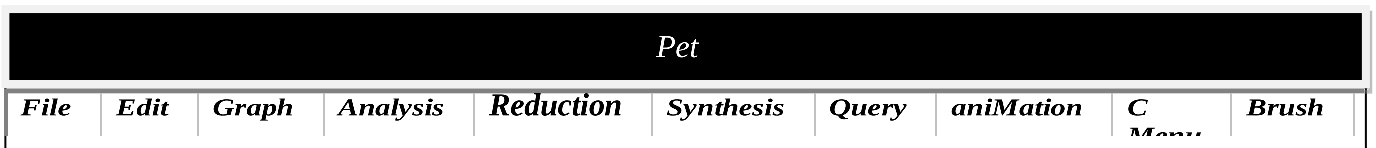
A New Approach to Petri Net Synthesis

The Knitting Technique

D.Y. Chao



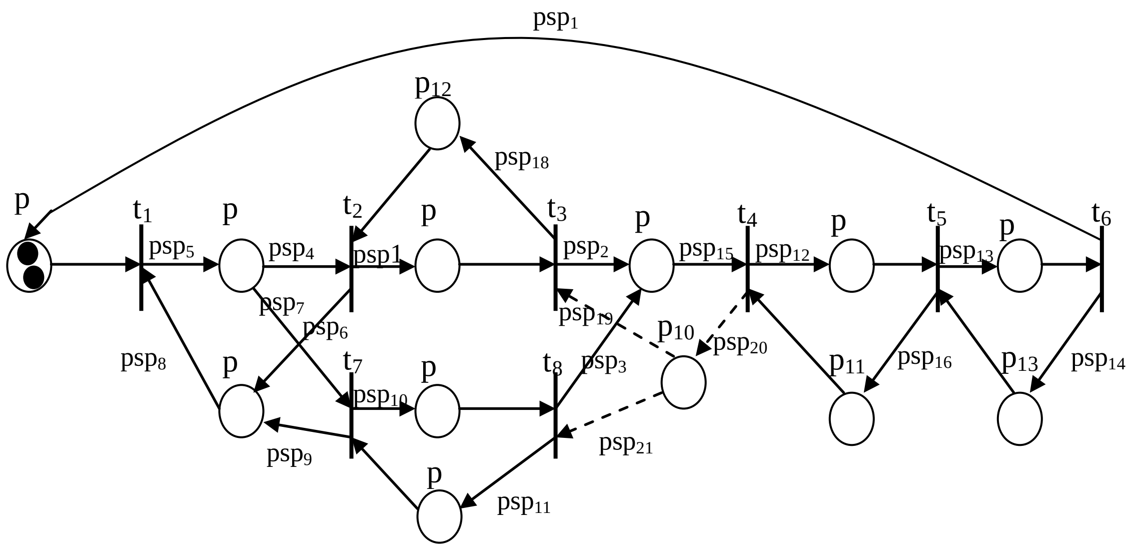
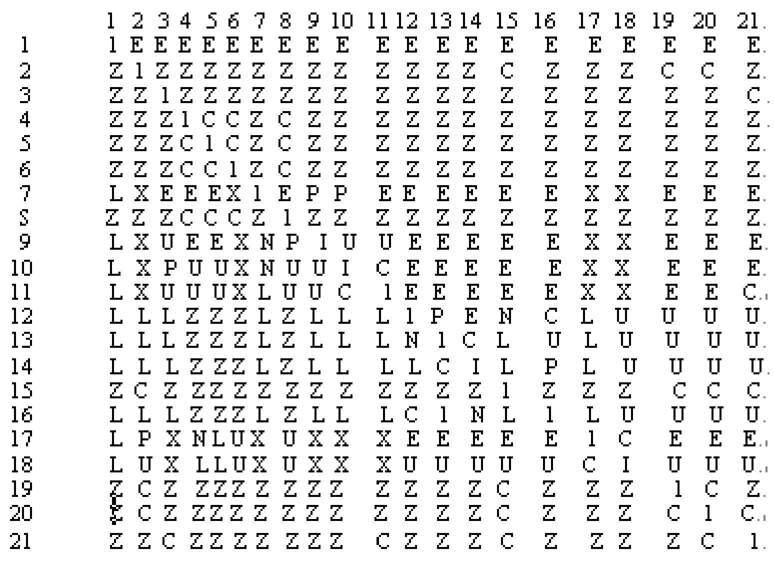


Fig. 12(d) Add [t_8 p_9 t_7], [t_6 p_{10} t_5], [t_5 $p_{11}t_4$], and [t_3 p_{12} t_2];
Add [t_4 p_{13} t_3] and [p_{13} t_8] via TT.2 & TT.4.

THISIS THE GENERATION POINT. THIS IS THE JOIN POINT. This is a PT generation.
Interactive generation!



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21.
1	1	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E.
2	Z	1	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	C	Z	Z	Z	C	C	Z.
3	Z	Z	1	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	C.
4	Z	Z	Z	1	C	C	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z.
5	Z	Z	Z	C	1	C	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z.
6	Z	Z	Z	C	C	1	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z.
7	L	X	E	E	E	X	1	E	P	P	E	E	E	E	E	E	X	X	E	E	E.
8	Z	Z	Z	C	C	C	Z	1	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z.
9	L	X	U	E	E	X	N	P	I	U	U	E	E	E	E	E	X	X	E	E	E.
10	L	X	P	U	U	X	N	U	U	I	C	E	E	E	E	E	X	X	E	E	E.
11	L	X	U	U	U	X	L	U	U	C	1	E	E	E	E	E	X	X	E	E	C.
12	L	L	L	Z	Z	Z	L	Z	L	L	L	1	P	E	N	C	L	U	U	U	U.
13	L	L	L	Z	Z	Z	L	Z	L	L	L	N	1	C	L	U	L	U	U	U	U.
14	L	L	L	Z	Z	Z	L	Z	L	L	L	L	C	I	L	P	L	U	U	U	U.
15	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	1	Z	Z	Z	C	C	C.
16	L	L	L	Z	Z	Z	L	Z	L	L	L	C	1	N	L	1	L	U	U	U	U.
17	L	P	X	N	L	U	X	U	X	X	X	E	E	E	E	E	1	C	E	E	E.
18	L	U	X	L	L	U	X	U	X	X	X	U	U	U	U	U	C	I	U	U	U.
19	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	C	Z	Z	Z	1	C	Z.
20	Z	C	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	C	Z	Z	Z	C	1	C.
21	Z	Z	C	Z	Z	Z	Z	Z	Z	Z	C	Z	Z	Z	C	Z	Z	Z	Z	C	1.

Preface

The advantages of distributed systems are many: resource sharing, cost reduction, improved accessibility and availability, and improved performance. The distributed nature allows concurrency or parallelism, and separate components of a system can work asynchronously. Concurrency increases the computation power, but different components have to cooperate with each other to allow resource sharing.

Due to the distributed nature, it is generally a very intricate problem to design a distributed system correctly and efficiently. The first step to a correct design lies in choosing the most appropriate model. The choice of which method to use for modeling concurrent systems is influenced by (1) ease of use and understanding, (2) modeling and analysis power, (3) generality, and (4) support of automation.

There are various models at hand: shared variables, exchange functions, communicating sequential processes, actors, data flow, and Petri nets. Petri nets is preferred to others due to the following reasons:

(1)The ability to show a precise and graphical representation,

(2)The availability of machine readable descriptions,

(3)The existence of analysis techniques for control aspects.

(4)The capability of employing top-down design methodology, and

(5)The possibility of design and analysis automation.

Thus, PNs have been used for modeling and analyzing concurrent systems. The net behavior depends not only on the graphical structure, but also on the initial marking of the net. Therefore they cannot be determined by static analysis such as dependency analysis; rather, they can be obtained with reachability analysis. The size of reachability graph depends not only by the structure of the net, but also by the initial marking. In general, the larger the initial marking (i.e., more tokens are involved), the larger the reachability graph. It has been shown that the complexity of the reachability analysis of the PNs is exponential. Though such analysis methods as reachability graph, reduction and linear algebra based methods are available, they are of limited use due to their limited capacity. Another disadvantageous fact is that modification and re-analysis may have to be conducted if the analysis methods have detected some undesired properties. Therefore, it is desired to have a synthesis approach which can build up a PN systematically such that the net has following desired logical properties: boundedness, liveness, and reversibility. These properties are critical for a system to operate in a stable, deadlock-free and cyclic way.

Since the PN synthesis research started in late 70's, significant results have been developed for special classes of PNs such as marked Graphs, free-choice PNs, and safe and live PNs. Two dominant synthesis approaches for general classes of PNs are bottom-up and top-down. Bottom-up approaches start with decomposition of systems into subsystems, construct sub-PNs for subsystems, and merge these subnets to reach a final PN by sharing places, transitions, and/or elementary paths or by linking subnets. Top-down approaches begin with a first-level PN and refine the net to satisfy system specification until a certain level is reached such as refinement of transitions and places by general modules with boundedness and liveness or well-defined modules. This approach has the advantage of reducing the scope of analysis from global to local.

However, the existing bottom-up approaches suffer from the difficulty to analyze the global system properties although such useful information as invariants of the global PN can be derived from those of its subnets. In the top-down approaches, there is no easy way to validate all general modules and their design is still a problem given the system specification. the other hand, many reduction rules are very much powerful in reducing PNs. Nevertheless, they are often difficult to be applied to synthesis directly. All these difficulties and problems motivate us to devise some simple but effective rules which can guide the synthesis of PNs with desired properties.

The Knitting Technique (KT) is a rule-based interactive approach. It tackles the synthesis problem from a different perspective. It aims to find the fundamental constructions to build any PN. There are two advantages of KT: (1) reduction of the complexity of synthesis as an interactive tool, and (2) providing knowledge of which construction building which class of nets. It therefore opens a novel avenue to the PN analysis.

Rather than refining transitions, KT generates new paths upon a PN, to produce a larger PN'. The new generations are performed in such a fashion that all reachable markings in PN remain unaffected in PN'; hence all transitions and places in PN stay live and bounded respectively. PN' is live and bounded by making the subnet of the new paths live and bounded. This notion is novel compared with other approaches and could synthesize more general PN than others.

