1989

Analysis of the C2 system effectiveness using continous processing time

Yongsu Kwak
Iowa State University

Analysis of the C^2 system effectiveness using continuous processing time

Kwak, Yongsu, Ph.D.

Iowa State University, 1989

Analysis of the C^2 system effectiveness using continuous processing time

by

Yongsu Kwak

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major: Industrial Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa
1989

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The completion of this thesis was possible thanks to the aid and encouragement of many people. First, I am deeply grateful to Dr. Way Kuo, who provided me perspective and technical support. Dr. Kuo always incited me for a deeper understanding of the problems. It was a real blessing to have such an understanding major professor.

The rest of my committee also deserves my thanks– Dr. Herbert T. David, Dr. John Even, Professor Victor M. Tamashunas, and Professor Charles P. Cox. They all showed a lot of interest in this thesis topic and provided many helpful suggestions.

There are many people deserving my thanks. Some of them include H. J. Park, S. C. Lee, G. S. Lee, and D. S. Rim. They kept me abreast of technical developments in many areas, patiently listened to my sometimes disorganized thoughts, suggested many useful references, and provided many hours of lively discussions.

Finally, it is beyond my words to express gratitude to my family, whose unfailing love and comfort strengthened my determination to finish the thesis.

1 INTRODUCTION

1.1 Background

With the advent of extremely sophisticated computer/communication networks, a tactical organization such as a military organization has enormous amounts of information to be transferred. In addition, the development of weapon systems demands quick reacting capabilities. Therefore, the time allowed for decision-making in a tactical environment is severely constrained. Consequently, the capability of such an organization to perform its tasks, such as information transition, in a timely manner without overloading components, manpower, or machines of organization is indeed a determinant factor of organization effectiveness.

The technological advance in sensors and weapon systems have primarily driven the trend toward decentralization of the command and control (C^2) tactical decision making function and the geographical distribution of the warfare commanders for reasons of survivability, together with the need to share data over a wide variety of physical locations. On the other hand, the sheer volume of information may render it unusable without regard to the problems of interpretation imposed by delay and distortion.[1] Therefore, it is crucial that the military organization be structured in

[1]The accident of Iranian A-300 Air-bus hit by Vincennes at July 3, 1988 is given as a typical example of the distortion of information [36].

such a way as they can process efficiently only the information that contributes to its overall organization objectives. Wilson [76] discussed some of the techniques needed to process information. Stabile and Levis [64] found that partitioning the input to a single-echelon organization, such that each member receives only the information that he/she is particularly well-suited to process, may improve the performance of the organization. This is a need for a study of how the military should distribute information in the C^2 process.

This type of study is fairly tractable in the context of a military organization for several reasons. First, the organizational objectives are well-defined. Second, the tasks which must be accomplished in order to achieve this objective become well-defined as soon as the commander exercises his authority and direction. Third, in spite of its size and complexity, a military organization has many elements which perform basically identical functions, although possibly at different rates [7]. This fact makes it easier to design a system structure and also model information flows of that organization.

Efficient use of resources is another issue of a military organization. It requires that each of the components operate concurrently with other components and coordinate their activities when necessary. Also, it requires this coordination to occur asynchronously, and not at set times. Finally, the activities that each component must perform can be quite complicated and affected by many factors. Therefore, the time it takes for a component to complete a task, which will be denoted later as processing time, is usually stochastic.

Wiley [75] denoted systems that are composed of components working concurrently, coordinating their activities asynchronously, and with random processing times as Stochastic Asynchronous Concurrent (SAC) systems. One way to analyze an SAC system is to predict its performance with respect to time-related measures such as average processing rate. With these measures, different designs can be compared or existing designs modified, and their improvement or degradation in performance evaluation assessed.

Several tools currently available to analyze SAC systems are either expensive, rudimentary, or non-existent. One popularly used technique is simulation, which allows the use of relatively accurate models. Moreover, methods exist to obtain certain sensitivity measures from simulation runs [35]. Despite this, cost and time required to build models are enormous, and extremely long runs are required to assure that the simulation is in a steady-state and that the results are statistically significant. Furthermore, if the system is substantially changed, then the model must be changed and another simulation run performed at an additional expense and effort.

Another tool that has been used to study SAC systems is queueing theory [9]. However, modeling the coordination of asynchronous tasks is not easy to fit naturally into the theory of queues and must be imposed artificially by the system analyst. Moreover, queueing theory itself cannot present the characteristics of the SAC system if the system has the hierarchical structure.

A third tool that has been used to analyze SAC systems is Markov chains [49, 58]. However, Markov chains are very limited in the size of problems they can handle since modeling even a SAC system with Markov chains is very difficult and requires an extremely large number of states.[2]

[2]For instance, if the number of nodes of a system is n, then the maximum number

Meanwhile, several methods have been used to present the relationship between system components such as hierarchical relationship, operating characteristics, etc. One of the most popular methods is the block diagram. It presents mainly the hierarchical and/or operational relationships. Therefore, it is useful to understand the overall operations and characteristics of the system, but it neither presents the sequence of tasks assigned to each components, nor presents time-related factors such as task processing times of system components.

Another popular method is the flow diagram, which is widely used in computing areas to present the logical relationship between the system components. The merit of this method is that it can present the execution schedule of tasks. However, like the block diagram, it cannot present time-related factors.

The third method is the process chart. It is mainly used in manufacturing areas to present the execution schedule of component tasks. This method may also present the hierarchical structure of the system, but it cannot present time relationships between tasks followed according to continuous task execution.

The fourth method, Timed Petri Net (TPN)[3] can naturally represent the concurrency of different activities, asynchronous coordination, and deterministic or random processing time. Also, this method can present all structural and operational relationships. Therefore, it can show even very sophisticated relations of components which are not possible to present with the other methods introduced above. With this advantage, many researchers have modeled SAC systems using

of states needed to represent the system behavior is 2^n.

[3]A TPN can be classified according to properties of time measure, i.e., if deterministic time is applied, then the TPN is called Deterministic TPN (DTPN) and if time is probabilistic, then the net is called Stochastic TPN (STPN).

some version of Petri Nets or original Petri Nets. Most of these models deal with either communication protocol [17, 51, 58] or computer systems [12, 29, 43, 56]. Petri Nets has also been used to model production systems [8, 18, 27] and organization of decision-makers in Bejjani [5], Boettcher [6], Levis [41], Stabile and Levis [64], Tabak and Levis [67], and Wiley [75].

1.2 Objectives

Today's advanced electronic technology, especially computer technology, is increasing the possibility of receiving intelligence directly from sensors, to control when and what weapons to launch against the enemy. Because of this, the commander's role is gradually confined to the strategic deployment of resources. However, those situations where the speed of events is beyond human management will not be realized in the near future, although the development of those techniques support the decision-makers in managing battles; hence, the speed of events is faster than before. In other words, the decision-makers are essential components of such a tactical organization. Despite this fact, the present military organizations need a model for a command and control system [5] because of following reasons: (1) existing command systems are primarily concerned with target acquisition and tracking, and weapon direction and control. They do not offer much assistance in higher-level decision processes such as planning. An evolutionary approach to the design of command aids, involving the adaption and refinement of existing systems, is therefore inappropriate; (2) a lack of scientific understanding of the functions carried out within a command system frustrates progress with the design; (3) doctrines needed in performing missions exist, but no understanding for the officer in tactical command nor for his subordinates to perform their command tasks; and (4) lack of knowledge for plan formulation and how a mission should be monitored.

Moreover, despite abundant research in this area, there is no adequate foundation for a theory of command and control, and hence, no guiding principles for system design and evaluation [73].

Therefore, in this thesis, a generic C^2 system will be set up and modeled by using Stochastic Timed Petri Net formalism. Then, a generalized technique for the evaluation of system effectiveness will be analyzed by the following ways.

First, as mentioned previously, a C^2 system has common elements which perform identical missions. Thus, it might be possible to find a functional model of the C^2 process which is generally applicable and can describe the processes taking place in any command and control subsystem within the overall command system. If all such subsystems are considered as instances of such a general functional model, then any particular command organization can be represented by the interconnection of a separate C^2 process to represent a subsystem with a declared responsibility within this structure.

Second, in order to analyze the C^2 system, Stochastic Timed Place Petri Net (STPPN)[4] formalism will be applied, because this system can be regarded as a SAC system. Problems for predicting system states and measures of effectiveness (MOEs) will be used in evaluating this system. To resolve these system state problems, conversion of the system to an STPPN will be performed.[5] Then, to

[4]STPNs can be categorized as Stochastic Timed Place Petri Nets (STPPNs) and Stochastic Timed Transition Petri Nets (STTPNs), according to locations placed time measure, places, or transitions, respectively. Thus, their evolution processes are not equivalent.

[5]A C^2 system is a type of decision-making process and thus, the resulting STPPN

obtain MOEs for the performance evaluation of the underlying system, simulation techniques will be applied to the STPPN. As a result of simulation to the net, the following measures can be obtained: (1) the average cycle processing time to perform a system task; (2) the maximum average processing time to perform a C^2 process; and (3) the dynamic response time.

Finally, a good system must be able to cope with shocks generated by the environment. Hence, in this thesis, dynamic behavior of the system, such as how the system structure is changed and how the additional tasks (or burden) due to shocks are assigned to the changed structure when shocks happen, will be discussed.

1.3 Literature Review

1.3.1 Timed Petri Nets

Several studies for Timed Petri Nets have been done so far. All of these works are extensions of the original Petri Nets. [6]

This literature review is divided into two sections according to the type of processing times that the algorithms in the cited articles can handle: deterministic or random.

The first study on Timed Petri Nets was represented by Ramchandani [57], who associated deterministic processing times with transitions and obtained exact transition firing rates for Decision-Free Timed Petri Nets,[7] but only approximate

model may be used for any decision-making process.

[6]There are several introductory articles and texts to original Petri Nets. The most comprehensive one is Petri Net Theory and the Modeling of Systems by J.L., Peterson [55]. Other include Agerwala [1], Genrich et al. [23], Genrich and Stankiewicz-Wiechno [24], and Reisig [59].

[7]Decision-Free Timed Petri Nets are a subclass of STPNs which transitions are

bounds on the transition firing rates for more general Timed Petri Nets. His work was slightly generalized by Sifakis [62], who associated deterministic processing times and essentially obtained the same results as Ramchandani, but who handled more general Petri Nets with multiple arcs between nodes. Sifakis [61] showed the distinction between associating processing times with transitions or with places was not important since one type of Timed Petri Net can be converted into the other. However, it is effective for the case of the deterministic processing time, not for the random processing time. Ramamoothy and Ho [56] showed how to obtain the same results given by Sifakis and Ramchandani. Cohen et al. [13] showed that Decision-Free Timed Petri Nets with deterministic processing times can be analyzed using (max, +) algebra results. Finally, Merlin and Farber [47] discussed TPNs where a time threshold and maximum delay were assigned to each transition. This was done to allow the incorporation of timeout into a protocol model.

Some of the works discussed above can be extended to analyze Timed Petri Nets with random processing times by replacing these times by their expected values. However, the results obtained in this way provide only very loose approximations of the average firing rates. Several researchers have tried to remedy this situation by converting the Timed Petri Net into an equivalent Markov chain and then analyzing the resulting Markov chain. Zuberek [78] was the first to perform this transformation and was able to analyze STPNs which only allow very simple decision rules based on independent probabilities. Razouk and Phelps [58] extended Zuberek's work to STPNs that can model time-outs situations where the completion of one activity may disable others, and also slightly more complex decision situa-

never in conflict. Therefore, no decision rule is needed.

tions. The decision rules are still based on independent probabilities. Both of these articles, however, fail to show that the resulting Markov chain has a well-defined steady state probability distribution, and their procedures are applicable to only very small problems.

Molloy [49] solved somewhat larger problems[8] by associating exponential processing times with transitions and by specifying a decision rule which stated that the transition whose processing time terminates first would fire. Marson et al. [43] extended Molloy's results to manage transitions with zero processing times.

As indicated before, the main weakness of all works mentioned above is that they need to construct an equivalent Markov chain modeling the evolution of the marking of the net to find the performance measures of interest.

Zuberek [78], and Jantzen and Valk [38] dealt with decision-conflict situations by introducing inhibitor arcs to the original Petri Net formalism (see Figure 1.1). Their works were based on the assumption that the underlying conflict situation was resolved by the priorities which occurred from inhibitor arcs. However, if decisions for conflict sets are uniquely determined by external factors, there may be situations which cannot be solved with inhibitors, i.e., the idea is limited to very specific cases.

1.3.2 Discrete Event Dynamical Systems (DEDSs)

Some of the literature pertinent to the study of STPNs bear the heading Discrete Event Dynamical Systems (DEDSs). Basically a DEDS is a collection of

[8]The size of the problem that can be handled actually depends on the number of markings that can be reached. This, in turn, depends on the number of places, the number of transitions, the number of tokens that can be in a place at one time, and other factors. These numerous factors make it hard to specify the size of the problem that can be solved.

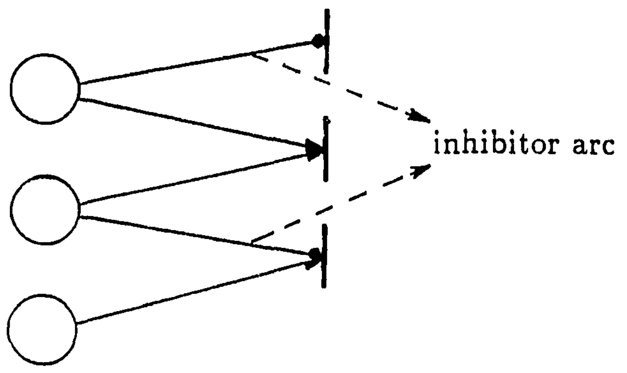


Figure 1.1: Example of Petri net with an inhibitor arc

servers and finite queues connected by a directed graph. Users move around the graph. When a user is finished with one server, it moves to another queue according to a specified destination rule.[9] If users are identified with tokens, servers with places, and finite queues with STPN models (see Figure 1.2), then the similarity between DEDSs and STPNs becomes evident. Whether the two types of systems are equivalent or one is a subset of the other can be decided by the exact formulation given by the different authors.

Most of the literature concerning DEDSs deals with their use in simulation. Recently, several researchers have extended this simulation work to include sensitivity measures. Articles on this subject include Ho and Cassandras [35], Cao and Ho [9], and Suri [66].

[9]Terminology from Ho and Cassandras [35].

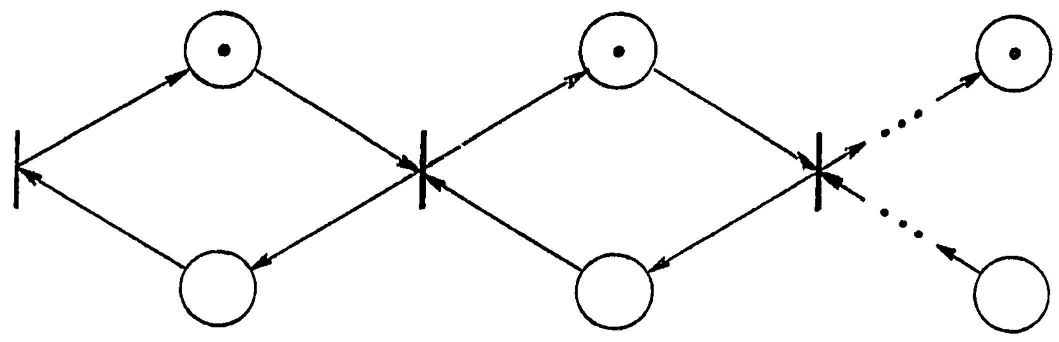


Figure 1.2: An STPN model of a finite first-in/out queue

2 REVIEW OF STOCHASTIC TIMED PLACE PETRI NET (STPPN) THEORY

2.1 Introduction

In this chapter, STPPN theory is more formally presented. A general definition of STPPN, conditions of well-formed STPPN, and subclasses of STPPN, Conflict-free STPPN and Free-choice STPPN are discussed. The main purpose of Section 2.2 is to formally define STPPN concepts. Additionally, some technical assumptions, which can be easily satisfied by expanding or shrinking the STPPN[1] to ensure that the STPPN is well-formed, are provided. It is also shown that the states of STPPN consist of the positions and status of tokens[2], plus the time left to serve all active tokens and how state transition is made. From this, a simulation algorithm is derived to generalize token status in the given STPPN. In Section 2.3, the STPPN requirements for the STPPN to have a well-formed structure: safeness, liveness, and strong connectedness are defined. Liveness is a condition for all transactions to fire regularly and infinitely. Safeness[3] ensures that each place in the net always

[1]Adding or deleting places and transitions according to the expansion theory in Genrich et al. [23] and Peterson [55].

[2]Active or inactive

[3]Safeness is, in other words, called 1-boundedness, since the upper bound of the number of tokens is one [55].

has at most one token, i.e., zero or one token in a place. Strong connectedness is the last condition to ensure the existence of sets of direct paths.

At this point in this thesis, the analysis of STPPNs is limited to safe, live, and strong connected STPPNs. Restricting the study to STPPNs having these conditions aids processing order placement and guarantees that the STPPNs dealt with have well-formed structures which make their long-term behavior analysis much easier.

Finally, Section 2.4 introduces two STPPN subclasses, Conflict-free STPPN and Free-choice STPPN.

2.2 General Definitions of STPPN

2.2.1 Basic definitions

Stochastic Timed Place Petri Net (STPPN) represents its status by static and dynamic components. The static component is the graph with related information such as processing times and operation rules, which does not change with time. The dynamic component is token statuses, which contain information such as their locations, states, and remaining waiting time before moving to the next position, which changes with time.

2.2.1.1 Graph The graph of STPPN can be defined by a quadruple,

$$N = (P, T, A, M_o)$$

where $P = p_1, p_2, \cdots, p_n$ is a finite, nonempty set of places, $T = t_1, t_2, \cdots, t_m$ is a finite, nonempty set of transitions, A is a set of input and output arcs, I and

O, respectively, such that $I \in (P\text{x}T)$ and $O \in (T\text{x}P)$, and M_O is an n-component vector of non-negative integers and indicates the initial marking of the net. By the definition, the fact that P, T, A, and M_O have no common elements is easily obtained.

2.2.1.2 Tokens and their statuses At any given time in the evolution of the STPPN, the number of tokens in the places can be specified by a vector,

$$M = [\, m_1\; m_2\; \cdots\; m_i\; \cdots\; m_n \,]^{-1}$$

where m_i is a non-negative integer which indicates the number of tokens in place p_i, n is the number of places, and M is called the *marking* of the net. Also, if the number of tokens in a place is strictly positive, then the place is called *marked place*. When a token arrives at a place, its status is inactive until the token is ready to leave the place, at which time the token becomes active.

2.2.1.3 Processing time In addition to tokens, a processing time distribution is assigned to each place. The processing time for place p_i is denoted as u_i. It indicates the amount of time that each token entering the place p_i must stay until its state changes to active. In this thesis, the processing times are assumed to be exponentially distributed and independent from one token to the next, independent over places, and independent of decision rules.

2.2.1.4 Transition firing Tokens travel the net by transition firings. When a transition fires, it removes one active token from all of its input places and adds one inactive token to all of its output places. A transition is defined as *enabled* if each of its places contains active tokens or as *potentially enabled* if each of its places contains an inactive token or if some of its places contain inactive tokens and the rest of them contain active tokens.

2.2.1.5 Initial condition To evolve an STPPN, the initial state of the net should be specified. To this effect, it is assumed that the initial marking of the net plus the status of each token at time zero is given. As will be seen later, the mentioned initial conditions can be used to implement the rules of operation.[4]

2.2.1.6 Conflict set Two or more transitions are said to be in conflict if a firing transition disables the others. As shown in Figure 2.1, transitions t_1 and t_2 are in conflict, since the firing of t_1 will disable t_2, and vice versa. This example is simple because all transitions become enabled at the same time. Another example shown in Figure 2.2 represents a more complicated conflict situation, since transitions may become enabled at different times, transitions t_1 and t_2. Note that according to the above definition, a transition is always in conflict with itself and its firing will disable itself. All transitions which can potentially be in conflict are assumed to be divided into mutually exclusive conflict sets [58].

2.2.2 Rules of operation

2.2.2.1 Decision rule Conflict situations are essential points in the evolution of an STPPN where a decision must be made that affects the future behavior of the STPPN. To resolve these conflicts, a decision rule must be specified which is assigned to every conflict set or equivalently to one of the places connected to

[4]There are several ways in which the initial conditions for a net can be specified.

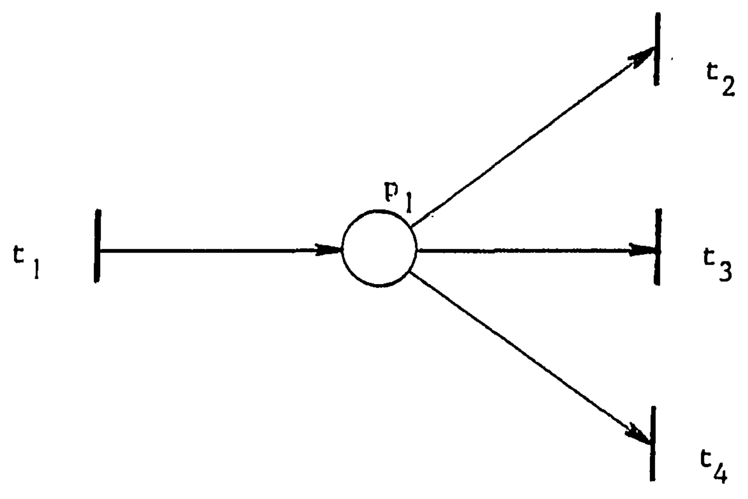


Figure 2.1: An STPPN with a simple conflict

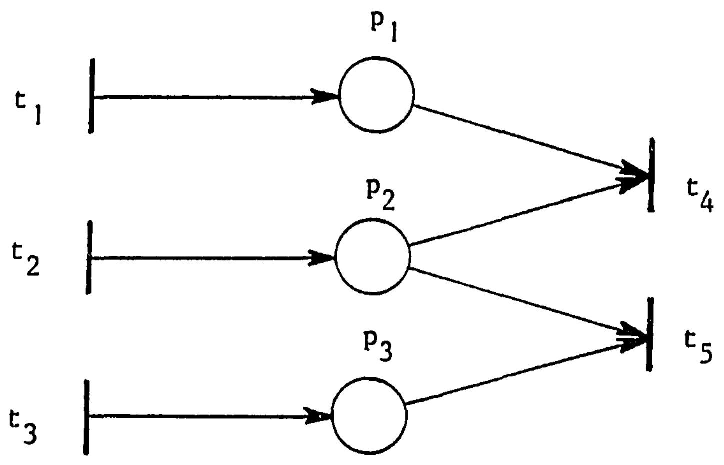


Figure 2.2: An STPPN with a more complicated conflict

each transition in the conflict set.[5] If several transitions in a conflict set are simultaneously enabled at any given point, a transition is chosen and then it may fire.

In general, this decision rule depends on the evolution state of the STPPN.[6] Also, since relative time at which tokens in the input places of the conflicting transitions become active will be used as the ordered index instead of the real time, the scope of decision rules of this thesis should be restricted to relative times.[7] If a decision rule is allowed to depend on the full state, then causality might be violated, since with the ordered index, the transition firings do not occur in the same order as if the STPPN were operating in real time. This scope of decision rules will allow to model priority decision-making and other time-dependent random events, or some combination of the two. This discussion is illustrated next with a pair of decision rule examples.

One conflict situation examined is that shown in Figure 2.1. In the STPPN, the transitions t_2, t_3, and t_4 are in conflict. As noted before, all of these transitions become enabled at the same time. A possible decision rule to resolve this conflict is to assign probabilities or priorities which might reflect the outcomes of independent random events. Then, every time a token becomes active in p_1, a transition is selected using the probability rule according to priorities preassigned, and the selected transition may fire.

The other conflict situation is shown in Figure 2.2. As mentioned before, this conflict situation is more complicated than the previous one since more than one

[5]A singleton conflict set has a trivial decision rule.
[6]STPPN states are discussed later in this chapter.
[7]A relative time will be formally defined later.

token is involved. Thus, for example, if the tokens in p_1 and p_2 are active, but not p_3, then t_4 is enabled and t_5 is not. Hence, the transition t_4 are not in conflict and may fire. If the tokens in p_1, p_2, and p_3 are all active, however, then both transitions t_4 and t_5 are enabled and in conflict. If the time the token in p_i arrives is τ_i, then the decision rule might specify that the times the tokens in p_1 and p_3 have been waiting should be compared. Hence, the transition that would consume the oldest token should fire. This could be done by specifying the following decision rule:

$$\text{If } \tau_1 \leq \tau_3, \begin{cases} \text{choose } t_4, \\ \text{otherwise, choose } t_5 \end{cases} .$$

Note that in case of a tie, transition t_4 fires first, and thus it has a slightly higher priority than t_5. Instead of the priority, a probability rule to resolve the tie can be specified. Of course, if active tokens only reside in p_1 and p_2, then t_4 fires regardless of the time t_3 last fired.

2.2.2.2 Transition selection rules As mentioned before, only enabled transitions may fire. If there is a single enabled transition at any given time, then it fires instantly. If several transitions belonging to the same conflict set are enabled, then the decision rule is applied for that conflict set to select one of those transitions and the selected transition fires instantly. Note that the transition selection rules just presented select a single transition to fire. As mentioned before, this firing consumes an active token from each input place and adds an inactive token to each output place. If the processing time of one of these output place is zero, then the token becomes immediately active and thus may instantly make other transitions enabled. It is only at this point that the transition selection rules are used to select the next transition to fire.

2.2.2.3 Firing time Given the above discussion, when there are several transitions enabled at the same time, they always fire in a particular order. Since each of these firings takes place instantly, the firing times of all these transitions are equal. This situation is a reflection of the fact that STPPNs have two notions of time associated with them. One is the real time as measured with respect to real time. The other is the event time which is measured by the number of transitions that have fired. There is generally neither a one to one nor even one to many relationship between them. The transition selection rules discussed above structure event time so that ambiguities in real time can be resolved in a consistent manner.

2.2.3 Simulation of an STPPN

2.2.3.1 Input specifications To more easily understand the operations of an STPPN with conflict situations, the shared resource example in Figure 2.3 will be discussed and simulated. First, however, two conditions should be specified, i.e., inputs (initial marking and conditions) and processing time for each transition. It is assumed that the initial marking places a token p_1, p_2, p_3, and p_4. For simplicity of the example, it is assumed that all markings are initially active and all processing times are deterministic and given by the following constraints:

$$\begin{aligned}
u_1 = u_2 = \; & u_3 = u_4 = 0 \\
u_5 = u_6 = \; & u_7 = 1 \\
u_8 = u_9 = \; & u_{10} = 4
\end{aligned}$$

In this example, only one non-trivial conflict set consisted of transitions t_1, t_2, and t_3. Since all transitions have p_1 as the common input, all transitions included in a conflict set must be mutually disabling.

Last, the decision rule mentioned previously should be specified which decides a firing order when transitions t_1, t_2, and t_3 are in conflict. A token in p_1 represents a shared resource as being active and a token in p_2, p_3, and p_4 represents demand sources, respectively, waiting their turn to use the resource. With this interpretation about the example of STPPN, it is necessary to determine the order to use the shared resource in accordance with the length of the waiting time of each demand source. That is, the shared resource will be used by the demand source waiting the longest time. In case of a tie, t_1 has the highest priority and t_3 has the lowest priority. Now, since all inputs have been specified, the STPPN example can be simulated. Figure 2.4 shows the first few steps of this simulation, and explains the implementation of the operation rules.

2.2.3.2 State of an STPPN In order to analyze STPPN behavior, it is necessary to determine what kind of information should be recorded to predict the future behavior of an STPPN state. Needed information is the marking of the net, token statuses (active or inactive), and how much time remains for that specific token to move forward if a token status is active.

As mentioned above, one component of the state consists of the position and status of every token. To keep track of this component, let vectors $\vec{M_a}$ and $\vec{M_i}$ denote the marking of active tokens and the marking of inactive tokens, respectively. The other component of the state is the time left for each active token to move out.

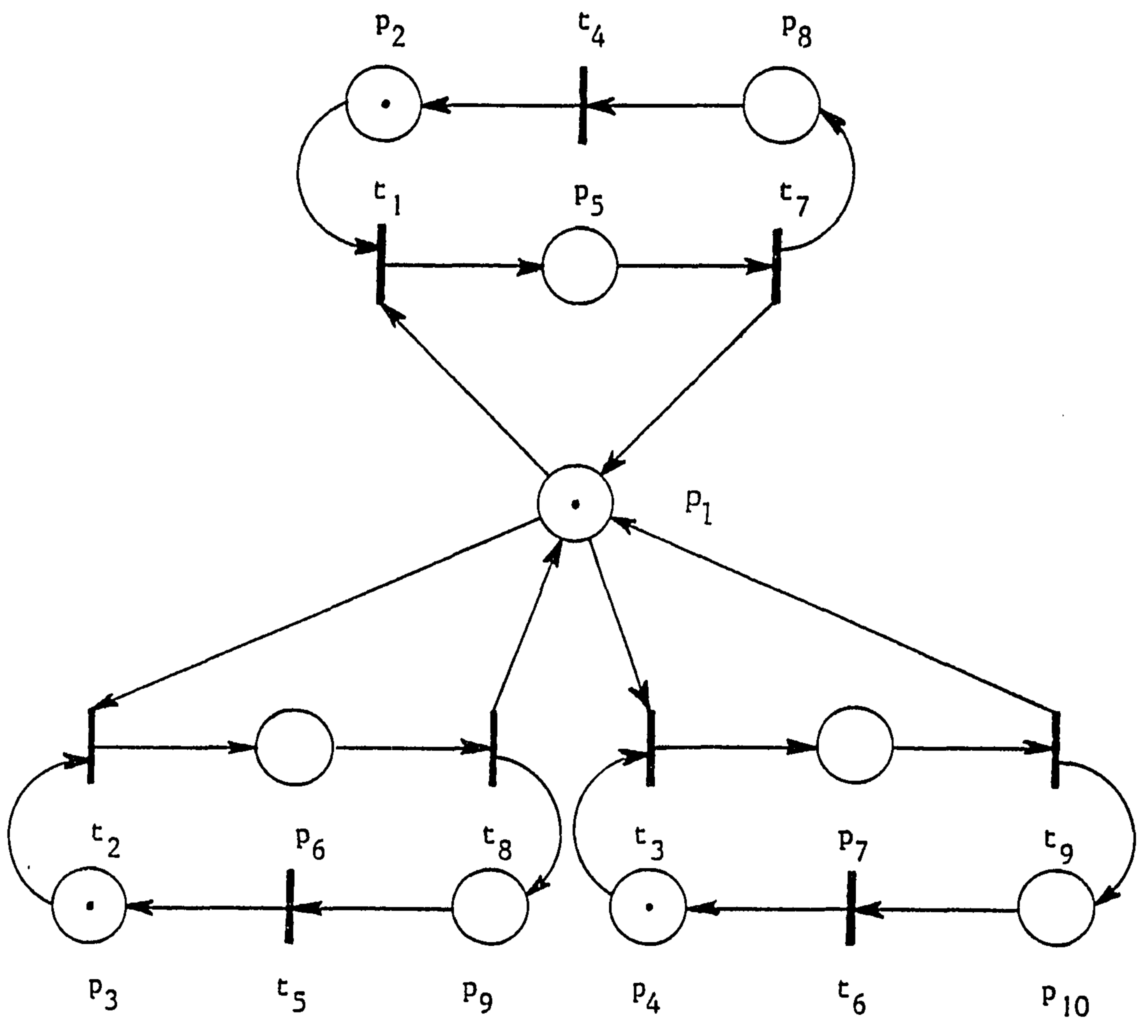
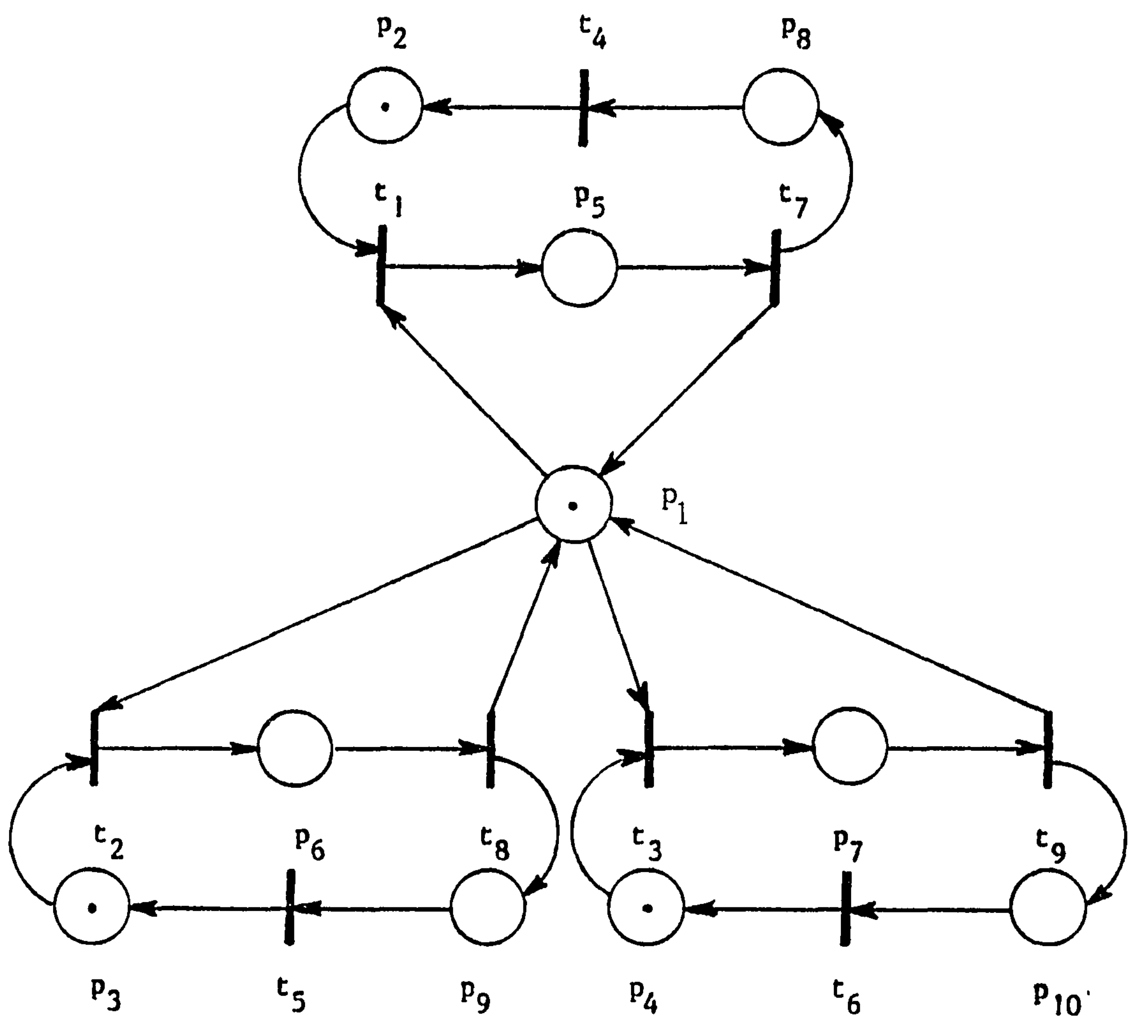


Figure 2.3: The shared resource example of an STPPN

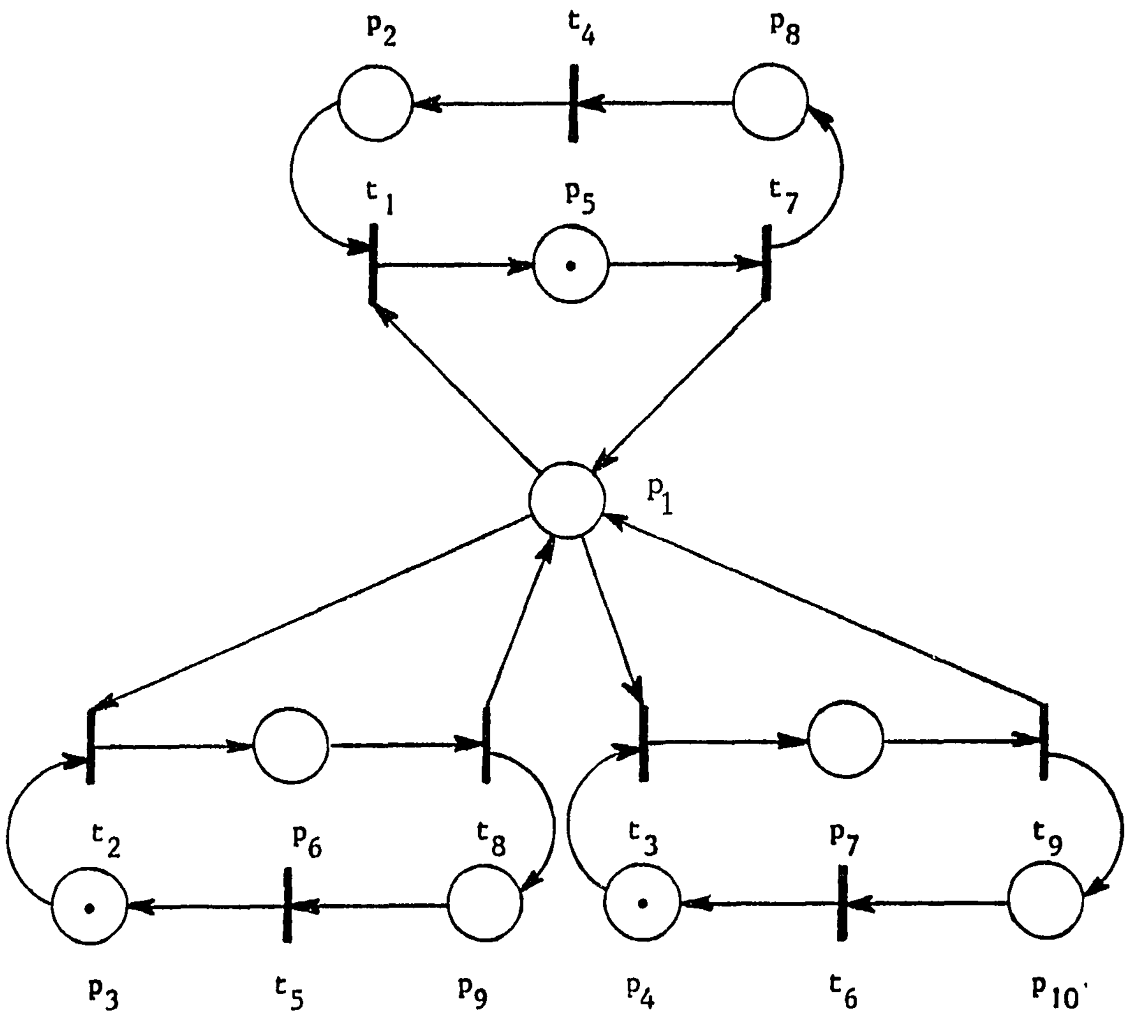


(a) Time = 0

: Token in p_1, p_2, and p_4 is initially active.

: Decision rule selects t_1.

Figure 2.4: States of the STPPN



(b) Time $= 0^-$

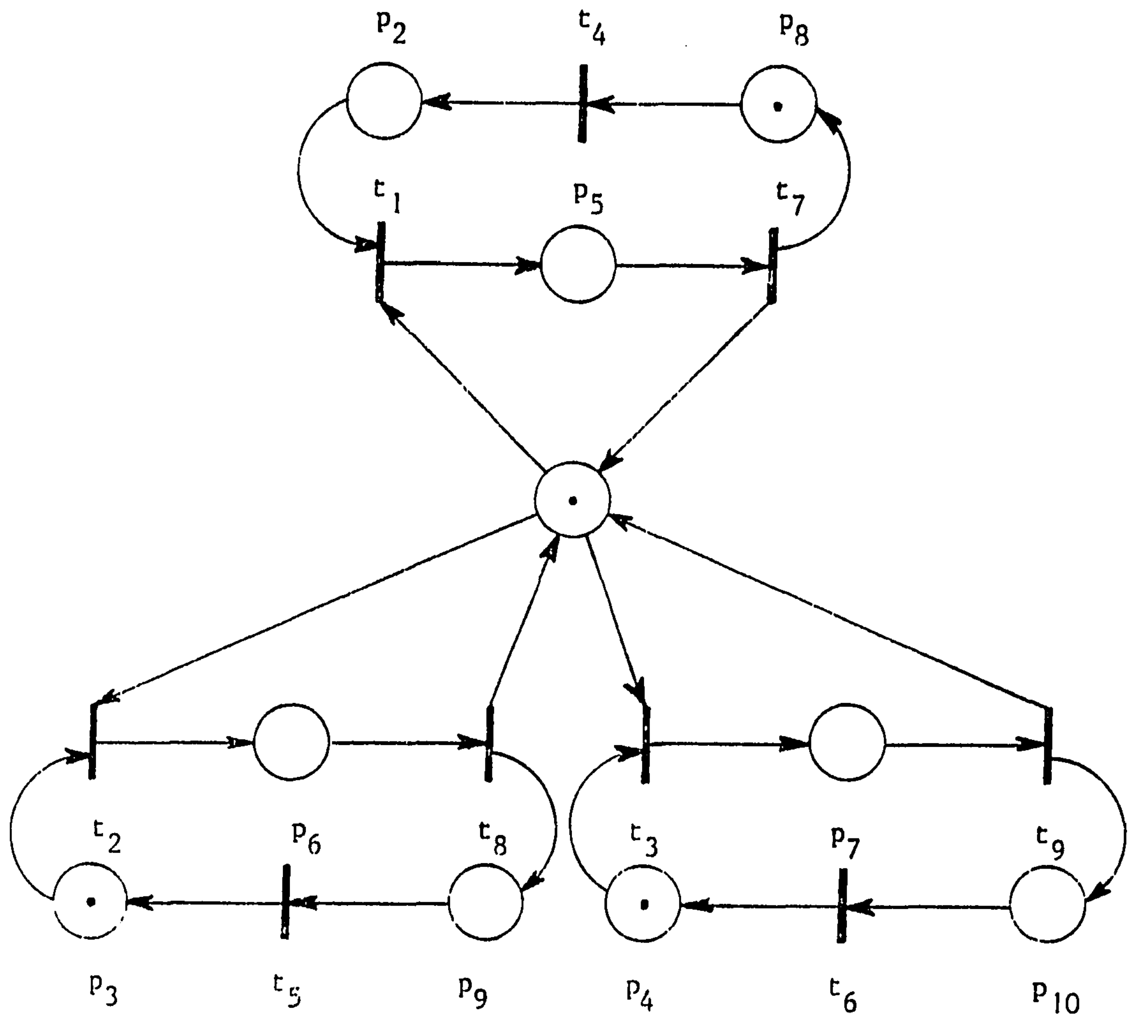
: Transition t_1 fires.

Time = 1

: Token in p_5 completes its processing and becomes active.

: Transition t_7 becomes enabled.

Figure 2.4: (Continued)

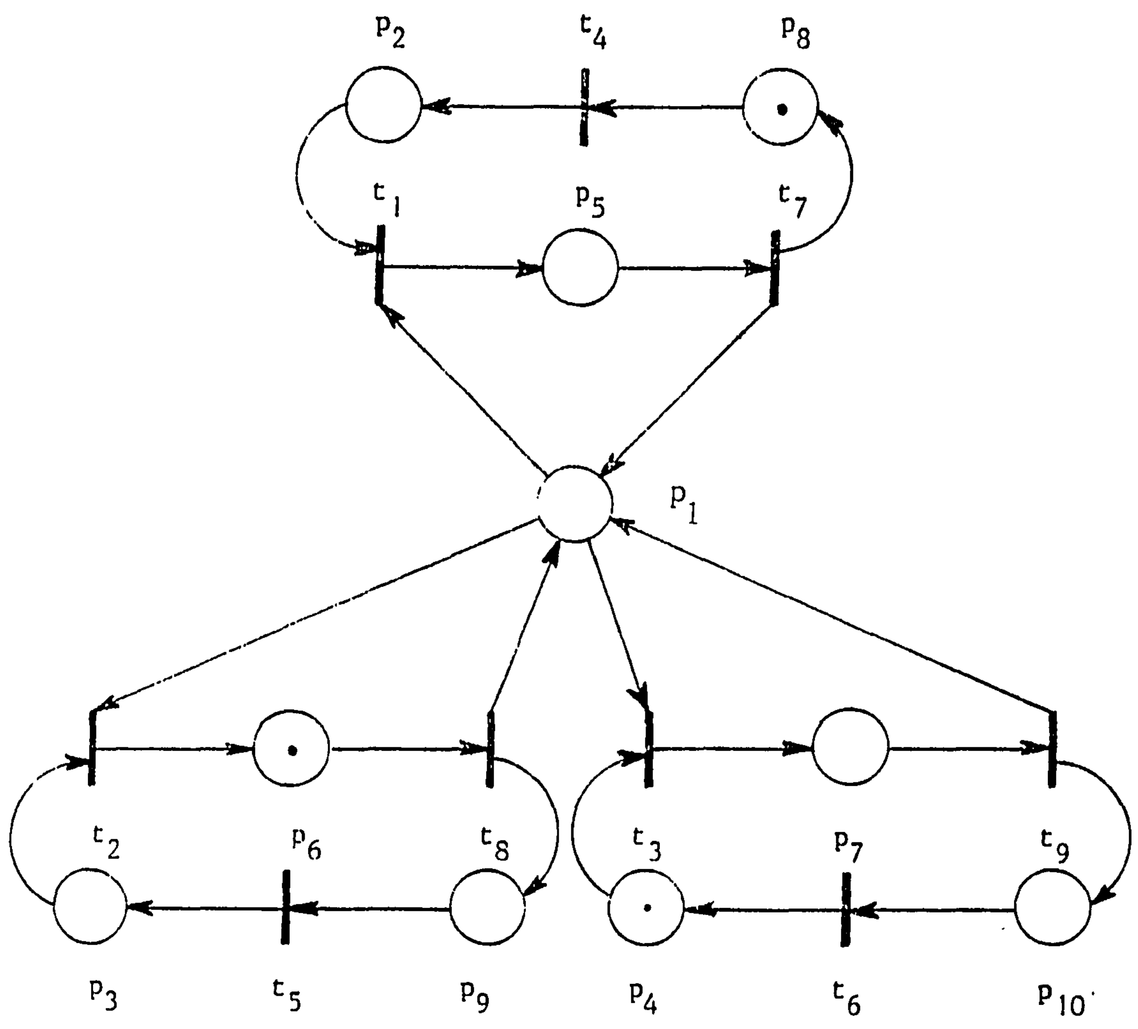


(c) Time = 1^+

: Transition t_7 fires.

: Decision rule selects t_2.

Figure 2.4: (Continued)



(d) Time = 1^{--}

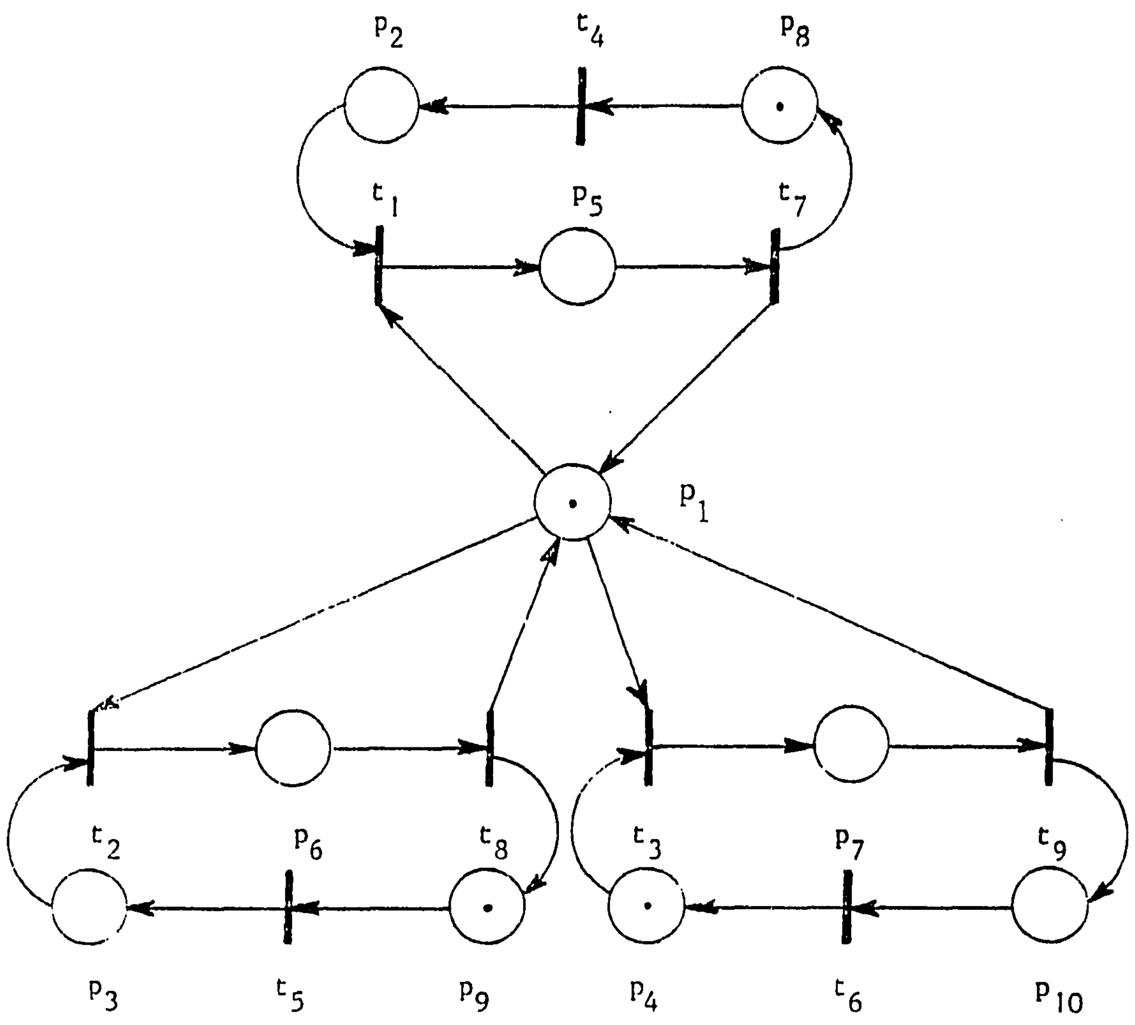
: Transition t_2 fires.

