","p30");
oarc("t11","p12");oarc("t11","p56");

iarc("t12","p12");
oarc("t12","p13");

iarc("t13","p13");iarc("t13","p32");iarc("t13","p34");
iarc("t13","p33");
oarc("t13","p14");oarc("t13","p30");

iarc("t14","p14");
oarc("t14","p15");

iarc("t15","p15");
oarc("t15","p34");oarc("t15","p1");oarc("t15","p16");
```

```
iarc("t16","p16");
oarc("t16","p32");oarc("t16","p33");

iarc("t17","p52");
oarc("t17","p17");oarc("t17","p18");
oarc("t17","p19");

iarc("t18","p53");
oarc("t18","p17");oarc("t18","p18");
oarc("t18","p19");

iarc("t19","p54");
oarc("t19","p22");oarc("t19","p23");
oarc("t19","p24");

iarc("t20","p55");
oarc("t20","p22");oarc("t20","p23");
oarc("t20","p24");

iarc("t21","p56");
oarc("t21","p27");oarc("t21","p28");
oarc("t21","p29");

iarc("t22","p57");
oarc("t22","p27");oarc("t22","p28");
oarc("t22","p29");

iarc("t23","p17");iarc("t23","p18");iarc("t23","p19");
iarc("t23","p36");
oarc("t23","p37");

iarc("t24","p37");
oarc("t24","p38");

iarc("t25","p20");iarc("t25","p38");
oarc("t25","p39");oarc("t25","p53");

iarc("t26","p39");
oarc("t26","p40");

iarc("t27","p22");iarc("t27","p23");iarc("t27","p24");
iarc("t27","p40");
oarc("t27","p20");oarc("t27","p41");

iarc("t28","p41");
oarc("t28","p42");

iarc("t29","p25");iarc("t29","p42");
oarc("t29","p43");oarc("t29","p55");

iarc("t30","p43");
oarc("t30","p44");
```

```
        iarc("t31","p27");iarc("t31","p28");iarc("t31","p29");
        iarc("t31","p44");
        oarc("t31","p25");oarc("t31","p45");

        iarc("t32","p45");
        oarc("t32","p46");

        iarc("t33","p30");iarc("t33","p46");
        oarc("t33","p47");oarc("t33","p57");

        iarc("t34","p47");
        oarc("t34","p48");

        iarc("t35","p32");iarc("t35","p33");iarc("t35","p34");
        iarc("t35","p48");
        oarc("t35","p49");oarc("t35","p30");

        iarc("t36","p49");
        oarc("t36","p50");

        iarc("t37","p50");
        oarc("t37","p34");oarc("t37","p36");oarc("t37","p51");

        iarc("t38","p51");
        oarc("t38","p32");oarc("t38","p33");
}

assert() {return(RES_NOERR);}
ac_init() {}
ac_reach() {}
reward_type ef0() {return(rate("t15")+rate("t37"));}
ac_final() {pr_expected("throughout=",ef0);
            pr_std_average();}
```

APPENDIX E

The Code of Figure 4-5 for SPNP

```
#include "user.h"
float x,y1,y2,z,v;

parameters() {
     iopt(IOP_PR_FULL_MARK,VAL_YES);
     iopt(IOP_PR_MC,VAL_YES);
     iopt(IOP_PR_RGRAPH,VAL_YES);
     iopt(IOP_PR_PROB,VAL_YES);
     v=input("AGV Speed Rate(value from 0.1 to 1.0):");
     x=4.0;
     y1=4.0*v;
     y2=1.0*v;
     z=1.0;
}

net() {
     place("p1");   init("p1",5);
     place("p2");
     place("p3");
     place("p4");
     place("p5");
     place("p6");
     place("p7");
     place("p8");
     place("p9");
     place("p10");
     place("p11");
     place("p12");
     place("p13");
     place("p14");
     place("p15");
     place("p16");
     place("p17");  init("p17",5);
     place("p18");  init("p18",1);
     place("p19");  init("p19",1);
     place("p20");  init("p20",1);
     place("p21");  init("p21",1);
     place("p22");
     place("p23");  init("p23",1);
     place("p24");  init("p24",1);
     place("p25");  init("p25",1);
     place("p26");  init("p26",1);
     place("p27");
     place("p28");  init("p28",1);
     place("p29");  init("p29",1);
     place("p30");  init("p30",1);
     place("p31");  init("p31",1);
     place("p32");
     place("p33");  init("p33",1);
place("p34");  init("p34",1);
place("p35");  init("p35",1);
place("p36");  init("p36",5);
place("p37");
place("p38");
place("p39");
place("p40");
place("p41");
place("p42");
place("p43");
place("p44");
place("p45");
place("p46");
place("p47");
place("p48");
place("p49");
place("p50");
place("p51");
place("p52");
place("p53");
place("p54");
place("p55");
place("p56");
place("p57");
place("p58");  init("p58",1);
place("p59");  init("p59",1);

