

Spring 1991

Petri net modeling and analysis of an FMS cell

Paresh A. Patel

New Jersey Institute of Technology

Follow this and additional works at: <https://digitalcommons.njit.edu/theses>



Part of the [Manufacturing Commons](#)

Recommended Citation

Patel, Paresh A., "Petri net modeling and analysis of an FMS cell" (1991). *Theses*. 1296.
<https://digitalcommons.njit.edu/theses/1296>

This Thesis is brought to you for free and open access by the Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

PETRI NET MODELING AND ANALYSIS OF AN FMS CELL

A Thesis Presented

by

Paresh A. Patel

to

The Department of Manufacturing Engineering

in partial fulfillment of the requirements
for the degree of

Master of Science

New Jersey Institute of Technology

Newark, New Jersey

October 1991

APPROVAL SHEET

Title of Thesis: Petri Net Modeling and Analysis of an FMS Cell

Name of Candidate: Paresh A. Patel
Master of Science in Manufacturing Engineering, 1991

Thesis and Abstract Approved by :

Dr. MengChu Zhou

Date

Assistant Professor

Electrical and Computer Engineering Dept.

Signature of other members of the
thesis committee

Dr. Raj Sodhi

Date

Director

Manufacturing Engineering Dept.

Associate Professor

Mechanical Engineering Dept.

Dr. Xiuli Chao

Date

Assistant Professor

Industrial Engineering Dept.

VITA

Name: Paresh A. Patel

Degree: M.S. in Manufacturing Engineering, October 1991

Collegiate institutions attended	Date	Degree	Date of Degree
New Jersey Institute of Technology	Jan. 1990-Aug. 1991	MSMnE	Oct. 1991
M.S.University of Baroda	Aug. 1984-Dec.1988	BSME	Jan. 1989

Dedicated

To my family

ACKNOWLEDGEMENTS

I take this opportunity to express my deep gratitude to Dr. Mengchu Zhou, Assistant Professor, Electrical and Computer Engineering Department of New Jersey Institute of Technology for his valuable guidance and help throughout the course of this work. It was his immaculate advise, which led my work to the set mark of culmination.

I am also very thankful to Dr. Raj Sodhi and Dr. Xiuli Chao for spending their precious time to review my work and to provide constructive insinuations for enhancing my work.

I would like to thank my parents for their help and understanding during the epoch of my study. Finally, I would like to thank my friends for helping me in whatever way they could, during the entire work of my thesis.

TABLE OF CONTENTS

	LIST OF FIGURE	iv
	LIST OF TABLES	v
	ABSTRACT	vi
CHAPTER I	INTRODUCTION	1
	✓ 1.1 Introduction and Previous Research	1
	1.2 Problem Definition	3
	1.3 Modeling, Analysis and Control of FMS with Petri Nets	4
	1.4 Petri Net Design Methods	6
	1.5 Outline of the Thesis	8
CHAPTER II	✓ PETRI NET THEORY.....	9
	2.1 Basic Definitions	9
	2.1.1 Structure of a Petri Net	9
	2.1.2 Marking of a Petri Net	13
	2.1.3 Rules of Operation	13
	2.1.4 Complementary Definitions	15
	2.2 Analysis of Petri Nets	17
	2.2.1 Qualitative Properties of Petri Nets	17
	2.2.2 Petri Net Analysis Approaches	20
	2.3 Petri Net Synthesis Approaches	25
	2.3.1 Top-Down Techniques	25
	2.3.2 Bottom-Up Techniques	27
	2.4 Graph Theoretic Definitions.....	28

	2.5 Classes of Petri Nets.....	29
	2.6 Conclusions	30
CHAPTER III	PETRI NET MODELING AND ANALYSIS OF FANUC MACHINING CENTER	31
	3.1 Modeling with Petri Nets	32
	3.2 Petri Net Model of a Fanuc Machining Center	34
	3.2.1 System Description	34
	3.2.2 Analysis of Petri Net Model	40
CHAPTER IV	PETRI NET MODELING AND ANALYSIS OF AN FMS CELL	42
	4.1 Description of a Flexible Manufacturing Cell	42
	4.2 Petri Net Design Process	45
	4.2.1 Modeling of Conveyor System	46
	A. Modeling	46
	B. Analysis	48
	4.2.2 Modeling of Milling Machine, GE Robot, Computer Vision System, and Parts Presentation Station	51
	A. Modeling	51
	B. Analysis	53
	4.2.3 Modeling of Drilling Station	54
	4.2.4 Final Petri Net Models	55
	A. Loading State	55
	B. Working State	57
CHAPTER V	TEMPORAL PETRI NETS FOR PERFORMANCE ANALYSIS	58
	5.1 Brief Literature Review	58

5.2 Coverability in Time	60
5.3 Analysis of an FMS Cell Using Deterministic Petri Nets	61
5.4 Analysis of an FMS Cell Using Stochastic Petri Nets	65
5.4.1 Overview of SPNP	66
5.4.2 Performance Analysis	66
CHAPTER VI	
CONCLUSIONS AND FUTURE RESEARCH	72
6.1 Conclusions and Summary of Results.....	72
6.2 Directions for Future Research	74
REFERENCES	76

LIST OF FIGURES

Figure 2.1	An Example of Petri Nets	12
Figure 2.2	An Example of Marked Petri Nets	13
Figure 2.3	A Pure Petri Net Equivalent of the given Impure Petri Net	16
Figure 2.4	An Example of Marked Graphs	30
Figure 2.5	An Example of State Machines	30
Figure 3.1	Petri Net Model of a Resource Sharing Problem	33
Figure 3.2	System Layout for Fanuc Machining Center	36
Figure 3.3.1-13	Petri Net Model for Fanuc Machining Center	36-39
Figure 3.4	The Reachability Graph of the Petri Net Model in Figure 3.3.1	40
Figure 4.1	Flexible Manufacturing System Cell	43
Figure 4.2	Loading State of the Conveyor System	46
Figure 4.3	Modeling of Material Handling System	47
Figure 4.4(a)	Modeling of the Milling Machine, GE Robot, Vision System, and Parts Presentation Station	52
Figure 4.4(b)	Reachability Graph for Figure 4.4(a)	52
Figure 4.5	Modeling of Drilling Workstation	54
Figure 4.6	Modeling of Loading Station	55
Figure 4.7	The Final Petri Net Model for the Loading State	56
Figure 4.8	The Final Petri Net Model for the Working State	57
Figure 5.1	The Marked Graph Equivalent of the Petri Net Model in Figure 4.8	62
Figure 5.2	System Throughput vs. Probability Rate of Part Acceptance α	69
Figure 5.3	Equipment Utilization vs. Probability Rate of Part Acceptance α	69
Figure 5.4	Average Cycle Time vs. Probability Rate of Part Acceptance α	70

LIST OF TABLES

Table 1	Places and Transitions in the Petri Net of Figure 3.3.1	35
Table 2	Places and Transitions in the Petri Net of Figure 4.3	48
Table 3	Places and Transitions in the Petri Net of Figure 4.4(a)	53
Table 4	Places and Transitions in the Petri Net of Figure 4.5	55
Table 5	Places and Transitions in the Petri Net of Figure 4.6	56
Table 6	Time Delays for Places and Transition Firings	63
Table 7	Elementary Circuits and their Cycle Times	64
Table 8	Elementary Circuits Considering Substructures	64
Table 9	Time Delays and Firing Rates Associated with the Transitions	68

ABSTRACT

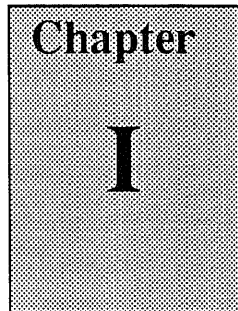
Petri nets have evolved into a powerful tool for the modeling, analysis and design of asynchronous, concurrent systems. This thesis presents the modeling and analysis of a flexible manufacturing system (FMS) cell using Petri nets. In order to improve the productivity of such systems, the building of mathematical models is a crucial step.

In this thesis, the theory and application of Petri nets are presented with emphasis on their application to the modeling and analysis of practical automated manufacturing systems. The theory of Petri nets includes their basic notation and properties. In order to illustrate how a Petri net with desirable properties can be modeled, this thesis describes the detailed modeling process for an FMS cell. During the process, top-down refinement, system decomposition, and modular composition ideas are used to achieve the hierarchy and preservation of important system properties. These properties include liveness, boundedness, and reversibility.

This thesis also presents two illustrations showing the method adopted to model any manufacturing systems using ordinary Petri nets. The first example deals with a typical resource sharing problem and the second the modeling of Fanuc Machining Center at New Jersey Institute of Technology.

Furthermore, this thesis presents the analysis of a timed Petri net for cycle time, system throughput and equipment utilization. The timed (deterministic) Petri net is first converted into an equivalent timed marked graph. Then the standard procedure to find the cycle time for marked graphs is applied. Secondly, stochastic Petri net is analyzed using SPNP software package for obtaining the system throughput and equipment utilization. This thesis is of significance in the sense that it provides industrial engineers and academic researchers with a comprehensive real-life example of applying Petri net theory to modeling and analysis of FMS cells. This will help them develop their own applications.

INTRODUCTION



1.1 INTRODUCTION AND PREVIOUS RESEARCH

Petri nets (PN) were originally developed by Carl Adam Petri in 1962 to model asynchronous concurrent systems. Since that time, considerable work has been done in both the theory and applications of Petri nets [1, 2, 26]. Petri nets were developed to model discrete event systems [4, 5]. By analyzing a Petri net model, useful information about the underlying system can be obtained [4, 5, 6, 8]. This information might reveal bottlenecks, deadlocks, etc., which exist and can be used to suggest changes without the need to run tests on the actual system. Because Petri nets are a mathematical model, various tools have been developed to analyze them [4, 6].

Petri nets are a useful tool for modeling systems with following characteristics:

1. *Concurrency and parallelism*: In a manufacturing system, many operations take place simultaneously.
2. *Asynchronous operations*: Machines complete their operations in variable amounts of time

and so the model must be able to represent asynchronous events or operations.

3. *Deadlock*: In this case, a state is reached where none of the processes can continue. This can happen when two processes share two resources. The order by which these resources are used and released could produce a deadlock.
4. *Conflicts*: This may occur when two or more processes require a common resource at the same time. e.g. two workstations might share a common transport system or might want access to the same database.
5. *Event driven*: The manufacturing system can be viewed as a sequence of discrete events. Since operations occur concurrently, the order of occurrence of events is not necessarily unique; it is one of many allowed by the system structure.

These types of systems have been difficult to accurately model with differential equations and queueing theory. Petri nets can provide accurate models for the following reasons:

- 1) Petri nets capture the precedence relations and structural interactions of concurrent and asynchronous events.
- 2) They are logical models derived from the knowledge of how the system works. As a result, they are easy to understand and their graphical nature is a good visual aid.
- 3) Deadlocks, conflicts, and buffer sizes can be modeled easily and concisely.
- 4) Petri net models have a well developed mathematical foundation that allows qualitative analysis of the system.
- 5) Finally, Petri net models can also be used to implement real-time control systems for automated manufacturing systems. They can sequence and coordinate the subsystems as a programmable logic controller does.

Generally speaking, Petri net modeling includes two parts [35] : ordinary Petri nets for system

behavior analysis and temporal Petri nets for system performance evaluation. The ordinary Petri nets, also called non-timed Petri nets, are used to analyze such system properties as deadlock-freeness, buffer-boundedness, reversibility (or re-initialization property), and conflict-freeness. After ordinary Petri nets are built, time variables can be used to be associated with their places and transitions, thus resulting in temporal Petri net models. These models serve to derive system performance indices including, for example, productivity and machine utilization. When there are several operational settings possible, different Petri net models can be obtained and used to study the optimal setting by comparing the results of Petri net based performance analysis.

Petri net theory previously developed for the modeling and analysis of manufacturing systems has been used in this report. The application of Petri nets for modeling flexible manufacturing systems (FMS) and production processes was previously proposed by Hack [9], Dubios and Stecke [10], and Narahari and Viswanadham [11]. Hack presented a brief description of the mapping of production schemata into Petri nets. His work centered on developing tools for analyzing safeness and liveness in a restricted class of Petri nets, free choice nets, using a top-down analysis approach. Dubios and Stecke emphasized the use of timed Petri nets in modeling FMSs for analyzing the quantitative aspects of FMS performances. A method for developing Petri net models for complex FMS by combining simpler subnets was proposed by Narahari and Viswanadham. They also discussed the analysis of qualitative system properties, such as the existence/absence of buffer overflows and deadlocks. Krogh and Sreenivas introduced the concept of essentially decision free places in Petri nets to represent the absence of unresolved resource allocation conditions.

1.2 PROBLEM DEFINITION

It is highly desirable for researchers, system analysts, and production engineers to have a uni-

fied mathematical model for modeling, analysis, simulation, and control of manufacturing systems [44]. Petri net theory developed and practiced during the past three decades provides such a possibility for this purpose. Applications of Petri nets to various fields in flexible manufacturing have resulted in many successes. The literature, however, do not pay sufficient attention to the detailed modeling process for practical flexible manufacturing systems, which is critical to a successful industrial application. The purpose of this thesis will provide real-life examples to demonstrate how a Petri net is built up and applied for cycle time analysis. Petri nets are a general graphical tool very well suited to the description of distributed and concurrent systems which exhibit synchronization and contention for share resource. Therefore, Petri nets have been claimed to be an ideal modeling tool for FMS. The objectives of this thesis are as follows:

1. To provide a tutorial on the basic Petri net terminology, Petri net properties, and the synthesis approaches;
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 describe an FMS cell which is operating in the factory floor of Information Technology Center, New Jersey Institute of Technology (NJIT);
4. To present the Petri net modeling process for the FMS cell; and
5. To analyze the resulting Petri net qualitatively and quantitatively.

1.3 MODELING, ANALYSIS AND CONTROL OF FMS WITH PETRI NETS

Mathematical modeling of FMS is the first step to analyze, simulate, and control the operations of such systems [35]. When concurrency and synchronization concepts become key to study of modern manufacturing systems control, Petri nets provide a most straightforward tool

to model these concepts. In addition, they can model conflicts, non-determinism, time information, and resource-sharing environments [20,42].

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 indicate the relationship between places and transitions. Tokens in places and their flow regulated by firing transition add the dynamics to Petri nets. Thus a marked Petri net can be used to study the dynamic behavior of the modeled discrete event systems.

The reachability graph approach is fundamental to analyze these properties. However it suffers from the state explosion. The second approach is the invariant method which uses the incidence matrix and related equations to analyze the properties. It may fail since invariants do not always convey a complete information of a Petri net. The third approach is reduction methods. At each reduction step, a subnet or structure is reduced to a simpler one by using the rules which will guarantee the preservation of concerned properties. The reduction terminates when no more substructures can be reduced or the reduced net can be easily proved on its properties. For the first case, the reachability graph method shall be further applied.

Furthermore, introduction of timing into Petri nets make it possible to conduct quantitative analysis as perturbation analysis methods [28] and queuing models do. This is done by associating timing information with transitions and/or places in a straightforward way. The Petri nets with deterministic times are used for decision-free systems whose behavior can be modeled by certain classes of Petri nets such as marked graphs. The cycle time can be obtained. The Petri nets with stochastic timing delays can be converted into their underlying Markov processes under certain assumptions [29] and then the classic Markovian approaches can be applied to derive performance measures such as throughput and utilization. Transfer function

based approaches have also become available for performance evaluation [40]. When arbitrary distributions are allowed, simulation is often needed.

Once system verification and performance analysis are conducted, we can convert the model into a Petri net controller studied by many researchers [13, 20, 34, 38]. Either an algorithm or compiled code can be implemented for real-time control of FMS. One advantage of Petri nets over automaton or formal language based approaches [30] lies in that it is not necessary to enumerate all possible global states at the design stage.

The top-down refinement and modular composition are very necessary for progressive modeling and design of complex FMS [20]. These two features are applicable to Petri net approaches, which are discussed next.

1.4 PETRI NET DESIGN METHODS

Since in a moderately sized manufacturing system, the complexity of design at the implementation level of detail may be unreasonable, researchers are seeking methods for the progressive synthesis of Petri net models. The discussion can be divided into two parts, bottom-up and top-down, according to the differences in the procedures that are used for constructing final system models. Bottom-up methods first divide the system into subsystems and specify them in detail. Then, sub-systems are modeled by subnets which are merged by sharing common transitions, places, and/or paths to obtain the system model.

Top-down methods, by contrast, use a macro-level model and neglect low-level detail initially. Then, the models are refined in a stepwise manner by expanding places and/or transitions to incorporate more detail into the models until the system is completely specified. Using such a method, a hierarchy can be well maintained.

A hybrid synthesis approach [42] has also been formulated in which top-down refinement of operations in flexible manufacturing is followed by bottom-up modeling of shared resources. The concepts: parallel and sequential mutual exclusions and the related theoretical results on synthesis are very important to shared resource modeling problems [41, 42].

System decomposition and modular composition are two keys to both approaches. One popular way to system decomposition is to separate an FMS into several functional parts such as transportation or material handling, and machining. Then Petri net models are constructed for each part. Finally, designers obtain a final net in three ways: sharing, linking, or refinement.

1. Sharing

Once subnets are constructed for each subsystem, one way to a complete net is to connect all nets by sharing the common places, transitions, or paths [3, 11, 16]. The assumption is that common places, transitions or paths are used when subnets are constructed. In order to get a system with desired properties, restrictions need to be imposed on the subnets, for example, all subnets being state-machine like nets. For general cases, designers can derive the invariants of the final net from those of subnets and then probably verify important properties according to these invariants.

2. Linking

By linking, we mean that a subnet links to another by adding extra arcs, places, and/or transitions. This method has been used to model some communication networks [24] but is rarely used in manufacturing since the physical layout or configuration of an FMS is often fixed.