Using KT, designers start with a basic process which is modeled by a set of closed-loop sequentially-connected places and transitions with a so-called home- place marked with a certain number of tokens. The tokens may represent the num- ber of raw materials which can be present in a system each time. Then parallel,alternative, and exclusive processes or operations are added according to the sys- tem pecification. Closed loops for the operations are added according to the resources required by the operations involved. Since expansions are conducted among nodes (either transitions or places) in a global way, the knitting technique is so called. This approach is easy to use due to the simple rules, and leads to a final net which is bounded, live, conservative, consistent, and reversible. The other advantage is that the approach is easily adapted to computer implementation to perform the synthesis of PNs in a user-friendly fashion.

The knitting technique is unconventional but powerful, specially practical in the field of manufacturing and robotics because it can be used to synthesize large and complex Petri nets. It may very well be the best and/or the unique method for synthesizing large and complex Petri nets.

Chapter 2 presents the basic knitting rules: TT and PP rules. Based on which, new paths are from transitions to transitions or from places to places maintaining good properties. he correctness of the rules are proved by finding the invariants of the synthesized net. It is simple, yet it creates more classes of nets than most other synthesis techniques using rules. Paths generations from transitions to places or vice versa are prohibited because the straight forward application of which renders the resulting net unbounded or deadlocked. This restriction is removed in Chapter 3 a second order generation where any generation from a transition to a place must be followed by a generation from a place to a transition and vice versa. This allows more complicated classes of nets to be created, further proving the superiority of our approach.

Chapter 4 presents the reduction algorithm based on the same set of synthesis rules. The algorithm takes polynomial time complexity and is very powerful in the sense that it can reduce nets where traditional methods, usually with no algorithms, could not do it. Chapter 5 further enhances the knitting technique to the synthesis of structures of sequential mutual exclusion which is essential for maintaining the fairness in resource sharing such as in flexible manufacturing systems and other distributed systems. It removes the previous restriction of disallowing IT generations among transitions exclusive to each other. Chapter 6 extends the knitting technique to general Petri nets where any arc can carry multiple weights. Chapter 7 presents the algorithm for finding the structural matrix which records the temporal relationship of processes in a Petri net. It further applies the structure matrix to derive deadlock-free conditions. Chapter 8 applies our knitting technique to a automated manufacturing system which frequently can suffer from an ill-design.

Table of Contents

Chapter 1 The fundamental knowledge for petri net theory

1.1 Introduction

Petri nets are a graphical and mathematical tool for modeling numerous systems owing to their generality and permissiveness []. They have been quite useful as a formal tool to model and analyze systems with such characteristics as being concurrent, asynchronous, distributed, parallel, nonderterministic, and/or stochastic. As a graphical tool, Petri nets, similar to flow charts and block diagrams, can be applied as a visual-communication aid. Besides, the dynamic and concurrent activities of systems can be simulated by moving tokens in these nets. It can also serves as a mathematical tool by building up state equations, algebraic equations, and other mathematics models governing the behavior of systems. Petri nets provide a powerful communication medium between practitioners and theoreticians. Practitioners can learn from theoreticians how nets have been proposed for a very wide variety of applications including modeling and analysis of distributed operating systems, distributed database systems, concurrent and parallel processes, flexible manufacturing/industrial control systems, discrete event systems, multiprocessor systems, data flow computing systems, fault-tolerant systems, programmable logic and VLSI arrays. Other interesting applications considered are communication protocol, neural networks, digital filters, and decision models.

1.2 Petri net structure

1.2.1 Petri net definition

A Petri net is kind of directed, weighted, bipartite graph consisting two kinds of nodes: places and transitions where arcs are either from a transition to a place or from a place to a transition.

Definition of Petri nets: A Petri net can be represented as four-tuple vector PN=(P,T,I,O) which consists of a set of places P, a set of transitions T, and input function I, and output function O.

(1) $P=(p_1, p_2, ... p_n\}$

(2) $T=\{t_1,t_2,\ldots,t_s\}$, $s>0$ with $P \bigcup T \neq \phi$ and $P \cap T \neq \phi$

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

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

where p_i, t_i is an arbitrary element of sets P and T, respectively and the cardinality of the sets P and T is n and s, respectively. The net is called an Ordinary Petri net (OPN) if k=h=1. Otherwise, it is called a General Petri net (GPN).

Definition: Define for a Petri net PN the following set functions as

$I(p, t)= \bullet t = \{p \mid (p, t) \in I)$ = the set of input places of transition t.

$O(p, t)= t\bullet = \{p \mid (t, p) \in O\}$ = the set of output places of transition t.

$I(t, p)= \bullet p = \{t \mid (t, p) \in O\}$ = the set of input transitions of place p.

$O(t, p)= p \bullet = \{t \mid (p, t) \in I)$ = the set of output transitions of place p.

1.2.2 Marked Petri nets

A marking is to assign a nonnegative integer number k in a place p. Place p is then called a place with k tokens.

Definition: The marking $M = \{M_1, M_2, ..., M_n\}$ can be defined as an n-vector where n is the total number of places. The marking for place p_i , denoted by M_i or $M(p_i)$, represents the number of tokens in place p_i.

Definition: A marked Petri net (N, M_o) or $PN(M_o)$ $\{T, P, I, O, M_o\}$ is a Petri net PN with an initial Marking M_o.

1.2.3 Node and elementary path

Definition: Given a marked Petri net $PN(M_o) =\{T, P, I, O, M_o\}$, a node is either a place in P or a transition in T.

Definition: An elementary path is a sequence of nodes: $x_1, x_2, \ldots x_n$, $n\geq1$, such that there exists an arc (x_i, x_{i+1}), $1\leqq i<n$ if $n>1$, and $x_i=x_j$ implies that $i=j \,\forall\, 1\leqq i, j\leqq n$.

1.2.4 Other related definitions

Definition: A transition without any input place is called a source transition.

Definition: A transition without any output place is called a sink transition.

Definition: If a place can accommodate an infinite amount of tokens, the corresponding Petri net is called a net with an infinite capacity. Otherwise, it is called a finite capacity net.

1.2.5 Graphical representation

In Petri net graphical representation, places are represented by circles, transitions by bars. Arcs are labeled with a positive integer to represent their weights. An m-weighted arc can be replaced by a set of m parallel arcs with unity weight (weight equals to 1). Usually, the label for unity weight can be omitted. Marking can be represented by placing tokens in a place. Pictorially, this is drawn as black dots in places. For instance, a place p which is marked with k tokens can be graphed as a circle with k dots in the place.

1.3 Petri net firing rule

1.3.1 Enabled transition

A transition t is said to be enabled if each of its input place p; consists of at least one token. The formal definition is described as follows:

Definition: A transition $t_j \in T$ in a marked Petri net PN with marking M is said to be enabled if for all $p_i \in \bullet t$, $M(p_i)\geq I(p_i, t_j)$.

1.3.2 Firing rules

In a marked Petri net with marking M, an enabled transition t may or may not fire depending on where the input tokens actually be consumed. The firing of an enabled transition t removes one token from each of its input places and adds one token into each of its output places. The consequence of firing a transition will result a change from original marking M to a new marking M'.

Definition: The firing rules are:

(1) A transition $t \in T$ is enabled if and only if $M(p) > 0$ when $I(p, t) = 1 \forall p \in P$.

(2) Firing an enabled transition t results in a new marking M' defined by

$M'(p) = M(p) + O(p, t) - I(p, t), \forall p \in P$.

For a transition in simple Petri nets, the time from enabling to firing is indeterminate. Firing of a transition is assumed to be an instantaneous event.

1.3.3 Conflict

Definition: A set of transitions $T_c \subset T$ is called in conflict if the firing of any subset of transitions $\{t_i\}$ of $T_c \subset T$ results in a marking in which some other transition $t_j \in T_c$ and $t_j \notin \{t_i\}$ is disabled.

In Figure 1-1, several examples of Petri nets being conflict are given. Note that transitions which share input places need not be in conflict. If the marking in the shared input places has enough tokens to enable each competing transition individually, then those transitions are not in conflict.

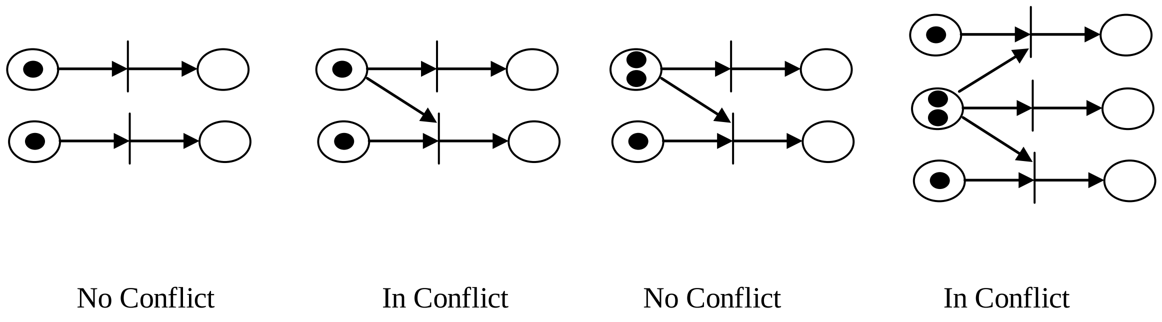


Figure 1-1. The figure is extended from Michael K. MOLLOY

1.3.4 Mutually exclusive

Definition: A set of transitions $T_e \subset T$ is termed mutually exclusive if the firing of any transition $t_i \in T_e$ results in a marking in which all other transitions $t_j \in T_e \forall i \neq j$ are disabled

1.3.5. Home place

The home place(s) is defined as the place(s) with tokens which can enable a transition at the initial marking. A formal definition is described as follows.

Definition: Suppose transition is enabled in the initial marking of a PN and a place p_h is an input place of t_i, p_h is defined as a home place.