trans("t1");   rateval("t1",100.0);
trans("t2");   rateval("t2",y1);
trans("t3");   rateval("t3",100.0);
trans("t4");   rateval("t4",z);
trans("t5");   rateval("t5",100.0);
trans("t6");   rateval("t6",y1);
trans("t7");   rateval("t7",100.0);
trans("t8");   rateval("t8",x);
trans("t9");   rateval("t9",100.0);
trans("t10");  rateval("t10",y1);
trans("t11");  rateval("t11",100.0);
trans("t12");  rateval("t12",x);
trans("t13");  rateval("t13",100.0);
trans("t14");  rateval("t14",y1);
trans("t15");  rateval("t15",100.0);
trans("t16");  rateval("t16",y2);
trans("t17");  rateval("t17",100.0);
trans("t18");  rateval("t18",100.0);
trans("t19");  rateval("t19",100.0);
trans("t20");  rateval("t20",100.0);
trans("t21");  rateval("t21",100.0);
trans("t22");  rateval("t22",100.0);
trans("t23");  rateval("t23",100.0);
trans("t24");  rateval("t24",y1);
trans("t25");  rateval("t25",100.0);
trans("t26");  rateval("t26",z);
trans("t27");  rateval("t27",100.0);
trans("t28");  rateval("t28",y1);
trans("t29");  rateval("t29",100.0);
trans("t30");  rateval("t30",z);
trans("t31");  rateval("t31",100.0);
trans("t32");  rateval("t32",y1);
trans("t33");  rateval("t33",100.0);
trans("t34");  rateval("t34",x);
trans("t35");  rateval("t35",100.0);
trans("t36");  rateval("t36",y1);
trans("t37");  rateval("t37",100.0);
trans("t38");  rateval("t38",y2);

iarc("t1","p1");iarc("t1","p17");iarc("t1","p18");
iarc("t1","p19");
oarc("t1","p2");

iarc("t2","p2");
oarc("t2","p3");oarc("t2","p18");

iarc("t3","p3");iarc("t3","p20");
oarc("t3","p4");oarc("t3","p52");

iarc("t4","p4");
oarc("t4","p5");

iarc("t5","p5");iarc("t5","p22");iarc("t5","p24");
iarc("t5","p23");
oarc("t5","p6");oarc("t5","p20");oarc("t5","p21");

iarc("t6","p6");
oarc("t6","p7");oarc("t6","p23");

iarc("t7","p7");iarc("t7","p25");
oarc("t7","p8");oarc("t7","p54");

iarc("t8","p8");
oarc("t8","p9");

iarc("t9","p9");iarc("t9","p27");iarc("t9","p29");
iarc("t9","p28");
oarc("t9","p10");oarc("t9","p25");oarc("t9","p26");

iarc("t10","p10");
oarc("t10","p11");oarc("t10","p28");

iarc("t11","p11");iarc("t11","p30");
oarc("t11","p12");oarc("t11","p56");

iarc("t12","p12");
oarc("t12","p13");

iarc("t13","p13");iarc("t13","p32");iarc("t13","p34");
iarc("t13","p33");
oarc("t13","p14");oarc("t13","p30");oarc("t13","p31");
```

```
iarc("t14","p14");
oarc("t14","p15");oarc("t14","p33");

iarc("t15","p15");iarc("t15","p35");
oarc("t15","p16");oarc("t15","p34");oarc("t15","p1");

iarc("t16","p16");
oarc("t16","p17");oarc("t16","p35");

iarc("t17","p52");iarc("t17","p21");
oarc("t17","p19");oarc("t17","p22");

iarc("t18","p21");iarc("t18","p57");
oarc("t18","p19");oarc("t18","p22");

iarc("t19","p54");iarc("t19","p26");
oarc("t19","p24");oarc("t19","p27");

iarc("t20","p26");iarc("t20","p55");
oarc("t20","p24");oarc("t20","p27");

iarc("t21","p56");iarc("t21","p31");
oarc("t21","p29");oarc("t21","p32");

iarc("t22","p31");iarc("t22","p53");
oarc("t22","p29");oarc("t22","p32");