Time = 2

: Token in p_6 becomes active.

: Transition t_8 becomes enabled.

Figure 2.4: (Continued)

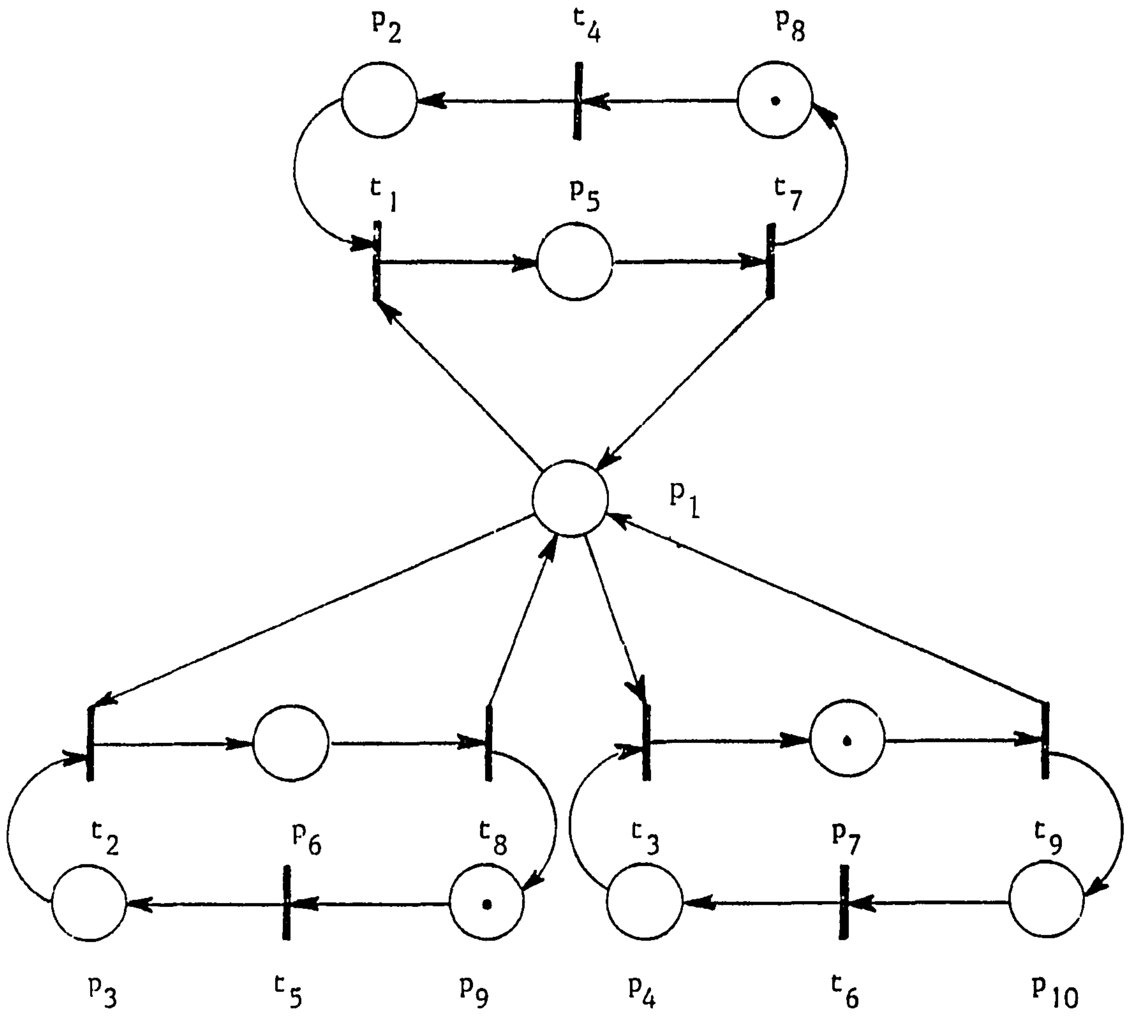


(e) Time = 2^+

: Transition t_8 fires.

: Decision rule selects t_3.

Figure 2.4: (Continued)



(f) Time = 2^{++}

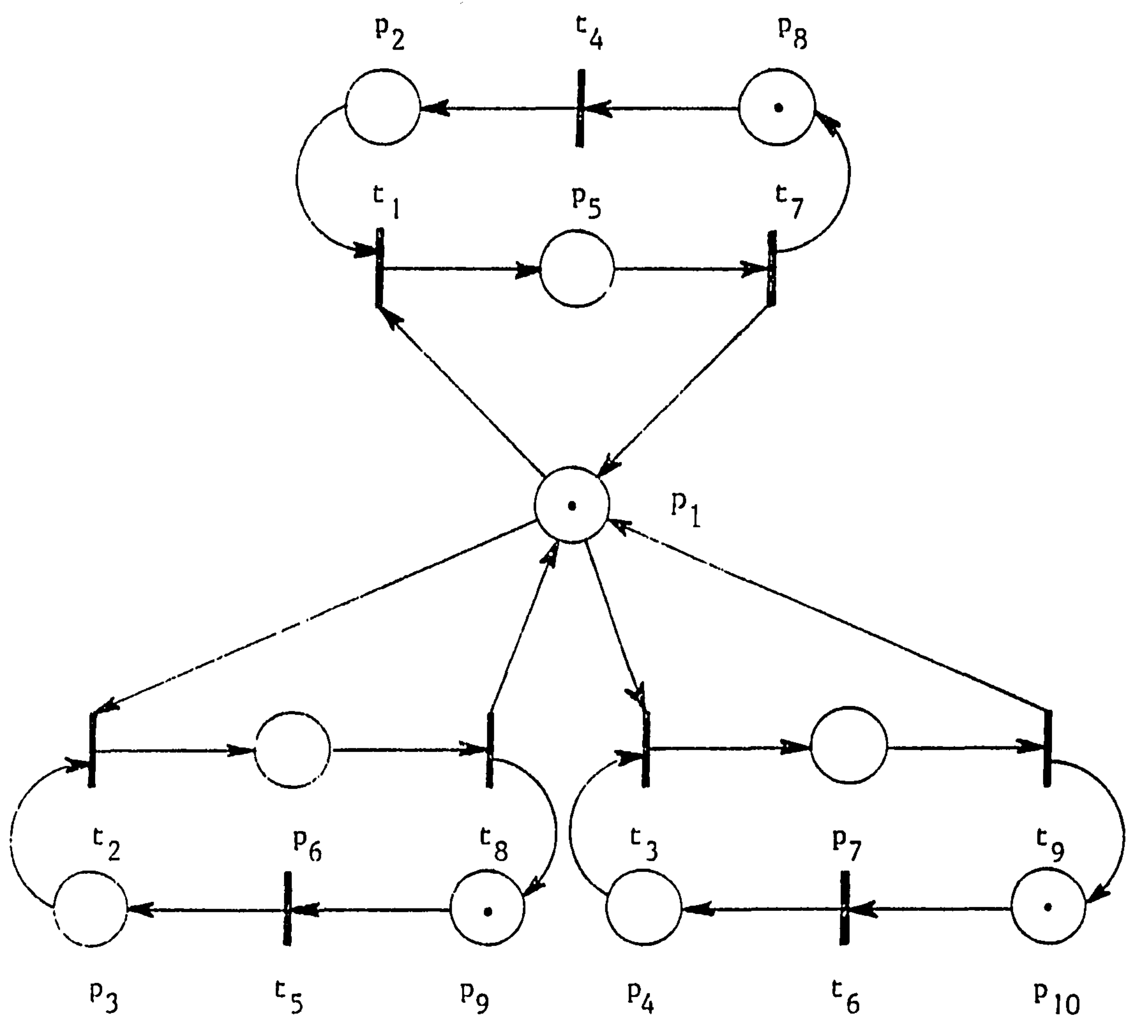
: Transition t_3 fires.

Time = 3

: Token in p_7 becomes active.

: Transition t_9 becomes enabled.

Figure 2.4: (Continued)



(g) Time = 3^{-}

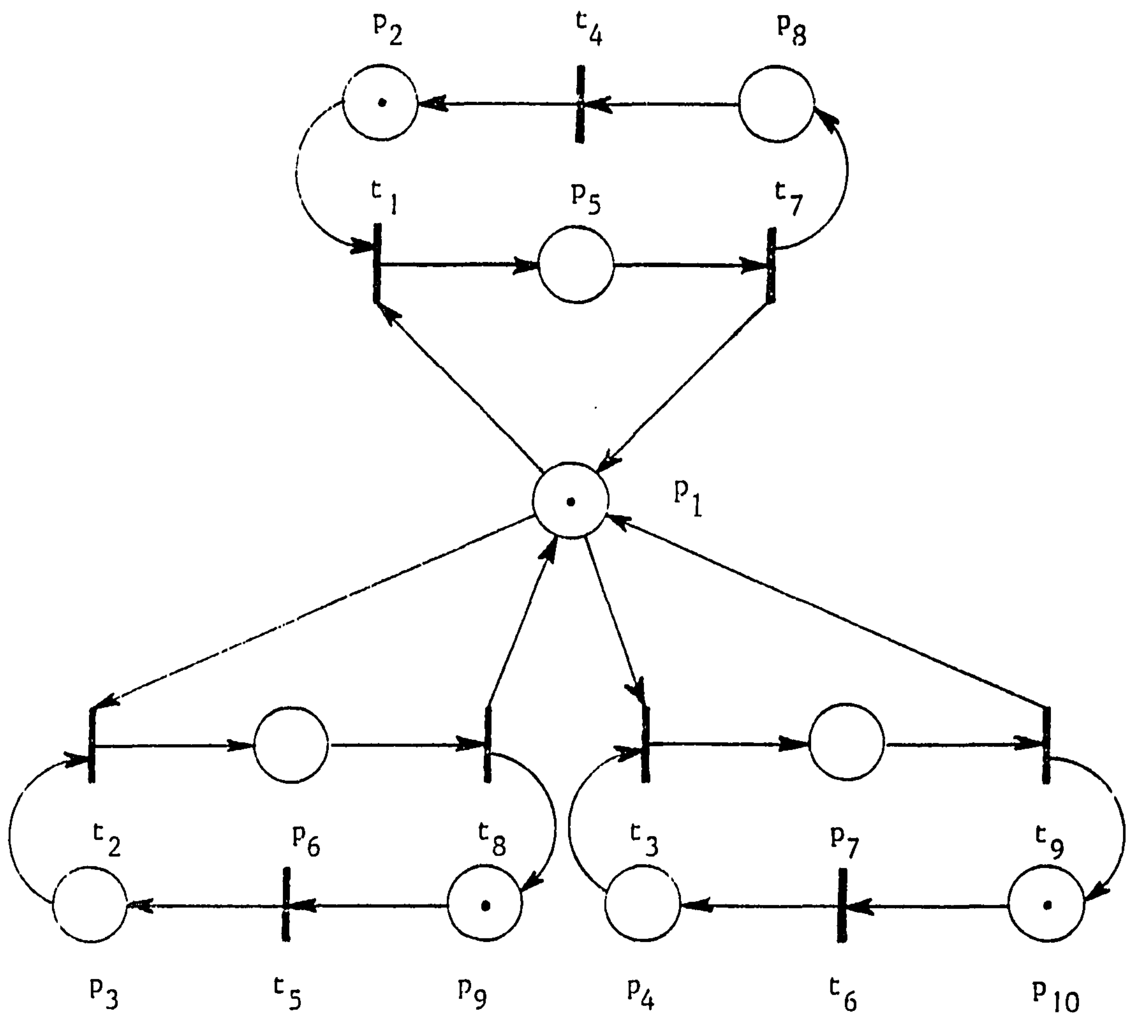
: Transition t_9 fires.

Time = 5

: Token in p_8 becomes active.

: Transition t_4 becomes enabled.

Figure 2.4: (Continued)

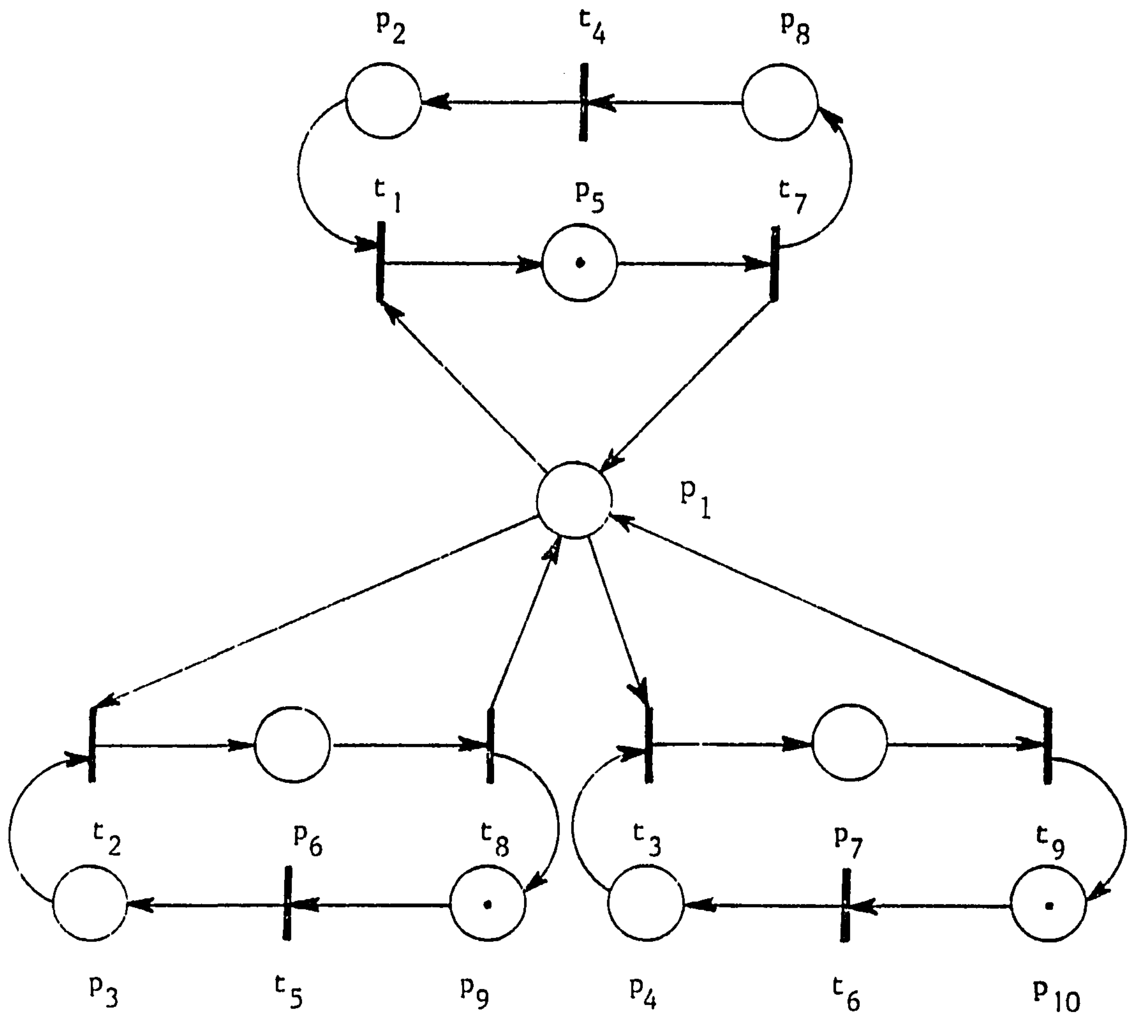


(h) Time = 5^{--}

: Transition t_4 fires.

: Transition t_1 becomes enabled. Since there is no conflict at this time, no decision rule is needed.

Figure 2.4: (Continued)



(i)Time = 5^{--}

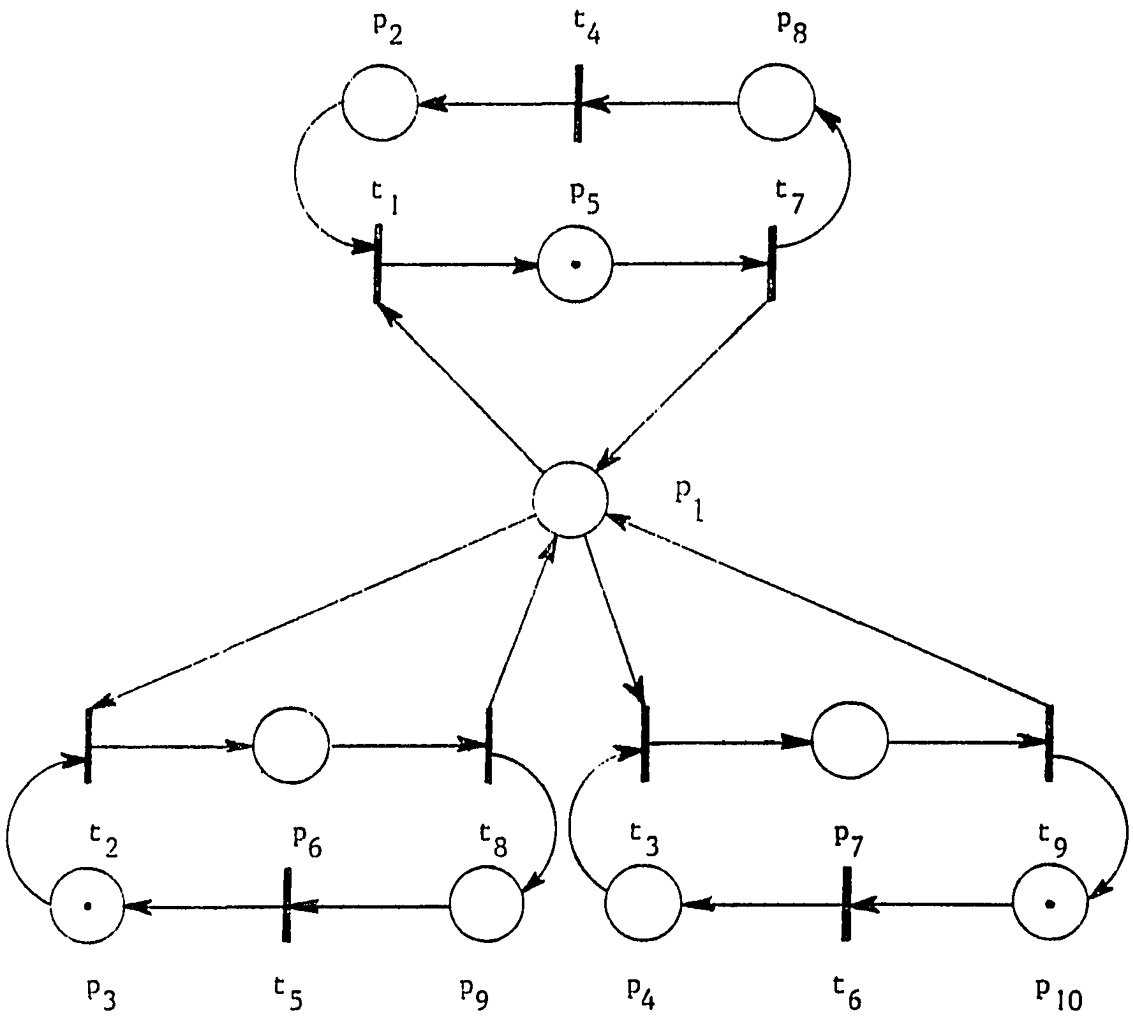
: Transition t_1 fires.

Time = 6

: Tokens in p_5 and p_9 become active.

: Transitions t_5 and t_7 become enabled.

Figure 2.4: (Continued)

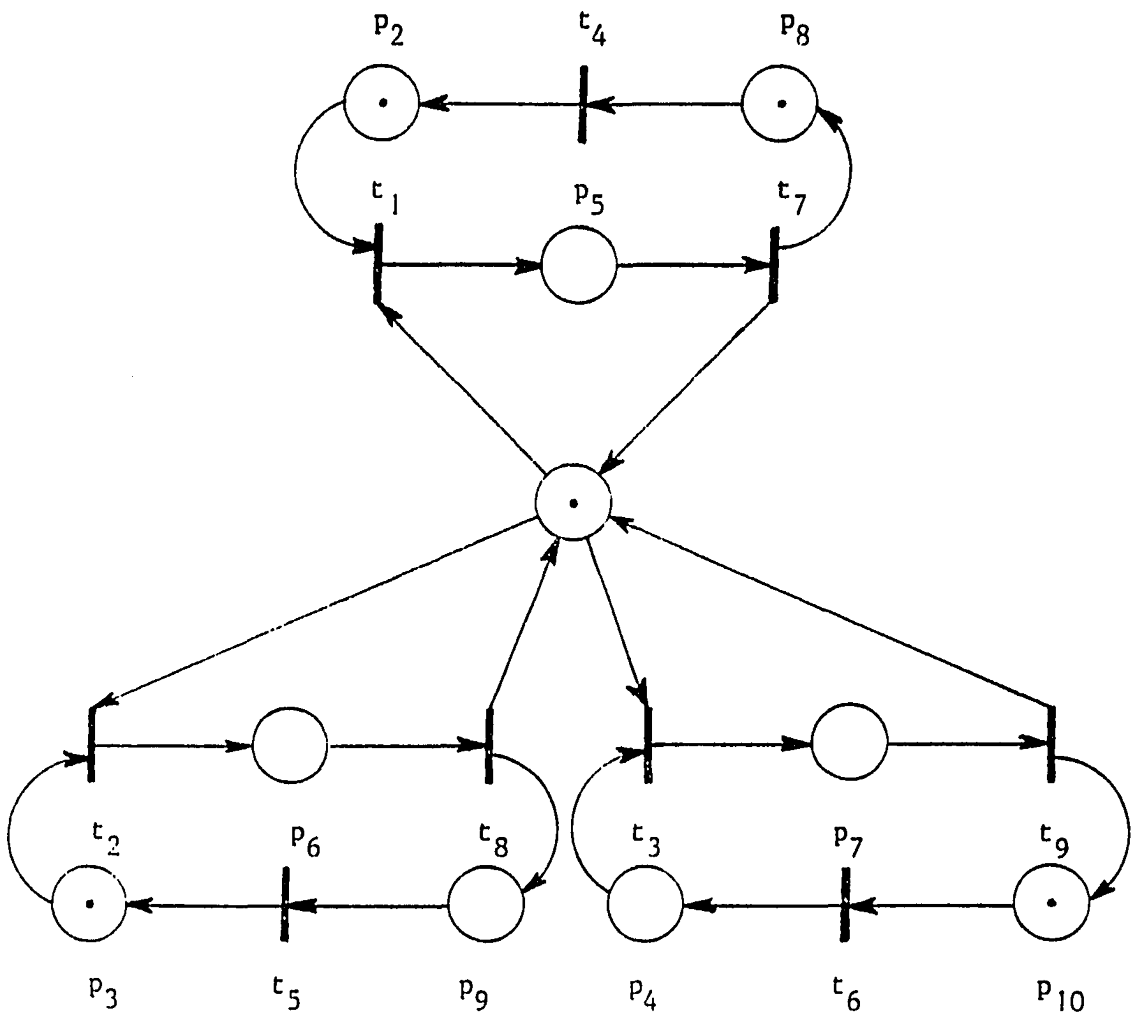


(j) Time = 6^-

: Transition t_5 fires.

: Transition t_7 is still enabled.

Figure 2.4: (Continued)



(k) Time = 6^{++}

: Transition t_7 fires.

Figure 2.4: (Continued)

To keep track of this component, if $\vec{W}$ denotes a vector which contains the time left for places which have active tokens to serve and s represents the number of places, then the dimension of $\vec{W}$ equals n, so that when place p_k contains no active tokens, $\vec{W}$ contains zero as the k^{th} element. Therefore, the state of an STPPN can be defined as follows:

$$(\vec{Ma}, \vec{M_i}, \vec{W}).$$

Figure 2.4 shows how to go from one state to the next in real time basis. From this, a real time simulation algorithm will be set up.

2.2.4 Real time simulation algorithm

Step 1: Construct initial markings, $\vec{Ma}$ and $\vec{M_i}$, and the time vector,[8] $\vec{W_i}$.

Step 2: Construct the transition set enabled by $\vec{Ma}$.

$$T_f = \{t_i |\ t_i \text{ is enabled by } \vec{Ma}\} \quad .$$

Step 3: If T_f is empty, then

Step 3.1: Subtract one time unit from all positive elements of $\vec{W_i}$.

$$w_k \longleftarrow w_k - 1, \quad w_k(> 0) \in \vec{W_i} \quad .$$

Step 3.2: If w_k equals to zero, then

$$(m_i)_k \longleftarrow (m_i)_k - 1$$

$$(ma)_k \longleftarrow (ma)_k + 1 \quad .$$

[8]The vector represents the time left to serve.

Step 4: If T_f is non-empty, then select $t_s \in T_f$ by transition selection rules.

Step 5: For all $in\{t_s\}$ [9],

$$m_a \longleftarrow m_a - 1 \ .$$

Step 6: For all $out\{t_s\}$, determine $u_{out\{t_s\}}$ from the processing time distributions.

Step 7: If $u_{out\{t_s\}}$ is zero, then

$$m_a \longleftarrow m_a + 1 \ .$$

Otherwise,

$$\vec{W}_i \longleftarrow + u_{out\{t_s\}} \text{ and } m_i \longleftarrow +1 \ .$$

Step 8: Repeat Steps (2)-(7).

2.3 Requirements for Establishing Well-formed STPPNs

The simulation algorithm specified in the previous section can be easily used only when the STPPN is well-formed. In order for the STPPN to be well-formed, three following requirements[10] are satisfied: safeness, liveness, and strong connectiveness. The reasons that well-formed STPPNs are used in this thesis are as follows: in many ill-formed STPPNs[11] the states vary in dimension and are constantly changing in forms. This complexity makes it very difficult to analyze STPPNs without loss of generality.

The mentioned requirements will impose easy ordering and assure that no parts of the STPPN become deadlocked and thus help for analyzing a SAC system which can be interpreted by STPPNs.

[9]The $in\{t_s\}(out\{t_s\})$ denotes the input (output) place set of transition t_s.

[10]Those requirements were defined as properties or conditions of STPPNs [2, 55].

[11]STPPNs with deadlocks or STPPNs with unbounded tokens in a place or both.

2.3.1 Safeness

The operation of an STPPN can be generally thought as a succession of marking [57], with tokens changing positions from one place to the next as specified by the rules of operation. If the number of tokens in any given place is 1-*bounded*, then the place is said to be *safe*. If the number of tokens for any given place in the net for all possible reachable markings can never exceed one regardless of what processing time distribution and decision rules are applied, then the STPPN is defined to be *safe*. At this point it would be better to check what differences exist between the above definition and the usual definition used in Petri Net formalism.[12] Two examples explain the difference between these definitions for net safeness. As shown in Figure 2.5, STPPN is safe. However, if processing times are changed as shown in Figure 2.6, STPPN cannot be safe, even though its structure is identical to the one in Figure 2.5. More precisely, the number of tokens can be two by using different processing times as shown in Figure 2.6. However, applying the usual definition to this example, i.e., not considering the time measure, the example net is safe.

The main reason the safeness property is used in this thesis is that it is necessary to impose ordering. If multiple tokens in a place are allowed, then they would be free to cross each other depending on their variable processing times. Any analytical procedure that might be used in this thesis would have to keep track of all possible orderings in which these tokens become active. In order to keep track, this study will be limited to safe STPPNs.

[12]A marked net is defined to be safe if each place in it is safe [2, 24, 55].

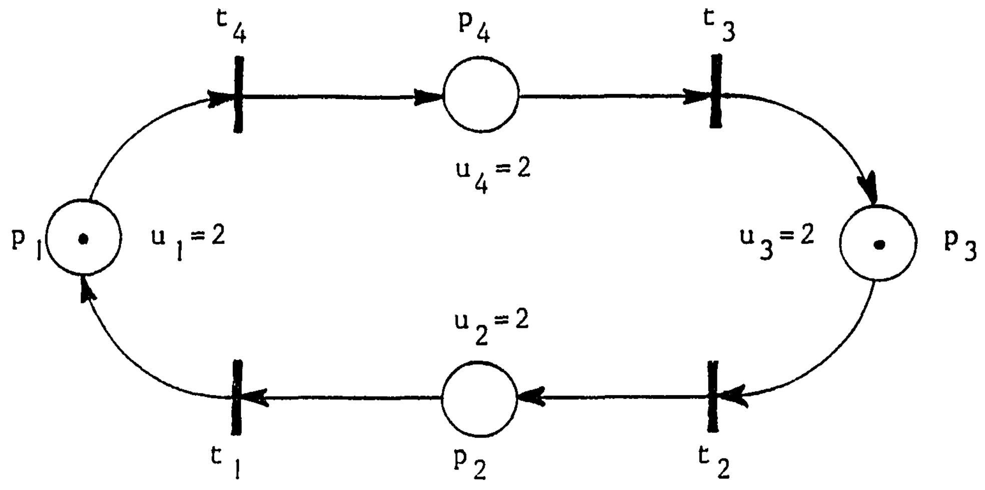


Figure 2.5: An STPPN which is safe

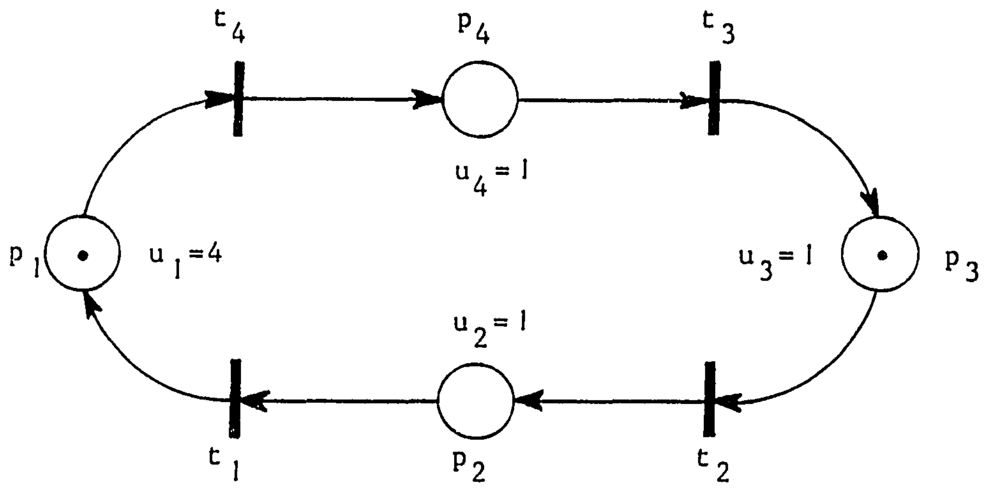


Figure 2.6: An STPPN which is not safe

2.3.2 Liveness

An STPPN is defined to be *live*, if for every transition t and $0 \leq \epsilon \leq 1$ there exists a finite τ_0 such that

$$P_r\{\text{Transition } t \text{ will fire before } \tau_0\} \leq 1 - \epsilon,$$

regardless of what state STPPN has studied and independent of how STPPN evolves. Liveness ensures that all transitions fire regularly. If for some reason a transition stops firing, i.e., the net is deadlocked, then the part of the STPPN associated with that transition should be ruled out, since it does not perform any continuing activities, or the problem should be reformulated so the transition can fire regularly.

Liveness of an STPPN depends on a complex interaction between the nodes[13] of the graph, the initial marking, the processing times and the decision rules. The STPPN shown in Figure 2.7, for instance, is deadlocked (and therefore, it is not safe), since place p_2 has no token. Hence, no transition will ever become enabled, and thus all tokens stop moving.

2.3.3 Strong connectedness

An STPPN is defined to be *strongly connected* if and only if there exists a direct path from any node to any other node. This requirement is needed for the following two reasons. First, It is a necessary condition for any simple connected, live STPPN to be safe [55]. As an illustration of this fact, consider the STPPN shown in Figure 2.8. This STPPN is not strongly connected and the numbers of

[13]Nodes indicate places and transitions.

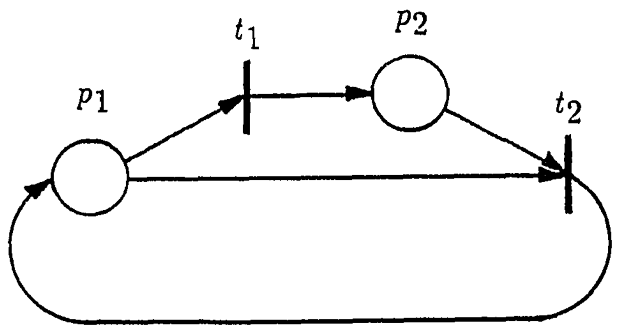


Figure 2.7: An STPPN which is deadlocked

tokens in p_2 can accumulate without bound. Second, strong connectedness implies simple connectedness.

2.4 Subclasses of STPPNs

STPPNs find their basis in few simple rules, yet can exhibit very complex behavior. However, the analysis of STPPNs in general requires some knowledge about the reachability set and can often be quite complex. Therefore, it is necessary to restrict STPPNs in some ways and to study their properties. For this reason, two important subclasses of STPPNs are defined here. Analysis of such subclasses has provided a relationship among net structure and marking on one hand, and dynamic behavior (liveness, safeness, etc.) on the other hand.

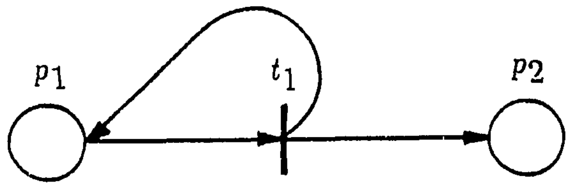


Figure 2.8: A live STPPN which is not strongly connected

2.4.1 Conflict-free STPPN

An STPPN is conflict-free if and only if for each place in the net there is only one input transition and only one output transition. This means that a token at a given place is generated by a predefined input transition and consumed by a predefined output transition. Therefore, the only way to disable an enabled transition is to fire it. Consequently, this kind of STPPN can represent concurrency but not conflict and therefore, no decision rules are needed (see Figure 2.9). Conflict-free STPPNs have been well-studied. The deterministic case is specifically studied in Ramamoothy and Ho [56] and Cohen et al. [13].

2.4.2 Free-choice STPPN

In Free-choice STPPNs, all transactions in conflict have exactly one input place, and that input place is common to all of them. An example is shown in Figure 2.10. The imposed restriction means that the decision to select a transition which will fire among conflicting transitions can be made independently of the state of the rest of the STPPNs. In the aforementioned example, either t_2 or t_3 may fire, and the

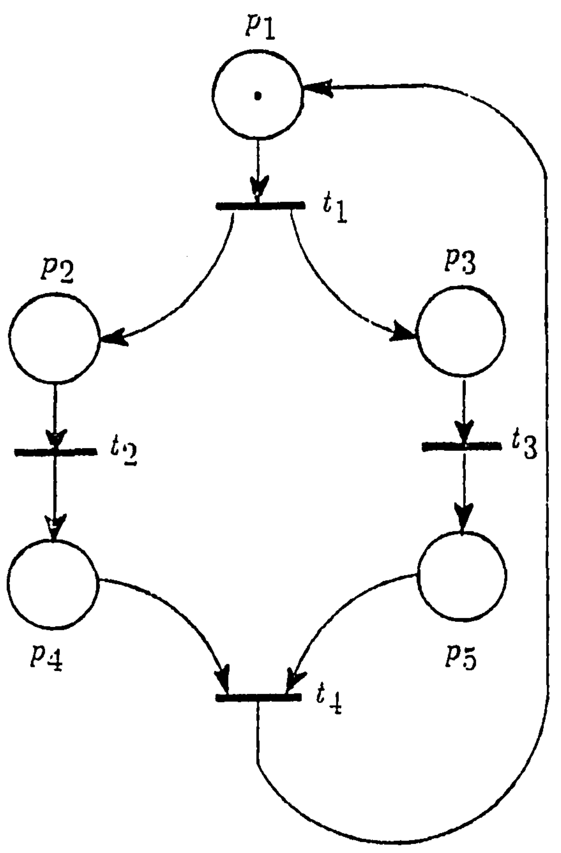


Figure 2.9: A conflict-free STPPN

the decision does not depend on tokens arriving from other parts of the net since the example net contains only two possible token paths which are exactly same. Therefore, the decision which transition can fire when a token arrives at the input place of those transitions depends on only the present marking states of $out\{t_2\}$ and $out\{t_3\}$. If $out\{t_2\}$ has a token, the token arrives at $in\{t_2\}$ will move out through the transition t_3, vice versa.

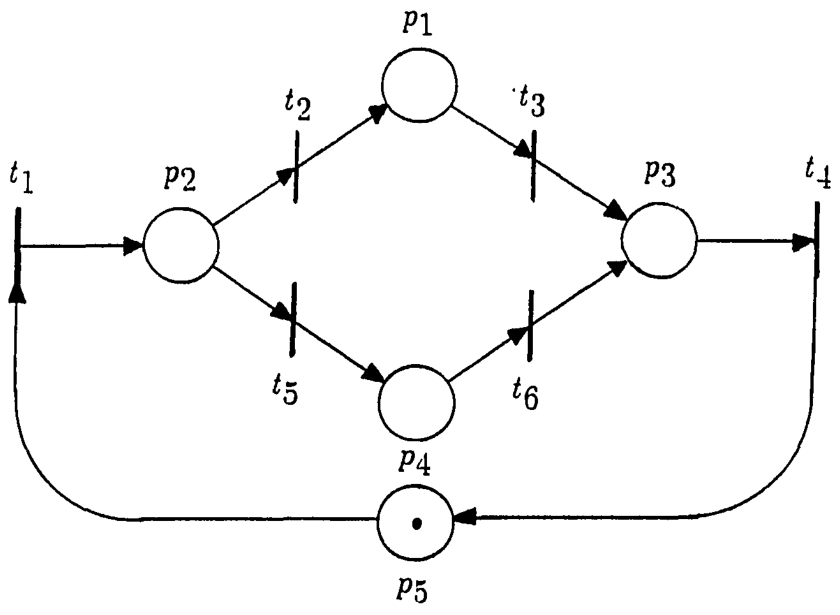


Figure 2.10: A free-choice STPPN

3 COMMAND-CONTROL (C^2) SYSTEM MODELING

3.1 Introduction

Abundant research has been done to improve C^2 systems from several points of view such as development of higher level decision aids [4, 22], preparation of new doctrines to cope with future changes of battle types by highly developed technology [7, 31, 60], design and evaluation of C^2 systems [25, 33, 39], etc. However, as previously described due to insufficient understanding about C^2 processes and systems, research in this area has not been settled. Therefore, in this chapter general characteristics of command-control processes and systems, i.e., how the command is exercised, how a task is monitored (C^2 process) and how C^2 processes are related to other C^2 processes operationally and structurally (C^2 system) are stated.

To do this, Section 3.2 defines terminologies commonly used in this area and their relationships. In Section 3.3, a typical C^2 process and a typical C^2 system are introduced. Then, basic features of the C^2 system are discussed. In Section 3.4, a generic C^2 system is modeled according to operational and structural relationships among C^2 processes by using the STPPN formalism.

3.2 Terminologies

Information: Raw data originated from changes of external and/or internal environments. The former is originated from events which activate the system operation: for example, unidentified objects which appear on the radar. The latter is obtained from events which occur in the system operationally and/or structurally.

Knowledge: Refined data generated from information based on objects or events which occur in the internal/external environments by a decision-maker which is used to be a process or a component in the C^2 process.

Command: The exercise of lawful authority done by high ranking personnel to their subordinates or units in order to carry out assigned missions and to further attain the entire unit's goals under their controls.

Control:[1] The act or power of asserting command or authority, especially in pursuance of a specific plan of action.

C^2 process: The means by which a team of military personnel makes decisions that relate to the deployment and motion of the resources and assets assigned to carry out a military mission specified by higher authority.

C^2 organization: The hierarchical way and organization rules by which military personnel organize themselves in terms of C^2 authority and responsibility by warfares and/or geographical sectors.

C^2 system: The hierarchical networks of C^2 processes.

[1]Too often *control* is confused with *command* in its meaning. The basic difference is the latter causes actions by subordinates that attain a result [10] and the former is actions to ensure that the command is complied with in such a manner to attain the right result [30].

3.3 C^2 System of a Battle Organization

3.3.1 C^2 system descriptions

A C^2 system of a battle organization can be presented as shown in Figure 3.1. The system not only has hierarchical structures, but also can be considered as a SAC system operated asynchronously and concurrently. Therefore, as mentioned in Chapter 1, this system can be presented well by Stochastic Timed Placed Petri Nets. The system consists of one commander (CMD) section, one combat-information-center (CIC), a battle groups (BGs),[2] ab sub-battle groups (SBGs), and c independent battle groups (IBGs) directly controlled by CIC. Once the system receives the information for threats of the counterforce obtained by detecting manners such as optical search or radar detection, each process in the battle group proceeds its task with its own decision process. Then, each unit reacts for coming threats. (See Figure 3.2.)

A decision process contains four decision steps: (1) threat analysis (TA); (2) availability evaluation/resource allocation (AR); (3) command interpretation (CI); and (4) response selection (RS). As seen in Figure 3.2, each battle group is connected by communication routes such as reporting and commanding (see Figure 3.1).

3.3.2 Basic features of a C^2 system

In this thesis, a fundamental assumption that the underlying system is in the battle mode is taken. Thus, several different situations in comparison with peacetime should be considered.

[2]Each battle unit is supported by b SBGs.

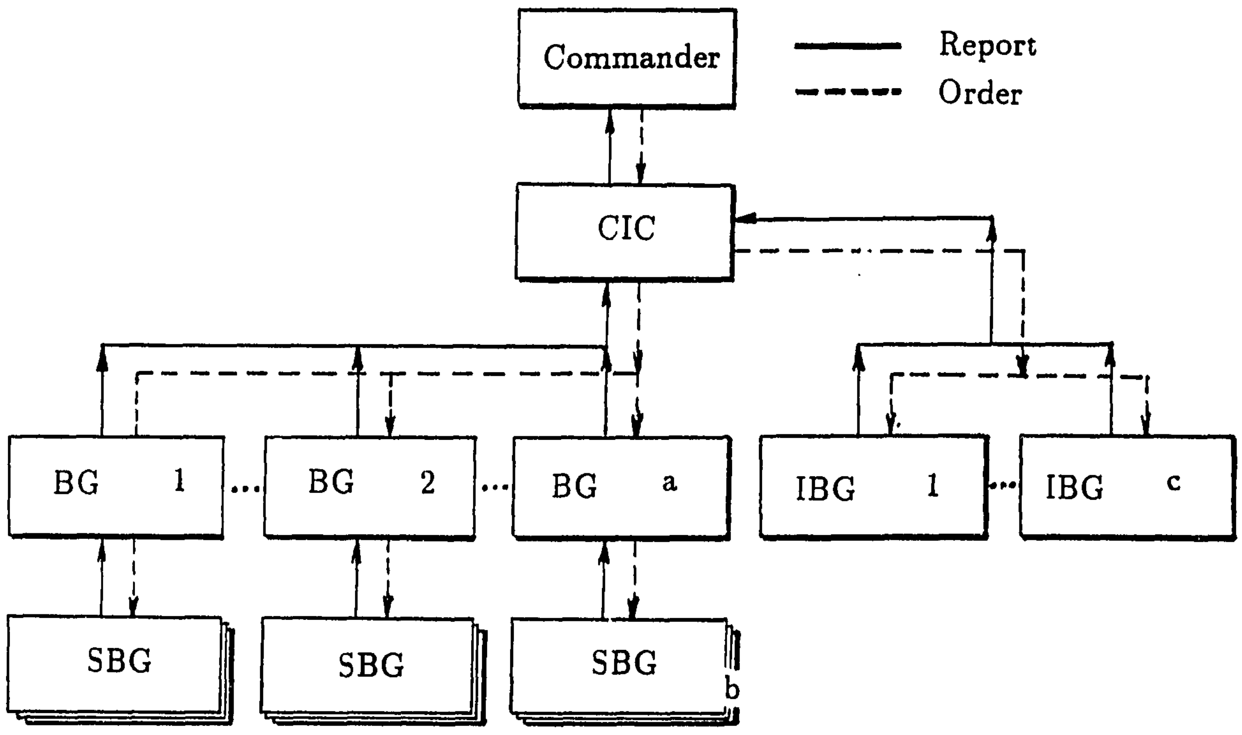


Figure 3.1: System structure of a C^2 system

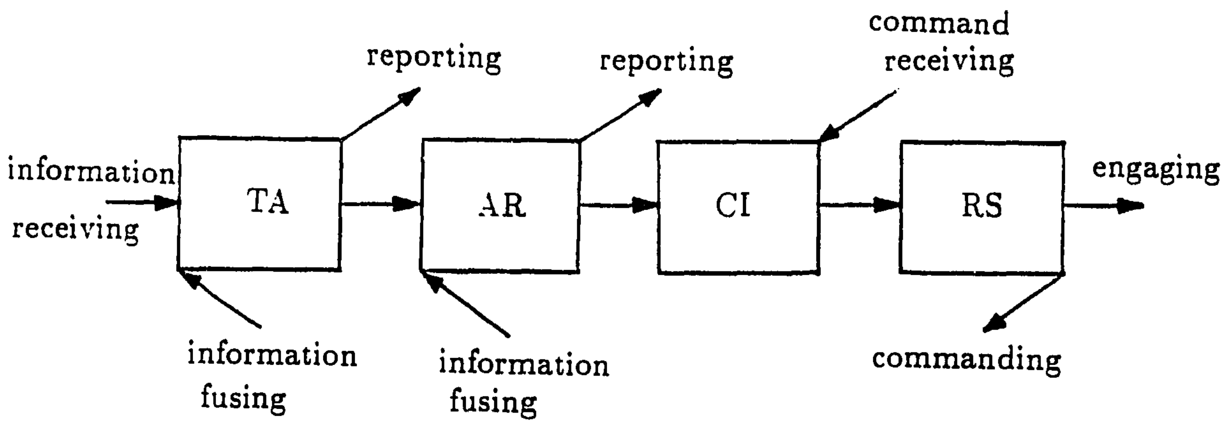


Figure 3.2: Typical decision process of a C^2 system

First, information collecting may not be regular in wartime, although events which occur may be predictable before information is received about these events; whereas in peacetime, information collecting is rather predictable and regular. Therefore, information collecting time is not deterministic, but random. However, since it is not the concern of this thesis, incoming information (i.e., system input) is assumed to always be available.

Second, task processing times are also stochastic, although time variation may become narrower than before by using decision aid devices such as computers and by repeating exercises.

Finally, system components in war time may be destroyed with some probabilities by engaging with their counterforces. Thus, system performance decreases by ruling out of destroyed combat units or decision groups. Thus, more burden should be assigned to the rest of combat units in order to perform system tasks unceasingly. To do this, deterministic hit and malfunction probabilities are assigned to each component. Also, the system structure is changed by the reasons described above as another feature of the wartime C^2 system.

Therefore, in this thesis, two C^2 systems, the system operating with normal communication routes (system N) and the system operating with changing communication routes (system C) are evaluated and compared in terms of system effectiveness.