1.4 Petri nets modelling applications

1.4.1 General concept

In modeling a particular discrete event system, places represent conditions, and transitions represent events. A set of input places to a transition can be viewed as pre-conditions of event, whereas the output places from a transition represent post-conditions of the event. The presence of a token in a place is explicated as holding the truth of the condition associated with the place. Some typical examples of interpretation of a transition and their input places and output places are shown in Table 1-1.

Transition	Input places	Output places
Computation step	Input data	Output data
Task or job	Resources requested	Resources released
Clause in logic	Conditions	Conclusions
Event	Precondition	Postcondition

Table 1-1 Some interpretation of transitions and places

1.4.2 Typical examples

A. Communication protocols

Petri nets can be employed to represent and model essential features of a communication system. The properties of a Petri net are usually applied to validate a communication protocol. The Petri net shown in Figure 1-2 is a typical example of modeling a communication protocol between two processes.

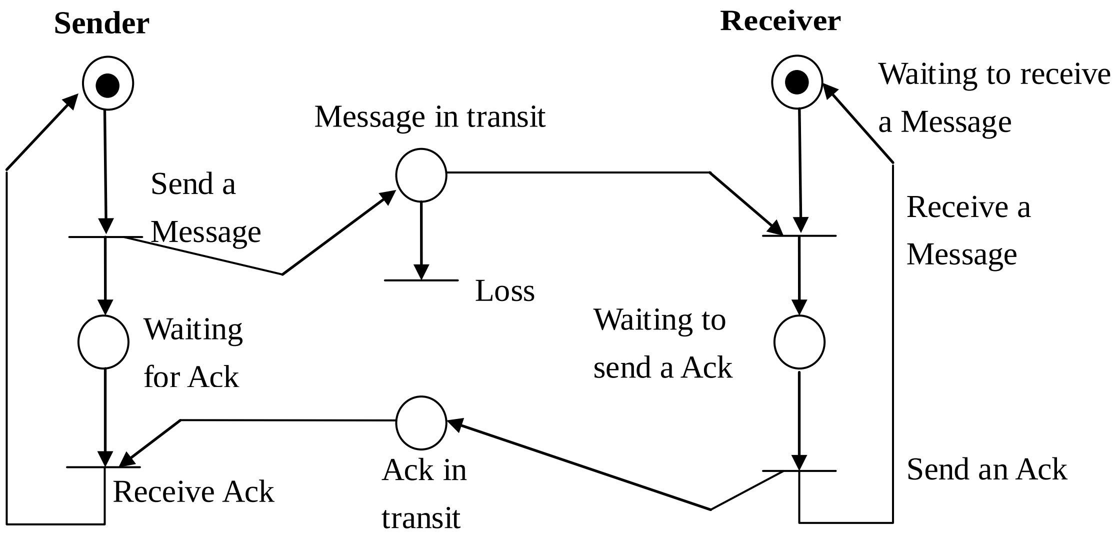


Figure 1-2 A example of modeling a communication protocol between two processes.

B. Synchronization control

In a multiprocessor concurrent or distributed system, resources and data are usually shared among sites. Processes must be controlled or synchronized to insure correct overall operation. Petri nets have been used to model various synchronization mechanism, including readers/writers processes, producers/consumers processes, and dining philosopher problems. Figure 1-3 illustrates a synchronization problem of readers/writers, where the k tokens in p_1 represent k processes which may invoke read and write actions in a shared storage modeled by place p_3. Writer processes must mutually exclusive all other reader and writer process, but up to k processes may be resided in place

p_2(reading) when there is no token in p_4(writing). p_4 is safe because only zero' or one token is allowed in that place.

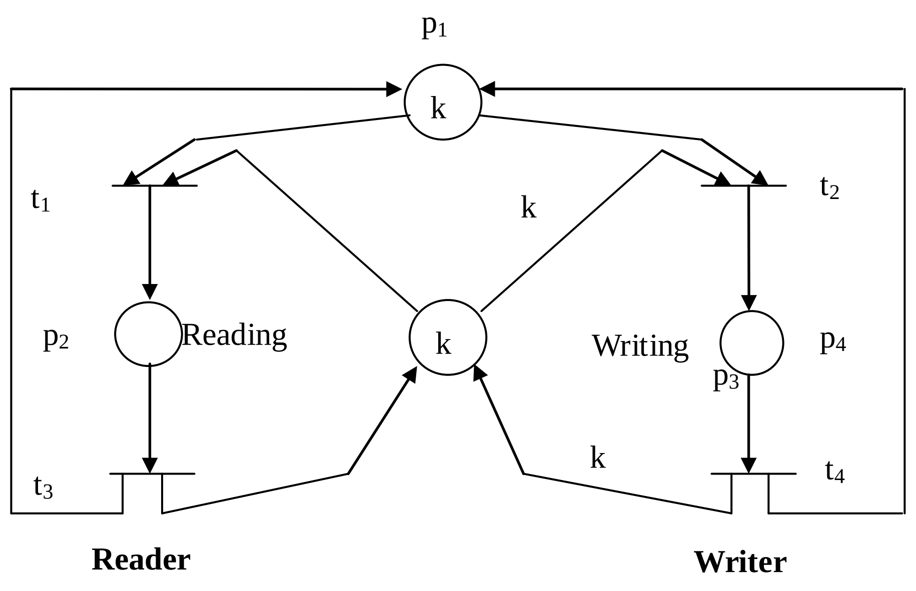


Figure 1-3 A reader/writers synchronization problem.

C. Multiprocessor systems

A multiprocessor system can be easily and explicitly represented by Petri nets. For example, Figure 1-4 shows a multiprocessor system with four processors, three common memories, and three buses. Tokens in place p_1 represent the processors executing in their private memory, and tokens in p_2 represent free buses. Transition t_1 indicates the issuing of access requests. p_3 contains outstanding requests and p_4 represents processors having acc s to common memories. t_2 and t_3 model the memory choice: firing t_3 choose the memory accessed by p_4, whereas firing t_2 refers to any other memory. Tokens in place p_5 represent processors are queued to the same common memory that has been accessed by another processor (token) in p_4. Firing t_5 represents the completion of the access to the memory for which processors in p_5 are queued. On the other hand, firing $t_4$represents the end of access to a memory for which there is no outstanding request.

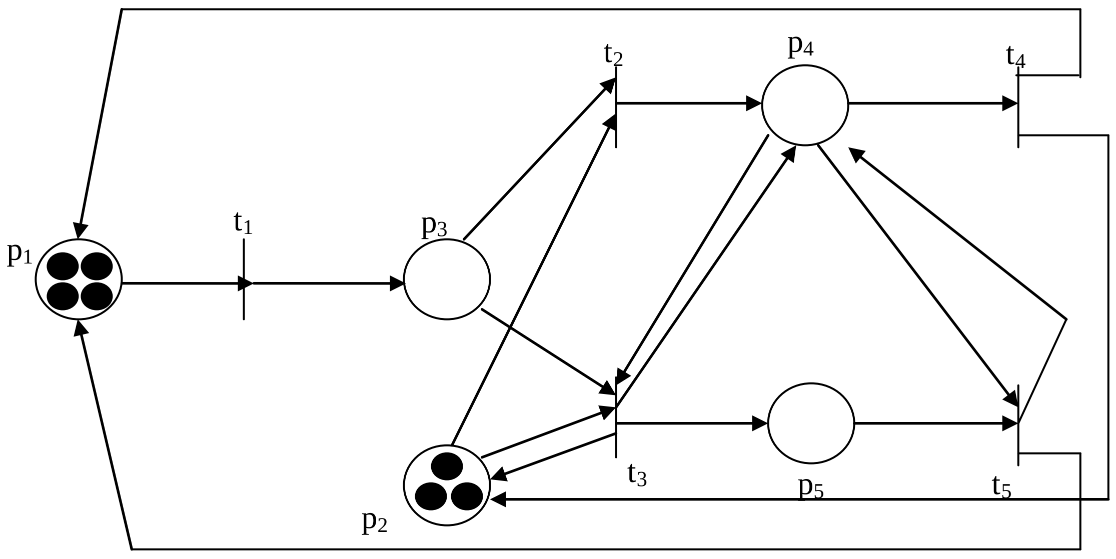


Figure 1-4 Multiprocessor systems with four processors, three common memories, and two buses.

1.5 Subclasses of Petri nets

This part briefly introduces some Petri net subclasses. These subclasses which have been defined are all syntactic or structural subclasses. The major advantage of this classification is to increase the modeling power for Petri nets. The characteristics reside in these subclasses can be used to determine if a given Petri net is a member of the specified subclass. The major properties, advantages, and drawbacks of each subclass is described later in this section. Before we introduce some subclasses of Petri nets, the ordinary Petri net is defined as follows:

Definition: A Petri net is called ordinary when all of its arc weights are 1's.

1.5.1 State Machine(SM)

Definition: An SM is an ordinary Petri net such that each transition has exactly one input place and one output place; i.e.,

$|\bullet t|=|t\bullet|= 1$ for all $t \in T$.

Several properties of SMs are apparent. First, since the number of tokens in SMs remains constant, they are strictly conservative.

1.5.2 Marked Graph(MG)

Definition: An MG is an ordinary Petri net such that-each place has exactly one input transition and one output transition. i.e.,

$|\bullet p|=|p\bullet|= 1$ for all $p \in P$.

Since in an SM, each transition has one input and one output. In an MG, each place has one input and one output, MGs are dual of SMs. They are also dual from modeling viewpoint. An SM can be used to model a conflict situation such that one place has multiple outgoing arcs but can't represent a concurrent activity or the waiting of synchronization such that one transition has multiple outgoing arcs. On the other hand, an MG can represent concurrency and synchronization but cannot model conflict or data dependent decisions.

The properties for an MG includes liveness, safeness, and reachability. In the analysis of these properties, the major structural parts of an MG of interest are its cycles.

Definition: A cycle in an MG is a sequence of transitions t_{j1}, t_{j2} t_{j3} ...t_{jk} such that for each t_{jr} and t_{jr+1} in the sequence there is a place p_{ir} with $p_{ir} = t_{jr}\bullet$, $p_{ir} = \bullet t_{jr+1}$ and $t_{j1} = t_{jk}$. A cycle is a closed path from a transition back to that same transition.