iarc("t23","p17");iarc("t23","p58");iarc("t23","p29");
iarc("t23","p36");
oarc("t23","p37");

iarc("t24","p37");
oarc("t24","p58");oarc("t24","p38");

iarc("t25","p30");iarc("t25","p38");
oarc("t25","p39");oarc("t25","p53");

iarc("t26","p39");
oarc("t26","p40");

iarc("t27","p28");iarc("t27","p32");iarc("t27","p24");
iarc("t27","p40");
oarc("t27","p30");oarc("t27","p31");oarc("t27","p41");

iarc("t28","p41");
oarc("t28","p28");oarc("t28","p42");

iarc("t29","p25");iarc("t29","p42");
oarc("t29","p43");oarc("t29","p55");

iarc("t30","p43");
oarc("t30","p44");

iarc("t31","p27");iarc("t31","p23");iarc("t31","p19");
```

```
     iarc("t31","p44");
     oarc("t31","p25");oarc("t31","p26");oarc("t31","p45");

     iarc("t32","p45");
     oarc("t32","p23");oarc("t32","p46");

     iarc("t33","p20");iarc("t33","p46");
     oarc("t33","p47");oarc("t33","p57");

     iarc("t34","p47");
     oarc("t34","p48");

     iarc("t35","p22");iarc("t35","p59");iarc("t35","p34");
     iarc("t35","p48");
     oarc("t35","p49");oarc("t35","p20");oarc("t35","p21");

     iarc("t36","p49");
     oarc("t36","p59");oarc("t36","p50");

     iarc("t37","p35");iarc("t37","p50");
     oarc("t37","p34");oarc("t37","p51");
     oarc("t37","p36");

     iarc("t38","p51");
     oarc("t38","p35");oarc("t38","p17");
}

assert() {return(RES_NOERR);}
ac_init() {}
ac_reach() {}
reward_type ef0() {return(rate("t15")+rate("t37"));}
ac_final() {pr_expected("throughout=",ef0);
            pr_std_average();}
```

APPENDIX F

The Code of Figure 4-6 for SPNP

```
#include "user.h"
float x,y1,y2,z,v;

parameters() {
     iopt(IOP_PR_FULL_MARK,VAL_YES);
     iopt(IOP_PR_MC,VAL_YES);
     iopt(IOP_PR_RGRAPH,VAL_YES);
     iopt(IOP_PR_PROB,VAL_YES);
     v=input("AGV Speed Rate(value from 0.1 to 1.0):");
     x=4.0;
     y1=4.0*v;
     y2=1.0*v;
     z=1.0;
}

net() {
     place("p1");    init("p1",4);
     place("p2");
     place("p3");
     place("p4");
     place("p5");
     place("p6");
     place("p7");
     place("p8");
     place("p9");
     place("p10");
     place("p11");
     place("p12");
     place("p13");
     place("p14");
     place("p15");
     place("p16");
     place("p17");   init("p17",2);
     place("p18");   init("p18",1);
     place("p19");   init("p19",1);
     place("p20");   init("p20",1);
     place("p22");   init("p22",1);
     place("p23");   init("p23",1);
     place("p24");   init("p24",1);
     place("p25");   init("p25",1);
     place("p27");   init("p27",1);
     place("p28");   init("p28",1);
     place("p29");   init("p29",1);
     place("p30");   init("p30",1);
     place("p32");   init("p32",1);
     place("p33");   init("p33",1);
     place("p34");   init("p34",1);
     place("p36");   init("p36",4);
     place("p37");
place("p38");
place("p39");
place("p40");
place("p41");
place("p42");
place("p43");
place("p44");
place("p45");
place("p46");
place("p47");
place("p48");
place("p49");
place("p50");
place("p51");
place("p52");
place("p53");
place("p54");
place("p55");
place("p56");
place("p57");
place("p58");  init("p58",1);
place("p59");  init("p59",1);