3. Refinement

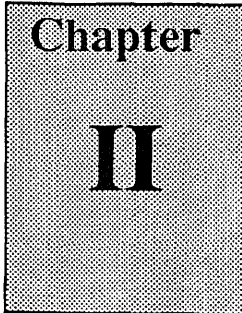
This method is a top-down method in the sense that certain places or transitions in a net will be refined by designed subnets or substructures. Thus two advantages can follow. One is that a hierarchy is naturally formulated. The other is that the desired properties can be easily verified

based on the properties of subnets. In order to achieve these two advantages, a macro-level net or certain subnets are constructed such that some of their places, transitions, and/or substructures represent more complex structures. Those complex structures are the other subnets which satisfy the certain conditions [19, 37]. It is this refinement method that will be used in our modeling process for the NJIT's FMS cell.

1.5 OUTLINE OF THE THESIS

The thesis is organized as follows: Chapter 2 gives a review of Petri net theory and presents the Graph theoretic view of Petri nets. In Chapter 3, a discussion on the application of Petri nets to model concurrent systems is presented. An example of resource sharing problem of a Fanuc Machining Center is presented for modeling and analysis of its qualitative properties. In Chapter 4, the Flexible Manufacturing System (FMS) Cell at NJIT is modeled using Petri nets and then analyzed to verify its qualitative properties. In process, the refinement techniques are used to develop the final Petri net model for the FMS Cell. In Chapter 5, the FMS under consideration is analyzed using timed Petri nets for cycle time, system throughput and equipment utilization. Finally, conclusions, as well as suggestions for future research are presented in Chapter 6.

PETRI NET THEORY

A rectangular box with a grey stippled background. Inside the box, the word "Chapter" is written in a serif font at the top, and a large Roman numeral "II" is centered below it.

Chapter II

Petri nets will be used throughout this report as a tool, for modeling, analyzing and simulating the existing physical FMS cell. Because of the various features of Petri nets, which will be discussed in Chapter III, they are very appropriate for modeling FMSs. They lead to a description of a system that can be investigated analytically, simulated and implemented for control. This chapter reviews the pertinent aspects of a Petri net theory [4, 5].

2.1 BASIC DEFINITIONS

2.1.1 STRUCTURE OF A PETRI NET

A Petri net is represented as a graph comprising a set of nodes and a set of arcs. The graph has two types of nodes: *Places* (represented as circles) and *Transitions* (represented as bars). These nodes, places and transitions, are connected by directed arcs from places to transitions and from transitions to places. If an arc is directed from node i to node j (either from a place to a transition or a transition to a place), then i is an *input* to j , and j is an *output* of i .

To complete the definition, it is necessary to define the relationship between places and transitions. This is done by specifying two functions connecting transitions to places: I , the input function, and O , the output function.

In a Petri net graph, the input and output functions are represented as arcs going from places to transitions and transitions to places, respectively. An arc is directed from place p_i to transition t_j if the place is an input of the transition. Similarly an arc is directed from a transition t_j to a place p_i if the place is an output of the transition. A Petri net is thus *directed* graph, since the arcs are directed. The arcs can be multiple.

Moreover, since the nodes can be partitioned into two sets (places and transitions) such that each arc is directed from an element of one set to an element of the other set, a Petri net is a *bi-partite directed graph*.

Formally, an ordinary Petri net (PN) is a five-tuple,

$$Z = (P, T, I, O, m) \quad (2.1)$$

where

$P = \{p_1, p_2, \dots, p_n\}$, $n > 0$, and is a set of n places;

$T = \{t_1, t_2, \dots, t_s\}$, $s > 0$, and is a set of s transitions;

I is an $n \times s$ matrix indicating the places which are the input of each transition. It is a mapping : $P \times T \rightarrow \{0, 1, 2, \dots\}$ corresponding to the set of directed arcs *from places to transitions*.

O is an $n \times s$ matrix indicating the places which are the output of each transition. It is

a mapping : $P \times T \rightarrow \{0, 1, 2, \dots\}$ corresponding to the set of directed arcs *from transitions to places*.

$m : P \rightarrow N$ and is a marking whose i^{th} component represents the number of tokens in the i^{th} place. An initial marking is denoted by m_0

These four items: the set of *places*, the set of *transitions*, the *input function*, and the *output function* define the *structure* of the Petri net [5].

A Petri net graphically consists of :

1. Circles (called places) representing conditions or availability of resources e.g. machines, parts, data, etc.
2. Bars (called transitions) representing the initiation or termination of an event. Transitions can be also used to represent events.
3. Black dot (called a token) in a operation place representing the operation in that place being executed, while that in a resource place representing the availability of the corresponding resources.
4. A pattern of tokens in a Petri net (called a marking) representing the state of the system.

We denote *postset* and *preset* as follows :

- t_j^\bullet is the set of all output places of transition t_j .
- ${}^\bullet t_j$ is the set of all input places of transition t_j .
- p_i^\bullet is the set of all output transitions of place p_i .
- ${}^\bullet p_i$ is the set of all input transitions of place p_i .

For the Petri net shown in Figure 2.1 :

$$P = \{p_1, p_2, p_3, p_4, p_5, p_6\}$$

$$T = \{t_1, t_2, t_3, t_4, t_5\}$$

$$\bullet t_1 = \{p_1\} \quad \bullet t_2 = \{p_1, p_2\} \quad \bullet t_3 = \{p_3\} \quad \bullet t_4 = \{p_4, p_5\} \quad \bullet t_5 = \{p_6\}$$

$$t_1^\bullet = \{p_3\} \quad t_2^\bullet = \{p_6\} \quad t_3^\bullet = \{p_4, p_5\} \quad t_4^\bullet = \{p_6\} \quad t_5^\bullet = \{p_4\}$$

$$\bullet p_1 = \emptyset \quad \bullet p_2 = \emptyset \quad \bullet p_3 = \{t_1\} \quad \bullet p_4 = \{t_3, t_5\} \quad \bullet p_5 = \{t_3\}$$

$$\bullet p_6 = \{t_2, t_4\}$$

$$p_1^\bullet = \{t_1, t_2\} \quad p_2^\bullet = \{t_2\} \quad p_3^\bullet = \{t_3\} \quad p_4^\bullet = \{t_4\} \quad p_5^\bullet = \{t_4\}$$

$$p_6^\bullet = \{t_5\}$$

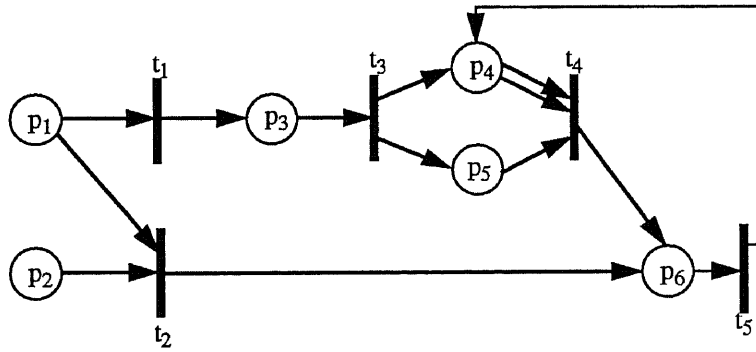


Figure 2.1 An Example of Petri Nets

$$I = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$O = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

2.1.2 MARKING OF A PETRI NET

A marking m of a Petri net is an assignment of *tokens* to the places in that net. Tokens reside in the places of the net. The number and position of the tokens in a net may change during its execution. On a Petri net graph, tokens are represented by small solid dots inside the circles representing the places of the net. Since any number of the tokens can be assigned to the places, there is an infinite number of markings for a Petri net structure.

$m = [m(p_1), m(p_2), \dots, m(p_n)]^T$ is an n -dimensional integer valued vector indicating the number of tokens in each place and is known as the marking of the Petri net.

For the example of Figure 2.2 : $m = [1, 0, 0, 2, 0, 1]^T$

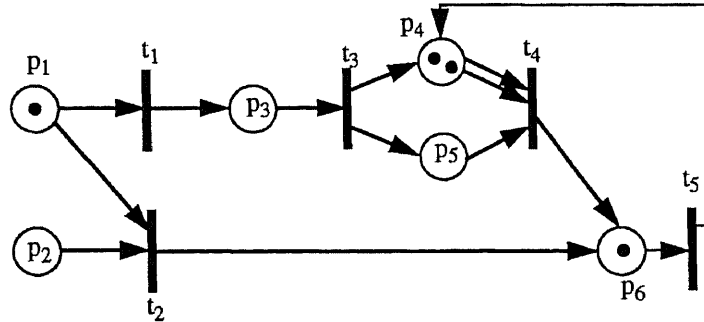


Figure 2.2 An Example of Marked Petri Nets

2.1.3 RULES OF OPERATION

A Petri net executes by *firing* transitions. A transition may fire if it is *enabled*. A transition is enabled if there are at least $I(p,t)$ tokens in $p \in P$. A transition fires by removing $I(p,t)$ tokens from P and then placing $O(p,t)$ into p . The movement of tokens through the Petri net graph rep-

resents the flow of information or control in the system.

Hence, a transition t is enabled by a marking m_0 if every input place of this transition contains at least $I(p,t)$ tokens, i.e., $\forall p \in {}^\bullet t, m_0(p) \geq I(p,t)$

Every transition enabled by marking m_0 can fire. When a transition fires $I(p,t)$ tokens are removed from p and $O(p,t)$ tokens are added to any $p \in P$. Therefore a new marking m_1 obtained by the firing of transition t , verifies for each p :

$$m_1(p) = m_0(p) - I(p,t) + O(p,t) \quad \forall p \in P \quad (2.2)$$

For example in the Petri net shown in Figure 2.2, t_1 and t_5 are enabled, but not t_2 . After t_1 has fired once, the new marking is : $m_1 = (0, 0, 1, 2, 0, 1)^T$.

This is represented as:

$$m_0 \xrightarrow{t} m_1$$

For any transition t_j and all places p_i ($i \in \{1, 2, \dots, n\}$), the relation (2.2) can be written in a compact form as:

$$m_1 = m_0 + (O - I) \cdot X_j = m_0 + CX_j \quad (2.3)$$

where,

X_j is a m -dimensional vector representing the number of firings of each transition and all the components of X_j equal to zero except the j^{th} one, which equals 1, i.e. $X_j = (0, 0, \dots, 1, \dots, 0)^T$ and

$C = O(p,t) - I(p,t)$ is called an incidence matrix.

2.1.4 COMPLEMENTARY DEFINITIONS

Firing Sequence

Consider a sequence of transition firings, denoted by:

$$\sigma_s = t_{j1}, t_{j2}, \dots, t_{js}$$

which means that transition t_{j1} fires first, then transition t_{j2} and so on until transition t_{js} . The resulting marking is denoted by:

$$m_0 \xrightarrow{\sigma_s} m_s \quad (2.4)$$

Parikh Mapping (Firing Vector)

A firing sequence σ_s can be associated with a *firing vector* N_s , also called the *Parikh mapping* of the sequence σ_s : N_s is an m dimensional non-negative integer vector whose j^{th} component corresponds to the number of occurrences of transition t_j in the sequence σ_s . Now, relation (2.3) can be generalized in the form :

$$m_s = m_0 + CN_s \quad (2.5)$$

The algebraic equation (2.5) enables the direct computation of the new marking reached by any sequence of transition firings. However, some information is lost when using the vector N_s , since the *order* of the firing sequence is not specified.

Reachability Set

A marking m' is immediately reachable from m if the firing of some enabled transition in m yields m' . Given the initial marking m_0 of the Petri net, we will call *reachability set* or *forward marking class* (denoted by $R(m_0)$) the set of all possible reachable markings. In other words, a marking m belongs to $R(m_0)$ if there exists a firing sequence σ leading from m_0 to m . Thus,

$$R(m_0) = \{m \mid \exists \sigma \in T^*, m_0 \rightarrow m\} \quad (2.6)$$

where T^* represents the set of all possible orders of the elements in T

Self-loop And Pure Petri Nets

A place p and a transition t are in a self-loop if p is both an input and an output place of t . A Petri net will be pure if it does not contain self-loops. Given a Petri net that is not pure, we can always get an equivalent pure Petri net by introducing dummy places and dummy transitions as shown in Figure 2.3.

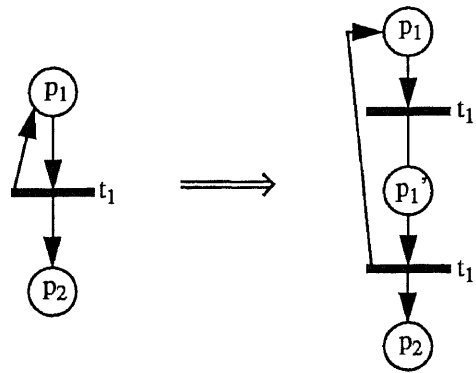


Figure 2.3 A Pure Petri net Equivalent of the given Impure Petri Net

Subnet Of A Petri Net

A subnet of a Petri net $PN = (P, T, I, O)$ is a Petri net $PN_s = (P_s, T_s, I_s, O_s)$ such that :

$$P_s \subset P ; T_s \subset T$$

I_s and O_s are the restrictions of I and O to $P_s \times T_s$, respectively.

2.2 ANALYSIS OF PETRI NETS

Once a system has been modeled by using a Petri net, it is desirable to analyze the net to determine which properties the net possesses. Various analysis methods for Petri nets such as reachability graph, invariant analysis and reduction approaches will be discussed in the later section.

2.2.1 QUALITATIVE PROPERTIES OF PETRI NETS

This section will discuss the properties we can study, given a Petri net.

(1) Boundedness

A Petri net is *bounded*, if for each place in the net, there exists an upper bound to the number of tokens that can be there simultaneously given initial marking m_0 . In other words, a marking m_0 is *k-bounded* if there exists a positive integer k , such that for every reachable marking m (an element of the reachability set $R(m_0)$), the number of tokens in each place is bounded by k .

If $k = 1$, the marking is said to be *safe*.

In a manufacturing environment, the boundedness or safeness of a Petri net indicates the absence of overflows in the modeled system [11, 37]. For example, a buffer in a production facility will have a finite capacity, and its representation as a bounded place will guarantee that the resulting control code will not allow this capacity to be exceeded. Safeness means no more than one token will mark the place. This implies that there is no possibility to restart the ongoing process if the place represents the process. For instance, if a place represents a machine that can only process one part at a time, safeness would guarantee that no other parts are loaded until the current one is completed. If a place happens to represent the availability of a single resource, then this place must be safe.

- A Petri net PN is *structurally bounded* if PN is bounded for any initial marking.

(2) Liveness

A Petri net is live given initial marking m_0 if there always exists a firing sequence σ to enable each transition in the net for any marking in $R(m_0)$.

A transition that cannot fire is a redundant transition and can be eliminated from the net. However, if such a transition exists in a net model, it needs to be identified since it may represent an error in the model or an inconsistency in the system being modeled. A transition is dead in a marking if there is no sequence of transition firings that can enable it. A transition is potentially firable if there exists some firing sequence that enables it. A transition is live if it is potentially firable in all reachable markings.

For an initial marking m_0 and any transition t_j , t_j is live if it is potentially firable in any reachable marking, $m \in R(m_0)$. In other words, every transition of the net can fire an infinite number of times.

Liveness is tied to the concept of deadlocks and deadlock-freeness, as related to the modeling

of operating systems [37]. Thus, it may be important not only that a transition be firable in a given marking, but that it stay potentially firable in all markings reachable from that marking. If this is not true, then it is possible to reach a state in which the transition is dead, perhaps signifying a possible deadlock.

(3) Reversibility

A Petri net is reversible if for every $m \in R(m_0)$ then $m_0 \in R(m)$. i.e. the initial marking is reachable from all reachable markings. Reversibility means re-initializability [37]. It implies that the system will finally return to its initial state from any current states (including failure states) [36].

(4) Conservative Petri Net

A Petri net is conservative if the number of tokens in the net is constant. This implies that each transition in such a net is conservative, in the sense that the number of inputs of each firable transition is equal to the number of outputs of that transition. More generally, weights can be defined for each place allowing the number of tokens to change as long as the weighted sum is constant. This is an important concept, since, if tokens are to represent resources, then it follows that since resources can neither be created nor destroyed, tokens should also be neither created nor destroyed. However, two resources can be combined into one. This corresponds to synchronization cases [42].

(5) Consistency

A Petri net is *consistent* if and only if there exists a marking m and a firing sequence σ such that:

- 1) σ brings the marking m of the net back to itself and

2) σ fires each transition at least once.

σ is called a *cyclic firing sequence*.

Consistency is different from reversibility in the sense that in case of reversibility, $m_0 \in R(m)$ $\forall m \in R(m_0)$ without guaranting that σ fires each transition at least once.

2.2.2 PETRI NET ANALYSIS APPROACHES

1) Reachability Graph

The reachability graph represents all of the possible reachable markings. Starting with the initial marking m_0 , firing a transition leads to a new marking that might enable other transitions. Taking each of these new markings as a new root, one can recursively generate all of the reachable markings. This tree structure preserves the firing order of the transitions.

The reachability tree for a marked Petri net is constructed as follows [47]:

- a) Let the initial marking be the root node and tag it “new.”
- b) *While* new markings exist *do*
- c) Select a new marking m .
- d) If m is identical to another node in the tree which is not new, then tag m to be “old” and stop processing m .
- e) If no transition is enabled in m , tag m to be “terminal.”

For every transition t enabled in m

- 1) Obtain the marking m' which results from firing t in m .
- 2) If there exists a path from the root to m containing a marking m'' such that $m' > m''$, then replace $(m')_i$ by ω wherever $(m')_i > (m'')_i$
 where $\omega \pm x = \omega$, $x < \omega$ and $\omega \leq \omega$ for every integer x
- 3) Introduce m' as a node, draw an arc from m to m' labeled t , and tag m' to be "new."

End

It can be shown that the above procedure always terminates in a finite number of steps resulting in a finite tree. Also, a place p_i is unbounded if and only if the tree contains a marking with $(m)_i = \omega$. For bounded nets, each node is a reachable marking and the tree contains all reachable markings.

In general, the liveness and boundedness properties are difficult to verify. This indeed requires the construction of the reachability graph. A basic approach to analyzing Petri nets is to use the reachability tree. The nodes of a reachability tree of a marked Petri net represent reachable markings of a net. For bounded nets, each node is a reachable marking and the tree contains a limited number of nodes. Detailed information about system's behavior can be obtained from analysis of its reachability tree. The major disadvantage is that the reachability tree can easily become complex, large and unmanagable.