1.5.3 Free Choice Net(FC)

Definition: A FC an ordinary Petri net such that each arc is either an unique output of a place or a unique input t a transition. i.e.,

$|p\bullet|$ 1 or$\bullet$〔$p\bullet$〕$= \{p\}$ for all $p \in P$.

And this is equivalent to

$p_1\bullet \cap p_2\bullet \neq \emptyset \rightarrow |p_1\bullet| = |p_2\bullet| = 1$ for all $p_1, p_2 \in P$.

By the definition of FC, if a place is an input to multiple transitions(i.e., potentially conflict), then it is the only input for all of these transitions. Therefore, either none of these conflicting transitions are enabled or all are enabled simultaneously. This makes the firing choices of these conflict transitions freely.

1.5.4 Extended Free-Choice.Net(EFC)

Definition: An EFC is an ordinary Petri net such that

$p_1\bullet \cap p_2\bullet \neq \emptyset \rightarrow p_1\bullet = p_2\bullet$for all $p_1, p_2 \in P$.

InEFC, each transition has at most one shared input place with another place and so also ' restrain the situation in which conflicts may occur.

1.5.5 Asymmetric Choice Net(AC)

Definition: An AC is an ordinary Petri net such that

$p_1\bullet p_2\bullet \neq \emptyset \rightarrow p_1\bullet \subseteq p_2\bullet$or $p_1\bullet \supseteq p_2\bullet$for all $p_1, p_2 \in P$.

Figure 1-5 displays the typical structures that represents these subclasses and an overview of these subclass classification in the Petri net structures is shown in Figure 1-6. It can be easily shown that FCs are a generalization of the structures of both SMs and MGs.

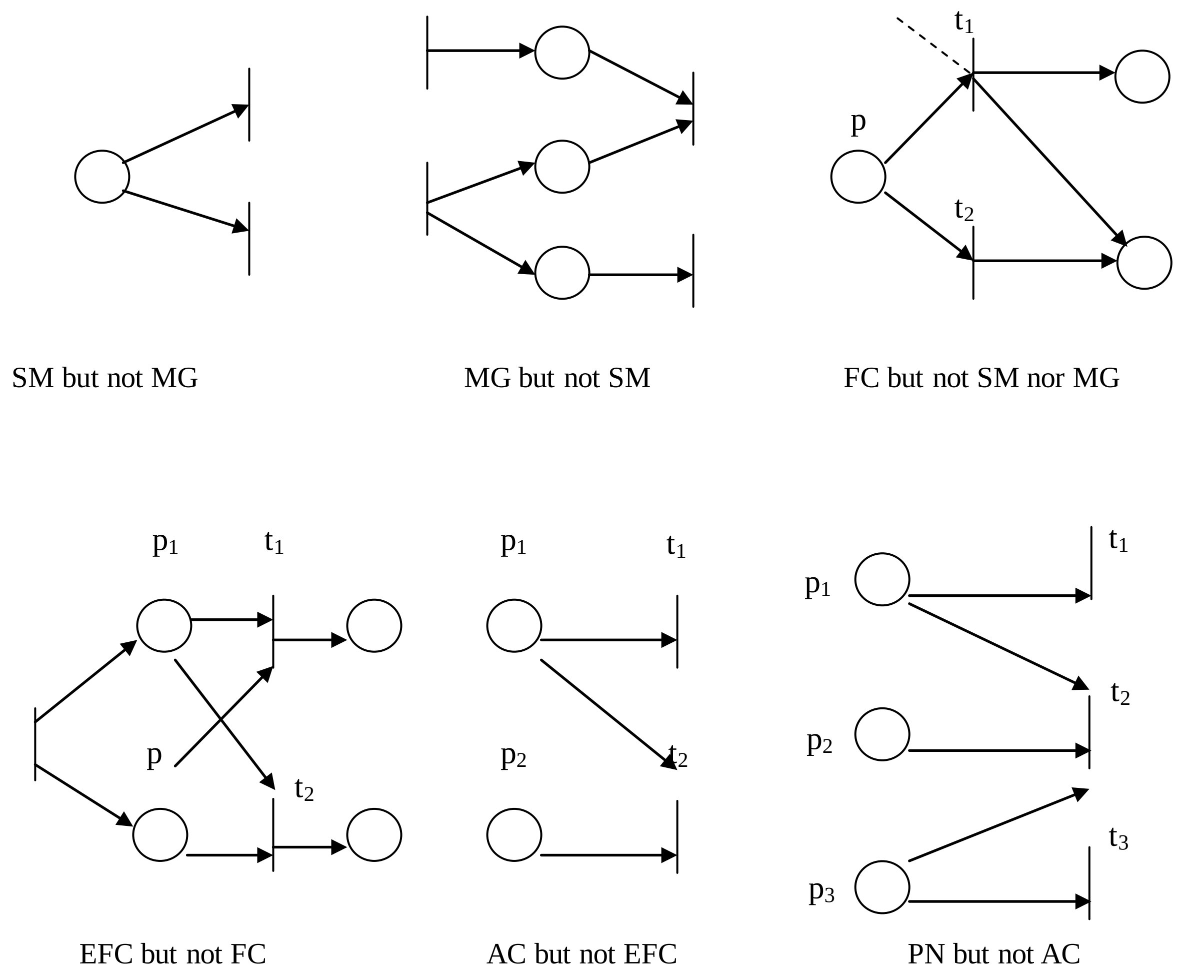
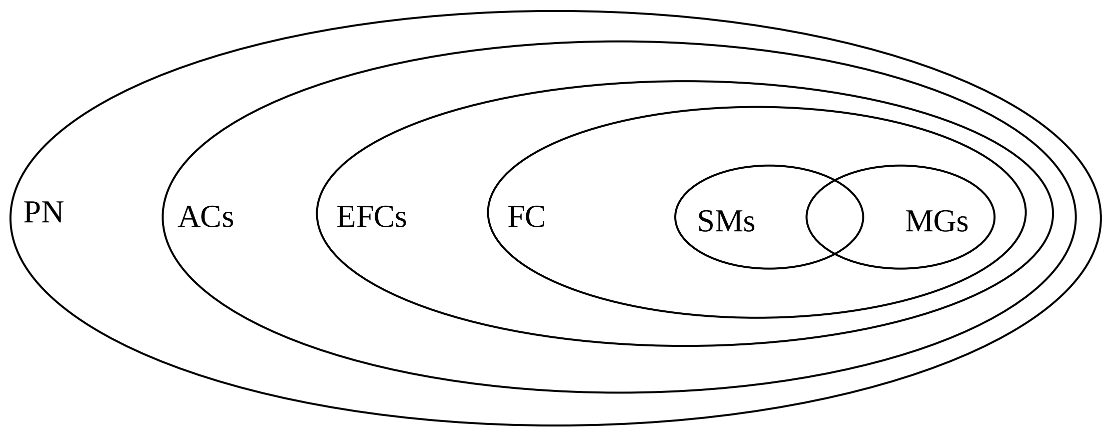


Figure 1-5 The typical structure that represents these subclasses



1.6 Petri net behavior properties

A major strength of Petri nets is their support for analysis of many properties and problems associated with concurrent systems. Two types of properties can be studied with a Petri net model: those which depend on the initial marking, and those which are independent of the initial marking. The former one referred to as marking-dependent or behavioral properties, whereas the latter type of properties is called structural properties. In this section, we only discuss behavioral properties and we will discuss the structural properties in section 1.8.

1.6.1 Safeness

Definition: A place $p_i \in P$ of a Petri net C= {P, T, I, 0, M_o} is safe if for all $Mi \in R(C, M_o)$, $M(p_i) \leq 1$. A Petri net is safe if each place in that net is safe.

A place in a Petri net is safe if the number if tokens in that place never exceeds one. A Petri net is safe if all places in the net are safe. Safeness is very important property for hardware device. If a place is safe, then it can be easily implemented by a single flip-flop.

1.6.2 Reachablility

Reachablility is perhaps a fundamental basis for analyzing the dynamic properties of any Petri net system. The firing of an enabled transition will change the markings. And a sequence of firings will result in a sequence of markings.

Definition: Given a Petri net PN, a marking M_k is reachable if there exists a sequence of firings that lead M_o to M_k . This expression can be denoted by $M_k \in R(PN, M_o)$.

1.6.3 Boundedness

Definition: A place in a PN is said to be k-bounded if there exists a finite number $k \in N$ such that
$M(p_i) <k \forall M \in S$. A PN is said to be bounded if each place p; is bounded $\forall p_i \in P$. For example, in some particular case as figure 1-2 shown, a Petri net is both bounded and safe. This result is specially useful for modeling an operating system where places are used to represent buffers or registers for storing intermediate data. By examining whether the net is bounded and safe, it can be predicted that if there will be overflows in the buffers or registers.

1.6.4 Liveness

Liveness usually means the complete absence of deadlocks in operating systems. A Petri net is said to be live if, no matter what marking has been reached from, it is possible to ultimately fire any transition of the net by progressing through some further firing sequence. This means that a live Petri net guarantees deadlock-free operation.

Definition: A transition ti is said to be live if for all $M_k \in S$ there exists a marking sequence M_j which enables t_i .

Definition: A PN is said to be "live" if each transition $t_i \in T$ is live.
There are other level concepts related to liveness which have been considered in studies of deadlock. A transition t_i can be categorized as follow:

Level 0: t_i can never be fired.
Level 1: t_i is potentially fireable.
Level2: For every integer n, there exists a firing sequence in which occurs at least n times.
Level 3: There exists an infinite firing sequence in which t_j occurs infinitely often.