trans("t1");   rateval("t1",100.0);
trans("t2");   rateval("t2",x);
trans("t3");   rateval("t3",100.0);
trans("t4");   rateval("t4",z);
trans("t5");   rateval("t5",100.0);
trans("t6");   rateval("t6",x);
trans("t7");   rateval("t7",100.0);
trans("t8");   rateval("t8",x);
trans("t9");   rateval("t9",100.0);
trans("t10");  rateval("t10",x);
trans("t11");  rateval("t11",100.0);
trans("t12");  rateval("t12",x);
trans("t13");  rateval("t13",100.0);
trans("t14");  rateval("t14",x);
trans("t15");  rateval("t15",100.0);
trans("t16");  rateval("t16",x);
trans("t17");  rateval("t17",x);
trans("t18");  rateval("t18",x);
trans("t19");  rateval("t19",x);
trans("t20");  rateval("t20",x);
trans("t21");  rateval("t21",x);
trans("t22");  rateval("t22",x);
trans("t23");  rateval("t23",100.0);
trans("t24");  rateval("t24",x);
trans("t25");  rateval("t25",100.0);
trans("t26");  rateval("t26",z);
trans("t27");  rateval("t27",100.0);
trans("t28");  rateval("t28",x);
trans("t29");  rateval("t29",100.0);
trans("t30");  rateval("t30",z);
trans("t31");  rateval("t31",100.0);
trans("t32");  rateval("t32",x);
trans("t33");  rateval("t33",100.0);
trans("t34");  rateval("t34",x);
trans("t35");  rateval("t35",100.0);
trans("t36");  rateval("t36",x);
trans("t37");  rateval("t37",100.0);
trans("t38");  rateval("t38",x);

iarc("t1","p1");iarc("t1","p17");iarc("t1","p18");
iarc("t1","p19");
oarc("t1","p2");

iarc("t2","p2");
oarc("t2","p3");

iarc("t3","p3");iarc("t3","p20");
oarc("t3","p4");oarc("t3","p52");

iarc("t4","p4");
oarc("t4","p5");

iarc("t5","p5");iarc("t5","p22");iarc("t5","p24");
iarc("t5","p23");
oarc("t5","p6");oarc("t5","p20");

iarc("t6","p6");
oarc("t6","p7");

iarc("t7","p7");iarc("t7","p25");
oarc("t7","p8");oarc("t7","p54");

iarc("t8","p8");
oarc("t8","p9");

iarc("t9","p9");iarc("t9","p27");iarc("t9","p29");
iarc("t9","p28");
oarc("t9","p10");oarc("t9","p25");

iarc("t10","p10");
oarc("t10","p11");

iarc("t11","p11");iarc("t11","p30");
oarc("t11","p12");oarc("t11","p56");

iarc("t12","p12");
oarc("t12","p13");

iarc("t13","p13");iarc("t13","p32");iarc("t13","p34");
iarc("t13","p33");
oarc("t13","p14");oarc("t13","p30");

iarc("t14","p14");
oarc("t14","p15");
```

```
iarc("t15","p15");
oarc("t15","p34");oarc("t15","p1");oarc("t15","p16");

iarc("t16","p16");
oarc("t16","p32");oarc("t16","p33");

iarc("t17","p52");
oarc("t17","p17");oarc("t17","p18");
oarc("t17","p19");

iarc("t18","p53");
oarc("t18","p17");oarc("t18","p58");
oarc("t18","p29");

iarc("t19","p54");
oarc("t19","p22");oarc("t19","p23");
oarc("t19","p24");

iarc("t20","p55");
oarc("t20","p28");oarc("t20","p32");
oarc("t20","p24");

iarc("t21","p56");
oarc("t21","p27");oarc("t21","p28");
oarc("t21","p29");

iarc("t22","p57");
oarc("t22","p27");oarc("t22","p23");
oarc("t22","p19");

iarc("t23","p17");iarc("t23","p58");iarc("t23","p29");
iarc("t23","p36");
oarc("t23","p37");

iarc("t24","p37");
oarc("t24","p38");

iarc("t25","p30");iarc("t25","p38");
oarc("t25","p39");oarc("t25","p53");

iarc("t26","p39");
oarc("t26","p40");

iarc("t27","p32");iarc("t27","p28");iarc("t27","p24");
iarc("t27","p40");
oarc("t27","p30");oarc("t27","p41");

iarc("t28","p41");
oarc("t28","p42");

iarc("t29","p25");iarc("t29","p42");
oarc("t29","p43");oarc("t29","p55");
```

```
        iarc("t30","p43");
        oarc("t30","p44");

        iarc("t31","p27");iarc("t31","p23");iarc("t31","p19");
        iarc("t31","p44");
        oarc("t31","p25");oarc("t31","p45");

        iarc("t32","p45");
        oarc("t32","p46");

        iarc("t33","p20");iarc("t33","p46");
        oarc("t33","p47");oarc("t33","p57");

        iarc("t34","p47");
        oarc("t34","p48");

        iarc("t35","p22");iarc("t35","p59");iarc("t35","p34");
        iarc("t35","p48");
        oarc("t35","p49");oarc("t35","p20");