3.4 STPPN Modeling of C^2 Systems

As assumed previously, the STPPN models of two C^2 systems are safe, live, and strongly-connected. In this section, in order for the underlying system to transform into STPPN models, two design factors, capacity and time are considered.

3.4.1 C^2 process model

Figure 3.3 show aggregated Petri net models of a C^2 process. As described in Subsection 3.3.1, the decision process occurs in four stages: (1) threat analysis (TA), (2) availability evaluation/resource allocation (AR), (3)command interpretation (CI), and (4) response selection (RS). Incoming information (either external or internal) are fused at each TA stage with the threat assessments transmitted from other C^2 processes and then assessed. This knowledge is then combined by the AR stage with availability evaluation and resource allocation. The resulting knowledge is combined with commands transmitted from its superior in the CI stage, so as to select a response for the threat in the RS stage.

Each of the four stages corresponding to the particular task performed by each C^2 process is modeled by a place and connected by a transition. The firing of a transition represents the receiving and transmitting of information or knowledge from one place to other places, according to the task sequence which follows the structural relationship of processes. Transitions receive and transmit the information that is exchanged between different processing stages.

Meanwhile, since places represent various tasks to be processed, characteristics of places need to be further specified. From Figure 3.3, it can be seen that places contain two types of information besides incoming information from the environment. That is, places process the information being exchanged between a process and other processes or environment, and places carry the information being internally processed at a C^2 process. The first type of place allows for modeling the

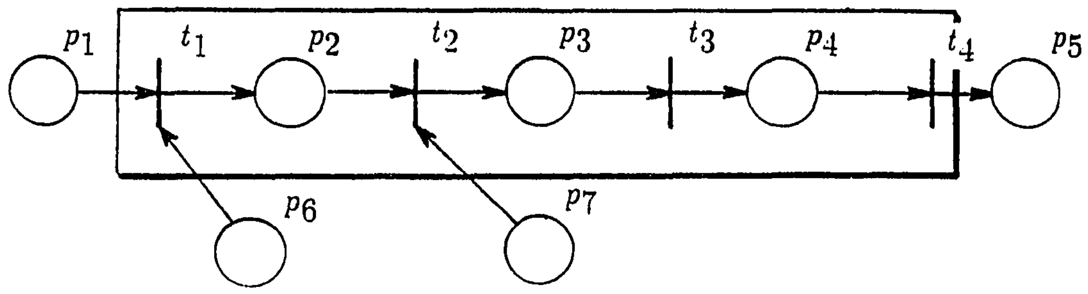


Figure 3.3: Model of a C'^2 process

interactions between C'^2 processes and the environment, or other C'^2 processes. Places containing such information or knowledge are inputs to the C'^2 process or outputs of the C'^2 process, but in no case can they be both at the same time. Therefore, they determine in fact the organizational structure (that is why the place is called *structural place* [32]), i.e., the internal interactions between C'^2 processes or the interactions between the C'^2 process and the environment. The second type of place makes possible modeling of the internal structure of the interacting C'^2 process. This type of place treats information used at the next stage. In contrast to the structural places, these places are both input and output places of the same C'^2 process. At this point, it should be considered that the processing of a specific input takes place in an asynchronous manner, i.e., delays generally occur between the different processing stages precisely due to the interaction with the other processes or the environment. Therefore, a place acts as a temporary storage of information until its output transition can be enabled.

3.4.2 C^2 process model with constraints

As seen in Figure 3.3, the model of the interacting C^2 process does not take into account the limited capacity for information processing which characterizes the process. Indeed, as long as information messages exist, i.e., tokens are available in places p_1 and p_6, the processing can start, i.e., the process corresponding to the TA stage can proceed by firing transition t_1. However, the information processing is subject to the bounded rationality constraint, as defined by Boettcher [6]. In the information-theoretic approach, the amount of information processed is measured by the total activity, Q, of the process, which also characterizes its workload. It is assumed that there exists an upper bound of Q, Q_u, above which the process becomes overloaded and its performance decreases:

$$Q \leq Q_u \ .$$

When the analysis is performed for the steady-state process, the above constraint takes the form:

$$Q \leq \frac{F}{\lambda} = Q_u \ ,$$

where F is the processing rate that characterizes the process and λ is the average arrival rate of an input. This constraint implies that the C^2 process must perform inputs at a rate at least equal to the rate at which they arrive.

As mentioned previously, in this thesis the performance measures are only concerned. In other words, the accuracy of the response is not dealt with. In particular, the way a C^2 process reacts to information overload and the extent to which C^2 affects its performance are not a matter of concern here. However, this remains allowable in so far as the actual processing constraint is included. This can be expressed by writing the inequality in another form,

$$\lambda \leq \frac{F}{Q} \ .$$

Using the above relation, the bounded rationality limitation is, in this case, a constraint on the allowable rate of incoming inputs, i.e., on the maximum rate of inputs that can be handled by the C^2 process without being overloaded.

Now, consider how to model this constraint using the Petri Net formalism. In fact, the limited processing capability of a C^2 process come from the limited capacity available to perform the various processing tasks. In particular, the bounded rationality constraint is very much related to the limited capacity of the human short-term memory, defined as the memory in which the information is held temporarily. Indeed, this bound means that a C^2 process cannot handle properly too much information at the same time. Therefore, the limited capacity of the short-term memory can be modeled, using the Petri Net formalism, as a capacity constraint on the corresponding places p_2, p_3, and p_4 as shown in Figure 3.4.

The added place p_c, whose input transition corresponds to the RS stage and output transition to the TA stage, allows for modeling the information processing constraint. p_c can be called *capacity place* because the number of tokens put initially into this place represents the capacity available for processing. Indeed, the direct path,

$$\rho = (p_c \, t_1 \, p_2 \, t_2 \, p_3 \, t_3 \, p_4 \, t_4) \ ,$$

determines a circuit in the net in which the token content remains invariant by transition firings, i.e., for any reachable marking M:

$$M(p_c) + \sum_{i=2}^{4} M(p_i) = M^o(p_c) + \sum_{i=2}^{4} M^o(p_i) = M^o(p_c). \qquad (3.1)$$

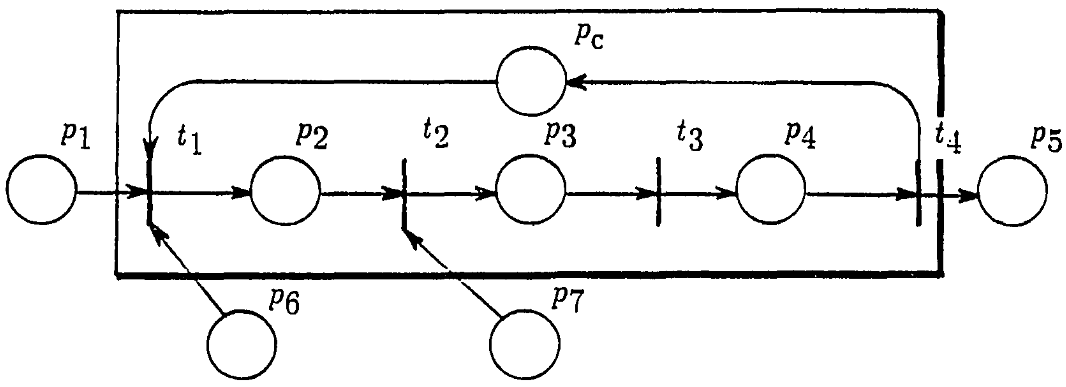


Figure 3.4: C^2 process model with limited capacities

It is assumed that places p_2, p_3, and p_4 contain initially no tokens, which means that there are no processes being performed initially. Now, the constraint on the memory capacity is trivially satisfied. Assuming that there are n capacities available, i.e., $M^o(p_c) = n$ which may represent the amount of memory space available, it can be deduced from the marking equation (3.1) that, at any reachable state of the process,

$$M(p_2) + M(p_3) + M(p_4) \leq n \quad ,$$

which precisely models the short-term memory limitation, as described previously.

At this point, it is possible to extend further analysis of the model, especially with respect to the processing rate. Now, specifying how the processing of inputs occurs with the same assumptions about the initial marking of each place, i.e., $M^o(p_c) = n$ and $M^o(p_2) = M^o(p_3) = M^o(p_4) = 0$, the processing of any new input starts with the firing transition t_1, which consumes one token from place p_c.

Likewise, when the processing is completed, transition t_4 fires, which produces one token back to p_c. In other words, one capacity is engaged at the beginning of the process and is released at the end of the process. Clearly, the C^2 process can only process at most n different inputs at the same time. If n inputs are currently being processed, the p_c is empty. Thus, transition t_1 is not enabled although place p_1 has an active token. In other words, the capacity (or resource) constraint turns out to be bounded by the number of inputs that the C^2 process can handle simultaneously.

To analyze deeper how the capacity constraint affects the processing rate, it is necessary to introduce the mean processing times of the processes, P_2, p_3, and p_4 that will be denoted by μ_2, μ_3, and μ_4, respectively. The main concern here is to make clear how the capacity limitation actually bounds the processing rate. Now, if the decision processes were fully synchronous, which means that no delays would occur between the different processing stages, it would take the amount of time,

$$\mu_s = \mu_2 + \mu_3 + \mu_4 \ ,$$

to complete the processing of any input. Since the C^2 process can handle at most n inputs at the same time, the processing rate, f, is necessarily bounded by

$$f = \frac{n}{\mu_s} = \frac{n}{\mu_2 + \mu_3 + \mu_4} \ .$$

There is, however, an additional constraint to include, i.e., the execution of any processing stage can take place for one input at a time as discussed in Chapter 2. Therefore, the actual bound of the processing rate is determined by

$$\phi = min(f, \frac{1}{\mu_2}, \frac{1}{\mu_3}, \frac{1}{\mu_4}) \ . \tag{3.2}$$

It will be shown in Chapter 5 that ϕ determines precisely the maximum information processing rate that characterizes the C^2 process. This rate constraint is derived from the limited capabilities of a C^2 process, which include both the capacity limitations and the processing time constraints. ϕ is, in fact, similar to the processing rate constraint f.

3.4.3 C^2 system model

So far, the model of the simple interacting C^2 process has been developed. Next, the aggregated Petri net model for the overall C^2 process, which forms a C^2 system, is presented.

3.4.3.1 C^2 system model interacting with the environment The interaction between a C^2 system and the external environment has been described in Subsection 3.3.2. The first processing stage of the C^2 system consists of the partitioning of the external information into a set of input information that is assigned to different C^2 processes. This kind of information allocation has been addressed by Stabile and Levis [64]. Assuming, for example, that the C^2 processes, $C_1^2, \cdots, C_5^2$, are interacting directly with the environment, the corresponding Petri net can be represented as shown in Figure 3.5.

The place p_o represents the source of information and the transition t_1 models the partitioning operation. Since it is held at the first processing stage, t_1 will be called the input transition of the process. Each time that t_1 fires, one token is sent to places, $p_1, \cdots, p_5$, which are the input places of C^2 processes, from the capacity places of C^2 processes, $p_{c1}, \cdots, p_{c5}$, respectively. It should be clear that this model implies an overall synchronization between the individual inputs received by each of the C^2 processes. In the following stages, however, the processing of the inputs

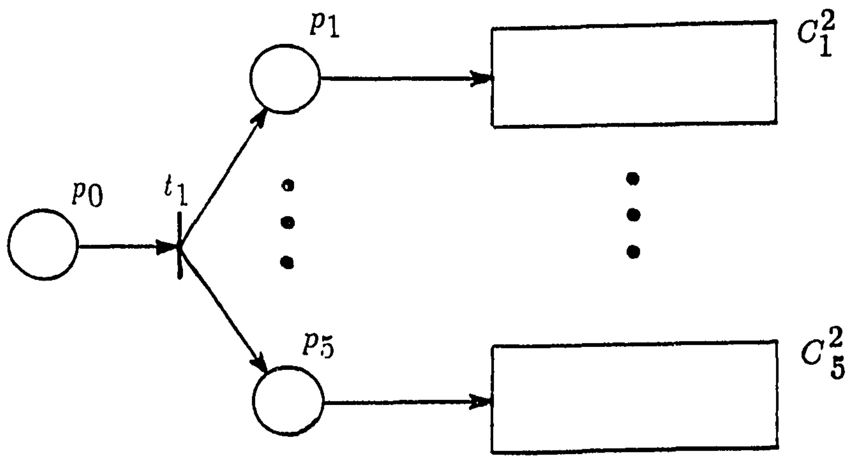


Figure 3.5: Petri net representation of the partitioning information

by each C'^2 process becomes asynchronous and concurrent.

Likewise, the Petri net representation of the C'^2 system has an output transition which characterizes the last processing stage. The output place of this transition contains the system responses for the input information.

Interestingly enough, the overall process can be imagined as taking place through a sequence of three major stages. Each of the stages corresponds to a particular action, i.e., the input action, the processing action, and the output action.

3.4.3.2 C'^2 system model with limited capacities The analysis of the capacity constraint performed in Subsection 3.4.2 can be applied, in fact, to the system as a whole. Indeed, the capacities used by the overall system may have various forms, but there exists always at least a processing constraint that comes from the limited capacity of the structural places. This constraint is very similar to the one existing for the storage places of each C^2 process.

As described in Subsection 3.4.1, the structural places receive and transmit the information between C^2 processes, and between C^2 processes and the environment. Since the process is asynchronous, the information is stored temporarily in these places, exactly as what occurs for the internal decision process of a C^2 process. These places are, in fact, the buffer storage within the system, where some information is stored temporarily. It is, therefore, important to assure that at no instant does the amount of information stored exceed the buffer capacity of the system, i.e., the capacity of the structural places.

Assuming that the arrival rate of external inputs exceeds the maximum processing rate of any one of the C^2 processes, it follows that t_1 will fire at the same rate with which input information arrives. Therefore, tokens will necessarily accumulate in the system and, in particular, the token contents of some structural places of the net will eventually grow to infinity over time. In such cases, the system is overloaded.

To treat this problem, the Petri net model should be modified by adding an extra place, exactly as done for the single C^2 process. The resulting model is shown in Figure 3.6. Q_O is the capacity of the overall system: the number of tokens put initially in this place bounds the number of input information that the C^2 system can process simultaneously. Indeed, each time that a new input is processed, a token is removed from Q_O and returns to Q_O once the process is completed. Assuming that Q_O contains initially n tokens and that n inputs are currently being processed, then Q_O is empty and no more information can be received because t_1 is not enabled.

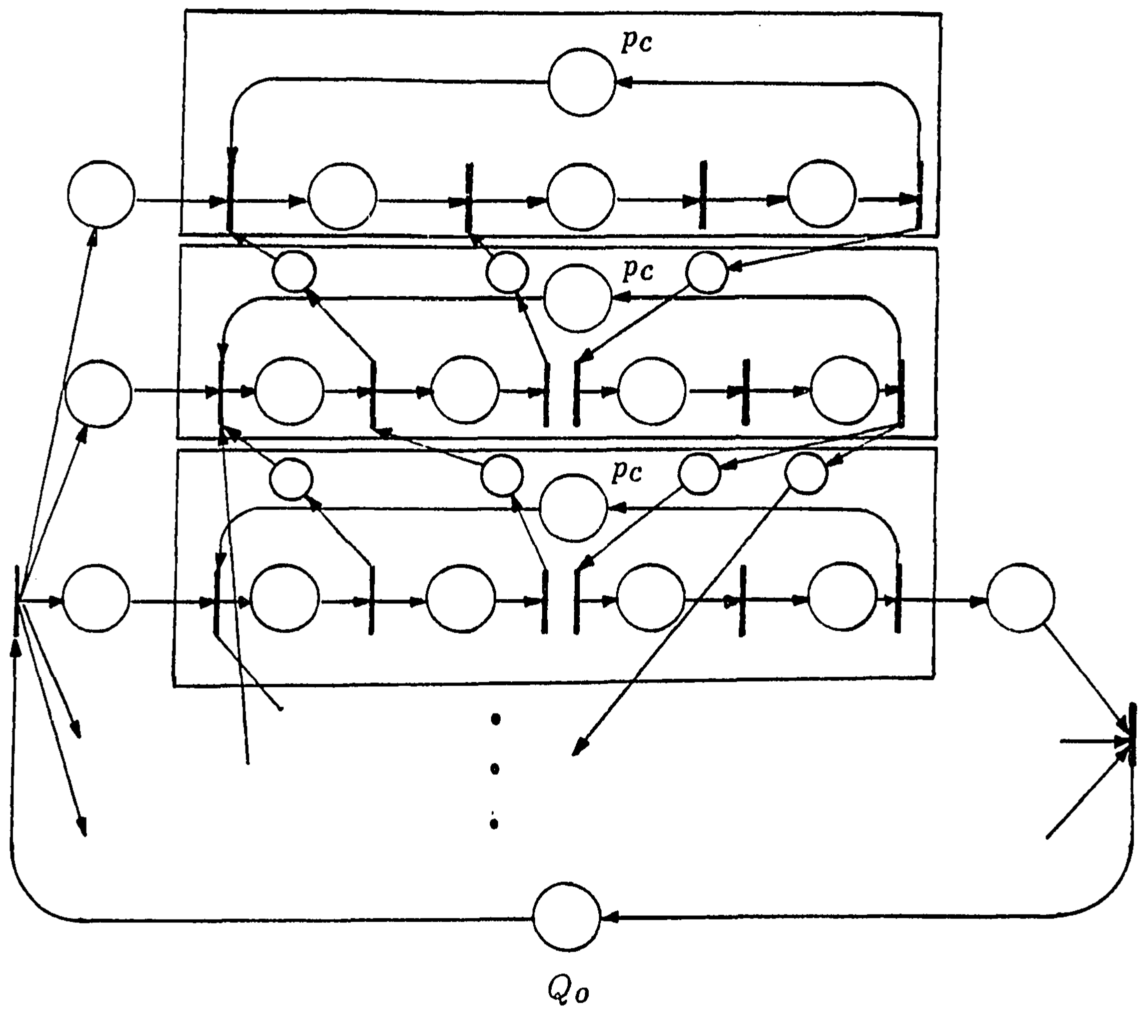


Figure 3.6: C'^2 system model with limited capacities

3.4.3.3 C'^2 system model with time constraints The task processing times are included in the model by assigning to each place a corresponding execution time. Now, all needs to model the C'^2 system described previously into an STPPN model are prepared. Therefore, the underlying C'^2 systems can be modeled as shown in Figures 3.7 and 3.8 for the normal and the changing structure cases, respectively.

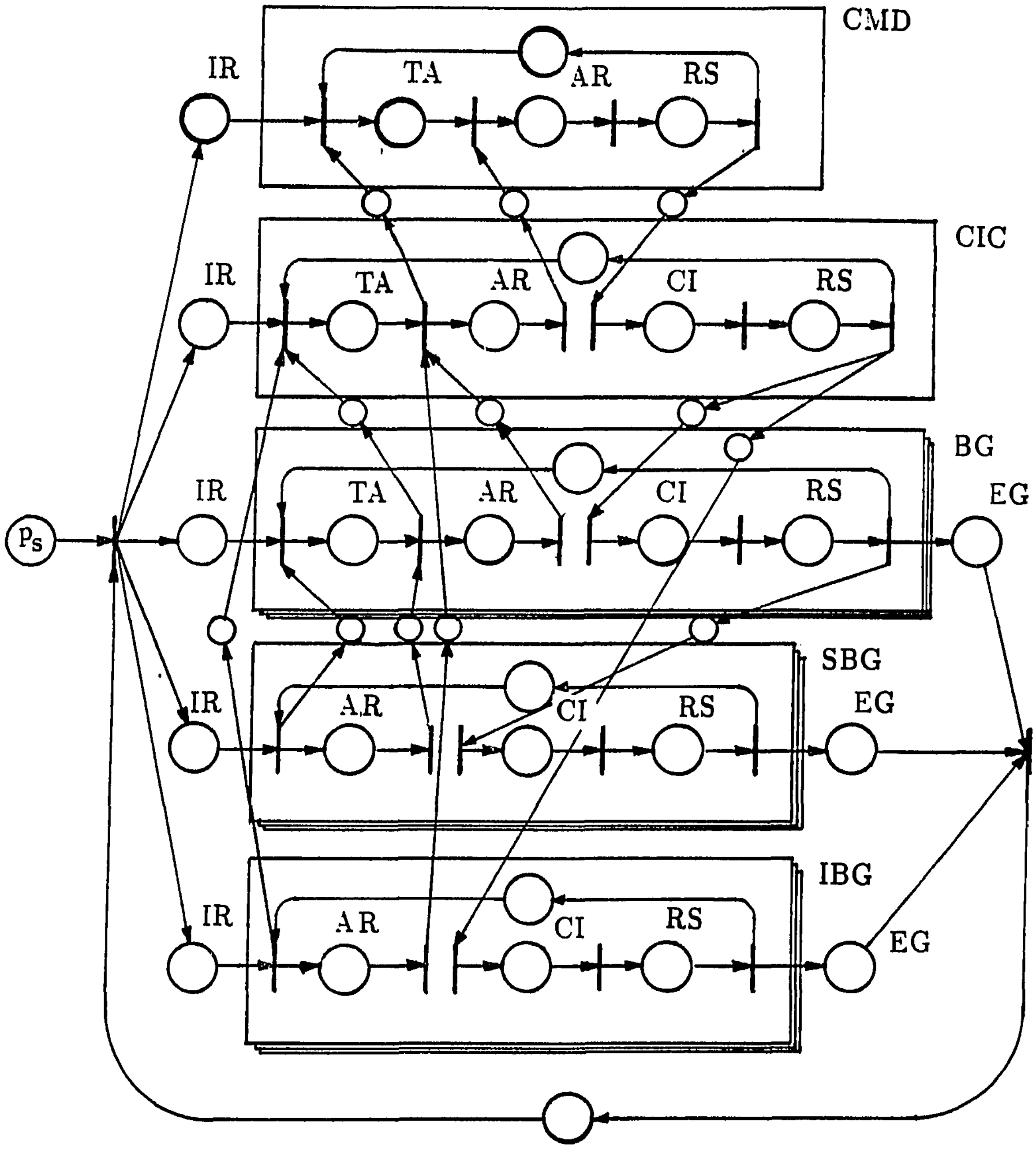


Figure 3.7: STPPN model of system N (model N)

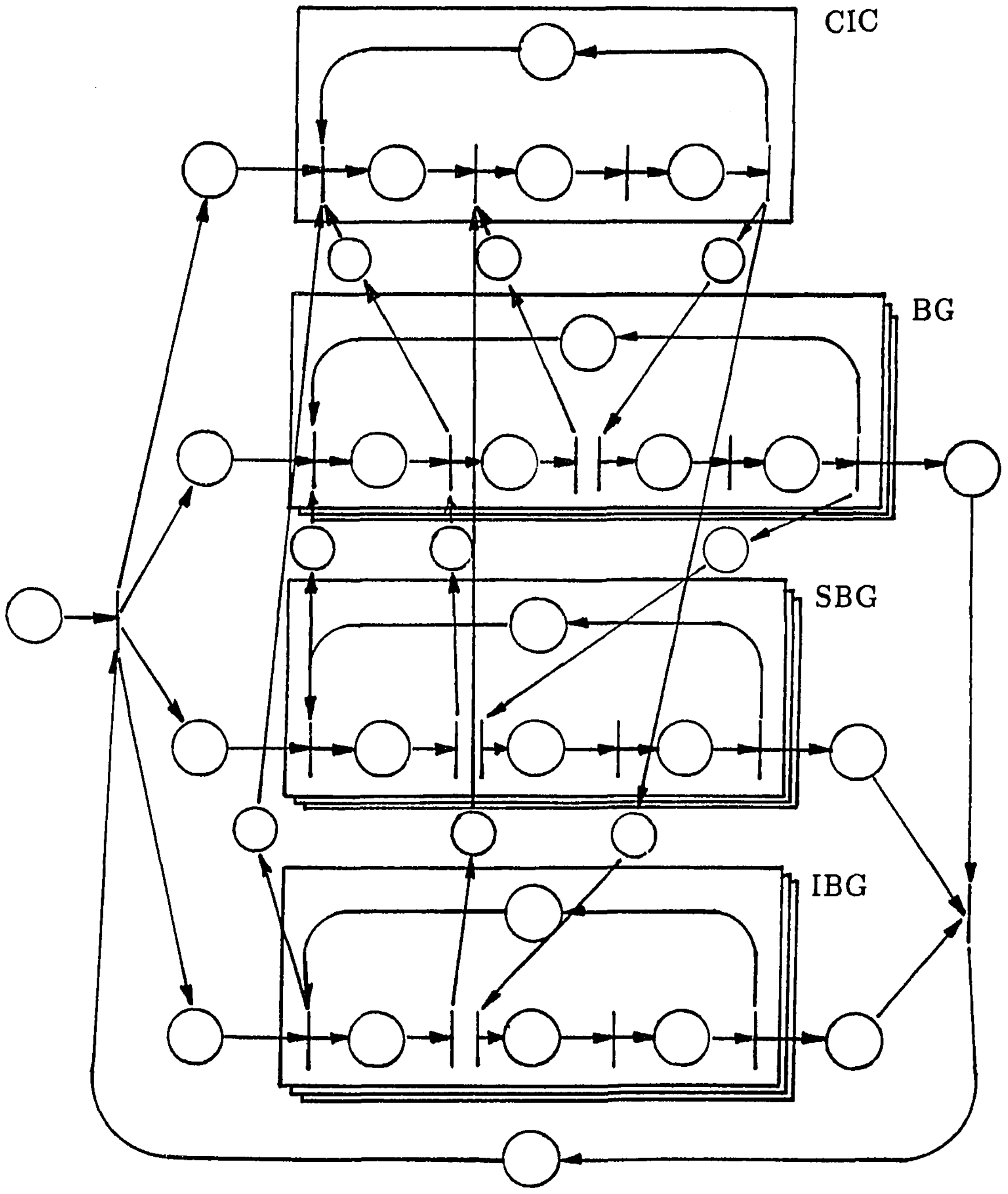


Figure 3.8: STPPN model of system C (model C): CMD destruction case

4 STATE FORMULATION OF STPPN

4.1 Introduction

In Chapter 2, it was shown that the state of an STPPN could be explained by the position and the creation time of every token. The time was decided by the firing time of the transition, which created the token. In this chapter, state equations are formulated to keep track of the marking and of the transition firing time.

In Section 4.2, the unfolded STPPN is presented, which is an another form of the original STPPN, in order to decompose the evolution states of the original STPPN. Also, the fact that the order in which transitions fire with respect to the event time in the unfolded STPPN can be changed without affecting the real time is shown. With the order alteration, the evolution of the original STPPNs can be visualized to an infinite collection of unfolded STPPNs. This visualization provides a partial order of the unfolded STPPNs. Section 4.3 discusses the marking process that accounts for all reachable markings the STPPN can go through. This is accomplished by using the partial order. Since the marking process is augmented with time elapse, an ordered index which can be used to describe the system can be obtained by counting the number of transitions fired. Once the state equation is obtained, the evolution of the net can be traced and its behavior can be investigated by iterating this equation from a state to the next state. These equations provide the way to simulate the evolution of the net. Therefore, in the final section, state equations which keep track of transition firing times are deduced. Additionally, the event simulation algorithm is developed.

4.2 Unfolded STPPN

One of the possible ways used to describe the STPPN mathematically is to obtain an ordered index. In order to get the ordered index among transition firings, unfolding the original STPPN is necessary. The resulting STPPN, the unfolded STPPN, should contain no simple direct circuits[1] with maintaining the causality of the original STPPN (see Figure 4.1). A partial order will be established by using the unfolded STPPN, which is the basis in obtaining the ultimate ordered index of the STPPN of interest.

In this section, an unfolding algorithm used to unfold the original STPPN in Wiley [75] is presented for the purpose of obtaining an ordered index. Then, a functioning unfolded STPPN is shown. Finally, equivalency of two evolutions of the original STPPN and of the corresponding unfolded STPPN is proven.

4.2.1 Unfolded STPPN construction

An unfolded STPPN is constructed by cutting every simple directed circuit[2] at an appropriate transition obtained from the original STPPN, unfolding, and con-

[1]A simple direct circuit indicates that a finite chain with the first and last nodes being coincident in which no nodes (transitions or places) in the path are repeated.

[2]In order to find all simple directed circuits in a net (or a graph), Martinez and Silva's algorithm [44] is used.

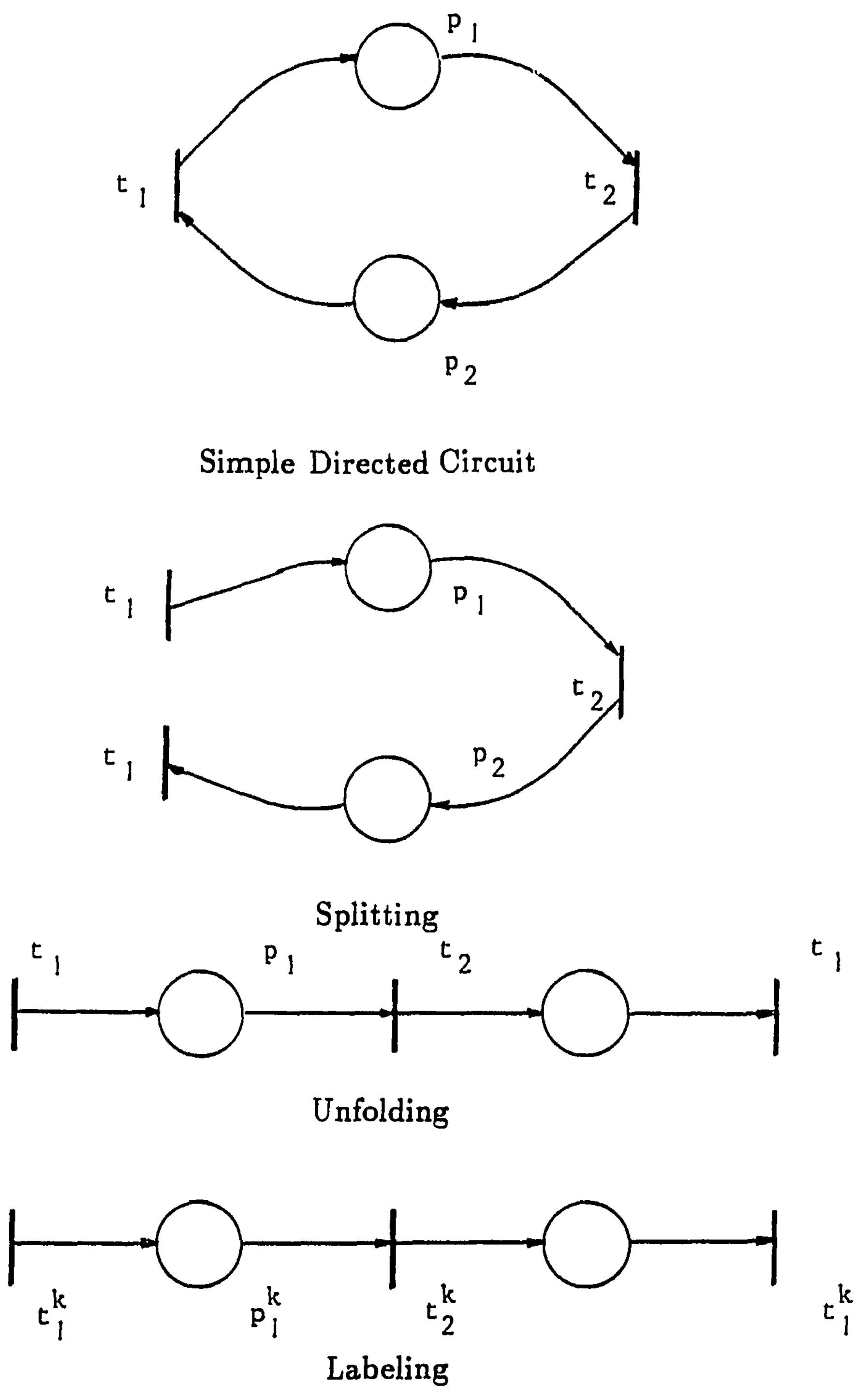


Figure 4.1: Example of splitting, unfolding, and labeling of a simple directed circuit

necting each circuit together in such a manner that no simple directed circuits are formed in the decomposed net. To split each circuit, care must be taken because wrong selection of the transition which cuts each circuit may not only break the causality of the original STPPN, but also result in altering the firing orders even though it may not break the causality. To make the evolution of the corresponding unfolded STPPN identical with the original STPPNs, two simple additional procedures are required.

One procedure is the labeling of unfolded, simple directed circuits. By splitting the selected transition into two copies and then unfolding it, a directed sequence of places and transitions beginning and ending at the same transition will be obtained. The labeling of a circuit is accomplished by labeling transitions and places in the circuit with the superscript k, except for the last transition. The last transition is labeled with the superscript k+1. Examples of splitting, unfolding, and labeling of a circuit are shown in Figure 4.1.

The second procedure is the union of unfolded STPPNs. Since each circuit can be considered as a small STPPN, the only problem to resolve is the combining of the small STPPNs. Suppose there are n STPPNs characterized by the quadruples $N_1, N_2, \cdots, N_n$, then the union of the n STPPNs is defined by the quadruple $\mathbf{N}$ [20]. An example is shown in Figure 4.2.

In the case of combining unfolded circuits, the above definition can be directly applied as shown in Figure 4.3. The algorithm to construct unfolded STPPNs follows with an illustration of how the algorithm works.

4.2.2 Algorithm: Construction of unfolded STPPNs

Step 1: Find all simple directed circuits from a given regular STPPN.

Step 2: Cut each circuit at an appropriate transition and copy that transition. Then, attach a copied transition to the other edge which does not contain the transition.

Step 3: Select, label, and unfold a circuit. If there exist circuits that are self-looped, choose one and make it a frame. If no self-looped circuits exist, then choose the longest circuit which contains the transition that will fire first.[3]

Step 4: Select, label, and unfold another circuit, and connect it to the existing frames. If there is no frame to be connected, then make another frame.[4]

Step 5: Continue Step 4 until there is no circuit left in the set of simple direct circuits.

Step 6: If there are no circuits left in the set, check the number of places for each transition. If the number of places for a transition is the same as in the original STPPN, then the algorithm is terminated. Otherwise, add places at the transition which has a different number of places to the original STPPNs and label using the following ways:

a. Label superscript k-1 for the place that is the input place of the transition with superscript k.

b. Label superscript k+1 for the place that is the input place of the transition with superscript k+1.

In Figure 4.4, by using the algorithm which was developed by Martinez and

[3]In practice, the problem is choosing a specific circuit to be unfolded first. The selection of a self-looped circuit or the longest one is a matter of convenience.

[4]Such a circuit will be connected before all circuits are selected.

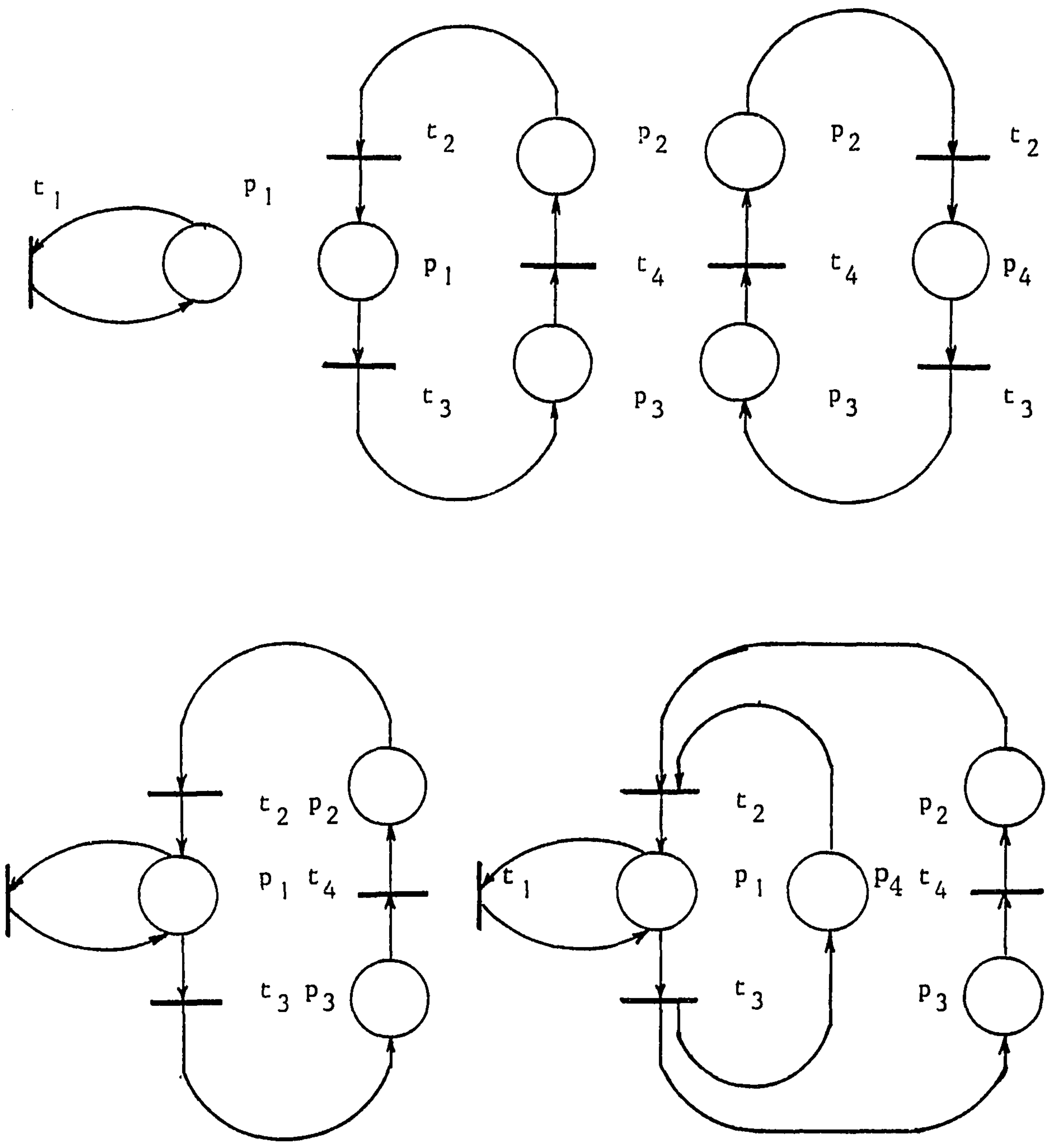


Figure 4.2: Example of union of STPPN

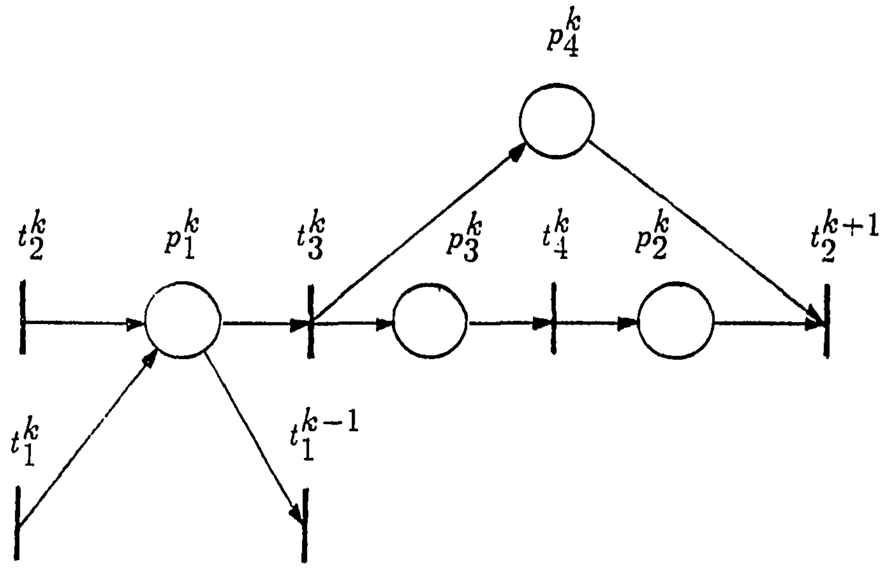


Figure 4.3: Unfolded graph of the regular STPPN in Figure 4.2

Silva [44], six simple directed circuits can be found. According to the algorithm, the circuit selected first is the 6th circuit which is self-looped under the assumption the STPPN of interest is safe. Note how an unfolded circuit is created in Step 3, and how extra places are added and labeled in Step 6 (see Figure 4.5).

The algorithm works by taking one circuit from the set at a time, unfolding and combining it at an appropriate transition. In the aligning process, circuits may be created inadvertently. However, since the algorithm selects a transition at which it must split when a circuit is considered and then splits all other circuits containing the selected transition at that transition, it is not possible to create circuits in the unfolded net after all circuits are connected by the algorithm. Therefore, the resulting net, i.e., the unfolded STPPN is circuit-free as mentioned previously.

As seen in the previous subsection, by specifying the value of the superscript k varies from 1 to infinite, an infinite collection of the unfolded STPPN can be obtained, since all places and transitions in the net are labeled with superscripts k-1, k, and k+1.

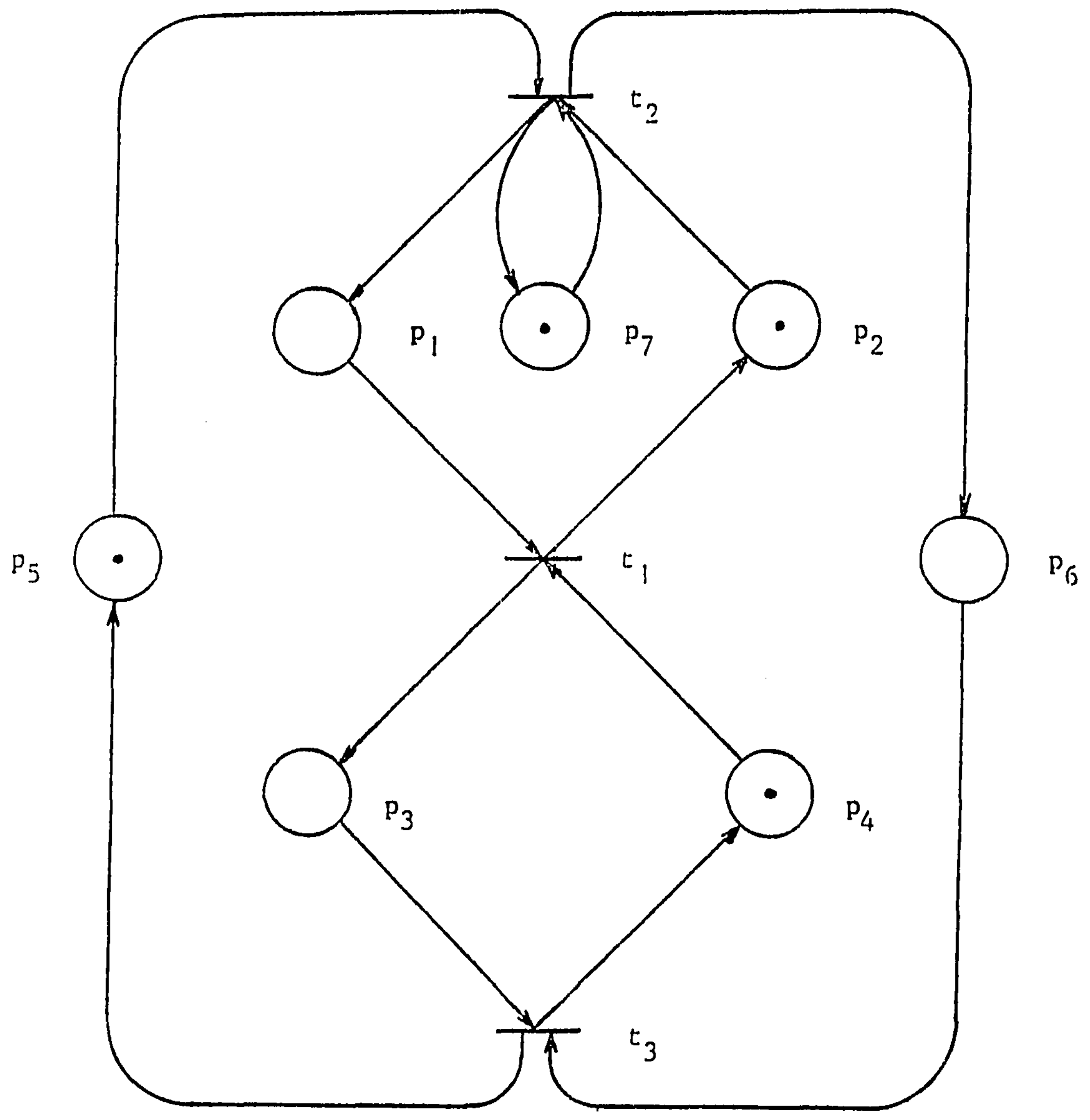
4.2.3 Evolution of unfolded STPPN

Since the operation of the regular STPPNs can be viewed in terms of the operation of this infinite collection, what remains to be considered is how an unfolded STPPN evolves.

Operation rules of unfolded STPPNs are basically identical to those of the regular STPPN in token advancing, i.e., if tokens in all input places for a transition are active, then they move to output places of the transition just fired. However, since unfolded STPPNs evolve one state at a time and thus the state transition is performed when the previous state reaches a deadlock,[5] a transition which contains active tokens in all its input places may prevent its firing.

As an illustration, consider the STPPNs in Figure 4.6. Initially, both STPPNs contain tokens at places p_3, p_5, p_6, and p_7. Initializing the corresponding processing times of those places as zero time units and selecting the processing times of places p_1, p_2, and p_4 as 1, 2, and 0 time units, respectively, the unfolded STPPN evolves as follows. After transition t_1^1 fires at time 0^+, tokens will be added places p_1^1 and p_2^1, one token for each place. Then the token in place p_2^1 will leave state 1 at time 2. Transition t_2^1 cannot fire since there is no token in place p_3^0. Eventually, the

[5] *Deadlock* used here has a different meaning rather than *deadlock* presented in Chapter 2. The latter indicates the ill-formed conversion of the system to the STPPN. Meanwhile, the former indicates that the evolution of the net is temporarily stopped since the unfolded net can evolve one stagt at a time.



List of Simple Directed Circuits:

(1) $t_2p_1t_1p_2t_2$ (2) $t_2p_6t_3p_5t_2$

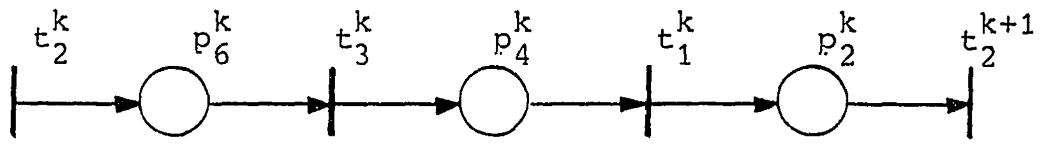
(3) $t_3p_4t_1p_3t_3$ (4) $t_2p_1t_1p_3t_3p_5t_2$

(5) $t_2p_6t_3p_4t_1p_2t_2$ (6) $t_2p_7t_2$

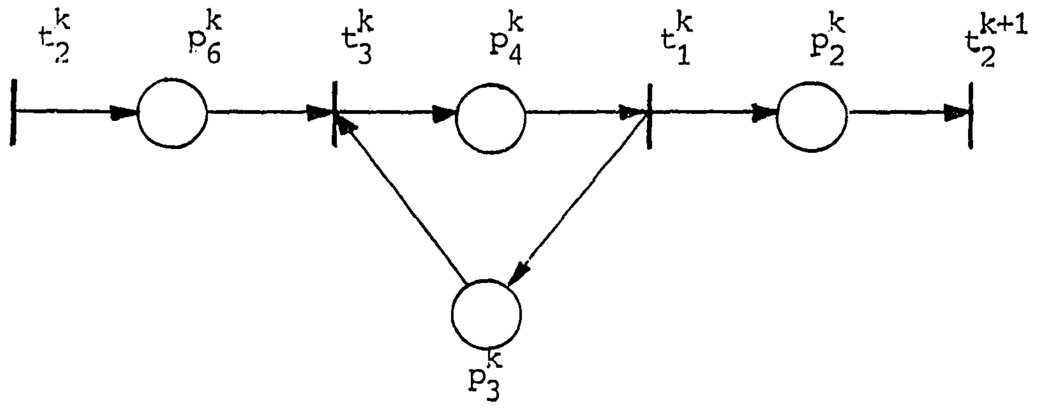
Figure 4.4: Example of the STPPN with a list of simple directed circuits

Step 3:

* Let circuit 5 be the main frame.