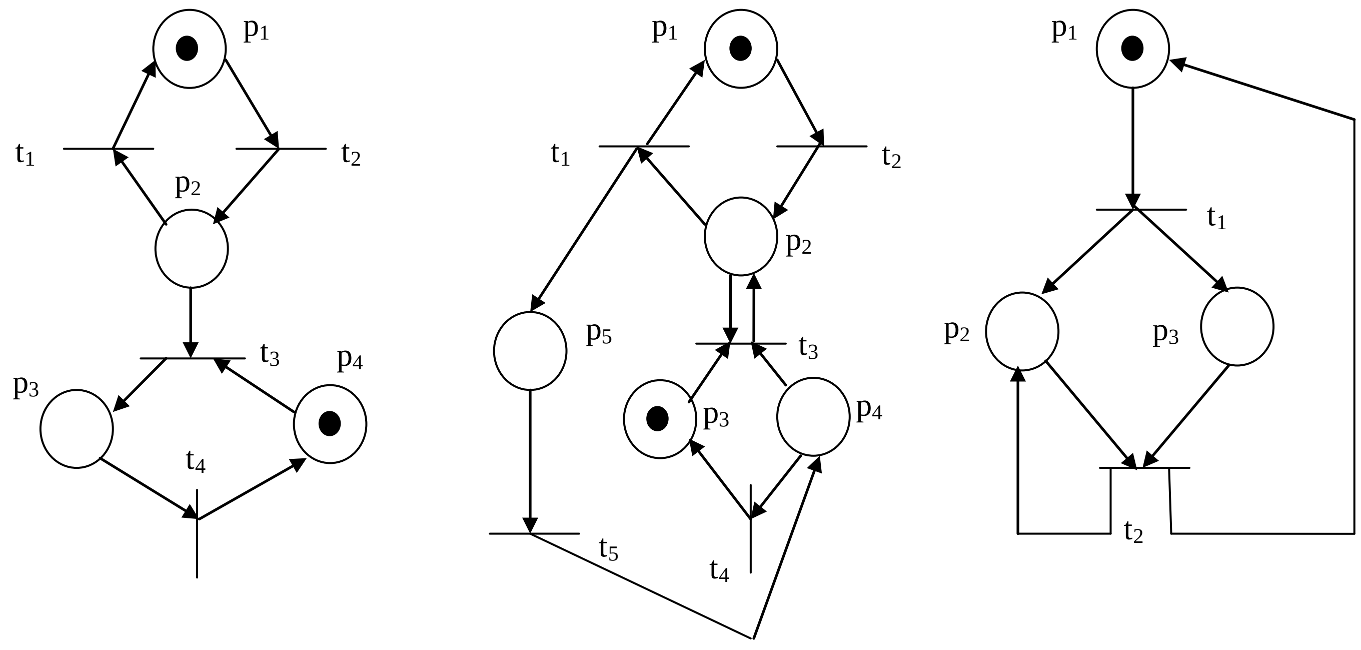
1.6.5 Reversibility and Home state

Definition: A Petri net PN is said to be reversible if, for each marking M in $R(M_o)$, M_o is reachable from M.

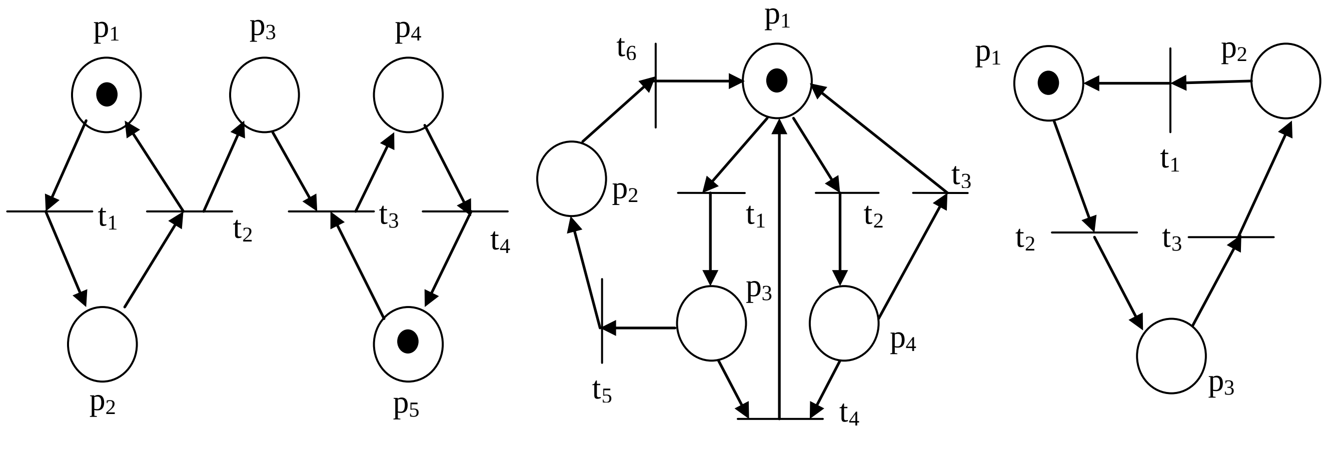
In other words, a reversible Petri net, one can-always gets back to the initial marking. In many applications one may want to go back to some(home) state instead of get back to the initial state. In this case, reversibility condition can be relaxed to a home state.

Definition: A marking M' is defined as a home state if, for each marking M in $R(M_o)$, M' is reachable from M.

Readers should notice that the above three properties, boundedness, liveness, andreversibility, are independent f each other. For example, a bounded Petri net can be live or nonliveand reversible or nonreversible. Figure 1-7 shows examples of Petri nets for all possible combinationsof three properties.



(a) Bounded only (b) Reversible only (c) Live only



(d) Live and reversible only (e) Bounded and Reversible (f) Bounded live reversible

Figure 1-7 Examples of Petri net for all possible combination of three properties.

1.6.6 Conservation

Conservation is an important property when Petri nets are used to model a resource sharing system where the number of resources in the system always remain' constant. For example, a set of two line printers is represented by an initial marking of two tokens. A request of the printing job can be represented by a transition which has this place as an input. A release of the line printer is a transition with an output to the line printer place. We would like to show that tokens represented by resources are neither created nor destroyed.

Definition: A marked Petri net PN(M) = {P,T,I,O,M} is strictly conservative if for M' $\in$ R(PN,M),

$\sum pi \in$ pM'(p_i) = $\sum pi \in$ pM(p_i).

Strict conservation is a very strong relationship. For instance, one can easily show that the number of inputs to each transition equals the number of outputs from that transition, |I(ti)| = |O(ti)|. However, a Petri net may not necessarily preserve the one-to one mapping between resources and tokens. For instance, one token may be later used to produce multiple tokens through a transition. In general, we would like to define a weighting of tokens(an integer is assigned to each place). For a broader perspective, the weighted sum for all reachable marking should be constant while considering the conservation with respect to a weighting factor.

Definition: A marked Petri net PN(M) = {P,T,I,O,M} is conservative with respect to a weighting vector w, w=(w_1i,w_2, ... ,$w_{n)}$, n=|P|, $w_i \geq 0$, if for all M' $\in$ R(PN,M), $\sum pi \in$ P w_i•M'(p_i)= $\sum pi \in$ P wi•M(p_i).

Readers can easily find that a strictly conservative Petri nets is just a special case with theweighting vector equals (1,1, … ,l). On the other hand, all Petri nets are conservative with respect toweighting vector (0,0, ... ,0).

1 .6.7 Coverability

Suppose we would like to know if any state is reachable with M(p_4)   l andM(p_9)   1. This problem is similar but slightly different to reachability problem. It is called coverability problem.

Definition: A marking M in a Petri net (PN, M_o) is said to be coverable if for each p in the net, there exits a marking M' in R(M_o) such that M'(p) $\geq$M(p). Coverability is closely related to level 1 liveness (potentially firable). Let M be the minimum marking needed to enable a transition t. Then t is dead(non level 1 live) if and only if M is not coverable.

Definition: t is level 1 live if and only if M is coverable.

1.6.8 Persistence

Remind that conflict means that for any two enabled transitions, the firing of any one will disable the other one, whereas persistence can be thought as a conflict-free property. A Petri net is said to be persistent if, for any two enabled transitions, the firing of any one will not disable the other.

In the other words, the persistence property guarantees that once a transition is enabled, it will stay enabled until it fires. Persistence property is useful in modeling the context of parallel programming schemata and speed-independent asynchronous circuits. Besides, a safe persistent Petri net can be transformed to a marked graph where for any transition, the input to a transition and the output from a transition is always 1. Note that all marked graphs are persistent, but not vice versa. For example, the unbounded(not safe) net shown in Figure 1-7 (c) is persistent but not a marked graph.

1.6.9 Synchronic Distance

Synchronic distance concept is firstly introduced by C.A.Petri. It is a metric used to determine the degree of mutual dependence between two events in a condition/event system. The synchronic distance between any two transitions can be obtained as follows:
For each firing sequence of the Petri net in one iteration, calculate the absolute value of the difference between the numbers of the firing of these two transitions.

Definition: Given any two transitions t_1 and t_2 in a initial marked Petri net PN={P, T, I, O, M_o}, the synchronic distance is defined byd_{12}=max $|D_1(t1)-D_2(t2)|$ where D is any firing sequence starting at any marking M in $R(M_o)$ and $D(t_i)$ is the number of times that transition t_i, i=1 or 2 fires in d. For example, in the net shown in Figure 1-7 (d), d_{12} =1, d_{34} =1, and d_{14} = ∞.

1.7 Analysis method for Petri nets

Usually, there are three categories of approaches used for analyzing Petri. nets: A(Reachability, tree approach, B) Incidence matrix and state equation approach. The reachability tree approach is usually limited to a small net because it has been shown that the: complexibility of reachability analysis for Petri nets is exponential. On the other hand, matrix approach is powerful in analysis but also applicable to some particular subclasses of Petri nets.

1.7.1 Reachability Tree

Given a initial marking Mo for a Petri net PN, it can result numerous new marking by a sequence of firing of enabled transitions. And from these new markings, we can obtain even more new markings. We can then generate a tree structure called reachability tree to represent these markings starting from Mo (the root). In reachability tree, markings generated from initial marking are represented by nodes. Each arc represents the firing of an enabled transition which will transform from one marking to another.

In order to simplify the representation, the symbol co is introduced to represent an infinite set of values. For any integer $n > \omega$, $n \pm \omega = \omega$ and $\omega \geq \omega$. The physical meaning for that is that if a component of a covering marking is co, then there exists a "loop" in the path from root to particular covering marking.

If one covering marking contains more than one it may indicate a "loop" interaction among the marking exchanges. Consider the example in Figure 1-8 and its corresponding reachability tree in Figure 1-9.