        iarc("t36","p49");
        oarc("t36","p50");

        iarc("t37","p50");
        oarc("t37","p34");oarc("t37","p36");oarc("t37","p51");

        iarc("t38","p51");
        oarc("t38","p22");oarc("t38","p59");
}

assert() {return(RES_NOERR);}
ac_init() {}
ac_reach() {}
reward_type ef0() {return(rate("t15")+rate("t37"));}
ac_final() {pr_expected("throughout=",ef0);
            pr_std_average();}
```

REFERENCES

1. Baker, Kenneth R., *Introduction to Sequencing and Scheduling*. New York: Wiley, 1974.

2. Christofides, Nicos, Mingozzi, Aristide, Toth, Paolo and Sandi, Claudio, *Combinatorial Optimization*. New York: John Wiley & Sons, 1978.

3. Conway, Richard, Maxwell, W. William L. and Miller, Louis W., *Theory of Scheduling*. Ontario: Addison-Wesley, 1967.

4. Cooper, Leon and Steinberg, David, *Introduction to Methods of Optimization*. Philadelphia: Saunders, 1970.

5. DiCesare, Frank and Desrochers, Alan A., "Modeling, Control, and Performance Analysis of Automated Manufacturing Systems using Petri Nets", *Control and Dynamic Systems*, vol 17, pp. 121-172, 1991.

6. Freedman, P., "Time, Petri nets, and robotics", *IEEE Trans. Robotics Automation*, vol. 7, no. 4, pp. 417-433, 1991.

7. Gupta, D. and Buzacott, J. A., "A Framework for Understanding Flexibility of Manufacturing Systems", *Journal of Manufacturing Systems*, vol. 8, no. 2, pp. 89-95, 1989.

8. Hillion, Herve P., and Proth, Jean-Marie, "Performance Evaluation of Job-Shop Systems Using Timed Event-Graphs", *IEEE Trans. on Automatic Control*, vol. 34, no. 1, pp. 149-154, 1989.

9. Lee, D. Y. and DiCesare, F., "Scheduling and Supervisory Control of Flexible Manufacturing Systems Using Petri Nets and Heuristic Search", Ph.D. Dissertation, Rensselaer Polytechnic Institute, Troy, NY, May 1993.

10. Muller, Thomas, *Automated Guided Vehicles*. New York: IFS Ltd. UK, 1983.

11. Murata, T., "Petri Nets : Properties, Analysis and Applications", *Processings of the IEEE*, vol. 77, no. 4, 1989.

REFERENCES (Continued)

12. Ramchandani, C., "Analysis of asynchronous concurrent systems by timed Petri nets", Ph.D. dissertation, MIT, Cambridge, MA, May 1974.

13. Rodammer, Frederick A. and White, K. Preston Jr., "A Recent Survey of Production Scheduling", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 18, no. 6, pp. 841-851, 1988.

14. Shen, L., Chen, Q., Luh, J. Y., Chen, S. C. and Zhang, Z., "Truncation of Petri Net Models of Scheduling Problems for Optimun Solutions", *Japan/USA Symposium on Flexible Automation"* vol 2, pp. 1681-1688, 1992,

15. Sun, Tien-Hsiang, Cheng, Chao-Weng and Fu, Li-Chen, "Petri-Net Based Modeling And Scheduling for an FMS", *IEEE Trans. Industrial Electronis*, Dec. 1994.

16. Xiong, H. Henry, Zhou, MengChu and Manikopoulos, Constantine N., "Modeling and Performance Analysis of Medical Services Systems Using Petri Nets", Proc. of 1994 IEEE Conf. on Systems, Man and Cybernetics, San Diego, TX, pp. 2339-2342, Oct. 1994.

17. Zhao, Xiaoyong, "Automating Mason's Rule and Its Application to Analysis of Atochastic Petri Nets", Master Thesis, NJIT, Newark, NJ, January 1993.

18. Zhou, MengChu (Ed.), *Petri Nets in Flexible and Agile Autonation*, Boston, MA: Klumer Academic, 1995.

19. Zhou, MengChu, McDermott, Kevin and Patel, Paresh A., "Petri Net Synthesis and Analysis of a Flexible Manufacturing System Cell", *IEEE Trans. on Systems, Man, and Cybernetics*, vol 23, no. 2, pp. 523-531, 1993.

20. Zhou, MengChu and Leu, Ming C., "Modeling and Performance Analysis of a Flexible PCB Assembly Station Using Petri Nets", *The American Society of Mechanical Engineers*, pp. 410-416, 1991.