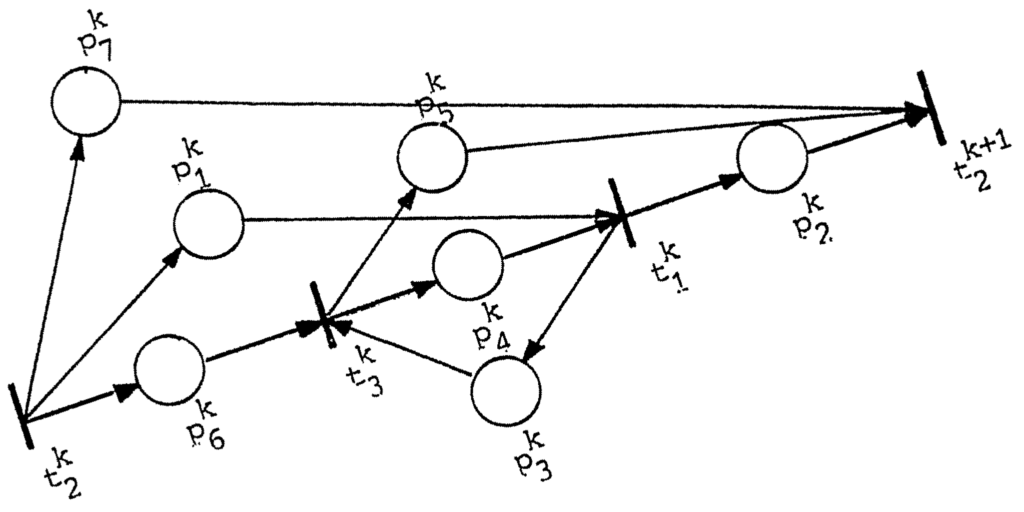
Step 4-5:



* Add circuit 3.

Figure 4.5: Unfolded STPPN construction

Step 5:



* Add circuits 1, 2, 4, and 6.

Step 6:

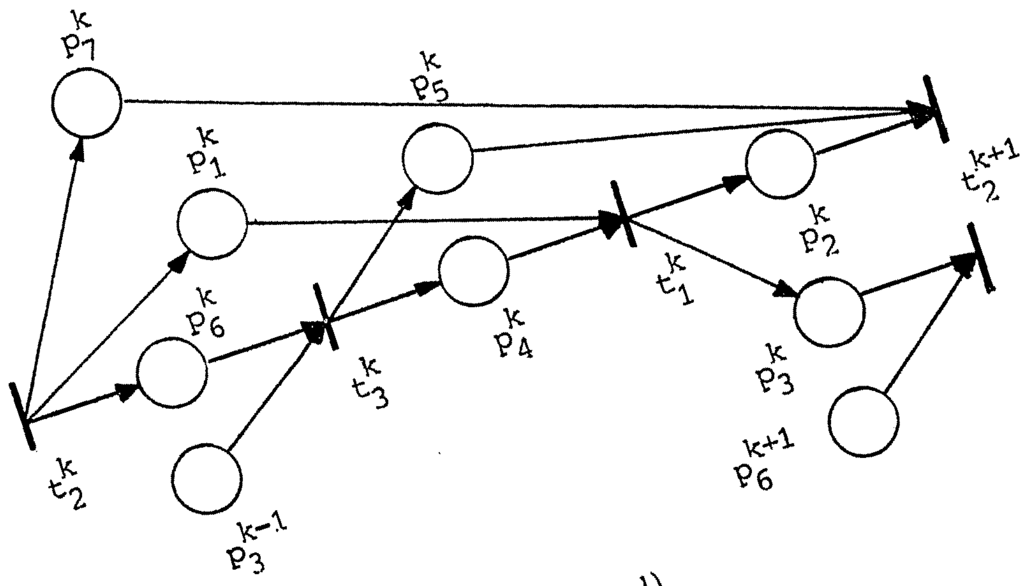
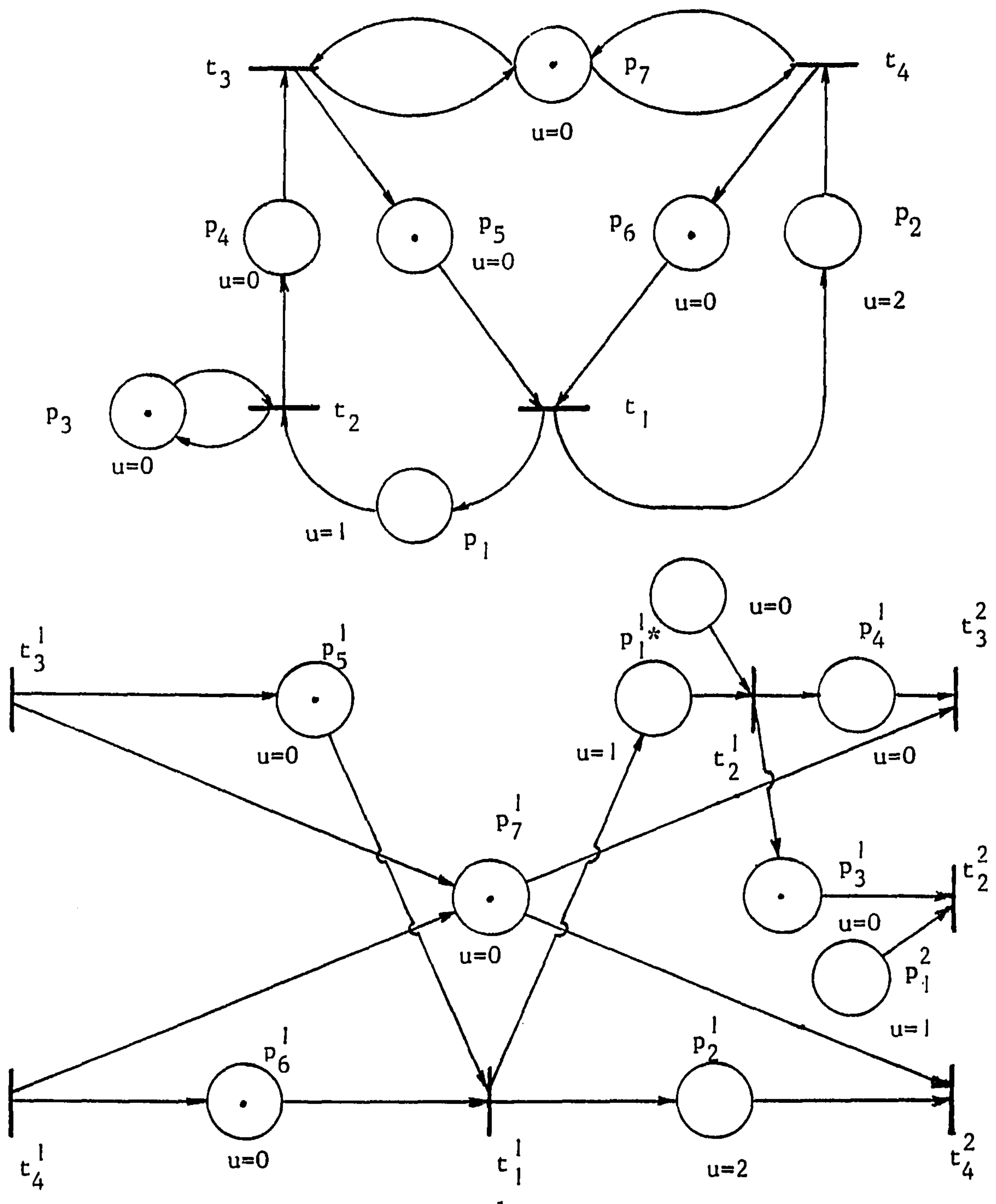


Figure 4.5: (Continued)

token in place p_1^1 is temporarily deadlocked. Meanwhile, in the original STPPN, transition t_2 fires at time 1 and t_4 fires at time 2. That is, an order alteration is occurred in the evolution of the unfolded STPPN (see Figure 4.6). Therefore, some modifications and additions for operation rules addressed in Chapter 2 are needed.

In order to prevent the discrepancy between evolutions of the original STPPN and the corresponding STPPN, two methods are considered. One method is placing a token at place p_3^0 instead of p_3^1, when tokens are copied initially from the original STPPN. The other method is placing a token at place P_3^1 instead of p_3^0 and delaying the firing of transition t_4 until t_2 fires. Both methods are identical in terms of the processing order and the cycle processing time. The former, however, does not need to delay transition t_4 to prevent the order alteration. Therefore, in this thesis, the methods in which placing tokens without delaying is used. Moreover, the unfolding algorithm developed in Subsection 4.2.2 cannot be directly applied to describe the former concept in order to prevent the evolution delay. Therefore, some modifications must be made. However, the manner of transporting tokens from one state to the next state does not need to be changed. For modifying the algorithm to remove discrepancy between the original STPPN and the corresponding unfolded STPPN, refer to the example in Figure 4.7. In step 6 of the algorithm, after adding and labeling places, change the label (superscript) of place p_3 from zero to one, i.e., from p_3^0 to p_3^1 and connect it to transition t_2^1. Change the label of place p_3^1 which is connected to transition t_2^2 from one to two, i.e., from p_3^1 to p_3^2. Then, delay of the evolution can be prevented (see Figure 4.8) and thus no discrepancy exists. For more generalizations, the modified algorithm is formally presented next.



* Deadlocked after firing of transition t_1^1.

Figure 4.6: Discrepancy of evolutions between the original STPPN and the corresponding unfolded STPPN

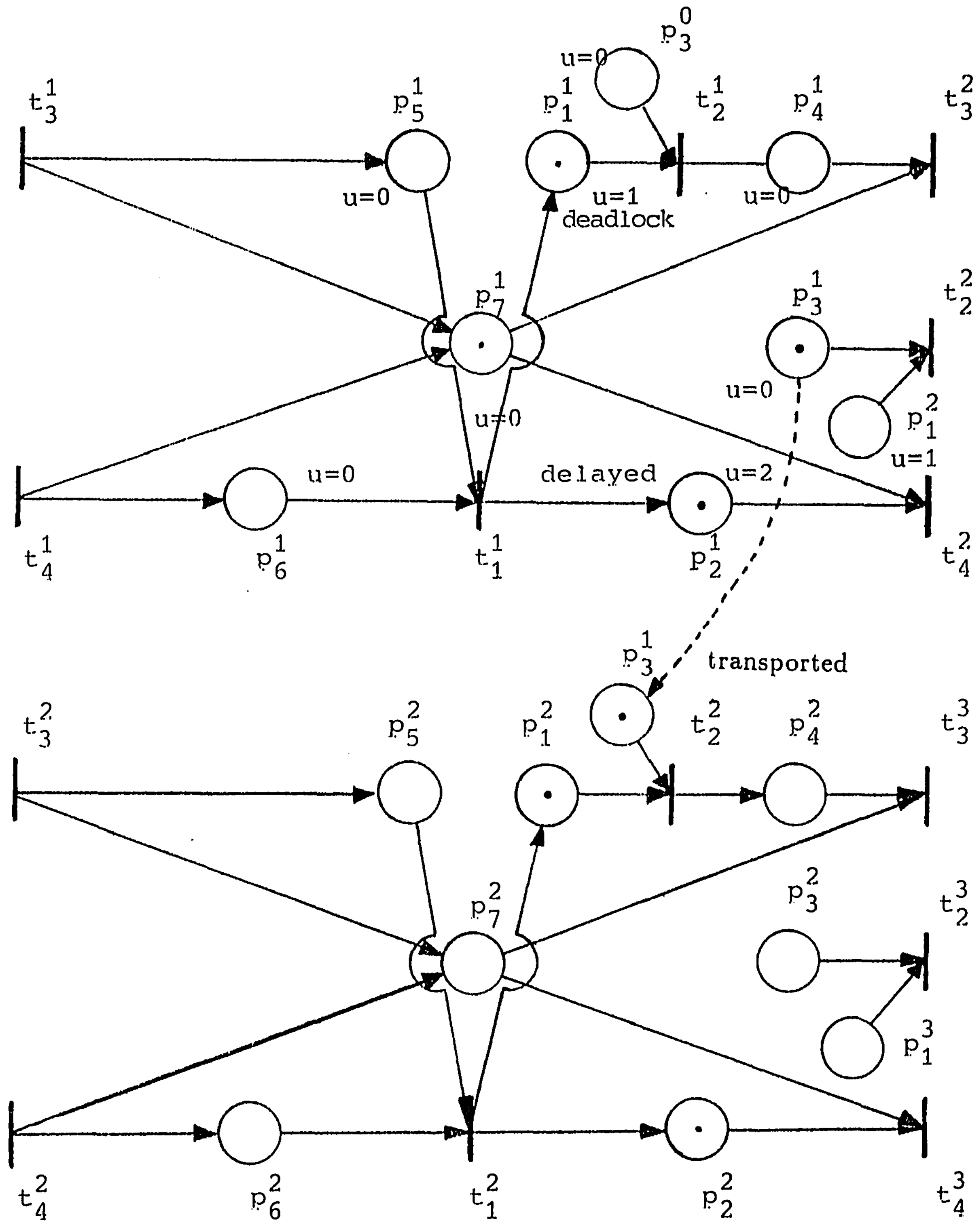


Figure 4.7: Delayed transition and delivery tokens as a state is transited

4.2.4 Algorithm: Construction of unfolded STPPNs with no discrepancy

Step 1: Find all simple directed circuits from a given regular STPPN.

Step 2: Cut each circuit at an appropriate transition and copy that transition. Then, attach a copied transition to the other edge which does not contain the transition.

Step 3: Select, label, and unfold a circuit. If there exist circuits that are self-looped, choose one and make it a frame. If no self-looped circuits exist, then choose the longest circuit which contains the transition that will fire first.

Step 4: Select, label, and unfold another circuit, and connect it to the existing frames. If there is no frame to be connected, then make another frame.

Step 5: Continue Step 4 until there is no circuit left in the set of simple direct circuits.

Step 6: If there are no circuits left in the set, check the number of places for each transition. If the number of places for a transition is the same as in the original STPPN, then the algorithm is terminated. Otherwise, add places at the transition which has a different number of places to the original STPPNs and label using the following ways.

a. Label superscript k-1 for the place that is the input place of the transition with superscript k, but if there exists an input place with a token, then label superscript k instead of k-1 to that place and label k+1 to the same place that is the input place of the transition with superscript k+1.

b. Label superscript k+1 for the place that is the input place of the transition with superscript k+1.

As mentioned previously, the infinite collection of the unfolded STPPN should reflect the original STPPN in its evolution. Therefore, to assure this reflection, the following should be specified when a state transits to the next state: (1) initialization of tokens in each state, and (2) copy of the processing time for each place.

In order to initialize tokens in the next state, two following cases should be considered. One case is that there are tokens left at the previous state, say state k. In this case, memorize the exact transition fired last by those tokens and then initialize tokens in output places of the last fired transitions in the present state, say state k+1. The second case is that there are tokens deadlocked temporarily or delayed in state k. In this case, place tokens in those places of state k+1 if the places with the same superscripts as in the previous state exist in state k+1. If no such places exist, place tokens in corresponding places with superscript k+1. An illustration is shown in Figure 4.8.

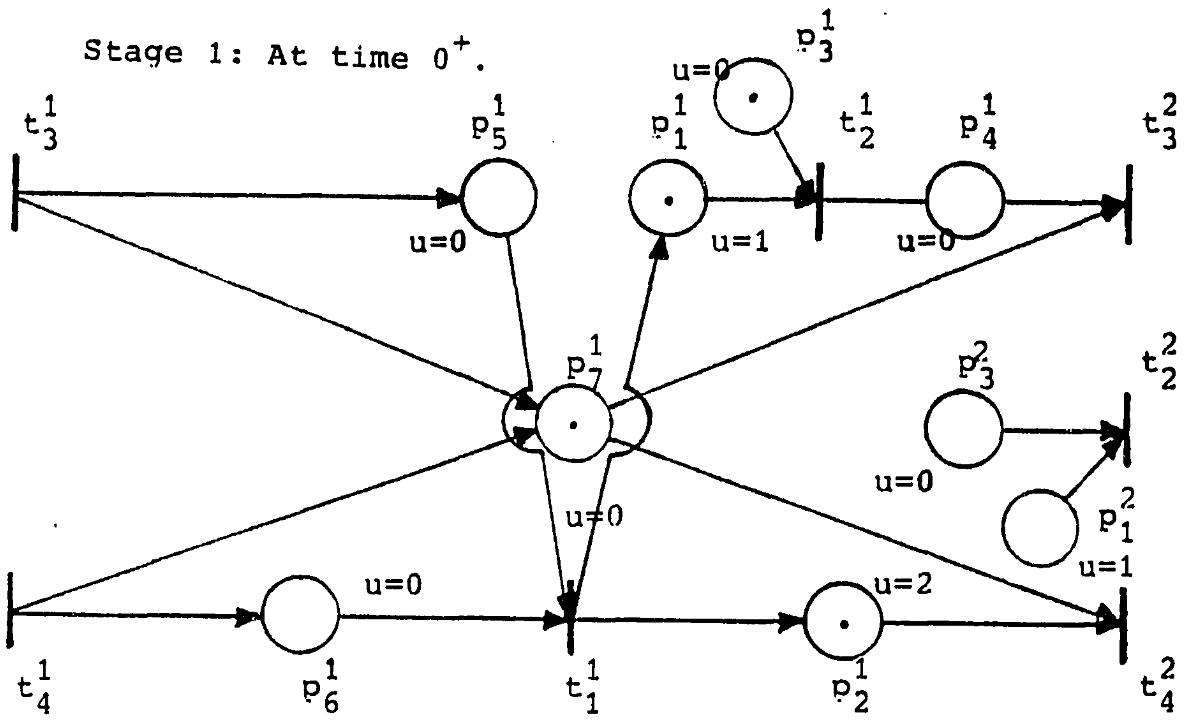
4.2.5 Safeness and Liveness of Unfolded STPPNs

So far in this chapter, the analysis has been based on the assumption of safeness and liveness of STPPNs. This subsection discusses these properties of the regular STPPN for the corresponding unfolded STPPN.

Safeness is easier to investigate. In Chapter 2, an STPPN was defined to be safe if and only if the underlying Petri Net is safe. Since the unfolded STPPN satisfies the same graphical constraints as the original STPPN, the conclusion can be made that the former is also safe.

Liveness is more difficult to address. In the regular STPPN, this property

Stage 1: At time 0^+.



Stage 2: At time 2^+.

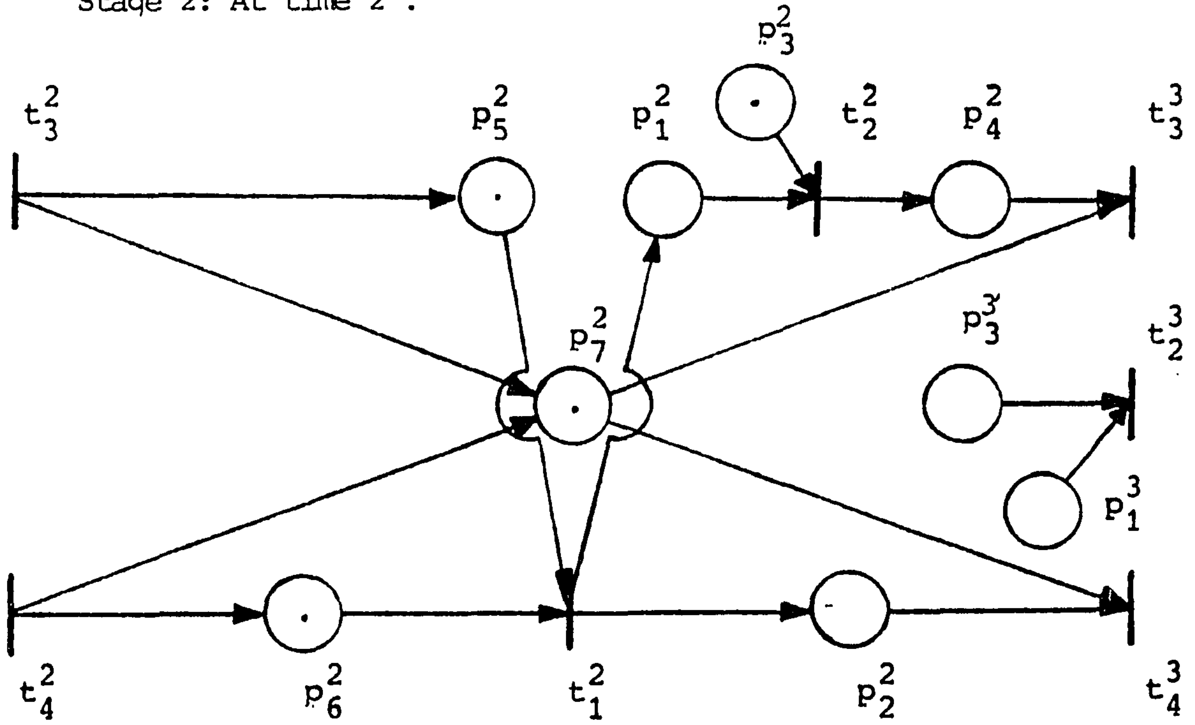


Figure 4.8: Corresponding unfolded STPPN with no discrepancy of an example in Figure 4.6

guarantees that the probability that any particular transition t_i will fire from any state will get arbitrarily close to one, if the evolution time is long enough. In the unfolded STPPN, however, the repeated firings of a given transition correspond to the firings of transition t_i^k, $k = 1, 2, \cdots$. Also, the condition for unfolded STPPN to transit from one state to the next state is that the STPPN reaches deadlock. In other words, all transitions included in the state before a state transition is made will never fire again in that state. Therefore, the unfolded STPPN is not live in this sense. However, since an unfolded STPPN is an infinite collection of small unfolded STPPNs, it can be shown that the probability for some transitions t_i^k to fire will get arbitrarily close to one, if the evolution time is long enough. Suppose that a given unfolded STPPN corresponding to a regular STPPN evolves in state k_0. Then, regardless of what state the unfolded STPPN has reached, for every transition t_i and $\epsilon > 0$, there exists a τ_0 such that

$$P_r\{\text{some transition } t_i^k,\ k \geq k_0 \text{ will fire before } \tau_0\} > 1 - \epsilon \ .$$

if and only if the original STPPN is live. It follows from the one-to-one correspondence between sample paths of the regular STPPN and the corresponding unfolded STPPN.

4.3 Evolution Path Construction of Unfolded STPPN

In order to analyze an STPPN, it is necessary to construct a certain structure[6] which enables explanation of the entire evolution path of an underlying STPPN. First of all, in order to obtain such a structure, all possible state indices that one

[6]The structure is equivalent to the reachability set of the underlying STPPN.

state of the unfolded STPPN can go through should be traced. As previously defined, since a state can be decided by positions of marking and token creation times, all possible state indices must contain the list of places which have tokens and the list of transitions whose firing times are included in that state. A connected form of each possible state according to the order, then, will consist of all possible paths of the underlying STPPN. Therefore, a single evolution path of the STPPN, called a firing process, can be obtained.

4.3.1 Marking process construction

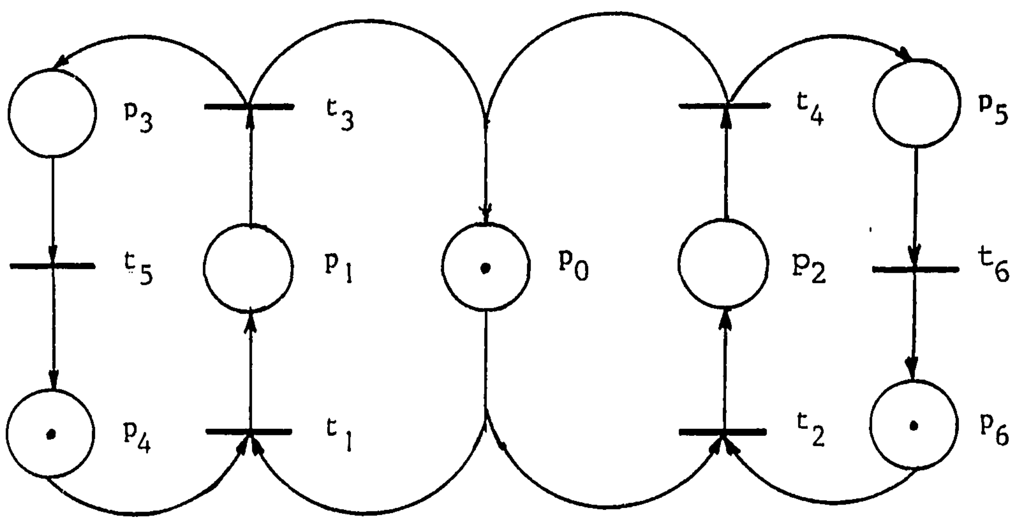
In this subsection, procedures that transform unfolded STPPNs to the marking process are presented. The basic idea used to construct this process begins by setting up the initial state index which indicates the initial state of the underlying unfolded STPPN, which is nominated the marking state, M^o. That is, M^o contains a list of marked places and a list of transitions which create tokens of marked places. From the marking in this index, several transitions have tokens in their input places. Then, a transition among output transitions of marked places is selected and fired. This transition firing follows a new state index. The new state index includes a transition just fired and output places of that transition in addition to places that still have a token in spite of the transition firing and their input transitions. That is, from the initial state index, input places which lost tokens by a transition firing and their input transitions are ruled out. Instead of these places and their input transitions removed, new marked places and the transition which just fired are added. By this way, a set of all possible marking states is created. Then, each marking state is connected to the previous marking state with a direct arc labeled with the transition just fired. From Figure 4.9, the first marking state, M^o, can be easily obtained from the initial marking state of the STPPN under consideration. Then, since transitions t_1 and t_2 have tokens in all their input places, new marking states M^1 and M^2 consist of places which have inactive tokens and their corresponding input transitions, i.e., a just fired transition. Also, new marking states include tokens of marking state M^o whose statuses are unchanged by firing t_1 and t_2. After creating new marking states, the marking state set is connected with directed arcs labeled with just fired transitions, i.e., from M^o to M^1 and to M^2 with labels t_1^k and t_2^k, respectively. In the same way, the marking state creation continues until the new marking state coincides with the initial marking state, M^o. In other words, the final marking state created in a state is an exact replica of the marking state M^o. Then, the resulting set of marking states forms a net and provides all possible sample paths of the underlying STPPN (see Figure 4.10).

As described above, in the marking state creation from the previous marking state, all possible decisions must be represented by choosing transitions which lead to the new marking state. In this example, since t_1 and t_2 are in conflict, two marking states are obtained as shown in Figure 4.10. The formal algorithm to construct the marking process is given next.

4.3.2 Algorithm: Marking process construction of the STPPN

Step 1: Set up initial marking state M^o from the initial marking positions.

Step 2: Create new marking state(s) by firing transitions which have tokens in their input places. In this step, care must be taken. First, if in a single transition set there are more than one possible enabled transitions, create for a new marking state



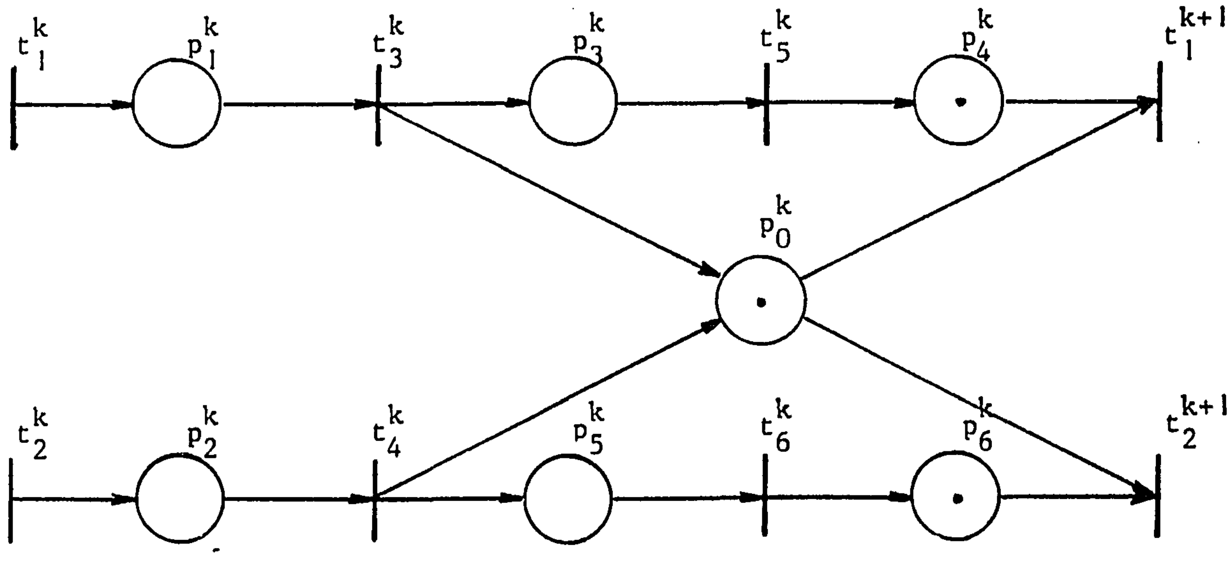


Figure 4.9: An example of STPPN and the corresponding unfolded STPPN

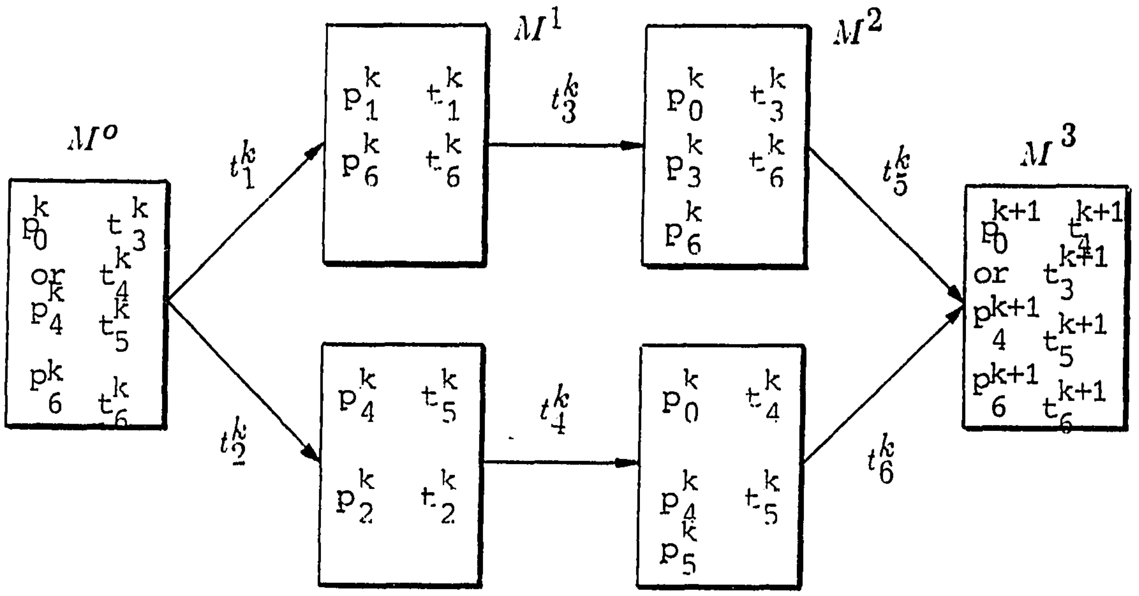


Figure 4.10: The k^{th} state marking process of Figure 4.9

for each enabled transition. The reason is that since the marking process does not contain time, which transition will fire first cannot be known. Second, if enabled transitions are in a non-trivial conflict transition set, fire all enable transitions and create a new marking state for each fired transition.

Step 3: Connect new marking state(s) from marking state M^o to each new marking state by directed arcs labeled with just fired transitions.

Step 4: Continue Steps 2 and 3 until marking state M^o is repeated.

Step 5: Label all places and transitions in all markings of the state with superscript k except the final marking. Then, label places and transitions in the final marking with superscript k−1.

As shown above, the algorithm is generalized by labeling superscript k. There-

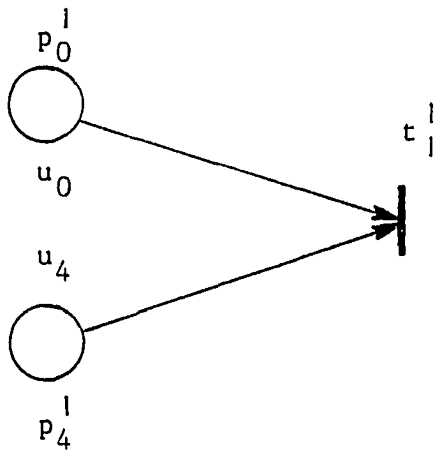


Figure 4.11: Part of Figure 4.9

fore, a sample path of the infinite collection of the unfolded STPPN can be represented by a single path through the infinite collection of the marking process.

4.3.3 Transition firing times and firing processes

The firing process augments the marking process with token creation times. Once the marking process is constructed, from the initial marking state, M^o, which may have several output marking states, exactly one transition among all outgoing transitions will be selected by the transition selection rules. Therefore, the firing time of that transition can be easily deduced by adding the token creation time to the processing time for each of the tokens absorbed by the transition firing. In Figure 4.10, the firing time of the transition t_1 at the first state is

$$S_1^1 = \text{Token creation time} + max\{u_0, u_4\}.$$

After a selected transition fires, another transition is selected from the new marking state and by the same procedure the next transition firing time can be obtained.

By iterating this procedure, the firing process is constructed. Consequently, the simulation of the net can be performed by using this process.

This procedure has many advantages. First of all, by the construction of the firing process, not all possible marking states are needed in the marking process. Second, this results in a smaller number of states needed to characterize STPPNs than would be necessary if they were analyzed in real time. Third, since only one transition can fire at one time at each state of the marking process, this makes it easier to obtain equations of the firing time. Finally, the firing process provides an ordered index which corresponds to the number of transitions fired since the evolution has begun.

4.4 Derivation of Firing Time Equations

In the previous three sections, the conversion was discussed of the regular STPPN into the corresponding firing process. In this section, equations of transition firing times are derived.

In order to derive firing time equations, the marking process constructed in Figure 4.10 is used. Assume that all tokens in places of the net are initially active and the evolution starts at time zero. Also, assume that transition t_1 fires first and denote the instant of time that t_1 fires at state 1 as S_1^1. Then, the firing sequence of state 1, $(S_1^1\, S_3^1\, S_5^1)$ is

$$S_1^1 = 0 \tag{4.1}$$

$$S_3^1 = u_1^1 \tag{4.2}$$

$$S_5^1 = S_3^1 + u_3^1 \tag{4.3}$$

Adding equations (4.1), (4.2), and (4.3),

$$S_5^1 = S_1^1 + u_1^1 + u_3^1 \ .$$

At stage 2, transition t_2 must fire first. Also, the firing instant of t_2 is determined by

$$max\{S_5^1 + u_4^1,\ S_3^1 + u_0^1\} \ .$$

Therefore, the firing sequence of stage 2, $(S_2^2\, S_4^2\, S_6^2)$ is

$$S_2^2 = max\{S_5^1 + u_4^1,\ S_3^1 + u_0^1\} \tag{4.4}$$

$$S_4^2 = S_2^2 + u_2^2 \tag{4.5}$$

$$S_6^2 = S_4^2 + u_5^2 \ . \tag{4.6}$$

Adding equation (4.3) to (4.4),

$$S_2^2 = max\{S_1^1 + u_1^1 + u_3^1 + u_4^1,\ S_1^1 + u_0^1 + u_1^1\} \ .$$

Let δ_{11}, δ_{12} be $u_1 + u_3 + u_4$, $u_0 + u_1$, respectively. Then,

$$\begin{aligned} S_2^2 &= max\{S_1^1 + \delta_{11}^1,\ S_1^1 + \delta_{12}^1\} \\ &= S_1^1 + max\{\delta_{11}^1, \delta_{12}^1\} \ . \end{aligned} \tag{4.7}$$

Adding equation (4.5) to (4.6),

$$S_6^2 = S_2^2 + u_2^2 + u_5^2 \ . \tag{4.8}$$

At stage 3, transition t_1 fires again. Also, its firing time is determined by

$$max\{S_6^2 + u_6^2,\ S_4^2 + u_0^2\} \ .$$

Therefore, the firing sequence of state 3, $(S_1^2\, S_3^2\, S_5^2)$ is

$$S_1^3 = max\{S_6^2 + u_6^2,\ S_4^2 + u_0^2\} \ . \tag{4.9}$$

Adding equations (4.5) and (4.8) to (4.9),

$$\begin{aligned} S_1^3 &= max\{S_2^2 + u_2^2 + u_5^2 + u_6^2,\ S_2^2 + u_2^2 + u_0^2\} \\ &= S_2^2 + max\{u_2^2 + u_5^2 + u_6^2,\ u_2^2 + u_0^2\} \ . \end{aligned} \tag{4.10}$$

Let δ_{21}, δ_{22} be $(u_2 + u_5 + u_6)$, $(u_2 + u_0)$, respectively. Then,

$$S_1^3 = S_2^2 + max\{\delta_{21}^2, \delta_{22}^2\} \ . \tag{4.11}$$

By reiterating, the instant of time that the first transition fired at state k can be deduced as

a. If $k = 1$,

$$S_1^1 = 0 \ . \tag{4.12}$$

b. If k is even, i.e., $k = 2n$,

$$S_2^k = S_1^{k-1} + max\{\delta_{11}^{k-1}, \delta_{12}^{k-1}\}, \ \ k = 2n, \ \ n = 1, 2, \cdots \ . \tag{4.13}$$

c. If k is odd, i.e., $k = 2n + 1$,

$$S_1^k = S_2^{k-1} + max\{\delta_{21}^{k-1}, \delta_{22}^{k-1}\}, \ \ k = 2n + 1, \ \ n = 1, 2, \cdots \ . \tag{4.14}$$

Equations (4.12)-(4.14) are based on the assumption that transition t_1 fires first at stage 1. In the reverse case, i.e., if t_2 fires first, then equations (4.12)-(4.14) can be rewritten as

a. If $k = 1$,

$$S_2^1 = 0 \ . \tag{4.15}$$

b. If k is even, i.e., $k = 2n$,

$$S_1^k = S_2^{k-1} + max\{\delta_{21}^{k-1}, \delta_{22}^{k-1}\}, \quad k = 2n, \quad n = 1, 2, \cdots \quad . \tag{4.16}$$

c. If k is odd, i.e., $k = 2n + 1$,

$$S_2^k = S_1^{k-1} + max\{\delta_{11}^{k-1}, \delta_{12}^{k-1}\}, \quad k = 2n + 1, \quad n = 1, 2, \cdots \quad . \tag{4.17}$$

Equations (4.11)-(4.13) and (4.14)-(4.16) show that the state of the present stage can be deduced if the state of the previous stage is known. Interestingly enough, these equations include the processing times of the circuits of the net (see Figure 4.10). That is, it implies that the initial state of the new stage is determined by the initial state of the previous stage plus maximum circuit processing time whose circuit is processed at the previous stage. Also, since the interval $(S^k - S^{k-1})$ means a cycle time of the net at stage k,

$$S^k - S^{k-1} = max\{\delta_{11}^{k-1}, \delta_{12}^{k-1}\}, \quad k = 1, 2, \cdots \quad .$$

This fact implies that the cycle processing time of the net corresponds to the maximum circuit processing time of the net. The resulting fact is formally proven in Chapter 5.

In this section, differences between the real time simulation and the event simulation have been discussed. Also, the methodologies to decompose the STPPN model into an infinite number of states has been developed. By using methodologies and knowledges gained so far, the algorithm to simulate the system behavior can be developed. This algorithm can be easily extended from the real time simulation algorithm developed in Chapter 2. Next, the algorithm is introduced.

4.5 Event Time Simulation Algorithm

Step 1: Construct initial markings, $\vec{M_a}$ and $\vec{M_i}$, and the time vector[7], $\vec{W_i}$.

Step 2: Construct the transition set enabled by $\vec{M_a}$.

$$T_f = \{t_i | \ t_i \text{ is enabled by } \vec{M_a}\} \ .$$

Step 3: If T_f is empty, then

Step 3.1: Find the minimum w^* from all positive elements of $\vec{W_i}$.

$$w^* = min\{w_1, w_2, \cdots, w_p\}, \ w_p(>0) \in \vec{W_i} \ .$$

Step 3.2: Subtract w^* time units from all positive elements of $\vec{W_i}$.

$$w_k \longleftarrow w_k - w^*, \ \ w_k(>0) \in \vec{W_i} \ .$$

Step 3.3: If w_k equals to zero, then

$$(m_i)_k \longleftarrow (m_i)_k - 1$$
$$(m_a)_k \longleftarrow (m_a)_k + 1 \ .$$

Step 4: If T_f is non-empty, then select $t_s \in T_f$ by transition selection rules.

Step 5: For all $in\{t_s\}$,

$$m_a \longleftarrow m_a - 1 \ .$$

Step 6: For all $out\{t_s\}$, determine $u_{out\{t_s\}}$ from the processing time distributions.

Step 7: If $u_{out\{t_s\}}$ is zero, then

$$m_a \longleftarrow m_a + 1 \ .$$

[7]The vector represents the time left to serve.

Otherwise,

$$\vec{W}_i \longleftarrow +u_{out\{t_s\}} \text{ and } m_i \longleftarrow +1 \quad .$$

<u>Step 8</u>: Repeat Steps (2)-(7).

5 SIMULATION

5.1 Introduction

In the previous three chapters, systems of interest were modeled and their characteristics were discussed from several points of view. In this chapter, the basic needs required to simulate the underlying C^2 models in order to analyze their effectiveness are established.

Before analyzing the behavior of the model, it is necessary to analyze how different operations take place in the process. To do this, Section 5.2 analyzes concurrent and sequential operations in the system. Also, in order to evaluate and compare the relative performance of each underlying system, three major measures of system effectiveness (MOEs): (1) the average cycle processing time, (2) the maximum system throughput rate, and (3) the average response time are derived and their relations are discussed in Section 5.3.

In Section 5.4, when defense is made against the threats of the counterforce, assumptions required to evaluate and to compare the underlying C^2 models are presented. Also, parameters characterizing models are defined. In addition to these, changes of system parameter values due to changes of system structures by internal and/or external shocks such as hit by attacks of the counterforce and malfunction of hardware are discussed. In simulating these models, the two following assumptions are taken. First, no task performing errors and communication errors exist between processes. Second, the communication time between processes is neglected [1] when processing times are considered.

Finally, inputs required to simulate two STPPN models are prepared. Changes of parameter values according to changes of system structure due to shocks are considered and discussed.

5.2 Analysis of Concurrent and Sequential Processes

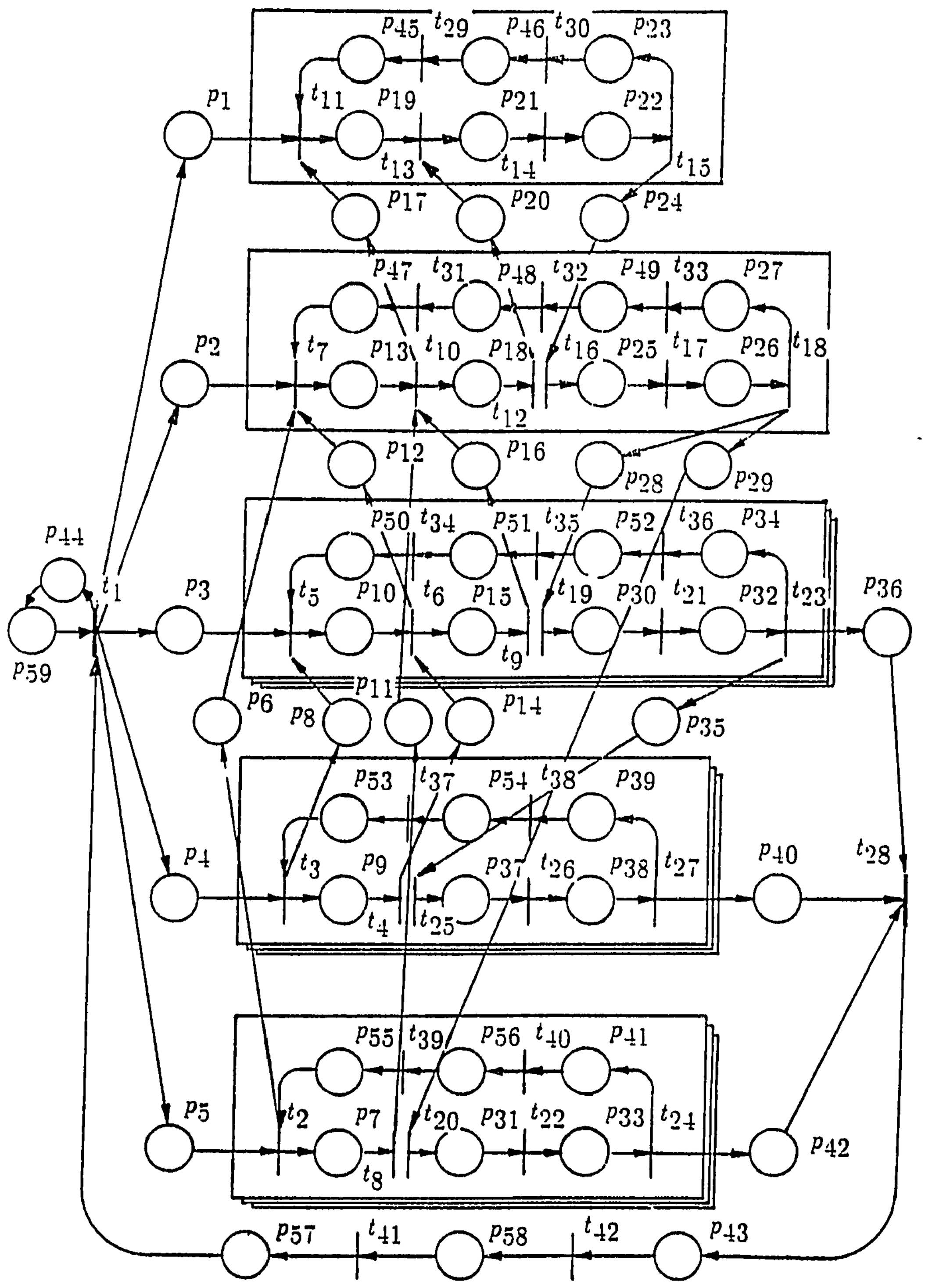
The dynamic behavior of the model can be analyzed by considering the process sequence in the model. Therefore, finding a way to track the operation sequence of the process in the system is very important in the analysis of its behavior, partly through the transformations of the STPPN discussed in Chapter 4. The precise sequence of operations in the process is indeed determined completely by the system structure of the net because the structure of the Petri net characterizes the causality of the system. Therefore, an operation (a place) can be said to be concurrent if its processing is independent, whereas an operation is said to be sequential if it must be performed before other operations. Apart from task processing times and capacity possessed, this partial order in the different operations is clearly a determinant factor of the firing schedule.

[1]Practically, the communication time between two combat units (C^2 processes) is very little in comparison with the task processing time due to the aid of highly developed electronic communication devices.

5.2.1 Slices and lines

As stated in Subsection 2.2.1, the places of the Petri Net that contain a positive number of tokens is called *marked places*. Assume there are no initial marked places in the net, i.e., no capacity places exist, and denote the input transition of the net, t_1 as a set, T_1. Assuming that t_1 fires, let P_1 be the resulting set of marked places, which are also the output places of t_1. Then, P_1 is $P_1 = \{p_1, p_2, p_3, p_4, p_5\}$ (see Figure 5.1). Let T_2 be the set of transitions enabled by P_1 and assume that all the transitions of T_2 fire, then the next set of marked places is $P_2 = \{p_6, p_7, p_9\}$. Similarly, T_3 will be the set of transitions enabled by P_2, etc. By iterating this procedure, a sequence $T_1, T_2, \cdots, T_s$ such that $T_s = \{t_m\}$ is finally constructed. The resulting sets, what is called *slice sets*, of the model in the Figure 5.1 are:

$T_1 = \{t_1\}$,	$P_1 = \{p_1, p_2, p_3, p_4, p_5\}$
$T_2 = \{t_2, t_3\}$,	$P_2 = \{p_6, p_7, p_8, p_9\}$
$T_3 = \{t_4, t_5, t_8\}$,	$P_3 = \{p_{10}, p_{11}, p_{14}\}$
$T_4 = \{t_6\}$,	$P_4 = \{p_{12}, p_{15}\}$
$T_5 = \{t_7, t_9\}$,	$P_5 = \{p_{13}, p_{16}\}$
$T_6 = \{t_{10}\}$,	$P_6 = \{p_{17}, p_{18}\}$
$T_7 = \{t_{11}, t_{12}\}$,	$P_7 = \{p_{19}, p_{20}\}$
$T_8 = \{t_{13}\}$,	$P_8 = \{p_{21}\}$
$T_9 = \{t_{14}\}$,	$P_9 = \{p_{22}\}$
$T_{10} = \{t_{15}\}$,	$P_{10} = \{p_{24}\}$
$T_{11} = \{t_{16}\}$,	$P_{11} = \{p_{25}\}$
$T_{12} = \{t_{17}\}$,	$P_{12} = \{p_{26}\}$
$T_{13} = \{t_{18}\}$,	$P_{13} = \{p_{28}, p_{29}\}$
$T_{14} = \{t_{19}, t_{20}\}$,	$P_{14} = \{p_{30}, p_{31}\}$
$T_{15} = \{t_{21}, t_{22}\}$,	$P_{15} = \{p_{32}, p_{33}\}$
$T_{16} = \{t_{23}, t_{24}\}$,	$P_{16} = \{p_{35}, p_{36}, p_{42}\}$
$T_{17} = \{t_{25}\}$,	$P_{17} = \{p_{37}\}$
$T_{18} = \{t_{26}\}$,	$P_{18} = \{p_{38}\}$
$T_{19} = \{t_{27}\}$,	$P_{19} = \{p_{40}\}$
$T_{20} = \{t_{28}\}$.	



p_{45} t_{29} p_{46} t_{30} p_{23}
p_1 t_{11} p_{19} p_{21} p_{22}
t_{13} t_{14} t_{15}
p_{17} p_{20} p_{24}
p_{47} t_{31} p_{48} t_{32} p_{49} t_{33} p_{27}
p_2 t_7 p_{13} t_{10} p_{18} t_{16} p_{25} t_{17} p_{26} t_{18}
t_{12}
p_{12} p_{16} p_{28} p_{29}
p_{44} p_{50} t_{34} p_{51} t_{35} p_{52} t_{36} p_{34}
t_1 p_3 t_5 p_{10} t_6 p_{15} t_{19} p_{30} t_{21} p_{32} t_{23} p_{36}
p_{59} t_9
p_{11} p_6 p_8 p_{14} p_{35}
p_{53} t_{37} p_{54} t_{38} p_{39}
p_4 t_3 p_9 p_{37} t_{26} p_{38} t_{27} p_{40} t_{28}
t_4 t_{25}
p_{55} t_{39} p_{56} t_{40} p_{41}
p_5 t_2 p_7 t_{20} p_{31} t_{22} p_{33} t_{24} p_{42}
t_8
p_{57} t_{41} p_{58} t_{42} p_{43}

Figure 5.1: STPPN model of a C^2 system

Next, the dependency and the independency of the sequence $\{T_i\}$, $i = 1, \cdots, s$, are analyzed.

Let $<$ denote the partial order relation and n_i, n_j be any two nodes (places or transitions) of the net, then

$n_i < n_j$ if and only if there exists a directed path going from n_i to n_j.

Naturally, the above relation is consistent, since the net has no directed circuits, i.e., the net is acyclical and thus no direct path satisfying $n_i < n_j$ and $n_i > n_j$ exists.

Line: A *line* is a set L of nodes such that

(1) For any two nodes $n_i \in L$, $n_j \in L$,

$$n_i < n_j \text{ or } n_j < n_i \ .$$

(2) L is maximal if and only if there does not exist a node n_k in the net satisfying (1), not belonging to L.

Slice A *slice* is a set S of nodes such that

(1) For any two nodes $n_i \in S$, $n_j \in S$,

$$n_i \not< n_j \text{ or } n_j \not< n_i \ .$$

(2) S is maximal if and only if there does not exist a node n_k in the net satisfying (1), not belonging to S.

Clearly, the elements of a slice are independent with respect to $<$. Therefore, they characterize the concurrent activities of the process. In contrast, the elements of a line are strictly dependent and thus they characterize the sequential activities of the process. In this thesis, only slices are of interest since they are needed to determine the concurrent tasks in the process.

5.2.2 Characteristics of slices

Slices have several characteristics which are useful for analyzing the STPPN. In this subsection, with descriptions of the slice characteristics, methodologies to analyze the model by using them are discussed.

First, the sequence of the slices $T_1, T_2, \cdots, T_S$ represents the steps of the process with the places that process concurrently at each step. The order of these slices determines precisely the partial order between the various processing tasks in the following sense. If a line is considered, then each place on the line belongs to a different slice. Therefore, each place is processed according to the order of corresponding slices, which corresponds to the order in which the places are processed in the sequential subprocess. Therefore, by using this property, the firing sequence can be obtained easily.

Second, slices determine the maximum capacity available to a process or the system itself. Since a slice consists of processes executed concurrently, all processes in a slices operate with information transferred from the same input. For instance, in Figure 5.1, if P_1 is performing now, it implies that $p_1, \cdots, p_5$ possess tokens. However, the tokens in those processes are actually originated from T_1, although the information brought by them is different to the original one, i.e., they were created from the same token. Also, there is no way for places in P_2 to have tokens before transitions in T_2 is processed. Therefore, the number of slices characterizes the upper bound of the system capacity. In other words, the number of slices implies the maximum number of tasks capable of being processed in the system at certain instant.

Finally, since the processes in a slices occur asynchronously in real time and the set of slices determines the execution order of the processes, it is possible to identify which process takes place at the latest instant, compared to other concurrent tasks in the same slice. In fact, this process is critical for the processing delay. Therefore, the model effectiveness can be improved by reducing its processing time.

5.3 Derivations of MOEs

This section develops three major measures of system effectiveness: (1) the average cycle processing time, (2) the maximum system throughput rate, and (3) the average response time. In deriving measures, the conflict situations are not considered since the underlying models dealt with in this thesis are conflict-free as seen in Figures 3.7 and 3.8.

5.3.1 Average cycle processing time, θ

In this subsection, the average cycle processing time is considered. Intuitively, the average cycle processing time is the same for all transitions and places in the cycle. Assume that the processing starts at $t = 0$ and then occurs repetitively. Denote the instant of time at which the transition t_i initiates its n^{th} firing, with correspondence to the n^{th} processing occurrence of task assigned to $out\{t_i\}$ as S_i^n. Then, $(S_i^n - S_i^{n-1})$ naturally represents the time that has elapsed between the $(n-1)^{th}$ and n^{th} firing of transition t_i. This can be interpreted as a cycle processing time with transition t_i because the cycle is assumed to be consistent and each cycle corresponds to the complete processing of one input. Therefore, the average cycle processing time can be defined as follows.

Definition: The average cycle processing time of transition t_i, θ_i, is defined as

$$\theta_i = \lim_{n\rightarrow\infty} \frac{\sum_{i=1}^{n}(S_i^k - S_i^{k-1})}{n} \quad . \tag{5.1}$$

The above equation can actually be rewritten in a more simple form,

$$\theta_i = \lim_{n\rightarrow\infty} \left(\frac{S_i^n}{n} - \frac{S_i^o}{n} \right) \quad .$$

Since S_i^o is constant (assumed to be zero),

$$\lim_{n\rightarrow\infty} \frac{S_i^o}{n} = 0 \quad .$$

Thus,

$$\theta_i = \lim_{n\rightarrow\infty} \frac{S_i^n}{n} \quad .$$

Next, it is proved that all the transitions (or places) have the same average cycle processing times.

theorem 5.1 *$\theta_i = \theta_j$, for all i and j such that i, $j \in I$ and $i \neq j$, where I is the set of transitions in the cycle.*

Proof Let t_i and t_j be any two transitions in the cycle, then there exists at least one directed path (or a circuit) that contains both transitions because the cycle is strongly connected. Without loss of generality, the circuit can be denoted by

$$\delta = (t_i p_1 t_1 \cdots p_j t_j p_{j+1} \cdots p_{i-1} t_i) \quad .$$

Because the circuit is conflict-free, each place of the circuit has exactly one input and one output transition. Also, the marking of the circuit becomes invariant by transition firings. Let M_l^o denote the initial marking of place p_l, and N_{ij}^o and N_{ji}^o

denote the number of tokens in the places initially belonging to the path from t_i to t_j and from t_j to t_i, then

$$N_{ij}^o = \sum_{l=1}^{j} M_l^o, \qquad N_{ji}^o = \sum_{l=j+1}^{i-1} M_l^o \ .$$

Here, transition firings must be taken into consideration. Previously, a transition firing time is assumed to be zero because it is indeed trivial in comparison with a task processing time. However, there exist slight differences between initiations and terminations of transition firings. Denote the number of initiations and terminations of transition t_i in the time interval $[0,\ t]$ by $I_i(t)$ and $T_i(t)$, respectively. Assume that the processing begins at $t = 0$ and thus, no places are processed. Then at any instant t,

$$I_i(t) \geq T_i(t) \ , \tag{5.2}$$

$$I_j(t) \geq T_j(t) \ . \tag{5.3}$$

Since the initiation of a transition takes one token from its input places and adds one token to its output places, for the transitions t_i and t_j at any instant t,

$$I_i(t) \leq N_{ji}^o + T_j(t), \quad I_j(t) \leq N_{ij}^o + T_i(t) \ . \tag{5.4}$$

From equations (5.2) and (5.4),

$$I_i(t) - N_{ji}^o \leq I_j(t) \ . \tag{5.5}$$

From equations (5.2) and (5.5),

$$I_i(t) + N_{ij}^o \geq I_j(t) \ . \tag{5.6}$$

Consider the instant of time t as

$$t = S_i^{n+N_{ji}^o} ,$$

where n is any positive integer. Then, at this instant,

$$I_i(t) = n + N_{ji}^o . \qquad (5.7)$$

Adding equation (5.7) to equation (5.5) yields

$$I_j(t) \geq n ,$$

which implies that

$$S_j^n \leq S_i^{n+N_{ji}^o} (= t) .$$

Likewise, consider the instant of time,

$$t = S_i^{n-N_{ij}^o} ,$$

where n is assumed to be greater than N_{ij}^o. Then

$$I_i(t) = n - N_{ij}^o , \qquad (5.8)$$

and adding equation (5.8) to equation (5.6) yields

$$I_j(t) \leq n ,$$

which implies that

$$S_j^n \geq S_i^{n-N_{ij}^o} (= t) .$$

Finally, for any positive integer n, such that $n > N_{ij}^o$,

$$S_i^{n-N_{ij}^o} \leq S_j^n \leq S_i^{n+N_{ji}^o} .$$

Hence, dividing by n and multiplying $(n - N^o_{ij})/(n - N^o_{ij})$ and $(n - N^o_{ji})/(n + N^o_{ji})$ for the first and the last term, respectively, then

$$\frac{n - N^o_{ij}}{n} \frac{S_i^{n-N^o_{ij}}}{n - N^o_{ij}} \le \frac{S_j^n}{n} \le \frac{n + N^o_{ji}}{n} \frac{S_i^{n+N^o_{ji}}}{n - N^o_{ji}} \ .$$

Taking the limits as n goes to infinity,

$$\lim_{n \to \infty} \frac{n - N^o_{ij}}{n} \frac{S_i^{n-N^o_{ij}}}{n - N^o_{ij}} \le \lim_{n \to \infty} \frac{S_j^n}{n} \le \lim_{n \to \infty} \frac{n + N^o_{ji}}{n} \frac{S_i^{n+N^o_{ji}}}{n - N^o_{ji}} \ .$$

which means exactly,

$$\theta_i \le \theta_j \le \theta_i$$

and thus $\theta_i = \theta_j$. Q.E.D.

5.3.2 Average circuit processing time, $\alpha(\delta)$

In the previous subsection, it has been proven that any two transitions in a directed circuit have the same average cycle processing time. Now it is possible to determine what this measure would be, assuming that the circuit functions independently of the remaining system. The directed circuit shown in Figure 5.2 is denoted by $\delta_i = (t_1 p_1 t_2 \cdots t_k p_k)$. Assume that the circuit has only one token in it, which is initially in place p_i, $i = 1, \cdots, k$ and initially active, then the processing time of place p_i can take an infinite number of values,

$$U_i = u_i, \quad 0 < u < \infty \ .$$

according to the continuous probability distribution,

$$f_{U}(u) = \int_0^u f(x_i) dx_i, \quad 0 < u < \infty \ .$$

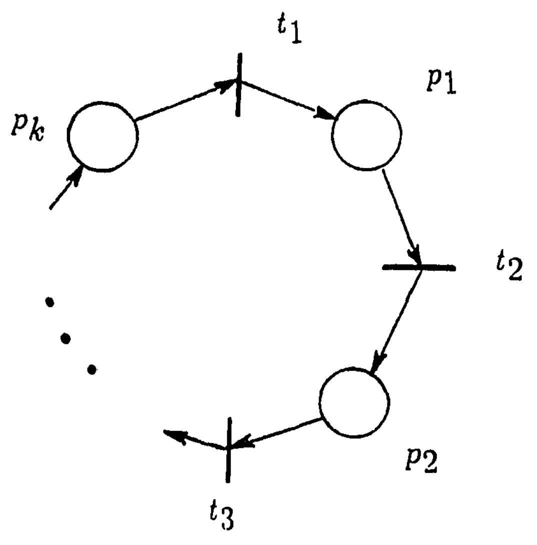


Figure 5.2: Example of directed circuit

since the token content of the circuit is assumed to be one, the expected processing time of p_i can be denoted as

$$\mu_i = \int_0^\infty u_i f(u_i) du_i \quad . \tag{5.9}$$

It turns out that the circuit processing time, i.e., the amount of time it takes for the token to complete the entire process of the circuit, takes values

$$u(\delta) = u_1 + u_2 + \cdots + u_i + \cdots + u_k \quad .$$

Therefore, the expected circuit processing time can be represented as

$$E[u(\delta)] = \sum_{i=1}^{k} \mu_i \quad ,$$

assuming that all place processing times are non-deterministic and have independent probability distributions. Likewise, assumed that there are n tokens in the circuit, then under the given assumption that the probability distribution is independent from one token to the next as stated in Section 2.2.1, the average circuit processing time would be

$$\alpha(\delta) = \frac{\mu(\delta)}{n}, \quad n \leq k \ . \tag{5.10}$$

5.3.3 Maximum average circuit processing time, α

In the previous section, the fundamental assumption has been taken that there is no time delay between the different processing operations because a circuit is assumed to be isolated from the net. Therefore, the average processing time stated above does not consider the firing delays.

Now consider the net as a whole. It is quite clear that the circuit is interconnected with the other. Therefore, it turns out that the interconnected circuits affect each other in processing times, as stated in Section 4.4. For example, in Figure 5.2, transition t_1 belongs to both circuits δ_1 and δ_2. Clearly, the average cycle processing time of this transition cannot be lower than the maximum of $\alpha(\delta_1)$ and $\alpha(\delta_2)$, which are the average circuit processing time of each circuit. Indeed, in order for transition t_1 to fire, both tokens in places p_1 and p_2 should be active. Also, as shown in Section 5.2.1, the average cycle processing time θ is the same for all transitions in the cycle. Therefore, the intuitive result that θ cannot be lower than any average circuit processing times in the cycle can be obtained, i.e.,

$$\theta \geq max\{\alpha(\delta_1), \alpha(\delta_2), \cdots, \alpha(\delta_r)\} \ ,$$

where r denotes the number of directed circuits in the cycle.

Definition: Let δ_1, $\delta_2, \cdots$, δ_r denote all the directed circuits of the net, then the

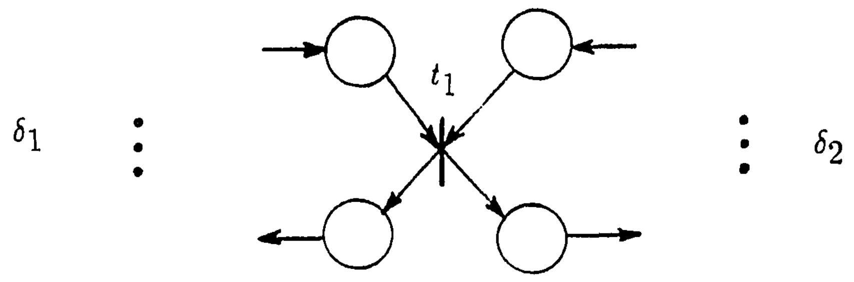


Figure 5.3: Example of interrelated directed circuits

maximum average circuit processing time is defined as

$$\alpha = max\{\alpha(\delta_1), \alpha(\delta_2), \cdots, \alpha(\delta_r)\} \quad , \tag{5.11}$$

where $\alpha(\delta_i)$ is the average processing time of circuit δ_i, as defined by equation (5.10). Also, the minimum average processing rate can be easily obtained by taking inverse of α, i.e.,

$$\frac{1}{\alpha} = min\{\frac{1}{\alpha(\delta_1)}, \cdots, \frac{1}{\alpha(\delta_r)}\} \quad .$$

Note that the minimum is taken over all the directed circuits in the net.

Now, the fact that the average cycle processing time cannot be lower than the maximum circuit processing time will be proven.

theorem 5.2 *The maximum average circuit processing time is a lower bound of the average cycle processing time, i.e., $\theta \geq \alpha$.*

Proof In order to prove the above theorem, it is sufficient to prove that, for any directed circuit δ of the net, $\theta \geq \alpha(\delta)$.

Let δ be any directed circuit denoted by $\delta = (t_1 p_1 t_2 \cdots t_i p_i t_{i+1} \cdots t_k p_k)$. Consider the sequence $S_i^n \, (n = 1, 2, 3 \cdots)$ corresponding to the instant at which transition t_i fires successively. Now, consider any pair of successive transitions (t_i, t_{i+1}) of the circuit that are connected by place p_i. Then, equations (5.2) and (5.3) can be applied for t_i, t_{i+1}, so that, at any instant t,

$$I_{i+1}(t) \leq M_i^o + T_i(t) \ , \tag{5.12}$$

where M_i^o denotes the initial marking of place p_i at time $t = 0$.

Now consider the instant of time,

$$t = S_i^{n - M_i^o} + u_i \ ,$$

where n is any positive integer, such that $n > M_i^o$ and u_i is the random processing time of p_i such that $u_i \geq 0$. Then,

$$T_i(t) = n - M_i^o \ .$$

Using equation (5.12) yields,

$$I_{i+1}(t) \leq n \ ,$$

which implies

$$S_{i+1}^n \geq S_i^{n - M_i^o} + u_i \ . \tag{5.13}$$

Now, by iterating equation (5.13) from $i = 1$ to $i = k$,

$$S_{k+1}^n \geq S_1^{n - M^o(\delta)} + u(\delta) \ , \tag{5.14}$$

where $M^o(\delta) = \sum_{i=1}^{k} M_i^o$ and $\sum_{i=1}^{k} u_i$.

Since $S_1^n = S_{k+1}^n$, equation (5.14) becomes

$$S_1^n \geq S_1^{n-M^o(\delta)} + u(\delta) \ . \tag{5.15}$$

Meanwhile, assume the processing starts at place p_1, the number of firing of transition t_1, n, is determined by

$$n = aM^o(\delta) + b, \quad 0 \leq a \leq n, \ 0 \leq b < M^o(\delta) \ ,$$

where a, b denote the number of processing occurrences of place p_1 and the number of tokens which are not initially in p_1, respectively. That is, all tokens in the circuit must pass through transition t_1 once they reach to p_1 for the first time.

By iterating equation (5.15) for $n = b + M^o(\delta), \cdots, b + aM^o(\delta)$,

$$S_1^n \geq S_1^b + au(\delta) \ , \tag{5.16}$$

By dividing equation (5.16) by n,

$$\frac{S_1^n}{n} \geq \frac{S_1^b}{n} + \frac{au(\delta)}{n} \ .$$

Since $\lim_{n\to\infty} \frac{S_1^b}{n} = 0$ and $a = \frac{n-b}{M^o(\delta)}$,

$$\begin{aligned}
\lim_{n\to\infty} \frac{S_1^n}{n} &\geq \lim_{n\to\infty} \frac{S_1^b}{n} + \lim_{n\to\infty} \frac{\frac{n-b}{M^o(\delta)} u(\delta)}{n} \\
\theta &\geq 0 + \frac{u(\delta)}{M^o(\delta)} \ ,
\end{aligned}$$

when n goes to infinity.

Taking expectations on both sides,

$$E[\theta] \geq \frac{\mu(\delta)}{M^o(\delta)} \ .$$

Since, $\frac{\mu(\delta)}{M^o(\delta)} = \alpha(\delta)$,

$$E[\theta] \geq \alpha(\delta) \ . \tag{5.17}$$

Q.E.D.

5.3.4 Maximum throughput rate, Φ

In the previous section , it has been shown that the maximum circuit processing time is a lower bound of the average cycle processing time. Since the throughput rate is defined as the inverse of the average cycle processing time, these results can be interpreted as follows. The throughput rate of the system, $\Phi = \frac{1}{\theta}$, is bounded from above by the minimum average circuit processing rate, $\frac{1}{\alpha}$. Now, the problem is whether this bound is actually achievable. In other words, whether the minimum average circuit processing rate characterizes the maximum throughput rate of the system must be determined. It is clear that if the arrival rate of external inputs (i.e., arrival rate of the enemy's attack) is low enough, the system will be able to handle all the arriving inputs. Thus, the throughput rate will be precisely the rate at the arriving inputs since the detection rate of the enemy's attack is assumed to be equal to the arrival rate of the enemy's attack in the underlying models. However, there is a rate above which the system will be overloaded in the sense that inputs will queue at the entry of the system. This queue will increase without limits over time. This rate characterizes the maximum throughput rate of the system.

Here, a consideration must be taken. So far, the analysis has been based on the assumption that the initialization of a transition is slightly different to the termination of the corresponding transition. However, as assumed in Chapter 2, transition times are practically the same and have zero processing times. Therefore, it can be stated that the minimum circuit processing rate determines the maximum throughput rate.

theorem 5.3 *The minimum average circuit processing rate determines the maximum throughput rate, i.e.,* $\Phi = \frac{1}{\alpha(\delta)}$.

This theorem make it easy to compute the maximum throughput rate, once all simple, directed circuits of the system are found. In order to determine all simple, directed circuits, Martinez and Silva's algorithm [44] is used. The algorithm is described in Appendix A of the thesis.

As described earlier, the time and capacity constraints are related to the performance evaluation of the system. Next, how these constraints are connected to the maximum throughput rate is analyzed.

<u>Definition</u>: A circuit δ_i is called critical, if its average processing time is maximal over all the simple, directed circuits in the net, i.e.,

$$\alpha(\delta_i) = max\{\alpha(\delta_1), \cdots, \alpha(\delta_r)\} \ .$$

However, the critical circuits are not necessarily the same since the processing times are non-deterministic. Therefore, the maximum throughput rate can be determined by averaging the processing times of critical circuits over the probability distributions.

This fact provides a way for determining which time and capacity constraints are actually binding. Indeed, there are task processing time (capacities) that correspond to the places (token content) of the critical circuits. Therefore, the problem of modifying the right constraints to improve the maximum throughput rate becomes straightforward. That is, since the maximum throughput rate has a lower[2] and an upper bounds. The upper bound is given by $\frac{1}{\alpha}$ from the above analysis. Therefore, if the information arrival (attack) rate exceeds the upper bound, it can be sure that the system will be overloaded. In the reverse case, the system will be able to handle all arrival information. Therefore, by determining the critical circuits of different system structures, comparative studies of system effectiveness can be achieved with respect to the maximum throughput rates.

5.3.5 Average response time, $\overline{RT}$

In the previous section, the maximum throughput rate as a function of the constraints, the processing time and the capacity, has been derived. In this section, the average response time characterizing the dynamic behavior of the system is discussed. In order to derive this measure, the firing sequence which has been introduced in Chapter 4 will be discussed. The firing sequence indicates the instant times at which the various operations occur in the process for allowable rates of incoming inputs. Naturally, the precise firing schedule can only be determined for the deterministic process. However, since this thesis deals with non-deterministic process, the only way to analyze this process is to take average processing times and to treat them as a deterministic process.

5.3.5.1 Feasible firing sequences Firing sequences are feasible if only if the transition can be fired as soon as the corresponding transition is enabled. As

[2]The lower bound of the maximum throughput rate is trivially obtained by assigning each places its worst. But in reality, the lower bound is determined in accordance with the requirement assigned when the system is designed.

stated in Section 2.2, the enabled transition is defined as the transition that has active tokens in all its input places. However, if the output places of the transitions are still processing when the transition is enabled, there is no place for tokens in input places to move out. Hence, the transition cannot fire until the processing of its output place is terminated. Therefore, in order for the firing sequence to be feasible, the definition of enabled transition should be modified as follows.

Definition: A transition is said to be enabled if and only if tokens in all input places of a transition are active and there are no tokens in its output places. Now, the definition of feasible firing sequence used by Ramchandani [57] can be directly used.

theorem 5.4 *The firing sequence is feasible if and only if*

$$S_j^{n+M_{ij}^o} \geq S_i^n + \widetilde{u_{ij}}$$

and

$$S_j^{n+1} \geq S_j^n + \widetilde{u_{j\cdot}} \quad ,$$

where S_i^n is the nth firing time of transition t_i, $\widetilde{u_{ij}}$ is the actual processing time[3] *with transitions t_i, t_j as its input transition and output transition, respectively. M_{ij}^o is the number of tokens in place p_{ij} initially, and $u_{j\cdot}$ is the output place of transition t_j.*

The first relation is intuitively clear. Transition t_i yields n tokens in place p_{ij} and transition t_j takes $n + M_{ij}^o$ tokens from place p_{ij}. Therefore, $S_j^{n+M_{ij}^o}$ should

[3]As pointed out in Section 4.4, $\widetilde{u_{ij}}$ is either the processing time assigned externally or the waiting time due to delay of processing of next places, both.

necessarily be processed after $S_i^n + \widetilde{u_{ij}}$ because p_{ij} initially has only M_{ij}^o tokens. This relation provides an easy way to compute the firing sequence. If the transition t_j has r input places, $p_{i_1}, p_{i_2}, \cdots, p_{i_r}$, then the relation can be represented as

$$S_j^{n+M_{i_kj}^o} \geq S_i^n + \widetilde{u_{i_kj}}, \text{ for } k = 1, \cdots, r \ . \tag{5.18}$$

Meanwhile, the transition cannot be fired if its output places are executing. Therefore, the only way to fire the transition is to wait until the processes of its output places are terminated and thus tokens in them are moved out. In this case, it can be treated as if the transition is self-looped by these places. The second relation represents this fact.

Here, if the number of repetition n is large enough an so $n > M^oij$, then equation (5.18) can be rewritten as

$$S_j^n > S_{i_k}^{n-M_{i_kj}^o} + \widetilde{u_{i_kj}}, \text{ for } k = 1, \cdots, r$$

and

$$S_j^n = max_{k=1,\cdots,r}\{S_{i_k}^{n-M_{i_kj}^o}, \ S_j^{n-1} + \widetilde{u_{j\cdot}}\} \ . \tag{5.19}$$

Now, if $n \leq M_{i_kj}^o$ for some k, $k \in \{1, 2, \cdots, r\}$, then the equation still holds except that the corresponding value of k is not considered, which means that the corresponding places p_{i_kj} have still strictly positive marking and thus, these places do not disable transition t_j.

Equation (5.19) turns out to be an implicit equation, which permits the computation of the firing sequence by iteration on the number of firing repetitions. If the initial markings of all input places are strictly positive, then S^n for all transitions in the system can be easily obtained. However, it is not the case of the system in this thesis. As previously seen in the corresponding STPPN C^2 model, since only the capacity places have a non-null initial marking, S_i^n should be determined in order to compute S_j^n for a pair of transitions, $(t_i,\ t_j)$. This implies that the computation of S_j^n for all the transitions in the model should be carried out according to the firing order. As stated in Section 4.4, this order is characterized by the slices of the model.

Consider any successive transitions t_i and t_j connected by p_{ij}. Then, it belongs to a line, since $S_i < S_j$ trivially and both transitions are elements of two different slices, say T_a and T_b, respectively. Since the slice reflects the partial order between the transitions, necessarily $a < b$. Thus, if the instants of n^{th} firings of all the transitions belonging to slices $T_1, T_2, \cdots, T_{k-1}$ have been determined, then the firing instants of transitions in slice T_k can be derived by using equation (5.19). Now, consider S_1^n corresponding to T_1. If the last transition denotes t_m, then by applying equation (5.19) the n^{th} firing instant of transition t_1 is

$$S_1^n = max\{S_m^{n-M_{m1}^o} + \widetilde{u_m},\ S_1^{n-1} + \widetilde{u_1}\} \ .$$

The computation order of S_j^n, for $j = 1, 2, \cdots, m$ is determined by the partial order obtained from the slice set. Note that for the transitions belonging to the same slice, the firing order has no importance because the transitions fire concurrently.

5.3.5.2 Derivation of average response time Denote

$$T_m^n = S_m^n + \widetilde{u_m} \ ,$$

where T_m^n is the instant of the n^{th} firing instant of transition t_m, which is the output transition of the model. Each time the t_m fires indicates the instant that the model has completed the processing of an input and has produced its response. As already assumed, since all inputs arrive simultaneously at time zero and wait their turns, T_m^k for $k = 1, 2, \cdots, n$, the instant of time at which the whole processing of the k^{th} input is terminated, represents the response time of the system. It also represents the dynamic behavior of the system. Therefore, the average response time can be defined as

$$\overline{RT} = \lim_{n \to \infty} \frac{T_m^n}{n} = \lim_{n \to \infty} \frac{S_m^n + \widetilde{u_m}}{n} \quad . \tag{5.20}$$

From Section 5.3, it will be realized that this limit is precisely equal to the maximum cycle processing time θ, which is also equal to $1/\Phi$.

5.4 Model Assumptions and Parameters

In this section, assumptions and parameters of the underlying STPPN models required to simulate are presented.

5.4.1 Model assumptions

In general, a scenario is needed in order to simulate a battle situation. The typical combat scenario used in this thesis is as follows. (1) The battle starts at time zero, i.e., detection of the enemy's threats is initially available. (2) The battle is terminated if the model state reaches one of the cases in Table 5.1. (3) In model C, if the battle unit is hit by attacks of the enemy, or if the combat unit malfunctions due to some internal reasons, then the unit has no more action and is ruled out from the system structure. Thus, the system structure is changed[4] (see Appendix B) and an extra burden is assigned to each unit according to the rules of the role

[4]Quantity of extra burden for the surviving units is driven in the next subsection.

Table 5.1: States conditions terminating system operations

Cases	Conditions
1	Commander and CIC is destroyed or malfunction.
2	Any combat unit, for instance, all BGs, except Commander and CIC is destroyed.
3	Over 50% of total combat units are destroyed and/or malfunction, i.e., the number of killed combat units > $(a+ab+c)/2$.

change as shown in Table 5.2 and Table 5.3. (4) The survivability of each unit is initially equal, i.e., survivability of each unit, P_S, is defined as

$$P_S = 1 - \text{Hit probability}, P_h \quad .$$

Therefore, the initial survivability is

$$P_S = \frac{1}{2+a+ab+c} \quad .$$

And the survivability of each unit decreases as the engagement continues, i.e., the survivability of each unit at time t can be deduced as

$$P_S(t) = \frac{1}{N-k}, \quad 1 \le k < \frac{N}{2}, \quad 0 \le t < \infty \quad .$$

where N is the total number of units in the system initially and k denotes the total number of destroyed units in time interval $(0, t)$, $0 < t < \infty$.

5.4.2 Model parameters

In order to simulate the underlying models, two parameters, the capacity and task processing times of each combat unit, should be defined. These parameters change their values according to the rules defined in Table 5.2. As an illustration, two examples are introduced. Suppose that Commander function is destroyed

Table 5.2: Role changes of each combat unit

Rules	Conditions	Remedies
1	Commander is destroyed.	CIC has an additional role of Commander. Therefore, task processing time of CIC increases and its Q should change.
2	CIC is destroyed.	1 BG has an additional role of CIC. Therefore, the task processing time of the BG increases, and its Q should changes. Processing times of the rest of the BGs increase, but the Qs do not change.
3	k BGs are destroyed.	Since the rest BGs have extra burdens, task processing times of (a-k) BGs increase, but the capacity of each unit does not change.
4	k SBGs are destroyed.	Since the rest of SBGs have extra burdens, task processing times of (ab-k) SBGs increase. but the capacity of each unit does not changed.
5	k IBGs are destroyed.	Since the rest of IBGs have extra burdens, task processing times of (c-k) IBGs increase, but the capacity of each unit does not change.

and/or malfunctions. Then, according to rule 1, CIC performs Commander's task. Therefore, the processing time of CIC, μ_{cic}, increases to μ'_{cic}, i.e.,

$$\mu'_{cic} = \alpha_1 * (\mu_{cic} + \mu_{cmd}), \quad 1 \leq \alpha_1 \quad ,$$

where α_1 denotes a weight factor and μ_i denotes the processing time of i^{th} combat unit.

Also, the capacity of CIC, Q_{cic} changes to Q'_{cic}, i.e.,

$$Q'_{cic} = min(Q_{cmd}, \ Q_{cic}) \quad .$$

Meanwhile, suppose that one BG is destroyed. Then, according to rule 3, the rest of the BGs have an extra burden. Therefore, their processing times change to

$$\mu'_{bg} = \alpha_4 * \mu_{bg}, \quad 1 \leq \alpha_4 \leq \frac{a^*}{a^* - 1} \quad ,$$

where α_4 denotes the weight factor and a^* is the total number of survived units in the previous state. However, the capacity of each BG does not change. By the same methods, processing times and Qs for all possible cases[5] can be obtained (see Table 5.3).

5.5 Model Inputs

This section deals with model inputs required to simulate the evolution of the underlying C^2 STPPN models.

[5]Total of 17 cases are considered.

5.5.1 Force powers

Since it is assumed that a surface-to-air battle is held at an air-base and the underlying system defends its base, the first thing to be considered is the size of power on both sides.

In this thesis, five sorties and the total of fifteen attacks are assigned for offensive power. As defensive power, one CMD, one CIC, four BGs, twelve SBGs, and two IBGs are prepared.

5.5.2 Processing time

Subprocesses in each C^2 process take time to perform their assigned tasks. As mentioned in Chapter 1, they are assumed to be random and distributed exponentially with mean μ, (Table 5.4). Note that as previously mentioned, processing times of dummy processes and places which indicate capacities are set to zero.

5.5.3 System and process capacities

Since models studied in this thesis are assumed to be safe, each place (subprocess and dummy process) can contain only one token at a time. Maximum capacity of each C^2 process can be easily obtained from the model structure. For example, since Commander unit in Model N has only 3 subprocesses, its capacity is maximum three. The capacities of the other C^2 process also can be obtained by this way. (See Table 5.5.) On the other hand, in the case where the capacity exceeds one, it destroys the *safeness* assumption. Therefore, in order to retain this assumption, the model developed in Chapter 2 should be reconstructed. It is very simple. Using the extension theory of petri net, the model can be constructed without destroying

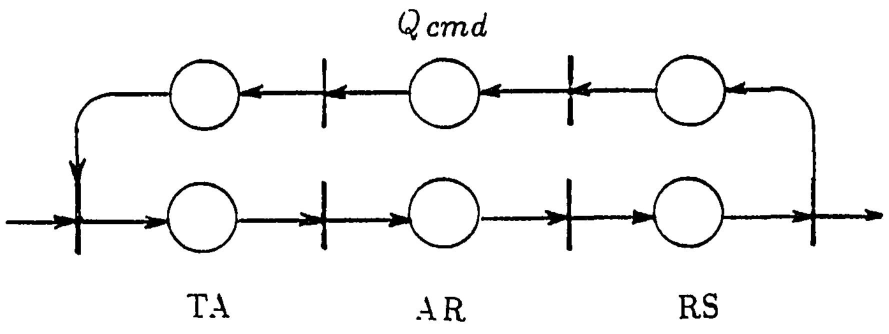


Figure 5.4: Modeling of place capacity

the original structure by connecting extra places into the corresponding capacity place, i.e., if the capacity of a process is three, then two extra places are connected to the corresponding places as shown in Figure 5.3. In this thesis, for simplicity, the system capacity is restricted to maximum three, i.e., the system can treat maximum three information simultaneously.

5.5.4 Weight factors

Weight factors used to adjust the processing times due to the destruction of a combat unit are assumed to be constant and their values are given in Table 5.6.

Table 5.3: Changes of parameter values by cases

Cases	Capacities	Processing time	Hit Pr.
1	$Q'_{cic} = min(Q_{cmd}, Q_{cic})$	$u'_{cic} = \alpha_1 * (u_{cmd} + u_{cic})$, $1 \leq \alpha_1$	$\frac{1}{(N-1)}$
2	$Q'_{cic} = min(Q_{cic}, Q_{bg})$ $Q'_{bg} = Q_{bg}$	$u'_{cic} = \alpha_2 * (u_{cic} + u_{bg})$, $1 \leq \alpha_2$ $u'_{bg} = \alpha_3 * u_{bg}$, $1 < \alpha_3 \leq \frac{a^*}{(a^*-1)}$, $a^* > 1$	$\frac{1}{(N-1)}$
3	no change	$u'_{bg} = \alpha_4 * u_{bg}$, $1 < \alpha_4 \leq \frac{a^*}{(a^*-1)}$, $a^* > 1$	$\frac{1}{(N-1)}$
4	no change	$u'_{sbg} = \alpha_5 * u_{sbg}$, $1 < \alpha_5 \leq \frac{ab^*}{(ab^*-1)}$, $ab^* > 1$	$\frac{1}{(N-k)}$
5	no change	$u'_{ibg} = \alpha_6 * u_{ibg}$ $1 < \alpha_6 \leq \frac{c^*}{(c^*-1)}$, $c^* > 1$	$\frac{1}{(N-k)}$

Table 5.3: (Continued)

Cases	Capacities & Processing time	Hit Pr.
6	Case 1∪Case 3	$1/\{N-(k+1)\}$
7	Case 1∪Case 4	$1/\{N-(k+1)\}$
8	Case 1∪Case 5	$1/\{N-(k+1)\}$
9	Case 2∪Case 3	$1/\{N-(k+1)\}$
10	Case 2∪Case 4	$1/\{N-(k+1)\}$
11	Case 2∪Case 5	$1/\{N-(k+1)\}$
12	Case 1∪Case 3∪Case 4	$1/\{N-(k_3+k_4+1)\}$
13	Case 1∪Case 3∪Case 5	$1/\{N-(k_3+k_5+1)\}$
14	Case 1∪Case 4∪CAse 5	$1/\{N-(k_4+k_5+1)\}$
15	Case 2∪Case 3∪Case 4	$1/\{N-(k_3+k_4+1)\}$
16	Case 2∪Case 3∪Case 5	$1/\{N-(k_3+k_5+1)\}$
17	Case 2∪Case 4∪Case 5	$1/\{N-(k_4+k_5+1)\}$

Table 5.4: Processing times of subprocesses

Process	1	2	3	4	5	6	7	8	9	10
Proc. time	1	1	1	1	1	5	0	8	5	0
Process	11	12	13	14	15	16	17	18	19	20
Proc. time	8	0	5	0	0	0	5	0	0	8
Process	21	22	23	24	25	26	27	28	29	30
Proc. time	5	10	0	0	2	10	0	0	0	2
Process	31	32	33	34	35	36	37	38	39	40
Proc. time	10	2	10	0	0	2	10	0	0	3
Process	41	42	43	44	45	46	47	48	49	50
Proc. time	3	3	0	0	0	0	0	0	0	0
Process	51	52	53	54	55	56	57	58	59	
Proc. time	0	0	0	0	0	0	0	0	0	

Table 5.5: Maximum system and process capacities

unit	system	CMD	BG	SBG	IBG
capacity	3	4	4	3	3

Table 5.6: Weight factors

α_1	α_2	α_3	α_4	α_5	α_6
1.0	1.0	$a^*/(a^* - 1)$	$a^*/(a^* - 1)$	$ab^*/(ab^* - 1)$	$c^*/(c^* - 1)$

6 NUMERICAL ANALYSIS

6.1 Introduction

In this chapter, simulation results from two C^2 models- model N and model C- are analyzed and compared. Two measures of effectiveness, the maximum throughput rate and the dynamic response time, which were derived in Chapter 5, and the maximum processing time discussed in Chapter 3, will be used to analyze the simulation results. In order to obtain the simulation results, a total of 30 runs for model N and a total of 40 runs for model C were carried out.

The results were analyzed according to the following steps. First, MOE and the processing rate for each C^2 process are calculated. Second, by analyzing them, if a problem exists, then, what kind of problem, and how it can be resolved are discussed. Third, after the remedy is taken, the resulting effectiveness is compared to the previous one in terms of MOE and the processing rates of C^2 processes. Finally, by repeating the previous procedures, the model with the best effectiveness is selected.

6.2 Normal Model, N

In this section, the normal model, N, is analyzed in terms of MOE with various inputs. Then, the model structure having the best effectiveness will be determined.

The model and its processing circuits are represented in Figure 6.1 and Table 6.1.

6.2.1 Case 1: Original input

In this subsection, a normal model with one capacity for each C^2 process is analyzed by using the processing times assigned in Subsection 5.5.2. From Table 6.4, the maximum average circuit processing time[1] can be directly obtained.

$$\begin{aligned} \alpha &= \max\{\alpha(\delta_1),\, \alpha(\delta_2), \cdots, \alpha(\delta_{76})\} \\ &= \max\{52.7,\, 64.2, \cdots, 23.5\} \\ &= 80.0 \ (= \alpha(\delta_{11})) \ . \end{aligned}$$

Therefore, maximum throughput rate, Φ, is

$$\Phi = \frac{1}{\alpha(\delta_{11})} = \frac{1}{80.0} = .0125 \ .$$

Also, the maximum processing rate of each C^2 process, ϕ, is

a. CMD

$$\begin{aligned} \phi &= \min\left\{f_{76}, \frac{1}{\overline{u}_{19}}, \frac{1}{\overline{u}_{21}}, \frac{1}{\overline{u}_{22}}\right\} \\ &= \min\left\{\frac{1}{\alpha(\delta_{76})}, \frac{1}{\overline{u}_{19}}, \frac{1}{\overline{u}_{21}}, \frac{1}{\overline{u}_{22}}\right\} \\ &= min\left\{\frac{1}{23.5}, \frac{1}{8.06}, \frac{1}{5.32}, \frac{1}{10.13}\right\} \\ &= .0426 \ (= f_{76}) \ . \end{aligned}$$

b. CIC

$$\phi = \min\left\{f_{68}, f_{72}, \frac{1}{\overline{u}_{13}}, \frac{1}{\overline{u}_{18}}, \frac{1}{\overline{u}_{25}}, \frac{1}{\overline{u}_{26}}\right\}$$

[1] Average circuit processing times are calculated from the program by using the statistics in Table 6.3.

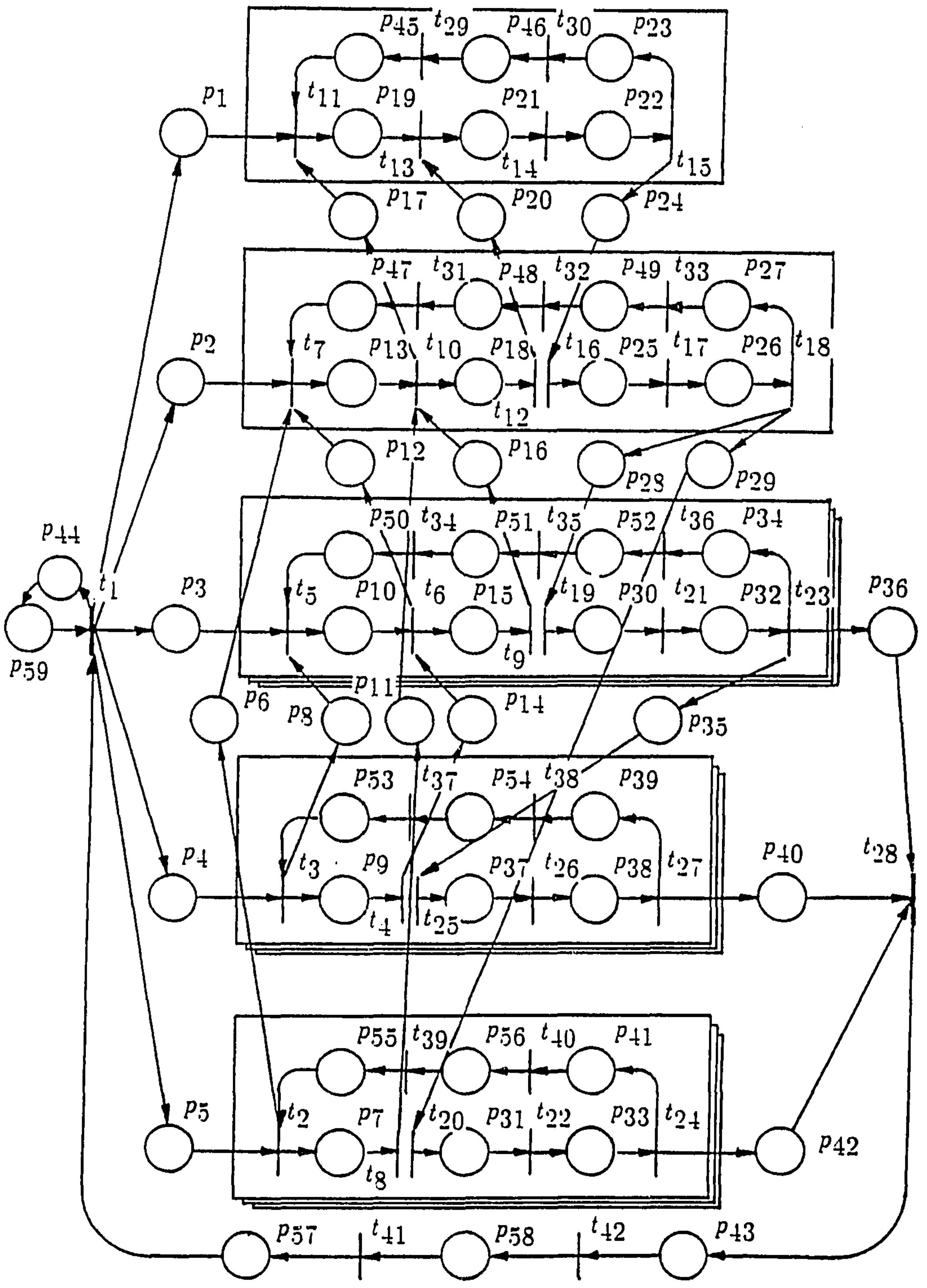

p_{45} t_{29} p_{46} t_{30} p_{23}
p_1 t_{11} p_{19} p_{21} p_{22}
t_{13} t_{14} t_{15}
p_{17} p_{20} p_{24}
p_{47} t_{31} p_{48} t_{32} p_{49} t_{33} p_{27}
p_2 t_7 p_{13} t_{10} p_{18} t_{16} p_{25} t_{17} p_{26} t_{18}
t_{12}
p_{12} p_{16} p_{28} p_{29}
p_{44} p_{50} t_{34} p_{51} t_{35} p_{52} t_{36} p_{34}
t_1 p_3 t_5 p_{10} t_6 p_{15} t_{19} p_{30} t_{21} p_{32} t_{23} p_{36}
p_{59} t_9
p_6 p_8 p_{11} p_{14} p_{35}
p_{53} t_{37} p_{54} t_{38} p_{39}
p_4 t_3 p_9 p_{37} t_{26} p_{38} t_{27} p_{40} t_{28}
t_4 t_{25}
p_{55} t_{39} p_{56} t_{40} p_{41}
p_5 t_2 p_7 t_{20} p_{31} t_{22} p_{33} t_{24} p_{42}
t_8
p_{57} t_{41} p_{58} t_{42} p_{43}

Figure 6.1: Normal model of the C^2system

Table 6.1: Simple directed circuits in Figure 6.1

· Circuits for place 1.

δ_1: $p_1 \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.
δ_2: $p_1 \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.
δ_3: $p_1 \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

· Circuits for place 2.

δ_4: $p_2 \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.
δ_5: $p_2 \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.
δ_6: $p_2 \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.
δ_7: $p_2 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.
δ_8: $p_2 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.
δ_9: $p_2 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

Table 6.1: (Continued)

· Circuits for place 3.

δ_{10}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow$
$p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$ ·

δ_{11}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow$
$p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$ ·

δ_{12}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow$
$p_{42} \rightarrow p_{43}$ ·

δ_{13}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$ ·

δ_{14}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow$
$p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$ ·

δ_{15}: $p_3 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow$
$p_{42} \rightarrow p_{43}$ ·

δ_{16}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow$
$p_{36} \rightarrow p_{43}$ ·

δ_{17}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow$
$p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$ ·

δ_{18}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow$
$p_{42} \rightarrow p_{43}$ ·

δ_{19}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow$
$p_{36} \rightarrow p_{43}$ ·

Table 6.1: (Continued)

δ_{20}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{21}: $p_3 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

· Circuits for place 4.

δ_{22}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{23}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{24}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{25}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{26}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{27}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{28}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

Table 6.1: (Continued)

δ_{29}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{12} \rightarrow p_{16} \rightarrow p_{17} \rightarrow p_{19} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{30}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow$
$p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow$
$p_{42} \rightarrow p_{43}$.

δ_{31}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{32}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{33}: $p_4 \rightarrow p_8 \rightarrow p_{10} \rightarrow p_{15} \rightarrow p_{16} \rightarrow p_{18} \rightarrow p_{20} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow$
$p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{34}: $p_4 \rightarrow p_9 \rightarrow p_{11} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{35}: $p_4 \rightarrow p_9 \rightarrow p_{11} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{36}: $p_4 \rightarrow p_9 \rightarrow p_{11} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow$
$p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{37}: $p_4 \rightarrow p_9 \rightarrow p_{11} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{38}: $p_4 \rightarrow p_9 \rightarrow p_{11} \rightarrow p_{12} \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow$
$p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow$
$p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

Table 6.1: (Continued)

δ_{39}: $p_4 \to p_9 \to p_{11} \to p_{12} \to p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{42} \to p_{43}$.

δ_{40}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{36} \to p_{43}$.

δ_{41}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{40} \to p_{43}$.

δ_{42}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{42} \to p_{43}$.

δ_{43}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{36} \to p_{43}$.

δ_{44}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{40} \to p_{43}$.

δ_{45}: $p_4 \to p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{42} \to p_{43}$.

· Circuits for place 5.

δ_{46}: $p_5 \to p_6 \to p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{36} \to p_{43}$.

Table 6.1: (Continued)

δ_{47}: $p_5 \rightarrow p_6 \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{48}: $p_5 \rightarrow p_6 \rightarrow p_{13} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{49}: $p_5 \rightarrow p_6 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{50}: $p_5 \rightarrow p_6 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{51}: $p_5 \rightarrow p_6 \rightarrow p_{13} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{52}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{53}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{54}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{17} \rightarrow p_{19} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

δ_{55}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{36} \rightarrow p_{43}$.

δ_{56}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{28} \rightarrow p_{30} \rightarrow p_{32} \rightarrow p_{35} \rightarrow p_{37} \rightarrow p_{38} \rightarrow p_{40} \rightarrow p_{43}$.

δ_{57}: $p_5 \rightarrow p_7 \rightarrow p_{14} \rightarrow p_{18} \rightarrow p_{20} \rightarrow p_{21} \rightarrow p_{22} \rightarrow p_{24} \rightarrow p_{25} \rightarrow p_{26} \rightarrow p_{29} \rightarrow p_{31} \rightarrow p_{33} \rightarrow p_{42} \rightarrow p_{43}$.

Table 6.1: (Continued)

· Circuits for place 6.

δ_{58}: $p_6 \to p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{41} \to p_{56} \to p_{55}$.

δ_{59}: $p_6 \to p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{41} \to p_{56} \to p_{55}$.

δ_{60}: $p_7 \to p_{14} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{41} \to p_{56} \to p_{55}$.

δ_{61}: $p_7 \to p_{14} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{29} \to p_{31} \to p_{33} \to p_{41} \to p_{56} \to p_{55}$.

· Circuits for place 8.

δ_{62}: $p_8 \to p_{10} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53}$.

δ_{63}: $p_8 \to p_{10} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53}$.

· Circuits for place 9.

δ_{64}: $p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53}$.

δ_{65}: $p_9 \to p_{11} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53}$.

Table 6.1: (Continued)

· Circuits for place 10.

δ_{66}: $p_{10} \to p_{15} \to p_{16} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{34} \to p_{52} \to p_{51} \to p_{50}$ ·

δ_{67}: $p_{10} \to p_{15} \to p_{16} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{34} \to p_{52} \to p_{51} \to p_{50}$ ·

· Circuits for place 13.

δ_{68}: $p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{27} \to p_{49} \to p_{48} \to p_{47}$ ·

δ_{69}: $p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{34} \to p_{52} \to p_{51} \to p_{50} \to p_{10} \to p_{12}$ ·

δ_{70}: $p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53} \to p_{8} \to p_{10} \to p_{12}$ ·

δ_{71}: $p_{13} \to p_{17} \to p_{19} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53} \to p_{9} \to p_{11} \to p_{12}$ ·

δ_{72}: $p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{27} \to p_{49} \to p_{48} \to p_{47}$ ·

δ_{73}: $p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{34} \to p_{52} \to p_{51} \to p_{50} \to p_{10} \to p_{12}$ ·

δ_{74}: $p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53} \to p_{8} \to p_{10} \to p_{12}$ ·

δ_{75}: $p_{13} \to p_{18} \to p_{20} \to p_{21} \to p_{22} \to p_{24} \to p_{25} \to p_{26} \to p_{28} \to p_{30} \to p_{32} \to p_{35} \to p_{37} \to p_{38} \to p_{39} \to p_{54} \to p_{53} \to p_{9} \to p_{11} \to p_{12}$ ·

· Circuits for place 19.

δ_{76}: $p_{19} \to p_{21} \to p_{22} \to p_{23}$ ·

$$= \min\left\{\frac{1}{\alpha(\delta_{68})}, \frac{1}{\alpha(\delta_{72})}, \frac{1}{\overline{u}_{13}}, \frac{1}{\overline{u}_{18}}, \frac{1}{\overline{u}_{25}}, \frac{1}{\overline{u}_{26}}\right\}$$

$$= \min\left\{\frac{1}{44.9}, \frac{1}{41.9}, \frac{1}{8.64}, \frac{1}{5.12}, \frac{1}{2.02}, \frac{1}{10.71}\right\}$$

$$= .0223 \; (= f_{68}) \; .$$

c. BG

$$\phi = \min\left\{f_{66}, f_{67}, f_{69}, f_{73}, \frac{1}{\overline{u}_{10}}, \frac{1}{\overline{u}_{15}}, \frac{1}{\overline{u}_{30}}, \frac{1}{\overline{u}_{32}}\right\}$$

$$= \min\left\{\frac{1}{\alpha(\delta_{66})}, \frac{1}{\alpha(\delta_{67})}, \frac{1}{\alpha(\delta_{69})}, \frac{1}{\alpha(\delta_{73})}, \frac{1}{\overline{u}_{10}}, \frac{1}{\overline{u}_{15}}, \frac{1}{\overline{u}_{30}}, \frac{1}{\overline{u}_{32}}\right\}$$

$$= \min\left\{\frac{1}{60.6}, \frac{1}{57.6}, \frac{1}{64.5}, \frac{1}{61.6}, \frac{1}{7.17}, \frac{1}{4.68}, \frac{1}{2.11}, \frac{1}{10.37}\right\}$$

$$= .0155 \; (= f_{69}) \; .$$

d. SBG

$$\phi = \min\left\{f_{62}, f_{63}, f_{64}, f_{65}, f_{70}, f_{71}, f_{74}, f_{75}, \frac{1}{\overline{u}_{9}}, \frac{1}{\overline{u}_{37}}, \frac{1}{\overline{u}_{38}}\right\}$$

$$= \min\left\{\frac{1}{\alpha(\delta_{62})}, \frac{1}{\alpha(\delta_{63})}, \frac{1}{\alpha(\delta_{64})}, \frac{1}{\alpha(\delta_{65})}, \frac{1}{\alpha(\delta_{70})}, \frac{1}{\alpha(\delta_{71})}, \frac{1}{\alpha(\delta_{74})}, \frac{1}{\alpha(\delta_{75})}, \frac{1}{\overline{u}_{9}}, \frac{1}{\overline{u}_{37}}, \frac{1}{\overline{u}_{38}}\right\}$$

$$= \min\left\{\frac{1}{72.3}, \frac{1}{69.3}, \frac{1}{70.1}, \frac{1}{67.1}, \frac{1}{76.2}, \frac{1}{74.0}, \frac{1}{73.3}, \frac{1}{71.1}, \frac{1}{4.97}, \frac{1}{1.98}, \frac{1}{9.72}\right\}$$

$$= .0131 \; (= f_{70}) \; .$$

e. IBG

$$\phi = \min\left\{f_{58}, f_{59}, f_{60}, f_{61}, \frac{1}{\overline{u}_{7}}, \frac{1}{\overline{u}_{31}}, \frac{1}{\overline{u}_{33}}\right\}$$

$$= \min\left\{\frac{1}{\alpha(\delta_{58})}, \frac{1}{\alpha(\delta_{59})}, \frac{1}{\alpha(\delta_{60})}, \frac{1}{\alpha(\delta_{61})}, \frac{1}{\overline{u}_7}, \frac{1}{\overline{u}_{31}}, \frac{1}{\overline{u}_{33}}\right\}$$
$$= \min\left\{\frac{1}{57.4}, \frac{1}{54.4}, \frac{1}{53.9}, \frac{1}{51.0}, \frac{1}{5.18}, \frac{1}{2.01}, \frac{1}{10.51}\right\}$$
$$= .0174\ (= f_{58}) \ .$$

The above results indicate that the maximum throughput rate of the system is determined only by the structural constraint since δ_{11} is the unique critical circuit of the net. Therefore, in this case, two possible remedies to improve the performance of the overall net can be considered: either increasing the capacities available to the system or reducing the task processing times of its subprocesses or both.

On one hand, the system with one capacity can can handle only one attack at a time no matter how many capacities the other C^2 processes have. In fact, since the number of the system capacity is determined by the model structure, i.e., it can be obtained from the number of slices discussed in Subsection 5.2.2. Also, since the marking process corresponds to the processing sequence of slices in the model, the maximum system capacity equals the number of slices of the net, i.e.,

$$\max\{Q_{sys}\} = \text{the number of slices} = 19 \ .$$

Since increasing the capacity of the system allows the system to treat more attacks at a time, it yields generally a better model effectiveness. Also, as stated in Subsection 5.2.2, since the capacity of a process is determined by the structural constraint, each C^2 process has its own capacity limit. Consequently, assuming that the capacity should be increased in order to improve the model effectiveness, it can be increased only up to its maximum limit as stated previously.

On the other hand, the processing time indicates the time required for a person or a group assigned previously to perform its task. Thus, the only way to improve

Table 6.2: Case 1: Task processing history

	1	2	3	4	5	6	7	8
1	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
2	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
3	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
4	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
5	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
6	1.0	93.8	182.9	274.4	372.2	454.4	543.3	634.4
7	1.0	93.8	182.9	274.4	372.2	454.4	543.3	634.4
8	1.2	93.8	182.9	274.2	372.3	454.8	543.0	634.0
9	1.2	93.8	182.9	274.2	372.3	454.8	543.0	634.0
10	1.7	94.3	183.3	274.6	372.6	455.3	543.6	634.5
11	5.6	99.9	188.4	279.7	377.1	460.0	550.3	638.4
12	9.8	104.0	191.2	283.9	380.3	464.8	554.0	646.3
13	9.8	104.0	191.2	283.9	380.3	464.8	554.0	646.5
14	6.8	98.9	189.9	279.0	375.8	460.1	549.9	639.3
15	9.8	104.0	191.2	283.9	380.3	464.8	554.0	646.3
16	13.9	108.7	194.7	290.5	384.0	470.0	558.6	651.5
17	18.5	112.8	204.3	295.4	390.2	476.8	565.7	657.0
18	18.5	112.8	204.3	295.4	390.2	476.8	565.7	657.0
19	18.5	112.8	204.3	295.4	390.2	476.8	565.7	657.0
20	25.5	116.8	211.6	301.2	395.0	481.2	570.7	661.8
21	29.6	121.7	215.0	306.2	398.4	486.4	575.1	668.3
22	36.7	126.6	220.4	313.5	402.7	493.0	579.4	672.5
23	47.8	137.7	230.8	324.6	409.7	503.0	591.2	681.6
24	47.8	137.7	230.8	324.6	409.7	503.0	591.2	681.6
25	47.8	137.7	230.8	324.6	409.7	503.0	591.2	681.6
26	50.3	139.5	232.0	327.2	412.1	505.5	593.1	683.3
27	61.9	149.4	242.0	339.3	422.3	515.4	605.8	695.4
28	61.9	149.4	242.0	339.3	422.3	515.4	605.8	695.4
29	61.9	149.4	242.0	339.3	422.3	515.4	605.8	695.4
30	61.9	149.4	242.0	339.3	422.3	515.4	605.8	695.4
31	61.9	149.4	242.0	339.3	422.3	515.4	605.8	695.4
32	63.7	151.3	243.3	341.1	424.4	517.5	608.0	697.2
33	63.9	151.2	243.5	341.0	424.2	517.5	608.3	697.4
34	75.0	161.4	253.2	356.9	434.9	526.0	615.5	707.3
35	75.0	161.4	253.2	356.9	434.9	526.0	615.5	707.3
36	75.0	161.4	253.2	356.9	434.9	526.0	615.5	707.3
37	75.0	161.4	253.2	356.9	434.9	526.0	615.5	707.3
38	76.6	163.7	255.3	358.6	436.8	528.1	617.1	709.9
39	88.3	174.5	265.0	366.7	447.3	537.5	624.9	717.6
40	88.3	174.5	265.0	366.7	447.3	537.5	624.9	717.6

Table 6.2: (Continued)

	1	2	3	4	5	6	7	8
41	73.1	163.5	258.2	345.9	435.3	526.9	621.4	706.6
42	73.1	163.5	258.2	345.9	435.3	526.9	621.4	706.6
43	92.9	182.1	273.4	371.1	453.6	542.2	633.0	721.8
44	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0
59	0.0	92.9	182.1	273.4	371.1	453.6	542.2	633.0

	9	10	11	12	13	14	15
1	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
2	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
3	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
4	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
5	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
6	722.7	809.1	900.4	988.5	1076.4	1167.0	1247.0
7	722.7	809.1	900.4	988.5	1076.4	1167.0	1247.0
8	722.7	809.1	900.5	987.9	1076.1	1167.0	1246.7
10	723.1	809.5	900.9	988.3	1076.7	1167.4	1247.1
11	727.6	813.4	905.8	992.2	1080.1	1172.1	1249.9
12	731.8	817.3	910.2	995.4	1085.9	1178.0	1255.8
13	731.8	817.3	910.2	995.4	1085.9	1178.0	1255.9
14	726.3	811.6	906.5	992.8	1083.4	1171.8	1252.8
15	731.8	817.3	910.2	995.4	1085.9	1178.0	1255.8
16	736.7	821.3	915.1	1000.4	1091.5	1180.9	1260.9
17	743.8	829.3	921.9	1004.1	1097.8	1187.1	1266.8
18	743.8	829.3	921.9	1004.1	1097.8	1187.1	1266.8
19	743.8	829.3	921.9	1004.1	1097.8	1187.1	1266.8
20	748.9	833.5	926.3	1009.1	1103.3	1191.8	1271.6
21	755.0	839.1	932.6	1015.6	1106.5	1195.8	1278.6
22	761.2	843.1	936.3	1021.0	1111.5	1201.3	1284.7
23	767.9	855.1	945.8	1031.8	1121.3	1209.9	1297.5
24	767.9	855.1	945.8	1031.8	1121.3	1209.9	1297.5
25	767.9	855.1	945.8	1031.8	1121.3	1209.9	1297.5
26	770.1	857.3	947.0	1033.9	1123.3	1211.6	1299.6
27	779.7	867.5	955.3	1046.6	1131.5	1219.0	1315.4
28	779.7	867.5	955.3	1046.6	1131.5	1219.0	1315.4
29	779.7	867.5	955.3	1046.6	1131.5	1219.0	1315.4
30	779.7	867.5	955.3	1046.6	1131.5	1219.0	1315.4
31	779.7	867.5	955.3	1046.6	1131.5	1219.0	1315.4
32	782.3	869.9	957.8	1049.3	1133.6	1221.0	1317.7
33	781.7	869.8	957.1	1048.6	1133.7	1220.9	1317.7
34	791.4	882.8	968.6	1059.7	1145.0	1229.1	1326.8
35	791.4	882.8	968.6	1059.7	1145.0	1229.1	1326.8

Table 6.2: (Continued)

	9	10	11	12	13	14	15
36	791.4	882.8	968.6	1059.7	1145.0	1229.1	1326.8
37	791.4	882.8	968.6	1059.7	1145.0	1229.1	1326.8
38	793.6	884.6	970.9	1061.4	1147.3	1230.7	1328.8
39	802.0	893.8	982.2	1071.3	1161.0	1240.2	1337.1
40	802.0	893.8	982.2	1071.3	1161.0	1240.2	1337.1
41	792.2	878.6	971.1	1058.9	1143.2	1230.3	1329.2
42	792.2	878.6	971.1	1058.9	1143.2	1230.3	1329.2
43	808.1	899.4	987.0	1075.3	1166.0	1245.9	1342.7
44	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9
59	721.8	808.1	899.4	987.0	1075.3	1166.0	1245.9

Table 6.3: Case 1: Average processing times

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
1	450	454.34	1.01	9138.46	9592.79	21.32
2	450	432.69	0.96	4277.34	4710.03	10.47
3	450	433.72	0.96	205.73	639.45	1.42
4	450	430.22	0.96	0.00	430.22	0.96
5	450	471.96	1.05	0.00	471.96	1.05
6	450	0.00	0.00	4238.07	4238.07	9.42
7	450	2329.49	5.18	0.00	2329.49	5.18
8	450	0.00	0.00	209.23	209.23	0.46
9	450	2236.86	4.97	0.00	2236.86	4.97
10	450	3226.67	7.17	836.86	4063.53	9.03
11	450	0.00	0.00	2035.89	2035.89	4.52
12	450	0.00	0.00	7.07	7.07	0.02
13	450	3885.83	8.64	997.27	4883.10	10.85
14	450	0.00	0.00	6791.43	6791.43	15.09
15	450	2105.68	4.68	0.00	2105.68	4.68
16	450	0.00	0.00	2784.48	2784.48	6.19
17	450	0.00	0.00	0.00	0.00	0.00
18	450	2303.18	5.12	0.00	2303.18	5.12
19	450	3629.12	8.06	946.96	4576.08	10.17
20	450	0.00	0.00	2272.88	2272.88	5.05
21	450	2393.75	5.32	0.00	2393.75	5.32
22	450	4557.10	10.13	0.00	4557.10	10.13
23	450	0.00	0.00	28197.91	28197.91	62.66
24	450	0.00	0.00	0.00	0.00	0.00
25	450	906.83	2.02	0.00	906.83	2.02

Table 6.3: (Continued)

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
26	450	4819.10	10.71	0.00	4819.10	10.71
27	450	0.00	0.00	17852.20	17852.20	39.67
28	450	0.00	0.00	0.00	0.00	0.00
29	450	0.00	0.00	0.00	0.00	0.00
30	450	948.76	2.11	0.00	948.76	2.11
31	450	902.33	2.01	0.00	902.33	2.01
32	450	4666.41	10.37	0.00	4666.41	10.37
33	450	4731.36	10.51	0.00	4731.36	10.51
34	450	0.00	0.00	8409.18	8409.18	18.69
35	450	0.00	0.00	0.00	0.00	0.00
36	450	1341.08	2.98	6478.49	7819.57	17.38
37	450	892.30	1.98	0.00	892.30	1.98
38	450	4376.09	9.72	0.00	4376.09	9.72
39	450	0.00	0.00	2947.11	2947.11	6.55
40	450	1260.60	2.80	1290.80	2551.40	5.67
41	450	0.00	0.00	8242.54	8242.54	18.32
42	450	1377.94	3.06	6423.21	7801.14	17.34
43	450	0.00	0.00	0.00	0.00	0.00
44	450	0.00	0.00	0.00	0.00	0.00
59	450	0.00	0.00	37376.52	37376.52	83.06

Table 6.4: Case 1: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	52.7	64.2	52.8	61.3	72.8	61.4	58.3	69.9	58.5
ACPT	:	52.7	64.2	52.8	61.3	72.8	61.4	58.3	69.9	58.5
NO.		10	11	12	13	14	15	16	17	18
TIME	:	68.5	80.0	68.6	65.5	77.0	65.6	64.5	76.0	64.6
ACPT	:	68.5	80.0	68.6	65.5	77.0	65.6	64.5	76.0	64.6
NO.		19	20	21	22	23	24	25	26	27
TIME	:	61.6	73.1	61.7	68.5	80.0	68.6	65.5	77.0	61.7
ACPT	:	61.6	73.1	61.7	68.5	80.0	68.6	65.5	77.0	61.7
NO.		28	29	30	31	32	33	34	35	36
TIME	:	68.5	71.3	64.6	61.6	73.1	61.7	66.3	77.8	66.4
ACPT	:	68.5	71.3	64.6	61.6	73.1	61.7	66.3	77.8	66.4
NO.		37	38	39	40	41	42	43	44	45
TIME	:	63.3	74.8	63.4	62.3	73.8	62.4	59.4	70.9	59.5
ACPT	:	63.3	74.8	63.4	62.3	73.8	62.4	59.4	70.9	59.5
NO.		46	47	48	49	50	51	52	53	54
TIME	:	61.4	72.9	61.5	58.4	70.0	58.6	57.9	69.4	58.0
ACPT	:	61.4	72.9	61.5	58.4	70.0	58.6	57.9	69.4	58.0
NO.		55	56	57	58	59	60	61	62	63
TIME	:	55.0	66.5	55.1	57.4	54.4	53.9	51.0	72.3	69.3
ACPT	:	55.0	66.5	55.1	57.4	54.4	53.9	51.0	72.3	69.3
NO.		64	65	66	67	68	69	70	71	72
TIME	:	70.1	67.1	60.6	57.6	44.9	64.5	76.2	74.0	41.9
ACPT	:	70.1	67.1	60.6	57.6	44.9	64.5	76.2	74.0	41.9
NO.		73	74	75	76					
TIME	:	61.6	73.3	71.1	23.5					
ACPT	:	61.6	73.3	71.1	23.5					

the model effectiveness in this case is to reduce the processing time by training the member involved in the process. Therefore, as the treatment to obtain a better effectiveness of Case 1, which has one capacity for each C^2 process and the overall system, the processing times on the critical circuit are reduced. After comparing this result to the original one, the effect by increasing capacity will be investigated.

6.2.2 Case 2: Reduced processing times on δ_{11}

The selection of the process to be reduced depends on the length of the token holding time of the process as stated in Subsection 5.2.2. For example, comparing the holding times of processes (or places), p_{10} and p_{11}, which are in the same slice, the p_{10} holding time is less than that of p_{11}. This implies that the process, p_{10}, is more critical for firing of t_6 since the firing time of t_6 only depends on the task termination time of the process which finishes its task the latest among the input places of t_6. Therefore, p_{10} should be reduced. Meanwhile, as shown in Table 6.3, the processes in the circuit, δ_{11} are all minimal in their slices. The reason is very simple. Since the circuit, δ_{11}, is the longest circuit in the system, all places in that circuit are critical.

Processing times of the critical circuit, δ_{11}, are reduced as shown in Table 6.5. In this case, the reduced amount of the processing time is assumed to be the maximum available to each process. From Table 6.8, the maximum average circuit processing time, α, is

$$\begin{aligned} \alpha &= \max\Big\{\alpha(\delta_1), \cdots, \alpha(\delta_{76})\Big\} \\ &= \max\{33.1, 39.8, \cdots, 14.1\} \\ &= 50.2\ (= \alpha(\delta_{38}))\ . \end{aligned}$$

Table 6.5: Reduced mean processing times of circuit. δ_{11}

place	10	13	19	21	22	25	26	30	32	37	38
old $\bar{\mu}$	8	8	8	5	10	2	10	2	10	2	10
new $\bar{\mu}$	4.8	4.8	4.8	3	6	1.2	6	1.2	6	1.2	6

Therefore, the maximum throughput rate is $\Phi = 1/\alpha = 1/50.2 = .0199$. Comparing this result with the previous one (=.0125), this model, i.e., the model with reduced processing times, is improved as expected. Now, the maximum processing rates of this case are

$$\begin{aligned}
\phi_{cmd} &= \min\left\{f_{76}, \frac{1}{\bar{u}_{19}}, \frac{1}{\bar{u}_{21}}, \frac{1}{\bar{u}_{22}}\right\} \\
&= \min\left\{\frac{1}{14.3}, \frac{1}{4.84}, \frac{1}{3.19}, \frac{1}{6.08}\right\} \\
&= .0709 \ (= f_{76}) \\
\phi_{cic} &= \min\left\{f_{68}, f_{72}, \frac{1}{\bar{u}_{13}}, \frac{1}{\bar{u}_{18}}, \frac{1}{\bar{u}_{25}}, \frac{1}{\bar{u}_{26}}\right\} \\
&= \min\left\{\frac{1}{26.9}, \frac{1}{27.2}, \frac{1}{5.18}, \frac{1}{5.12}, \frac{1}{1.21}, \frac{1}{6.43}\right\} \\
&= .0368 \ (= f_{72}) \\
\phi_{bg} &= \min\left\{f_{66}, f_{67}, f_{69}, f_{73}, \frac{1}{\bar{u}_{10}}, \frac{1}{\bar{u}_{15}}, \frac{1}{\bar{u}_{30}}, \frac{1}{\bar{u}_{32}}\right\} \\
&= \min\left\{\frac{1}{38.0}, \frac{1}{38.3}, \frac{1}{38.5}, \frac{1}{38.7}, \frac{1}{4.30}, \frac{1}{4.70}, \frac{1}{1.26}, \frac{1}{5.96}\right\} \\
&= .0258 \ (= f_{73}) \\
\phi_{sbg} &= \min\left\{f_{62}, f_{63}, f_{64}, f_{65}, f_{70}, f_{71}, f_{74}, f_{75}, \frac{1}{\bar{u}_{9}}, \frac{1}{\bar{u}_{37}}, \right. \\
&\qquad \left. \frac{1}{\bar{u}_{38}}\right\} \\
&= \min\left\{\frac{1}{45.0}, \frac{1}{45.3}, \frac{1}{45.7}, \frac{1}{46.0}, \frac{1}{45.5}, \frac{1}{46.2}, \frac{1}{45.8}, \frac{1}{46.5}, \right. \\
&\qquad \left. \frac{1}{4.97}, \frac{1}{1.18}, \frac{1}{5.87}\right\}
\end{aligned}$$

Table 6.6: Case 2: Task processing history

	1	2	3	4	5	6	7	8
1	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
2	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
3	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
4	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
5	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
6	1.0	66.8	130.6	197.1	263.7	322.5	384.8	450.6
7	1.0	66.8	130.6	197.1	263.7	322.5	384.8	450.6
8	1.2	66.8	130.7	196.9	263.8	322.8	384.5	450.1
9	1.2	66.8	130.7	196.9	263.8	322.8	384.5	450.1
10	1.7	67.2	131.1	197.3	264.1	323.3	385.2	450.6
11	5.6	72.9	136.2	202.3	268.6	328.0	391.9	454.6
12	7.5	74.9	137.5	204.3	270.1	330.6	393.7	458.7
13	7.5	74.9	137.5	204.3	270.1	330.6	393.7	458.9
14	6.7	71.9	137.7	201.7	267.3	328.1	391.5	455.1
15	7.5	74.9	137.5	204.3	270.1	330.6	393.7	458.7
16	11.7	79.6	141.0	211.0	273.7	335.8	398.3	464.2
17	14.9	81.9	146.9	213.5	277.2	339.5	402.7	466.9
18	14.9	81.9	146.9	213.5	277.2	339.5	402.7	466.9
19	14.9	81.9	146.9	213.5	277.2	339.5	402.7	466.9
20	21.8	85.9	154.2	219.4	282.0	343.9	407.8	471.7
21	23.7	88.5	155.7	221.6	283.5	346.6	409.6	475.1
22	28.0	91.3	158.9	226.0	286.1	350.6	412.2	477.6
23	34.7	98.0	165.2	232.6	290.3	356.5	419.3	483.1
24	34.7	98.0	165.2	232.6	290.3	356.5	419.3	483.1
25	34.7	98.0	165.2	232.6	290.3	356.5	419.3	483.1
26	36.2	99.1	165.9	234.2	291.7	358.0	420.4	484.2
27	43.1	105.1	171.9	241.4	297.8	363.9	428.1	491.4
28	43.1	105.1	171.9	241.4	297.8	363.9	428.1	491.4
29	43.1	105.1	171.9	241.4	297.8	363.9	428.1	491.4
30	43.1	105.1	171.9	241.4	297.8	363.9	428.1	491.4
31	43.1	105.1	171.9	241.4	297.8	363.9	428.1	491.4
32	44.2	106.2	172.7	242.5	299.1	365.2	429.4	492.5
33	45.2	106.9	173.4	243.2	299.8	366.0	430.5	493.4
34	49.5	111.7	179.7	251.0	306.0	370.3	433.9	498.2
35	49.5	111.7	179.7	251.0	306.0	370.3	433.9	498.2
36	49.5	111.7	179.7	251.0	306.0	370.3	433.9	498.2
37	49.5	111.7	179.7	251.0	306.0	370.3	433.9	498.2
38	50.4	113.0	181.0	252.2	307.0	371.7	434.9	499.8
39	56.1	119.0	187.6	257.5	314.9	377.2	439.7	504.0
40	56.1	119.0	187.6	257.5	314.9	377.2	439.7	504.0

Table 6.6: (Continued)

	1	2	3	4	5	6	7	8
41	57.6	120.6	185.8	248.8	310.0	375.7	442.6	502.2
42	57.6	120.6	185.8	248.8	310.0	375.7	442.6	502.2
43	65.9	129.8	196.0	262.6	321.6	383.8	449.1	510.3
44	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1
59	0.0	65.9	129.8	196.0	262.6	321.6	383.8	449.1

	9	10	11	12	13	14	15
1	510.3	572.5	632.8	694.9	758.1	821.7	877.1
2	510.3	572.5	632.8	694.9	758.1	821.7	877.1
3	510.3	572.5	632.8	694.9	758.1	821.7	877.1
4	510.3	572.5	632.8	694.9	758.1	821.7	877.1
5	510.3	572.5	632.8	694.9	758.1	821.7	877.1
6	511.2	573.5	633.8	696.3	759.2	822.6	878.2
7	511.2	573.5	633.8	696.3	759.2	822.6	878.2
8	511.2	573.5	633.8	695.7	759.0	822.6	878.0
9	511.2	573.5	633.8	695.7	759.0	822.6	878.0
10	511.6	573.9	634.3	696.1	759.5	823.1	878.4
11	516.1	577.8	639.2	700.0	763.0	827.8	881.1
12	518.3	579.6	641.4	701.7	765.8	830.3	884.2
13	518.3	579.6	641.4	701.7	765.8	830.4	884.3
14	514.8	576.0	639.9	700.7	766.3	827.4	884.2
15	518.3	579.6	641.4	701.7	765.8	830.3	884.2
16	523.2	583.7	646.3	706.7	771.3	833.3	889.2
17	527.0	587.8	650.0	708.7	775.8	836.9	892.4
18	527.0	587.8	650.0	708.7	775.8	836.9	892.4
19	527.0	587.8	650.0	708.7	775.8	836.9	892.4
20	532.0	592.0	654.4	713.6	781.3	841.6	897.2
21	534.7	594.6	657.7	716.9	783.0	843.3	900.8
22	538.3	597.0	659.9	720.2	786.0	846.6	904.4
23	542.4	604.2	665.5	726.6	791.8	851.8	912.1
24	542.4	604.2	665.5	726.6	791.8	851.8	912.1
25	542.4	604.2	665.5	726.6	791.8	851.8	912.1

Table 6.6: (Continued)

	9	10	11	12	13	14	15
26	543.7	605.6	666.3	727.9	793.0	852.8	913.4
27	549.5	611.7	671.3	735.5	798.0	857.2	922.9
28	549.5	611.7	671.3	735.5	798.0	857.2	922.9
29	549.5	611.7	671.3	735.5	798.0	857.2	922.9
30	549.5	611.7	671.3	735.5	798.0	857.2	922.9
31	549.5	611.7	671.3	735.5	798.0	857.2	922.9
32	551.0	613.1	672.8	737.1	799.2	858.4	924.2
33	551.4	614.0	673.1	737.5	800.2	859.1	925.1
34	555.4	620.6	679.1	742.3	806.2	863.4	929.8
35	555.4	620.6	679.1	742.3	806.2	863.4	929.8
36	555.4	620.6	679.1	742.3	806.2	863.4	929.8
37	555.4	620.6	679.1	742.3	806.2	863.4	929.8
38	556.8	621.9	680.2	743.2	807.3	864.5	930.9
39	561.8	627.4	686.9	748.5	815.2	871.3	936.0
40	561.8	627.4	686.9	748.5	815.2	871.3	936.0
41	564.9	620.9	687.1	751.5	808.6	866.5	936.6
42	564.9	620.9	687.1	751.5	808.6	866.5	936.6
43	572.5	632.8	694.9	758.1	821.7	877.1	943.9
44	510.3	572.5	632.8	694.9	758.1	821.7	877.1
59	510.3	572.5	632.8	694.9	758.1	821.7	877.1

Table 6.7: Case 2: Average processing times

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
1	450	454.34	1.01	6921.94	7376.28	16.39
2	450	432.69	0.96	3248.12	3680.81	8.18
3	450	433.72	0.96	205.73	639.45	1.42
4	450	430.22	0.96	0.00	430.22	0.96
5	450	471.96	1.05	0.00	471.96	1.05
6	450	0.00	0.00	3208.85	3208.85	7.13
7	450	2319.87	5.16	0.00	2319.87	5.16
8	450	0.00	0.00	209.23	209.23	0.46
9	450	2236.86	4.97	0.00	2236.86	4.97
10	450	1935.99	4.30	1093.22	3029.20	6.73
11	450	0.00	0.00	1001.58	1001.58	2.23
12	450	0.00	0.00	12.14	12.14	0.03
13	450	2331.48	5.18	1364.17	3695.66	8.21
14	450	0.00	0.00	4584.57	4584.57	10.19
15	450	2115.29	4.70	0.00	2115.29	4.70
16	450	0.00	0.00	1592.50	1592.50	3.54
17	450	0.00	0.00	0.08	0.08	0.00
18	450	2303.18	5.12	0.00	2303.18	5.12
19	450	2177.45	4.84	1215.96	3393.41	7.54
20	450	0.00	0.00	1090.31	1090.31	2.42
21	450	1436.23	3.19	0.00	1436.23	3.19
22	450	2734.28	6.08	0.00	2734.28	6.08
23	450	0.00	0.00	20306.62	20306.62	45.13
24	450	0.00	0.00	0.00	0.00	0.00
25	450	544.09	1.21	0.00	544.09	1.21

Table 6.7: (Continued)

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
26	450	2891.53	6.43	0.00	2891.53	6.43
27	450	0.00	0.00	13396.02	13396.02	29.77
28	450	0.00	0.00	0.00	0.00	0.00
29	450	0.00	0.00	0.00	0.00	0.00
30	450	569.25	1.26	0.00	569.25	1.26
31	450	902.33	2.01	0.00	902.33	2.01
32	450	2682.75	5.96	0.00	2682.75	5.96
33	450	4821.48	10.71	0.00	4821.48	10.71
34	450	0.00	0.00	7278.84	7278.84	16.18
35	450	0.00	0.00	0.00	0.00	0.00
36	450	1383.70	3.07	5305.63	6689.32	14.87
37	450	531.80	1.18	0.00	531.80	1.18
38	450	2642.19	5.87	0.00	2642.19	5.87
39	450	0.00	0.00	3911.17	3911.17	8.69
40	450	1248.01	2.77	2267.44	3515.44	7.81
41	450	0.00	0.00	4659.05	4659.05	10.35
42	450	1380.13	3.07	2837.41	4217.54	9.37
43	450	0.00	0.00	0.00	0.00	0.00
44	450	0.00	0.00	0.00	0.00	0.00
59	450	0.00	0.00	26314.40	26314.40	58.48

Table 6.8: Case 2: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	33.1	39.8	38.5	38.2	44.9	43.7	38.5	45.2	43.9
ACPT :	33.1	39.8	38.5	38.2	44.9	43.7	38.5	45.2	43.9
NO.	10	11	12	13	14	15	16	17	18
TIME :	42.5	49.2	48.0	42.8	49.5	48.3	42.0	48.8	47.5
ACPT :	42.5	49.2	48.0	42.8	49.5	48.3	42.0	48.8	47.5
NO.	19	20	21	22	23	24	25	26	27
TIME :	42.3	49.0	47.8	42.5	49.2	48.0	42.8	49.5	47.8
ACPT :	42.3	49.0	47.8	42.5	49.2	48.0	42.8	49.5	47.8
NO.	28	29	30	31	32	33	34	35	36
TIME :	42.5	44.1	47.5	42.3	49.0	47.8	43.2	49.9	48.6
ACPT :	42.5	44.1	47.5	42.3	49.0	47.8	43.2	49.9	48.6
NO.	37	38	39	40	41	42	43	44	45
TIME :	43.4	50.2	48.9	42.7	49.4	48.2	42.9	49.7	48.4
ACPT :	43.4	50.2	48.9	42.7	49.4	48.2	42.9	49.7	48.4
NO.	46	47	48	49	50	51	52	53	54
TIME :	38.3	45.0	43.8	38.6	45.3	44.0	38.2	45.0	43.7
ACPT :	38.3	45.0	43.8	38.6	45.3	44.0	38.2	45.0	43.7
NO.	55	56	57	58	59	60	61	62	63
TIME :	38.5	45.3	44.0	39.6	39.9	39.6	39.9	45.0	45.3
ACPT :	38.5	45.3	44.0	39.6	39.9	39.6	39.9	45.0	45.3
NO.	64	65	66	67	68	69	70	71	72
TIME :	45.7	46.0	38.0	38.3	26.9	38.5	45.5	46.2	27.2
ACPT :	45.7	46.0	38.0	38.3	26.9	38.5	45.5	46.2	27.2
NO.	73	74	75	76					
TIME :	38.7	45.8	46.5	14.1					
ACPT :	38.7	45.8	46.5	14.1					

$$
\begin{aligned}
&= .0215 \ (= f_{75}) \\
\phi_{ihg} &= \min\left\{f_{58}, f_{59}, f_{60}, f_{61}, \frac{1}{\overline{u}_7}, \frac{1}{\overline{u}_{31}}, \frac{1}{\overline{u}_{33}}\right\} \\
&= \min\left\{\frac{1}{39.6}, \frac{1}{39.9}, \frac{1}{39.6}, \frac{1}{39.9}, \frac{1}{5.16}, \frac{1}{2.01}, \frac{1}{10.71}\right\} \\
&= .0251 \ (= f_{61}) \ .
\end{aligned}
$$

The maximum throughput rate of this case is also determined by the structural constraint, since the circuit, δ_{38}, is also the critical circuit which consists of the processes covered by the overall system, not any specific C^2 process. Therefore, as for the previous case, the remedy to improve the net performance is either increasing the Q_{sys} or reducing the processing times of the critical circuit. By comparing the previous critical circuit to the present critical circuit in terms of their processing times of the tasks, a bottle-neck can be easily found, since the other processing times are equivalent to each other, except p_{10} in δ_{11} and p_9 and p_{18} in δ_{38}. Therefore, the only way to improve the effectiveness is to reduce μ_9 and μ_{18}.

6.2.3 Case 3: Reduced processing time of p_9 and p_{18}

As the previous case, reduce μ_9 and μ_{18} to their maximum available. Then the maximum throughput rate changes from $\Phi = 1/\alpha(\delta_{38}) = .0199$ to $\Phi = 1/\alpha(\delta_{11}) = 1/49.2 = .0203$. (See Table 6.9.) This case also indicates that the Φ is determined by the structural constraint. From Cases 1 and 2, note that there is no further way to improve the system effectiveness with the processing times, since the critical circuit, δ_{11}, not only characterizes that Φ is determined by the structural constraint, but also all related processing times to the critical circuit have been reduced. Therefore, the rest of the remedies to improve the system effectiveness is to increase the capacity of the system. Next, the effect by increasing the system capacity to the maximum will be investigated.

6.2.4 Case 4: Maximum system capacity

Taking the upper limit of the system's capacity (Q_{sys}=3), the result can be obtained as shown in Table 6.10. The maximum throughput rate is, $\Phi = 1/\alpha(\delta_{70}) = 1/46.2 = .0216$ and the average circuit processing rate of each C^2 process is

$$\phi_{cmd} = f_{76} = .0714, \quad \phi_{cic} = f_{68} = .0370, \quad \phi_{bg} = f_{69} = .0255,$$
$$\phi_{sbg} = f_{70} = .0216, \quad \phi_{ibg} = f_{58} = .0269 \ .$$

Therefore, from the above result, the maximum processing rate of the SBG process reflects to the maximum throughput rate of the overall net in this case. Now, there is only one possible alternative to improve the system effectiveness, i.e., the increasing of the SBG's capacity, since all the processing times related to critical circuit, δ_{70}, have been already reduced.

6.2.5 Case 5: Maximum SBG capacity

By increasing the capacity of the SBG process up to its maximum (Q_{sbg}=3), the maximum throughput rate can be obtained from Table 6.11 as follows.

$$\Phi = \frac{1}{\alpha(\delta_{66})} = \frac{1}{38.1} = .0262 \ .$$

Comparing the critical circuit of Case 5 to the critical circuit of Case 4, the fact that only one task, p_{15} needs to be reduced can be found. Therefore, the next move

Table 6.9: Case 3: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	33.1	39.8	38.5	38.2	44.9	43.7	36.4	43.2	41.9
ACPT :	33.1	39.8	38.5	38.2	44.9	43.7	36.4	43.2	41.9
NO.	10	11	12	13	14	15	16	17	18
TIME :	42.5	49.2	48.0	40.7	47.5	46.2	42.0	48.7	47.5
ACPT :	42.5	49.2	48.0	40.7	47.5	46.2	42.0	48.7	47.5
NO.	19	20	21	22	23	24	25	26	27
TIME :	40.2	47.0	45.7	42.5	49.2	48.0	40.7	47.5	45.7
ACPT :	40.2	47.0	45.7	42.5	49.2	48.0	40.7	47.5	45.7
NO.	28	29	30	31	32	33	34	35	36
TIME :	42.5	44.1	47.5	40.2	47.0	45.7	41.2	47.9	46.6
ACPT :	42.5	44.1	47.5	40.2	47.0	45.7	41.2	47.9	46.6
NO.	37	38	39	40	41	42	43	44	45
TIME :	39.4	46.1	44.9	40.7	47.4	46.1	38.9	45.6	44.4
ACPT :	39.4	46.1	44.9	40.7	47.4	46.1	38.9	45.6	44.4
NO.	46	47	48	49	50	51	52	53	54
TIME :	38.3	45.0	43.8	36.5	43.3	42.0	38.3	45.0	43.7
ACPT :	38.3	45.0	43.8	36.5	43.3	42.0	38.3	45.0	43.7
NO.	55	56	57	58	59	60	61	62	63
TIME :	36.5	43.2	42.0	39.6	37.9	39.6	37.9	45.0	43.2
ACPT :	36.5	43.2	42.0	39.6	37.9	39.6	37.9	45.0	43.2
NO.	64	65	66	67	68	69	70	71	72
TIME :	43.7	41.9	38.0	36.2	26.9	38.5	45.5	44.2	25.2
ACPT :	43.7	41.9	38.0	36.2	26.9	38.5	45.5	44.2	25.2
NO.	73	74	75	76					
TIME :	36.7	43.7	42.4	14.1					
ACPT :	36.7	43.7	42.4	14.1					

Table 6.10: Case 4: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	33.1	40.1	36.2	38.1	45.1	41.2	36.3	43.2	39.3
ACPT	:	11.0	13.4	12.1	12.7	15.0	13.7	12.1	14.4	13.1
NO.		10	11	12	13	14	15	16	17	18
TIME	:	43.0	50.0	46.1	41.2	48.1	44.2	43.0	49.9	46.0
ACPT	:	14.3	16.7	15.4	13.7	16.0	14.7	14.3	16.6	15.3
NO.		19	20	21	22	23	24	25	26	27
TIME	:	41.1	48.1	44.2	43.1	50.1	46.1	41.2	48.2	44.2
ACPT	:	13.7	16.0	14.7	14.4	16.7	15.4	13.7	16.1	14.7
NO.		28	29	30	31	32	33	34	35	36
TIME	:	43.1	45.0	46.1	41.1	48.1	44.2	41.0	48.0	44.1
ACPT	:	14.4	15.0	15.4	13.7	16.0	14.7	13.7	16.0	14.7
NO.		37	38	39	40	41	42	43	44	45
TIME	:	39.1	46.1	42.2	40.9	47.9	44.0	39.1	46.0	42.1
ACPT	:	13.0	15.4	14.1	13.6	16.0	14.7	13.0	15.3	14.0
NO.		46	47	48	49	50	51	52	53	54
TIME	:	38.1	45.1	41.2	36.3	43.2	39.3	37.5	44.5	40.6
ACPT	:	12.7	15.0	13.7	12.1	14.4	13.1	12.5	14.8	13.5
NO.		55	56	57	58	59	60	61	62	63
TIME	:	35.7	42.7	38.7	37.2	35.4	36.6	34.8	46.1	44.2
ACPT	:	11.9	14.2	12.9	37.2	35.4	36.6	34.8	46.1	44.2
NO.		64	65	66	67	68	69	70	71	72
TIME	:	44.0	42.1	39.1	37.2	27.0	39.2	46.2	44.1	25.2
ACPT	:	44.0	42.1	39.1	37.2	27.0	39.2	46.2	44.1	25.2
NO.		73	74	75	76					
TIME	:	37.3	44.3	42.2	14.0					
ACPT	:	37.3	44.3	42.2	14.0					

Table 6.11: Case 5: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	32.3	39.1	36.5	37.0	43.9	41.2	35.1	42.0	39.3
ACPT :	10.8	13.0	12.2	12.3	14.6	13.7	11.7	14.0	13.1
NO.	10	11	12	13	14	15	16	17	18
TIME :	41.6	48.5	45.8	39.7	46.6	43.9	41.8	48.7	46.0
ACPT :	13.9	16.2	15.3	13.2	15.5	14.6	13.9	16.2	15.3
NO.	19	20	21	22	23	24	25	26	27
TIME :	39.9	46.8	44.1	41.7	48.6	45.9	39.8	46.7	44.2
ACPT :	13.3	15.6	14.7	13.9	16.2	15.3	13.3	15.6	14.7
NO.	28	29	30	31	32	33	34	35	36
TIME :	41.7	43.8	46.1	40.0	46.9	44.2	40.0	46.9	44.2
ACPT :	13.9	14.6	15.4	13.3	15.6	14.7	13.3	15.6	14.7
NO.	37	38	39	40	41	42	43	44	45
TIME :	38.1	45.0	42.3	40.2	47.1	44.4	38.3	45.2	42.5
ACPT :	12.7	15.0	14.1	13.4	15.7	14.8	12.8	15.1	14.2
NO.	46	47	48	49	50	51	52	53	54
TIME :	37.0	43.9	41.2	35.1	42.0	39.3	37.0	43.9	41.2
ACPT :	12.3	14.6	13.7	11.7	14.0	13.1	12.3	14.6	13.7
NO.	55	56	57	58	59	60	61	62	63
TIME :	35.1	41.9	39.3	37.4	35.4	37.4	35.4	44.7	42.8
ACPT :	11.7	14.0	13.1	37.4	35.4	37.4	35.4	14.9	14.3
NO.	64	65	66	67	68	69	70	71	72
TIME :	43.0	41.1	38.1	36.2	26.4	37.9	44.5	42.8	24.5
ACPT :	14.3	13.7	38.1	36.2	26.4	37.9	14.8	14.3	24.5
NO.	73	74	75	76					
TIME :	36.0	42.6	40.9	14.5					
ACPT :	36.0	14.2	13.6	14.5					

is to reduce μ_{15}. By iterating these procedures, the most effective normal model can be obtained under the given assumptions.

On the other hand, the average response time can be calculated from task processing history tables. For example, the average response time of Case 1 is

$$\begin{aligned}\overline{RT} &= \frac{(S_{43}^{450} - S_{1}^{450})}{450} \\ &= \frac{(\overline{S}_{43}^{15} - \overline{S}_{1}^{15})}{15} \\ &= \frac{(1342.7 - 0.0)}{15} \\ &= 89.5 \ ,\end{aligned}$$

where 450 indicates the total number of attacks (or incoming information) and 15 denotes total number of attacks per run. The inverse of this value is very close to the Φ_1.

$$\Phi_1 = 0.0125 \doteq \overline{RT}_1 = 1/89.5 = 0.0112 \ .$$

All input changes and results according to the procedures stated so far in this section are summarized in Table 6.17.

As the result of the analysis of the normal model, N, the model with the best effectiveness under the given conditions is the model which has a maximum system and process capacities, except a CMD process with reduced processing times on corresponding critical circuits. Thus, the conclusion can be derived as in the present C^2 system, all personnel are trained and stimulated more in order to execute their tasks faster. Additionally, 14 more capacities (or people) should be supplemented.

Table 6.12: Case 6: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	32.2	39.2	36.4	37.0	43.9	41.2	35.1	42.0	39.3
ACPT	:	10.7	13.1	12.1	12.3	14.6	13.7	11.7	14.0	13.1
NO.		10	11	12	13	14	15	16	17	18
TIME	:	41.6	48.5	45.8	39.7	46.6	43.9	39.8	46.7	44.0
ACPT	:	13.9	16.2	15.3	13.2	15.5	14.6	13.3	15.6	14.7
NO.		19	20	21	22	23	24	25	26	27
TIME	:	37.9	44.8	42.1	41.7	48.6	45.9	39.8	46.7	42.2
ACPT	:	12.6	14.9	14.0	13.9	16.2	15.3	13.3	15.6	14.1
NO.		28	29	30	31	32	33	34	35	36
TIME	:	41.7	43.9	44.1	38.0	44.9	42.2	40.0	46.9	44.2
ACPT	:	13.9	14.6	14.7	12.7	15.0	14.1	13.3	15.6	14.7
NO.		37	38	39	40	41	42	43	44	45
TIME	:	38.1	45.0	42.3	38.2	45.1	42.4	36.3	43.2	40.5
ACPT	:	12.7	15.0	14.1	12.7	15.0	14.1	12.1	14.4	13.5
NO.		46	47	48	49	50	51	52	53	54
TIME	:	36.9	43.9	41.1	35.1	42.0	39.2	37.0	43.9	41.1
ACPT	:	12.3	14.6	13.7	11.7	14.0	13.1	12.3	14.6	13.7
NO.		55	56	57	58	59	60	61	62	63
TIME	:	35.1	42.0	39.2	37.3	35.4	37.3	35.4	42.7	40.8
ACPT	:	11.7	14.0	13.1	37.3	35.4	37.3	35.4	14.2	13.6
NO.		64	65	66	67	68	69	70	71	72
TIME	:	41.0	39.1	36.1	34.2	26.3	37.9	44.5	42.8	24.4
ACPT	:	13.7	13.0	36.1	34.2	26.3	37.9	14.8	14.3	24.4
NO.		73	74	75	76					
TIME	:	36.0	42.6	40.9	14.5					
ACPT	:	36.0	14.2	13.6	14.5					

Table 6.13: Case 7: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	32.0	39.0	36.1	36.8	43.7	40.8	34.7	41.7	38.8
ACPT	:	10.7	13.0	12.0	12.3	14.6	13.6	11.6	13.9	12.9
NO.		10	11	12	13	14	15	16	17	18
TIME	:	41.3	48.2	45.3	39.2	46.2	43.3	39.7	46.6	43.7
ACPT	:	13.8	16.1	15.1	13.1	15.4	14.4	13.2	15.5	14.6
NO.		19	20	21	22	23	24	25	26	27
TIME	:	37.6	44.6	41.7	41.3	48.3	45.4	39.3	46.2	41.7
ACPT	:	12.5	14.9	13.9	13.8	16.1	15.1	13.1	15.4	13.9
NO.		28	29	30	31	32	33	34	35	36
TIME	:	41.3	43.5	43.8	37.7	44.6	41.7	39.7	46.7	43.8
ACPT	:	13.8	14.5	14.6	12.6	14.9	13.9	13.2	15.6	14.6
NO.		37	38	39	40	41	42	43	44	45
TIME	:	37.7	44.6	41.8	38.1	45.1	42.2	36.1	43.0	40.2
ACPT	:	12.6	14.9	13.9	12.7	15.0	14.1	12.0	14.3	13.4
NO.		46	47	48	49	50	51	52	53	54
TIME	:	36.7	43.7	40.8	34.7	41.6	38.7	36.9	43.9	41.0
ACPT	:	12.2	14.6	13.6	11.6	13.9	12.9	12.3	14.6	13.7
NO.		55	56	57	58	59	60	61	62	63
TIME	:	34.9	41.8	38.9	37.1	35.0	37.3	35.2	42.6	40.6
ACPT	:	11.6	13.9	13.0	37.1	35.0	37.3	35.2	14.2	13.5
NO.		64	65	66	67	68	69	70	71	72
TIME	:	41.1	39.0	35.8	33.7	25.9	37.4	44.3	42.7	23.9
ACPT	:	13.7	13.0	8.9	8.4	25.9	9.3	14.8	14.2	23.9
NO.		73	74	75	76					
TIME	:	35.3	42.2	40.6	14.2					
ACPT	:	8.8	14.1	13.5	14.2					

Table 6.14: Case 8: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	32.8	39.6	32.3	37.3	44.2	36.9	35.0	41.9	34.6
ACPT	:	10.9	13.2	10.8	12.4	14.7	12.3	11.7	14.0	11.5
NO.		10	11	12	13	14	15	16	17	18
TIME	:	42.0	48.8	41.5	39.7	46.5	39.3	40.4	47.3	40.0
ACPT	:	14.0	16.3	13.8	13.2	15.5	13.1	13.5	15.8	13.3
NO.		19	20	21	22	23	24	25	26	27
TIME	:	38.1	45.0	37.7	42.0	48.9	41.6	39.7	46.6	37.7
ACPT	:	12.7	15.0	12.6	14.0	16.3	13.9	13.2	15.5	12.6
NO.		28	29	30	31	32	33	34	35	36
TIME	:	42.0	44.3	40.0	38.2	45.0	37.7	40.1	46.9	39.7
ACPT	:	14.0	14.8	13.3	12.7	15.0	12.6	13.4	15.6	13.2
NO.		37	38	39	40	41	42	43	44	45
TIME	:	37.8	44.7	37.4	38.5	45.4	38.1	36.3	43.1	35.8
ACPT	:	12.6	14.9	12.5	12.8	15.1	12.7	12.1	14.4	11.9
NO.		46	47	48	49	50	51	52	53	54
TIME	:	37.4	44.2	36.9	35.1	41.9	34.6	35.9	42.7	35.5
ACPT	:	12.5	14.7	12.3	11.7	14.0	11.5	12.0	14.2	11.8
NO.		55	56	57	58	59	60	61	62	63
TIME	:	33.6	40.5	33.2	33.3	31.0	31.9	29.6	43.3	41.1
ACPT	:	11.2	13.5	11.1	33.3	31.0	31.9	29.6	14.4	13.7
NO.		64	65	66	67	68	69	70	71	72
TIME	:	41.4	39.1	36.4	34.1	26.4	37.9	44.9	43.0	24.1
ACPT	:	13.8	13.0	9.1	8.5	26.4	9.5	15.0	14.3	24.1
NO.		73	74	75	76					
TIME	:	35.6	42.6	40.7	14.9					
ACPT	:	8.9	14.2	13.6	14.9					

Table 6.15: Case 9: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	30.8	38.3	30.8	35.7	43.2	35.7	34.0	41.6	34.1
ACPT	:	10.3	12.8	10.3	11.9	14.4	11.9	11.3	13.9	11.4
NO.		10	11	12	13	14	15	16	17	18
TIME	:	40.9	48.4	40.9	39.3	46.8	39.3	39.0	46.5	39.0
ACPT	:	13.6	16.1	13.6	13.1	15.6	13.1	13.0	15.5	13.0
NO.		19	20	21	22	23	24	25	26	27
TIME	:	37.4	44.9	37.4	40.8	48.3	40.9	39.2	46.7	37.3
ACPT	:	12.5	15.0	12.5	13.6	16.1	13.6	13.1	15.6	12.4
NO.		28	29	30	31	32	33	34	35	36
TIME	:	40.8	43.5	39.0	37.3	44.8	37.3	38.7	46.2	38.7
ACPT	:	13.6	14.5	13.0	12.4	14.9	12.4	12.9	15.4	12.9
NO.		37	38	39	40	41	42	43	44	45
TIME	:	37.1	44.6	37.1	36.8	44.3	36.8	35.2	42.7	35.2
ACPT	:	12.4	14.9	12.4	12.3	14.8	12.3	11.7	14.2	11.7
NO.		46	47	48	49	50	51	52	53	54
TIME	:	35.7	43.2	35.7	34.1	41.6	34.1	33.8	41.3	33.8
ACPT	:	11.9	14.4	11.9	11.4	13.9	11.4	11.3	13.8	11.3
NO.		55	56	57	58	59	60	61	62	63
TIME	:	32.2	39.7	32.2	31.9	30.3	29.9	28.3	42.3	40.7
ACPT	:	10.7	13.2	10.7	10.6	10.1	10.0	9.4	14.1	13.6
NO.		64	65	66	67	68	69	70	71	72
TIME	:	40.2	38.6	34.8	33.2	25.1	36.7	44.2	42.1	23.5
ACPT	:	13.4	12.9	8.7	8.3	25.1	9.2	14.7	14.0	23.5
NO.		73	74	75	76					
TIME	:	35.1	42.6	40.5	13.5					
ACPT	:	8.8	14.2	13.5	13.5					

Table 6.16: Case 10: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	32.2	39.5	31.5	36.5	43.8	35.8	34.7	42.0	34.0
ACPT	:	10.7	13.2	10.5	12.2	14.6	11.9	11.6	14.0	11.3
NO.		10	11	12	13	14	15	16	17	18
TIME	:	41.0	48.3	40.3	39.3	46.6	38.6	39.8	47.1	39.1
ACPT	:	13.7	16.1	13.4	13.1	15.5	12.9	13.3	15.7	13.0
NO.		19	20	21	22	23	24	25	26	27
TIME	:	38.1	45.4	37.4	41.0	48.3	40.3	39.2	46.5	37.3
ACPT	:	12.7	15.1	12.5	13.7	16.1	13.4	13.1	15.5	12.4
NO.		28	29	30	31	32	33	34	35	36
TIME	:	41.0	44.0	39.1	38.0	45.3	37.3	39.5	46.8	38.8
ACPT	:	13.7	14.7	13.0	12.7	15.1	12.4	13.2	15.6	12.9
NO.		37	38	39	40	41	42	43	44	45
TIME	:	37.7	45.0	37.0	38.3	45.6	37.6	36.5	43.8	35.8
ACPT	:	12.6	15.0	12.3	12.8	15.2	12.5	12.2	14.6	11.9
NO.		46	47	48	49	50	51	52	53	54
TIME	:	36.5	43.8	35.8	34.7	42.0	34.1	35.1	42.4	34.5
ACPT	:	12.2	14.6	11.9	11.6	14.0	11.4	11.7	14.1	11.5
NO.		55	56	57	58	59	60	61	62	63
TIME	:	33.4	40.7	32.7	31.9	30.1	30.5	28.8	43.4	41.6
ACPT	:	11.1	13.6	10.9	10.6	10.0	10.2	9.6	14.5	13.9
NO.		64	65	66	67	68	69	70	71	72
TIME	:	41.9	40.1	35.9	34.2	25.2	37.1	44.6	43.1	23.5
ACPT	:	14.0	13.4	9.0	8.6	6.3	9.3	14.9	14.4	5.9
NO.		73	74	75	76					
TIME	:	35.4	42.8	41.3	13.7					
ACPT	:	8.9	14.3	13.8	13.7					

Table 6.17: Effectivenesses of model N

Cases	Q	reduced μ's	Φ	RT
1	All Qs = 1	none	0.0125	89.5
2	"	μ's on δ_{11}	0.0199	62.9
3	"	μ_9, μ_{18}	0.0203	60.2
4	Q_{sys}=3	none	0.0216	55.7
5	Q_{sbg}=3	none	0.0262	49.5
6	"	μ_{15}	0.0264	48.5
7	Q_{bg}=4	none	0.0268	43.7
8	Q_{sbg}=3	μ_7, μ_{31}, μ_{33}	0.0300	38.9
9	Q_{ibg}=3	none	0.0398	30.2
10	Q_{cic}=4	none	0.0428	24.9

6.3 Change Model, C

As stated in Chapter 3, the system should be designed to absorb shocks from out-world and/or in the system, especially if the system is the battle system. Therefore, in this thesis, the unit destruction due to the enemy's attacks are considered and simulated with hit probabilities changing their values according to the number of survival units.

In this section, the change model, C, is analyzed by using the same procedures taken in the previous subsection. Then, the condition that the model has the best effectiveness will be investigated.

6.3.1 Case 1: Original input

The change model with one capacity for each C^2 process is analyzed by using the processing times assigned in Section 5.5.2. From Table 6.13, the maximum average circuit processing time can be directly obtained.

$$\begin{aligned}\alpha &= \max\left\{\alpha(\delta_1),\, \alpha(\delta_2), \cdots, \alpha(\delta_{76})\right\} \\ &= \max\{48.1,\, 52.0, \cdots, 22.6\} \\ &= 65.7\ (= \alpha(\delta_{11}))\ .\end{aligned}$$

Therefore, the maximum throughput rate, Φ, is

$$\Phi = \frac{1}{\alpha(\delta_{11})} = \frac{1}{77} = .0152\ .$$

Also, the maximum processing rate of each C^2 process, ϕ, is

$$\begin{aligned}\phi_{cmd} &= f_{76} = \frac{1}{22.6} = .0442 \\ \phi_{cic} &= f_{68} = \frac{1}{46.5} = .0215 \\ \phi_{bg} &= f_{69} = \frac{1}{58.2} = .0172 \\ \phi_{sbg} &= f_{70} = \frac{1}{61.9} = .0162 \\ \phi_{ibg} &= f_{58} = \frac{1}{55.0} = .0182\ .\end{aligned}$$

The above results indicate that the maximum throughput rate of the system is determined only by the structural constraint since δ_{11} is the unique critical circuit of the net. Therefore, in this case, two possible remedies to improve the effectiveness of the overall system can be considered: either increase the capacities available to the system or reduce the task processing times of its subprocesses or both.

Therefore, as performed in the previous section, the processing times on the critical circuit are reduced as the treatment to get a more effective Change system. Then, this result is compared to the original and its effect by increasing the capacity investigated.

Table 6.18: Case 1: Task processing history

	1	2	3	4	5	6	7	8
1	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
2	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
3	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
4	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
5	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
6	1.0	86.3	176.2	253.9	329.7	408.1	482.4	560.6
7	1.0	86.3	176.2	253.9	329.7	408.1	482.4	560.6
8	0.9	86.2	176.1	253.6	329.5	407.6	481.6	560.3
9	0.9	86.2	176.1	253.6	329.5	407.6	481.6	560.3
10	1.4	86.7	176.5	254.5	330.1	408.2	482.4	560.7
11	6.8	89.7	178.7	256.0	331.1	409.3	482.8	561.6
12	12.1	94.0	182.1	260.8	334.5	414.3	489.5	565.5
13	12.1	94.0	182.5	261.1	335.0	414.8	490.2	566.1
14	4.6	89.8	182.2	257.8	333.6	412.2	487.8	563.1
15	12.1	94.0	182.1	260.8	334.5	414.3	489.5	565.5
16	15.3	98.0	185.9	264.3	340.2	417.4	492.9	567.6
17	21.9	102.5	191.1	274.0	349.4	428.3	504.3	580.5
18	22.1	108.0	194.2	273.6	348.1	425.1	501.9	575.9
19	21.9	102.5	191.1	274.0	349.4	428.4	504.3	580.6
20	27.3	107.9	195.3	279.3	354.7	431.8	510.7	585.2
21	32.1	110.7	199.9	283.3	359.2	436.1	516.8	588.3
22	37.1	116.4	204.5	287.7	362.9	439.5	521.7	592.0
23	46.8	126.2	213.9	296.3	371.4	452.4	532.1	600.0
24	46.8	126.2	213.9	296.3	371.4	452.4	532.1	600.0
25	46.8	126.2	213.9	296.3	371.4	452.4	532.1	600.0
26	48.3	132.8	218.7	295.0	371.3	449.1	528.1	594.8
27	56.3	144.8	227.7	308.0	386.2	463.6	539.2	606.3
28	56.3	144.8	227.7	308.0	386.2	463.6	539.2	606.3
29	56.3	144.8	227.7	308.0	386.2	463.6	539.2	606.3
30	56.3	144.8	227.7	308.0	386.2	463.6	539.2	606.3
31	56.3	144.8	227.7	308.0	386.2	463.6	539.2	606.3
32	57.9	146.7	229.3	309.2	387.7	465.1	540.5	607.7
33	58.5	146.3	229.5	309.5	387.7	465.2	540.5	607.4
34	70.0	157.6	237.0	315.9	395.1	471.5	547.4	611.5
35	70.0	157.6	237.0	315.9	395.1	471.5	547.4	611.5
36	70.0	157.6	237.0	315.9	395.1	471.5	547.4	611.5
37	70.0	157.6	237.0	315.9	395.1	471.5	547.4	611.5
38	71.4	159.7	238.2	316.8	395.9	472.3	547.8	611.7
39	80.3	168.2	245.7	322.3	399.3	474.0	550.3	613.3
40	80.3	168.2	245.7	322.3	399.3	474.0	550.3	613.3

Table 6.18: (Continued)

	1	2	3	4	5	6	7	8
41	67.6	156.3	238.7	316.0	396.3	472.5	548.4	615.6
42	67.6	156.3	238.7	316.0	396.3	472.5	548.4	615.6
43	85.4	175.2	253.1	328.9	407.2	481.5	560.1	624.2
44	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1
59	0.0	85.4	175.2	253.1	328.9	407.2	481.5	560.1

	9	10	11	12	13	14	15
1	624.2	691.5	758.8	816.9	882.6	949.5	991.7
2	624.2	691.5	758.8	816.9	882.6	949.5	991.7
3	624.2	691.5	758.8	816.9	882.6	949.5	991.7
4	624.2	691.5	758.8	816.9	882.6	949.5	991.7
5	624.2	691.5	758.8	816.9	882.6	949.5	991.7
6	624.9	692.2	759.5	817.5	883.2	950.1	992.1
7	624.9	692.2	759.5	817.5	883.2	950.1	992.1
8	624.5	691.6	759.0	817.0	882.6	949.6	991.8
9	624.5	691.6	759.0	817.0	882.6	949.6	991.8
10	625.0	692.1	759.6	817.2	882.8	949.7	992.0
11	625.0	692.2	759.9	817.1	882.6	949.7	991.9
12	630.1	696.2	763.1	820.6	885.4	951.1	994.5
13	630.7	697.2	763.8	821.2	886.1	951.8	995.1
14	627.3	695.2	762.5	820.1	885.7	951.9	994.4
15	630.1	696.2	763.1	820.6	885.4	951.1	994.5
16	632.9	698.5	764.9	822.3	887.8	952.0	995.8
17	644.1	710.8	787.9	850.4	914.6	994.1	1034.3
18	642.2	707.3	773.9	833.5	897.1	962.3	1007.4
19	644.1	710.8	787.9	850.4	914.6	994.2	1034.3
20	648.5	716.7	794.3	856.2	917.8	1001.4	1041.8
21	653.9	720.2	800.4	862.2	929.2	1004.0	1044.8
22	657.9	726.1	804.9	865.5	933.5	1009.0	1051.1
23	665.8	741.0	818.4	873.0	944.7	1018.4	1058.6
24	665.8	741.0	818.4	873.0	944.7	1018.4	1058.6
25	665.8	741.0	818.4	873.0	944.7	1018.4	1058.6

Table 6.18: (Continued)

	9	10	11	12	13	14	15
26	662.0	735.5	799.6	853.0	923.8	981.5	1027.8
27	672.7	748.6	810.1	869.2	933.6	993.6	1042.4
28	672.7	748.6	810.1	869.2	933.6	993.6	1042.4
29	672.7	748.6	810.1	869.2	933.6	993.6	1042.4
30	672.7	748.6	810.1	869.2	933.6	993.6	1042.4
31	672.7	748.6	810.1	869.2	933.6	993.6	1042.4
32	673.6	749.7	811.0	870.3	934.8	994.4	1043.2
33	674.3	749.5	811.0	869.9	934.5	994.3	1043.2
34	678.6	755.6	817.8	874.4	939.5	996.9	1045.2
35	678.6	755.6	817.8	874.4	939.5	996.9	1045.2
36	678.6	755.6	817.8	874.4	939.5	996.9	1045.2
37	678.6	755.6	817.8	874.4	939.5	996.9	1045.2
38	678.8	755.9	818.2	874.5	939.5	997.0	1045.2
39	680.4	758.0	818.7	874.8	939.8	997.3	1045.2
40	680.4	758.0	818.7	874.8	939.8	997.3	1045.2
41	681.8	753.9	817.6	874.0	939.9	1000.0	1048.0
42	681.8	753.9	817.6	874.0	939.9	1000.0	1048.0
43	691.5	765.2	828.4	882.6	949.5	1008.0	1054.5
44	624.2	691.5	758.8	816.9	882.6	949.5	991.7
59	624.2	691.5	758.8	816.9	882.6	949.5	991.7

Table 6.19: Case 1: Average processing times

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
1	466	425.67	0.91	7501.97	7927.64	17.01
2	584	625.55	1.07	3126.73	3752.29	6.43
3	584	347.13	0.59	134.06	481.19	0.82
4	584	198.19	0.34	0.00	198.19	0.34
5	584	431.99	0.74	0.00	431.99	0.74
6	584	0.00	0.00	3320.31	3320.31	5.69
7	584	2027.71	3.47	0.00	2027.71	3.47
8	584	0.00	0.00	283.02	283.02	0.48
9	584	906.26	1.55	0.00	906.26	1.55
10	584	2465.50	4.22	506.40	2971.90	5.09
11	584	0.00	0.00	2348.66	2348.66	4.02
12	584	0.00	0.00	299.21	299.21	0.51
13	584	5726.40	9.81	941.90	6668.30	11.42
14	584	0.00	0.00	7960.73	7960.73	13.63
15	584	1649.61	2.82	0.00	1649.61	2.82
16	584	0.00	0.00	5317.96	5317.96	9.11
17	466	0.00	0.00	11.53	11.53	0.02
18	584	3651.84	6.25	0.00	3651.84	6.25
19	466	3724.99	7.99	1040.82	4765.80	10.23
20	466	0.00	0.00	2302.30	2302.30	4.94
21	466	2162.79	4.64	0.00	2162.79	4.64
22	466	4645.16	9.97	0.00	4645.16	9.97
23	466	0.00	0.00	22683.98	22683.98	48.68
24	466	0.00	0.00	0.00	0.00	0.00

Table 6.19: (Continued)

PROCESS NO.	PASSING TIMES	PROCESS TIME	AVE. TIME FOR PROCESS	HOLDING TIME	TOTAL TIME	AVE. ACT. PROC.TIME
25	466	959.96	2.06	0.00	959.96	2.06
26	584	7027.63	12.03	0.00	7027.63	12.03
27	584	0.00	0.00	14838.45	14838.45	25.41
28	584	0.00	0.00	0.00	0.00	0.00
29	584	0.00	0.00	0.00	0.00	0.00
30	584	721.85	1.24	0.00	721.85	1.24
31	584	747.97	1.28	0.00	747.97	1.28
32	584	3668.90	6.28	0.00	3668.90	6.28
33	584	4213.10	7.21	0.00	4213.10	7.21
34	584	0.00	0.00	7604.75	7604.75	13.02
35	584	0.00	0.00	0.00	0.00	0.00
36	584	1715.45	2.94	5462.39	7177.84	12.29
37	584	359.56	0.62	0.00	359.56	0.62
38	584	1787.14	3.06	0.00	1787.14	3.06
39	584	0.00	0.00	5194.92	5194.92	8.90
40	584	1868.21	3.20	3163.12	5031.33	8.62
41	584	0.00	0.00	7000.31	7000.31	11.99
42	584	1781.67	3.05	4825.92	6607.59	11.31
43	584	0.00	0.00	0.00	0.00	0.00
44	584	0.00	0.00	0.00	0.00	0.00
59	584	0.00	0.00	40549.14	40549.14	69.43

Table 6.20: Case 1: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	48.1	52.0	49.2	58.0	62.0	59.1	56.3	60.2	57.4
ACPT	:	48.1	52.0	49.2	58.0	62.0	59.1	56.3	60.2	57.4
NO.		10	11	12	13	14	15	16	17	18
TIME	:	61.8	65.7	62.9	60.0	64.0	61.1	54.8	58.7	55.9
ACPT	:	61.8	65.7	62.9	60.0	64.0	61.1	54.8	58.7	55.9
NO.		19	20	21	22	23	24	25	26	27
TIME	:	53.1	57.0	54.1	61.5	65.5	62.6	59.8	63.7	53.9
ACPT	:	53.1	57.0	54.1	61.5	65.5	62.6	59.8	63.7	53.9
NO.		28	29	30	31	32	33	34	35	36
TIME	:	61.5	55.7	55.6	52.8	56.7	53.9	58.8	62.8	59.9
ACPT	:	61.5	55.7	55.6	52.8	56.7	53.9	58.8	62.8	59.9
NO.		37	38	39	40	41	42	43	44	45
TIME	:	57.1	61.0	58.2	51.9	55.8	53.0	50.1	54.1	51.2
ACPT	:	57.1	61.0	58.2	51.9	55.8	53.0	50.1	54.1	51.2
NO.		46	47	48	49	50	51	52	53	54
TIME	:	57.7	61.6	58.8	56.0	59.9	57.0	51.4	55.3	52.5
ACPT	:	57.7	61.6	58.8	56.0	59.9	57.0	51.4	55.3	52.5
NO.		55	56	57	58	59	60	61	62	63
TIME	:	49.6	53.6	50.7	55.0	53.3	48.7	46.9	54.9	53.2
ACPT	:	49.6	53.6	50.7	55.0	53.3	48.7	46.9	54.9	53.2
NO.		64	65	66	67	68	69	70	71	72
TIME	:	52.3	50.5	51.3	49.5	46.5	58.2	61.9	59.2	44.8
ACPT	:	52.3	50.5	51.3	49.5	46.5	58.2	61.9	59.2	44.8
NO.		73	74	75	76					
TIME	:	56.5	60.2	57.5	22.6					
ACPT	:	56.5	60.2	57.5	22.6					

Table 6.21: Reduced mean processing times of circuit. δ_{23}

place	10	13	19	21	22	25	26	30	32	37	38
old $\bar{\mu}$	8	8	8	5	10	2	10	2	10	2	10
new $\bar{\mu}$	4.8	4.8	4.8	3	6	1.2	6	1.2	6	1.2	6

6.3.2 Case 2: Reduced processing times on δ_{11}

Processing times of the critical circuit. δ_{11}. are reduced as shown in Table 6.14. In this case. the reduced amount of the processing time is assumed to be the maximum available to each process.

From Table 6.17. maximum average circuit processing time. α. is

$$\begin{aligned}\alpha &= \max\left\{\alpha(\delta_1), \cdots. \alpha(\delta_{76})\right\} \\ &= \max\{30.4, 32.6, \cdots, 13.6\} \\ &= 44.1 \,(= \alpha(\delta_{15})) \;.\end{aligned}$$

Therefore. the maximum throughput rate is $\Phi = 1 \;\alpha = 1 \; 47.1 = .0227$. Comparing this result to the previous one (Φ_1=.0140). this model shows a 49% increase in terms of the maximum throughput rate. Now. the maximum processing rates for this case are

$$\begin{aligned}\phi_{cmd} &= f_{76} = \frac{1}{13.6} = .0735 \\ \phi_{cic} &= f_{72} = \frac{1}{29.0} = .0345 \\ \phi_{bg} &= f_{73} = \frac{1}{36.0} = .0278 \\ \phi_{sbg} &= f_{74} = \frac{1}{38.2} = .0262 \\ \phi_{ibg} &= f_{59} = \frac{1}{37.9} = .0264 \;.\end{aligned}$$

The maximum throughput rate for this case is also determined by the structural constraint, since the circuit, δ_{15}, is also the critical circuit which consists of the processes covering the overall system, not any specific C^2 process. Therefore, as for the previous case, the remedy to improve the system effectiveness is either increase the Q_{sys} or reduce the processing times of the critical circuit.

6.3.3 Case 3: Reduced processing time of δ_{15}

By comparing the previous critical circuit to the present critical circuit in terms of the processing times of their tasks, a bottle-neck can be easily found, since the other processing times are equivalent to each other except p_{10} in δ_{11} and p_{18}, p_{31}, and p_{33} in δ_{15}. Therefore, the only way to improve is to reduce all processing times related to those tasks. Reducing μ_{18}, μ_{31}, and μ_{33} by the same rate as the previous case, then the maximum throughput rate is changed from $\Phi = 1/\alpha(\delta_{15}) = .0227$ to $\Phi = 1/\alpha(\delta_{11}) = 1/40.9 = .0244$. (See Table 6.23.) There is no way to improve the model effectiveness more with the processing times, since not only the critical circuit, δ_{11}, characterizes the Φ, but also all related processing times to the critical circuit have been reduced. Therefore, the rest of the remedies to improve the system effectiveness is to increase the capacity of the system. Then, the effect by increasing the system capacity to its maximum will be investigated.

6.3.4 Case 4: Maximum system capacity

Taking the system's capacity as three, the result from Table 6.24 shows a 24.6% improvement ($\Phi_4 = 1/\delta_{70}$= .0304). Also, the average circuit processing rate of each

Table 6.22: Case 2: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	30.4	32.6	35.0	36.5	38.7	41.0	37.5	39.7	42.1
ACPT	:	30.4	32.6	35.0	36.5	38.7	41.0	37.5	39.7	42.1
NO.		10	11	12	13	14	15	16	17	18
TIME	:	38.5	40.7	43.1	39.6	41.8	44.1	35.5	37.7	40.0
ACPT	:	38.5	40.7	43.1	39.6	41.8	44.1	35.5	37.7	40.0
NO.		19	20	21	22	23	24	25	26	27
TIME	:	36.5	38.7	41.1	38.3	40.5	42.8	39.3	41.5	40.8
ACPT	:	36.5	38.7	41.1	38.3	40.5	42.8	39.3	41.5	40.8
NO.		28	29	30	31	32	33	34	35	36
TIME	:	38.3	34.6	39.8	36.3	38.5	40.8	37.3	39.5	41.8
ACPT	:	38.3	34.6	39.8	36.3	38.5	40.8	37.3	39.5	41.8
NO.		37	38	39	40	41	42	43	44	45
TIME	:	38.3	40.6	42.9	34.2	36.4	38.8	35.3	37.5	39.8
ACPT	:	38.3	40.6	42.9	34.2	36.4	38.8	35.3	37.5	39.8
NO.		46	47	48	49	50	51	52	53	54
TIME	:	36.1	38.3	40.7	37.2	39.4	41.7	33.7	35.9	38.3
ACPT	:	36.1	38.3	40.7	37.2	39.4	41.7	33.7	35.9	38.3
NO.		55	56	57	58	59	60	61	62	63
TIME	:	34.8	37.0	39.3	36.8	37.9	34.4	35.5	34.1	35.1
ACPT	:	34.8	37.0	39.3	36.8	37.9	34.4	35.5	34.1	35.1
NO.		64	65	66	67	68	69	70	71	72
TIME	:	33.1	34.2	31.9	33.0	27.9	35.0	37.1	36.2	29.0
ACPT	:	33.1	34.2	31.9	33.0	27.9	35.0	37.1	36.2	29.0
NO.		73	74	75	76					
TIME	:	36.0	38.2	37.2	13.6					
ACPT	:	36.0	38.2	37.2	13.6					

Table 6.23: Case 3: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	30.4	32.8	31.1	36.4	38.9	37.1	35.5	38.0	36.2
ACPT	:	30.4	32.8	31.1	36.4	38.9	37.1	35.5	38.0	36.2
NO.		10	11	12	13	14	15	16	17	18
TIME	:	38.5	40.9	39.2	37.6	40.0	38.3	35.4	37.9	36.1
ACPT	:	38.5	40.9	39.2	37.6	40.0	38.3	35.4	37.9	36.1
NO.		19	20	21	22	23	24	25	26	27
TIME	:	34.5	37.0	35.2	38.2	40.7	38.9	37.3	39.8	34.9
ACPT	:	34.5	37.0	35.2	38.2	40.7	38.9	37.3	39.8	34.9
NO.		28	29	30	31	32	33	34	35	36
TIME	:	38.2	34.8	35.9	34.2	36.7	34.9	37.2	39.7	37.9
ACPT	:	38.2	34.8	35.9	34.2	36.7	34.9	37.2	39.7	37.9
NO.		37	38	39	40	41	42	43	44	45
TIME	:	36.3	38.8	37.0	34.2	36.6	34.9	33.3	35.7	34.0
ACPT	:	36.3	38.8	37.0	34.2	36.6	34.9	33.3	35.7	34.0
NO.		46	47	48	49	50	51	52	53	54
TIME	:	36.1	38.6	36.8	35.2	37.6	35.9	33.7	36.1	34.4
ACPT	:	36.1	38.6	36.8	35.2	37.6	35.9	33.7	36.1	34.4
NO.		55	56	57	58	59	60	61	62	63
TIME	:	32.8	35.2	33.5	33.0	32.1	30.6	29.7	34.1	33.2
ACPT	:	32.8	35.2	33.5	33.0	32.1	30.6	29.7	34.1	33.2
NO.		64	65	66	67	68	69	70	71	72
TIME	:	33.1	32.2	31.9	31.0	27.9	34.9	37.2	36.2	27.0
ACPT	:	33.1	32.2	31.9	31.0	27.9	34.9	37.2	36.2	27.0
NO.		73	74	75	76					
TIME	:	34.0	36.2	35.3	13.6					
ACPT	:	34.0	36.2	35.3	13.6					

C'^2 process is

$$\begin{aligned}
\phi_{cmd} &= f_{76} = \frac{1}{12.9} = .0775, \\
\phi_{cic} &= f_{68} = \frac{1}{26.3} = .0380, \\
\phi_{bg} &= f_{69} = \frac{1}{31.6} = .0316, \\
\phi_{sbg} &= f_{70} = \frac{1}{32.9} = .0304, \\
\phi_{ibg} &= f_{58} = \frac{1}{31.1} = .0322 \ .
\end{aligned}$$

Therefore, from the above result, the maximum processing rate of the SBG process reflects the maximum throughput rate of the overall net in this case. Now, there is only one possible alternative to improve the system effectiveness, i.e., increasing the SBG's capacity, since the processing times related to the critical circuit, δ_{70}, has been already reduced.

6.3.5 Case 5: Maximum SBG capacity

By increasing the capacity of the SBG process up to its maximum (Q_{sbg}=3), the maximum throughput rate can be obtained from Table 6.25 as follows.

$$\Phi = \frac{1}{\alpha(\delta_{69})} = \frac{1}{34.0} = .0294 \ .$$

The result is quite close to that of the previous case (3% decreasing in terms of Φ). This difference can be regarded as the difference due to applying different random sequences to the process as processing times are calculated. This result also implies that only increasing the SBG capacity does not affect the model effectiveness, since the average processing time of the critical circuit, δ_{69}, is influenced by tasks in the BG process. Therefore, further remedy should be considered.

Table 6.24: Case 4: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	28.4	29.8	30.2	33.5	34.9	35.3	32.5	33.9	34.2
ACPT :	9.5	9.9	10.1	11.2	11.6	11.8	10.8	11.3	11.4
NO.	10	11	12	13	14	15	16	17	18
TIME :	35.1	36.4	36.8	34.0	35.4	35.8	32.1	33.5	33.9
ACPT :	11.7	12.1	12.3	11.3	11.8	11.9	10.7	11.2	11.3
NO.	19	20	21	22	23	24	25	26	27
TIME :	31.1	32.5	32.8	34.9	36.3	36.6	33.8	35.2	32.6
ACPT :	10.4	10.8	10.9	11.6	12.1	12.2	11.3	11.7	10.9
NO.	28	29	30	31	32	33	34	35	36
TIME :	34.9	31.1	33.7	30.9	32.3	32.6	33.7	35.1	35.5
ACPT :	11.6	10.4	11.2	10.3	10.8	10.9	11.2	11.7	11.8
NO.	37	38	39	40	41	42	43	44	45
TIME :	32.7	34.0	34.4	30.8	32.2	32.5	29.7	31.1	31.5
ACPT :	10.9	11.3	11.5	10.3	10.7	10.8	9.9	10.4	10.5
NO.	46	47	48	49	50	51	52	53	54
TIME :	33.3	34.7	35.0	32.3	33.6	34.0	32.1	33.4	33.8
ACPT :	11.1	11.6	11.7	10.8	11.2	11.3	10.7	11.1	11.3
NO.	55	56	57	58	59	60	61	62	63
TIME :	31.0	32.4	32.7	31.1	30.0	29.9	28.8	30.0	28.9
ACPT :	10.3	10.8	10.9	31.1	30.0	29.9	28.8	30.0	28.9
NO.	64	65	66	67	68	69	70	71	72
TIME :	28.8	27.8	28.7	27.6	26.3	31.6	32.9	31.7	25.3
ACPT :	28.8	27.8	28.7	27.6	26.3	31.6	32.9	31.7	25.3
NO.	73	74	75	76					
TIME :	30.6	31.8	30.7	12.9					
ACPT :	30.6	31.8	30.7	12.9					

Table 6.25: Case 5: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	30.1	31.2	32.3	35.8	36.9	38.0	34.4	35.5	36.6
ACPT :	10.0	10.4	10.8	11.9	12.3	12.7	11.5	11.8	12.2
NO.	10	11	12	13	14	15	16	17	18
TIME :	37.7	38.8	39.9	36.3	37.4	38.4	34.5	35.6	36.7
ACPT :	12.6	12.9	13.3	12.1	12.5	12.8	11.5	11.9	12.2
NO.	19	20	21	22	23	24	25	26	27
TIME :	33.0	34.1	35.2	37.5	38.6	39.6	36.0	37.1	35.0
ACPT :	11.0	11.4	11.7	12.5	12.9	13.2	12.0	12.4	11.7
NO.	28	29	30	31	32	33	34	35	36
TIME :	37.5	32.8	36.4	32.8	33.9	35.0	36.2	37.4	38.4
ACPT :	12.5	10.9	12.1	10.9	11.3	11.7	12.1	12.5	12.8
NO.	37	38	39	40	41	42	43	44	45
TIME :	34.8	35.9	37.0	33.0	34.1	35.2	31.6	32.7	33.8
ACPT :	11.6	12.0	12.3	11.0	11.4	11.7	10.5	10.9	11.3
NO.	46	47	48	49	50	51	52	53	54
TIME :	35.5	36.6	37.7	34.1	35.2	36.3	33.7	34.8	35.9
ACPT :	11.8	12.2	12.6	11.4	11.7	12.1	11.2	11.6	12.0
NO.	55	56	57	58	59	60	61	62	63
TIME :	32.3	33.4	34.5	33.6	32.2	31.8	30.4	32.1	30.7
ACPT :	10.8	11.1	11.5	33.6	32.2	31.8	30.4	10.7	10.2
NO.	64	65	66	67	68	69	70	71	72
TIME :	30.9	29.5	30.8	29.4	27.9	34.0	35.4	34.2	26.5
ACPT :	10.3	9.8	30.8	29.4	27.9	34.0	11.8	11.4	26.5
NO.	73	74	75	76					
TIME :	32.6	33.9	32.7	13.9					
ACPT :	32.6	11.3	10.9	13.9					

6.3.6 Case 6: Maximum BG capacity

As stated in Case 5, the circuit, δ_{69}, is influenced by other C^2 processes which the processes of the circuit is going along. Therefore, the remedy taken in this case is to increase the circuit which has the next biggest average circuit processing time. By increasing the BG capacity to its maximum, the maximum throughput rate is increased from Φ_5=.0294 to $\Phi_6 = .0308$. However, comparing this to Φ_4, they are almost same. Therefore, the capacity increasing of C^2 process with the next biggest circuit is performed.

By iterating these procedures, the final best effective model structure is determined. All input changes and results are summarized in Table 6.30. Note that the effectiveness of the final model is not absolutely the best, but it is the best under the given conditions.

Note that results of both models-model N and model C- does not consider the personnel training cost and the manpower cost by increasing the process capacities. Therefore, the cost/effectiveness analysis is an another matter to be analyzed, but it is not a concern in this thesis.

Table 6.26: Case 6: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	29.0	30.1	31.4	34.1	35.1	36.5	33.0	34.0	35.4
ACPT	:	9.7	10.0	10.5	11.4	11.7	12.2	11.0	11.3	11.8
NO.		10	11	12	13	14	15	16	17	18
TIME	:	35.6	36.7	38.0	34.5	35.5	36.9	32.8	33.8	35.2
ACPT	:	11.9	12.2	12.7	11.5	11.8	12.3	10.9	11.3	11.7
NO.		19	20	21	22	23	24	25	26	27
TIME	:	31.7	32.7	34.1	35.3	36.4	37.7	34.2	35.3	33.8
ACPT	:	10.6	10.9	11.4	11.8	12.1	12.6	11.4	11.8	11.3
NO.		28	29	30	31	32	33	34	35	36
TIME	:	35.3	31.3	34.9	31.4	32.4	33.8	34.6	35.6	37.0
ACPT	:	11.8	10.4	11.6	10.5	10.8	11.3	11.5	11.9	12.3
NO.		37	38	39	40	41	42	43	44	45
TIME	:	33.5	34.5	35.9	31.8	32.8	34.2	30.7	31.7	33.1
ACPT	:	11.2	11.5	12.0	10.6	10.9	11.4	10.2	10.6	11.0
NO.		46	47	48	49	50	51	52	53	54
TIME	:	33.9	35.0	36.3	32.8	33.9	35.2	32.2	33.2	34.6
ACPT	:	11.3	11.7	12.1	10.9	11.3	11.7	10.7	11.1	11.5
NO.		55	56	57	58	59	60	61	62	63
TIME	:	31.1	32.1	33.5	32.5	31.4	30.8	29.7	30.4	29.3
ACPT	:	10.4	10.7	11.2	32.5	31.4	30.8	29.7	10.1	9.8
NO.		64	65	66	67	68	69	70	71	72
TIME	:	29.7	28.6	29.2	28.1	27.4	32.1	33.3	32.5	26.3
ACPT	:	9.9	9.5	7.3	7.0	27.4	8.0	11.1	10.8	26.3
NO.		73	74	75	76					
TIME	:	31.0	32.2	31.4	14.2					
ACPT	:	7.7	10.7	10.5	14.2					

Table 6.27: Case 7: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	29.0	30.7	32.3	34.3	36.0	37.6	32.9	34.6	36.2
ACPT	:	9.7	10.2	10.8	11.4	12.0	12.5	11.0	11.5	12.1
NO.		10	11	12	13	14	15	16	17	18
TIME	:	36.1	37.8	39.4	34.7	36.4	38.0	33.0	34.7	36.3
ACPT	:	12.0	12.6	13.1	11.6	12.1	12.7	11.0	11.6	12.1
NO.		19	20	21	22	23	24	25	26	27
TIME	:	31.6	33.3	34.8	35.9	37.7	39.2	34.5	36.2	34.7
ACPT	:	10.5	11.1	11.6	12.0	12.6	13.1	11.5	12.1	11.6
NO.		28	29	30	31	32	33	34	35	36
TIME	:	35.9	32.4	36.1	31.4	33.1	34.7	35.2	36.9	38.4
ACPT	:	12.0	10.8	12.0	10.5	11.0	11.6	11.7	12.3	12.8
NO.		37	38	39	40	41	42	43	44	45
TIME	:	33.7	35.5	37.0	32.0	33.7	35.3	30.6	32.3	33.9
ACPT	:	11.2	11.8	12.3	10.7	11.2	11.8	10.2	10.8	11.3
NO.		46	47	48	49	50	51	52	53	54
TIME	:	34.2	35.9	37.5	32.8	34.5	36.0	33.2	34.9	36.5
ACPT	:	11.4	12.0	12.5	10.9	11.5	12.0	11.1	11.6	12.2
NO.		55	56	57	58	59	60	61	62	63
TIME	:	31.8	33.5	35.1	33.4	32.0	32.5	31.1	31.0	29.6
ACPT	:	10.6	11.2	11.7	11.1	10.7	10.8	10.4	10.3	9.9
NO.		64	65	66	67	68	69	70	71	72
TIME	:	30.3	28.8	29.4	27.9	27.2	32.5	34.2	33.4	25.8
ACPT	:	10.1	9.6	7.3	7.0	27.2	8.1	11.4	11.1	25.8
NO.		73	74	75	76					
TIME	:	31.1	32.8	32.0	13.9					
ACPT	:	7.8	10.9	10.7	13.9					

Table 6.28: Case 8: Average circuit processing times

NO.		1	2	3	4	5	6	7	8	9
TIME	:	28.5	29.8	31.1	34.2	35.6	36.9	32.8	34.1	35.4
ACPT	:	9.5	9.9	10.4	11.4	11.9	12.3	10.9	11.4	11.8
NO.		10	11	12	13	14	15	16	17	18
TIME	:	36.0	37.4	38.7	34.6	36.0	37.2	32.7	34.1	35.4
ACPT	:	12.0	12.5	12.9	11.5	12.0	12.4	10.9	11.4	11.8
NO.		19	20	21	22	23	24	25	26	27
TIME	:	31.3	32.7	33.9	35.8	37.2	38.5	34.4	35.7	33.7
ACPT	:	10.4	10.9	11.3	11.9	12.4	12.8	11.5	11.9	11.2
NO.		28	29	30	31	32	33	34	35	36
TIME	:	35.8	31.6	35.2	31.1	32.4	33.7	35.0	36.4	37.7
ACPT	:	11.9	10.5	11.7	10.4	10.8	11.2	11.7	12.1	12.6
NO.		37	38	39	40	41	42	43	44	45
TIME	:	33.6	35.0	36.3	31.7	33.1	34.4	30.3	31.7	33.0
ACPT	:	11.2	11.7	12.1	10.6	11.0	11.5	10.1	10.6	11.0
NO.		46	47	48	49	50	51	52	53	54
TIME	:	34.1	35.4	36.7	32.6	34.0	35.3	32.4	33.8	35.1
ACPT	:	11.4	11.8	12.2	10.9	11.3	11.8	10.8	11.3	11.7
NO.		55	56	57	58	59	60	61	62	63
TIME	:	31.0	32.3	33.6	32.8	31.4	31.2	29.8	30.6	29.2
ACPT	:	10.3	10.8	11.2	10.9	10.5	10.4	9.9	10.2	9.7
NO.		64	65	66	67	68	69	70	71	72
TIME	:	29.9	28.4	29.1	27.7	26.9	32.4	33.9	33.2	25.4
ACPT	:	10.0	9.5	7.3	6.9	9.0	8.1	11.3	11.1	8.5
NO.		73	74	75	76					
TIME	:	31.0	32.5	31.7	13.5					
ACPT	:	7.8	10.8	10.6	13.5					

Table 6.29: Case 9: Average circuit processing times

NO.	1	2	3	4	5	6	7	8	9
TIME :	29.8	31.9	30.3	35.6	37.7	36.0	35.0	37.1	35.4
ACPT :	9.9	10.6	10.1	11.9	12.6	12.0	11.7	12.4	11.8
NO.	10	11	12	13	14	15	16	17	18
TIME :	37.8	39.9	38.2	37.2	39.3	37.6	34.7	36.8	35.1
ACPT :	12.6	13.3	12.7	12.4	13.1	12.5	11.6	12.3	11.7
NO.	19	20	21	22	23	24	25	26	27
TIME :	34.1	36.2	34.5	37.6	39.7	38.0	37.0	39.1	34.2
ACPT :	11.4	12.1	11.5	12.5	13.2	12.7	12.3	13.0	11.4
NO.	28	29	30	31	32	33	34	35	36
TIME :	37.6	33.9	34.8	33.8	35.9	34.2	36.6	38.7	37.0
ACPT :	12.5	11.3	11.6	11.3	12.0	11.4	12.2	12.9	12.3
NO.	37	38	39	40	41	42	43	44	45
TIME :	36.0	38.1	36.4	33.5	35.5	33.9	32.8	34.9	33.3
ACPT :	12.0	12.7	12.1	11.2	11.8	11.3	10.9	11.6	11.1
NO.	46	47	48	49	50	51	52	53	54
TIME :	35.4	37.5	35.8	34.8	36.9	35.2	33.1	35.2	33.5
ACPT :	11.8	12.5	11.9	11.6	12.3	11.7	11.0	11.7	11.2
NO.	55	56	57	58	59	60	61	62	63
TIME :	32.5	34.6	32.9	32.0	31.4	29.7	29.1	33.0	32.4
ACPT :	10.8	11.5	11.0	10.7	10.5	9.9	9.7	11.0	10.8
NO.	64	65	66	67	68	69	70	71	72
TIME :	32.0	31.4	31.2	30.6	27.7	34.4	36.2	35.2	27.1
ACPT :	10.7	10.5	7.8	7.7	6.9	8.6	12.1	11.7	6.8
NO.	73	74	75	76					
TIME :	33.8	35.6	34.6	13.5					
ACPT :	8.4	11.9	11.5	4.5					

Table 6.30: Effectivenesses of model C

Cases	Q	reduced μ's	Φ	$\overline{RT}$
1	All Qs = 1	none	0.0152	70.3
2	"	μ's on δ_{11}	0.0227	50.3
3	"	$\mu_{18}, \mu_{31}, \mu_{33}$	0.0244	46.2
4	$Q_{sys}=3$	none	0.0304	41.7
5	$Q_{sbg}=3$	"	0.0294	41.5
6	"	"	0.0308	36.2
7	$Q_{bg}=4$	"	0.0372	38.3
8	$Q_{sbg}=3.$	"	0.0300	30.3
9	$Q_{ibg}=3$	"	0.0752	20.8

7 CONCLUSIONS AND DISCUSSIONS FOR FUTURE RESEARCH

7.1 Conclusions

In this thesis, a version of Stochastic Timed Place Petri Nets (STPPNs) capable of modeling a typical C^2 system, which operates asynchronously and concurrently, was developed. Also, in order to construct the STPPN having a well-formed structure, ill-formed situations of were identified and ways to resolve the situations naturally were provided by making operation rules with non-restrictive assumptions. In addition, sufficient conditions for an STPPN to have a statistically meaningful long term behavior was analyzed. Modeling was performed for peace-time and war-time. The latter model changes its structure in accordance with the result of the engagement against the opposite forces.

Besides the modeling, a basic methodology to simulate and analyze underlying models was provided. By decomposing the STPPN to a circuit-free net, the representative model state of the underlying STPPN was established, which is useful for finding the exact process execution order. This made it easier to formulate the relationships between processes belonging to the different slices. Also, the execution order was used to show the verification of the simulation program.

The modeling addressed two types of constraints: time and structure. The time constraints are derived from the task processing times, whereas the structural constraints characterizes the capacity of a C^2 system or the system itself. The capacity limitation corresponds to the limit of the human short-term memory in the case of the one-man C^2 process and the maximum number of people capable of being involved in a C^2 process. For the overall system, the capacity limitation was derived from the limited capacity of the system to store the information concurrently processed, which is determined by the model structure. In both cases, the capacity limitation bounds the amount of information that can be handled at the same time. Also, in a C^2 process, the capacity indicates the number of people assigned in order for them to perform the process.

The maximum throughput rate is expressed as a function of the structural and time constraints in the following manner. The inclusion of the capacity in modeling the C^2 system results in simple directed circuits characterized by the circuit processing time, $\alpha(\cdot)$. $\alpha(\cdot)$ represents the average amount of time it takes for one input to complete the tasks of the circuit. The amount of capacity available, n, which bounds the total number of inputs can be processed concurrently in the circuit. Also, it indicates the total number of people involved in the circuit. For a given circuit, the ratio $n/\alpha(\cdot)$ characterizes the average circuit processing rate . As shown in Chapter 5, the minimum average circuit processing rate determines the maximum throughput rate of the system. The determination of the critical circuits, for which the corresponding average processing rate is precisely minimal, is very useful for evaluating and comparing systems with different structures. In fact, the critical circuits characterize the particular time and structural constraints that actually bound the throughput rate. Therefore, the way with which the different constraints affect the effectiveness of the model can be completely specified. In particular, the problem of modifying the right constraints to improve the effectiveness becomes clear.

Simulation was performed for two C^2 models. One is the normal model (model N), which is designed with the C^2 system under the assumption that the system operates in peace time. The other is the model with structures changed (model C) according to the results of engagements against the opposite force. The processing times assigned to the task of each process were assumed to be exponentially distributed and the other inputs were assigned deterministically. For the main results of the simulation, the procedures to determine the most effective model structure was established.

7.2 Discussions for Future Research

7.2.1 Model design

STPPN models dealt with in this thesis are Conflict-free models that a place in the net has only one input and one output transition. Also, the token quantity that each place can possess at a time is limited to one. This type of model has several advantages and disadvantages. Since the Conflict-free model has no distinction between tokens passed a place during its operations, i.e., this model operates under the assumption that all arriving information is identical, model states can be established very simply. Therefore, this type of model can be easily analyzed and no special operation rules are needed, which means operation rules are no different than the original Petri net. However, if the arriving information handled in a place is not identical, the tokens should be treated differently every time they enter the place since the processing time of each place will have different processing time distributions. This represents the system more realistically than the previously one used in this thesis. However, the unique algorithm to assign the specific processing time distribution for the specific token to each place should be prepared.

Meanwhile, since the model designed in this thesis has only one token at a time, the number of states required in order to represent the model is relatively small, compared the model which has more than one token in a place at a time. Also, its reachability can be easily proven by the structural analysis. The latter model, called Colored Petri Net (CPN), has been researched by Diaz [17]. The CPN is based on the introduction of colored tokens which represent different information or different tasks in a place. Therefore, this formalism can express the system more realistically, but several problems remain to be solved. For example, the restriction for the maximum number of tokens capable of staying simultaneously in a place, the need of the transition firing algorithm in order to release a specific token from a place, and the difficulty of the reachability proof due to increasing the complexity of the system states should be solved.

7.2.2 Model expansion

As mentioned before, time is the most crucial factor of a C^2 system. Especially, if a battle system is in contact with its counterforce, the time factor affects directly on the system's survivability. Therefore, if the system is in the battle mode, emergency situations such as no time to communicate by the normal route may happen during the battle. If the normal route is applied in such a case, then due to delayed reactions, the system may not perform its task successfully and may be destroyed more rapidly by the counterforce. Therefore, the consideration of the emergency communication route which can reduce the system's reaction time for the enemy's threat is necessary. For example, if an emergency communication route is connected between IBGs and CMD, all information are reported from IBGs to the commander directly without passing through the intermediate reporting channel CIC. After the CMD selects its response for the information, its order is directly sent to the IBGs, thus, the system's reaction can be performed more rapidly, as compared to the system with only normal communication routes. Therefore, the STPPN designed in this thesis can be easily extended by placing emergency communication routes to the original net as shown in Figure 7.1. The problem of how many routes are placed is strongly related to the cost required by adding extra routes and depends upon the tactics taken as the course of action when the system operates.

7.2.3 Model applications

The STPPN model designed in this thesis can be used in many areas, not only in combat organizations, but also civil organizations such as manufacturing systems, business organizations, if they can be represented as a hierarchical structure. Also, the model can be used for designing a new system by comparing changes of the system's effectiveness, according to the changes of the component relations such as the communication route, as well as finding out the specific system component which causes the delay of the system operation. An another way to apply this model is to determine the trade-off between the effectiveness level and the cost.

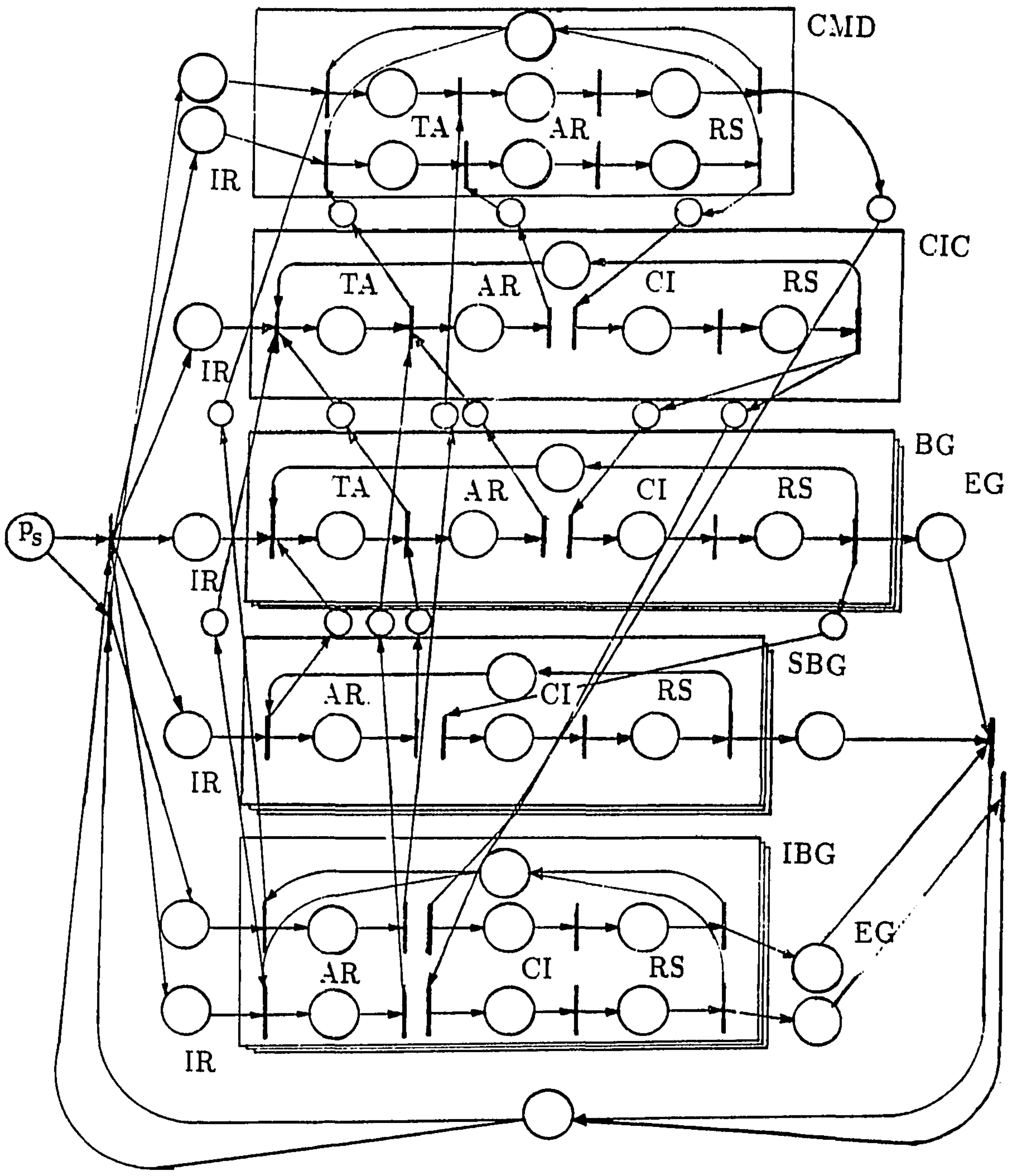


Figure 7.1: Example of C^2 model with emergency communication routes

8 BIBLIOGRAPHY

[1] Agerwala, T. "Putting Petri Nets to Work." Computer Networks, 12(12) (1979): 85-94.

[2] Alaiwan, H. and Toudic, J. M. "Recherche des Semi-Flots, des Verrous et des Trappes dan les Reseaux de Petri." Technique et Science Informatiques, 4(1) (1985): 103-112.

[3] Andre, C., Diaz, M., Girault C., and Sifakis, J. "Survey of French Research and Applications based on Petri Nets." Lecture Notes in Computer Science, Vol. 84, pp. 321-345. New York: Springer-Verlag, 1980.

[4] Athans, M. "The Expert Team of Experts Approach to Command-and-control (C^2) Organizations." I.E.E.E. Control Systems Magazines, 2(3) (1982): 30-38.

[5] Bejjani, G. J. "Information Storage and Access in Decision-making Organizations." M.S. thesis, M.I.T., Cambridge, MA, 1985.

[6] Boettcher, K. L. "Information Structures in Decision-making Organizations." M.S. thesis, M.I.T., Cambridge, MA, 1983.

[7] Bohannan, A. G. "C^3I in Support of the Land Commander." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 1-8. Bournmouth, Dorset: I.E.E., 1985.

[8] Bruno, G. and Biglia, P. "Performance Evaluation and Validation of Tool Handling in Flexible Manufacturing Systems Using Petri Nets." International Workshop on Timed Petri Nets, pp. 64-71. Torino, Italy: I.E.E.E. Computer Society Press, 1985.

[9] Cao, X. and Ho, Y. C. "Perturbation Analysis and Optimization of Queueing Networks." International Large Scale Systems Symposium, pp. 25-26. Virginia Beach, VA: I.E.E.E. Inc., 1982.

[10] Carrington, J. H. "Command Control Compromise: Values and Objectives for the Military Manpower." Annapolis, MD: Naval Institute Press, 1972.

[11] Castelaz, P., Angus, J., and Mahoney, J. "Application of Neural Networks to Expert Systems and Command and Control Systems." Proceedings of Western Conference on Expert Systems, pp. 118-125. Anaheim, CA: Computer Society Press, 1987.

[12] Chyen, G. H. "Information Theoretic Models of Preprocessors and Decision Aids." M.S. thesis, M.I.T., Cambridge, MA, 1981.

[13] Cohen, G., Moller, P., Quadrat, J. P., and Viot, M. "Linear System Theory for Discrete Event Systems." Proceedings of 23rd I.E.E.E. Conference on Decision and Control, Vol. 1, pp. 539-544. Las Vegas, NV: I.E.E.E. Inc., 1984.

[14] Coolahan, J. E., Jr. and Roussopoulos, N. "Timed Petri Net Methodology for Specifying Real-Time System Timing Requirements." International Workshop on Timed Petri Nets, pp. 24-31. Torino, Italy: I.E.E.E. Computer Society Press, 1985.

[15] Corsi, F. and Castagnolo, B. "Probabilistic Delay Evaluation in Combinational Digital Circuits by Petri Nets." Microelectronics and Reliability, 23(3) (1982): 541-553.

[16] Dallery, Y. and David, R. "A New Approach based on Operational Analysis for FMS Performance Evaluation." Proceedings of the 22nd I.E.E.E. Conference on Decision Control, Vol. 3, pp. 1056-1061. Orlando, FL: I.E.E.E. Inc., 1983.

[17] Diaz, M. "Modeling and Analysis of Communication and Cooperation Protocols using Petri Net Based Model." Computer Networks, 6(6) (1982): 419-441.

[18] Dubois, D. and Stecke, K. E. "Using P.N.'s to represent Production Process." Proceedings of the 22nd I.E.E.E. Conference on Decision Control, Vol. 3, pp. 1062-1067. Orlando, FL: I.E.E.E. Inc., 1983.

[19] Findeisen, W., Bailey, F. N., Bradys, N. Malinowski, K., Tatjewski, P., and Wozniak, A. "Control and Coordination in Hierarchical Systems." New York: John Wiley & Sons, 1980.

[20] Frolin, G. and Natkin, S. "Evaluation based upon Stochastic Petri Nets of the Maximum Throughput of A Full Duplex Protocol." 1st and 2nd European Workshop on Application and Theory of Petri Nets, eds. C. Girault and W. Reisig, Vol. 52, pp. 280-288. Rennes, France: I.E.E.E. Computer Society Press, 1982.

[21] Galbraith, J. R. "Designing Complex Organizations." Reading, MA: Addison-Wesley Pub. Co., 1973.

[22] Galley, D. G. "The C-Process: A Model of Command." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 70-79. Bournmouth, Dorset: I.E.E., 1985.

[23] Genrich, H. J., Lautenbach, K., and Thiagarajan, P. S. "Elements of General Net Theory." Lecture Notes in Computer Science, Vol.84, pp. 21-164. New York: Springer-Verlag, 1980.

[24] Genrich, H. J. and Stankiewicz-Wiechno, E. "A Dictionary of Some Basic Notions of Net Theory." Lecture notes in computer science, Vol.84, pp. 519-531. New York: Springer-Verlag, 1980.

[25] Georgiou, A. S. and Lammers, G. H. "C^3 Effectiveness Studies." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 80-85. Bournmouth, Dorset: I.E.E., 1985.

[26] Gundry, A. J. "The Application of Human Factors in C^3I System Development." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 40-42. Bournmouth, Dorset: I.E.E., 1985.

[27] Hack, M. H. T. "Analysis of Production Schemata by Petri Nets." Project MAC-TR-94, M.I.T., Cambridge, MA, 1972.

[28] Hall, S. A. "Information Theoretic Models of Storage and Memory." M.S. thesis, M.I.T., Cambridge, MA, 1982.

[29] Han, Y. C. "Performance Evaluation of a Digital System using a Petri Net-like Approach." Proceedings of the National Electronics Conference, pp. 166-172, Chicago, IL: National Engineering Consortium Inc., 1978.

[30] Hays, S. H. and Thomas, W. N. "Taking Command." Harrisburg, PA: Stackpole Books, 1967.

[31] Hill, J. S. and Richard, F. A. "Standards for Naval Combat Systems." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 182-185. Bournmouth, Dorset: I.E.E., 1985.

[32] Hillion, H. P. "Performance Evaluation of Decision Making Organizations using Timed Petri Nets." M.S. thesis, M.I.T., Cambridge, MA, 1983.

[33] Hitchins, D. K. "Mosaic Concepts for the Future Development of Air Power in European NATO." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 18-28. Bournmouth, Dorset: I.E.E., 1985.

[34] Hitchins, D. K. "The Human Element in C^3I." Advanced in C^3 System: Theory and Applications. International Conference, No. 247, pp. 29-39. Bournmouth, Dorset: I.E.E., 1985.

[35] Ho, Y. C. and Cassandras, C. "A New Approach to the Analysis of Discrete Event Dynamic Systems." Automatica, 19(12) (1983): 149-167.

[36] Iowa State Daily. "Vincennes was warned about Iranian Airliner." Iowa State Daily, September 9, 1988, p. 4.

[37] Ho, Y. C., Cao, X., and Cassandras, C. "Infinitesimal and Finite Perturbation Analysis for Queueing Networks." Automatica, 19(4) (1983): 439-445.

[38] Jantzen, M. and Valk, R. "Formal Properties of Place/Transition Nets." Lecture Notes in Computer Science, Vol. 84, pp. 165-212. New York: Springer-Verlag, 1982.

[39] Kyle, D. N. "A Methodology for Air Command and Control System(ACCS) Design." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 86-89. Bournmouth, Dorset: I.E.E., 1985.

[40] Lawson, J. S. Jr. "What is Command Control?" System issues in C^3 Problems, Vol. 1, Ed. Athans M. ESL-R-844. Laboratory for Information and Decision Systems, M.I.T., Cambridge, MA, 1978.

[41] Levis, A. H. "Information Processing and Decision-making Organization." Large Scale Systems, 7 (1984): 86-89.

[42] Magott, J. "New NP-Complete Problems in Performance Evaluation of Concurrent Systems Using Petri Nets." Transactions on Software Engineering, SE-13(5) (1987): 587-581.

[43] Marson, M. A., Balbo, G., and Conte, G. "A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor System." ACM Transactions on Command Systems, 2(2) (1984): 93-122.

[44] Martinez, J. and Silva, M. "A Simple and Fast Algorithm to obtain All Invariants of a Generalized Petri Net." 1st and 2nd European Workshop on Application and Theory of Petri Nets, eds. C. Girault and W. Reisig, pp. 301-310. Rennes, France: I.E.E.E. Computer Society Press, 1982.

[45] Memmi, G. "Notion de Dualite et de Symmetrie dans les Reseaux de Petri." Lecture Notes in Computer Science, No. 70, pp. 91-108. New York: Springer-Verlag, 1979.

[46] Memmi, G. and Roucairol, G. "Net Theory and Applications." Lecture Notes in Computer Science, Vol. 84, pp. 213-223. New York: Springer-Verlag, 1980.

[47] Merlin, P. M. and Farber, D. J. "Recoverability of Communication Protocols– Implication of a Theoretical Study." I.E.E.E. Transactions on Communication, COM-24(9) (1976): 1036-1045.

[48] Middleton, S. "Air Defense Threat Assessment." Advances in Command, Control, and Communication Systems, eds. C. J. Harris, I. White, pp. 378-397. London, England: Peter Peregrinus Ltd., 1987.

[49] Molloy, M. K. "Performance Analysis using Stochastic Petri Nets." I.E.E.E. Transactions on Computers, C-31(9) (1982): 913-917.

[50] Morgan, P. D. "Modeling the Decision Maker within a Command System." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 65-69. Bournmouth, Dorset: I.E.E., 1985.

[51] Munro, D. Warren, C. S., and Hunt, J. D. "Dynamic Management of Military Communication System." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 155-163. Bournmouth, Dorset: I.E.E., 1985.

[52] Murata, T. "Synthesis of Decision-Free Concurrent Systems for Prescribed Resources and Performance." I.E.E.E. Transactions on Software Engineering, SE-6(6) (1980): 525-530.

[53] Noe, J. D. "Application of Net-Based Models." Lecture Notes in Computer Science, Vol. 84, pp. 342-345. New York: Springer-Verlag, 1980.

[54] Noe, J. D. "Nets in Modeling and Simulation." Lecture Notes in Computer Science, Vol. 84, pp. 347-363. New York: Springer-Verlag, 1980.

[55] Peterson, J. L. "Net Theory and the Modeling of Systems." Englewood Cliffs, NJ: Prentice-Hall Inc.,1981.

[56] Ramamoothy, C. V. and Ho, G. S. "Performance Evaluation of Asynchronous Concurrent Systems using Petri Nets." I.E.E.E. Transactions on Software Engineering, SE-6(5) (1980): 440-449.

[57] Ramchandani, C. "Analysis of Asynchronous Concurrent Systems by Petri Nets, Ph.D. diss., M.I.T., Cambridge, MA, 1974.

[58] Razouk, R. R. and Phelps, C. V. "Performance Analysis using Timed Petri Nets." Protocol Specification, Testing, and Verification, eds. Y. Yemini, R. Storm and S. Yemini, pp. 561-576. North-Holland: Elsevier Science Publishers, 1982.

[59] Reisig, W. "Petri Nets: An Introduction." New York: Springer-Verlag, 1982.

[60] Shellard, D. J. "C^3 within a Naval Ship." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 186-191. Bournmouth, Dorset: I.E.E., 1985.

[61] Sifakis, J. "Performance Evaluation of Systems using Nets." Lecture Notes in Computer Science, Vol. 84, pp. 307-319. New York: Springer-Verlag, 1980.

[62] Sifakis, J. "Use of Petri Nets for Performance Evaluation : Measuring, Modeling, and Evaluating Computer Systems." eds. H. Beimer and E. Gelenbe, pp. 75-93. North-Holland: Elsevier Science Publishers, 1982.

[63] Smigelski, T., Murata, T., and Sowa, M. "A Timed Petri Net Model and Simulation of a Data Flow Computation." International Workshop on Timed Petri Nets, pp. 56-63. Torino, Italy: I.E.E.E. Computer Society Press, 1985.

[64] Stabile, D. A., Levis, A. H. "Information Structures of Single Echelon Organization." M.S. thesis, M.I.T., Cambridge, MA, 1982.

[65] Stotts, P. D., Jr. and Pratt, T. W. "Hierarchical Modeling of Software Systems with Timed Petri Nets." International Workshop on Timed Petri Nets, pp. 32-39. Torino, Italy: I.E.E.E. Computer Society Press, 1985.

[66] Suri, R. "Infinitesimal Perturbation Analysis of D.E.D.S.: A General Theory." Proceedings of the 22nd I.E.E.E. Conference on Decision and Control, Vol. 3, pp. 1030-1038. Orlando, FL: I.E.E.E. Inc., 1983.

[67] Tabak, D. and Levis, A. H. "Petri Net Representation of Decision Models." I.E.E.E. Transactions on Systems, Man and Cybernetics, SMC-15(6) (1985): 812-818.

[68] Tainsh, M. A. "Decision-aiding in Command System." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 186-191. Bournmouth, Dorset: I.E.E., 1985.

[69] Thomasian, A. and Bay, P. "Performance Analysis of Task Systems Using a Queueing Network Model." International Workshop on Timed Petri Nets, pp. 234-241. Torino, Italy: I.E.E.E. Computer Society Press, 1985.

[70] Tong, C. "Goal-directed Planning of the Design Process." I.E.E.E. 3rd Conference on AI applications, Vol. 247, pp. 284-289. Washington D.C.: I.E.E.E. Computer Society Press, 1987.

[71] Ulug, M. E. "A Realtime System for Military Communications." I.E.E.E. 3rd Conference on AI applications, Vol. 247, pp. 250-255. Washington D.C.: I.E.E.E. Computer Society Press, 1987.

[72] Walter, L. H. "A Petri Net Approach to System Level Fault Tolerance Analysis." Proceedings of the National Electronics Conference, Vol. 32, pp. 161-165. Chicago, IL: National Engineering Consortium Inc., 1978.

[73] White, I. "The Future for Command Systems." Advances in Command, Control, and Communication Systems, eds. C.J. Harris and I. White, I.E.E.E. Computing Series, pp. 1-15. London: Peter Pereginus Ltd., 1987.

[74] Whol, J. G. "Force Management Decision Requirements for Air Force Tactical Command and Control." I.E.E.E. Transactions on Systems, Man, and Cybernetics, SMC-11(9) (1981): pp. 618-639.

[75] Wiley, R. P. "Performance Analysis of Stochastic Timed Petri Nets." Ph.D. diss., M.I.T., Cambridge, MA, 1985.

[76] Wilson, G. B. "Some Aspects of Data Fusion." Advanced in C^3 Systems: Theory and Applications. International Conference, No. 247, pp. 99-105. Bournmouth, Dorset: I.E.E., 1985.

[77] Wong, C. Y., Dillon, T. S., and Forward, K. E. "Timed Placed Petri Nets with Stochastic Representation of Place Time." International Workshop on Timed Petri Nets, pp. 96-103. Turin, Italy: I.E.E.E. Computer Society Press, 1985.

[78] Zuberek, W. M. "Timed Petri Nets and Preliminary Performance Evaluation." Computer Architecture News, 8(3) (1980): 88-96.

9 APPENDIX A: ALGORITHM TO OBTAIN ALL SIMPLE DIRECT CIRCUITS OF A PETRI NET

This appendix presents the algorithm that has been used to obtain all simple direct circuits of a Petri net. As discussed in Chapters 4 and 5, it is necessary to determine all circuits of the net, so as not only to transform it into a corresponding unfolded Petri net, but also to compute measures of effectiveness of the net such as the average cycle time of processes. The algorithm described here was developed by Alaiwan and Toudic [2] and was proven by Martinez and Silva [44]. This algorithm determines all the minimal support S-invariants of the net, which correspond to all its simple direct circuits.

Next, Alaiwan and Toudic's algorithm is described and illustrated how it works through a simple example.

9.1 Alaiwan and Toudic's Algorithm to Obtain all the Minimal Support S-invariants of a Petri Net

In the following description, C is denotes the incidence matrix of the Petri net, with dimensions nxm, where n is the number of places and m the number of transitions. C_{ij} denotes, the element corresponding to the ith row and the jth column of C, and I_n denotes the $n \text{x} n$ identity matrix.

9.1.1 Algorithm for finding simple directed circuits

Step 1: $A = C$; $D = I_n$. (Initialization)

Step 2: Repeat for $j = m$. (Main Loop)

Step 2.1: Determine the sets S_1 and S_2 such that

$$S_1 = \{i_1 \mid A_{i_1 j} > 0\} \text{ and } S_2 = \{i_2 \mid A_{i_2 j} < 0\} \ .$$

Step 2.2: For all pairs $(i_1, i_2)^1 \in S_1 \mathrm{x} S2$,

Step 2.2.1: Append to the matrix A the row vector:

$$A_{i_1 j} * (i_2\text{th row of } A) - A_{i_2 j} * (i_1\text{th row of } A) \ .$$

Step 2.2.2: Append to the matrix D the row vector:

$$A_{i_1 j} * (i_2\text{th row of } D) - A_{i_2 j} * (i_1\text{th row of } D) \ .$$

Step 2.3: Eliminate from A and D all the rows with index $i \subset S_1 \cup S_2$.

Step 2.4: Eliminate from D all the rows whose supports are non-minimal with respect to the other rows of D. If two rows have the same support, eliminate one of them.

Step 2.5: Eliminate from A the rows corresponding to the rows eliminated from D.

Step 3: The rows of D determine all the minimal supports of S-invariants in the net.

[1]In the algorithm, the set of elements, $i \in \{1, 2, \cdots, n\}$, denotes the support of a n-positive integer vector, $\vec{v}$ such that the ith component of v is non-null. If $G = \{\vec{v_1}, \vec{v_2}, \cdots, \vec{v_k}\}$ is a set of vectors, $\vec{v_i}$, $i \in \{1, 2, \cdots, k\}$ will be a vector whose support is minimal in G, if and only if there does not exist in G a non-null vector, $\vec{v_j}$, such that its support is strictly included in the support of v_i.

Let's make a brief comment on how the algorithm works. The underlying idea is to proceed in m steps by finding the minimal support S-invariants of the R^i net, $i = 0, 1, \cdots, m-1$, which results from the elimination of the transitions $(i+1, \cdots, m)$. The matrix D is constructed so that at the ith iteration the row vectors of D are precisely the minimal support S-invariants of the net R^i. Initially $D = I_n$, because R^0 corresponds to the net without transitions and therefore, each single place constitutes a minimal support invariant. Once the minimal support S-invariants of the net R^i are obtained, those of the net R^{i+1} are generated in the following manner: they are obviously invariants for the net R^i. They can be expressed as a positive linear combination of the minimal support S-invariants of R^i, i.e., as the row of D. However, the invariants generated are not all minimal support S-invariants and it is necessary to eliminate the ones that are by comparing their supports.

9.2 Determination of Simple Direct Circuits for a Net

It has been proved that the simple direct circuits of a net are minimal S-components of the net. Note that an S-component is a subnet constructed as follows:

- (1) the set of places P_S is the support of the corresponding minimal S-invariant.

- (2) the set of transitions T_S is all the transitions of the Petri net connected to the places of P_S, i.e.,

$$T_S = \cup_{p \subset P_S} (p^{\cdot} \cup {}^{\cdot}p) \ .$$

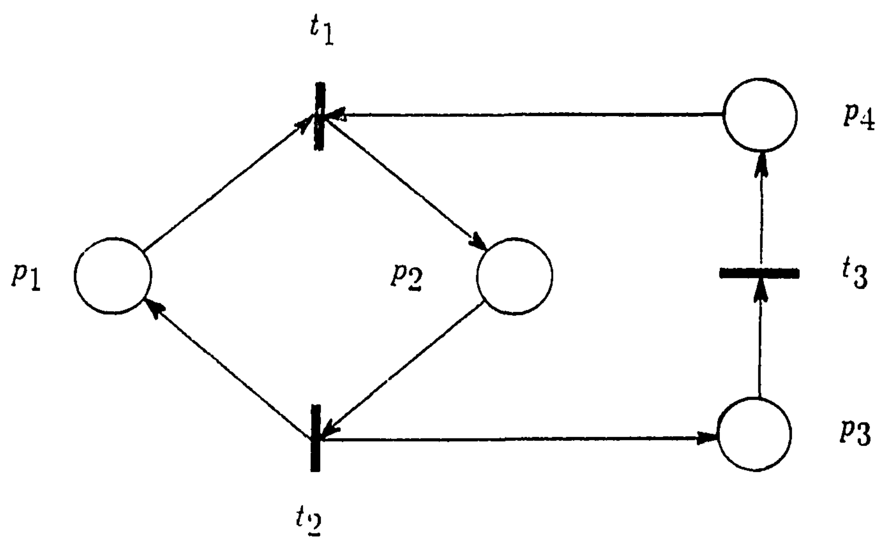


Figure 9.1: An example of simple petri net

Once all the minimal support S-invariants of the net is obtained. using the algorithm of Alawain and Toudic [2], the determination of the circuits becomes straightforward, i.e., for each minimal support, the unique output(or input) transition of each place of the support is determined and then the corresponding S-component, which is a circuit, is obtained immediately.

9.3 Example

Let introduce a simple example to show how the algorithm works (see Figure 9.1). From Figure 9.1, the incident matrix, C, can be obtained as follows:

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

Let carry out the steps of the algorithm described previously.

Step 1: Initialize matrices A and D.

$$A = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} . \quad D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Step 2: For $j = 1$,

Step 2.1: $S_1 = \{2\}$, $S_2 = \{1, 4\}$.

Therefore, append row 1 to row 2 and row 4 to row 2 and make it the last rows of the corresponding matrices, A and D.

Step 2.2: Then, eliminate row 1, row 2, and row 4 in A and D. The resulting matrices are:

$$A = \begin{bmatrix} 0 & 1 & -1 \\ 0 & 0 & 0 \\ 0 & -1 & 1 \end{bmatrix} . \quad D = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} .$$

Step 2.3: Since the rows of D are obviously minimal support, no elimination occurs at Steps 2.4 and 2.5.

For $j = 2$,

Step 2.1: $S_1 = \{1\}$, $S_2 = \{2, 3\}$.

Step 2.2 and 2.3: Append row 1 to row 3 and make it the last rows of the corresponding matrices A and D. Then, eliminate rows 1 and 3. Therefore, the resulting matrices are:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} . D = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} .$$

Once again, the two rows of D are minimal supports. Now the last iteration, corresponding to $j = 3$ is useless since the third column of A is already null. The two rows of D are therefore the minimal support S-invariants of the underlying net, which correspond to the set of places: $\{p_1, p_2\}$ and $\{p_2, p_3, p_4\}$. The two simple direct circuits are immediately obtained:

$$\text{Circuit } 1 = t_1 p_2 t_2 p_1 t_1$$

$$\text{Circuit } 2 = t_1 p_2 t_2 p_3 t_3 p_4 t_1 \quad .$$

As have been seen above, the algorithm works for the simple net. However, if the net contains a self-loop or non-trivial conflict set, the algorithm does not work: in the former case, a self-loop can not be presented in the incident matrix without changing its form and in the latter case, the common input place of the conflict set contains several ways to go, i.e., the place creates several supports which equal the number of transitions in conflict. However, the algorithm only creates one support. Therefore, if the net contains self-loops or non-trivial conflict sets, transform the net by adding dummy places and transitions with the extension rules [23] and then construct the incident matrix. Then, the algorithm will work nicely.

10 APPENDIX B: CHANGED STRUCTURES OF MODEL C

The following figures represent the changes, according to the destruction of the combat units(C^2 processes), of STPPN models investigated with in this thesis.

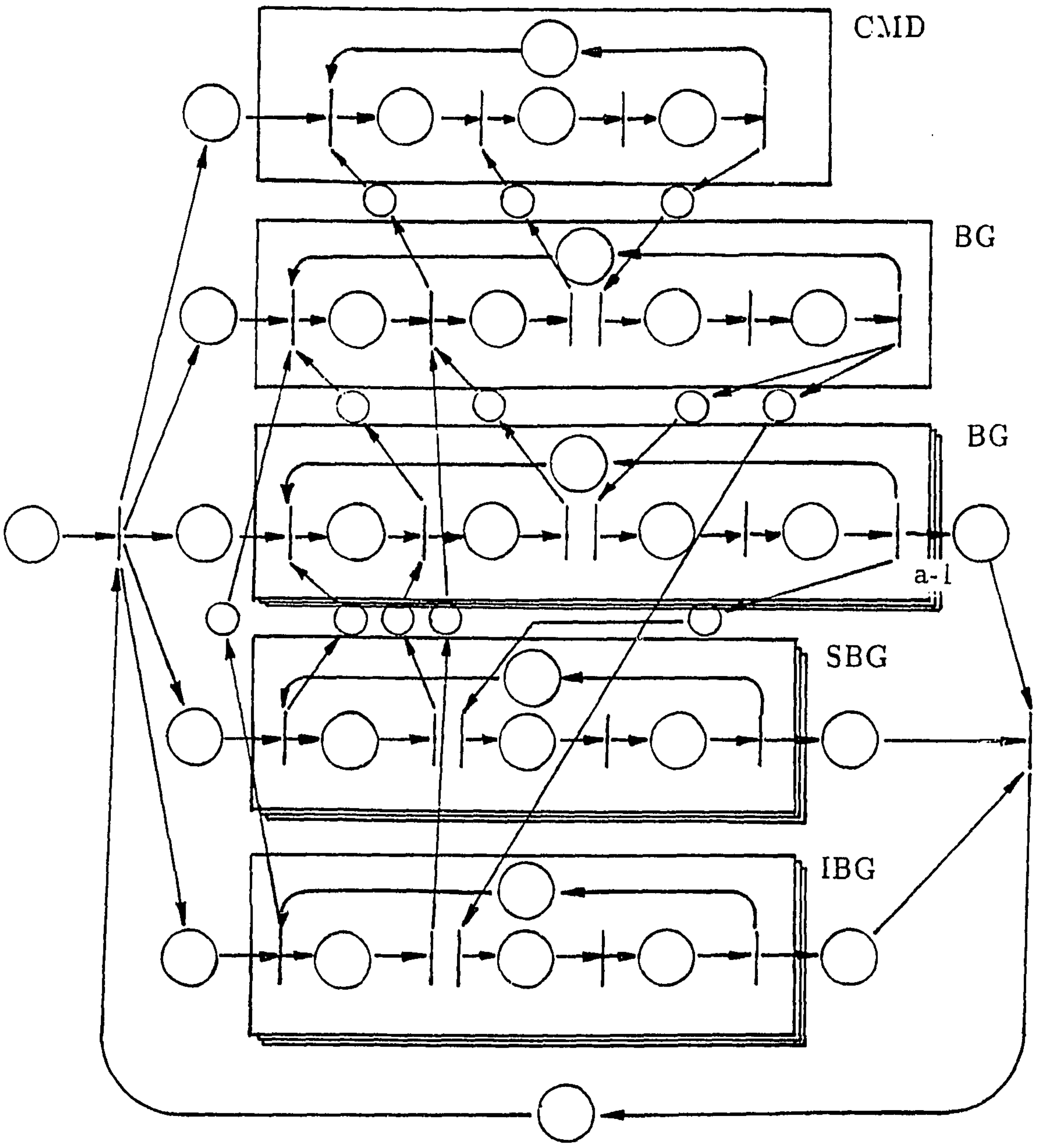


Figure 10.1: Changed structure of model C due to destruction of CIC

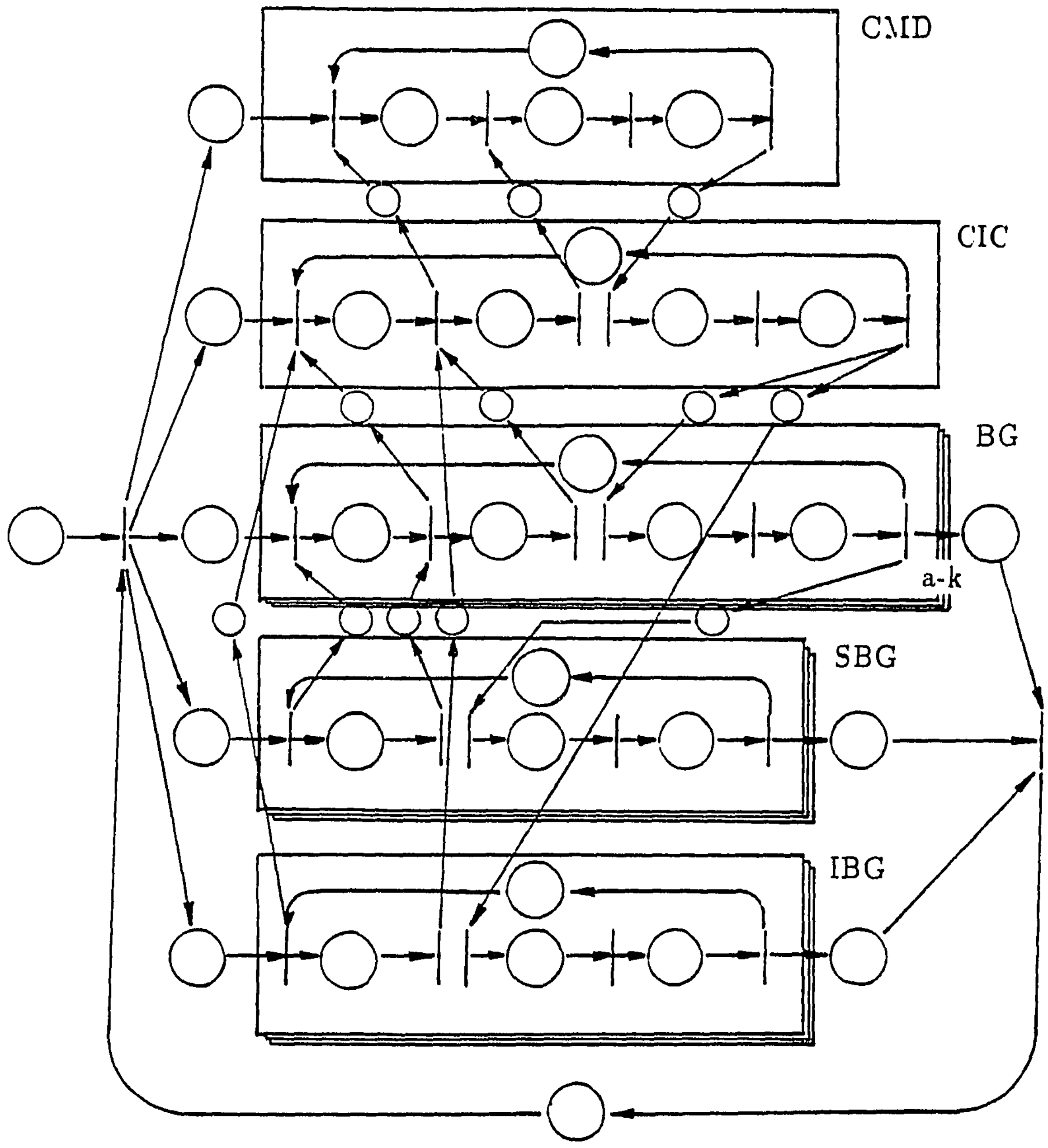


Figure 10.2: Changed structure of model C due to destruction of k BGs ($1 \leq k < a$)

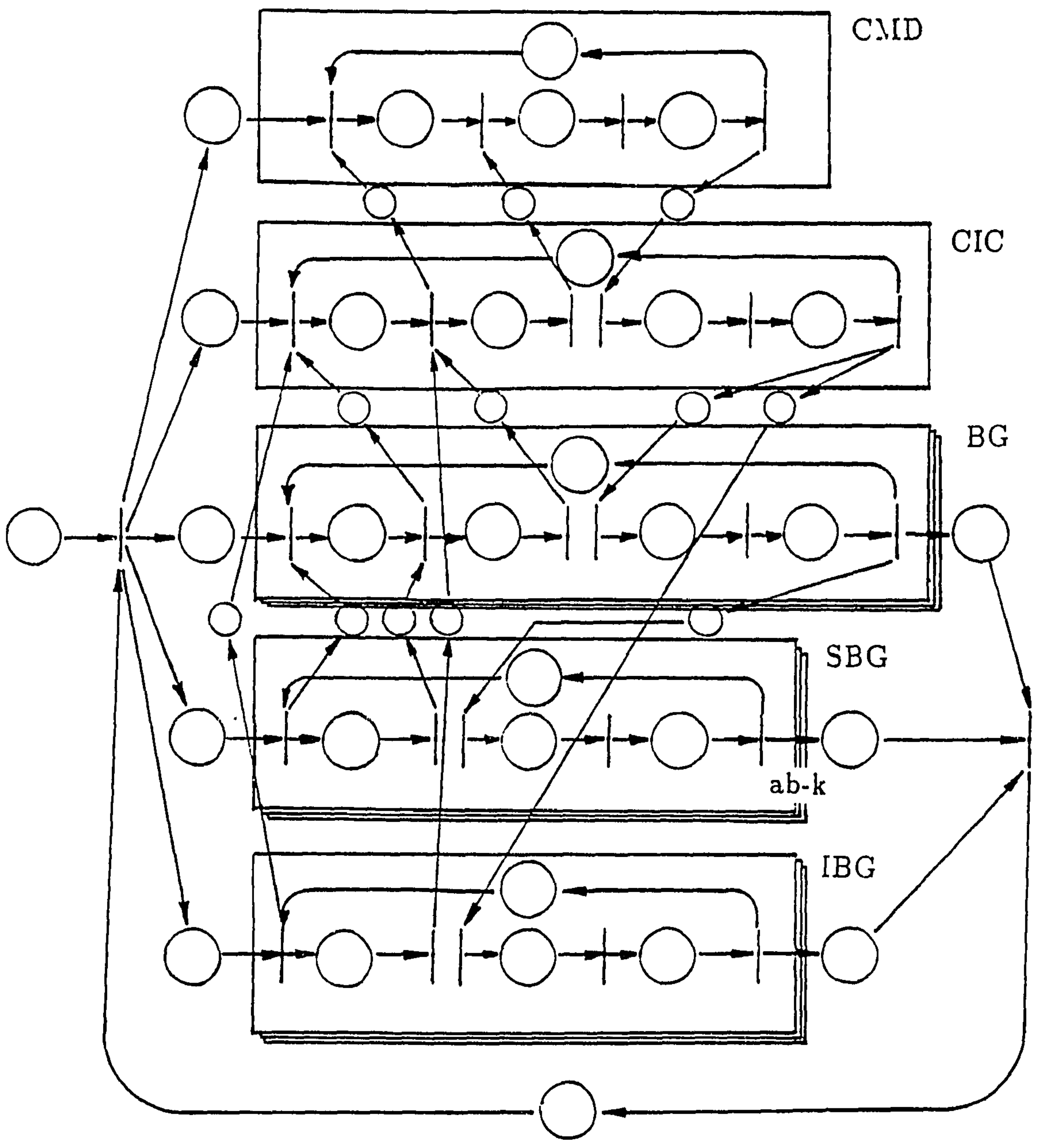


Figure 10.3: Changed structure of model C due to destruction of k SBGs ($1 \leq k < ab$)

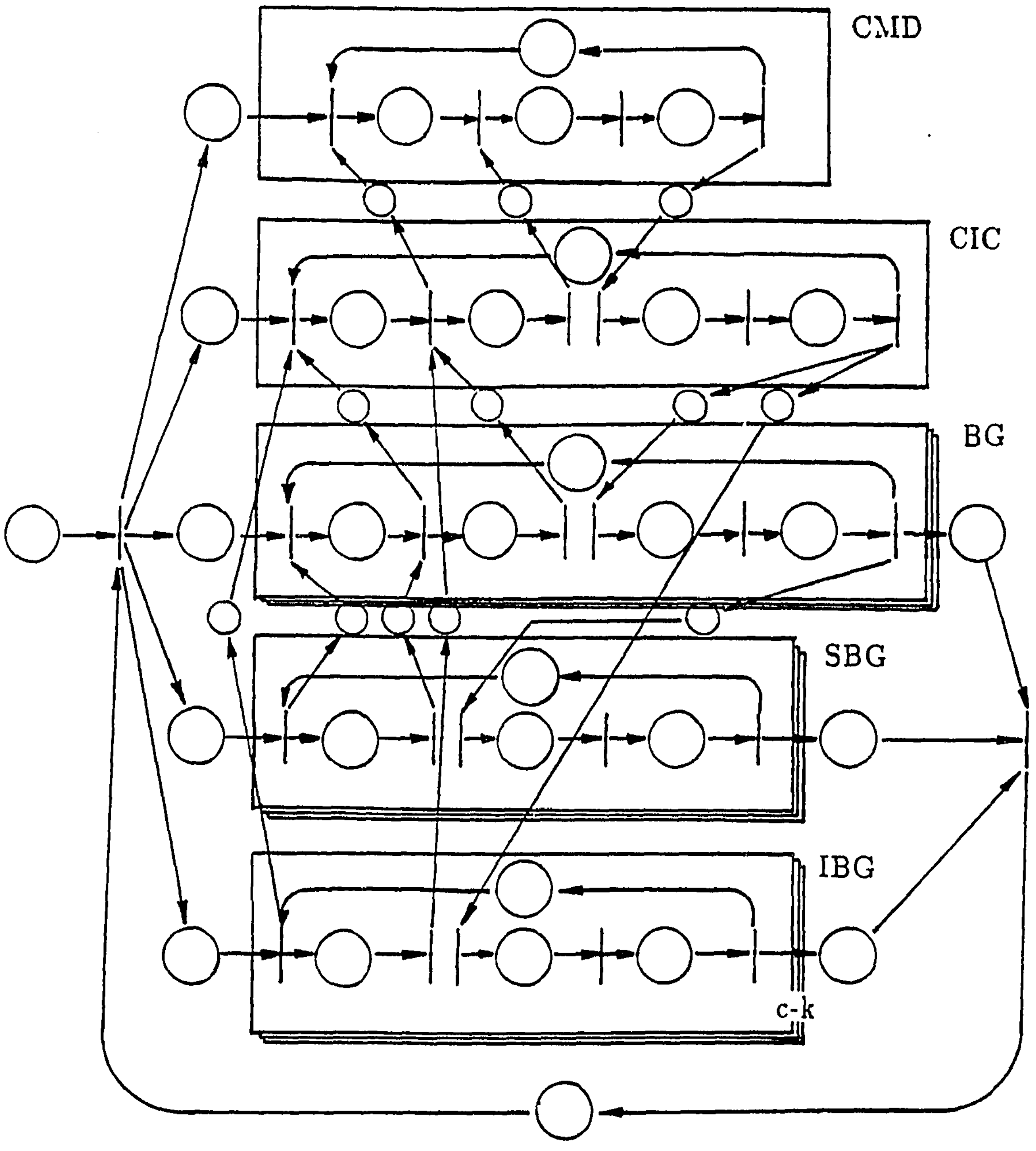


Figure 10.4: Changed structure of model C due to destruction of k IBGs ($1 \leq k < c$)