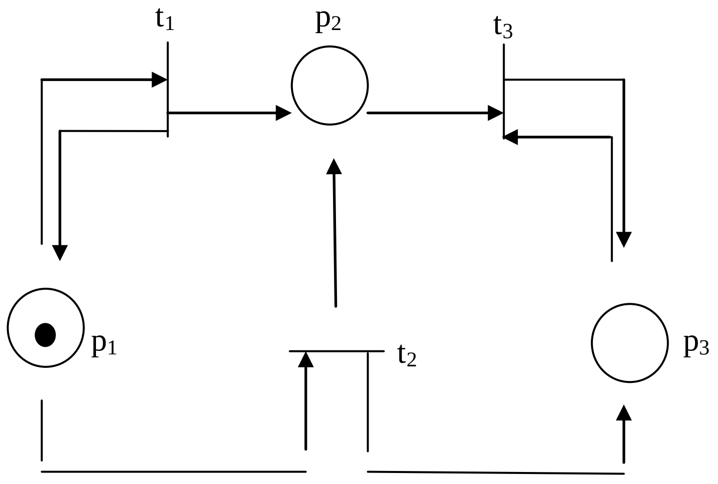


Figure 1-8

$$\begin{tikzcd}
 & & (1, 0, 0) \arrow[dl, "t_1"'] \arrow[dr, "t_2"] & & \\
 & (1, \omega, 0) \arrow[dl, "t_1"'] \arrow[dr, "t_2"] & & (0, 1, 1) \arrow[dr, "t_3"] & \\
(1, \omega, 0) & & (0, \omega, 1) \arrow[dr, "t_3"] & & (0, 0, 1) \\
 & & & (0, \omega, 1) &
\end{tikzcd}$$

Figure 1-9

Again, the reachability tree can be used to detect the safeness, boundness, conservation , and coverability problems. However, the use of w instead of an exact value also degrades its analytical power due to a loss of information. It can't use to solve the reachability or liveness problems or determine the possible firing sequences For instance, two different Petri nets may result in the same readability tree Figure 1-10(a) and Figure 1-10(b) describe the problems, whose reachability tree is given in Figure 1-11. Another problem exists in the liveness analysis where a covering marking in a reachability tree may be a deadlock. Figure 1-12 presents this problem.

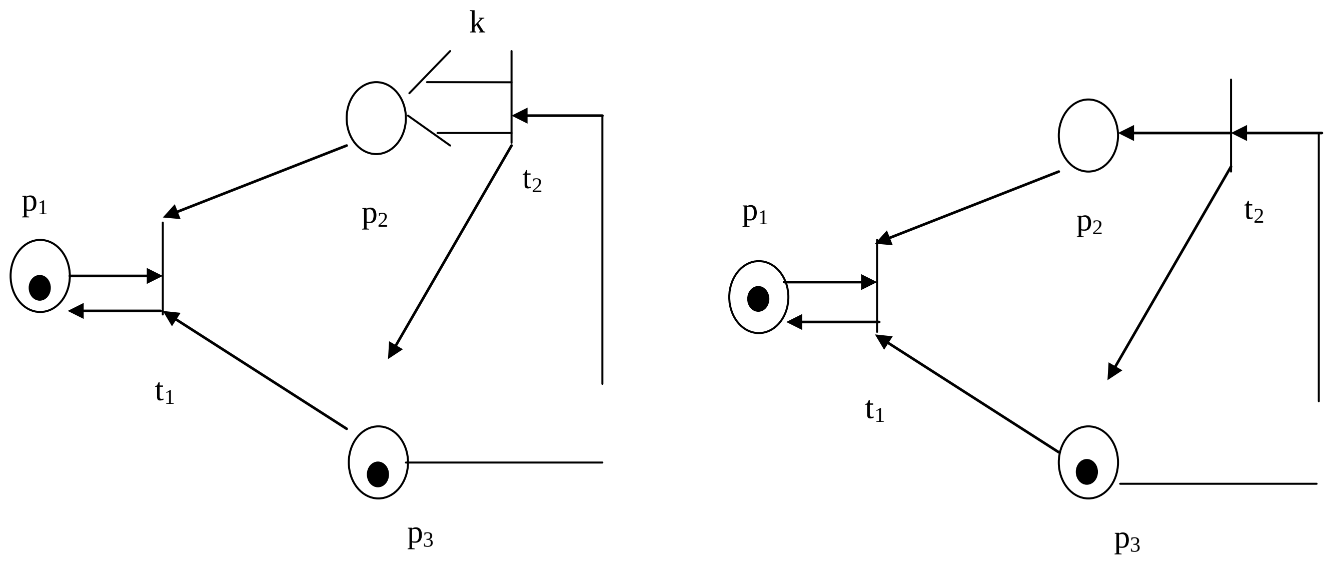


Figure 1-10

$$\begin{tikzcd}
 & (1, 0, 1) \arrow[d, "t_2"] & \\
 & (1, \omega, 1) \arrow[dl, "t_1"'] \arrow[dr, "t_2"] & \\
(1, \omega, 0) & & (1, \omega, 1)
\end{tikzcd}$$

Figure 1-11

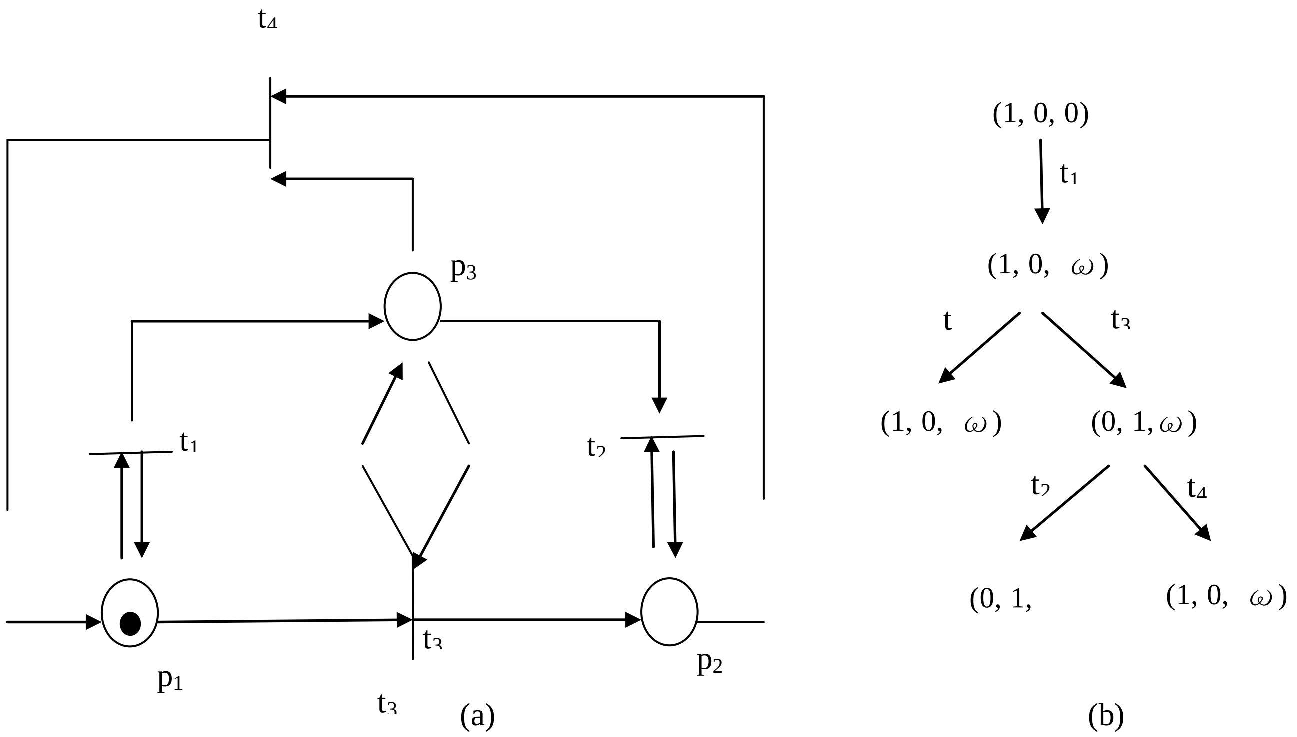


Figure 1-12

1.7.2 Incidence Matrix and State Equation

This incidence matrix and state equation approach basically uses matrix equation to govern and present the dynamic behaviors of a Petri net system. However, because of non deterministic nature in Petri net and some pre-constraint in the equations, the solvability of these equations is somewhat limited.

A. Incidence Matrix

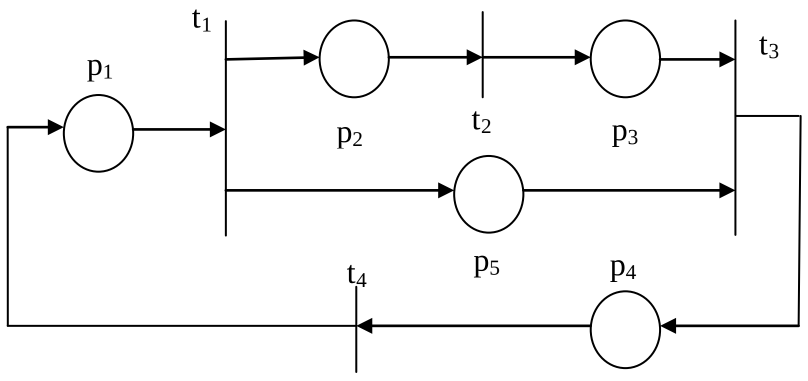
Definition: The incidence matrix A = [a_{ij}] is an..inxi matrix of integers for a Petri net PN with n places and m Its entry aij is defined by

$a_{ij}=a_{+ij}-a_{-ij}$

where a_{+ij} = w(i,j) is theweight for the outgoing are from transition i to place j and a_{-ij} = w(j,i) is the weight for the incoming are from place j transition i.

The physical meaning fors the number of tokens changed for place j when each time transition i fires. Accordingly, a_{+ij} represents the number of tokens removed and a_{-ij} represents the number of tokens added when transition i fires. Therefore, each row of A corresponds to a transition and each column of A corresponds to a place.