2) Invariant Analysis

In this section, we review the important concept of Petri net invariants. A knowledge of the invariants of a Petri net is very useful for establishing/disproving some important properties of Petri nets such as boundedness, conservativeness, properness, and liveness.

P-invariant

A P-invariant is a column vector x , denoting a set of weighted places, such that

$$x^T C = 0 \quad (2.7)$$

where C is the incidence matrix. Considering (2.3) and (2.7), we have the following result

$$x^T m = x^T m_0 \quad (2.8)$$

This equality implies that the total number of tokens, weighted by the P-invariant, for any reachable marking is constant. P-invariants can be used for checking liveness and boundedness in the analysis of Petri net (see [18] for detail).

T-invariant

A T-invariant is a column vector y , representing a set of firing counts of transitions, such that

$$C y = 0 \quad (2.9)$$

Considering (2.3) and (2.9) with $y = X_j$, we obtain the following equation

$$m = m_0 \quad (2.10)$$

This means that firing of transitions as in a T-invariant brings the marking back to the initial marking. When analyzing a Petri net, the net's T-invariants can be used for determining whether the net is not reversible (see [18] for detail). In other words, (2.9) does not guarantee reversibility but every reversible net must satisfy (2.9).

Incidence Matrix

The incidence matrix (also called Flow Matrix), denoted by C , characterizes the structure of a PN in the following way : The columns of the matrix correspond to the transitions of the net and the rows, to the places. C is therefore, a $n \times s$ matrix, such that:

$$C(p, t) = O(p, t) - I(p, t) \quad \forall p \in P, t \in T \quad (2.11)$$

It provides a description of the resultant connectivity of the Petri net.

The Petri net of Figure 2.1 has, for example, the following incidence matrix:

$$C = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 1 & -1 \end{bmatrix}$$

The following theorems [18] give a sufficient condition for boundedness, a necessary and sufficient condition for conservativeness, and a necessary condition for reversibility, respectively.

Theorem A

A Petri net is bounded if there exists a p -invariant x all of whose entries are strictly positive.

Theorem B

A Petri net is conservative iff there exists a p -invariant x all of whose entries are equal to unity.

Theorem C

A Petri net is not reversible if its only r -invariant is the trivial invariant, that is with all its components equal to zero.

In most cases, a knowledge of the invariants together with some additional information about the modeled system will be sufficient to completely investigate these above properties and also liveness.

3) Reduction Approaches

Reduction techniques are primarily used to mitigate the effort of checking Petri net properties because most reduction steps do not lose important information about the net. Moreover, if we consider the converse process of reduction, i.e., expansion (refinement), then the same techniques may be used to synthesize Petri net models while preserving important properties. In this area, Berthelot [22] has developed a set of powerful transformation rules where the reduction procedure is realized by eliminating arcs and by replacing subnets with places and transitions. However, Lee and Favrel [21] noticed two problems with existing reduction methods: (1) no hierarchical reduction or decomposition is provided, and (2) the conditions for a reducible subnet are defined on the dynamic aspect of the net so that the test of a reducible subnet is often complex. They developed a hierarchical reduction method for Petri nets. This method is based on a set of reduction rules and reduces the reducible subnets (RSN) in a Petri net into the macronodes (i.e., macroplaces and macrotransitions) without changing the important proper-

ties of liveness, boundedness, and proper termination. The conditions for applying these rules depend solely on the static structure of the net.

2.3 PETRI NET SYNTHESIS APPROACHES

Since in a moderately sized manufacturing system, the complexity of design at the implementation level of detail may be unreasonable, researchers are seeking methods for the synthesis of Petri net models. The discussion is divided into two parts [37, 38, 39], bottom-up and top-down, according to the differences in the procedures that are used for constructing final system models in the methods considered in this article. Bottom-up methods first divide the system into subsystems and specify them in detail. Then, sub-systems are merged by sharing common transitions and/or common places to obtain the system model. Top-down methods, by contrast, use an aggregate model and neglect low-level detail initially. Then, the models are refined in a stepwise manner by expanding places and/or expanding transitions to incorporate more detail into the models until the system is completely specified.

2.3.1 TOP-DOWN TECHNIQUES

A transition and place in a Petri net can be replaced by a more detailed sub-net, step by step so that an arbitrary large Petri net can be obtained. This approach is characterized by the step-wise refinement of an aggregate Petri net model. Each successive refinement contains increasing detail until the implementation level is reached. This method is very attractive from the design point of view because detail is introduced in an incremental manner thereby reducing the complexity to be dealt with at any phase in the process. From the view of such stepwise refinement of transitions and places [14, 15, 19], a number of theorems have been established about the preservation of boundedness or safeness and liveness by Valette [19] in 1979 and further gen-

eralized by Suzuki and Murata [14, 15]. Their theory can be used for both the reduction of Petri nets to analyze their properties such as liveness or boundedness and the expansion of Petri nets to build arbitrarily large Petri nets using basically the substitution of a transition by a sub-Petri net which satisfies certain properties. Also, Datta and Ghosh presented a modular approach to synthesize bounded and live Petri nets by selecting regular nets as their basic modules and their results also were used to reduce large Petri nets [23].

However, an approach for systematic synthesis of Petri net models with desirable properties for discrete event dynamic systems, especially manufacturing systems, remains undeveloped.

A Petri net model can be derived for a manufacturing system given a list of machines, a list of operations, and their relations. The procedure is outlined as follows [37]:

1. A simple Petri net is first chosen which describes the aggregate level system and satisfies the safeness, liveness, and reversibility;
2. Stepwise refinement of this Petri net is done in a manner which maintains the structural properties while adding the process and control detail;
3. Stepwise refinement is accomplished by replacing places by basic design modules that describe more detailed logic for the process represented by that place; these basic modules are defined as a sequence PN, parallel PN, conflict PN, mutual exclusion PN;
4. If the method is used, the desirable detail will be achieved and the structural properties guaranteed.

This top-down modular method is a natural decomposition of a manufacturing system.

2.3.2 BOTTOM-UP TECHNIQUES

Agerwala and Choed-Amphai [17] have done pioneer work in bottom-up techniques and have proposed a systematic bottom-up approach for synthesis of concurrent systems modeled by Petri nets. They suggest that synthesis can start with simple nets and that at each synthesis step these sub-nets can be merged in such a way that a set of places, which represent the same meaning in the model, are merged into a new place. They refer to this as a 1-way merge because each synthesis step combines only one set of places. In order to facilitate the analysis they provide a theorem so that after every 1-way merge the P-invariants of the resultant net can be known from the P-invariants of the sub-nets. This theorem is very useful because, if the invariant method is adopted for analyzing the model, it eliminates the necessity of solving large linear system equations of the final model to obtain P-invariants. Moreover, if the synthesis rule is restricted so that a merge operation is allowed only if each place in the resultant net is in some invariant, then the class of nets that can be synthesized is equivalent to the class of bounded Petri nets.

Similar work has been investigated by Narahari and Viswanadham [11] for modeling and analyzing flexible manufacturing systems. In their approach, a Petri net is first created for every basic operation such as an intermediate machine operation for a product. Because a basic operation is simple, its corresponding Petri net is easy to verify and analyze. A product might need more than one machine operation, and there might be several choices for selecting machines to execute these machine operations. In addition, several different types of products might be manufactured in the system. Therefore, a resultant Petri net can be constructed from the union (i.e. merging of places) of these sub-nets, which represent the basic operations.

Narahari and Viswanadham developed two theorems for finding P-invariants and T-invariants. One of the theorems facilitates the computation of P-invariants of the net, which results from the union of the individual sub-nets, when the P-invariants of the individual sub-nets are

known. The other theorem serves the same purpose, except for T-invariants. The theorem for P-invariants can be treated as an extension of Agerwala and Choed-Amphai's theorem, which allows places to be merged in more than one way at each synthesis step. That is, more than one set of places can be merged at one step, and the properties of the resultant net can be obtained. Krogh and Beck [1, 2, 3] invented a bottom-up technique for the synthesis of live and safe Petri nets which is different from the above two methods. Their method shares two types of simple elementary paths in which no place or transition appears more than once. The synthesis procedure starts with a collection of simple elementary circuits (SEC) corresponding to basic activity cycles in the system under the assumption that in the applications the system is designed to repeatedly perform the same function. The next step is choosing one SEC as the initial model. At each synthesis step an SEC is included in the model by sharing only one simple elementary path. The Petri net thus obtained will be live and safe with respect to any initial marking for which there is exactly one token in each of the P-invariants of the system. The result they provide, i.e. liveness and safeness are preserved, is very strong but the limitation of their method is that only safe places are allowed. This puts some difficulties in modeling some resources such as buffers. In this case, instead of using a place to denote a buffer we have to use a set of places to describe all its possible states which might be large if the buffer is bounded by a large number. Their method also places more restrictions on the modeling of shared resources than the former two methods.

Krogh and Beck's result is extended by Koh and DiCesare [16] who invented a similar method which can be applied to model bounded places and generalized Petri nets, i.e. Petri nets with multiple arcs.

2.4 GRAPH THEORETIC DEFINITIONS

1) Strong Connectivity: A Petri net is strongly connected iff there exists a directed path from

any node (place or transition) to any other node.

2) Directed Circuit: A directed circuit is a directed path from one node back to itself.

3) Directed Elementary Circuit (γ): A directed elementary circuit is a directed path from one one node back to itself such that none of the nodes are repeated.

In the example of Figure 2.4, the sequence $(p_1 - t_1 - p_2 - t_2 - p_3 - t_3 - p_1)$ constitutes a directed elementary circuit. However, the sequence $(p_1 - t_1 - p_2 - t_2 - p_4 - t_3 - p_1 - t_1 - p_2 - t_2 - p_3 - t_3 - p_1)$ is a directed circuit, which is not elementary.

2.5 CLASSES OF PETRI NETS

Two important classes of Petri nets include :

1. Marked Graph : A marked graph (also called event graph) is a Petri net where each place has exactly one input transition and exactly one output transition. It is decision-free but allow concurrency[12]. One example is shown in Figure 2.4.

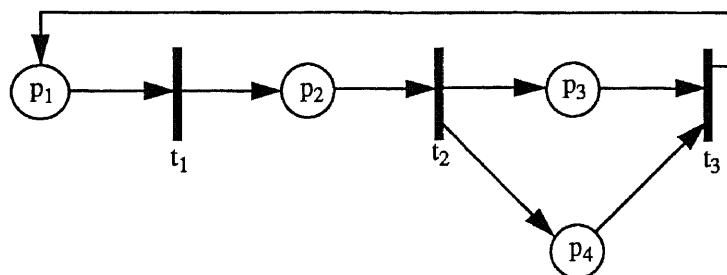


Figure 2.4 An Example of Marked Graphs

2. State Machine : A state machine is a Petri net where each transition has exactly one input place and exactly one output place. It cannot model concurrency but decision/choices[12]. One example is shown in Figure 2.5.

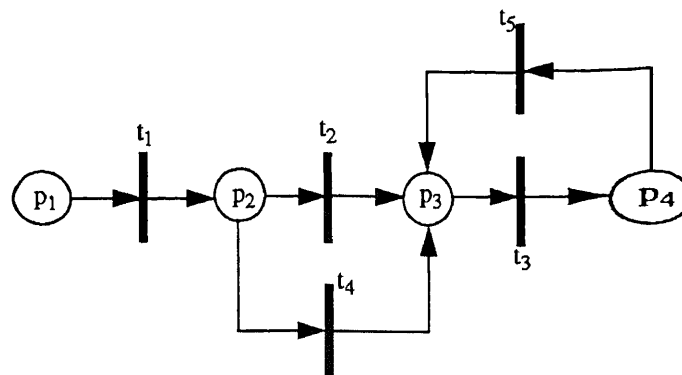
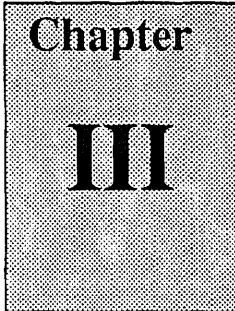


Figure 2.5 An Example of State Machines

2.6 CONCLUSIONS

In this chapter, the primitives that constitute a Petri net were discussed and their graphical representation was presented. The execution of the Petri net was discussed along with some important properties of Petri nets.

PETRI NET MODELING AND ANALYSIS OF FANUC MACHINING CENTER

A rectangular box with a halftone dot pattern. Inside the box, the word "Chapter" is written in a serif font at the top, and the Roman numeral "III" is written in a large, bold serif font in the center.

Chapter III

In many sciences, a phenomenon is studied by examining not only the actual phenomenon but also a model of the phenomenon [48]. A model is a representation, often in mathematical terms, of what are judged to be the important features of the object under study. By manipulation of the representation, it is hoped that new knowledge about the modeled phenomenon, and the model itself, will be obtained without the cost, inconvenience, or danger of manipulating the real phenomenon itself.

Petri nets, a graphical tool very well suited to the description of distributed and concurrent systems which exhibit synchronization and contention for shared resources [1, 2], have been used for modeling logic controllers, computer networks, communication protocols, operating systems, as well as production systems. This chapter discusses how Petri nets can be used to model the production systems.

3.1 MODELING WITH PETRI NETS

The Petri net description of systems concentrates on two concepts: events and conditions [4, 5]. Events are conditions that occur in the system and conditions describe the state of various parts of the system. The condition is either true or false. In order for events to occur, certain conditions, referred to as pre-conditions, must exist. After an event occurs these pre-conditions usually change and another set of conditions, referred to as post-conditions, becomes valid. The post-conditions of one event may be the pre-conditions of another and so a sequence of events may occur.

By listing the events that take place in a system along with the conditions necessary for each event to occur and the conditions that result after each event, a system can be modeled.

Consider a simple resource sharing problem, with two processes sharing a single resource as shown in Figure 3.1. The possible conditions that might exist in such a system are:

- a) Process PR_1 is waiting for resource R : p_1
- b) Process PR_2 is waiting for resource R : p_2
- c) Resource R is available : p_3
- d) Process PR_1 is running : p_4
- e) Process PR_2 is running : p_5

Based on these conditions the events that may take place are :

- a) Process PR_1 begins : t_1
 - b) Process PR_2 begins : t_2
 - c) Process PR_1 ends : t_3
 - d) Process PR_2 ends : t_4
-

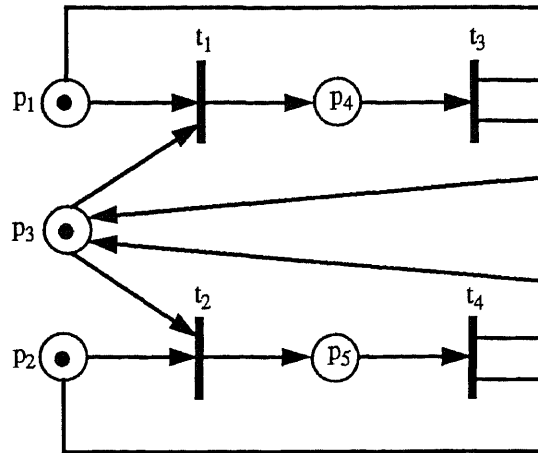


Figure 3.1 Petri Net Model of a Resource Sharing Problem

The initial marking is $m_0 = (1, 1, 1, 0, 0)^T$. The operation of the above resource sharing system and the relation between the conditions that exist and the events that occur can be described as follows. The system starts in the idle state with both the processes waiting for resource R, and resource R being available. At this point both processes can begin. However, since both processes are waiting for the same resource and only one resource is available, only one of the two processes can begin. Thus, at the very outset there is a *conflict*. Either of the two events can occur and the choice is made randomly. Let us consider that the event that occurs is 'Process PR₁ begins'. The state of the system now changes. Process PR₂ can no longer begin as resource R has been used up by process PR₁. The only event that can possibly occur with the existence of these conditions is 'Process PR₁ ends'. This brings the state of the net back to the initial state, with once again a conflict which needs to be resolved randomly. In this manner a sequence of events occur and the operation of the system goes on.

3.2 PETRI NET MODEL OF A FANUC MACHINING CENTER

Consider as an example the resource sharing problem of a Fanuc Machining Center, with two processes (milling and drilling) sharing a single resource (robot), as shown in Figure 3.2.

3.2.1 SYSTEM DESCRIPTION

The Fanuc Machining Center at NJIT consists of one robot R and two workstations; a milling machine (WS_1) and a drilling machine (WS_2). The specification of this system is as follows.

1. Before the cycle starts, the workstations, a robot and raw material are available.
2. The robot first picks up the part from the incoming conveyor, loads it on the milling machine and backs off.
3. The milling operation is performed on the raw material.
4. Once the milling operation is completed, the robot unloads the part from WS_1 and then loads it on WS_2 where it is drilled.
5. While the part is being drilled, the robot goes to the incoming conveyor, picks up another part and loads it on WS_1 .
6. Next the robot unloads the first part from the WS_2 and places it on the out-going conveyor and the cycle goes on.

In this system, a single robot is being shared by two other resources, namely, the milling machine and the drilling machine. One typical run of the Fanuc Machining Center is illustrated by Petri nets as shown in Figures 3.3.1 - 3.3.13 by firing the transitions. It is noted that a hollow box represents a transition which is enabled in each state. Initially, two raw materials are available on the incoming conveyor.

For this system, its Petri net model can be designed as shown in Figure 3.3.1. The meaning of places p_i ($1 \leq i \leq 9$) and transitions t_i ($1 \leq i \leq 6$) is indicated below.

Table 1. Places and Transitions in the Petri Net of Figure 3.3.1

p_1 : Raw material available	t_1 : Starting to load WS_1
p_2 : Robot available	t_2 : Initiation of milling operation
p_3 : WS_1 available	t_3 : Termination of milling operation
p_4 : WS_1 loaded with raw material by robot	t_4 : Initiation of drilling operation
p_5 : Milling operation performed on the raw material	t_5 : Termination of drilling operation
p_6 : Semi-finished part loaded on WS_2 by robot	t_6 : Completion of action in P_9
p_7 : Drilling operation performed on the part	
p_8 : WS_2 available	
p_9 : Out-going conveyor loaded by robot	

PETRI NET MODEL FOR FANUC MACHINING CENTER

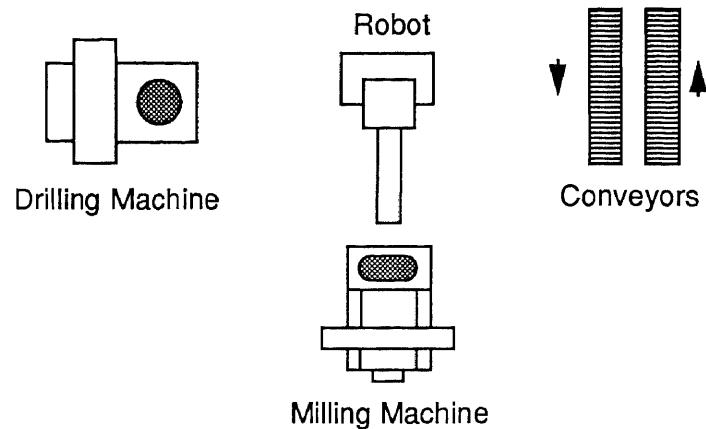


Figure 3.2 System Layout for Fanuc Machining Center

$$m_0 = (2, 1, 1, 0, 0, 0, 0, 1, 0)^T$$

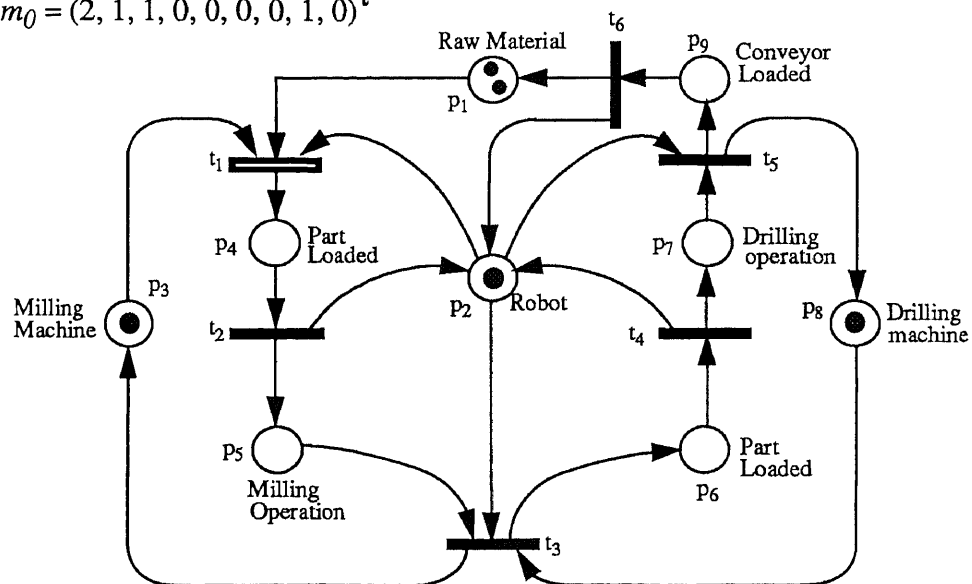
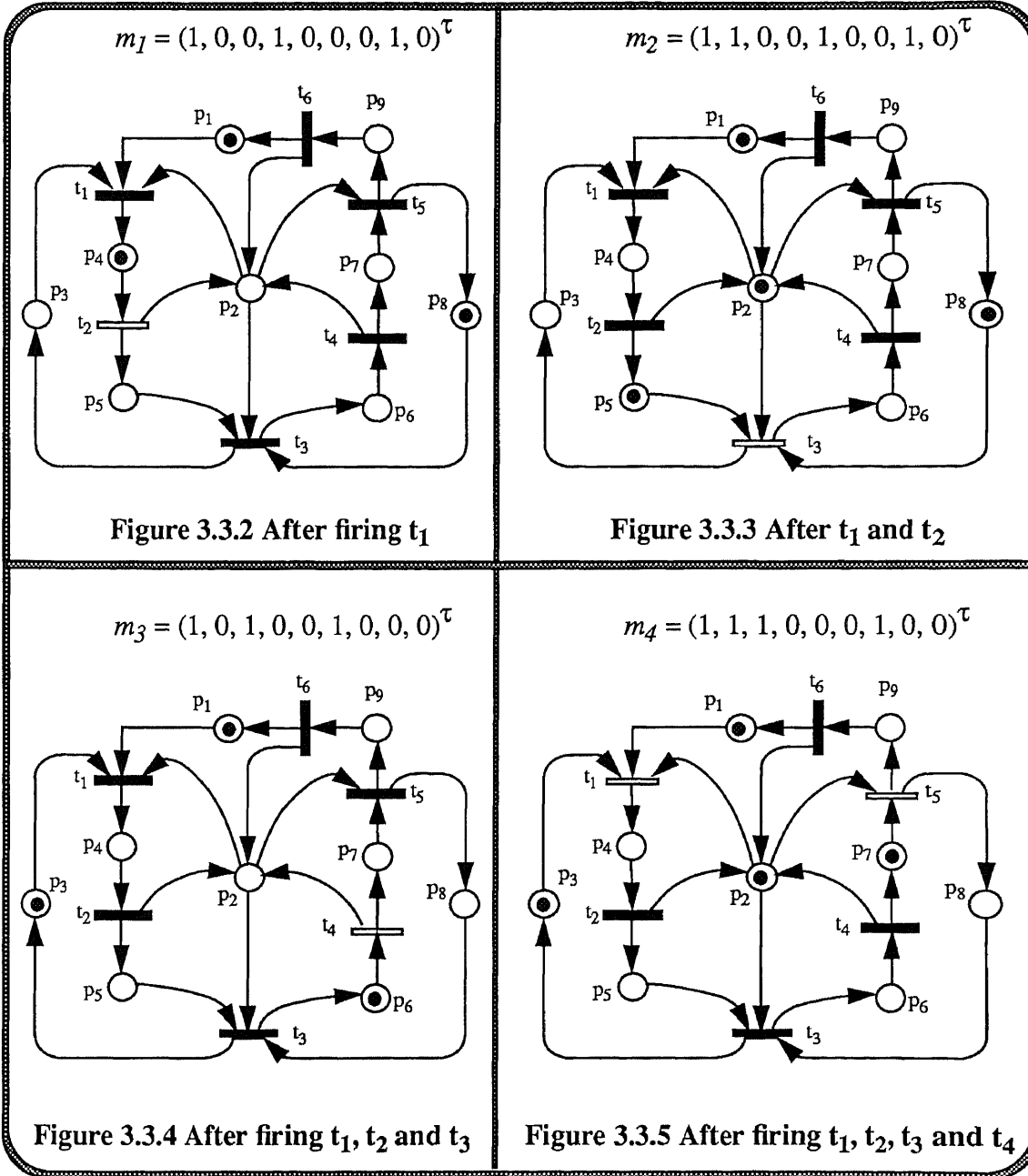


Figure 3.3.1 Petri Net Model



$$m_5 = (0, 0, 0, 1, 0, 0, 1, 0, 0)^T$$

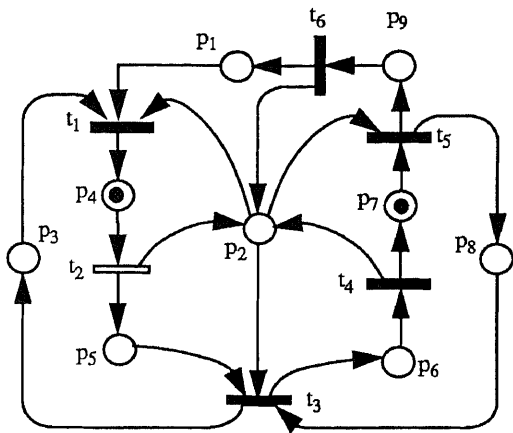


Figure 3.3.6 After firing t_1, t_2, t_3, t_4 and t_1

$$m_6 = (0, 1, 0, 0, 1, 0, 1, 0, 0)^T$$

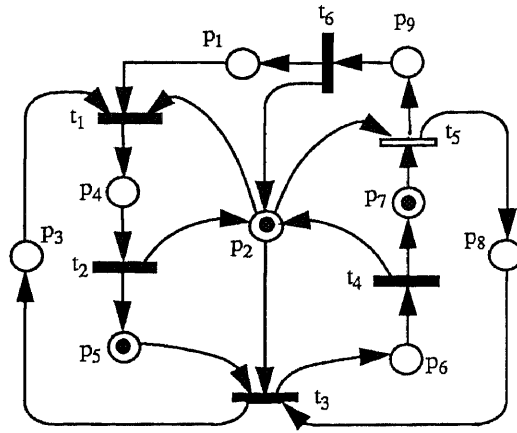


Figure 3.3.7 After firing t_1, t_2, t_3, t_4, t_1 and t_2

$$m_7 = (0, 0, 0, 0, 1, 0, 0, 1, 1)^T$$

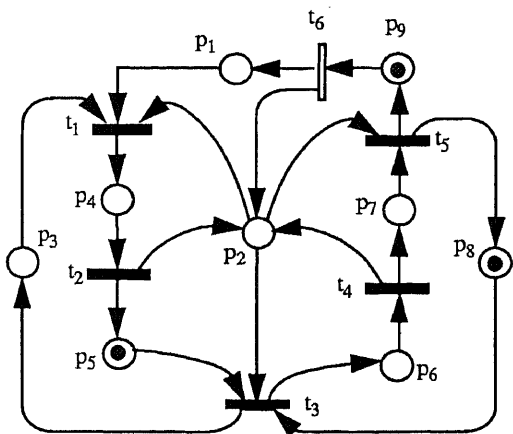


Figure 3.3.8 After firing $t_1, t_2, t_3, t_4, t_1, t_2$ and t_5

$$m_8 = (1, 1, 0, 0, 1, 0, 0, 1, 0)^T = m_2$$

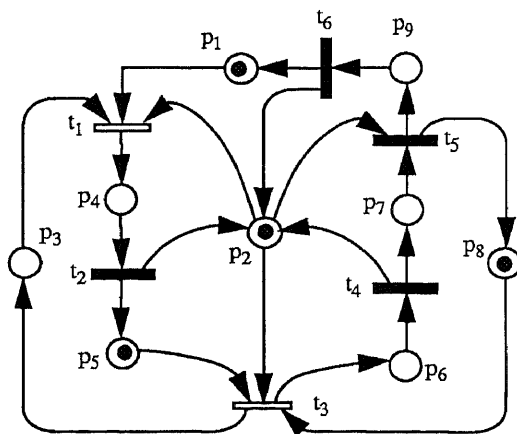


Figure 3.3.9 After firing $t_1, t_2, t_3, t_4, t_1, t_2, t_5$ and t_6

$$m_9 = (1, 0, 1, 0, 0, 1, 0, 0, 0)^T = m_3$$

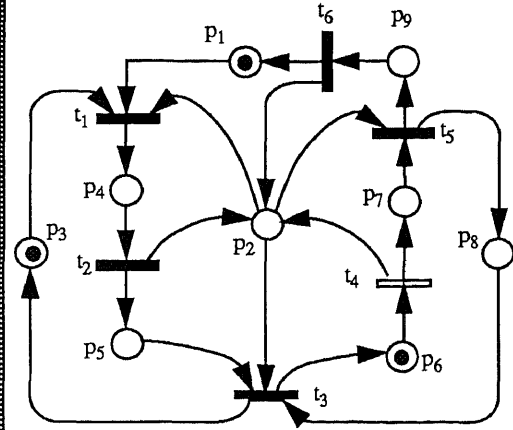


Figure 3.3.10 After firing $t_1, t_2, t_3, t_4, t_1, t_2, t_5, t_6$ and t_3

$$m_{10} = (1, 1, 1, 0, 0, 0, 1, 0, 0)^T = m_4$$

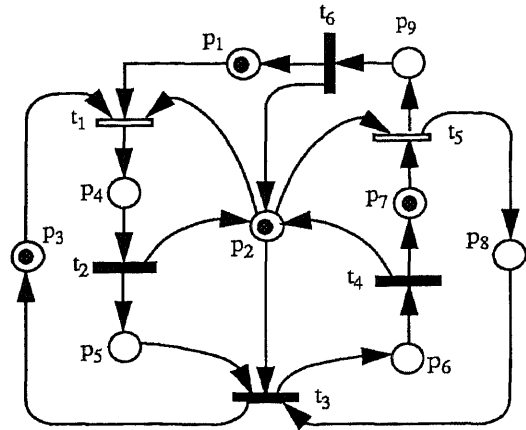


Figure 3.3.11 After firing $t_1, t_2, t_3, t_4, t_1, t_2, t_5, t_6, t_3$ and t_4

$$m_{11} = (1, 0, 1, 0, 0, 0, 0, 1, 1)^T$$

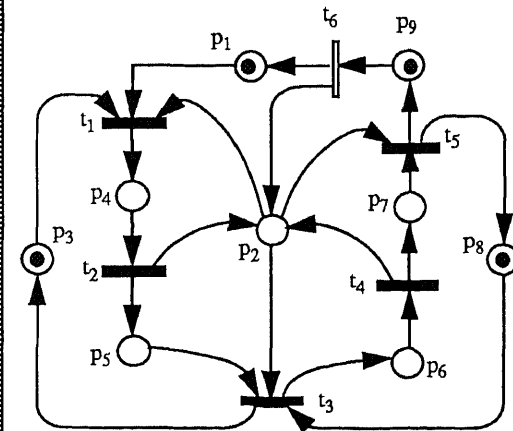


Figure 3.3.12 After firing $t_1, t_2, t_3, t_4, t_1, t_2, t_5, t_6, t_3, t_4$ and t_5

$$m_{12} = (2, 1, 1, 0, 0, 0, 0, 1, 0)^T = m_0$$

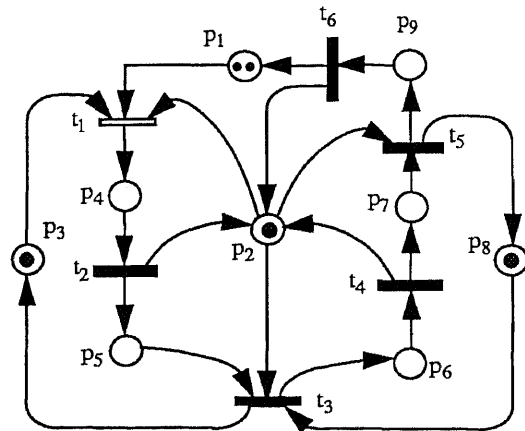


Figure 3.3.13 After firing $t_1, t_2, t_3, t_4, t_1, t_2, t_5, t_6, t_3, t_4, t_5$ and t_6

3.2.2 ANALYSIS OF PETRI NET MODEL

The initial marking is $m_0 = (2, 1, 1, 0, 0, 0, 0, 1, 0)^T$ which indicates that the milling machine, the drilling machine, the robot and two raw materials are available. A basic approach to analyzing Petri nets is to use the reachability tree. The reachability graph for the Petri net in Figure 3.3.1 is as shown in Figure 3.4.

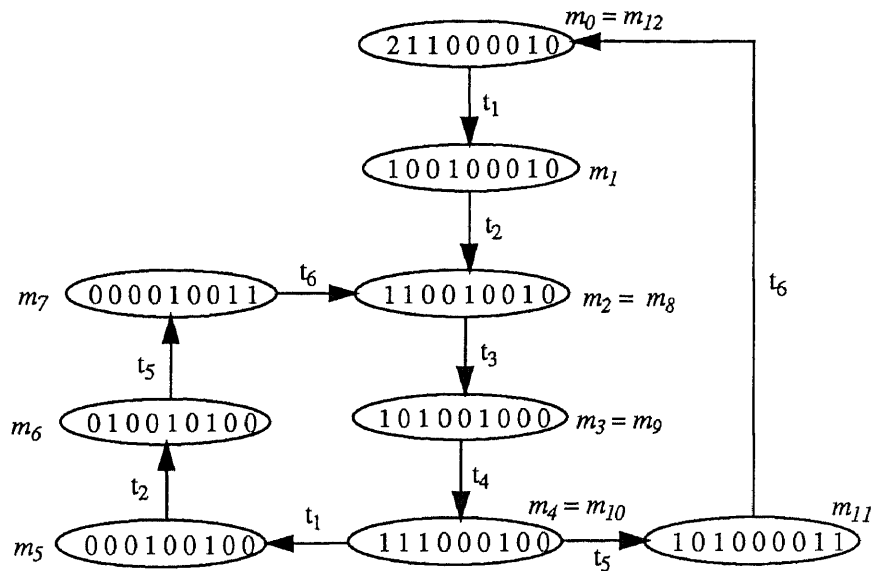


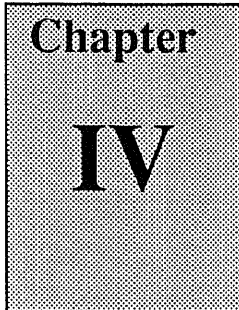
Figure 3.4 The Reachability Graph of the Petri Net Model in Figure 3.3.1

Analysis of this tree yields some useful information. It indicates exactly the set of reachable markings.

- 1) Since there are finite markings and at each marking the number of tokens in each place is either 0, 1 or 2, the net is 2-bounded.

- 2) Since the reachability graph has a directed circuit $(m_0, t_1, m_1, t_2, m_2, t_3, m_3, t_4, m_4, t_5, m_{11}, t_6, m_0)$ containing all the transitions at least once, the net is consistent.
- 3) An analysis of the tree indicates that from any marking in the tree any transition can be enabled by an appropriate firing sequence. The net is thus live.
- 4) Since every node in the reachability graph is in a directed circuit containing m_0 , the Petri net is reversible with respect to an initial marking. Reversibility implies repetitivity. Hence the net is repetitive.

PETRI NET MODELING AND ANALYSIS OF AN FMS CELL

A rectangular box with a grey stippled background. Inside the box, the word "Chapter" is written in a serif font at the top, and the Roman numeral "IV" is written in a large, bold serif font in the center.

Chapter IV

The construction analysis of Petri net model of the FMS cell at the New Jersey Institute of Technology is discussed in this chapter. In process, the refining techniques are used to develop the final Petri net model of the FMS Cell and the substructures thus obtained are checked for validness of qualitative properties.

4.1 DESCRIPTION OF A FLEXIBLE MANUFACTURING CELL

The top view of the FMS cell developed at Information Technology Center by NJIT is shown in Figure 4.1 [45]. It consists of the following major components :

1. SI Handling Conveyor System : It consists of four carts, A, B, C and D, with a fixture mounted on each, two transfer tables, TT_1 and TT_2 , and dual conveyors which transport the material to each workstation.
2. NASA II CNC Milling Machine : The milling machine accepts rectangular solid blanks and

machines “NJIT FMS” on the part.

3. The GE P50 robot is a shared resource used to load and unload the material between the CNC milling machine and the conveyor system, and between the parts presentation station and the conveyor system.
4. Parts Presentation Station : This includes a gravity-chute which supplies rectangular solid blanks as raw materials. This station also contains two bins; one each for accepted parts and rejected parts.
5. Computer Vision System : The vision system provides for the visual automated inspection of the parts.

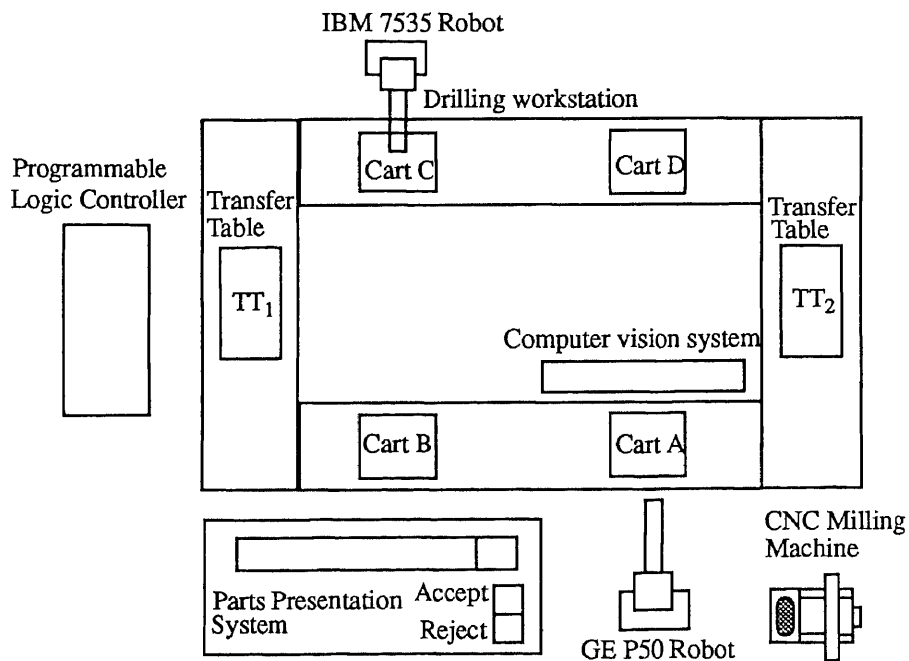


Figure 4.1 Flexible Manufacturing System Cell

6. Drilling Workstation : It includes an IBM 7535 Industrial Robot with a 1/4" drill as an end-effector and drills holes at the four corners of the part.
7. System Controller : The system is currently controlled by a Programmable Logic Controller (PLC) which coordinates all the movements of the individual components.

The working cycle for this system proceeds in the following manner :

- 1) Initially, all the four carts on the conveyor system are empty and available for the raw materials to be loaded into them from the parts presentation station.
- 2) The GE P50 robot will load four parts, one by one, into the four carts on the conveyor system. The carts move clock-wise as they are being loaded.

The steps (1) and (2) constitute Loading State for this FMS cell.

- 3) Once the four parts are loaded, the positions acquired by the four carts are as shown in Figure 4.2.
- 4) The IBM 7535 robot drills four holes (one at each corner) on each blank part as the cart stops at the drilling workstation.
- 5) The GE P50 robot goes to the conveyor, removes the blank part from the cart at position X_1 and loads it into the fixture located on the CNC machine tool table.
- 6) Once the part is loaded on the CNC milling machine, the robot backs off and the milling machine mills "NJIT FMS" on the rectangular part.

- 7) After the milling operation, the robot arm goes to the milling machine to remove the piece that was machined from the fixture.
- 8) The robot returns the finished part to the same cart on the conveyor.
- 9) A signal is sent to the vision camera to inspect the part.
- 10) The vision system outputs a signal that directs the robot to accept or reject the part.
- 11) The robot runs either an accept program to place the part on the accepted pile or runs a reject program to place on the rejected pile.
- 12) The GE robot goes to the parts presentation station and loads a new blank part into the cart.
- 13) The PLC releases this cart to the system and the next cycle is started.

The steps (3) to (13) constitute Working State for this FMS cell.

4.2 PETRI NET DESIGN PROCESS

This section applies the system decomposition, refinement, and modular composition to the modeling process of Petri nets for the described FMS cell. The system has two stages: initialization or loading state and working state. Also the system can be naturally decomposed into the following parts: 1. Conveyor System, 2. NASA II CNC Milling Machine, GE Robot, Computer Vision System, and Parts Presentation Station, and 3. Drilling Workstation.

The initial positions for the carts with raw materials in the loading state are shown in Figure 4.2.

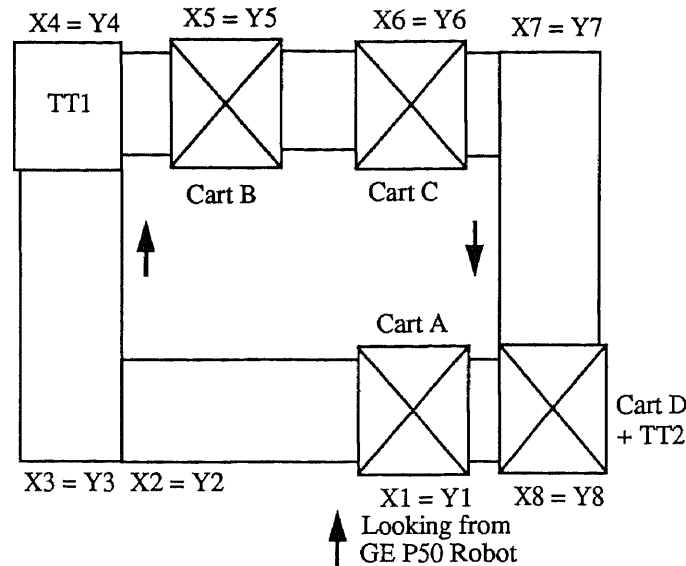


Figure 4.2 Loading State of the Conveyor System

4.2.1 MODELING OF CONVEYOR SYSTEM

A. Modeling

It is observed that all four carts will move from their original position back to the exactly same one during a complete cycle. Hence, the conveyor system is repetitive. Since every cart can move toward only one direction and transfer tables carry a cart from one to the other and go back once it is done, the system is decision-free. The basic operations with carts and transfer tables are *moving* and *waiting*. Positions are important and can be modeled as single resources since every position cannot be occupied by two carts at any time. Four carts will be modeled then as four tokens since they are moving from a position to another.

Based on the above analysis, we represent each *moving* between two positions as a transition, and each *waiting* as a place and picture transitions t_1 - t_{10} and places, px_1 - 8 . Then the availability of the positions x_1 - x_8 is modeled as places py_1 - 8 . For example, t_1 can fire only when py_2 is marked and px_1 is marked, or when a cart is occupied at position x_1 and position x_2 is vacant. Correspondingly, we picture py_1 , py_5 and py_6 . Place py_3 is designed and represents the availability of transfer table TT_1 at position x_3 . Places py_4 , py_7 and py_8 can be modeled similarly. Since firing t_1 requires that a cart occupies position x_1 and position x_2 is available, arcs (px_1, t_1) and (py_2, t_1) have to be added. Also, arc (t_1, py_1) is added since firing t_1 means that position x_1 becomes available, or py_1 should be marked. Based on the same reason, we add all arcs as shown in Figure 4.3.

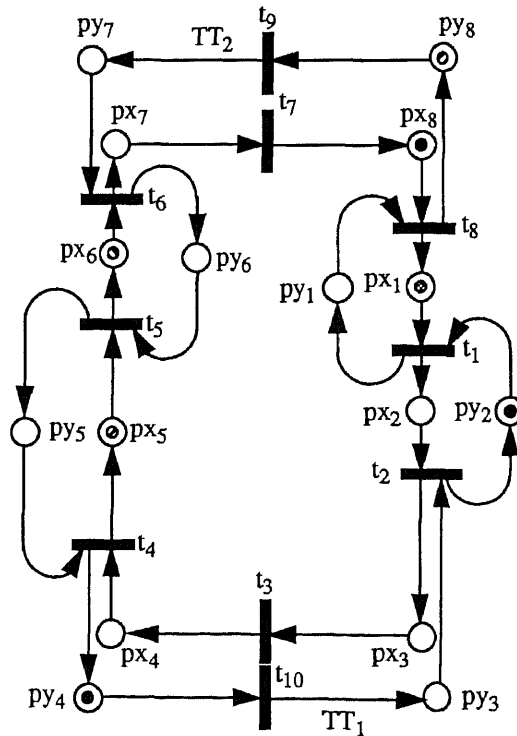


Figure 4.3 Modeling of Material Handling System

The meaning of places Px_i and Py_i ($1 \leq i \leq 8$) and transitions t_i ($1 \leq i \leq 10$) is indicated in Table 2.

Table 2. Places and Transitions in the Petri Net of Figure 4.3

px_1 : Cart available at the loading station at X_1	py_1 : Availability of cart halting position at X_1
px_2 : Cart halting position at X_2	py_2 : Availability of cart halting position at X_2
px_3 : Cart loaded on the transfer table TT_1 at X_3	py_3 : Availability of TT_1 at position X_3
px_4 : Cart transported to position X_4 by TT_1	py_4 : Availability of TT_1 at position X_4
px_5 : Cart available at drilling workstation at X_5	py_5 : Availability of cart halting position at X_5
px_6 : Cart halting position at X_6	py_6 : Availability of cart halting position at X_6
px_7 : Cart loaded on the transfer table TT_2 at X_7	py_7 : Availability of TT_2 at position X_7
px_8 : Cart transported to position X_8 by TT_2	py_8 : Availability of TT_2 at position X_8
t_1 : Cart moving from loading station to X_2	t_6 : Cart being loaded on transfer table TT_2
t_2 : Cart being loaded on transfer table TT_1	t_7 : TT_2 and the loaded cart move together to X_8
t_3 : TT_1 and the loaded cart move together to X_4	t_8 : Cart unloads from TT_2 and moves to X_1
t_4 : Cart unloads from TT_1 and moves to drilling station	t_9 : TT_2 moves back from position X_8 to X_7
t_5 : Cart moves from drilling station to X_6	t_{10} : TT_1 moves back from position X_4 to X_3

The initial marking can be designed as $m_0 = (1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1)^T$. It is noted that place px_1 and px_5 are macro-places which represent more detailed activities discussed in Sections 4.2.2 and 4.2.3.

B. Analysis

The Petri net in Figure 4.3 is a marked graph, i.e., each place has exactly one input and one output transition. This net can be analyzed for validness of qualitative properties by making

use of the following facts, established by Commoner and Holt [25], which describe useful properties of event-graphs:.

Fact 1: The number of tokens in an elementary circuit is invariant by any transition firings.

Fact 2: An event graph is live iff there exists at least one token in every elementary circuit.

Also, an important result obtained by Sifakis [8] is stated below:

Fact 3: A Petri net which is structurally live and structurally bounded is necessarily strongly connected.

Based on the results about marked graphs, Hillion [26] has deduced that:

Fact 4: An marked graph is bounded iff it is strongly connected.

The marked graph of Figure 4.3 contains the seven elementary circuits:

$$\gamma_1 = (px_1, t_1, px_2, t_2, px_3, t_3, px_4, t_4, px_5, t_5, px_6, t_6, px_7, t_7, px_8, t_8, px_1)$$

$$\gamma_2 = (px_1, t_1, py_1, t_8, px_1)$$

$$\gamma_3 = (px_2, t_2, py_2, t_1, px_2)$$

$$\gamma_4 = (px_5, t_5, py_5, t_4, px_5)$$

$$\gamma_5 = (px_6, t_6, py_6, t_5, px_6)$$

$$\gamma_6 = (px_3, t_3, px_4, t_4, py_4, t_{10}, py_3, t_2, px_3)$$

$$\gamma_7 = (px_7, t_7, px_8, t_8, py_8, t_9, py_7, t_6, px_7)$$

$$\gamma_8 = (py_1, t_8, py_8, t_9, py_7, t_6, py_6, t_5, py_5, t_4, py_4, t_{10}, py_3, t_2, py_2, t_1, py_1)$$

The initial marking for this event-graph is

$$\begin{array}{llll}
m_0(px_1) = 1 & m_0(py_1) = 0 & m_0(px_2) = 0 & m_0(py_2) = 1 \\
m_0(px_3) = 0 & m_0(py_3) = 0 & m_0(px_4) = 0 & m_0(py_4) = 1 \\
m_0(px_5) = 1 & m_0(py_5) = 0 & m_0(px_6) = 1 & m_0(py_6) = 0 \\
m_0(px_7) = 0 & m_0(py_7) = 0 & m_0(px_8) = 1 & m_0(py_8) = 1
\end{array}$$

Fact 1 means that, for any reachable marking M , the total number of tokens in circuits $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6$ and γ_7 , respectively, given by

$$m(\gamma_1) = m(px_1) + m(px_2) + m(px_3) + m(px_4) + m(px_5) + m(px_6) + m(px_7) + m(px_8)$$

$$m(\gamma_2) = m(px_1) + m(py_1) \qquad m(\gamma_3) = m(px_2) + m(py_2)$$

$$m(\gamma_4) = m(px_5) + m(py_5) \qquad m(\gamma_5) = m(px_6) + m(py_6)$$

$$m(\gamma_6) = m(px_3) + m(px_4) + m(py_3) + m(py_4)$$

$$m(\gamma_7) = m(px_7) + m(px_8) + m(py_7) + m(py_8)$$

$$m(\gamma_8) = m(py_1) + m(py_2) + m(py_3) + m(py_4) + m(py_5) + m(py_6) + m(py_7) + m(py_8)$$

is invariant and therefore equal to the initial marking; i.e.,

$$m(\gamma_1) = m_0(\gamma_1) = 4 \qquad m(\gamma_2) = m_0(\gamma_2) = 1 \qquad m(\gamma_3) = m_0(\gamma_3) = 1$$

$$m(\gamma_4) = m_0(\gamma_4) = 1 \qquad m(\gamma_5) = m_0(\gamma_5) = 1 \qquad m(\gamma_6) = m_0(\gamma_6) = 1$$

$$m(\gamma_7) = m_0(\gamma_7) = 2 \qquad m(\gamma_8) = m_0(\gamma_8) = 3$$

Since there is at least one token in each of the seven circuits, Fact 2 guarantees that the net is live. Moreover, the event-graph is strongly connected (which means that there exists a directed path from any node to any other node). Therefore, as per Fact 4, the net is bounded.

4.2.2 MODELING OF MILLING MACHINE, GE ROBOT, COMPUTER VISION SYSTEM, AND PARTS PRESENTATION STATION

A. Modeling

We should model all these components together because they are tied with the GE P50 robot. For this set of components, we can list the activities as *{loading, milling, unloading, inspecting, moving to accepted area or rejected area, loading a new raw-material to the cart}*. The places used to represent these activities are pictured as p_2 - p_8 in sequel. Before loading, a place, i.e., p_1 , representing the availability of a raw material in a cart needs to be added at the top. After the robot loads a new raw material to the cart (p_8), a place p_9 needs to be added to represent the availability of the cart with a new part to be released. Two places, i.e., p_6 , moving to the accepted area, and p_7 , moving to the rejected area are pictured in parallel. Insert the transitions t_{11} - t_{19} between the places as shown in Figure 4.4(a). They represent the start or termination.

The places to model the availability of robot, milling machine, and vision system are drawn in parallel with the related places. Robot, p_{10} , should be available to start loading in p_4 and is released after the loading. Thus two arcs (p_{10}, t_{11}) and (t_{12}, p_{10}) are added. Similarly, transitions t_{13} , t_{14} , t_{15} , t_{16} , and t_{19} are connected to p_{10} . For the milling machine or place p_{11} , the relationship is shown through the arcs among p_{11} , t_{12} and t_{13} . This is also done for vision system, p_{12} and raw material in the parts presentation station p_{13} . Since the part inspection result is either accepted or rejected. This means there is a conflict in this net structure. Thus p_5 has two arcs to t_{15} and t_{16} , respectively. Nevertheless, the next operation is that robot picks up a

Figure 1 consists of two parts: (a) a Petri net and (b) a state transition graph.

(a) The Petri net has 14 places ($p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}$) and 19 transitions ($t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}$). Places $p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$ are arranged in a vertical column. Places $p_{10}, p_{11}, p_{12}, p_{13}$ are arranged in a horizontal row. Transitions $t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9$ are arranged in a vertical column between the two rows of places. Transitions $t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}$ are arranged in a horizontal row between the two rows of places. Places $p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$ are initially empty. Places $p_{10}, p_{11}, p_{12}, p_{13}$ each contain one token. Transitions $t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9$ are initially disabled. Transitions $t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}$ are initially enabled.

(b) The state transition graph shows the sequence of states $m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9$ reached by the sequence of transitions $t', t'', t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}$. The states are represented by ellipses containing a binary string of 14 bits. The sequence of states is: m_0 (10000000001111), m_1 (01000000001111), m_2 (00100000000111), m_3 (00010000001011), m_4 (00001000000111), m_5 (00000100001101), m_6 (00000010000111), m_7 (00000001000111), m_8 (00000000100110), and m_9 (00000000011111). The sequence of transitions is: t' , t'' , t_1 , t_2 , t_3 , t_4 , t_5 , t_6 , t_7 , t_8 , t_9 , t_{10} , t_{11} , t_{12} , t_{13} , t_{14} , t_{15} , t_{16} , t_{17} , t_{18} , t_{19} .

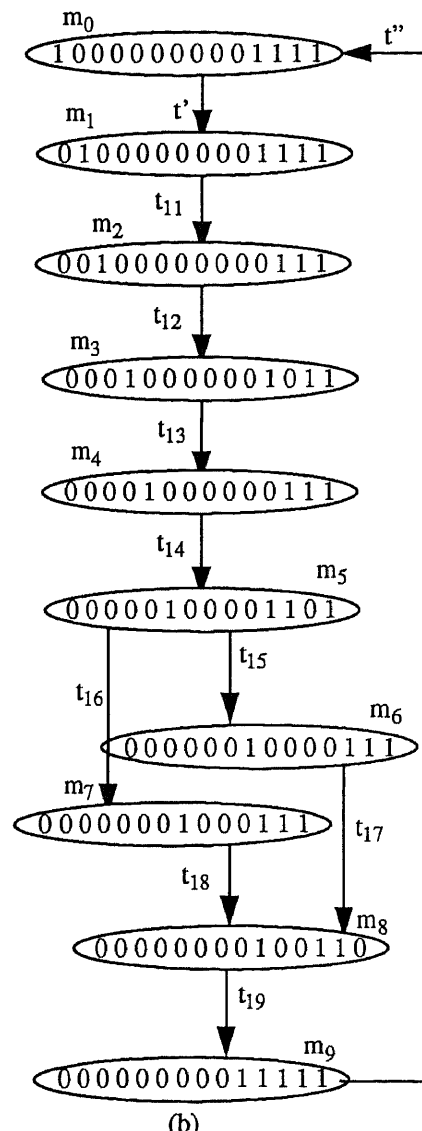


Figure 4.4 (a) Modeling of the Milling Machine, GE Robot, Vision System, and Parts Presentation Station; (b) Reachability Graph

The initial marking is $(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1)^T$. It means that initially the robot, milling machine, vision system, and raw material in the parts presentation station are available and no operation is ongoing. The meaning of places p_i ($1 \leq i \leq 13$) and transitions t_i ($11 \leq i \leq 19$) is indicated below Table 3.

Table 3. Places and Transitions in the Petri Net of Figure 4.4(a)

p_1 : Raw material to be processed with the cart	t_{11} : Starting to load the milling machine
p_2 : Milling machine loaded with raw material by GE P50 robot	t_{12} : Initiation of milling operation
p_3 : Milling operation performed on the raw material	t_{13} : Termination of milling operation
p_4 : Finished part returned to the cart	t_{14} : Completion of action in P_4
p_5 : Part inspected by vision camera	t_{15} : Initiation of action in P_6
p_6 : The robot places the accepted part in a bin	t_{16} : Initiation of action in P_7
p_7 : The robot places the rejected part in another bin	t_{17} : Completion of action in P_6
p_8 : The robot loads the cart with a new blank part	t_{18} : Completion of action in P_7
p_9 : Cart available to be released to the conveyor system	t_{19} : Completion of action in P_8
p_{10} : GE P50 Robot available	
p_{11} : Milling machine available	
p_{12} : Vision system available for inspection purpose	
p_{13} : Raw material available from the parts presentation station	

B. Analysis

In this section, the Petri net model for working state FMS cell is checked for qualitative properties by analyzing the detailed sub-net (replacement of px_1). To achieve this objective, an idle place initially marked with one token, two transitions t and t' are associated with place p_1 and p_9 . The reachability graph is constructed as shown in Figure 4.4(b). It can be noticed that :

- Since each marking has its element being either 0 or 1, the net is safe.

- Since the reachability graph has a directed circuit containing all the transitions at least once, the sub-net is consistent.
- From any marking in the reachability graph any transition can be enabled by an appropriate firing sequence. The sub-net is thus live.
- Since every node in the reachability graph is in a directed circuit containing node m_0 , the sub-net is reversible with respect to an initial marking.

Thus the substructure in Figure 4.4(a) is live, safe and reversible.

4.2.3 MODELING OF DRILLING STATION

There is only one operation: drilling. Three more places are needed to represent the availability of a raw material with a cart, a finished part with the cart, and the drilling station. Thus the structure is easily constructed as shown in Figure 4.5. The places and transitions are explained in Table 4. The initial marking for this substructure is $m_0 = (0, 0, 0, 1)^T$. Note that the drilling operation may be further decomposed as four drilling operations for four holes. This structure can be proved to be live, safe, and reversible using the previous technique.

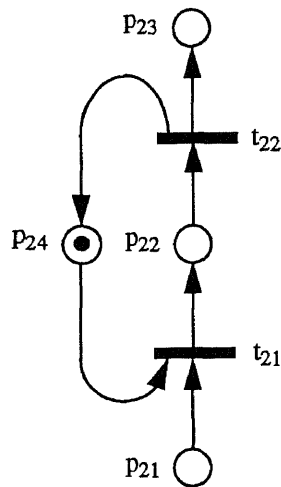


Figure 4.5 Modeling of Drilling Workstation

Table 4. Places and Transitions in the Petri Net of Figure 4.5

p_{21} : Raw material available with a cart	t_{21} : Initiation of drilling operation
p_{22} : Drilling operation performed on the raw material	t_{22} : Completion of drilling operation
p_{23} : Finished part available with a cart	
p_{24} : Drilling machine available	

4.2.4 FINAL PETRI NET MODELS

A. Loading State

A GE P50 Robot is used to load raw materials from the parts presentation station to the conveyor system. The operation involved is *loading*. In addition, raw material available and the cart with the raw material available are two more places in the submodel. Thus, a structure as same as that in Figure 4.5 can be shown in Figure 4.6 with explanation in Table 5. The refinement of place px_1 of Figure 4.3 can be done with the structure in Figure 4.6 which gives the final net for the loading state as shown in Figure 4.7.

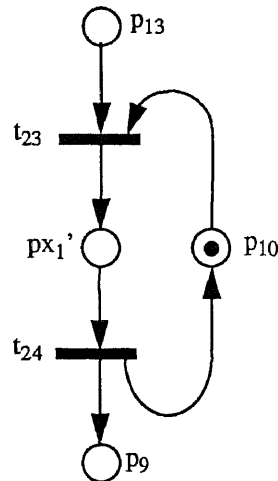
**Figure 4.6 Modeling of Loading Station**

Table 5. Places and Transitions in the Petri Net of Figure 4.6

p_{13} : Raw material available	t_{23} : Initiation of loading operation
px_1' : Cart being loaded with raw material by GE Robot	t_{24} : Completion of loading operation
p_9 : Raw material available with a cart	
p_{10} : GE P50 Robot available	

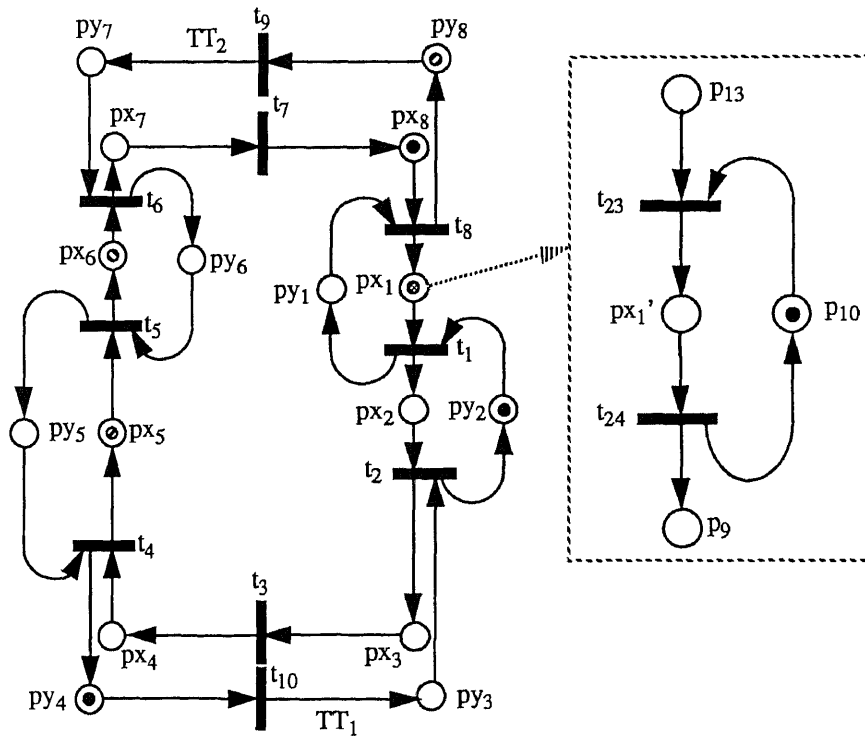


Figure 4.7 The Final Petri Net Model for the Loading State

Since the final net for the loading state is still a marked graph model, thus the net can be easily proved to be live, safe, and reversible.

B. Working State

The final Petri net model is obtained by refining the place px_1 and px_5 in Figure 4.3 by the structures shown in Figure 4.4(a) and Figure 4.5, respectively. Note that both detailed structures in Figure 4.4(a) and Figure 4.5 are live, safe, reversible, as already proved in sections 4.2.1 and 4.2.2. These two refinements will guarantee the resulting net to be live, safe, and reversible [19, 37]. Thus we have shown the procedure to get this Petri net for an FMS cell. More importantly, we have shown that the final net has the desired system properties such as liveness, safeness, and reversibility.

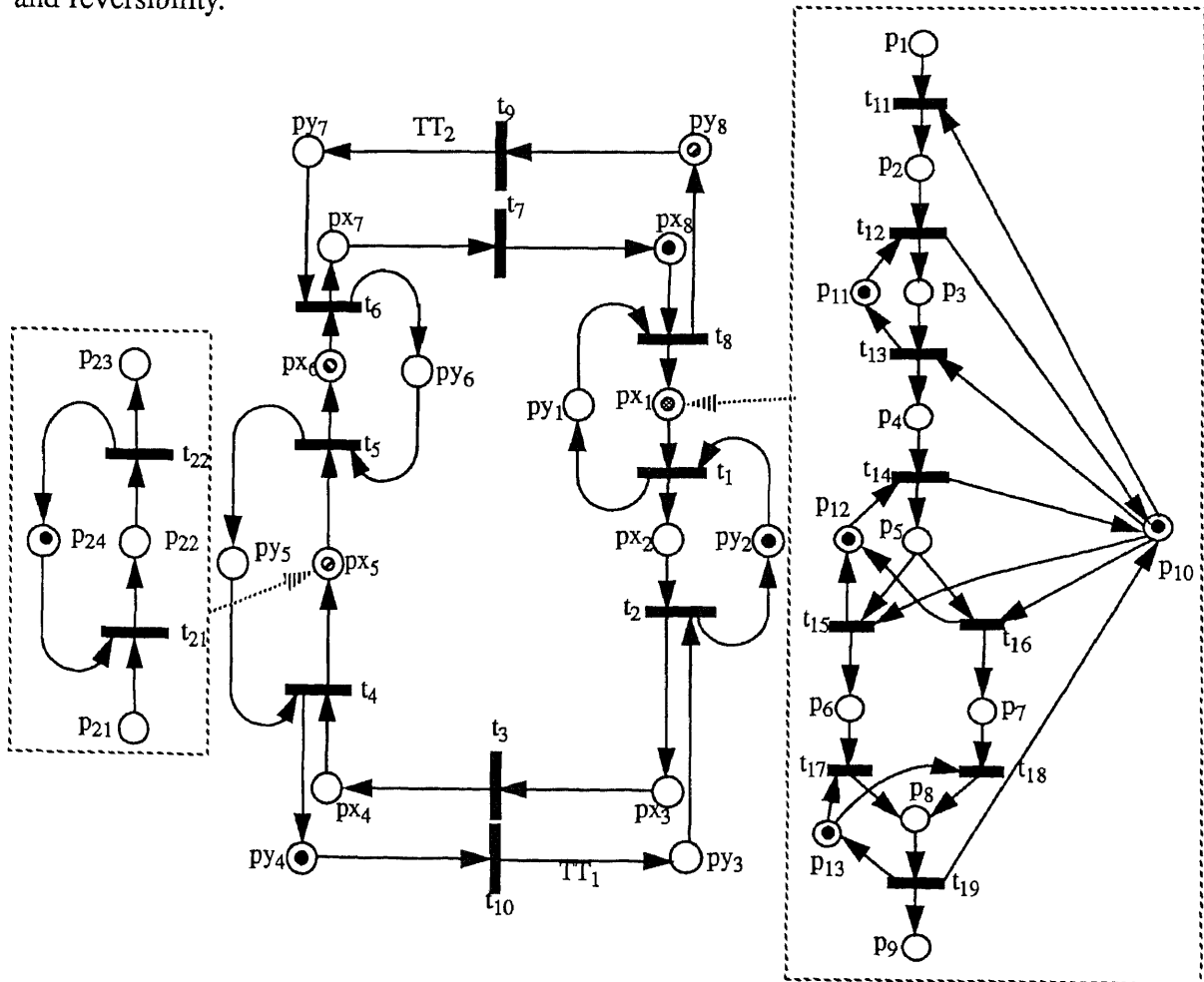
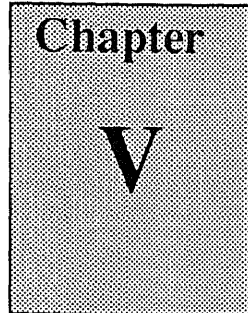


Figure 4.8 The Final Petri Net Model for the Working State

TEMPORAL PETRI NETS FOR PERFORMANCE ANALYSIS



5.1 BRIEF LITERATURE REVIEW

So far Petri nets are used to model complex discrete event dynamic systems. These models represent a logical point of view of the system. In the following discussion, we focus on the application of Petri nets for performance analysis of manufacturing systems.

In order to conduct temporal performance analysis, i.e., to determine production rate, resource utilization, and the like, a system needs to be modeled as a Petri net with timing. It is also possible to detect a bottleneck in an FMS or to determine optimal buffer size, optimal pallet distribution, etc. as indicated by Dubois and Stecke [10]. In fact, Dubois and Stecke were the first to apply timed Petri nets to describe, model, and analyze production processes. In their research, deterministic (i.e., fixed) time variables are assumed, and a Petri net-based simulation method is utilized to find maximum cycle time and to identify the bottleneck machine for an FMS with three machines and three part-types in a fixed route.

To evaluate the performance of job-shop systems under deterministic and repetitive functioning of a production process, Hillion and Proth [26] apply a special class of timed Petri nets called timed event-graphs for an FMS with three machines and three job types. The number of jobs in-process is nearly minimized using integer programming, while the system still works at its maximal productivity.

Deterministic (timed) Petri nets are not the only Petri nets with time. Alternative models have been developed, called stochastic Petri nets [29]. Those models are closer to queueing networks because they assume an exponentially distributed firing time. Viswanadham and Nara-hari [11] have provided an excellent introduction to the use of generalised stochastic Petri nets in analyzing the system performance of automated manufacturing systems. They have used a software package they developed to evaluate two representative systems: a manufacturing cell with multiple material handling robots, and an FMS with three machines and two part-types.

Ramchandani developed timed Petri nets (TPN) which enabled him to incorporate time constraints in the Petri net model [5]. This allowed a performance analysis to be carried out on these systems by calculating process cycles, resource utilization, average number of processed parts and throughput rate. Merlin developed timed Petri net which is more general temporal model than Ramchandani's timed PN, since the fixed duration d of an event can be simply modeled as $[d, d]$.

Time can be included in a Petri net model by associating time with the transitions, to form a timed transition Petri net (TTPN), or with the places, resulting in a timed place Petri net (TPPN). Both representations are equivalent [6]. In a TTPN, the firing of a transition takes a certain amount of time. Note that this time is fixed, which makes TTPNs deterministic. In a TPPN, a token enters a place and is unavailable for some time delay after which it becomes available. In these nets, only available tokens in a marking can enable a transition. A TPPN or TTPN with all delays set to zero reduces to an ordinary Petri net.

5.2 COVERABILITY IN TIME

Given a Petri net that is k -bounded for $k > 1$, there may be markings in the reachability set that differ only in the number of tokens. Given two markings $m_1, m_2 \in R(m)$, we say that m_1 cover m_2 if, for every place, the token count in $m_1 \geq$ the token count in m_2 . Thus, coverability [27, 31] implies that m_1 is a better characterization of extreme behavior than m_2 . Instead of having places that can be occupied by multiple tokens, we can have transitions that may fire at any time within a bounded time interval, i.e., once transition t is enabled, it fires after a time delay between τ^{\min} and τ^{\max} (unless it is disabled by the firing of some other transition).

The notion of covering suggests that the “extreme” firing behavior of a timed PN might be more simply characterized than by markings and classes of markings. These extreme firing behavior of the PN can be described by the two “periods” π^E and π^L representing the earliest and the latest time instant, respectively, at which transition t fires. For systems that function in a repetitive way, a characterization has been developed [27, 31] in terms of the minimum and maximum periods of repetition (cycle time).

For a PN in which the transitions fire sequentially, π^E and π^L can be computed directly from the firing intervals of the transitions as shown below:

$$\pi^E = \tau_1^{\min} + \tau_2^{\min} \qquad \pi^L = \tau_1^{\max} + \tau_2^{\max}$$

where t_1 of time delay $[\tau_1^{\min}, \tau_1^{\max}]$ and t_2 of time delay $[\tau_2^{\min}, \tau_2^{\max}]$ can fire sequentially.

5.3 ANALYSIS OF AN FMS CELL USING DETERMINISTIC PETRI NETS

In this section, the final Petri net model of an FMS cell is analyzed using Ramchandani's and Merlin's method of performance analysis of timed Petri nets

For timed marked graphs, there exists already the formula to find the system cycle time [6, 12, 27]. For a marked graph which has time delays in its transition and places, the maximum cycle time C is given by

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

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 ,
 n = Number of circuits in the marked graph.

By observing the operations of this FMS cell, we have measured the timing delays for all operations in the FMS cell, which is summarized in Table 6. The first column contains single value durations corresponding to Ramchandani's timed PN, while the second column contains minimum and maximum bounds on the durations corresponding to Merlin's timed PN. This final net is not a marked graph. Thus before the above formula for calculating cycle time for timed marked graphs is used, the net in Figure 4.6 needs to be converted to a marked graph.

There are two places, i.e., p_{10} and p_5 which has more than one input transitions and/or output transitions. For p_5 , designers can find that both followed places p_6 and p_7 has the exact time delays, thus there is no difference whether the actual operation is executed in p_6 or p_7 . This also holds for zero-time-delay transitions t_{15} - t_{18} . Thus an equivalent structure, i.e., two transitions

and a place or t_{56} , p_{67} , and t_{78} can be used to replace the original four transitions and two places. Time delay $\tau(p_{67}) = \tau(p_6) = \tau(p_7) = 3$ seconds and $\tau(t_{56}) = \tau(t_{78}) = 0$.

For place p_{10} , since all related operations in p_2 - p_8 follow a strict order as shown in Figure 4.4(b). Thus we can simply assume that the robot is available over the whole period from the start of loading (t_{11}) to the end of loading a new material to the cart. In other words, the arcs between p_{10} and t_{12} - t_{16} can be eliminated without changing the role of the robot, the sequence of the operation, and related timing information. Now the net is a marked graph, as shown in Figure 5.1, since there is no place which has multiple input or multiple output transitions.

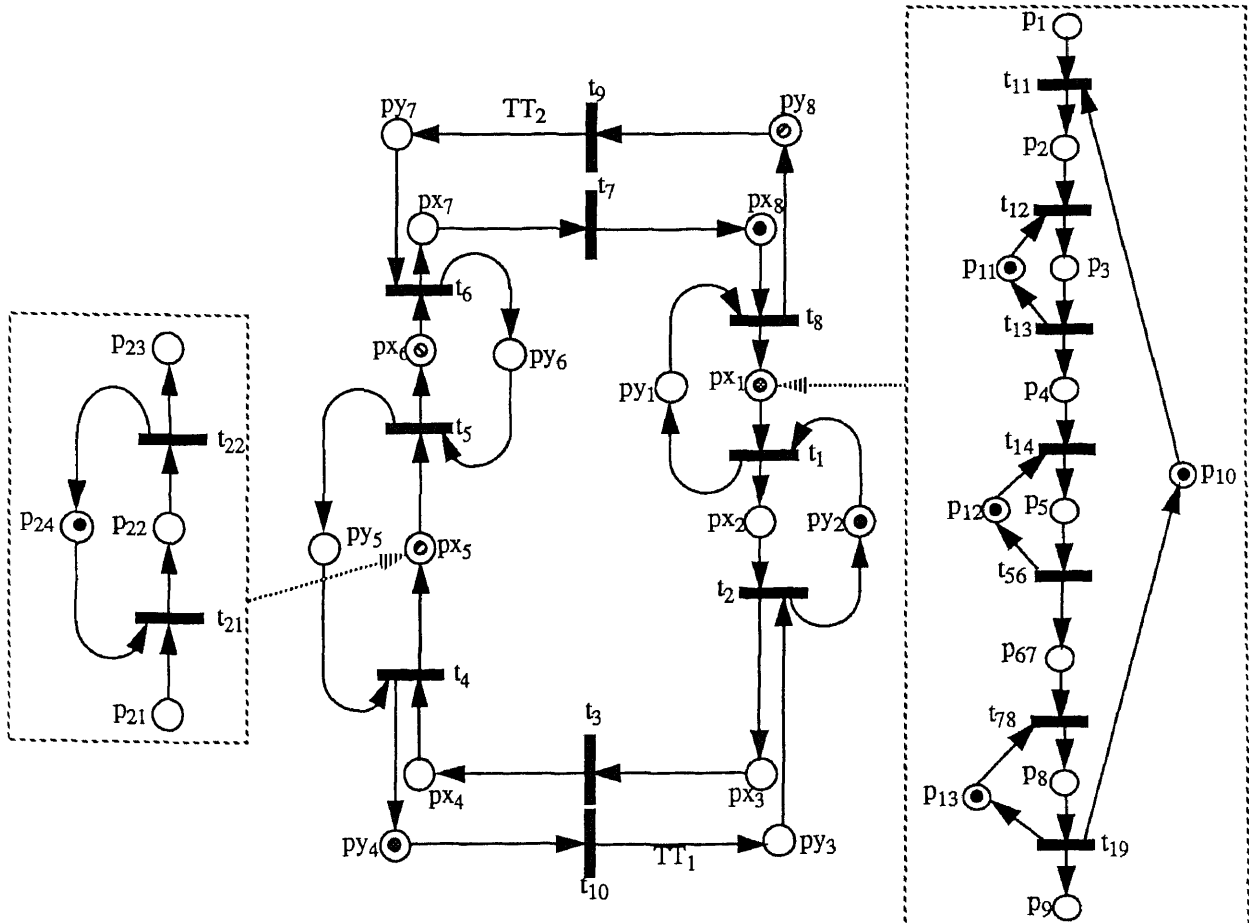


Figure 5.1 The Marked Graph Equivalent of the Final Petri Net Model

Furthermore, the maximum time delay for a token to go through from p_1 to p_9 is $\tau(p_1) + \tau(p_2) + \tau(p_3) + \tau(p_4) + \tau(p_5) + \tau(p_{67}) + \tau(p_8) + \tau(p_9)$ since all transitions have zero time delay as shown in Table 5. Thus, the substructure in Figure 4.4(a) is equivalent to place px_1 with $\tau(px_1) = 74$ (sec). Similarly, the substructure in Figure 4.5 is equivalent to px_5 with $\tau(px_5) = 73$ (sec). Therefore, we can use px_1 and px_5 instead of the two substructures to enumerate the circuits for the final net as shown in Table 7. The cycle times are evaluated. The maximum cycle time is 83 seconds and the bottleneck circuit is γ_1 . Table 8 lists the elementary circuits for the marked graph equivalent of the final Petri net model after substituting the two substructures for px_1 and px_5 .

Table 6. Time Delays for Places and Transition Firings

Place	Delay time (secs)		Transition	Firing time (secs)	
$P_9, P_{13}, P_{Y_i}, P_{23}, P_{24}$	0	(0, 0)	t_1	3	(2, 3)
P_2	5	(5, 6)	t_2	3	(3, 4)
P_3	45	(45, 46)	t_3	3	(3, 4)
P_4	5	(5, 6)	t_4	4	(3, 4)
P_5	10	(9, 10)	t_5	3	(2, 3)
P_6, P_7, P_{67}	3	(3, 4)	t_6	3	(3, 4)
P_8	5	(5, 6)	t_7	3	(2, 3)
P_1, P_{21}	1	(0, 0)	t_8	5	(4, 5)
px_1	74	(72, 78)	t_9	3	(2, 3)
px_2	5	(5, 6)	t_{10}	3	(3, 4)
px_3	2	(2, 3)	t_{11}, t_{12}	0	(0, 0)
px_4	3	(2, 3)	t_{13}, t_{14}	0	(0, 0)
px_5	73	(72, 73)	t_{15}, t_{16}, t_{56}	0	(0, 0)
px_6	75	(74, 75)	t_{17}, t_{18}, t_{78}	0	(0, 0)
px_7	2	(2, 3)	t_{19}	0	(0, 0)
px_8	68	(68, 69)	t_{21}, t_{22}	0	(0, 0)
P_{22}	72	(72, 73)			

Table 7. Elementary Circuits and their Cycle Times

Elementary Circuits (γ_i)	T_i (sec)	N_i	$C_i = T_i / N_i$
$\gamma_1 = (px_1, t_1, px_2, t_2, px_3, t_3, px_4, t_4, px_5, t_5, px_6, t_6, px_7, t_7, px_8, t_8, px_1)$	332 (319, 340)	4	83 (79.75, 85)
$\gamma_2 = (px_1, t_1, py_1, t_8, px_1)$	82 (78, 86)	1	82 (78, 86)
$\gamma_3 = (px_2, t_2, py_2, t_1, px_2)$	11 (10, 13)	1	11 (10, 13)
$\gamma_4 = (px_5, t_5, py_5, t_4, px_5)$	80 (77, 80)	1	80 (77, 80)
$\gamma_5 = (px_6, t_6, py_6, t_5, px_6)$	81 (79, 82)	1	81 (79, 82)
$\gamma_6 = (px_3, t_3, px_4, t_4, py_4, t_{10}, py_3, t_2, px_3)$	18 (16, 22)	1	18 (16, 22)
$\gamma_7 = (px_7, t_7, px_8, t_8, py_8, t_9, py_7, t_6, px_7)$	84 (81, 87)	2	42 (40.5, 43.5)
$\gamma_8 = (py_1, t_8, py_8, t_9, py_7, t_6, py_6, t_5, py_5, t_4, py_4, t_{10}, py_3, t_2, py_2, t_1, py_1)$	27 (22, 30)	3	9 (7.3, 10)
Maximum cycle time (for bottleneck circuit γ_1) = $\text{Max} \{ T_i / N_i \} = 83$ secs			

Table 8. Elementary Circuits considering Substructures

$\gamma_1 = (p_1, t_{11}, p_2, t_{12}, p_3, t_{13}, p_4, t_{14}, p_5, t_{56}, p_{67}, t_{78}, p_8, t_{19}, p_9, t_1, px_2, t_2, px_3, t_3, px_4, t_4, p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_5, px_6, t_6, px_7, t_7, px_8, t_8, p_1)$
$\gamma_2 = (p_1, t_{11}, p_2, t_{12}, p_3, t_{13}, p_4, t_{14}, p_5, t_{56}, p_{67}, t_{78}, p_8, t_{19}, p_9, t_1, py_1, t_8, p_1)$
$\gamma_3 = (p_{21}, t_{21}, p_{22}, t_{22}, p_{23}, t_5, py_5, t_4, p_{21})$
$\gamma_4 = (p_{24}, t_{21}, p_{22}, t_{22}, p_{24})$
$\gamma_5 = (p_{10}, t_{11}, p_2, t_{12}, p_3, t_{13}, p_4, t_{14}, p_5, t_{56}, p_{67}, t_{78}, p_8, t_{19}, p_{10})$
$\gamma_6 = (p_{11}, t_{12}, p_3, t_{13}, p_{11})$
$\gamma_7 = (p_{12}, t_{14}, p_5, t_{56}, p_{12})$
$\gamma_8 = (p_{13}, t_{78}, p_8, t_{19}, p_{13})$
$\gamma_9 = (px_2, t_2, py_2, t_1, px_2)$
$\gamma_{10} = (px_6, t_6, py_6, t_5, px_6)$
$\gamma_{11} = (px_3, t_3, px_4, t_4, py_4, t_{10}, py_3, t_2, px_3)$
$\gamma_{12} = (px_7, t_7, px_8, t_8, py_8, t_9, py_7, t_6, px_7)$
$\gamma_{13} = (py_1, t_8, py_8, t_9, py_7, t_6, py_6, t_5, py_5, t_4, py_4, t_{10}, py_3, t_2, py_2, t_1, py_1)$

By observing Tables 6 and 7, one can notice the complexities involved in listing the elementary circuits for a marked graph when the substructures replaces the macro places, px_1 and px_5 . It can be seen that

- 1) The number of elementary circuits to be enumerated increases, from 8 to 13;
- 2) The number of places and transitions for the same elementary circuit increases. In other words, the length of an elementary circuit increases; and
- 3) The calculations of cycle time for elementary circuits becomes more.

5.4 ANALYSIS OF AN FMS CELL USING STOCHASTIC PETRI NETS

The purpose of this section is to demonstrate the use of Stochastic Petri Net Package (SPNP) for evaluating a resource-sharing manufacturing system by deriving system performance indices such as throughput and resource utilization.

Stochastic Petri nets (SPNs) are timed Petri nets in which the transition times have exponentially distributed firing rates. In SPNs, the time delays are associated with the transitions. The value of SPNs becomes apparent when one examines the following theorem due to Molloy [29].

Theorem : Any finite place, finite transition, marked stochastic Petri net is isomorphic to a one-dimensional discrete space Markov process. Note: A one-dimensional Markov process is one that has only one random variable (e.g., time is the only random variable).

This opens the way to performance analysis of automated manufacturing systems. This is achieved by modeling the system with a Petri net; then, based on the initial marking, generating the reachability tree and thereby obtaining the equivalent Markov chain and analyzing the

same.

The steps involved in going from the Petri net model to the reachability tree and then to the Markov chain have all been automated and can be found in several software packages. We have used the Petri net tool called SPNP developed at Duke University [32, 33] to evaluate an FMS cell at NJIT.

5.4.1 OVERVIEW OF 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 transition's random firing delay times with exponential distributions.

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 equipment utilization. The package can also be used to derive the transient solution for a 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 [33].

5.4.2 PERFORMANCE ANALYSIS

This section discusses as to how stochastic Petri net models can be used to study the system

performance by evaluating system productivity and utilization of resources using SPNP software package.

It can be observed in Table 6 that the time delays are associated with both; the transitions and the places. In order to make the direct use of the SPNP package, it is necessary to have the time delays be associated with transitions only. This can be done by transferring the time delays associated with any given place to the immediate next transition. e.g., the time delay associated with transition t_{12} will be 5 seconds, as the place p_2 is having a time delay of 5 seconds. Similarly, the time delay for t_{13} will be 45 seconds as the place p_3 being associated with a time delay of 45 seconds and so on. Table 9 summarizes the time delays (τ) and the corresponding firing rates ($1/\tau$) associated with all the transitions which are used to evaluate the Petri net model using SPNP package.

Suppose λ_i is the firing rate of transition t_i and let α denote the probability rate of part acceptance, a variable.

$$\text{Let } \lambda_{15} = 10\alpha \text{ and } \lambda_{16} = 10(1 - \alpha)$$

where λ_{15} and λ_{16} are the firing rates of transitions t_{15} and t_{16} , respectively, representing the initiation of part acceptance and part rejection activities, respectively.

Now, transitions t_{17} and t_{18} being the terminating transitions for the part acceptance and part rejection activities, respectively, their firing rates can be calculated as follows:

$$\begin{aligned} \frac{1}{\lambda_{15}} + \frac{1}{\lambda_{17}} &= 3 & \text{and} & & \frac{1}{\lambda_{16}} + \frac{1}{\lambda_{18}} &= 3 \\ \therefore \frac{1}{10\alpha} + \frac{1}{\lambda_{17}} &= 3 & \text{and} & & \therefore \frac{1}{10(1-\alpha)} + \frac{1}{\lambda_{18}} &= 3 \\ \therefore \lambda_{17} &= \frac{\lambda_{15}}{(3\lambda_{15}-1)} & \text{and} & & \therefore \lambda_{18} &= \frac{\lambda_{16}}{(3\lambda_{16}-1)} \end{aligned}$$

A Petri net text file is prepared using the timing information available in Table 9 which is, then, run using SPNP package. The output file, thus, obtained contains the information regarding the system productivity and the resource utilization. Then, the average cycle time (C) for the system is given by

$$C = \frac{1}{\text{System Throughput}}$$

The results of performance analysis are shown in Figures 5.2, 5.3 and 5.4.

Table 9. Time Delays and Firing Rates Associated with the Transitions

Transition	Time Delays (secs)	Firing Rate (1/secs)	Transition	Time Delays (secs)	Firing Rate (1/secs)
t ₁	3	0.333	t ₁₂	5	0.200
t ₂	8	0.125	t ₁₃	45	0.022
t ₃	5	0.200	t ₁₄	5	0.200
t ₄	7	0.143	t ₁₅	1/10 α	10 α
t ₅	3	0.333	t ₁₆	1/10(1- α)	10(1- α)
t ₆	78	0.013	t ₁₇	(3t ₁₅ -1)/t ₁₅	t ₁₅ /(3t ₁₅ -1)
t ₇	5	0.200	t ₁₈	(3t ₁₆ -1)/t ₁₆	t ₁₆ /(3t ₁₆ -1)
t ₈	68	0.015	t ₁₉	5	0.200
t ₉	3	0.333	t ₂₁	1	1.000
t ₁₀	3	0.333	t ₂₂	72	0.014
t ₁₁	1	1.000			

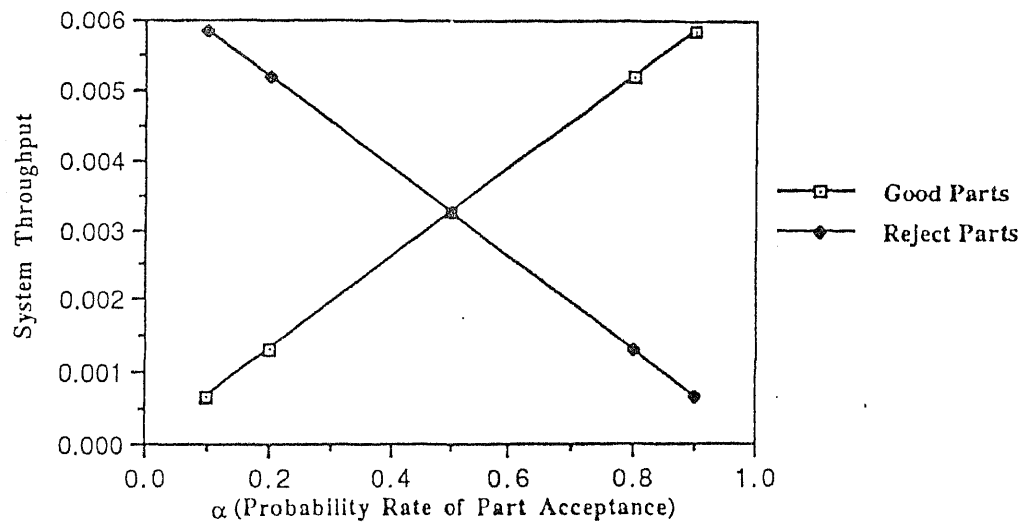


Figure 5.2 System Throughput vs. Probability Rate of Part Acceptance α

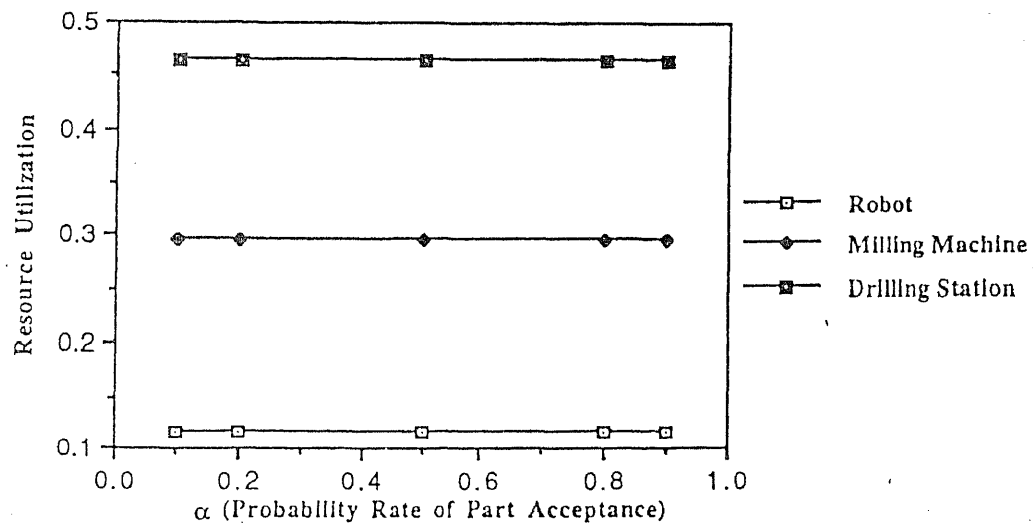


Figure 5.3 Equipment Utilization vs. Probability Rate of Part Acceptance α

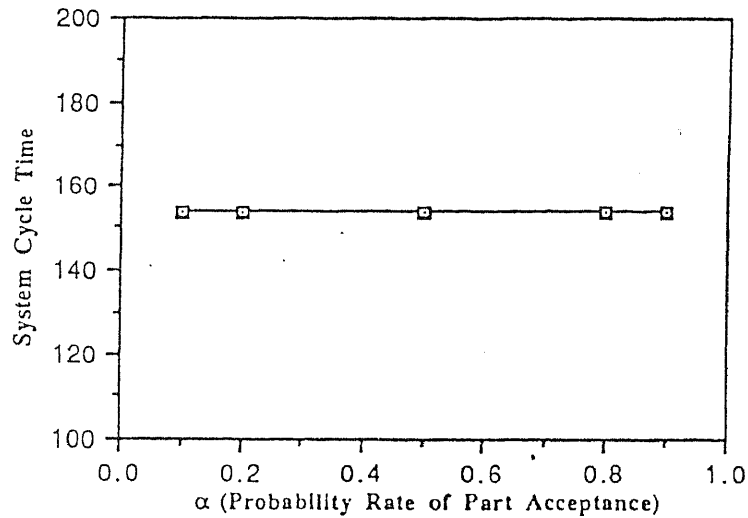


Figure 5.4 Average Cycle Time vs. Probability Rate of Part Acceptance α

The results show that

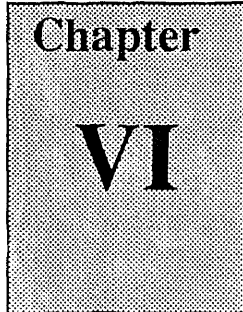
- 1) The throughput of good parts increases as α increases and that of reject parts decreases as α increases;
- 2) The equipment (GE Robot, Milling Machine and Drilling Workstation) utilization remains constant for any value of α ; and
- 3) The average system cycle time remains same for any value of α .

It can be noted that the system cycle time obtained from the evaluation of stochastic Petri net (153.6 secs) is different from that obtained by the evaluation of deterministic Petri net (83 secs) which can be considered as significant deviation between the two values. The error re-

sults from the inappropriate replacement of the deterministic time delay τ with the random time delay of the exponential distribution with rate $1/\tau$. Therefore, one should be careful to use exponential distribution instead of deterministic timing information for performance evaluation. Deterministic Petri nets should be used when a system is of deterministic nature such as the discussed FMS cell.

Using the SPNP package, one may also derive the variations in performance measure when other parameters such as robot operating time delay, milling operation time, drilling activity time and speed of handling system are changing.

CONCLUSIONS AND FUTURE RESEARCH



This chapter briefly outlines the contribution of this thesis and discusses the results obtained. Thereafter, some suggestions to enhance the work done are given and directions for future research work, in the application of Petri nets for modeling and analysis of manufacturing systems, are outlined.

6.1 CONCLUSIONS AND SUMMARY OF RESULTS

This thesis is a tutorial on Petri nets and their models of manufacturing systems. A general introduction to how systems can be modeled using Petri nets is given. The concepts of how the events and conditions of a system can be mapped to the transitions and places of a Petri net model have been illustrated with examples. The execution of the Petri net is discussed along with some important properties of Petri nets.

The mathematical model of a flexible manufacturing system (FMS) cell has been built using Petri nets. The concurrency, conflicts, resource-sharing, and sequential operations have been

built in this model. The detailed design process is illustrated by applying system decomposition, refinement, and modular composition ideas. The resulting net model has the system properties such as liveness, boundedness, and reversibility. Furthermore, the deterministic timing information is incorporated into the places and transitions of this model. This makes possible the analysis of a timed Petri net for cycle time. The timed Petri net is first converted into an equivalent timed marked graph. Then, the standard procedure to find the cycle time for marked graphs is applied. The maximum cycle time calculated for the FMS cell is 83 seconds. In addition, the stochastic timing information is incorporated into the transitions which makes possible the direct use of SPNP package to evaluate the Petri net model of an FMS Cell for system throughput and equipment utilization. The cycle time derived using this package is 153.6 seconds. It has also been found that the system throughput and resource utilization remains constant for any value of probability rate of part acceptance ' α '.

This thesis shows the modeling power of Petri nets by considering models of Fanuc Machining Center and NJIT FMS Cell and following advantages of using it can be seen :

1. They describe the modeled system graphically and hence enable an easy visualisation of complex systems;
2. A systematic and complete qualitative analysis of the system is possible by well developed Petri net analysis techniques [5];
3. The structural properties of Petri nets like liveness, boundedness, connectivity and consistency can be directly linked to certain desirable performance criteria of systems [8]; and
4. Performance evaluation of systems with respect to time is possible using Timed Petri nets [6].

However, certain limitations are also associated with the current modeling methods to deal with

complicated systems, which are indicated below:

1. No detailed CAD information is available in a Petri net model;
2. Timed (deterministic) Petri nets cannot deal with the resource-sharing problem; and
3. Stochastic Petri nets cannot predict the performance of the system which relies heavily on the deterministic timing information.

6.2 DIRECTIONS FOR FUTURE RESEARCH

Application of Petri net theory to practical FMS is a very important area in order to develop modern manufacturing control systems. This thesis provides industrial engineers and academic researchers with a comprehensive real-life example of applying Petri net theory to modeling and analysis of FMS cells. This may help them develop their own applications. In the future, the research is aimed at development of the Petri net based controller with the capacity of error prevention and recovery for this FMS cell [43]. It will enhance the current PLC controller for this cell. In addition, Petri net can compliment the present research in Expert Systems [46]. Certain modifications to increase concurrency will be proposed and Petri nets will be used to evaluate them. Thus designers can judge whether it is worth to accommodate these changes.

The future research in evaluating manufacturing system using Petri nets will focus on the following aspects:

1. Investigation of the discussed issues for more complicated manufacturing systems consisting more complicated interactions and shared-resources. However, the analysis of such complex systems becomes difficult. This problem can be overcome by using colored Petri nets instead of ordinary Petri nets, which lead to compact models for even complex FMS;

-
2. Reduction methods for large stochastic Petri net evaluation and their application to flexible manufacturing systems;
 3. Optimal settings for FMS Cell - The optimal settings can be obtained by comparing the values of performance indices for different temporal PN models. By studying the sensitivity of a performance index as a function of different parameters, we can identify what aspects of the system should be enhanced for significant improvement in system performance; and
 4. Construction of PN controllers based on Petri net model - Once the Petri net models are constructed for manufacturing systems, the PN based controller can be implemented. Artificial intelligence technique should be used to handle errors and emergency situations for a better productivity [43].
-

REFERENCES

- [1] C. L. Beck and B. H. Krogh, "Models for simulation and discrete control of Manufacturing Systems," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp.305-310, April 1986.
- [2] B. H. Krogh and R. S. Sreenivas, "Essentially decision free Petri nets for real time resource allocation," *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, N.C., pp. 1005-1011, April 1987.
- [3] B. H. Krogh and C. L. Beck, "Synthesis of place/transition nets for simulation and control of manufacturing systems," *Proceedings of IFIP Symp. of Large Scale Syst.*, Zurich, August 1986.
- [4] J. L. Peterson, "Petri nets", *ACM Computing Surveys*, Vol. 9, pp. 223-252, September 1977.
- [5] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981.
- [6] C. Ramchandani, "Analysis of asynchronous concurrent systems by timed Petri nets," Ph.D. dissertation, MIT, Cambridge, Project MAC Rep. MAC-TR-120, 1974.
- [7] J. Sifakis, "Petri nets for performance evaluation," in *Measuring, modeling and evaluating-computer systems*, *Proc. 3rd Int. Symp. IFIP Working Group 7.3*, H. Beilner and E. Gelenbe, Eds., North-Holland Publishing Company, pp. 75-93, 1977.
- [8] J. Sifakis, "Structural properties of Petri nets," *Mathematical Foundations of Computer Science, Lecture notes in Computer Science*, No. 64, Springer-Verlag, Berlin, FRG, pp. 474-483, 1978.

-
- [9] M. Hack, "Analysis of production schemata by Petri nets," Technical Report 94, MIT, February 1972.
- [10] D. Dobois and K. Stecké, "Using Petri nets to represent production processes," *Proceedings of the 22nd IEEE conference on Decision and Control*, San Antonio, TX, pp. 1062-1067, 1983.
- [11] Y. Narahari and N. Viswanadham, "A Petri net approach to the modeling and analysis of Flexible Manufacturing Systems," *Annals of Operations Research*, vol. 3, pp. 449-472, 1984.
- [12] T. Murata, "Petri nets: properties, analysis and applications," *Proceedings IEEE*, vol. 77, no. 4, pp. 541-579, April 1989.
- [13] T. Murata et al., "A Petri net-based controller for flexible and maintainable sequence control and its applications in factory automation," *IEEE Trans. Industrial Electron.*, vol. IE-33, no. 1, pp. 1-8, 1986.
- [14] I. Suzuki and T. Murata, "A method for stepwise refinements and abstractions of Petri nets," *Journal of Computer and System Science*, Vol. 27, No. 1, pp. 51-76, August 1983.
- [15] I. Suzuki and T. Murata, "Stepwise refinements of transitions and places," *Informatik-Fachberichte 52: Application and Theory of Petri Nets*, C Girault and W. Reisig (ed.), pp. 136-141, Springer-Verlag, 1982.
- [16] I. Koh and F. DiCesare, "Transformation methods for generalized Petri nets and their applications in flexible manufacturing systems," in *Rensselaer's Second Int. Conf. on Computer-Integrated Manufacturing*, Troy, NY, pp. 364-371, 1990.
-

-
- [17] T. Agerwala and Y. Choed-Amphai, "A synthesis rule for concurrent systems," *Proceedings of 15th Design Automation Conference*, Las Vegas, June 1978, pp. 305-311.
 - [18] R. Y. Al-Jaar and A. Desrochers, "Petri nets in automation and manufacturing," in *Advances in Automation and Robotics*, G. N. Saridis (ed.), vol. 2, JAI Press, 1991.
 - [19] R. Valette, "Analysis of Petri nets by stepwise refinements," *Journal of Computer and System Science*, Vol. 18, pp. 35-46, 1979.
 - [20] M. Silva and R. Valette, "Petri nets in flexible manufacturing," in *Advances in Petri Nets*, G. Rozenberg (Ed.), Springer-Verlag, pp. 375-417, 1990.
 - [21] K. H. Lee, J. Favrel, and P. Baptiste, "Generalized Petri net reduction method," *IEEE Trans. Systems, Man, and Cybernetics*, SMC-17, No. 2, pp. 297-303, 1987.
 - [22] G. Berthelot, "Checking properties of nets using transformations," in *Advances in Petri Nets 1985*, G. Rozenberg (ed.), Springer-Verlag, pp. 19-40, 1985.
 - [23] A. Datta and S. Ghosh, "Synthesis of a class of deadlock-free Petri nets," *Journal of the ACM*, 31(3), pp. 486-506, July 1984.
 - [24] A. Datta and S. Ghosh, "Modular synthesis of deadlock-free control structures," *Foundations of Software Technology and Theoretical Computer Science*, Vol. 241, G. Goos and J. Hartmanis, pp. 288-318, 1986.
 - [25] F. Commoner and A. Holt, "Marked directed graphs," *Journal of Computer and System Science*, Vol. 5, pp. 511-523, May 1971.
-

-
- [26] H. P. Hillion, "Performance evaluation of decisionmaking organizations using timed Petri nets," Master's Thesis Report LIDS-TH-1590, Laboratory for Information and Decision Systems, MIT, Cambridge, MA., August 1986.
- [27] P. Freedman, "Time, Petri Nets, and Robotics," *IEEE Trans. Robotics & Automation*, vol. 7, no. 4, pp. 417-433, 1991.
- [28] Y. C. Ho, "Performance evaluation and perturbation analysis of discrete event dynamic systems," *IEEE Trans. on Automatic Control*, Vol. AC-32, No. 7, pp. 563-572, 1987.
- [29] M. Molloy, "Performance analysis using stochastic Petri nets," *IEEE Trans. Comput.*, vol. C-31, no. 9, pp. 913-917, 1982.
- [30] P. J. G. Ramage and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, Vol. 71, No. 1, pp. 81-98, 1989.
- [31] P. Merlin, "A methodology for the design and implementation of communications protocols," *IEEE Trans. Commun.*, pp. 614-621, 1976.
- [32] J. B. Dugan, A. Bobbio, A. Ciardo, and K. S. Trivedi, "The design of a unified package for the solution of Stochastic Petri net models," *Int. Workshop on Timed Petri nets*, Torino, Italy, IEEE Computer Society, pp. 6-13, 1985.
- [33] G. Ciardo and J. K. Muppala, *Manual for the SPNP Package*, Version 3.0, Department of Computer Science, Duke University, 1990.
-

-
- [34] D. Crockett, A. A. Desrochers, F. DiCesare, and T. Ward, "Implementation of a Petri net controller for a machining workstation," *Proceedings of IEEE Int. Conference on Robotics and Automation*, Raleigh, NC, pp. 1861-1867, 1987.
- [35] M.C. Zhou and Ming C. Leu, "Petri net modeling of a flexible assembly station for printed circuit boards," *Proceedings of IEEE Int. Conference on Robotics and Automation*, Sacramento, CA, pp. 2530-2535, April, 1991.
- [36] M. C. Zhou and F. DiCesare, "Adaptive design of Petri net controllers for error recovery in automated manufacturing systems," *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-19, 5, pp. 963-973, 1989.
- [37] M. C. Zhou, F. DiCesare and A. A. Desrochers, "A top-down modular approach to systematic synthesis of Petri net models for manufacturing systems," *Proceedings of IEEE Robotics and Automation Conference*, Scottsdale, AZ, pp. 534-539, May 1989.
- [38] M. C. Zhou, F. DiCesare and D. Rudolph, "Control of a flexible manufacturing system using Petri nets," in *1990 IFAC Congress Conference*, Vol. 9, pp. 43-48, Tallin, USSR, August 1990.
- [39] M. C. Zhou, F. DiCesare and Dianlong Guo, "Modeling and performance analysis of a resource sharing manufacturing system using stochastic Petri nets," *Proceedings of 5th IEEE Int. Symposium on Intelligent Control*, Philadelphia, PA, pp. 1005-1010, 1990.
- [40] D. L. Guo, F. DiCesare, and M. C. Zhou, "A moment generating function-based approach for evaluating extended stochastic Petri nets," Accepted for *IEEE Trans. Automatic Control*, 1991.
-

-
- [41] M. C. Zhou and F. DiCesare, "Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems," *IEEE Trans. on Robotics and Automation*, Vol. 7, no. 7, pp. 515-527, 1991.
 - [42] M. C. Zhou, *A Theory for the Synthesis and Augmentation of Petri Nets in Automation*, Doctoral Dissertation, ECSE, Rensselaer Polytechnic Institute, Troy, NY, 1990.
 - [43] M. C. Zhou, "Combination of Petri nets and intelligent decision makers for manufacturing system control," *6th IEEE Int. Symp. Intelligent Control*, Arlington, VA, pp. 146-151, 1991.
 - [44] M. C. Zhou, K. J. McDermott, P. Patel, and D. Tang, "Construction of Petri net based mathematical models of an FMS cell," *Proceedings of 1991 IEEE Int. Conference on Systems, Man, and Cybernetics*, Charlottesville, Virginia, pp. 146-151, October 1991.
 - [45] K. J. McDermott and K. V. Kamisetty, "Development of an Industrial Engineering Based Flexible Manufacturing System," *Industrial Engineering Magazine*, October 1991.
 - [46] K. J. McDermott, "An Expert System Based Flexible Manufacturing Cell," *IBM Manufacturing Technology Digest*, Vol. 5, No. 1, pp. 71-75, 1987.
 - [47] T. Agerwala, "Putting Petri nets to work," *IEEE Computer Magazine*, December 1979.
 - [48] J. S. Ahuja, *Advanced Petri Net Techniques for the Comprehensive Modeling, Analysis and Simulation of Flexible Manufacturing Systems*, Master's Thesis, Northeastern University, Boston, MA, 1988.
-