For example, suppose the weight for each arc in Figure. 1-13.(a)*is 1. The incidence matrix is shown Figure 1-13(b).



$$A=\left|\begin{array}{c|ccccc} & p_1 & p_2 & p_3 & p_4 & p_5 \\ t_1 & -1 & 1 & 0 & 0 & 1 \\ t_2 & 0 & -1 & 1 & 0 & 0 \\ t_3 & 0 & 0 & -1 & 1 & -1 \\ t_4 & 1 & 0 & 0 & -1 & 0 \end{array}\right|$$

(a) (b)

Figure 1-13

B. State Equation

In writing matrix equation, we write a marking Mi as a M×1 colum vector. The jth entry of M_i denotes the number of tokens in place j right after the ith firing in certain firing sequence The ith firing vector u_i is an n ×1 colum vector of n-1 0's and one nonzero entry. We can write the following state equation for a Petri net

$M_i = M_{i-1} + A^T u_i$ i=1,2, ...

C.Necessary Reachability Condition

A coverable marking Mk is reachable from initial Mo through a firing sequence {ul, u2, ... ,uk}, then Mk ca>i be written. as follows:

$$M_k = M_0 + A^T \sum_{i=1}^{k} \sum_{\rightarrow}^{\rightarrow} ui$$

or

$A^T x = \Delta M$ where $x = \sum_{i=1}^{k} ui$ is a firing count vector and $\Delta M = M_k - M_o$.

For a homogeneous system, $\Delta M = 0$.

1.8 Structural Properties of Petri Nets

In the following sections, we will introduce several important structural properties of Petri nets. The basic difference between behavior properties and structural properties is the dependency on initial marking. Structural properties are those properties that are independent on initial marking but dependent on topological structures of Petri nets. That is, those properties remain true no matter what initial marking is or are concerned with certain firing sequence from some initial marking. These properties can often be identified and categorized by means of the incidence matrix A and its associated homogeneous equations or inequalities.

1.8.1 Structural Liveness

Definition: A Petri net PN is considered to be structurally live if there exists a live initial markingfor PN.

It can be easily shown that a MG is structurally live.

1.8.2 Controllability

Definition: A Petri net is controllable if any marking is reachable from any other marking. That is,for $\forall M$, $\exists M'$ such that $M \in R(M')$.

Theorem: If a Petri net PN with m places is completely controllable, then we have

$$\text{Rank } A = m.$$

1.8.3 Structural Boundedness

Definition: If a Petri net PN is bounded for any finite initial marking M_o, then it is said to bestructurally bounded.

Theorem: A Petri net PN is structurally bounded iff there exists an integer column vector y>0(i.e., each component of the vector y is greater than zero), such that $A^T y \leq 0$, where A is an incidence matrix.

Corollary: A Petri net PN is structurally unbounded iff there exists an integer column vector x?0 (i.e., each component of the vector is nonnegative), such that $A^T x = \Delta M > 0$, where $\Delta M(p) > 0$ (i.e., the pth entry of $\Delta M > 0$), p P AT is the transpose of A.

1.8.4 Conservativeness

Theorem: A Petri net PN is said to be (strictly) conservative if there exists a (positive) nonnegative integer y(p) for every place p such that the weighted sum of tokens, $M^T y = M_O{}^T y$ is a constant, for all $M \in R(M)$.

Theorem: A Petri net is (strictly) conservative iff there exists an (positive) nonnegative in eger column vector y such that Ay = 0.

1.8.5 Repetitiveness

Definition: A Petri net PN is (partially) repetitive if there exists an initial marking and a firing sequence such that any transition in can appear infinite times.

Theorem: A Petri net PN is (partially) repetitive iff there exists an positive (nonnegative) integer such that $A^T x \geq 0$,

1.8.6 Consistency

Definition: A Petri net is consistent if there exists a firing sequence a such that the initial markingM_o can be recovered (starts from M_o and back to M_o),and this firing sequence contains every transition
in the net at least once.

Theorem: A Petri net PN is consistent iff there exists a positive integer vector x > 0 such that $A^T{}_x=0$.

It is apparent that conservativeness is a stronger property than structural boundedness, whereasconsistency is a special case of repetitiveness. Also, if a Petri net is structurally bounded and structurally live, then it is both conservative and consistent. Table1-2 summarizes these structural properties.

Properties	Necessary and Sufficient conditions
Structurally Bounded	$\exists$ y>0, Ay $\leq$ 0 (or Not $\exists$ x > 0, A^Tx> 0)
Conservative	$\exists$ y>0, Ay=0 (or Not $\exists$ x > 0, A^Tx> 0)
Repetitive	$\exists$ x > 0, A^Tx> 0
Consistent'	$\exists$ x > 0, A^Tx = 0 (or Not $\exists$ y,Ay>0)

Table 1-2

1.8.7 P-invariant, T-invariant, support and T-condition

Definition of P-invariant: An integer y vector is called a P-invariant if Ay= 0.

Theorem: An m-vector y is an P-invariant iff $M^T{}_y = M_o{}^T$for any M_o and any $M \in R(M_o)$

Definition of T-invariant: An integer x vector. is called a T-invariant if A^Tx =0. Let Z be a set of places, y_i the i-th component of the y vector and $Y(Z) = \sum_{p_i \in Z} Y_i$

Theorem: An n-vector $x \geq 0$ is T-invariant iff there exists a initial marking M_o and a firing sequence σ starting from Mo back to Mo where its firing count vector equals to x.

Definition: The set of places p(transition t such that y(p)>0, x(t)>0) for P-invariant (T-invariant)is called the support of the P-invariant(T-invariant) and is defined as ||y || (|| x ||).

Definition: A minimum support is a support and none of its proper subset is a support.

Definition: An invariant y is said to be minimal if there is no other invariant y' such that y'(p) <y(p) for all

p. Minimal support invariant is a unique minimal invariant corresponding to the minimalsupport.

Theorem: The condition is defined as a T-condition if for any ordinary Petri net(OP), $Ay \leq 0$ iff $\forall t \in T$, $Y(\bullet t) \geq Y(t \bullet)$.

1.9 Summary

In this chapter, we provide a brief review of the fundamental knowledge for the Petri net theory. The structure of Petri nets are introduced and some examples of application are given for illustration. We also explain some important properties for Petri nets. These properties especially the structural properties, are very important concepts when we discuss the synthesis rules for Petri nets in later chapters. We also introduce the subclass of Petri nets. These subclasses enhance the analytical power but sacrifice some modeling capability. Best of all, readers will see that the knitting technique that will be introduced starting from chapter 3 through this book covers even more than asymmetric nets does and automatically reserves all those well behavior properties as well.

9. REFERENCES

[BER 85] Berthelot, G., "Checking properties of nets using transformations," In G. Rozenberg (ed.), Advances in Petri Nets 1985, Springer-Verlag, pp. 19-40.

[BER 86]_, "Transformations and decompositions of nets," LNCS, Advances inPetri nets, pp. 359-376, Part I, 1986.

[CHA 92] Chao, D.Y and Wang, D.T., A Reduction algorithm of Petri net, Proc. Int'lComp Symp, Taichung, Taiwan, Dec. 13-15, 1992, pp.16-23.

[CHA 93a] _,Zhou, M.C., and Wang, D.T., "Extending knitting technique to Petri net Synthesis of automated manufacturing systems", The Computer Journal, Oxford University Press, Vol. 37, No. 1, Jan. 1994, pp. 67-76.

[CHA 93b] _, Zhou, M.C., and Wang, D.T., Multiple-weighted marked graphs, Proc. IFAC 1993 World Congress, Sydney, Australia, 18-23 July, 1993.

[CHA 93c] _ and Wang, D.T., "A Synthesis technique of general Petri nets," Journal of Systems Integration, Vol. 4, No. 1, Feb. 1994, pp. 67-102.

[CHA 93d] _ and Wang, D.T., "Iteration bounds of single-rate data flow graphs for concurrent processing", IEEE Tran. Circuits and Systems, Vol. 40, CAS-1, September, 1993, pp. 629-634.

[CHA 93f] _,"A CAD tool for constructing large Petri nets", revised for IEEE Trans.SMC.

[CHA 93g]_, "A Linear algebra approach to validate knitting rules for Petri net syn-thesis and invariants," submitted.

[CHA 93h] _ and Wang, D.T., "XPN-FMS: A Modeling and simulation software for FMS Using Petri nets and X window", accepted by Int'l. J. FMS.

[CHA 94a] _ and Wang, D.T., "The knitting technique and its application to commun-ication protocol synthesis", MASCOTS'94, Durham, NC, Jan. 31 - Feb. 2, 1994, pp. 234-238.

[CHA 94b]_ and Wang, D.T., "An Interactive tool for design, simulation, verification,and synthesis of protocols", Software-Practice and Experience, Vol.24,No. 8, pp. 747-783, Aug. 1984.

[CHA 94c]_ and Wang,'D.T., "The structural matrix and reduction algorithm of knit-ting technique for Petri nets," submitted.

[CHA 94d]_ and Wang, D.T., "Application of knitting technique and structural matrix to deadlock analysis and synthesis of Petri nets with sequential exclusion," submitted.

[CHA 94e]_ and Wang, D.T., "Petri Net Synthesis and Synchronization Using Knit-ting Technique," 1994 IEEE Int'l Conf. SMC, San Antonio, TX, October2-5, pp. 652-657.

[CHA 94i]_ and Wang, D.T., "Synchronized Choice Ordinary Petri Nets", submitted.

[CHE 93]Chen, Y., Tsai, W.T., and Chao, D.Y., "Dependency analysis - A Com-positional technique for building large Petri net", IEEE Trans. on Parallel and Distributed Systems, Vol 4, No.

[CIN 85] F. De Cindio, G. De Michelis, C. Simone, "Giving back some freedom to system designer," System research, No. 2.4, 1985.

[DAT 84] Datta, A., and Ghosh, S., "Synthesis of a class of deadlock-free Petri nets," Journal of ACM, pp. 486-506, Vol. 31, No. 3, 1984.

[DAT 86] Datta, A. and Ghosh, S., "Modular Synthesis of Deadlock-Free Control Structures," LNCS 241, Foundation of Software Technology and Theoretical Computer Science, pp. 288-318, 1986.

[DON 83] Dong, T., "The Modeling, Analysis, and Synthesis of Communication Pro-tocols," Ph.D Thesis, Computer Science Division, EECS, UCB, 1983.

[ESP 91a]Esparza, J., and Silva, M., "On the analysis and synthesis of free choicesystems," LNCS, Advances in Petri Nets 1991, Springer-Verlag, pp.243-286.

[ESP 91b]_ and Silva, M., "Circuits, handles, bridges, and nets," LNCS, Advances inPetri nets, 1991, Springer-Verlag, pp.210-242.

[HYU 82]Hyung, L-K and Favrel, J., "Analysis of Petri nets by hierarchical reduc-tion and partition," in LASTED Modelling and Simulation. Zurich, Switzerland: Acta Press, 1982, pp. 363-366.

[HYU 85] and Favrel, J., "Hierarchical reduction method for analysis and decompo-sition of Petri nets," IEEE Trans.*Syst.*,Man, Cybern., Vol. SMC-15, Mar. 1985, pp. 272-280.

[HYU 87] - "Generalized Petri Net Reduction Method" IEEE Trans. Syst., Man, Cybern., Vol. SMC-17, No. 2, 1987, pp. 297-303.

[JEN 91] Jeng, M.D., and DiCesare, F., "A modular synthesis techniques for Petri nets," 1992 Japan-USA Sym. on Flexible Automation,,pp. 1163-1170.

[JOH 81]Johnsonbaugh, R., and Murata, T., "Additional method for reduction andexpansion of

marked-graphs," IEEE Trans. Circuits Syst., Vol. CAS, Oct.1981, pp. 1009-1014.
[KOH 91]Koh, I., and DiCesare, F., "Transformation methods for generalized Petrinets and their applications in flexible manufacturing systems," IEEE TransSystem, Man, and Cybernetics, 1991. Vol. 21(6), pp. 963-973.
[KRO 86]Krogh, B.H., and Beck, C.L., "Synthesis of place/transition nets for simu-lation and control of manufacturing Systems," proc.4th IFAC/IFORSSymp. Large Scale System, Zurich, 1986.
[KWO 77]Kwong, Y.S., "On reduction. of asynchronous systems." Theoret. Comput.Sci., Vol. 5,1977, pp. 25-50.
[LEE 85]Lee, K.H., and Favrel, J., "Hierarchical reduction method for analysis anddecomposition of Petri nets," IEEE Trans. Systems, Man, and Cybernetics, SMC-15, 2, 1985, pp. 272-280.
[Lip 76] Lipton, R., "The reachability problem requires exponential space," Dept. of CS, Yale University, Report No. 62, January, 1976.
[MUR 77] Murata, T., "Circuit theoretic analysis and synthesis of marked graphs,"IEEE Trans. Circ. and*Sys.*, CAS-27, 1977, pp. 400-405.
[MUR 80a]_, "Synthesis of decision-free concurrent systems for prescribed resourcesand performance," IEEE Trans.. Software Engineering, SE-6, NO. 6, 1977,pp. 400-405.
[MUR 80b] _, and Koh, J.Y., "Reduction and expansion of live and safe marked-graphs," IEEE Trans. Circuits Syst., Vol. Cas-27, Jan. 1980, pp. 68-70.
[MUR 84] _, "Modeling and Analysis of Concurrent Systems," in Handbook of
Software Engineering, edited by C. Vick and C.V. Ramamoorthy, Van Nostrand Reinhold, 1984, pp. 39-63.
[MUR 86]Murata, T., Komoda, N., and Matsumoto, K., "A Petri net based controllerfor flexible and maintainable sequence control and its applications in fac-tory automation," IEEE Trans. on Industrial Electronics, IE-33,1986, pp.1-8.
[MUR 89]-, "Petri nets: Properties, analysis and applications," IEEE Proceedings,Vol. 77, No. 4, April 1989, pp. 541-580.
[NAR 85]Narahari, Y., and Viswanadham,-N., "A Petri net approach to the modelingand analysis of flexible manufacturing systems," Annals of Operatio nsResearch, 3, 1985, pp. 449-472.
[PET 81]Peterson, J.L., petri net theory and the modeling of systems, Prentice-Hall,Englewood Cliffs, New Jersey, 1981.
[PET 94]Petriu, D.C., Archibald, C.C., and Petriu, E.M.,. "Petri Net Model of aResource Management system for a Multi-Robot Assembly Cell", Proc.IEEE SMC'93 Conf., pp. 409-414, Le Touquet, France, October17-20,1993.
[RAM 85]_, Dong, S.T., and Usuda, Y., "The Implementation of an automated proto-col synthesizer(APS) and its application to the X.21protocol," IEEETrans. on Software Engineering, No. 9, September 1985, pp. 886-908.
[RAM 86a]_, Yaw, Y. (now Chao, D.Y.)., and Tsai, W.T., " A Petri net reductionalgorithm for protocol analysis ,"'Computer Communication Review(USA), Vol. 16, No. 3, Aug. 1986, pp. 157-166.
[RAM 86b] _,Yaw, Y., Tsai, W.T., Aggarwal, R., and Song, J., "Synthesis of two-party error-recoverable protocols," Computer Communication Review (USA), Vol. 16, No. 3, Aug. 1986, pp. 227-235.
[RAM 86c] _,Yaw, Y., Tsai, W.T., Aggarwal, R., and Song, J., "Synthesis and per-formance evaluation of two-party error-recoverable protocols, COMSAC Symp. Oct. 1986, pp. 214-220.
[SIL 85]Silva, M. Las redes de Petri en la Automatica y la Informatica, EditorialAC, 1985, Madrid.
[SUZ 83]Suzuki, I. and Murata, T., "A Method of stepwise refinement and abstrac-tion of Petri nets," Journal of Computer and System Sciences 27, 1983, pp.51-76.
[VAL 79] Valette, R., "Analysis of Petri nets by stepwise refinement," Journal of . Computer and System Sciences 18, 1979, pp. 35-46.
[VAL 90] Valvanis, K.S. "On the hierarchical analysis and simulation of flexible manufacturing systems with extended Petri nets," IEEE Trans. on System, Man, and Cybernetics, SMC-20, 1, 1990, pp. 94-100.
[VIL 88] Villarroel, J., and Martinex, L.J., and Silva, M., "GRAMAN: a graphic system for manufacturing system design,." IMACS Int. Symp. on Sys. Model. & Simul. (SMS'88), Cetraro, Italy, 1988.
[WAN 94] Wang, D.T. and Chao, D.Y., Extension of Knitting Technique to Petri Net Synthesis and Application to Flexible Manufacturing System Cell Preserving Well-Behaved Properties, Research Repoar: CIS-94-38, NJIT, Newark, NJ, also revised for IEEE Trans. SMC.
[YAW 87] Yaw, Y. (now Chao, D.Y.), "Analysis and synthesis of distributed systems and protocols," Ph.D. Dissertation, Dept. of EECS, U.C. Berkeley, 1987.

[YAW 88a]_, Ramamoorthy, C.V., and Tsai, W.T., "A Synthesis technique for design-ing concurrent systems," Second Parallel Processing Symposium, April1988, pp.143-166.
[YAW 88b]_, ,Ramamoorthy, C.V. and Tsai, W.T., "Synthesis rules for cyclic interac-tions among processes in concurrent , systems" COMSAC Symp. Oct.1988, pp. 496-504.
[YAW 89]_ and Foun, F.L., "The algorithm of a synthesis technique for concurrentsystems," 1989IEEE Int. Workshop on Petri Nets and PerformanceModels," pp. 266-276, Tokyo.
[ZHO 89a] Zhou, M.C., and DiCesare, F., "Adaptive design of Petri net controllers for error recovery in automated manufacturing systems," IEEE Trans. on Systems, Man, and Cybernetics, SMC-19, 5, 1989, pp. 963-973.
[ZHO 89b] Zhou, M.C., DiCesare, F. and Desrochers, A. A. "A top-down modular approach. to synthesis of Petri net models for manufacturing systems," Proc. of IEEE Robotics and Automation Conference, cottsdale, AZ, 1989, pp. 534-539.
[ZHO 90] Zhou, M.C. A Theory for the Synthesis and Augmentation of Petri Nets in Automation. Doctoral Dissertation, Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY, May 1990.
[ZHO 91a] Zhou, M.C., McDermott, K., Patel, P. and Tang, T., "Construction of Petri Net Based athematical Models for an FMS cell," in 1991 IEEE. Int. Conf. Systems, Man, and Cybernetics, pp. 367-372, Charlotteville, VA.
[ZHO 91b] Zhou, M.C. and DiCesare, F., "Parallel and sequential mutual exclusions
for Petri net modeling for manufacturing systems with shared resources," IEEE Trans. on Robotics and Automation, 7(4), 1991, pp. 515-527.
[ZHO 92a] Zhou, M.C., DiCesare, F., and Rudolph, D., "Design and Implementation of a Petri Net based supervisor for a flexible manufacturing system," Automatica, Vol. 28, No. 6, 1992, pp. 1199-1208.
[ZHO 92b] Zhou, M.C., DiCesare, F. and Desrochers, A.A., "A hybrid methodology for Petri net synthesis of manufacturing systems," IEEE Trans. on Robot-ics. and Automation, Vol. 8 No. 3, 1992, pp. 350-361.
[ZHO 93] Zhou, M.C., McDermott, K. and Patel, A., "Petri Net Synthesis and Analysis of a Flexible Manufacturing Systems Cell", IEEE Trans. SMC, Vol. 23, No.1, March 1993, pp. 524-531.

書名: A New Approach to Petri Net Synthesis: The Knitting Technique

著（編、譯）者: 趙玉

出版者: 趙玉
電話: 02-29387694
地址: 台北市文山區指南路二段六四號政大資管系
出版年月: 民國九十九年五月
版（刷）次: 第壹版
定價: 非賣品

GPN:　　　　　　　　　　ISBN: