ON THE GENERATION OF VARIABLE STRUCTURE DISTRIBUTED ARCHITECTURES

by

JACQUES J. DEMAËL

Ingénieur de l'Ecole Polytechnique
(1987)

SUBMITTED TO THE DEPARTMENT OF
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE
IN TECHNOLOGY AND POLICY
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
May 1989

Signature of Author ____
Department of Electrical Engineering and Computer Science
May 15, 1989

Certified by ____
Dr. Alexander H. Levis
Thesis Supervisor

Accepted by ____
Professor Richard de Neufville, Chairman
Technology and Policy

Accepted by ____
Professor Arthur C. Smith, Chairman
Departmental Graduate Committee
Electrical Engineering and Computer Science Department

ON THE GENERATION OF VARIABLE STRUCTURE DISTRIBUTED ARCHITECTURES

by

JACQUES J. DEMAËL

Submitted to the Department of Electrical Engineering and Computer Science
on May 15, 1989
in partial fulfillment of the requirements for the degree of
Master of Science in Technology and Policy

Thesis Supervisor : Dr. Alexander H. Levis
Title : Senior Research Scientist

ON THE GENERATION OF VARIABLE STRUCTURE DISTRIBUTED ARCHITECTURES

by

JACQUES J. DEMAËL

Submitted to the Department of Electrical Engineering and Computer Science
on May 15, 1989
in partial fulfillment of the requirements for the degree of
Master of Science in Technology and Policy

ABSTRACT

A methodology to model and generate variable structure distributed intelligence systems is presented. First, the objects and the functional entities that belong to the system are defined, as well as generic interactions between them. A mathematical framework is developed to represent these interactions. This framework is embedded in Colored Petri Net theory, which is used as the basic technique to generate variable structures. The set of variable structures that satisfy both general constraints and user-defined requirements is analyzed with results from Lattice theory. A class of solutions to the design problem is characterized. This class corresponds to the variable structures whose variability corresponds exactly to the requirements of the user. This class of solution is decomposed into subsets of structures with the same input links. Each subset is delimited by minimal and maximal elements. There is a layer of partially ordered subsets between one minimal element and one maximal element. Each layer is a lattice. The methodology is applied to two illustrative examples, the coordination of tasks in a submarine and the coordination of tasks among air traffic controllers at an airport. Finally, policy considerations of the research on distributed intelligence systems are presented.

Thesis Supervisor : Dr. Alexander H. Levis
Title : Senior Research Scientist

ACKNOWLEDGEMENTS

I wish to express my gratitude to:

Alexander Levis for his support and guidance throughout my research assistantship. His professional commitment to excellence and clarity of exposition has been a challenging experience from all personal and academic points of view.

Marcel Roulet, who helped me make these two years at MIT possible.

Manal Wardé, who encouraged me all along the way.

Nibras El Fadel, whose company is a constantly enlightning one.

Andreas Kussmaul, who assisted me in discovering how tasks can be assigned in a two roommate distributed intelligence system. Over the past two years, I became the expert in vacuum cleaning on Beacon Hill, and still think that Andreas is a very nice person.

François Valraud, for hours of discussion we had over issues of various interests, mostly unrelated to Petri Net theory.

Lauren Seeley for her patience at polishing my English.

For the friends I have at MIT, Anne-Claire, Jinane, Louise, Sabina, Vicky, Didier, Jose, and Camé, who brought smiles, laughs, and enjoyable times.

My family.

This research was conducted at the MIT Laboratory for Information and Decision Systems with support provided by the Office of Naval Research under Contract no. N00014-84-K-0519 (NR 649-003).

To Jean G. Suard

TABLE OF CONTENTS

Page

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 MOTIVATION

Administrative authorities, because they address the global needs of society, are involved more and more in the development of extremely complex organizations. Typically, the legislative or executive branches of government mandate an authority to design, develop, and monitor systems that perform desirable functions. The administrative authority has to specify requirements for the system, to define a development plan, to improve the system over time, and to submit its conclusions and projects to the political authorities for approval and funding.

Feasibility studies constitute critical stages for determining whether or not a project should be developed, and whether or not a system should be upgraded. The team in charge of research and development has to answer unambiguously the concerns of policymakers and contractors. The team must define the technical requirements and demonstrate to the policymakers that a system can be implemented or upgraded within the given technological and financial constraints. The team must show that the proposed system will offer substantial advantages. To sustain the claims, estimates of system performance have to be developed. The political authorities, and expert panels that are consulted, can base their decision on these quantitative measures. The predictions furnish criteria by which the actual performance of the system will be assessed by policymakers, and by which decisions about the outcome of the project will be made.

Many unresolved issues remain in the design and analysis of large-scale distributed systems, such as the Air Traffic Control System, the various systems of the Department of Defense, and the information systems of large administrative centers. Such systems are large-scale because they mobilize many resources: Human decisionmakers, computer and communication systems. Such systems are distributed because they must perform several functions to accomplish their mission, and the functions are spread out over a range of locations and divided into individual subtasks, so that each part's activity contributes a little to each of the several functions (Minsky, 1986). The different entities coordinate their activities by exchanging information over communications systems. Following are three examples:

- Air Traffic Control: Several air traffic control centers are located throughout a country to cover its airspace, to direct planes into flight corridors, and to schedule traffic flow. In each center, the controllers are highly specialized professionals who operate according to certain rules. Sensors, computers, and communications systems help them accomplish their tasks.
- Command and Control systems. The best definition of command and control is given in Publication 1 of the Joint Chiefs of Staff: "Command and Control is the exercise of authority and direction by a properly designated commander over assigned forces in the accomplishment of his mission. Command and control functions are performed through an arrangement of personnel, equipment, communications, facilities, and procedures which are employed by a commander in planning, directing, coordinating and controlling forces and operations in the accomplishment of his mission". Most systems of the Department of Defense fall within this paradigm.
- Office Automation. In this context, the model describes the exchange of information that takes place within a large administrative complex. The exchange of information can be aided by a computerized information system, for example, when timeliness and throughput rate are crucial.

Design and analysis of such organizations and their supporting information systems are highly complex tasks. They involve the cooperation of people with different backgrounds (policymakers, users, analysts, engineers) and conflicting interests. Consequently, some method is needed to achieve a common understanding, or to illustrate their differences. To do so, there are two major options, either modeling or real-scale testing.

Usually, these problems are addressed by building formal models. The assumption is that models precisely define specific properties or characteristics of a system under study, and provide the foundation for verifying these properties. As quoted in Hommel (1985): "What is not specified cannot be verified, and what is not verified may be in error." The economic rationale is the fact that the systems are generally so complex that the other option, exhaustive real scale testing or test bed experimentations, requires large amounts of time and resources that cannot be spent on every dimension of the system. The system designers and the policymakers have to rely on tools that yield, "cheaply and quickly," reasonable, comprehensible insights into the proposed structures. By these criteria the analysts determine the critical aspects to be tested extensively .

The difficulty is that the development of a common understanding usually involves customers, users of the proposed system on one side, and developers, designers on the other side. This means that the participants are often divided between those who understand abstraction and formal terminology and those who cannot. For the latter, formal terminology is unacceptable. Yet without formal terminology, the specifications of a system cannot be a basis for development or analysis.

1.2 PROBLEM DEFINITION

Recent developments in the theory of Distributed Intelligence Systems (Levis, 1988) have addressed the problem of analyzing the performance of a given architecture, or of designing an organization whose performance would meet specific requirements. Issues related to the design and analysis of fixed structure systems are becoming well understood. In fixed structure systems, the interactions between components are fixed and well defined. To meet requirements of reliability and reconfigurability from users, variable structure systems need to be addressed. In variable structure systems, the interactions between components can change depending on the task, while the same task can be performed with different combinations of resources. Indeed, some pattern of interactions may be more suitable for the processing of a given input than others. If designed properly, a variable structure system can be expected to achieve a higher overall performance, provided that it adapts its structure to the most appropriate interactions for each type of input.

A framework has been already developed to compare quantitatively organizational designs of large scale distributed systems. This framework includes both fixed and variable organizations. Three problems have to be addressed to implement a methodology that links the design problem to existing analytic tools:

(a) A framework that is appropriate for a mathematical formulation of the design problem should be identified. It should give the designer the means to tackle quantitatively his practical problem and should be applicable to fixed as well as variable structure architectures.

(b) The concept of variability has to be refined. Several types of variability must be distinguished, based on the characteristics of the system that vary.

(c) In generating distributed systems architectures, designers face the problem that the enumeration of all the systems that satisfy the constraints of the design is intractable, even for systems with a small number of resources. The design problem should be kept computationally feasible by putting restrictions on the methodology without drastically altering its scope.

This effort would fill a gap between the analytic tools and the design issues. Once a designer has specified his requirements for the system, he can obtain every candidate structure that satisfies his requirements. Once the structure is chosen, it can be evaluated quantitatively.

1.3 THEORETICAL BACKGROUND

A quantitative methodology for modeling, evaluation, and design of fixed structure systems has been developed at the MIT Laboratory for Information and Decision Systems (Boettcher and Levis, 1982; Andreadakis and Levis, 1987; Remy et al., 1988).

The organization is seen as a system performing a task. The processing of the task is achieved through the execution of well defined procedures or algorithms that human decisionmakers, intelligent nodes, and the supporting information system possess. In this model, the human decisionmakers and the intelligent nodes have a four stage internal structure that makes it possible to differentiate types of interactions.

The mathematical formulation of the fixed structure problem is based on Ordinary Petri Net theory (Reisig, 1985). Petri Nets have been introduced to model organizational forms, as they show explicitly the interactive structure and the sequence of operations between the components of the organization. Petri Nets have proven their efficiency as modeling and analytical tools. In particular, the formalism of System Effectiveness Analysis (Dersin and Levis, 1981; Bouthonnier and Levis, 1984; Cothier and Levis, 1986) yields quantitative estimates of the extent to which a specific structure satisfies the mission requirements.

An indirect approach to the generation of architectures has been developed for fixed structure organizations. In Remy and Levis (1988) a framework was presented which allows designers to express their design problem in mathematical terms. Then, an algorithm was developed that makes it possible to characterize and generate partially ordered sets of fixed structures that satisfy the designers' requirements. The computational requirements of the algorithm are very modest.

Monguillet and Levis (1988) initiated the investigation of variable structure decisionmaking organizations. He introduced the use of extensions of Ordinary Petri Net theory, High Level Nets, to deal with variability. Two major models of High Level Nets have been developed by Net theorists: Predicate Transition Nets (Genrich, 1987) and Colored Nets (Jensen, 1987). These models follow different approaches, but are equivalent (Every Colored Petri Net can be translated into a Predicate Transition Net, and vice versa). Based on the theory of Predicate Transition Nets, Monguillet extended the framework of System Effectiveness Analysis for comparing both variable and fixed structure organizations.

1.4 GOALS AND CONTRIBUTION

This thesis takes the point of view of the members of a Research & Development team. They want to study all feasible configurations of the system, given the constraints of both the mission and the technology. An additional constraint is that they must interact with non technically oriented people. This thesis answers the question of whether or not it is possible to devise a methodology that satisfies the conflicting interests, i.e., a methodology that yields significant improvements in the design process.

This thesis presents a major extension of the earlier work by addressing the problem of designing variable structure organizational forms. An appropriate mathematical framework, that of Colored Petri Nets, is defined to investigate systems that adapt their structure of interactions to the input they process. The properties of such a set of variable structure organizations is presented. An illustrative application, the design of two hypothetical systems - one civilian and one military - is described. Both the advantages and the limitations of the methodology proposed in the thesis are addressed, based on these examples. Then, a policy analysis is performed that analyzes how the methodology better satisfies the needs of the policymakers, the users, and the analysts. Finally, recommendations on the use of the methodology are presented.

1.5 THE THESIS IN OUTLINE

This thesis is organized as follows. Chapter II is an introduction to Petri Net theory, and describes basic notions about Ordinary Petri Nets and Colored Petri Nets. Chapter III is a review of the results on equivalence relations and partial orderings that are used in the subsequent chapters. In Chapter IV, a methodology to model, represent, and analyze variable structure distributed intelligence systems is defined. Chapter V translates this framework into the language of Colored Petri Nets, and formulates the design problem of a Research and Development team. Chapter VI introduces the constraints that must be satisfied by a solution to the design problem. Chapter VII characterizes the set of solutions to a design problem as well as some of its internal properties. Two illustrative applications are presented in Chapter VIII. Chapter IX analyzes the political and scientific environments within which this research has been conducted. Finally, Chapter X concludes this thesis and suggests some directions for further work.

CHAPTER II

PETRI NET THEORY

This chapter is an introduction to Petri Net theory. First, the basic level of the formalism, that of Ordinary Petri Net, is presented. Then the limitations of this framework, as far as the modeling of nets with variable structures is concerned, are identified. One extension of the theory that overcomes some difficulties has been described in the literature as High Level Nets. Two major models have been developed within that approach, Predicate Transition Nets and Colored Petri Nets. The concepts of High Level Nets are presented using Colored Petri Nets, which will be used below. More introductory material can be found in Peterson (1981), Brams (1983), and Reisig (1985). High Level Nets have been described in Genrich and Lautenbach (1981). Advanced materials on Predicate Transition Nets are provided in Genrich (1987) and Monguillet (1988). An extensive presentation of Colored Petri Nets is given in Jensen (1987).

2.1 INTRODUCTION

Large-scale distributed systems have certain characteristics:

They exhibit *concurrency* or parallelism. Several components can work at the same time on the same task. There is thus a need to represent the precedence relations between the processing of the different components. The precedence relations determine the degree of parallelism that is present in the system.

These systems very often offer *alternatives*. One process may be done by several components, or several combinations of components. Conversely, a particular component is usually able to perform different types of processes.

A choice may create a *conflict*. Suppose, for example, that a component A has been assigned to some task B. Suppose that some task C arrives that can only been processed by A. C cannot be processed. One option is that the conflict will be resolved, so that the processing of C will be done as soon as A finishes the processing of B. However, if it is not resolved, the system may reach a *deadlock*.

The operations executed by the various components are *asynchronous*. There are no global mechanisms that coordinate the scheduling of the processings. Each component usually starts its processing as soon as it has received all the information it needs. If several tasks are requested, a queuing discipline (First In First Out (FIFO), Last In First Out (LIFO), etc...) is enforced to schedule the individual requests.

Petri Nets have been introduced in the modeling of Distributed Systems because they give a graph-theoretic representation of the communication and control patterns, and a mathematical framework for analysis and validation. Petri Net modeling is appealing for the following reasons:

- Petri Nets provide an integrated methodology, with well developed theoretical and analytical foundations, for modeling physical systems together with complex cognitive decision processes.
- Petri Nets capture the precedence relations and structural interactions of concurrent and asynchronous events. Deadlocks and conflicts can be easily identified on a Petri Net .
- The graphical nature of Petri Nets helps to visualize easily the complexity of the system. They are thus appealing both to the layman as to the analyst.
- Various extensions of the basic theory allow for quantitative analysis of resource utilization, throughput rate, effect of failures, and real time implementation.

2.2 ORDINARY PETRI NETS

2.2.1 Definitions

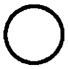
Definition 1.1

An *Ordinary Petri Net* is a bipartite directed graph: (P, T, I, O).

There are two sets of nodes:

- P = {p1, ..., pn} a finite set of *places*.

  A place is depicted by a circle node.

  A place models a resource, a buffer, or a condition.
- T = {t1, ..., tm} a finite set of *transitions*.

  A transition is represented by a bar node.

  A transition stands for a process, an event, or an algorithm.

- The arcs or connectors that connect those nodes are directed and fixed. They can only connect a place to a transition, or a transition to a place. They are given by:
  - I : P × T -> {0,1}

    I is an input function that defines the set of *directed arcs* from P to T.

    I(p,t) = 1 if the arc exists, *I(p,t) = 0* otherwise.

    An arc from a place p to a transition t indicates that the process t requires the availability of the resource p, the fulfillment of the condition p, or the availability of information in the buffer p, in order to occur.
  - O : P × T -> {0,1}

    O is an output function that defines the set of *directed arcs* from T to P.

    O(p,t) = 1 if the arc exists, *O(p,t) = 0* otherwise.

    An arc from a transition t to a place p indicates that when the process t is finished, it either enables the condition p, makes the resource p available, or sends an item of information to the buffer p.

Example 2.1: Consider the Ordinary Petri Net shown in Figure 2.1

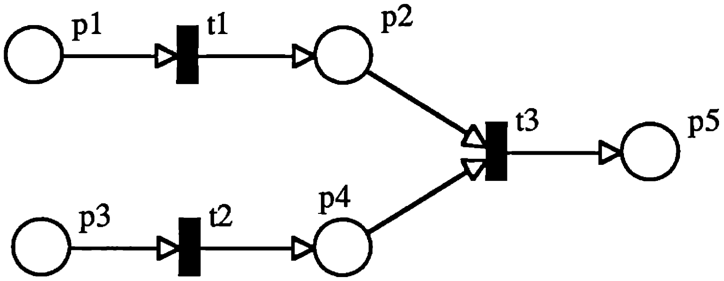


Fig. 2.1 Ordinary Petri Net

The set of places P, the set of transitions T, and the input and output functions that define the arcs for this net are:

P = { p1, .. p5}　　T = {t1, t2, t3}

I(p1, t1) = I(p2, t3) = I(p3, t2) = I(p4, t3) = 1　　I(p, t) = 0 otherwise.

O(p2, t1) = O(p4, t 2) = O(p5, t3) = 1　　O(p, t) = 0 otherwise.

This Petri Net describes the elementary task of writing. The place p1 describes a resource, the pens on the desk. The place p2 represents the condition that one has taken a pen in the hand. Place p3 represents a stack of paper in the desk drawer, place p4 indicates that paper is on the desk. Finally, place p5 indicates that one is writing. There are three processes in the system. The process t1 models the act of picking a pen, while the act of getting some paper is modeled by t2. Transition t3 models the start of the writing activity.

One can pick up a pen only from the pens on the desk (Link from p1 to p2). Similarly, some paper is on the desk only if it has been taken from the drawer (Link from p3 to p4). Finally, the links from p2 and p4 to t3 indicate that one needs both a pen in the hand and some paper on the desk to start writing.

Definition 2.2

A Petri Net is *pure* if and only if it has no self loop, i.e., no place that can be both an input and an output of the same transition.

The net of Fig. 2.1 is pure. All Petri Nets that are considered in this thesis are pure. For an extensive discussion of this modeling issue, see Hillion and Levis (1986).

Definition 2.3

A *path* is a set of k nodes and k - 1 connectors, for some integer k, such that the i-th connector either connects the i-th node to the i+1-th node or the (i + 1)-th node to the i-th node. The path is *directed* if the i-th connector connects the i-th node to the (i + 1)-th node for all i = 1,..k..

Example 2.2: In Figure 2.1

p3 - t2 - p4 - t3 - p5 is a directed path,

p5 - t3 - p2 is not a directed path.

If a Petri Net has sources and sinks, then any path from a source to the sink is called an *information flow path.* If an information flow path is a set of k nodes such that the k nodes are distinct, then the information flow path is said to be *simple*. The path p1 - t1 - p2 - t3 - p5 is, for example, a simple information flow path of the Petri Net of Figure 2.1.

Definition 2.4

A Petri Net is *connected* if and only if there exists a path - not necessarily directed - from any node to any other node.

Fig. 2.1 depicts a connected net. Intuitively, this definition formalizes the idea that a Petri Net models a whole system. There are no partitions of the set of nodes into disjoint subsets, such that the nodes in one subset are not connected to the other subsets.

Definition 2.5

A Petri Net is *strongly connected* if and only if there exists a directed path from any node to any other node.

The net of Fig. 2.1 is not strongly connected because, for example, there is no directed path from p1 to p2.

2.2.2 Petri Nets with Markings

A Petri Net can contain *tokens*. Tokens are depicted graphically by indistinguishable dots (•), and reside in places. The existence of one or more tokens represents either the availability of the resource, or the fulfillment of the condition, or the number of items of information in the buffer. The travel of tokens through the net is controlled by the transitions. A *marking* of a Petri Net is a mapping M that assigns a non negative integer (the number of tokens) to each place.

Example 2.3: Consider the Petri Net in Fig. 2.2 with the indicated marking.

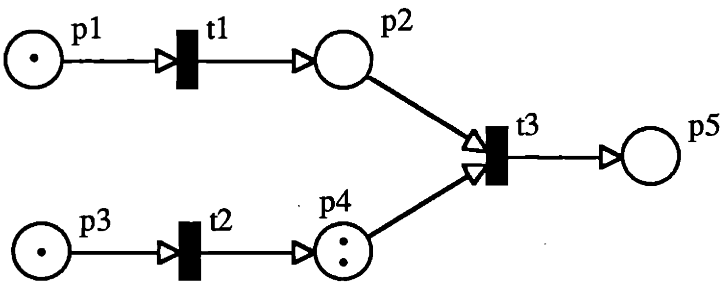


Fig. 2.2 Petri Net with Marking

$$M(p1) = M(p3) = 1; \qquad M(p4) = 2; \qquad M(p2) = M(p5) = 0.$$

It is the same net shown in Fig. 2.1. If the interpretation of Fig. 2.1 is used, this marking indicates that there is one pen on the desk, one sheet of paper in the drawer, and two sheets of paper on the desk

Definition 2.6

A transition is *enabled* by a marking, if and only if all of its input places contain at least one token.

In Example 2.3, t1 and t2 are enabled. All the conditions to be satisfied are fulfilled. One can either pick up the single pen or put another sheet of paper on the desk.

Definition 2.7

An enabled transition can *fire*. The firing of the transition corresponds to the execution of the process or the algorithm. The dynamical behavior of the system is embedded in the movement of the tokens; when the firing takes place, a new marking is obtained by *removing* a token from each input place and *adding* a token to each output place.

Example 2.4: In Fig. 2.2, if t1 fires, then the resulting marking is shown in Fig. 2.3.

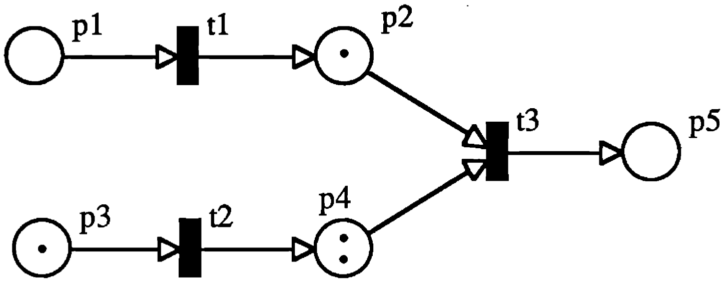


Fig. 2.3 Petri Net after Firing

Transitions t3 and t2 are now enabled. If t3 fires, the new marking is shown in Figure 2.4.

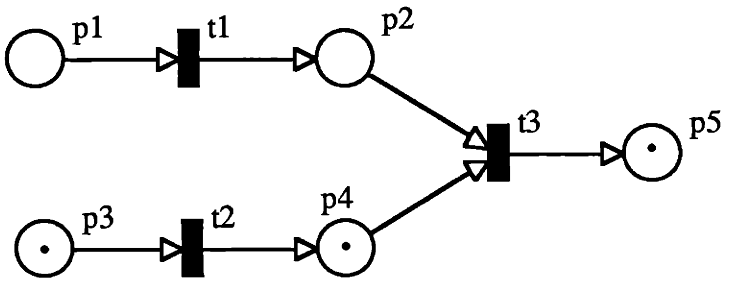


Fig. 2.4 Petri Net after second Firing

Remark: A transition may fire concurrently more than one token, i.e., a process may handle several tasks at the same time. Each firing of a transition is thus characterized by an integer k, the *firing pattern* of the transition. A transition can fire according to the firing pattern k, if and only if all of its input places have at least k tokens. When the firing takes place, k tokens are removed from each input place, and k tokens are added to each output place. The firing pattern is 0 if a transition does not fire.

2.2.3 Linear Algebraic Approach

So far, Petri Nets have been described as graphs. An alternative and equivalent approach can be developed using linear algebra with integer coefficients (Memmi and Roucairol, 1980).

Definition 2.8

A Petri Net with n places and m transitions can be represented by a n × m matrix C, the *Incidence Matrix*. The rows correspond to places, the columns correspond to transitions.

- $C_{i,j} = 1$ if there is a directed arc from the j-th transition to the i-th place. 1 indicates that the firing of the j-th transition adds one token to the i-th place.
- $C_{i,j} = -1$ if there is a directed arc from the i-th place to the j-th transition. -1 indicates that the firing of the j-th transition removes one token from the i-th place.
- $C_{i,j} = 0$ if there is no arc from the j-th transition to the i-th place.

Example 2.5: The incidence matrix of the net on Fig. 2.1 is

$$C = \begin{array}{c|c} \begin{array}{ccc} t1 & t2 & t3 \end{array} & \\ \left[\begin{array}{ccc} -1 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{array}\right] & \begin{array}{c} p1 \\ p2 \\ p3 \\ p4 \\ p5 \end{array} \end{array}$$

Properties

- The marking of a net can be represented by a n × 1 vector *M*, where $M_i = M(pi)$. The i-th entry corresponds to the number of tokens in the i-th place.
- The firing pattern of the net can be represented by an m × 1 firing vector *F*, where F_j is the firing pattern of the i-th transition.
- Given an incidence matrix C, an initial marking M, and a firing pattern F, the new marking M' is

$$M' = M + C * F. \qquad (2.1)$$

Example 2.6: The matrix equation that corresponds to the firing of Fig. 2.3 is

$$M' = \underset{M}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \\ 0 \end{bmatrix}} + \underset{C}{\begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}} * \underset{F}{\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \\ 0 \end{bmatrix} \qquad (2.2)$$

2.2.4 Invariants

An incidence matrix makes it possible to use results from linear algebra to infer properties of the net. Much of the literature is devoted to the study of S- invariants.

Definition 2.9

Given an incidence matrix C, an *S-invariant* is a n × 1 non-negative integer vector X of the kernel of C^T, i.e.,

$$C^T * X = 0 \qquad (2.3)$$

Remark: One must pay particular attention to the fact that S must have non-negative integer coefficients. The rationale for this constraint results from Theorem 2.1, which gives a physical interpretation to S-invariants.

Theorem 2.1

Let M_0 be any initial marking, and M be any marking that is reachable from M_0 after a sequence of firings. X is an S-invariant if and only if for any M_0 and any M,

$$X^T * M = X^T * M_0. \tag{2.4}$$

This relation is interpreted as a weighted conservation of tokens. A marking is by definition a vector of non-negative integers. Conservation of tokens must thus be expressed with non-negative integers.

Definition 2.10

If X is an S-invariant, the set of places whose corresponding components in X are strictly positive is the *support* of the invariant, noted <X>.

The support of an S-invariant is said to be *minimal* if and only if it does not contain the support of another S-invariant but itself and the empty set.

Theorem 2.2

If X^1 and X^2 are two S-invariants with the same non empty minimal support, then X^1 and X^2 are linearly dependent.

Proof.

Consider $X^1 = [x^1_i]$ and $X^2 = [x^2_i]$, i = 1,..,n. By assumption, X^1 and X^2 are non null vectors. Nothing is changed if it is assumed that the support is made out of the first p, $0 < p \leq n$, places.

Define $r = \min_{i = 1..k} (x^1_i / x^2_i)$ and m an integer large enough so that for every i $m*r*x^2_i$ is an integer.

Then $m * (X^1 - r * X^2)$ is an S-invariant whose support is strictly included in the support of X^1 and X^2.

Indeed, $C^T * m * (X^1 - r * X^2) = m * C^T * X^1 - m * r * C^T * X^2 = 0 - 0 = 0$. For every i, $m * x^1_i - m * r * x^2_i$ is an integer (Definition of m), and $x^1_i - r * x^2_i$ is non negative

(Definition of r). Finally, by definition of r there exists some i0 such that $r = x^1_{i0} / x^2_{i0}$, hence $m.(X^1 - rX^2)_{i0} = 0$.

Consequently, the support of $m * (X^1 - r * X^2)$ is Ø. Thus $m * (X^1 - r * X^2)$ is zero. The vectors are linearly dependent.

Definition 2.11

A *minimal support S-invariant* X is an S-invariant whose support <X> is minimal.

The following important result, due to Memmi and Roucairol (1979), highlights the importance of minimal support S-invariants. Valraud (1989) presents an application of this result to analyze structural properties of a net.

Theorem 2.3

Consider a net P. The set of minimal supports of the net P is finite.

If $<X>_1$, ..., $<X>_k$ are the k finite supports, and X_1,..., X_k is a family of S-invariants, with $<X_i> = <X>_i$, then the family X_1,..., X_k constitutes a minimal generating family of the S-invariants, i.e.,

every S-invariant can be written as a linear combination of X_1,..., X_k with rational coefficients.

Definition 2.12

The *S-component* associated with an S-invariant X of a Petri Net P is the subnet of P whose places are the places of <X> and whose transitions are the input and output transitions of the places of <X>.

By extension, a *minimal S-component* is the S-component of a minimal support S-invariant.

Example 2.7: Consider the Petri Net P of Figure 2.5.

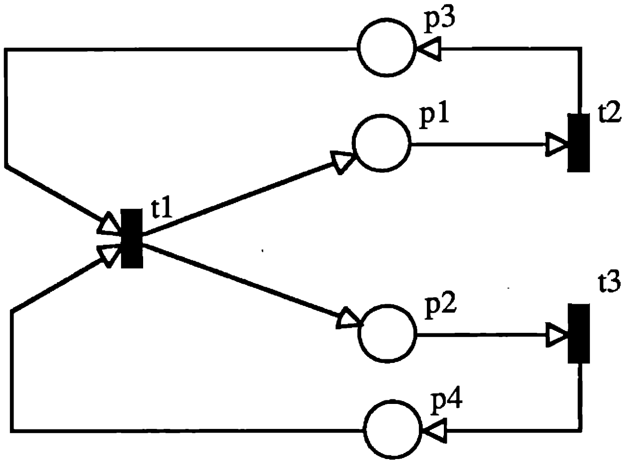


Fig. 2.5 Petri Net P

The incidence matrix of P is

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

X = [x1, x2, x3, x4] is an S-invariant if and only if $C^T * X = 0$.

This yields x1 = x3 and x2 = x4. There are two minimal supports <X1> = {p1, p2} and<X2> = {p3, p4}. The S-components associated with <X1> and <X2> are depicted in Figures 2.6 and 2.7.

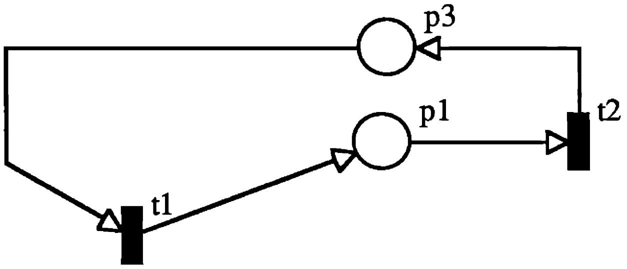


Fig. 2.6 S-component associated with <X1>

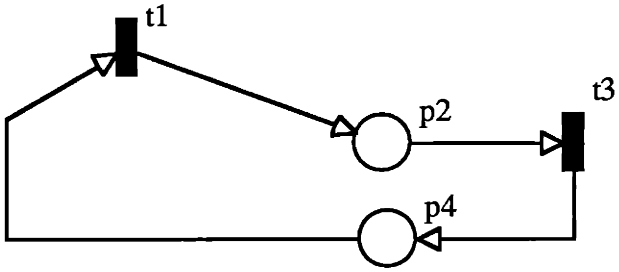


Fig. 2.7 S-component associated with <X2>.

2.2.5 Marked Graphs

Definition 2.13

A *marked graph* is a connected Petri Net in which each place has exactly one input and one output transition.

Throughout this thesis, marked graphs play an important role. One crucial result about marked graphs is Theorem 2.4 (Hillion, 1986). This result has been applied extensively in Remy (1986) to characterize Petri Net model of fixed structure systems, and is used in Chapter VII.

Example 2.8: The net in Figure 2.1 is not a marked graph., because this net has two sources, i.e. two places without input arc, and one sink, i.e. a place without output connector. Figure 2.8 shows a marked graph.

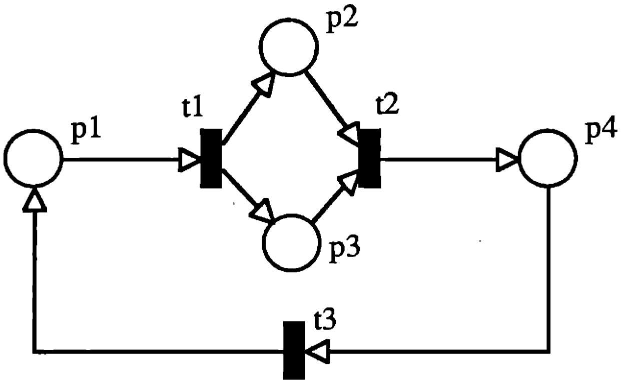


Fig. 2.8 Marked Graph

Theorem 2.4 is stated after the introduction of two new terms.

Definition 2.14

A *directed circuit* is a directed path from one node back to itself.

In Fig. 2.8 p1-t1-p3-t2-p4-t3-p1-t1-p2-t2-p4-t3-p1 is a directed circuit.

A *directed elementary circuit* is a directed circuit in which only one node appears more than once.

In Fig. 2.8, p1-t1-p3-t2-p4-t3-p1 is a directed elementary circuit. The place p1 is the node that appears more than once.

Theorem 2.4

The minimal S-components of a marked graph are exactly its directed elementary circuits.

Theorem 2.4 is important, because it indicates that the computation of the minimal S-components can be done by an efficient algorithm based on Linear Algebra, such as the algorithm of Alaiwan and Toudic (1985).

In this thesis, a particular type of nets are of importance. In these nets, all the places but two have exactly one input and one output transition. There is one place with only one output transition (the source or the external place) and one place with one and only one input transition (the sink). These nets can be transformed into marked graphs by merging the external place and the sink into a single place p0. Under those circumstances, the simple information flow paths from the source to the sink are exactly the directed elementary circuits that contain the place p0. The simple information flow paths can be computed in that case using the efficient algorithm of Alaiwan and Toudic. See Valraud (1989) for an extensive treatment.

2.2.5 PETRI NETS WITH SWITCHES

The theory of Ordinary Petri Nets does not provide a convenient way to model a stage at which several alternatives exist. For that purpose, a modified transition, a *switch*, has been introduced in Levis (1984).

A switch is a node with multiple output places. As with any transition, a switch is enabled whenever there is at least one token in each of its input places. When a switch fires, a token is put in *only one* of its output places. This place is chosen according to some decision rule.

The decision rules associated with the switch can be anything. They can be deterministic or stochastic. They can take the information that is contained in the inputs into account, etc. It is thus possible to model distributed variable structures with switches.

Example 2.9: Figure 2.9 represents a Petri Net with a switch.

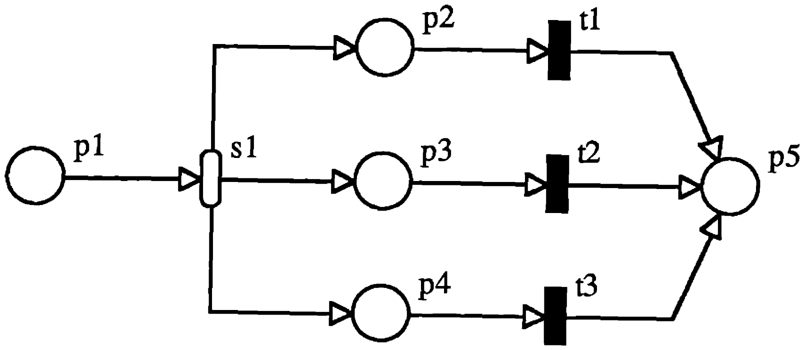


Fig. 2.9 Petri Net with Switch

At the stage modeled by the switch s1 there are three alternative courses of action. According to some rule, only one is chosen. In each case, the course of action that is chosen will satisfy the condition modeled by p5.

2.3 COLORED PETRI NETS

2.3.1 Introduction

The theory of Ordinary Petri Net has some drawbacks. First, when it comes to the modeling and analysis of real distributed systems, an Ordinary Petri Net model may become very large. A detailed analysis shows that the size of the nets tends to increase dramatically because the model fails to exploit some symmetries. For example, when one process is used repetitively, unnecessary replications of parts of the net occur.

Second, as was stated in section 2.2.5, the grammar of the theory is poor as far as varying structures are concerned. To overcome some difficulties, switches were introduced. However, these tools are far from perfect. Consider for example the case of the net on Figure 2.10.

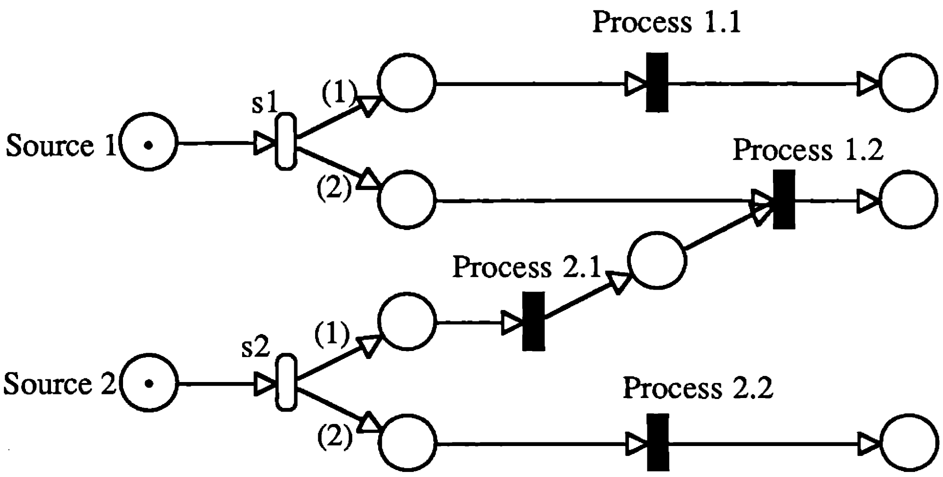


Fig. 2.10 Petri Net with two Switches

There are two stages, s1 and s2, at which some decision must be made. Each switch has two settings and each source has one token. Switch s1 chooses the setting (2), and switch s2 chooses (2). This yields the net and the marking of Figure 2.11.

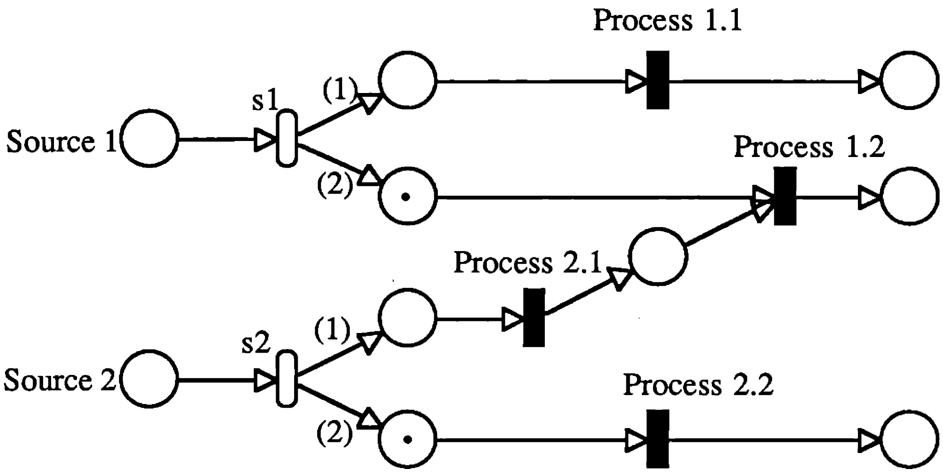


Fig. 2.11 Petri Net after Switch Settings

The system has reached a deadlock! Process 1.2 must receive some item of information from process 2.1, but this item will never arrive because the token of Source 2 has been put into the input place of process 2.2. Unless having a deadlock is a desirable property under these circumstances, one can observe in this example that the settings of the switches cannot be independent. There is a thus a need for a tool that indicates the correlation of the rules. Furthermore, to be consistent with the graphical nature of Petri Nets, the correlation of the rules should be expressed on the graph.

Finally, in most cases one would like to have the information content of the tokens. This information is important if the structure varies depending on the content, or/and if the system can handle several types of tasks simultaneously, etc.

In order to improve the modeling and analytical power of Ordinary Petri Nets, extensions called High Level Nets have been devised. These models are based on common ideas:

- The tokens can be differentiated. They have an identity (color).
- The identity describes some information about the physical meaning of the token. If a token models a communications message, its identity may be the pair (Sender, Receiver), which describes the source and the destination of the message.
- They provide a way for describing the logic that drives the control process of the transitions.

2.3.2 Definitions

Let us illustrate these notions by introducing Colored Petri Nets, as defined in Jensen (1986), which are used in the rest of this thesis to model variable structure distributed systems.

Definition 2.15

A *Colored Petri Net* (CPN) is given by (P, T, $C(t)_{t \text{ in } T}$, $C(p)_{p \text{ in } P}$, $M(p)_{p \text{ in } P}$, I, O):

- *P*, a set of places.
  As in Ordinary Petri Nets, a place models a resource, a buffer, or a condition, and is depicted by a circle node.
- *T*, a set of transitions.
  As in Ordinary Petri Nets, a transition models a process, an event, or an algorithm and is depicted by a bar node.

- Each transition t has attached to it a finite set of *occurrence-colors* with π_t elements: $C(t) = \{ C(t)1, ..., \quad C(t)\pi_t\}$. Each occurrence color corresponds to one firing mode: to one pattern of behavior of the process. In the case of Ordinary Petri Nets, when a process offers s > 1 courses of action, it is modeled by a switch with s branches. In the case of Colored Petri Nets, every process is modeled by a transition and the set C(t) keeps track of the alternative courses of action.
- Each place has attached to it a finite set of *token-colors* with π_p elements. $C(p) = \{C(p)1, ..,C(p)\pi_p\}$. Each token color corresponds to one type of information content attached to a token. Only tokens that have their color in C(p) can be placed in p and one place can simultaneously contain several tokens with the same color. Conversely, one place can contain tokens of different types, provided that their color belongs to C(p).
- The marking of a place is a $\pi_p \times 1$ vector M(p) $M(p) = [\beta_1, .., \beta_i, ..., \beta_{\pi p}]^T$, where β_i indicates that p contains β_i tokens of color C(p)i.

Finally, the input and output incidence functions *I* and *O* are are such that:

- *I(p,t) is a* $\pi_p \times \pi_t$ *matrix* for any transition t and any place p.

  The rows correspond to the elements of the set of token colors, C(p), that is the colors of the tokens that can be put in the place p. The columns correspond to the elements of the set of occurrence colors of t, C(t), the alternative courses of actions.

  $I(p,t)_{i\,j}$ describes the number of tokens of color C(p)i that are removed from the place p, when the transition t fires according to the firing mode j.

  If some entries of I(p, t) are non null, an arc is drawn from the place p to the transition t in the graphical representation of the Colored Petri Net. This arc is annotated by the matrix I(p, t).
- *O(p,t) is a* $\pi_p \times \pi_t$ *matrix* for any transition t and any place p.

  The rows correspond to the elements of the set of token colors, C(p), the colors of the tokens that can be put in the place p. The columns correspond to the elements of the set of occurrence colors of t, C(t), the alternative courses of actions.

  $O(p,t)_{i\,j}$ describes the number of tokens of color C(p)i that are put in the place p, when the transition t fires according to the firing mode j.

  If some entries of O(p, t) are non null, an arc is drawn from the place p to the transition t in the graphical representation of the Colored Petri Net. This arc is annotated by the matrix O(p, t).

2.3.3 Examples

Let us develop two examples to understand the concepts of Colored Nets.

Example 2.10: Consider a mail clerk sorting mail. This clerk sorts the mail for three people and has to put it in appropriate boxes. Suppose that the initials of the persons are J, RN, and RT. The job of sorting the mail can be represented by a Colored Petri Net in Figure 2.12.

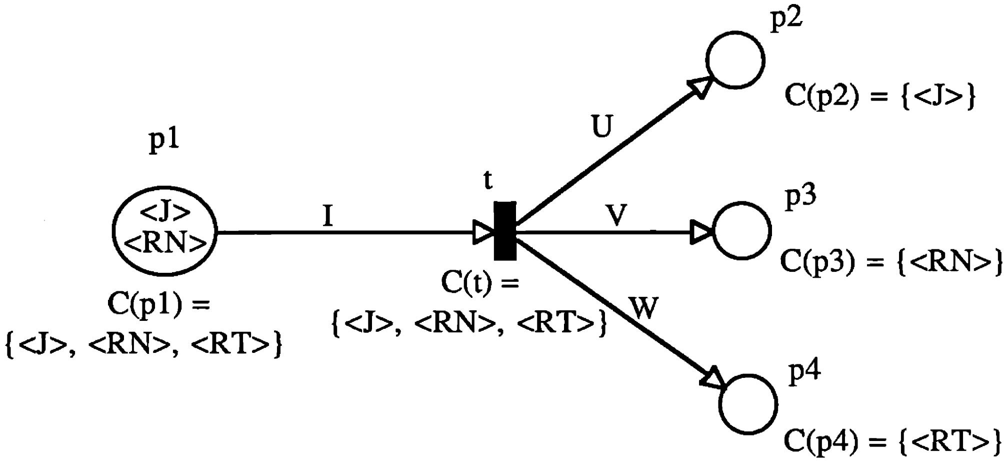


Fig. 2.12 Mail Sorting

In this example, every place contains tokens, which represent letters. It is thus very natural to assume that the color of the tokens belong to {<J>, <RN>, <RT>}, the names on the envelope.

Place p1 stands for the letters to be sorted. Its Color set is {<J>, <RN>, <RT>}, as this place can contain any combination of letters to be distributed to J, RN, and RT. The initial conditions depicted on Figure 2.12 are that two letters, one for J and one for RN have to be sorted.

Place p2 models J's mailbox, which contains exclusively the letters for J, C(p2) = {<J>}. Similarly, p3 is the mailbox for RN, and C(p3) = {<RN>}. Finally, the place p4 is the mailbox for RT, and C(p4) = {<RT>}.

The transition t models sorting. The clerk has three courses of actions, which have also been labeled by <J>, <RN>, and <RT>. J models the fact that the clerk is sorting some mail for J, <RN> models sorting mail for RN, and <RT> models putting mail into RT's mailbox.

The arcs of the CPN have been annotated by the matrices I, U, V, W, where I is

$$I = \begin{array}{cccc} \langle J\rangle & \langle RN\rangle & \langle RT\rangle & \\ 1 & 0 & 0 & \langle J\rangle \\ 0 & 1 & 0 & \langle RN\rangle \\ 0 & 0 & 1 & \langle RT\rangle \end{array}$$

I indicates that the clerk takes only one letter for J when he sorts mail for J, that he takes one letter for RN when he sorts mail for RN, and that he takes only one letter for RT when he sorts mail for RT. The other matrices that annotate the arcs are

$$U = \begin{array}{ccc} (J) & (RN) & (RT) \\ [1 & 0 & 0] \end{array} \quad V = \begin{array}{ccc} (J) & (RN) & (RT) \\ [0 & 1 & 0] \end{array} \quad W = \begin{array}{ccc} (J) & (RN) & (RT) \\ [0 & 0 & 1] \end{array}.$$

U indicates that one letter is put into J's mailbox if the clerk sorts mail for J.
V indicates that one letter is put into RN's mailbox only if the clerk sorts mail for RN.
W indicates that one letter is put into RT's box if and only if the clerk sorts mail for RT.

This example is simple, because the token-colors are natural, and because the firing modes of t correspond exactly to the token-colors. Note that this is variable system, the clerk puts the mail in different mailboxes, depending on the name on the envelope. Example 2.11 describes a Colored Petri Net that is less obvious.

Example 2.11:

Figure 2.13 represents another Colored Petri Net.
The set of places is P = {p1, p2, p3, p4, p5}.
The set of transitions is T = {t1, t2, t3}.
The set of token-colors for p1 and p2 is A = {a1, a2}.
The set of token-colors for p3 and p4 is B = { b1, b2}.
The set of token-colors for p5 is C = {c1, c2, c3}.
The transitions t1 and t2 have two firing modes, which are labeled 1 and 2.
The transition t3 has three firing modes, which are labeled 1, 2, and 3.

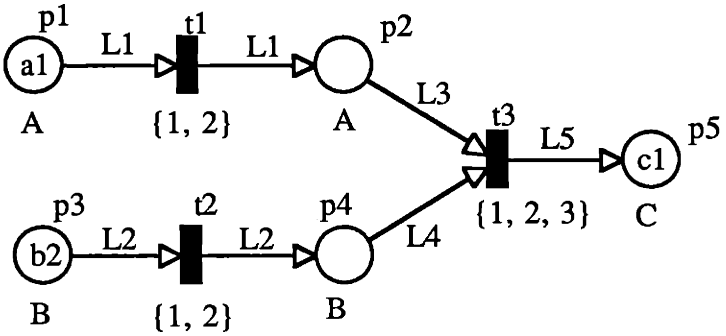


Fig. 2.13 Colored Petri Net

where the matrices that annotate the arcs are

$$L1 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, L2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, L3 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, L4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, L5 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- L1 indicates that the first firing mode of t1 (first column of L1) removes one token of color a1 from p1, and places one token in p2. The second firing mode (second column) removes one token of color a1 and one token of color a2, and places both in p2 .
- L2 indicates that the first firing mode of t2 removes one token of color b1 from p3, and places it in p4. The second firing mode removes a token of color b2, and places it in p4.
- L3 indicates that the first two firing modes of t3 remove one token of color a1 from p2. The third firing mode removes one token of color a2.
- L4 indicates that the first firing mode of t3 removes one token of color b1 from p4. The other two firing modes remove one token of color b2.
- L5 indicates that all firing modes of t3 place one token of color c1 in p5.

2.3.4 Firing Rules

The firing rules for Colored Nets are similar to those of Ordinary Petri Nets. The firing mode C(t)i is *enabled* for the transition t if and only if each input place of t contains at least the colored tokens that are indicated by the i-th column of I(p,t). An enabled transition can *fire* if at least one of its firing modes is enabled. If the transition t fires according to the firing mode C(t)i, the colored tokens that are indicated by the i-th column of I(p,t) are removed from each input place p. For each output place p', colored tokens that correspond to the i-th column of O(p',t) are added to the place p'.

One transition may fire concurrently according to several modes, i.e. a process may be able to handle tasks of different natures at the same time. Within each category, i.e. given one firing mode, several tasks of the same nature may be processed concurrently. The *firing pattern* of a transition in a Colored Petri Net is thus given by a $n_t \times 1$ vector F_t, where $[F_t]_i$ describes the number of concurrent activations of the i-th firing mode. If the firing pattern of the transition is F_t then the combination of colored tokens indicated by the i-th column of I(p, t) is removed $[F_t]_i$ times from each input place p. Similarly, the combination of colored tokens indicated by the i-th column of I(p', t) is added $[F_t]_i$ times to every output place p'.

Example 2.12:

In the CPN of Fig 2.12, only the first mode of t1 and the second mode of t2 are enabled:

- The input place of t1, p1, contains only one token of color a1. L1, the annotation of the adjacent arc is

$$L1 = \begin{array}{cc} 1 & 2 \\ \left[\begin{array}{cc} 1 & 1 \\ 0 & 1 \end{array}\right] & \begin{array}{c} a1 \\ a2 \end{array} \end{array}$$

  The firing mode 1 removes one token of color a1. The initial marking of p1 enables this firing mode. This is not the case for firing mode 2, because p1 does not contain a token of color a2. Finally, as p1 contains one and only one token of color a1, t1 can process at most one task a1, and $F(t)_1$ is either 0 or 1.

- The input place of t2, p3, contains only one token of color b2. L2, the annotation of the adjacent arc is

$$L2 = \begin{array}{cc} 1 & 2 \\ \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] & \begin{array}{c} b1 \\ b2 \end{array} \end{array}$$

  Firing mode 1 removes one token of color b1. The marking of p3 does not enable this firing mode. This is not the case for firing mode 2, because p3 does contain a token of color b2. Finally, as p3 contains one and only one token of color b2, t2 can process at most one task b2, and $F(t)_2$ is either 0 or 1

If t1 fires according to the firing mode 1, and t2 according to its firing mode 2, the marking of the net is changed into the marking of Figure 2.14.

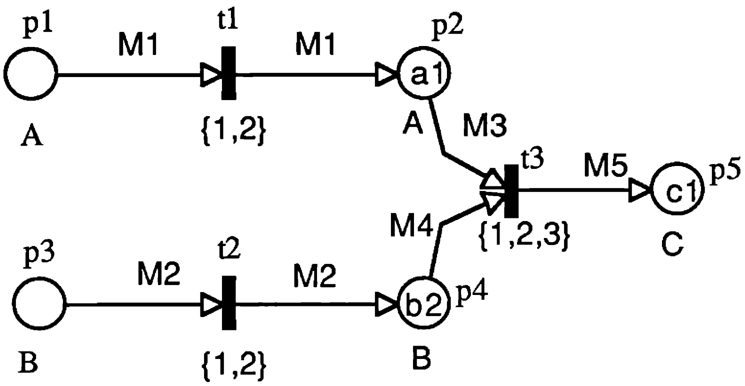


Fig. 2.14 Colored Petri Net after Firing

2.3.5 Incidence Matrix

A Colored Petri Net with n places and m transitions can be represented by a n × m block matrix C: the *Incidence Matrix.* The entries of the matrix are themselves matrices.

The rows correspond to places, the columns correspond to transitions.

- $C_{i,j}$ = *O(t,p)* if there is a directed arc from the j-th transition to the i-th place. O(t,p) indicates the colored tokens that can be added, as determined by the firing modes.
- $C_{i,j}$ = *-I(t,p)* if there is a directed arc from the i-th place to the j-th transition. -I(t,p) indicates the colored tokens that can be removed, as determined by the firing modes.
- *Ci,j = Null matrix* if there are no arcs.

The marking of a net can be represented by a n×1 block vector *M*, where M_i = M(pi). The i-th entry corresponds to the colored tokens in the i-th place.The i-th entry is thus a column with one row for each element of C(pi).

The firing pattern of the net can be represented by a m×1 firing vector *F*, where F_j, the entry that corresponds to the j-th transition, is itself a vector with π_{tj} entries, i.e. as many rows as firing modes of tj.

Given an incidence matrix C, an initial marking M, and a firing pattern F the new marking is

$$M' = M + C * F \qquad (2.5)$$

Example 2.13:
The Colored Petri Net of Fig. 2.12 has the following Incidence matrix:

$$C = \begin{bmatrix} -L1 & 0 & 0 \\ L1 & 0 & -L3 \\ 0 & -L2 & 0 \\ 0 & L2 & -L4 \\ 0 & 0 & L5 \end{bmatrix}$$

The initial marking is given by:

$$M = \begin{bmatrix} M(p1) \\ M(p2) \\ M(p3) \\ M(p4) \\ M(p5) \end{bmatrix}, \text{ with } M(p1) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, M(p3) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, M(p2) = M(p4) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, M(p5) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

The place p1 contains only one token of color a1, p3 contains only one token of color b2, p2 and p4 contain no tokens, and p5 contains only one token of color c1. Finally, the firing pattern of Fig. 2.13 corresponds to

$$F = \begin{bmatrix} F1 \\ F2 \\ F3 \end{bmatrix}, \text{ with } F1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, F2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, F3 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

This chapter introduced basic concepts of Ordinary Petri Net theory and Colored Petri Net theory. Only the notions that are of relevance in the subsequent chapters have been defined, and the reader may find more material in the references given in the introduction of this chapter. Next, this thesis turns to review of equivalence relations and partial orderings.

CHAPTER III

EQUIVALENCE RELATIONS AND LATTICES

This chapter gives basic results about equivalence relations and lattices. The fundamental theorem of section 3.2 about equivalence relations is used in Chapter VII to characterize variable structures. Lattice theory is used extensively in Chapters IV, VI, and VII to address the generation of structures. Complementary material on lattices can be found in Birkhoff (1948) and Grätzer (1971). Relationships between lattices and graphs are explained in Carré (1979).

3.1 DEFINITIONS.

Definition 3.1

A relation R on a set A is called a *binary* relation if and only if

$\forall\ (x,y) \in A^2$ the condition x R y either does or does not hold.

In other words, for each (x, y) "x R y" is meaningful, being either true or false.

Example 3.1: Let A be the set of graduate students at MIT, and R be the relation "is in the same department." R is a binary relation.

Definition 3.2

A relation R on a set A is an *equivalence relation* if and only if

- R is *reflexive*: $\forall\ x \in A\ \ x R x$.
- R is *symmetric*: $\forall\ (x,y) \in A^2\ \ (x R y) \Rightarrow (y R x)$.
- R is *transitive*: $\forall\ (x,y,z) \in A^3\ (x R y)$ and $(y R z) \Rightarrow (x R z)$.

Example 3.2: The relation R defined in Example 3.1 is an equivalence relation.

Definition 3.3

A relation R on a set A is an *ordering*, if and only if

- R is *reflexive*: $\forall\ x \in A\ \ x R x$.

- R is *antisymmetric*: $\forall$ (x,y) $\in A^2$ (x R y) and (y R x) $\Rightarrow$ (x = y).
- R is *transitive*: $\forall$ (x,y,z) $\in A^3$ (x R y) and (y R z) $\Rightarrow$ (x R z).

Example 3.3:

- In {0,1) the relation "is smaller than", denoted by $\leq$, is an ordering.
- Let S be the set of vectors with three entries in {0,1}: $X=[x_1, x_2, x_3]$ x_1, x_2, x_3 in {0,1}. Define on S the relation « :

  X « Y, $X = [x_1, x_2, x_3]$ $Y = [y_1, y_2, y_3]$, if and only if

  $x_1 \leq y_1$ $x_2 \leq y_2$ $x_3 \leq y_3$.

  It is easy to conclude that « is an ordering of S.
- A last classical example is the relation "is included in" among sets, which is an ordering. In this thesis, the ordering "is included in" is used essentially to order the elements of the set P(X), the set that contains all the subsets of a given set X.

Definition 3.4

An ordering R of a set A is a *total ordering* if and only if

Given any (x, y) $\in A^2$ either x R y or y R x.

If an ordering is not a total ordering, it is called a *partial* ordering.

Example 3.4:

- The set of real numbers is totally ordered by the binary relation "is smaller than" ($\leq$).
- The set S of example 3.2 is not totally ordered by «. Neither [1, 0, 0] « [0, 1, 0] nor [0, 1, 0] « [1, 0, 0] are true.
- The set P(X) is not totally ordered by the ordering "is included in".

3.2 EQUIVALENCE RELATIONS

There is an important connection between equivalence relations on a set A and partitions of A into disjoint subsets, as expressed in the following theorem (Carré, 1979).

Theorem 3.1

Every equivalence relation on a set A induces a partition of A into disjoint subsets, called equivalence classes. Conversely, given any partition of A into disjoint subsets $P_1,..., P_k$, there exists an equivalence relation whose equivalence classes are exactly $P_1,..., P_k$.

Proof:

1) Let R be an equivalence relation on A. For each element x let us define the set $E_{R,x} = \{y \text{ in } A \text{ s.t. } x \mathrel{R} y\}$. Note that R reflexive implies that for every x, $x \in E_{R,x}$. For each pair $(x, y) \in A^2$, the equivalence classes are either equal or disjoint with

$$E_{R,x} = E_{R,y} \qquad \text{if } x \mathrel{R} y.$$
$$E_{R,x} \cap E_{R,y} = \emptyset \qquad \text{if } x \mathrel{R} y \text{ does not hold.}$$

This is easily seen from the definition of an equivalence relation. Since every element of A belongs to some equivalence class, and distinct equivalence classes are disjoint, the set of all equivalence classes is a partition of A induced by R.

2) Conversely, given any partition P of a set A into disjoint subsets $P_1, ..., P_k$, let us define the relation R on A by:

$$x \mathrel{R} y \qquad \text{if and only if} \qquad x \text{ and } y \text{ belong to the same } P_i.$$

One can easily be convinced that this is an equivalence relation, and that the equivalence classes with respect to R are exactly $P_1, ..., P_k$.

3.3 ORDERINGS

An ordered set can be depicted very conveniently by a diagram, called the *Hasse diagram.* In this diagram, each element is represented by a point, so placed that if x R y then the point representing x lies below the point representing y. Lines are drawn between two points x and y if and only if y *covers* x, i.e., x R y but there is no element z, z ≠ x,y such that x R z R y. Figure 3.1 shows the Hasse diagram of the set S described in Example 3.3.

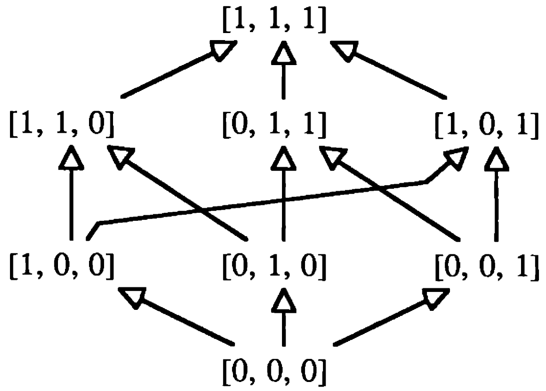


Fig. 3.1 Hasse Diagram

If a set is ordered, totally or partially, some elements with remarkable properties must be distinguished. In the next paragraphs, some of these elements are defined.

Definition 3.5

Let R be an ordering of A.

- If A contains an element ω such that $\omega R x$ for all x in A, then ω is unique, and is called the *least element* of A.
- If A contains an element Ω such that $x R \Omega$ for all x in A, then Ω is unique, and is called the *greatest element* of A.

Remark: These elements do not always exist. For example, in S, these elements exist. The least element is [0, 0, 0], and the greatest element is [1, 1, 1]. However, if « is restricted to the subset of s S - {[0, 0, 0], [1, 1, 1]}, it is impossible to find a greatest and a least element.

Definition 3.6

- An element m of A is a *minimal element* if there does not exist any element in A that is strictly inferior to m:

$$x \leq m \text{ implies } x = m.$$

- An element M of A is a *maximal element* if there does not exist any element in A that is strictly superior to M:

$$M \leq x \text{ implies } x = M.$$

Theorem 3.2 (Birkhoff, 1948)

Every finite ordered set A has at least one minimal and one maximal element.

Proof: Let the elements of A be $x_1,..., x_n$. Define the finite sequences m_k and M_k by:
$m_1 = M_1 = x_1$
$m_k = x_k$ if $x_k \leq m_k$ and $m_k = m_{k-1}$ otherwise
$M_k = x_k$ if $M_k \leq x_k$ and $M_k = M_{k-1}$ otherwise.
Then m_n is by construction a minimal element of A, and M_n is by construction a maximal element.

Example 3.6: In S - {[0, 0, 0], [1, 1, 1]} we have three minimal elements, and three maximal elements.
The minimal elements are [1, 0, 0], [0, 1, 0], [0, 0, 1].
The maximal elements are [1, 1, 0], [1, 0, 1], [0, 1, 1].

3.3 LATTICES

If the set is totally ordered, its structure is particularly simple. In most cases however, an ordering is not total. In order to gain some insights into the structure of the set, some new concepts are needed. For that purpose the notion of lattice is introduced, which is based on local properties of the set.

Definition 3.7

Let B be a subset of a partially ordered set A.

- An *upper bound* of B is an element of A such that $y \leq a$ for all y in B.
  The *least upper bound* (l.u.b) of B, if it exists, is the least element of the set of all upper bounds of B.
- By analogy, the *greatest lower bound* (g.l.b) is the greatest element, if it exists, of the set of all lower bounds of B.

Example 3.7: Let B be {[0, 0, 0], [1, 0, 0], [0, 1, 0]}. B is a subset of S.
It has a l.u.b, which happens to belong to B: [0, 0, 0].
It has a g.l.b, which does not belong to B: [1, 1, 0].

Definition 3.8

A *lattice* is a partially ordered set L in which any two elements x,y have

- a g.l.b or *meet* (denoted by $x \cap y$) that belongs to L.
- a l.u.b or *join* (denoted by $x \cup y$) that belongs to L.

By extension, L' is a *sublattice* of a lattice L if and only if L' is a subset of L such that the join and meet of any two elements of L' are in L'.

Figure 3.2 illustrates the local condition. Given two elements x and y, there exists only one element in the Hasse diagram, the join, which covers both x and y. Similarly, there is only one element, the meet, which is simultaneously covered by both x and y.

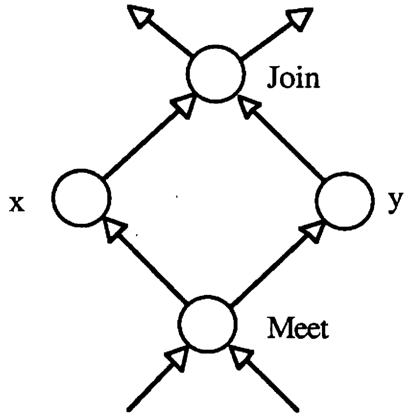


Fig. 3.2 Local Condition

The lattice property ensures that the set L has some structuring patterns. It is possible to identify, for any two elements x and y, two unique boundaries in L, the join and the meet. Every element that is below x and y must be below the join. Any element that is above x and y must be above the meet.

Remark: Every Lattice L has a least and a greatest element. The least element is the join, which belongs to L, of all the elements of L. The greatest element is the meet of all the elements of L. This meet belongs to L, by definition of the lattice.

Example 3.8:

- S is a lattice.

  On {0, 1} the join and meet operators are defined as:

  $0 \cap 0 = 0 \qquad 0 \cup 0 = 0$

  $0 \cap 1 = 0 \qquad 0 \cup 1 = 1$

  $1 \cap 0 = 0 \qquad 1 \cup 0 = 1$

  $1 \cap 1 = 1 \qquad 1 \cup 1 = 1.$

  The operators are extended on a component-wise basis:

  $X \cap Y = [x_1 \cap y_1,\ x_2 \cap y_2,\ x_3 \cap y_3].$

  $X \cup Y = [x_1 \cup y_1,\ x_2 \cup y_2,\ x_3 \cup y_3].$

- Let X be a set, and P(X) be the set of all subsets of X. (P(X) is a lattice with the partial ordering "is included in".
  The meet of two subsets of X, A and B, is the intersection $A \cap B$.
  The join of two subsets of X, A and B, is the intersection $A \cup B$.

Definition 3.9

If $x_1, x_2, ..., x_n$ are n elements of a lattice L, the sublattice generated by the n elements $x_1, x_2,..., x_n$ in performing join and meet operations is called the *lattice polynomial*, and is denoted by $L(x_1, x_2,...,x_n)$.

This last notion plays an important role in Chapter VII for a constructive characterization of variable structure systems. After this brief review of orderings and equivalence relations, the next chapter, Chapter IV, introduces the reader to the fundamentals of the theory of Distributed Intelligence Systems. Some of the results of this chapter are used in Chapter IV, while most of them recur throughout this thesis.

CHAPTER IV

DISTRIBUTED INTELLIGENCE SYSTEMS

This chapter is an introduction to the methodology developed in this thesis. Section 4.1 provides basic concepts of distributed intelligence systems. Section 4.2 restricts the scope of the thesis to the modeling and the design of a particular class of systems, called variable structure systems. Each system in that class is characterized by several parameters that are described in section 4.3. A unified framework to represent variable structure systems is defined in Section 4.4. Finally, sections 4.4 and 4.5 provide notions and results to study variable structures systems. The relationships between variable structure systems and Colored Petri Nets are described in Chapter V.

4.1 BASIC CONCEPTS

This thesis models a class of distributed intelligence systems, teams of decisionmakers (Grevet et al., 1988; Levis, 1988). A *team* is defined as a number of persons working together to accomplish a task. The team members have a common goal. They share the same interests and beliefs. Their skills must be coordinated under well-specified rules of operation so as to achieve high efficiency. The human decisionmakers are assumed to be well trained, to execute well defined tasks, and to be constrained by bounded rationality (Boettcher and Levis, 1982). The latter notion refers to a limited ability of human beings to process information. While performing his tasks, the decisionmaker may decide to use a decision aid (Perdu et al., 1988).

The *objects* are the components necessary to carry out the task. They are physical, technological, and human components, which receive, process, generate, and transmit information. They include the members of the team, communication devices, computers, software, decision-aids, etc. From the organizational perspective, the objects have synergistic activities. The individual processes result from partitioning the organizational task into subtasks. It is necessary to combine the results of the different processes to perform the mission.

The *structure* is the description of the relationships between objects, as determined by the objectives of the mission. These relationships can be considered at different levels: they may describe the physical position of such elements (location, layout in a room); they may describe the communication links, or the rules and protocols that trigger sending of information through these links; they may describe the way tasks performed by the objects are coordinated.

The objects, the structure, and the interface with the rest of the universe form the *system.* The system can be thought as an entity because of these relationships. The system is included in an environment, which in turn is part of a context.

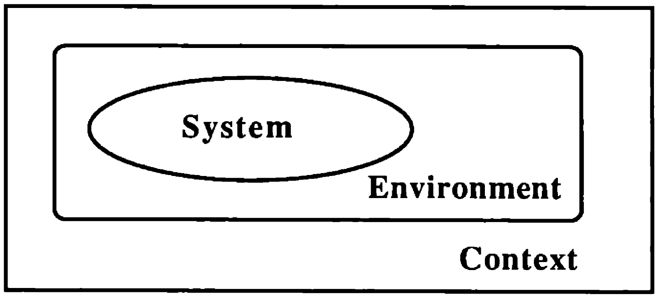


Fig. 4.1 The System and the Rest of the Universe

The *Environment* is the part of the universe on which the system has some ability to act. The system can sense the environment, and acts upon it, if necessary. Reciprocally, the environment can act upon the system. The *Context* denotes the rest of the Universe: the set of conditions and assumptions within which system and environment exist. The context has some influence on the system, yet the system cannot act on it.

In general, the mission is to achieve a particular (desired) state of the environment, given a specific context. For that purpose, the system senses the environment with *sensors*. The inputs to the system are the observations or messages carried out by the sensors. These items of information are transmitted to their proper destinations within the organization. They are analyzed, and the selected response is finally implemented by the *effectors.*

4.2 VARIABLE STRUCTURES

This thesis focuses on one body of research about distributed intelligence systems, the modeling and generation of structures, as opposed to the design of efficient objects, accurate sensors, responsive effectors etc. There are two main reasons for this choice.

- One reason has to do with current engineering practice. Systems are being developed without much consideration of the impact of structure on performance. The requirements for a system are translated into a natural structure. Typically, only small variations around that basic structure only investigated. Most of the effort is concentrated on designing efficient objects. Over the last decade, the engineering community has realized that considerable improvement in performance could be achieved by optimizing over structure. There is thus a need to investigate the whole set of structures that makes sense given the requirements for a system.
- The second reason is that little is known about the generation of structures. The generic problem of investigating the whole set of structures is computationally too demanding, due to its combinatorial nature. There is thus a need to highlight some properties of this set in order to gain some understanding of it.

4.2.1 Variability

The structure of a system can exhibit some properties that are defined below. It must be noted that throughout this thesis, the emphasis is on describing how the structure is influenced by the inputs, as opposed to a description of the information content of the interaction.

Definition 4.1

A structure is *fixed* if the relationships between objects remain constant, whatever the mission, the state of the environment, or the context. Reciprocally, a structure is *variable* if the interactions between objects can vary.

Of course, variable structure systems are far more common than fixed structure systems. However, a well developed body of theory has been developed only for the latter. This thesis makes an effort to extend results that have been obtained on fixed structures systems to the much larger class of variable structure systems. It is convenient to distinguish three mechanisms that trigger variability (Monguillet, 1988). Each mechanism focuses on one aspect of variable structure that is of practical relevance. A system is said to be

- *Type 1-variable,* if it adapts its structure to the *input it processes.* For example, one pattern of interactions may yield a higher efficiency than others for a given input.

- *Type 2-variable,* if it adapts its structure of interactions to the *environment.* The performance of a given structure may depend strongly on the state of the environment. A structure may accomplish its mission perfectly if the arrival rate of the inputs is low. This structure may not be the optimal one if the arrival rate is high. There may be a need to change the relationships between objects.
- *Type 3-variable,* if it adapts its structure of interactions to the *system's parameters.* In case of failure or destruction of some components, for example, it might not be possible to accomplish the mission with the current structure. The system has to be reconfigured.

Of course, a particular system may exhibit simultaneously the three types of variability; the Airport Surface Traffic Control system (ASTC) is an example. This distributed intelligence system is treated in more depth in Chapter VIII. It monitors the landings and the takeoffs as well as the movements of planes in an airport and in its immediate vicinity. The interactions between air traffic controllers are Type 1-variable if they change depending on the planes, their destination. The interactions are Type 2-variable if they change depending on the weather conditions. The interactions between controllers are Type 3-variable if they change when some radar or some communication networks do not function properly. In order to limit the scope of the thesis, assumption 4.1 is made

Assumption 4.1

This thesis models structures whose variability is triggered by parameters that are *external* to the system, can be *sensed* by some processes, and can be *communicated* to the objects.

The model obviously encompasses Type 1 variability because the sensors communicate observations to some objects. The model addresses also Type 2 variability, if the parameters that characterize the state of the environment can be assessed by some objects from a source of information. The weather condition is an example of the latter type in the ASTC system. Even though the Control Tower does not have a sensor that describes explicitly the condition on the field, the controllers can assess it by looking outside. In the rest, the term variable is restricted to mean variability as defined in this paragraph, the term inputs is used to mean the observations of the sensors as well as the parameters of the environment, and the expression variable structure stands as a short hand for variable structure systems.

4.2.2 Functional Approach

The modeling approach follows the formulation (Levis, 1988) of the research issues for distributed intelligence systems within the framework advocated by Minsky (1986). In this approach, the team is seen as an information processing system that must perform several functions to accomplish its mission. The functions are spread out over a range of locations and divided into individual tasks, so that each object's activity contributes a little to each of the several functions. The individual tasks can be performed both by humans and by intelligent computerized nodes, i.e. by an intelligent decisionmaking process. Each individual task is itself a combination of well defined algorithms that describe the manipulation to be done to their inputs. Two individual tasks are said to be equivalent if and only if they incorporate the same algorithms.

An individual task is thus a *role* : a prescribed pattern of behavior, as determined by the mission. The same role can be performed by several decisionmakers, provided that they have received adequate training. Similarly, one decisionmaker can perform several roles, depending on his experience and training. Finally, an intelligent node can perform several roles, according to the knowledge and resources that have been embedded in it.

The emphasis is placed on describing the roles of the system. The components of the system, human and physical, are treated as resources. This approach leads to the distinction of two mechanisms:

Functional Flexibility. If the interactions between roles can vary, the system is *functionally flexible*. This flexibility induces varying interactions between the physical elements *once* they have been assigned to the roles.

Resource Flexibility. If the assignment of resources can vary, there is flexibility in the utilization of resources. This flexibility induces changing patterns of interactions between physical elements *because* they are being assigned.

Of course, one organization can have both mechanisms simultaneously. Suppose that the system is a team engaged in an automobile competition. There are two roles, the role of the pilot and the role of the copilot, and two resources, human beings A and B. There is flexibility in the assignment of the resources if A and B can perform both roles: A can pilot and B can copilot, or B can pilot and A can copilot. There is functional flexibility in the system if the copilot is subordinated to the pilot under normal racing conditions, and if the copilot takes the lead if an error of navigation has been made. A structure is thus characterized by two substructures.

- *The functional structure.* This is the description of the interactions between roles, and between roles and the external environment through sensors and effectors. It depicts the flow of data from the sources to the roles, the exchange of information between roles, and the communication of messages to the effectors. Functional flexibility corresponds to varying exchanges of information within that structure, which is also called the Data Flow structure.
- *The resource structure.* This is the description of how resources can be assigned to roles. Resource flexibility corresponds to different interactions in the resource structure.

4.2.3 Consistency and Determinism.

As defined in section 4.2.2, a variable structure system adapts its interactions to the inputs it processes. Further properties of the variability modeled in this thesis are defined in this section.

Definition 4.3

A system is *temporally consistent* if it processes only observations that refer to the same temporal origin, i.e., to an event with a specific time of occurrence (Grevet, 1988).

Many systems have this property. For example, the ASTC system monitors in real time the movements of planes. To add another example in a very different context, the automated trading centers of large brokerage firms are distributed intelligence systems that monitor "on-line" the conditions of various financial markets.

Assumption 4.2

The thesis models only variable structures that are temporally consistent.

It must be noted that this assumption does not imply anything about the nature of the processing, which may be globally synchronized, locally synchronized, or totally asynchronous (Bertsekas and Tsitsiklis, 1988). This assumption is made because most of the systems for which this methodology is developed have this property. The Air Traffic Control system or any Command and Control system work under relatively high tempos of input arrivals, and monitor the environment "on line".

Definition 4.4

A variable structure is said to be *deterministic* if and only if the processing of one set of simultaneous observations is achieved while involving a unique set of interactions.

Assumption 4.3

This thesis models only variable structures whose *functional structures* are deterministic.

From the functional approach taken by this thesis, it seems reasonable to assume that the systems under study verify assumption 4.3, which proceeds from several rationales. The first rationale is that a role corresponds to a set of algorithms. Each algorithm describes a manipulation of its inputs according to some rules of first order logic. It therefore produces for each set of inputs to the algorithm one and only one output, and interacts with one and only one pattern. The second rationale is that the activities of the roles are assumed to be coordinated, so as to produce a unique response, that of the structure, for each set of simultaneous inputs to the system. Each set of simultaneous observations must thus be processed according to one and only one global pattern of interactions. Finally, because this separation between functional and resource structures permits modeling not only of structures in which the interactions between objects are deterministic, but also of structures with a deterministic functionality and some randomness in the assignment of resources to roles. In the rest of the thesis, the term variable structure is used as a short hand for a variable structure that is consistent and functionally deterministic. The next section provides a mathematical model of the terms that have been described so far.

4.3 MATHEMATICAL MODEL

4.3.1 Sensors

A variable structure processes data from N sources of information, i.e. sensors. Each Sensor n, n = [1,..N], can output one letter from its associated alphabet $Xn = \{xn_1, ..., xn_{|Xn|}\}$. These alphabets describe the basic items of information, also called colors or attributes, which are communicated to the system. The alphabets include the null element, i.e., the case where no item of information is transmitted. It is assumed that each source is statistically independent of the others. This is accomplished by introducing supersources, which aggregate the observations of sensors that are statistically correlated (see Stabile and Levis, 1984).

As shown in Fig. 4.2, each independent Sensor n, n = 1..N, is modeled by a place annotated by Xn. A transition whose unique input place is Sensor i models the communication of the sensor's observations. The temporal consistency of the observations is modeled by the fact that all sensors are the output of a single process. This process has a single input place p0, which is called the external place.

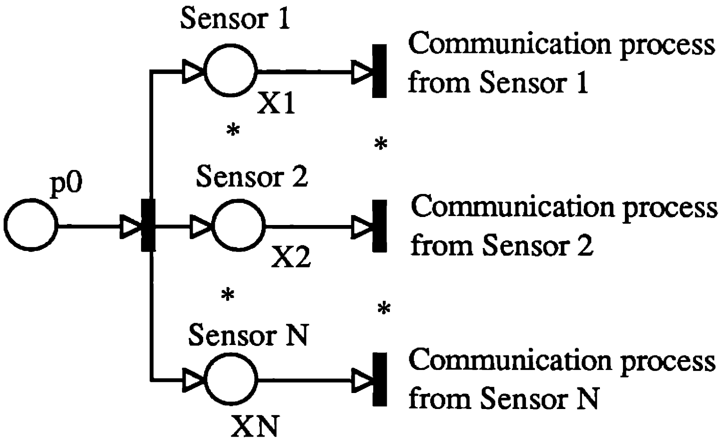

Sensor 1
Communication process from Sensor 1
X1
*
*
p0
Sensor 2
Communication process from Sensor 2
X2
*
*
Sensor N
Communication process from Sensor N
XN

Fig. 4.2 Sensors

From the system point of view, temporal consistency implies that the input is a N-dimensional vector: $x = (x_1, x_2, ..., x_N)$. This vector has as components the N independent observations and belongs to the alphabet X, cross-product of the source alphabets:

$$X = X_1 \times X_2 \times ... \times X_N.$$

4.3.2 Four Stage Model

In the Petri Net formalism, each role in a fixed structure has been modeled by a subnet with four transitions and three internal places, presented in Figure 4.3. As explained in Chapter II, places are depicted by circles and stand for resources or messages. Transitions between places are depicted by bars and stand for algorithms. The arcs denote the precedence relations between these algorithms. The four stage decisionmaking process consists of four algorithms SA, IF, CI, and RS. This model shows explicitly the different kinds of interactions that can exist between roles, and between a role and the environment.

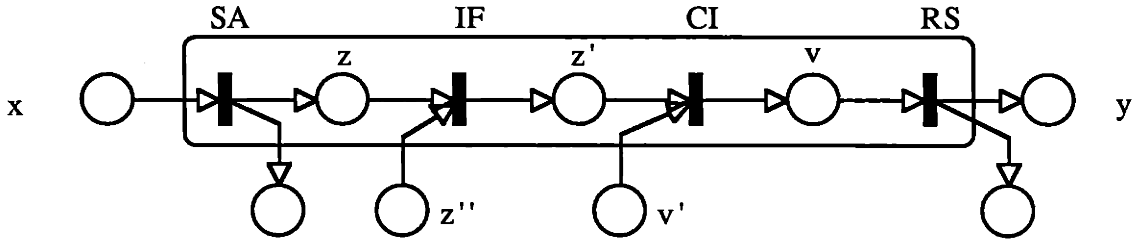


Fig. 4.3 Four Stage Model of a Decisionmaker or Intelligent Node

In Figure 4.3, x represents an input signal from an external source of information or from the rest of the organization, i.e. from another role. The *Situation Assessment (SA)* algorithm processes the incoming signal to obtain an assessment of the situation. The assessed situation z may be transmitted to other decisionmaking processes. In order to provide some insights into these definitions, let us consider the case of an air traffic controller. Let us suppose that he wants to direct a plane that has just entered the controller's jurisdiction. An air traffic controller receives information from a radar screen and over radio networks. At the SA stage, he surveys the traffic by observing radar screens and by listening to radio communications.

Concurrently, the role may incorporate one or several signals z" from other parts of the system. The signals z and z" are merged together in the *Information Fusion* stage *(IF)* to produce the final situation assessment z'. At this stage, a controller can incorporate information about traffic density in other jurisdictions. The next algorithm, the *Command Interpretation* algorithm *(CI)* receives and interprets possible commands (v') from other roles, which restrict the set of responses that can be generated. At the CI stage, an air traffic controller can receive some instructions about the course of action to be taken, for example, if some planes have suddenly priority over others (Case of emergency). The CI stage outputs a command v which is used in the *Response Selection* algorithm *(RS)* to produce the response of the role, the output y. This output can be sent to the effectors and/or to other roles in the system. An air traffic controller sends a pilot directions, operating procedures, etc. This information can also be sent to other controllers.

Of course, this model does not capture all the subtleties of an intelligent decisionmaking process. However, it has proven its usefulness by showing explicitly different types of interactions between decisionmaking processes. Furthermore, it is broad enough to tackle realistic examples within acceptable computational requirements. This model is easily extended to a four stage decisionmaking process that is variable by putting switches instead of transitions or by modeling varying patterns of interactions with a Colored Petri Net, as described in Chapter V. Three configurations are allowed for the structure of a role :

- SA, IF, CI, RS.
- IF, CI and RS.
- CI and RS.

These assumptions (Remy, 1986) are based on the idea that a role need not have all four stages. If two given stages are present however, any intermediate stage must also be present. In the model, the first stage is SA, IF or CI, which are the stages at which the role may receive external inputs. The last stage must be RS, the stage in which the role selects its response. Definition 4.5 characterizes roles that are equivalent in a functional sense.

Definition 4.5

Two roles are *equivalent* if and only if they have the same configuration and the same algorithms SA, IF, CI and RS.

Example 4.1

Consider the roles in Figures 4.4 and 4.5. Let two IF algorithms IF 1 and IF 2 be different, and the algorithms for the other stages be identical. Then the roles are different.

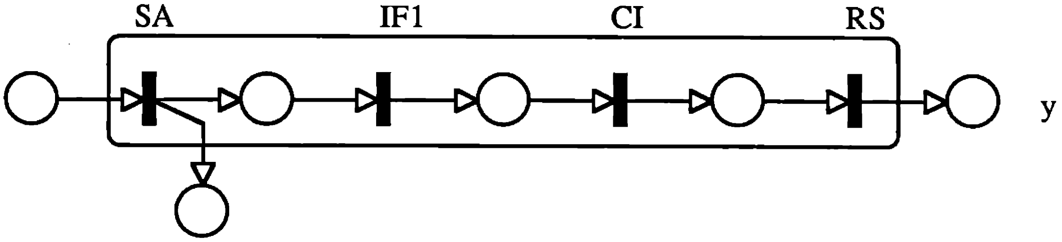


Fig. 4.4 Role 1

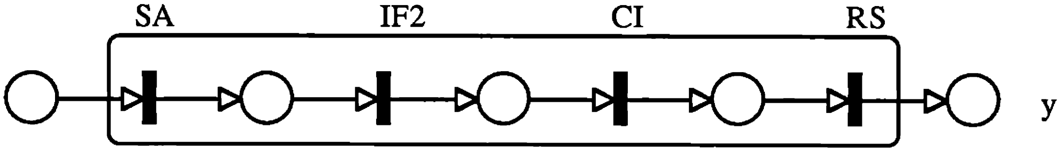


Fig. 4.5 Role 2

4.3.3 Interactions between Roles

To be consistent with the interpretation of the different stages, only certain types of interactions are allowed. The interactions between two roles, i and j, are the ones suggested for interactions between decisionmakers in Remy et al. (1988). They are presented on Fig. 4.6.

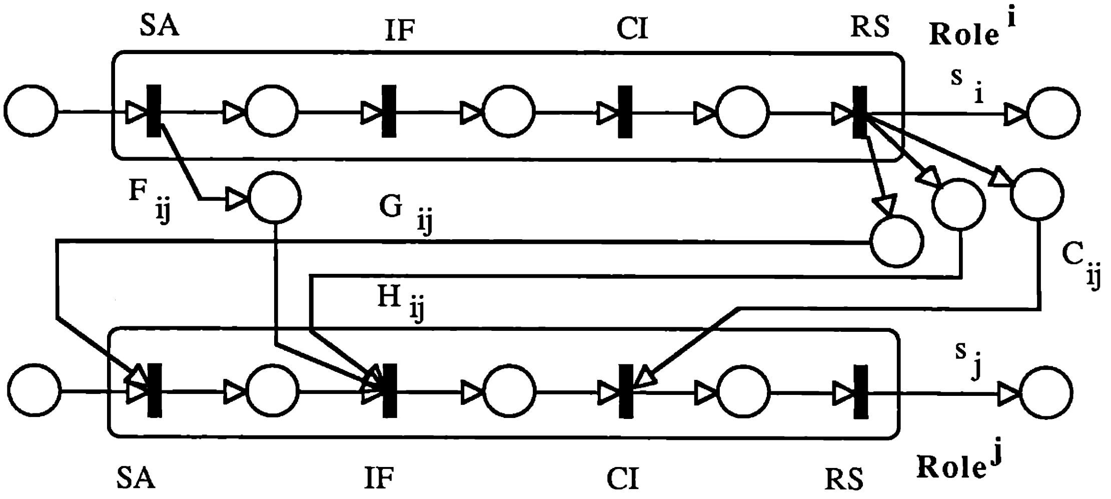


Fig. 4.6 Allowable Interactions between two Roles

Remark: For the sake of clarity, only the links from the i-th role to the j-th role have been represented. The symmetrical links are valid interactions as well.

Physical significance of the allowable interactions:

- RS of i-th role to external environment : s_i.
  The i-th role communicates the response it has selected to the effectors. At the system level, the coordination of the responses is shown on a Petri Net model by adding an output transition, as in Fig. 4.7. If Role i sends its response to the effectors, then there exists a link between the RS stage of Role i and this output transition. This output transition has a unique output place, which is called the sink.

- SA of i-th role to IF of j-th role: F_{ij}.
  The situation assessment which is produced as an output of the SA stage is sent to the j-th role to be fused with the assessment of the j-th role, and/or assessments from other roles.
- RS of i-th role to SA of j-th role: G_{ij}.
  The response selected by the i-th role is the input of the j-th role.
- RS of i-th role to IF of j-th role: H_{ij}.
  This link shows the sharing of a result. The i-th role informs the j-th role of its final decision. The j-th role may or may not take this information into account.
- RS of i-th role to CI of j-th role: C_{ij}.
  This interaction has been introduced to model hierarchy between roles. It describes the possibility of role i sending a command to role j.

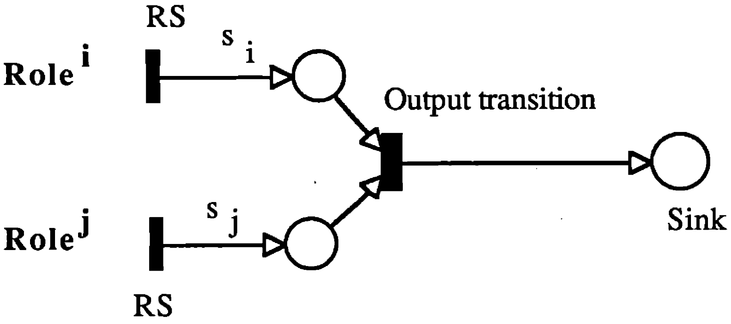


Fig. 4.7 Interaction Role - Sink

4.3.4 Interactions with the Sensors

Every component of the system might not have access to all the sensors' observations. It might base its processing on a restricted number of observations. The communication of the observations from the N sensors are modeled by the processed represented on Figure 4.8.

Physical significance of the links:

S_{ij} for i = 1..N : This link models the fact that the observation from the i-th sensor is communicated to the SA stage of Role j.

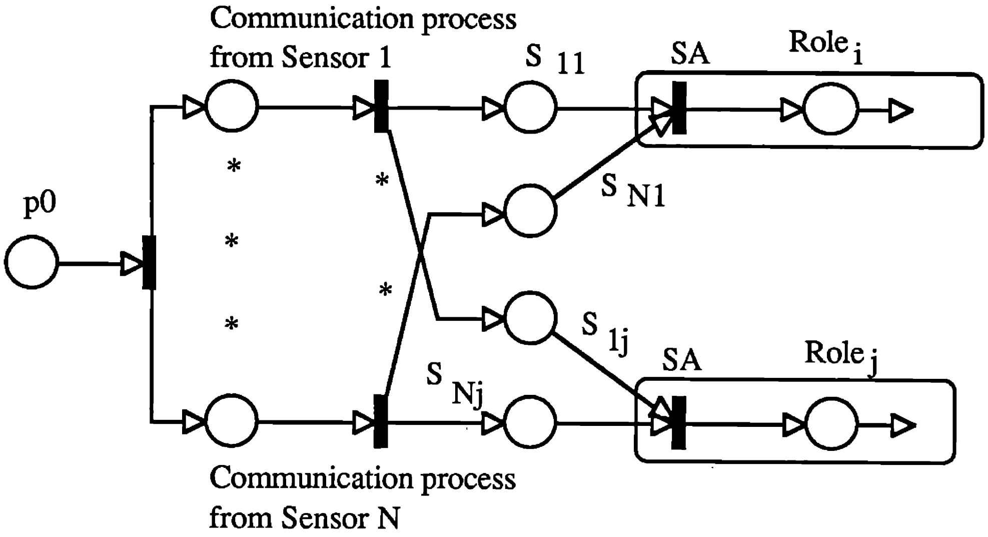


Fig. 4.8 Interaction Role-Sensors

4.3.5 Resource Allocation

A role is performed by a human being with an adequate training or by an intelligent node programmed for that purpose. The resource structure describes how resources can be allocated to roles, and one example such example is depicted on Figure 4.9.

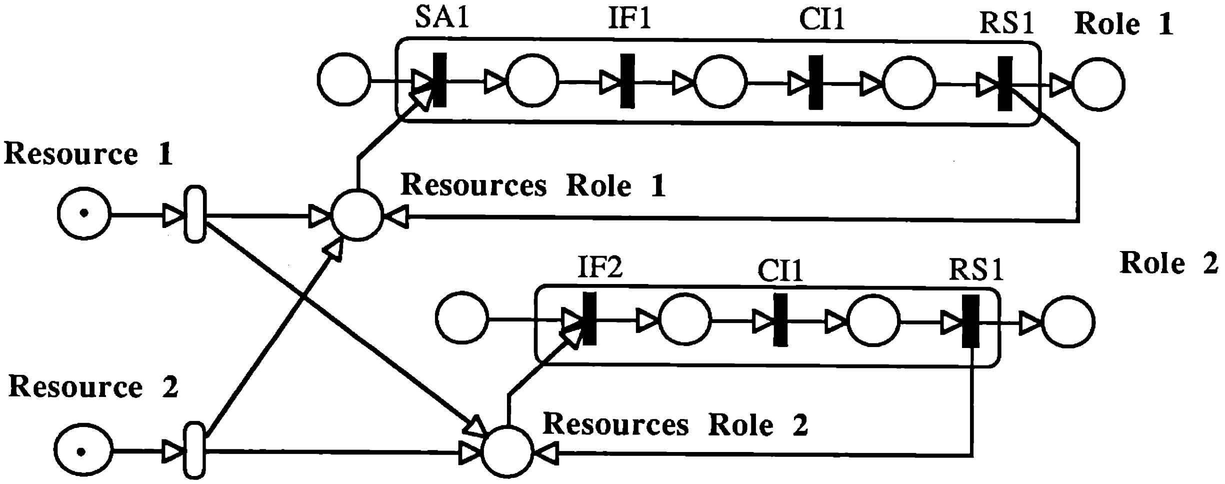


Fig. 4.9 Interactions between two Roles and two Resources

The resource structure is constructed in several steps.

First, one resource place is created for every physical resource. The marking of this place describes whether the resource is available or has been assigned to some role.

Then, a place is attached to each role, whose marking indicates how many of the physical resources that have been assigned to the roles are not processing some input. If a role starts a process, it removes one token from the place that contains the physical resources that have been assigned to the role. A token is put back in that place at the end of the process.

Finally, the assignment of a physical resource to a role is depicted by a switch. If there is one link from the switch to a role, then the resource can be assigned to the role. If there are no links, then the resource cannot be used by that role. The switch corresponds to the fact that a resource can be assigned to one and only one role. On Figure 4.9, for example, both resources can perform Role 1 and Role 2. Both resources are available and unassigned.

4.4 MATRIX REPRESENTATION

Section 4.3 described a model of the components from which a variable structure can be built, and a set of interactions between these components. This section presents a unified framework to describe a system.

4.4.1 Matrix Form

From section 4.3, the model of a structure is characterized by the following parameters

- R roles, which are ordered Role 1..Role R according to some appropriate rule.
- N sensors, which are ordered Sensor 1..Sensor N.
- $X = X_1 \times X_2 \times ... \times X_N$, the set of inputs. X is ordered by the lexicographic ordering.
- D decisionmakers or intelligent nodes, which are ordered Resource 1.. Resource D.

Within the model of section 4.3, the functional structure is described by the interactions F, G, H, C, S, and s. The resource structure is described by the interactions between the resource places and the resource structures of the roles. Below, Proposition 4.1 gives a straightforward characterization of a variable functional structure, while Proposition 4.2 gives a simple description of the resource structure.

Proposition 4.1

A variable functional structure is determined by $\Pi = (S,\ s,\ F,\ G,\ H,\ C)$.

- S is a N × R array. For i = 1,..N and j = 1..R, S_{ij} models the link between the i-th sensor and the SA stage of the j-th role.
- s is a 1 × N array. For i = 1..R, s_i models the link between the RS stage of the i-th stage and the effectors.
- $F,\ G,\ H,\ C$ are four R × R arrays. For i = 1..R and j = 1..R,

  F_{ij} models the link from the SA stage of Role i to the IF stage of Role j.

  G_{ij} models the link from the RS stage of Role i to the SA stage of Role j.

  H_{ij} models the link from the RS stage of Role i to the IF stage of Role j.

  Cij models the link from the RS stage of Role i to the CI stage of Role j.
- Every entry in S, s, F, G, H, C is a |X| × |X| diagonal matrix L.

  $L_{ii} = 1$ if the i-th input in the lexicographic ordering activates the link.

  $L_{ii} = 0$ if the i-th input in the lexicographic ordering does not activate the link.

Proof:

The matrices S, s, F, G, H, C describe uniquely all the interactions that can be found in the functional structure.

The functional structure is deterministic. The |X| × |X| matrix attached to each interaction keeps track of the fact that each set of inputs, i.e. each x in X, either activates or does not activate the interaction.

Example 4.2

Consider a system with two roles, Role 1 and Role 2 and two sensors, Sensor 1 and Sensor 2. The output alphabet X1 is {a, b}. The output alphabet X2 is {c, d}. Therefore, X is {a, b}×{c, d} and <a,c> is the first input in the lexicographic ordering, <a, d> is the second, <b, c> is the third and <b, d> is the fourth.

A variable pattern of interaction is given by Π as follows

$$S = \begin{bmatrix} I & I \\ I & I \end{bmatrix}, s = [\,I \quad I\,], F = G = H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} 0 & M \\ 0 & 0 \end{bmatrix}$$

- I is the 4 × 4 identity matrix. It indicates that every input activates this link.
- 0 the 4 × 4 null matrix. It indicates that the link is never activated.

- M is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{array}{l} \langle a, c\rangle \\ \langle a, d\rangle \\ \langle b, c\rangle \\ \langle b, d\rangle \end{array}$$

  M indicates that the link is activated only for the first two inputs: (a,c) and (a,d).

- S shows that the outputs of Sensors 1 and 2 are sent to Roles 1 and 2.
- s indicates that both roles send their response to the effectors.
- F, G, H indicate that the roles do not exchange information between the SA and IF stages, the RS and SA stages, and the RS and IF stages.
- The matrix C indicates that Role 1 issues a command to Role 2 if and only if the observations processed by the organizations are <a, c> and <a, d>.

Similarly, the following proposition characterizes the resource structure.

Proposition 4.2

The resource structure is given by a R × D matrix. The rows stand for the roles and the columns stand for the intelligent resources Resource 1...Resource D.

$RD_{ij} = 1$ if role i can be performed by the intelligent resource j.

$RD_{ij} = 0$ if role i cannot be performed by the intelligent resource j.

Proof

An intelligent resource is either able or unable to perform a particular role. The matrix RD keeps track of it.

Example 4.3: Consider the resource structure of Figure 4.9. Then

$$RD = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

4.4.2 Well Defined Functional Structures

The functional and resource structures play different roles in this thesis. More emphasis is placed on understanding the properties of functional structures than resource structures. There are two main reasons for this approach. The first reason is that distributed systems are generally generated by creating their data flow structures (Andreadakis, 1988), or analyzed using a functional approach (Valraud, 1989). Within the framework of this thesis, the exchange of information between the elementary functions is depicted by the functional structure. The second reason is that separating the functional structure from the resource structure reduces the complexity of the design problem. The designer is prevented from computing architectures that are symmetrical, in the sense that they correspond only to different assignments of resources. In the rest of this section, two sets V and W are defined.

Definition 4.6

A *Well Defined Variable Structure* (WDVS) of dimensions R, the number of roles, N, the number of sensors, X, the set of inputs, is any

$$\Pi = (S, s, F, G, H, C)$$

where

- S is an $N*|X| \times |X|$ array, which is decomposed as a $N \times 1$ block array.
- s is a $R*|X| \times |X|$ array, which is decomposed as a $R \times 1$ block array
- F, G, H, C are four $R*|X| \times R*|X|$ arrays, which are decomposed as $R \times R$ block arrays.
- Each block is a $|X| \times |X|$ diagonal matrix with 0s and 1s on the diagonal.

Let V (R, N, X) be the set of Well Defined Variable Structures of dimensions R, N, X.

Each variable functional structure characterized by R, N, X is an element of V (R, N, X) (Proposition 4.1). It is important to remark that the converse might not be true. Structures that correspond to some elements of V might not make much physical sense.

Below, it is assumed that the parameters, R, N, X are explicitly fixed. To make notation less cumbersome, V(R, N, X) is abbreviated by V. In order to gain some insights into V, it is important to note that a fixed functional structure is a special case of a variable structure. In the case of a fixed structure, the interactions in the system do not depend on the inputs to the system, and a link is either present (and the diagonal entry in the appropriate block array is the |X| × |X| identity matrix) or absent (and the diagonal entry in the appropriate block array is the |X| × |X| null matrix). Proposition 4.2 provides a notation for fixed structures.

Proposition 4.2

A fixed functional structure is given by $\Sigma = (S', s', F', G', H', C')$.

- The N × R matrix S' describes the interaction between sensors and roles; the 1 × R matrix s' describes the interactions between roles and effectors; the R × R matrices F', G', H', C' correspond to the interactions between roles.
- The entry of each matrix belongs to {0, 1}.

  It is 1 if the corresponding link is present in the system.

  It is 0 if the corresponding link is not present in the system.

Example 4.4:

The following matrices describe a fixed structure system.

$$S' = \begin{bmatrix} 1\,0 \\ 0\,1 \end{bmatrix}, \quad s' = \begin{bmatrix} 1, & 1 \end{bmatrix}, \quad F' = G' = H' = \begin{bmatrix} 0\,0 \\ 0\,0 \end{bmatrix}, \quad C' = \begin{bmatrix} 0\,1 \\ 0\,0 \end{bmatrix}$$

This is a functional structure with two roles and two sources.

Role 1 receives the observations from Sensor 1, Role 2 receives the observations from Sensor 2 (Matrix S').

Both roles send their response to the effectors (Matrix s').

Role 1 issues a command to Role 2 (Matrix C'), and there are no other interactions between roles (Matrices F', G', H').

As was done for variable structures, this proposition leads to the following definition.

Definition 4.7

A *Well Defined Fixed Structure* (WDFS) of dimensions R, N is any

$$\Sigma = (S', s', F', G', H', C')$$

where

- S' is an N × 1 array,
- s' is a R × 1 array,
- F', G', H', C' are four R × R arrays.
- Their entries belong to {0,1}.

Let W (R, N) be the set of Well Defined Fixed Structures of dimensions N, R. Each fixed functional structure characterized by R, N is an element of W(R, N) (Proposition 4.2). Note that here again the converse might not be true. Some elements of W(R, N) might not make much physical sense. To make notations less cumbersome, W(R, N) is also abbreviated by W.

4.4.3 Decomposition of Variable Structures

Intuitively, one can think of a variable functional structure as a collection of fixed structures and some rule for assigning a unique fixed structure to each input. This fixed structure describes the exchange of information in the structure while x is being processed. Proposition 4.3 gives a mathematical characterization of this concept within the larger framework of the sets V and W introduced in section 4.3.2.

Proposition 4.3

Any WDFS Π = (S, s, F, G, H, C) in V can be represented as a mapping from the set of inputs X into the set W. Reciprocally, any mapping from the set of inputs X into the set of fixed structures W corresponds to one variable structure in V.

Proof:

For each x, if x is the k(x)-th input in the lexical ordering of X,
let us define Π (x) = (S'(x), s'(x), F'(x), G'(x), H'(x), C'(x))

- S'(x) is an N × R array. For i = 1..N and j = 1..R,

  $S'(x)_{ij} = [S_{ij}]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[S_{ij}]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates S_{ij}.
- s'(x) is an R × 1 array. For i = 1..R

  $s'(x)_i = [s_i]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[s_i]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates s_i.

- $F'(x)$, $G'(x)$, $H'(x)$, $C'(x)$ are four R × R arrays. For i = 1..R and j = 1..R

  $F'(x)_{ij} = [F_{ij}]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[F_{ij}]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates F_{ij}.

  $G'(x)_{ij} = [G_{ij}]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[G_{ij}]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates G_{ij}.

  $H'(x)_{ij} = [H_{ij}]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[H_{ij}]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates H_{ij}.

  $C'(x)_{ij} = [C_{ij}]_{k(x)\,k(x)}$

  where, by proposition 4.1, $[C_{ij}]_{k(x)\,k(x)}$ takes its value in {0, 1} and indicates whether or not the set of observations x activates C_{ij}.
- The entries of each array are thus 0s and 1s. They describe whether or not the corresponding link is present in the structure when the set of inputs is x.

It is easy to see that each $\Pi(x)$ is a fixed structure in W, and that it describes the interactions between roles, and between roles and environment, when the system processes the set of inputs given by $x = <x_1,...,x_n>$. Each well defined structure is thus given by a mapping from X to W that assigns to each x the fixed structure $\Pi(x)$.

Reciprocally, consider a mapping which associates to each x in X

$\Pi(x) = (S'(x), s'(x), F'(x), G'(x), H'(x), C'(x))$ in W, and define

- $\mathcal{S}$ a N × R block array. For i = 1,..N and j = 1..R, S_{ij} is a $|X| \times |X|$ diagonal matrix with $[S_{ij}]_{u\,u} = [S'(x)]_{i\,j}$ and u = rank of x in lexicographic ordering of X
- $\mathcal{s}$ a 1 × N block array. For i = 1..R, s_i is a $|X| \times |X|$ diagonal matrix with $[s_{ij}]_{u\,u} = [s'(x)]_{i\,j}$ and u = rank of x in lexicographic ordering of X.
- $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}$, $\mathcal{C}$ four R × R block arrays. For i = 1..R and j = 1..R,

  F_{ij} is a $|X| \times |X|$ diagonal matrix with

  $[F_{ij}]_{u\,u} = [F'(x)]_{i\,j}$

  and u = rank of x in lexicographic ordering of X.

  G_{ij} is a $|X| \times |X|$ diagonal matrix with

  $[G_{ij}]_{u\,u} = [G'(x)]_{ij}$

  and u = rank of x in lexicographic ordering of X

H_{ij} is a $|X| \times |X|$ diagonal matrix with

$[H_{ij}]_{u\,u} = [H'(x)]_{ij}$

and u = rank of x in lexicographic ordering of X.

C_{ij} is a $|X| \times |X|$ diagonal matrix with

$[C_{ij}]_{u\,u} = [C'(x)]_{i\,j}$

and u = rank of x in lexicographic ordering of X.

Π is by construction a WDVS such that if one link is present in $\Pi(x)$, then this link is activated by x in Π. Thus Π represents a variable structure for which each $\Pi(x)$ describes the interactions in the system when it processes the set of observations x.

Proposition 4.3 states therefore that it is equivalent to work in the set V or to work in the set of all assignments from X to W. In the rest of this thesis an effort is made to use the framework that is most convenient for each problem. For any WDVS, the notation Π is used either to mean the set of arrays S, s, F, G, H, C or the mapping that associates to each x a WDFS. From the characterization of Π as a mapping, one can easily understand Definition 4.8.

Definition 4.8

Let Π be a WDVS; then the range of the mapping in W is called the *support* of the WDFS.

It follows from the definition that if a WDFS J belongs to the support of Π, then at least one input is processed according to the fixed pattern J.

4.4.4 Lattice Approach

The modeling of variable and fixed structures in a matrix form leads to a natural ordering of the sets V and W. The importance of this ordering appears in Propositions 4.4 and 4.5, which state that V and W are lattices. First, Definition 4.9 provides an essential tool.

Definition 4.9

An array $[A_{ij}]_{i \text{ in } I,\, j \text{ in } J}$, $A_{ij} = 0$ or 1, is *"smaller than"* $[B_{ij}]_{i \text{ in } I,\, j \text{ in } J}$, $B_{ij} = 0$ or 1 if and only if

For every i and for every j, $A_{ij} \leq B_{ij}$.

One sees without any difficulty that the relation "is smaller than" is a partial ordering. First, a binary relation SUB is defined on W.

Definition 4.10

Consider two WDFS Σ and Σ' in some W. The binary relation SUB is defined by

$\Sigma = (S, s, F, G, H, C)$ SUB $\Sigma' = (S', s', F', G', H', C')$ if and only if

- S is smaller than S', s is smaller than s',
- F is smaller than F', G is smaller than G',
- H is smaller than H', C is smaller than C'.

SUB is trivially a partial ordering of W. Σ SUB Σ' means that every interaction in Σ is present in Σ'. The WDFS Σ' has more interactions than WDFS Σ. In other words, Σ SUB Σ' means that the Ordinary Petri Net that represents Σ is a subnet of the Ordinary Petri Net that represents Σ'. Similarly, a binary relation on W is given by Definition 4.11.

Definition 4.11

Consider two WDVS Π and Π' in some V. The binary relation ACT is defined by

$\Pi = (S, e, s, F, G, H, C)$ ACT $\Pi = (S', e', s', F', G', H', C')$

if and only if

$\forall$ i,j S_{ij} is smaller than S_{ij}', s_i is smaller than s'_i,
F_{ij} is smaller than F'_{ij}, G_{ij} is smaller than G'_{ij},
H_{ij} is smaller than H'_{ij}, C_{ij} is smaller than C'_{ij}.

In other words, Π ACT Π' if and only if for each x, $\Pi(x)$ SUB $\Pi'(x)$. Here again, one sees that ACT is a partial ordering. Π ACT Π' means that every input that activates an interaction in Π activates the same interaction in Π'. Π has fewer interactions than Π'.

The partial orderings SUB and ACT are important in this thesis. They provide the major tools that are used in Chapter VII to structure the set of solutions to the design problem. They have been introduced for two reasons. First of all they make sense. Since this thesis focuses on the interactions as they are induced by the observations, it seems logical to say that a structure Π is less active than another Π' if every link is activated at least as often over the set of inputs in Π' as in Π. The second reason comes from Propositions 4.4 and 4.5.

Proposition 4.4

(V, ACT) is a lattice

Proposition 4.5

(W, SUB) is a lattice

Proof

The meet of two structures Π and Π' (Σ and Σ' resp.) is the structure which consists of the interactions that are common to Π and Π' (Σ and Σ' resp.).

Every structure Π'' (Σ'' resp.) that is $\leq$ than both Π and Π'' (Σ and Σ'resp.) is necessarily $\leq$ than the meet.

The join of two structures Π and Π' (Σ and Σ' resp.) is the structure which consists of all the interactions of Π and Π' (Σ and Σ' resp.).

Every structure Π'' (Σ'' resp.) s.t. $\Pi \leq \Pi''$ and $\Pi' \leq \Pi''$ ($\Sigma \leq \Sigma''$ and $\Sigma' \leq \Sigma''$ resp.) is necessarily $\leq$ than the join.

4.5 PARTITIONS OF X

There are three types of interactions in a Well Defined Variable Structure.

- The *inadmissible* links. These are the links which correspond to the entries of Π that are the $|X| \times |X|$ null matrix. No input requires this interaction to be processed.
- The *permanent* links. These are the links which correspond to the entries of Π that are the $|X| \times |X|$ identity matrix. Every input requires this interaction to be processed.
- The *variable* links. These are the links which correspond to the entries of Π that have 0s and 1s on the diagonal. Some inputs require this interaction to be processed, while some do not. Each variable link defines a partition of X into two subsets: the set of inputs that activates the link and the set of inputs that does not activate the link.

Inadmissible links and permanent links are of little interest as far as variability is concerned. If a link is inadmissible or permanent between two stages, then the stages have a fixed pattern of interaction irrespective of the input to the system. This cannot be the case for variable links. If a system is processing some input, the stages must know whether they interact or not. This decision must be based on the information received by each role, and the major difficulty is that different roles might not have access to the same sources of information. Thus there is a need to describe the relationships between the sources of information that are commonly accessed by the stages, and the potential partitions of the set of inputs X into the subset for which a message is exchanged and the subset for which no messages are exchanged.

First, Definition 4.12 introduces a family of binary relations on X, to characterize the partitions of X that are based on a single common source of information. Recall that a system processes information from N sensors, Sensor 1,.., Sensor N. The observation of Sensor i, xi, belongs to the output alphabet, $Xi = \{xi_1,.., xi_{|Xi|}\}$ where |Xi| is the number of outputs in Xi.

Definition 4.12

For i = 1..N, the binary relation R_i is defined on $X = X_1 \times .. \times X_N$ by:

$$x\ R_i\ x' \ ,\ x = <x1,...,xN> \quad x' = <x'1,...x'N>$$

if and only if

$$xi = x'i .$$

Two inputs are equivalent according to the relation R_i if and only if they correspond to the same output from Sensor i, irrespective of the outputs of the other sensors. Proposition 4.6 describes the fundamental result of these binary relations.

Proposition 4.6

The binary relations R_i, i = 1..N, are equivalence relations.

Proof:

Let i be any integer between 1 and N.

Any $x = <x_1,...,x_N>$ has the same i-th component as itself, thus R_i is reflexive.

If $x_i = x'_i$ is true, then x'i = xi is true thus x' R_i x is true, and R_i is symmetric.

Finally, if x and x' have the same i-th component and if x' and x" have the same i-th component, i.e. $x_i = x'_i$ and $x'_i = x"_i$, then by transitivity of equality, x and x" have the same i-th component, i.e. $x_i = x"_i$. Therefore R_i is transitive.

Every equivalence relation R_i induces a partition of X into equivalence classes, as described by Theorem 3.1. By definition, two inputs to the system belong to the same equivalence class if and only if they correspond to the same output of Sensor i. Each equivalence class is thus an element of $Xi = \{xi_1, ..., xi_{|Xi|}\}$, where xi_k is the subset of all inputs whose i-th entry is xi_k.

Suppose that a variable interaction is triggered by the outputs of Sensor i only. For example, an interaction exists if and only if Sensor 1 outputs $x1_1$, irrespective of the observations of Sensors 2 to N. The set of inputs to the system that activates the link is xi_1, and the set of inputs that does not activate the link is given by the union of the equivalence classes $x1_k$, $k \neq 1$. More generally, a variable interaction triggered by the output of a unique sensor, Sensor i, is characterized by Proposition 4.7.

Proposition 4.7

A variable interaction is based exclusively on the output of Sensor i if and only if there exists a partition of $Xi = \{xi_1, ..., xi_{|Xi|}\}$ into two sets I1 and I2 such that:

$AC = \cup_{xi_k \text{ in I1}} \ xi_k$ is the set of inputs that activate the link.

$DC = \cup_{xi_k \text{ in I2}} \ xi_k$ is the set of inputs that do not activate the link

Proof

If the variability of an interaction is based exclusively on the observations of Sensor i, then there exists a partition of the output alphabet Xi into I1 and I2 such that

- I1 is the subset of outputs for which the interaction exists and
- I2 is the subset of outputs for which the interaction does not exist.

Let $AC = \cup_{xi_k \text{ in I1}} xi_k$ and $DC = \cup_{xi_k \text{ in I2}} xi_k$ then $A \cup B = X$ and $A \cap B = \emptyset$. Consider any input x to the system. The statement "x belongs to AC" is equivalent to "The output of Sensor i belongs to $\{xi_k$ k is in I1$\}$," i.e., x activates the link.

Consider a partition of X into two subsets AC and DC. If there exists a partition of Xi into two sets I1 and I2 such that $AC = \cup_{xi_k \text{ in I1}} xi_k$ and $DC = \cup_{xi_k \text{ in I2}} xi_k$ then the output alphabet Xi is said to be an *effective alphabet* of the partition AC, DC. Proposition 4.7 indicates that the partitioning of the inputs is based, in this case, on the information carried out by the output of Sensor i exclusively. These definitions and propositions can be easily extended to characterize partitions that are determined by several sources of information.

Definition 4.13

Let i1,.., ik be in [1.. N]. The binary relation $R_{i1,\ i2,...,ik}$ is defined on X by:

$$x \ R_{i1,\ i2,...,ik}\ x' , \qquad x = <x1,...,xN> \quad x' = <x'1,...x'N>$$

if and only if

$$x \ R_{i1}\ x' \text{ and...and } x\ R_{ik}\, x'.$$

In other words, two sets of simultaneous observations x and x' verify x $R_{i1,\ i2,...,ik}$ x' if and only if they correspond to the same outputs of the sensors Si1 to Sik. These binary relations verify without any further comment Proposition 4.8.

Proposition 4.8

The relations $R_{i1,\ i2,...,ik}$ $i_1,.., i_k$ in [1..N] are equivalence relations.

Each relation $R_{i1,\ i2,...,ik}$ defines a partition of X into $|X_{i1}|*..*|X_{ik}|$ equivalence classes, that is the number of different observations of the sensors Si1 to Sik. Each one can be represented without any difficulty as an element of $Xi_1 \times .. \times Xi_k$.

Example 4.5:

Let us suppose that there are only two sensors. Their output alphabets are X1= { A, B, C} and X2 = {T, U}. The equivalence classes of

- R_1 are

  A = {<A,T>, <A, U>}

  B ={<B,T>, <B, U>}

  C = {<C,T>, <C, U>}
- R_2 are

  U = {<A, U>, <B, U>, <C, U>}

  T = {<A, T>, <B, T>, <C, T>}
- R_{12} are

  <A,T> = {<A,T>} *<A,U>* = {<A,U>}

  <B,T> = {<B,T>} *<B,U >* = {<B,U>}

  <C,T> = {<C,T>} *<C,U>* = {<C,U>}.

Proposition 4.9

A variable interaction is based exclusively on the outputs of Sensor i_1,...,Sensor i $_k$ if and only if there exists a partition of $Xi_1 \times .. \times Xi_k$ into two sets I1 and I2 such that:

- $AC = \bigcup_{<xi1_{j1}, ..., xik_{jk}> \text{ in I1}} <xi1_{j1}, ..., xik_{jk}>$

  is the set of inputs that activates the link.

- $DC = \bigcup_{<xi1_{j1}, ..., xik_{jk}> \text{ in I2}} <xi1_{j1}, ..., xik_{jk}>$

  is the set of inputs that does not activate the link.

- There exist k outputs xi_1,.. xi_k , some i_m in i_1,..,i_k and some $x'i_m$ in X such that $<xi_1,.\ xi_m,..\ xi_k>$ is in AC and $<xi_1,.\ x'i_m,\ xi_k>$ is in DC.

Proof

The first part of the proof is similar to the proof of Proposition 4.7. The second part describes the conditions under which a partitioning depends on the observations of k sensors exclusively. There must exist two sequences of k observations that differ by one and only one output, such that one sequence activates the interaction and the other one does not activate it. It is necessary to know the k observations xi_1,.., xi_k to decide whether an input that contains the observations xi_p for p in [1..k] p ≠ m activates or does not activate the interaction.

Consider a partition of X into AC and DC such that $Xi_1 \times .. \times Xi_k$ can be partitioned into two sets I1 and I2 with:

- $AC = \bigcup_{<xi1_{j1}, ..., xik_{jk}> \text{ in I1}} <xi1_{j1}, ..., xik_{jk}>$
- $DC = \bigcup_{<xi1_{j1}, ..., xik_{jk}> \text{ in I2}} <xi1_{j1}, ..., xik_{jk}>$
- There exist k outputs xi_1,.. xi_k , some i_m in i_1,..,i_k and some $x'i_m$ in X s.t. $<xi_1,.\ xi_m,..\ xi_k>$ is in AC and $<xi_1,.\ x'i_m,\ xi_k>$ is in DC.

Then, Xi_1,.. Xi_k are said to be *effective alphabets* of the partition AC, DC. Proposition 4.9 indicates that the partitioning of the inputs between AC and DC is based in that case on the information carried out by the outputs of the sensors Sensor i_1,..., Sensor i_k. Effective alphabets are important in Chapter VI in order to define some structural constraints on the set V and to decompose some user defined constraints in Chapter VII.

Note that such a decomposition always exists. The classes defined by the equivalence relation $R_{1,2,..N}$ are indeed exactly the elements of X. Thus, each subset S of X can be written as the union of the equivalence classes corresponding to its elements. X1, .., Xn are potential effective alphabets of any partition of X between some and some DC. An alphabet Xi is effective if and only if there exist k outputs $xi_1,..xi_k$, some i_m in $i_1,..,i_k$ and some $x'i_m$ in X such that:

- $<xi_1,.. xi_m,.. xi_k>$ is in AC
- $<xi_1,.. x'i_m, xi_k>$ is in DC,
- i is one of the $i_1,..,i_k$.

Note that the set of effective alphabets is Ø, i.e. the activation of the link is not based on any source of information,

- if AC = Ø, DC = X (inadmissible link)
- if AC = X, DC = Ø (permanent link).

Example 4.6:

Consider the case of the interaction represented by the matrix M of Example 4.2

$$\left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] \begin{array}{l} <a, c> \\ <a, d> \\ <b, c> \\ <b, d> \end{array}$$

The set X is partitioned into {<a,c>, <a,d>}, which is the set of observations that activate the link, and {<b,c>, <b,d>}, which is the set of observations that does not activate the link. Consider the relation R1, which has two equivalence classes *a* and *b*, where *a* = {<a,c>, <a,d>} and *b* = {<b,c>, <b,d>}.

The partition of X as AC = {<a,c>, <a,d>} and DC = {<b,c>, <b,d>} corresponds to AC = *a* and DC = *b* . X1 is thus the effective alphabet of the partition.

This chapter defined variable structure distributed intelligence systems. The objects and the functional entities have been described, and a matrix representation of a variable structure has been introduced. Finally, variable links have been addressed in more depth. The next chapter, Chapter V, describes the relationships between variable structures and Colored Petri Nets.

CHAPTER V

COLORED PETRI NET MODELING

A model of variable structures has been described in Chapter IV. This chapter shows how to translate the concepts that have been defined into the language of Colored Petri Nets (CPN). A class of CPN is introduced as well as some of its properties, which provides the theoretical foundation for the methodology. The main rationale for introducing CPNs is that Ordinary Petri Nets have been used extensively to analyze and generate fixed structure distributed systems and that CPNs furnish a powerful extension that makes it possible to model variable interactions.

5.1 PRINCIPLES OF COLORED PETRI NET MODELING

First, this thesis addresses variability in the functional structure rather than variability in the resource structure. The model has been elaborated primarily to describe variable interactions between the functional units, the roles.

Second, this thesis describes variability based exclusively on the sensors' observations. For each set of N simultaneous observations $<x_1,..., x_N>$, the pattern of interaction between roles, as expressed by the entries of the matrix representation of any WDVS, is uniquely determined.

Finally, this thesis models the exchanges of information as they are induced by the sensor's observations, but does not address the actual nature of the information being transmitted, the information content of the messages.

5.1.1 Token-Colors and Occurrence-Colors

The emphasis of this approach is reflected in the CPN modeling, in which X, the set of inputs to the system, plays a critical role.

Proposition 5.1

Every transition in a CPN model of a WDVS has the same set of occurrence colors: X.

Proof:

It is known from Chapter III that each transition models a process in the system. Each process introduced in Chapter IV is an algorithm performed by a role, and each process interacts deterministically based on the observations that have been communicated to it. Each process has one and only one pattern of interaction for each input $x = <x_1,...,x_N>$ to the system. X is thus a set whose elements parametrize completely the behavior of the process: X can be selected as the set of occurrence colors for any transition in the system. Let x be any element of X and t be any stage in the CPN model. Then the firing mode of the transition t, $x = <x_1,...,x_N>$ describes that t does its part of the total processing for the case where $x = <x_1,...,x_N>$ is the set of N simultaneous observations.

Two different types of links have been introduced in Chapter IV: Interactional links, and internal links. Each link corresponds to two arcs and one place in a CPN model of a WDVS.

- One interactional link corresponds to one link between two different objects in the functional structure. This is either a link from the communication process of a sensor to the RS stage of one role, or it is a link from a stage of a role to a stage of another role.
- One internal link corresponds to one link between two stages within the same role. It describes the fact that the internal processing of a role is continuous.
- Interactional and internal links belong to the functional structure, and messages that have been generated by the inputs to the system are exchanged in the functional structure.

There is a third class of links in the system, i.e., the links that represent interactions that must be present in any system, and thus in any CPN model of the system. These are the links from the external place to the communication processes of the sensors, and the link from the output transition to the sink. These links are present and permanent because they model the conditions needed by a system to operate. There must be a constant flow of information from the environment to the sensors (links from the external place to the communication processes), and a flow of information from the system once it has selected a response (the output stage) to the effectors (the sink). These links do not raise particular issues as far as modeling or analysis are concerned. They are assumed to be present in any CPN model.

Proposition 5.2

Every place in a CPN model of the functional structure has the same set of token colors: X.

Proof:

Let us consider one place in the functional structure. This place receives messages that have been generated by some input to the system x = <x1,...,xN>. As this thesis is not concerned with the nature of the message, it is equivalent to say that a place receives a message generated by x = <x1,...,xN> or that a token whose color is x = <x1,...,xN> is put in the place. X can thus be the set of token colors for all places in the functional structure.

Propositions 5.1 and 5.2 indicate that a functional structure can be represented by a very simple class of Colored Petri Nets, that in which the sets of token colors and occurrence colors are identical for all places and transitions in the net. If a token <x1,...,xN> is in some place p in the CPN model of a functional structure, this shows that some item of information generated by the set of observations x = <x1,...,xN> is being exchanged between objects. If firing mode x=<x1,...,xN> is enabled, the stage can do its part of the processing when x = <x1,...,xN> is the input to the system. When the transition fires according to the firing mode <x1,...,xN>, one token of color <x1,...,xN> is removed from some input places of the transition and a token of color <x1,...,xN> is put in some output places.

A variable interaction corresponds to the case of a transition such that tokens are not always removed from the same places and are not put in the same output places over all firing modes in X, messages can be received from different processes and can be sent to a variable number of processes. As described in Chapter III, variable interactions are embedded in the annotations of the arcs. Each arc is annotated by a matrix and, as described in Chapter IV, each arc in the functional structure belongs to a link that models an allowable interaction. It is assumed that a link (the place and the two arcs) is represented on a CPN model of the structure if and only if the interaction is activated for at least one set of inputs to the system, i.e., one x in X.

The rows of the matrix attached to an arc stand for the token colors and the columns stand for the occurrence colors, which happen to be the same in this model: the matrix is a $|X| \times |X|$ matrix. The column corresponding to a firing mode <x1,...,xN> describes the tokens that are carried through the arc when <x1,...,xN> is the firing mode of the transition. The column is a null column if the interaction is not activated when some piece of information generated by <x1,...,xN> is being processed by the transition. The column is non null if the interaction is activated when <x1,...,xN> is being processed by the transition. Since this model implies that the arc carries exclusively a token of color $<x_1,...,x_N>$ if the firing mode of the adjacent transition is $<x_1,...,x_N>$, an interaction is activated if and only if the diagonal entry that corresponds to $<x_1,...,x_N>$ in the $|X| \times |X|$ matrix x is "1" and all other entries in the column are 0s.

5.1.2 Example

Consider the functional structure shown on Figure 5.1. It has two roles and one sensor whose output alphabet is {A, B, C}.

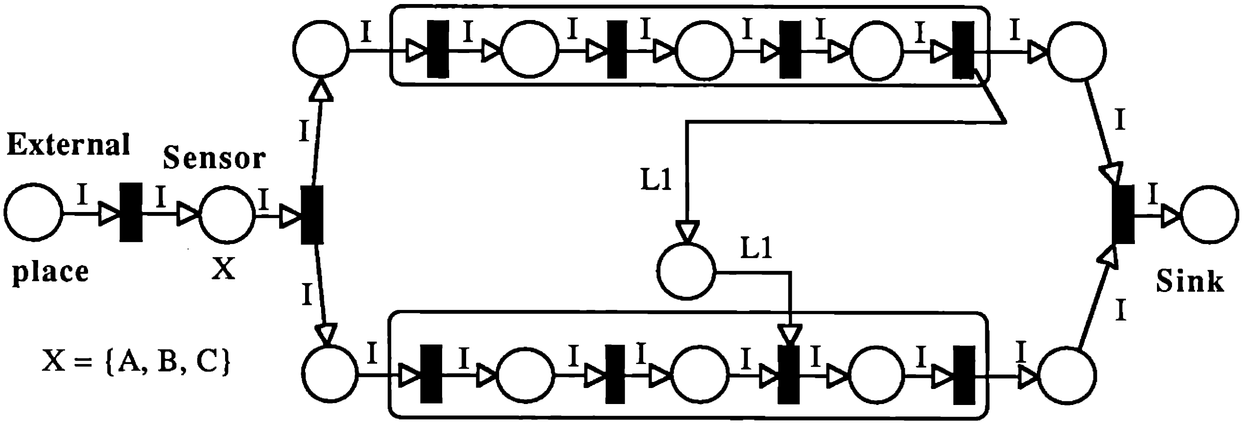


Fig. 5.1 Colored Petri Net model of a Functional Structure.

X is the set of token colors for every place, and the set of occurrence colors of every transition. Each arc in the net has been annotated by either I or L1.

The matrix I is the 3 × 3 identity matrix $\begin{array}{cccc} <A> & <B> & <C> & \\ 1 & 0 & 0 & <A> \\ 0 & 1 & 0 & <B> \\ 0 & 0 & 1 & <C> \end{array}$

The matrix L1 is the 3 × 3 matrix $\begin{array}{cccc} <A> & <B> & <C> & \\ 1 & 0 & 0 & <A> \\ 0 & 0 & 0 & <B> \\ 0 & 0 & 0 & <C> \end{array}$

The matrix I indicates that the arc is activated for every input in the system, whereas L1 indicates that the arc is used only if the input to the system is <A>. The functional structure of

Figure 5.1 is a structure in which all links are activated over all possible inputs, except the link from the RS stage of the upper role to the CI stage of the lower role. This link is activated if and only if the input to the system is <A>.

Figures 5.2 to 5.7 describe variable patterns of interaction in this system. Figure 5.2 describes a case in which the input to the system is <A>. The upper role has done its processing up to its RS stage, whereas the lower role has done its processing up to its CI stage.

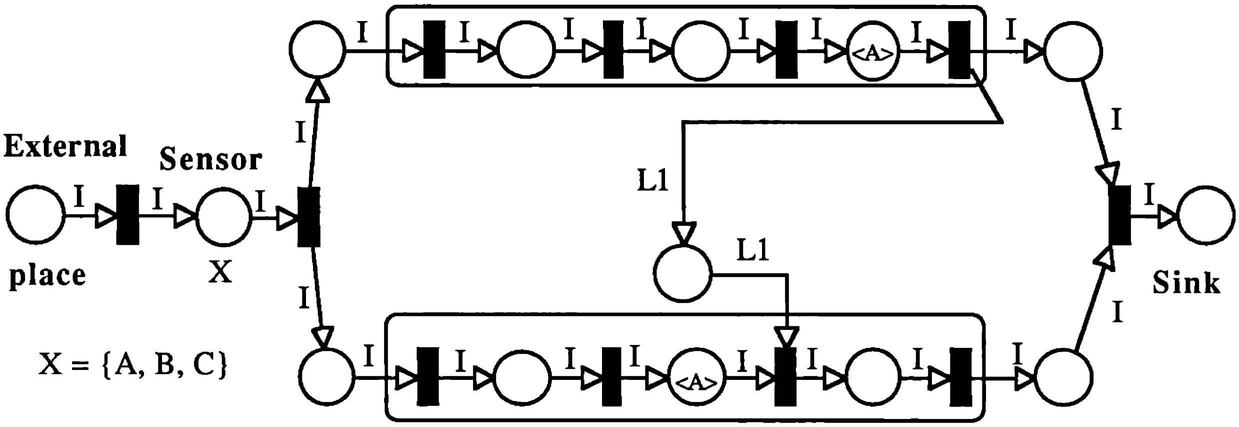


Fig. 5.2 Colored Petri Net with Marking

Note that only the firing mode <A> of the RS stage of the upper role is enabled. This RS stage has indeed one and only one input place, which contains one token of color <A>. However, the CI stage of the lower role is not enabled. This stage has two input places and the annotations of the arcs from the input places to the transition indicate that the arcs must carry one token of color <A> if the transition is to contribute to the processing of the input <A>. The firing mode <A> is not enabled because one input place from which a token of color <A> should be removed does not contain one such token.

Figure 5.3 depicts the Colored Petri Net and its marking once the upper role has selected a response induced by <A>, i.e., once the transition RS of the upper role has fired according to its firing mode <A>. Observe that the firing mode <A> of the CI stage of the lower role is now enabled.

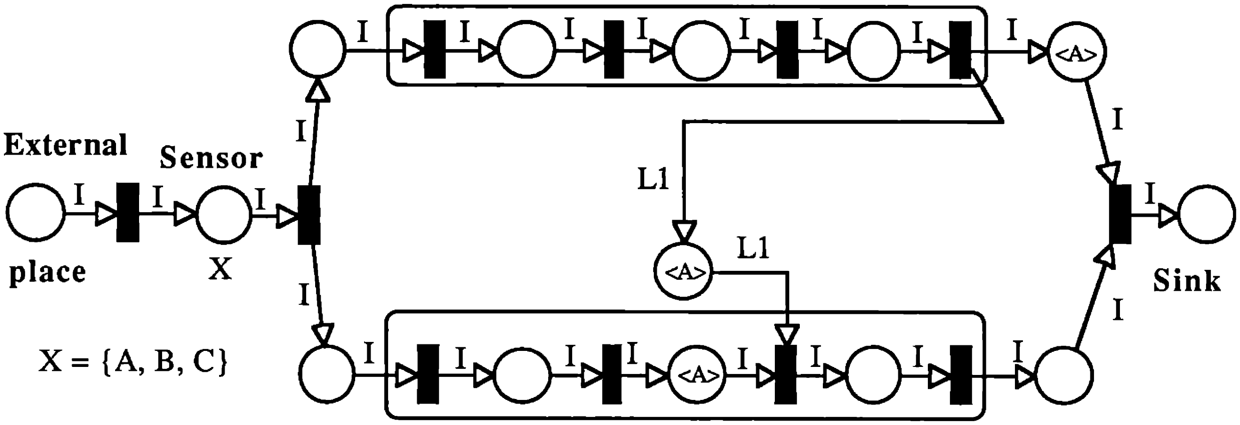


Fig. 5.3 Colored Petri Net after Firing

If the CI stage of the lower role fires according to its firing mode <A>, the marking becomes as shown in Figure 5.4.

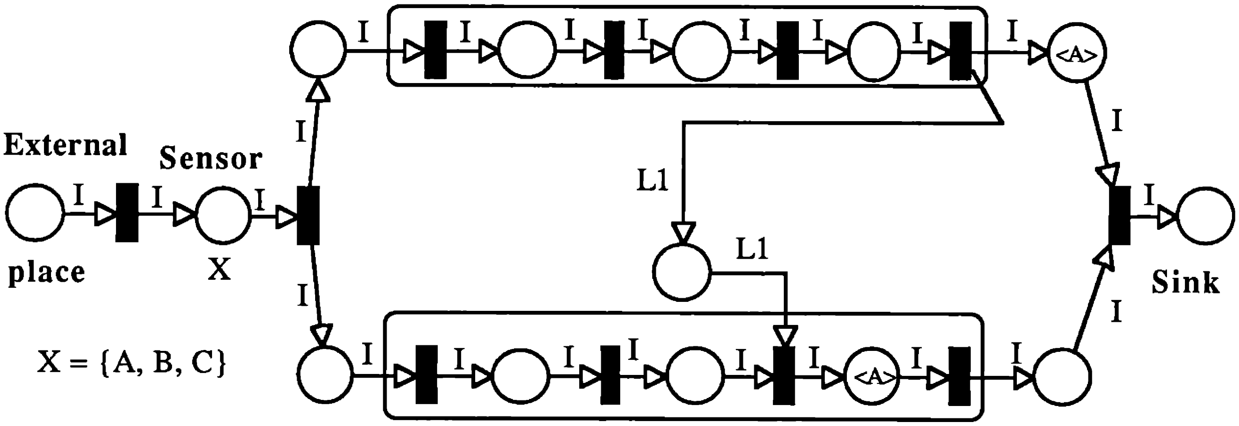


Fig. 5.4 Colored Petri Net after second Firing

Figure 5.5 depicts a case in which the upper role has done its processing up to its RS stage, whereas the lower role has done its processing up to its CI stage. The firing mode <B> of the RS stage of the upper role is enabled because its unique input place contains one token of color <B>. Unlike the case of figure 5.2, one firing mode of the CI stage of the lower role is also enabled. This stage has two input places but the annotations of the arcs indicate that only the arc from the internal place to the CI stage must carry one token of color <B> if the transition is to contribute to the processing. This place contains one token of color <B>; thus the lower role can proceed with its command interpretation, irrespective of the fact that its other input place is empty.

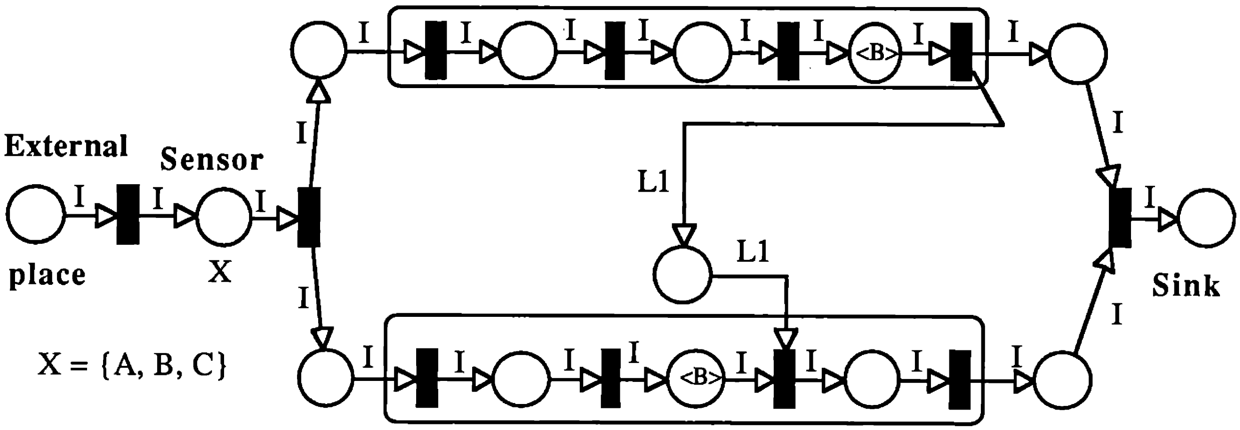


Fig. 5.5 Colored Petri Net

Both the RS stage of the upper role and the CI stage of the lower role can fire. Figure 5.6 depicts the new marking if both roles fire simultaneously.

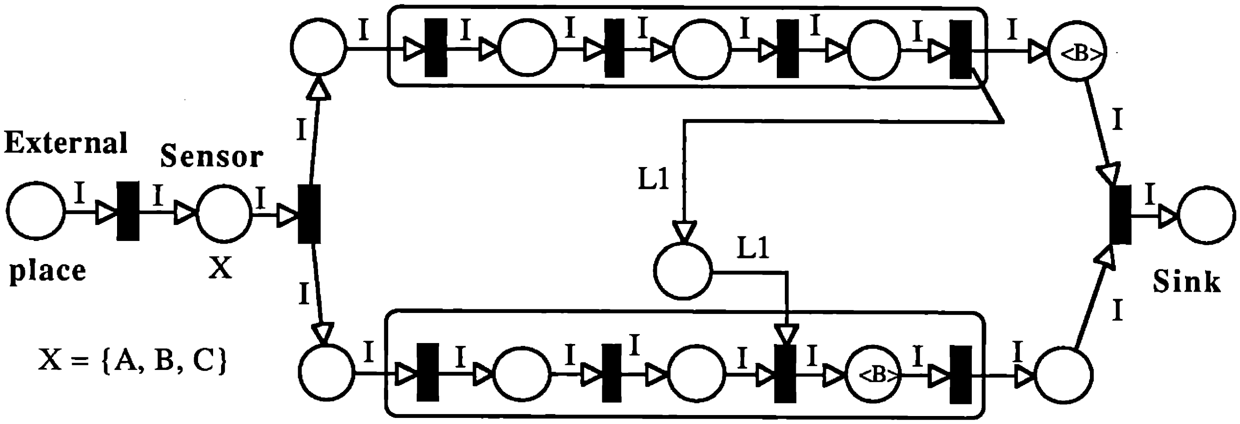


Fig. 5.6 Colored Petri Net after Firing

5.1.3 Token-Colors Continued

In order to simplify the notations, this thesis adopts the convention that a link that is always activated, i.e. which is annotated by the |X| × |X| identity matrix, can be represented by a simple arc without annotations. This convention applies in particular to the links from the external place to the sensors, and from the output transition to the sink. Another convention has to do with the annotations of the arcs that belong to the same link. These arcs are annotated by the same matrix, so there is no need to annotate twice the same matrix. In the rest of this thesis, most of the CPN models annotate one and only one arc of any link. Next, this subsection turns to the tokens in the resource structure.

Proposition 5.3

A place of the resource structure contains tokens whose colors are <1>,..., <D>.

Proof: Each place in the resource structure indicates whether a physical resource is available. Each physical resource has a number, and there is no need to indicate any more information than the number of the resource involved.

The resource structure is depicted in a very simple way.

The place Resource i, i = 1,.., D, can contain only one token of <i>, because it indicates the availability of the i-th physical resource. C(Resource i) = {<i>}.

For each role j, j = 1..R, the place Resources Role j, which contains the resources assigned to Role j, can receive any physical resource. The set of token colors is C = { <1>,..., <D>}.

The transitions that describe the assignment of one resource to the roles have the same set of occurrence colors C = { <Role 1>,..., <Role R>}, the set of roles.

The arc from every place Resource i to the transition that describes the assignment of the resource is always activated. Here again the convention is adopted that these arcs need not be annotated.

If the physical resource i can be assigned to Role j, there exists an arc from the transition that represents the assignment of resource i to the place Resources Role j. This arc is annotated by the 1 × R matrix whose columns correspond to the roles. The j-th entry is 1 because the physical resource can be assigned to Role j, the other entries are 0. This arc carries a token of color <i>.

A model of a resource structure is depicted on Figure 5.7. This resource structure corresponds to two roles and two resources. Resource 1 and Resource 2 can be assigned to either role. The marking indicates that Resource 2 has been assigned to Role 2 and that Resource 1 has not been assigned. The arcs have been annotated by A and B, where

$$A = \begin{array}{cc} \text{Role 1} & \text{Role 2} \\ [\ 1 & 0\ ] \end{array} \quad \text{and} \quad B = \begin{array}{cc} \text{Role 1} & \text{Role2} \\ [\ 0 & 1\ ] \end{array} .$$

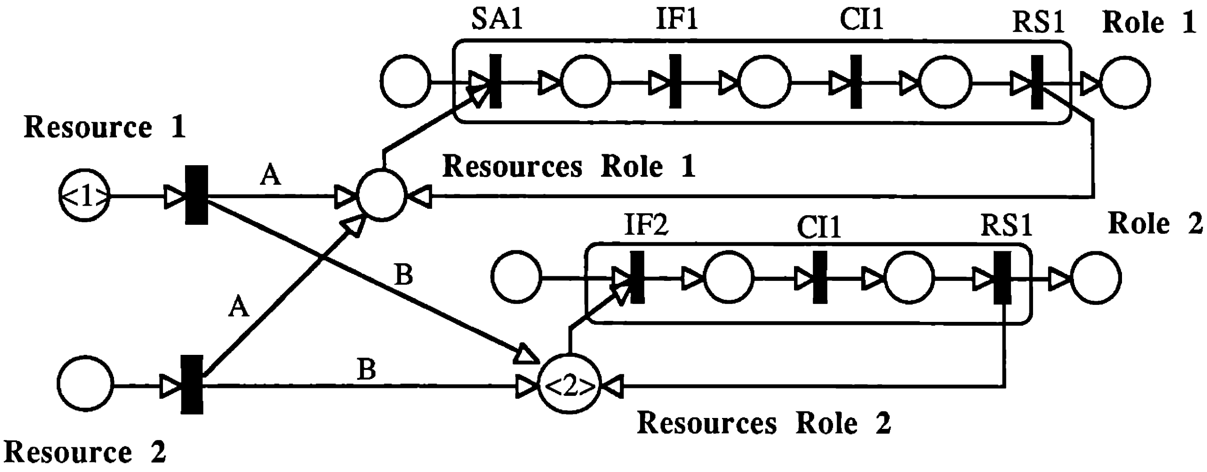


Fig. 5.7 Colored Petri Net model of Resource Structure

5.2 COLORED PETRI NET REPRESENTATION OF A WDVS

Section 5.1 described the principles of a CPN modeling of a WDVS. This section proves that it is equivalent to specify a WDVS in a matrix form or a WDVS with a CPN model. Throughout this section, the parameters that characterize a family of variable structures: N, the number of sensors: X, the set of inputs to the system; R, the number of roles; D, the number of physical resources are supposed to be fixed and well known.

5.2.1 Transitions

Each process of Chapter IV is modeled by a transition whose set of occurrence colors is X.

a) Each role is made of at most four stages. Each role is thus modeled by at most four transitions, and R roles contribute at most 4∗R transitions to the structure.

b) N transitions model the communication process of the sensors' observations.

c) The transition between the external place and the sensors, which models the fact that the observations are temporally consistent, and the transition with a single link to the sink, which models the communication of the response selected by the system to the effectors, must also be added to a CPN model of the structure.

Hence, there are at most $4R + N + 2$ transitions in a CPN model of a WDVS.

5.2.2 Places

Several notions to be modeled by a place in a CPN model have been defined in Chapter IV.

a) The external place. This place models the environment. A token of color $<x_1,..., x_N>$ in the external place indicates that $<x_1,..., x_N>$ is the set of observations that will be processed by the system.

b) The sink. This place models the effectors. A token of color $x = <x_1,..., x_N>$ is put in the sink to model the reception by the effectors of the response selected,when the input is x.

c) The sensors. N places are created, one for each sensor, Sensor 1,..., Sensor N. A token of color $<x_1,..., x_N>$ put in a place that models a sensor indicates that the observation of the sensor can be accessed by the roles.

d) Interactional places. An interactional place belongs to a link that models an interaction. A place is represented in the graph if the interaction is activated for some inputs in X.

- At most $N * R$ places correspond to one link from a sensor to a SA stage, because some sources of information might not be accessible to some roles.
- At most $R * (R - 1)$ places model an interaction between the SA stage of one role and the IF stage of another role.
- At most $R * (R - 1)$ places model an interaction between the RS stage of one role and the SA stage of another role.
- At most $R * (R - 1)$ places model an interaction between the RS stage of one role and the IF stage of another role.
- At most $R * (R - 1)$ places model an interaction between the RS stage of one role and the CI stage of another role.
- At most R places model the communication of the response selected by a role to the effectors.

e) Internal places are the places that belong to an internal link. Each role is composed of at most four stages which account for at most three internal links and three places. There are at most $3 * R$ internal places.

In conclusion, there are at most $2 + N * R + 4 R * (R - 1) + R + 3 R = 4R^2 + N*R + 2$ places in a CPN model of a WDVS.

5.2.3 Arcs

Two arcs are created for every link that is activated by at least one input to the system. Observe, however, that internal links and interactional links are not equivalent as far as the matrix representation of a WDVS is concerned. A matrix form describes the activation of the interactional links, but does not describe the interactional links.

Proposition 5.4

The activations of all links in a WDVS are uniquely determined by the activation of the interactional links.

Proof:

Suppose that the activation of the interactional links is given. The activation of the internal links is uniquely determined according to the following rules.

- Links SA --> IF.

  A link from the SA stage of a role to its IF stage is activated if and only if the SA transition has at least one interactional input link that is activated. Furthermore, the internal link is activated by any input that activates at least one of the input links, i.e., an internal link processes all information received by its input process.

- Links IF --> CI.

  A link between IF and CI of a role is activated if and only if the IF transition has at least one input link - interactional or internal - that is activated. Furthermore, the internal link is activated by every input that activates at least one of the input links of IF.

- Links CI --> RS.

  A link between the CI and RS stages of a role is activated if and only if the CI transition has at least one input link - interactional or internal - that is activated, and the internal link is activated by every input that activates one of the input links of CI.

Proposition 5.5

Let A be an arc that corresponds to an interactional link L in a WDVS Π. The matrix that annotates A in a CPN model of the functional structure is exactly the entry that corresponds to L in (S, s, F, G, H, C), the matrix representation of Π.

Proof:

By definition, the entry that corresponds to L in (S, s, F, G, H, C) is a |X| × |X| diagonal matrix, $L_{ii} = 1$ if the i-th input in the lexicographic ordering activates the link and $L_{ii} = 0$ if the i-th input in the lexicographic ordering does not activate the link. If the set of token-colors and the set of occurrence-colors are ranked in the lexicographic ordering, then this matrix indicates that a token of color x=<x1,...,xN> is carried by the arcs of the link when the system processes the input x=<x1,...,xN>, i.e., that the link is activated and that the activation is represented by an exchange of a token whose color is x=<x1,...,xN>. The diagonal matrix $[L_{ii}]$ has all the properties described in section 5.1 and is the proper annotation of the arc.

Propositions 5.5 and 5.4 imply that a CPN model of a WDVS is uniquely determined by the representation of Π in a matrix form. The matrix form shows the interactions that are activated and must be depicted by an interactional link (one place and two arcs) in the CPN model. Furthermore, the entry that corresponds to an interactional link is the matrix that annotates the arcs. Finally, internal links are automatically created once interactional links are given. The annotations of both arcs of an internal link are computed as the meet of the matrices that annotate the input transition. Conversely, it is easy to translate a CPN model of a WDVS into a matrix form. The representation of a WDVS in a matrix form or as a CPN are equivalent.

5.2.4 Example

Let us illustrate the translation from a matrix form into a CPN model. Consider a WDVS with one sensor, whose output alphabet is {A, B, C}, and two roles, Role 1 and Role 2. Suppose further that the matrix form is

S = [0, I], where 0 is the 3 × 3 null matrix and I is the 3 × 3 identity matrix.

$$s = [I\ ,\ 0],\ F = H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \text{ and } G = \begin{bmatrix} 0 & 0 \\ L1 & 0 \end{bmatrix},\ C = \begin{bmatrix} 0 & 0 \\ L2 & 0 \end{bmatrix}, \text{ where L1 and L2 are}$$

$$L1 = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right]\begin{array}{c} <A> \\ <B> \\ <C> \end{array} \text{ and } L2 = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right]\begin{array}{c} <A> \\ <B> \\ <C> \end{array}$$

A CPN model of the interactional links exclusively is depicted in Figure 5.8.

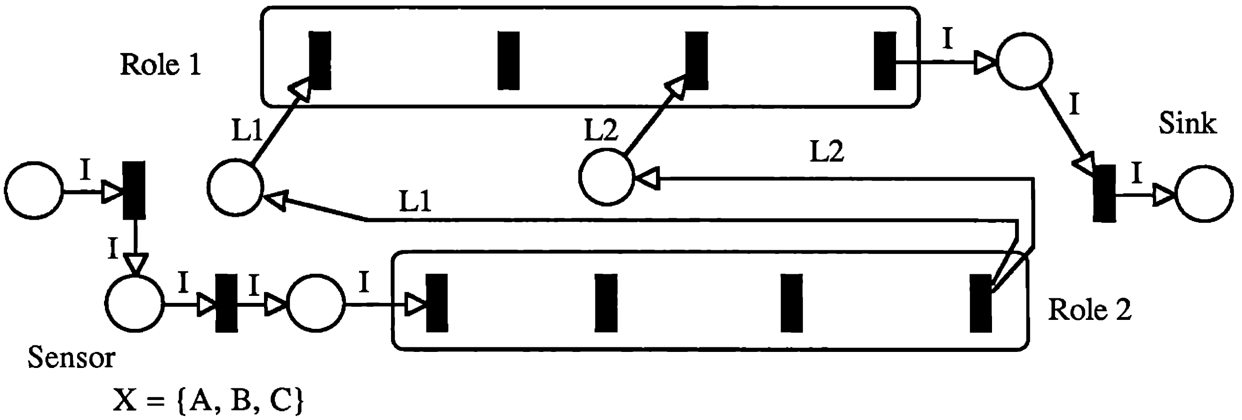


Fig. 5.8 CPN model of Interactional Links

The internal links can be added to the CPN model using the rules of Proposition 5.4. A CPN model that contains both internal and interactional links is depicted on Figure 5.9.

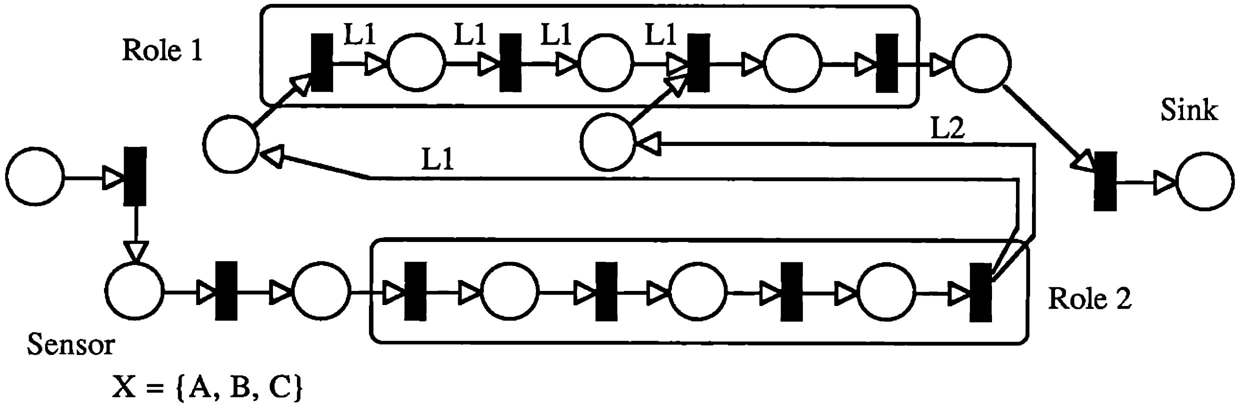


Fig. 5.9 CPN model with Internal and Interactional links

This subsection illustrated the one to one correspondence that exists between matrix representations of WDVS and CPN models of the structures. As indicated in Chapter IV, a WDVS is also a mapping from X into the set of fixed structures W. The next subsection relates this approach to the CPN modeling.

5.3 FOLDING AND UNFOLDING

It is assumed in this subsection that the parameters X, N, R are fixed and well known.

Proposition 5.6

Each element of W can be represented by one and only one Ordinary Petri Net.

Proof:

Let $\Sigma = (S, s, F, G, H, C)$ be some element of W. Σ describes fixed interactions in a structure. The methodology of 5.2 can be almost literally adapted to associate one Ordinary Petri Net model to each fixed structures in W. A transition is created for each process in the fixed structure. Places that represent the external place, the sink, and the sensors are automatically created. Each fixed interaction is modeled by one place and two arcs without annotations. If the interaction exists in the fixed structure, the place and the arcs are incorporated in the net. If the interaction does not exist, the place and the arcs are not represented. Finally, internal links are uniquely determined by the interactional links, as for variable structures, and the rules described in proposition 5.4 are valid.

Reciprocally, each Ordinary Petri Net model of a structure can be translated without any difficulty into a matrix form.

By combining propositions 4.3 and 5.6 one easily sees that a WDVS is a mapping from X into the set of Ordinary Petri Nets since each fixed structure in the support of the WDVS is associated with a unique Ordinary Petri Net. The elements of the support can be described alternatively as matrices or Ordinary Petri Nets. Proposition 5.7 states an important relationship between the Ordinary Petri Nets in the support of a WDVS Π and the CPN model of Π.

Proposition 5.7

Let Π be a WDVS, and $\Sigma_1,..., \Sigma_k$ be the Ordinary Petri Nets in the support.

The CPN model of Π is obtained by

- Computing the join $\Sigma_1 \cup \Sigma_2 \cup ... \cup \Sigma_k$, which is the graphical support of the CPN.
- Let A be an arc that belongs to $\Sigma_1 \cup \Sigma_2 \cup ... \cup \Sigma_k$. A is annotated by a $|X| \times |X|$ diagonal matrix such that, if k(x) is the rank of x in the lexicographic ordering of X then,

  $A_{k(x)\,k(x)} = 1 \Leftrightarrow \Pi(x)$ contains A and $A_{k(x)\,k(x)} = 0 \Leftrightarrow \Pi(x)$ does not contain A.

Proof:

If an arc is represented on a CPN model of the structure, this arc is activated by some input x. Therefore $\Pi(x)$ belongs to the support and contains A, as does $\Sigma_1 \cup \Sigma_2 \cup .. \cup \Sigma_k$. Reciprocally, if an arc belongs to $\Sigma_1 \cup \Sigma_2 \cup ... \cup \Sigma_k$ then at least one element of the support Σ_i contains A. A is activated by at least one x such that $\Pi(x) = \Sigma_i$.

Finally, it is clear from the definition of the support that an arc A is activated by every input such that $\Pi(x)$ contains A.

Proposition 5.8

Any CPN model of a WDVS can be translated into a mapping of X into the set of Ordinary Petri Nets.

Proof:

Any CPN model can be translated into a matrix form. This matrix form can be translated into a mapping from X to W, which is automatically translated into a mapping from X into the set of Ordinary Petri Nets.

Propositions 5.7 and 5.8 present two methods to describe and analyze a WDVS. These methods can be referred to as *folding* and *unfolding*. Proposition 5.7 describes an automatic way to translate a mapping from X to a set of k Ordinary Petri Nets into a CPN, i.e., a way to fold the k Ordinary Petri Nets into one CPN, and to annotate the arcs of the CPN so as to indicate the elements of X that activate them. Conversely, Proposition 5.8 provides an automatic procedure to unfold a CPN model of a WDVS into a mapping from X to a subset of k different Ordinary Petri Nets. These possibilities are exploited extensively in the rest of this thesis.

A folding and an unfolding are given below.

Consider the WDVS of Figure 5.9. This variable structure can be unfolded as a mapping:

<A> --> to Σ_1 , <B> --> Σ_2, and <C> --> Σ_2

with Σ_1 the Ordinary Petri Net of Figure 5.10
and Σ_2 being the Ordinary Petri Net of Figure 5.11

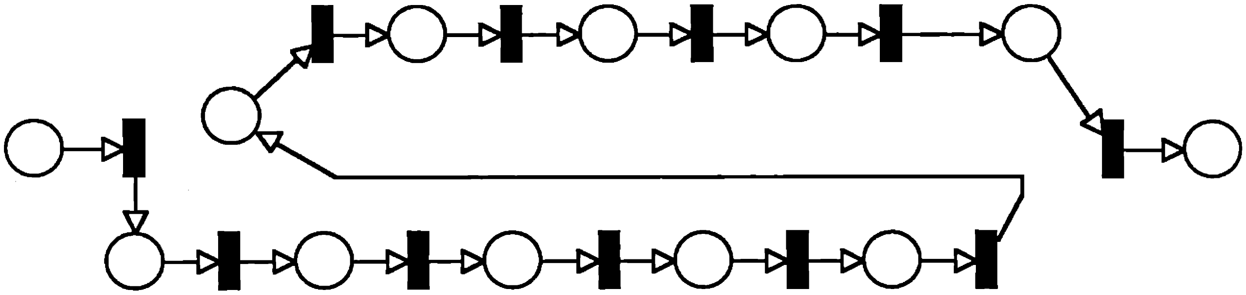


Fig. 5.10 WDFS Σ_1

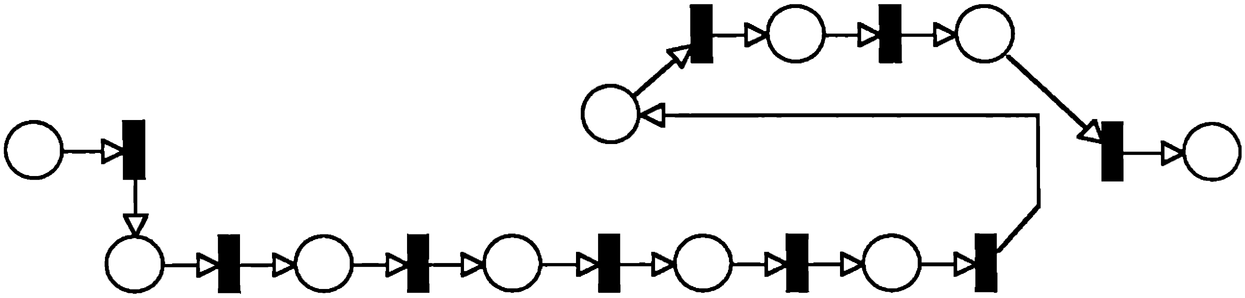


Fig. 5.11 WDFS Σ_2

Reciprocally, consider the mapping:

<A> --> Σ'_1, <B> --> Σ'_2, <C> --> Σ'_3

with Σ'_1 , Σ'_2 and Σ'_3 represented in Figures 5.12, 5.13 and 5.14.

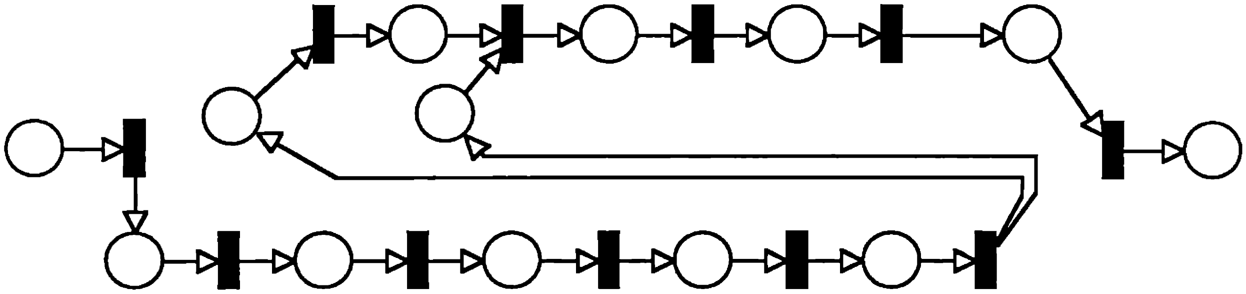


Fig 5.12 WDFS Σ'_1

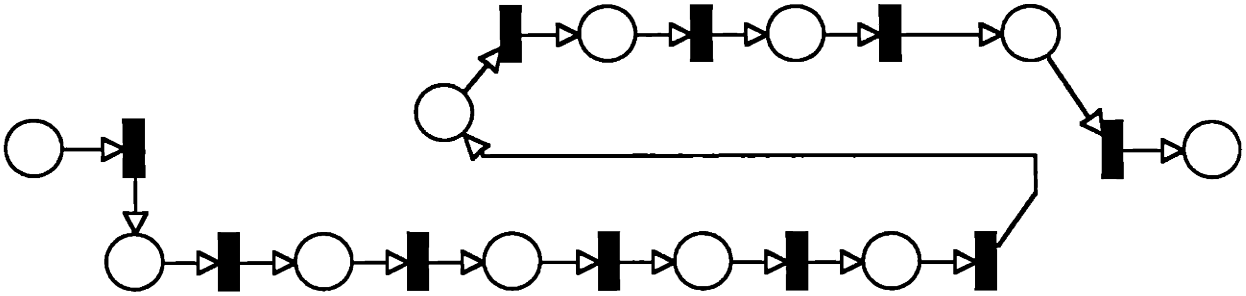


Fig. 5.13 WDFS Σ'_2

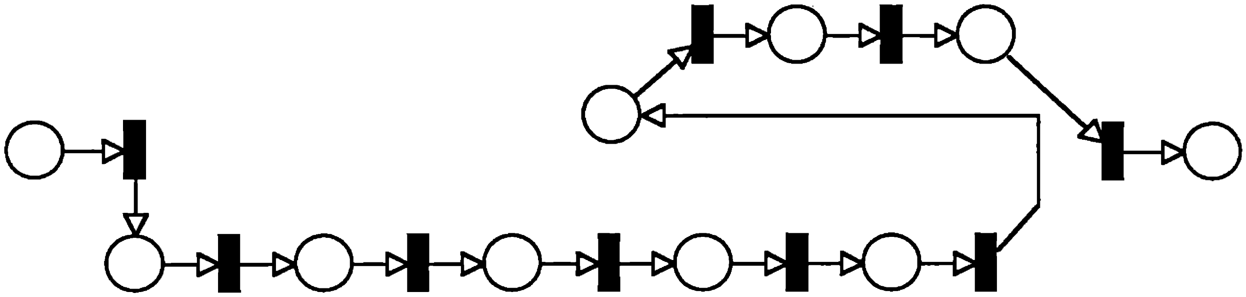


Fig. 5.14 WDFS Σ'_3

Then, the folding into one CPN model of the WDVS yields the CPN of Figure 5.15

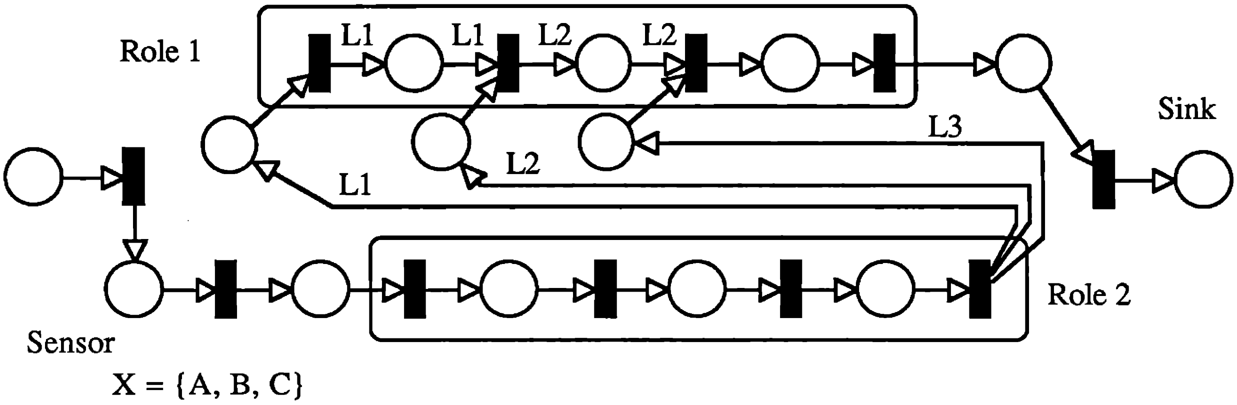


Fig. 5.15 Folded Structure

$$\text{where L1} = \begin{array}{c} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{l} \text{<A>} \\ \text{<B>} \\ \text{<C>} \end{array}, \quad \text{L2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{array}{l} \text{<A>} \\ \text{<B>} \\ \text{<C>} \end{array}, \text{ and L3} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} \text{<A>} \\ \text{<B>} \\ \text{<C>} \end{array}$$

5.4 USEFUL PROPERTIES OF WDFS

Proposition 5.9 characterizes a class of Ordinary Petri Nets that model some elements of W.

Proposition 5.9 (Remy)

> Let the external place and the sink of the Ordinary Petri Net representing a WDFS be combined into a unique place. If the resulting Petri Net is strongly connected, it is a marked graph.

The proof is straightforward. Each internal or interactional place has exactly one input and one output transition. The sink has one input transition but no output transitions, while the opposite holds for the external place. If the external place and the sink are merged together into one single place, every place in the net will therefore have one input and one output transition. Since the net is strongly connected, every transition has at least one input place and one output place, and the net is a marked graph.

Proposition 5.9 implies, in particular, that if each WDFS in the support is strongly connected, each element in the support is a marked graph, and such is their join: every place has exactly one input and one output transition, in the Colored Petri Net model as well as in the Ordinary Petri Net model. The graphical support of the CPN is a thus a marked graph if every element in the support is strongly connected.

Note finally that combining the external place and the sink in a single place has no bearing on the internal topology of the net. This assumption is introduced because it facilitates the exposition of the results in Chapter VIII as well as an implementation of the methodology. However, this assumption can become critical in other contexts, such as the study of the dynamical behavior of the CPN. In those cases, the external place and the sink should not be merged.

5.5 DESIGN PROBLEM

It is now possible to formulate the part of the feasibility study that is addressed by this thesis. In Chapter I, the needs of a team responsible for the feasibility study of a distributed intelligence system were considered. This thesis assumes that the team is at a very early stage of the study, and wants to generate all physical structures that can be solutions to the problem. The ultimate goal is to the performance of a large set of solutions, with the System Effectiveness Analysis methodology for example, to gain a fair insight into the feasibility of the problem. Once the performance of the structures is predicted, the team will opt for a small number of candidate structures that will be studied in more detail. This thesis suggests that the first step of the design problem is to generate the CPN models of the variable structures which satisfy the requirements of the design. The performance of the variable structures can be determined using the extension of System Effectiveness Analysis in Monguillet (1988), which require the use of CPNs.

This thesis has presented a model of variable structure systems that can be automatically translated into the language of Colored Petri Nets. Next, the problem of generating a variable structure is tackled in two steps. The first is to generate a CPN model of the functional structure, and the second is to add to this CPN a description of the assignment of resources to the roles.

In the first step, the set of WDVS is characterized by four set of parameters

- X, the set of inputs to the system.
- N, the number of sensors.
- *R*, the set of different roles from which the system may be built. This set contains all the combinations of decisionmaking processes that can be performed by some entity. Let r be its cardinality.
- $\{r_i, i = 1..r\}$, the set of *degrees of redundancy* of each role. The degree of redundancy r_i is the number of entities that operate concurrently according to the i-th role during the processing of some input to the system. For any input, its functional structure activates R roles with

$$R = \sum_{i=1}^{r} r_i \, .$$

This thesis assumes that these parameters are fixed by the team in charge of research and development before using the methodology of this thesis. The idea is that the team can start investigating the candidate structures only once it has gained some knowledge of the functions (the roles) that must be embedded in the system. These parameters are fairly general, and furnish the basic dimensions of the system. If the team wants to use other combinations of the parameters, it must reapply the methodology each time.

In the second step, the parameters are the physical resources. It is assumed in this thesis that the assignment of the resources to the different roles is an input to the design problem. The description of the assignment of the resources can be done automatically once a functional structure is created. The main rationale behind this assumption is the scope of the thesis. The generation of resource structures could not be tackled within the framework of a Master's thesis, and remains an open question.

CHAPTER VI

CONSTRAINTS

Well defined variable and fixed structures have been introduced in Chapter IV. It has been noted that any variable functional structure is a Well Defined Variable Structure, but that the converse might not be true. In this chapter, the constraints that must be verified by Well Defined Variable Structures are presented.

6.1 INTRODUCTION

Several properties of the set of well defined variable structures V lead to the definition of constraints that must be satisfied by the elements of V to solve the design problem of a Research & Development team.

(a) Some Well Defined Variable Structures correspond to patterns of interactions between roles that do not make sense. The designers do not want to obtain these structures as candidate architectures for the system under study. There is thus a need to define *structural constraints*, which define the types of WDVS that must be ruled out within the model of this thesis.

(b) For that purpose it is convenient to consider V as the set of mappings from X to W, which makes it possible to distinguish two types of structural constraints. First, constraints are imposed on the set W to obtain fixed data flow structures that make physical sense. Then, admissible assignments associate an admissible fixed structure with each x and must verify additional properties as well.

(c) Any practical procedure has to allow the designer to investigate only those candidate structures that correspond to the degree of knowledge he has about the system, the technological limitations, the physical limitations, etc. The methodology has to compute only the structures that are of interest for the problem addressed by the designer. He must thus be given the possibility to translate his knowledge about the system into mathematical terms by imposing *user-defined constraints*.

(d) Finally, the introduction of constraints on the set of variable structures can significantly reduce the computational requirements, while still allowing one to benefit from the lattice structure of V and W.

6.2 STRUCTURAL CONSTRAINTS

A WDVS that makes physical sense must satisfy two sets of properties:

(1) Its support must be made of fixed structures that make physical sense. These constraints are described in section 6.2.1.

(2) If the support of a variable structure is made out of the k fixed structures $\{J_1,...,J_k\}$, the mapping of X onto {J1,...,Jk} must be realistic. Those constraints are described in section 6.2.3

6.2.1 Constraints on Fixed Structures

Constraints on the set of fixed structures have been defined in Remy (1986) using a different model. In the sequel, these constraints are adapted to the context of this thesis. Let Π be a variable structure. For any x, x in X, its associated fixed structure $\Pi(x)$ must satisfy

- (R1) (a) The Ordinary Petri Net that corresponds to $\Pi(x)$ should be connected, i.e., there should be at least one (undirected) path between any two nodes in the Net (b) A directed path should exist from the external place to every node of the PN and from every node to the sink.

- (R2) The Ordinary Petri Net that corresponds to $\Pi(x)$ should have no loops, i.e., the structure is acyclic.

- (R3) In the Ordinary Petri Net that corresponds to $\Pi(x)$, there can be at most one link from the RS stage of a role i to another role j, i.e., for each i and j, only one element of the triplet $\{G(x)_{ij}, H\ (x)_{ij}, C\ (x)_{ij}\}$ can be non-zero.

- (R4) Information fusion can take place only at the IF and CI stages. Consequently, the SA stage of a role can either receive observations from sensors, or receive one and only one response sent by some other role.

- (R5) There cannot be one link from the SA stage of role i to the IF stage of role j and a link from the RS stage of role i to the SA stage of role j.

Each WDFS that fulfills these constraints is called an admissible fixed structure (AFS). The set of all AFSs is called AW. These constraints can be interpreted as follows. Constraint R1(a) eliminates a data flow structure that does not represent a single structure. Constraint R1(b) insures that the flow of information is continuous within the organization, that there are no other sources of information than the N Sensors.

Constraint R2 allows acyclical fixed dataflow structures only. The acyclical hypothesis is very general as far as data flow structures are concerned. This restriction is imposed to avoid deadlocks and infinite circulation of messages within the organization. Deadlock occurs when one decisionmaker is waiting for a message from another, while the second one is in turn waiting for an input from the first. An infinite circulation of messages occurs when the reception of message 1 at role 1 triggers the communication of message 2 to role 2, and when the reception of message 2 at role 2 triggers the communication of message 1 to role 1. Note, however, that constraint R2 does not imply that the graphical representation of the variable structure is acyclical, because the folding of acyclical nets can yield a net with loops. Furthermore, when the resource structure is added to a CPN model of the functional structure, resource loops and role loops are created. The constraint of acyclicity is restricted to the elements of W.

Constraint R3 indicates that it does not make sense to send the same output to the same role at several stages. It is assumed that once the output has been received by a role, this output is stored in its internal memory and can be accessed at later stages.

Constraint R4 has to do with the nature of the IF stage.The IF stage has been introduced explicitly to perform a fusion between the situation assessments performed by the other roles. There are thus two cases. Either the role receives the observations from at least one sensor and fuses the inputs from the other roles at the IF stage, or the role monitors no direct observations. In the latter case, there can be at most one link to the SA stage, which indicates that the role is activated. Inputs that come from different roles in the system are fused together in the IF stage.

Finally, Constraint R5 has been introduced to avoid the interactions of Figure 6.1. In this pattern of interactions Role i would send its situation assessment to Role j, but Role j would not process it until it had received the final response of Role i. The response of Role i is thus no real input for Role j, but rather a response to be fused with the information received. The physical interaction is thus much more appropriately described by Fig. 6.2

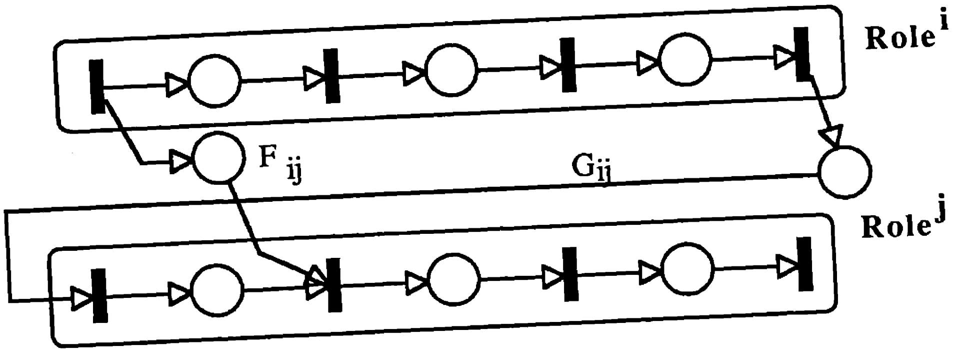


Fig. 6.1 Forbidden Interactions

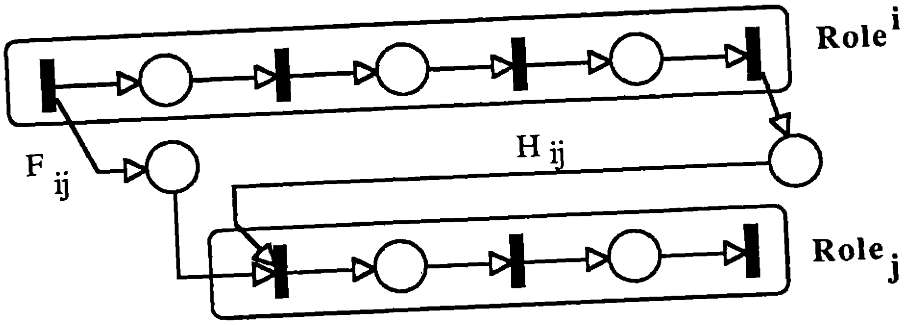


Fig. 6.2 Allowable Interactions

6.2.2 Mathematical Representation of the Constraints on W.

The constraints on WDFS can be translated into formal ones as follows.

Constraint R1.

If the external place and the sink are merged in the Petri Net that depicts $\Pi(x)$, then R1(a) and R1(b) can be formulated by

The Petri Net representing $\Pi(x)$ should be strongly connected.

Corollary. Proposition 5.9 indicates that any element of W that satisfies Constraint

R1 is a marked graph.

Constraint R2.

Suppose that R1 is fulfilled and that the external place and the sink are merged together in a single place. Constraint R2 becomes

All simple information flow paths of the Petri Net contain the external place.

where simple information flow paths have been described in Definition 2.14.

Constraint R3.

The analytical expression of this constraint is straightforward:

$$\forall (i, j) \in [1..R]^2 \qquad G_{ij} + H_{ij} + C_{ij} \leq 1$$

Constraint R4.

Like R3, R4 is translated easily into

$$\forall j \in [1..R] \quad (Max_{i\ in\ [1..R]}\ \{S_{ij}\}) + \sum_{i\ in\ [1..R]} G_{ij} \leq 1$$

Constraint R5.

Finally R5 is trivially expressed by

$$\forall (i, j) \in [1..R]^2 \qquad F_{ij} * G_{ij} = 0$$

6.2.3 Constraints on Assignments

Suppose now that $\{J_1, ..., J_k\}$ is a set of k fixed structures that fulfill R1,..., R5. This section describes the conditions that must be fulfilled by a mapping from X onto $\{J_1, ..., J_k\}$.

Definition 6.1

Let L be a link

If L is present in at least one fixed structure J_i, L is said to be *admissible.*

If L is not present in any of the k fixed structures, L is said to be *inadmissible.*

If L is admissible and present in all WDFS $J_1, ..., J_k$, L is said to be *permanent.*

If L is admissible but not permanent, L is said to be *variable* over $\{J_1, ..., J_k\}$.

Example 6.1:

Consider the four fixed structures represented on Figure 6.3.

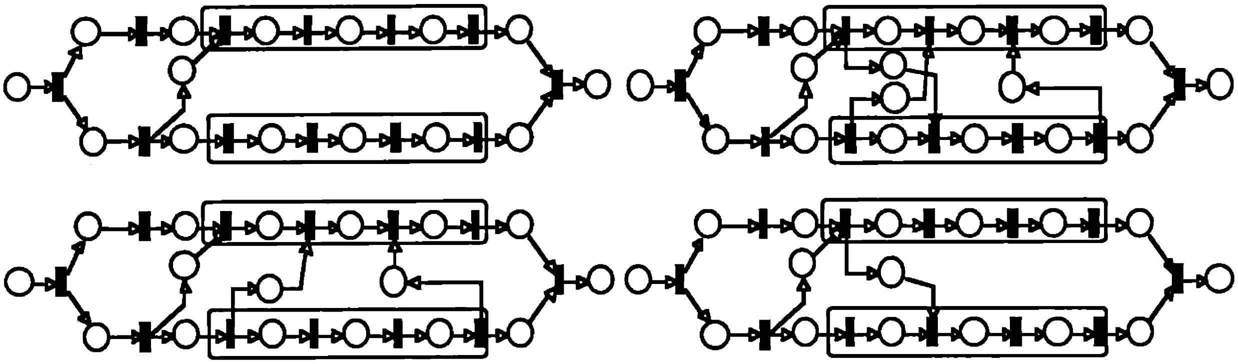


Fig. 6.3 Four WDFS

Over those four structures three links are variable. Two from the lower role to the upper role and one from the upper role to the lower role.

Let Π be a mapping from the set of inputs X to the set W. Π is an admissible variable structure (AVS) if it satisfies the following constraints

(R6) The support of Π is a subset of AW.

(R7) Any link in S and G that is admissible is permanent over the support of Π.

(R8) If the first stage of a role i is IF, then each input link F_{ji}, H_{ji} for j in [1..R] is permanent.

(R9) If the first stage of a role i is CI, then each input link C_{ji} for j in [1..R] is permanent

Finally, the last constraint R10 can be understood with the notions of Definition 6.2.

Definition 6.2

Let Σ be a fixed structure, and T be any process modeled by a transition t. The alphabet Xi is said to be *accessible* at t if and only if there is a path from Sensor i to transition t in the Petri Net that depicts Σ.

In other words, Xi is accessible at T if the information that is contained in the output of Sensor i can be used at the stage T. Conversely, if an alphabet Xi is not accessible at some stage T, then the stage cannot base its processing and pattern of interactions on the observations of Sensor i.

Example 6.2:

Fig 6.4 represents one fixed structure with three sensors and three roles. Each internal stage in the structure as well as the output stage have been annotated with the alphabets that are accessible at that stage. Some processes have no accessible alphabets, and have been annotated by ∅.

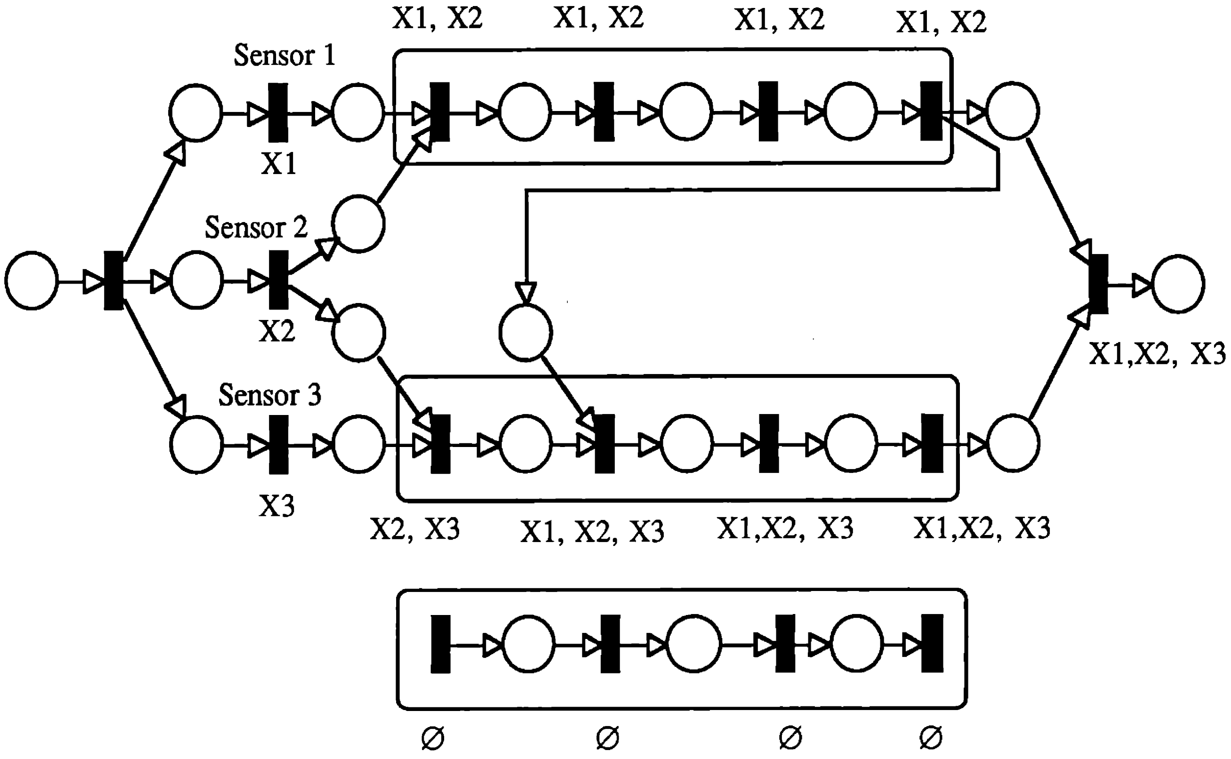


Fig. 6.4 Accessible Alphabets

Constraint R10 has to do with the partitioning of the set of inputs at variable links.

(R10) Let L be a variable link between two stages t1 and t2. Let AC = {x s.t. $\Pi(x)$ contains L}, DC = {x s.t. $\Pi(x)$ does not contain L} and $Xi_1,.., Xi_k$ be the effective alphabets of the partition of X into AC and DC.
The variable interaction that corresponds to L is properly coordinated
if and only if
The alphabets $Xi_1,.., Xi_k$ are accessible in every fixed structure $\Pi(x)$

- at t1
- at the internal stage that precedes t2.

The rationale for these constraints is as follows. Constraint R6 is just a reminder of the fact that the fixed structure associated with the processing of each input must be an admissible fixed structure. Constraints R7, R8, and R9 proceed from a common rationale. They simply state that a role at its input stage has no previous knowledge about the input to the system, and can only await information from sources over permanent links. Thus, at the SA stage, any link between the sensors and the roles must be fixed. Similarly, if a role receives the response from another role, the latter must always communicate its response (R7). Constraints R8 and R9 incorporate the fact that the input stage of a role can be the Information Fusion or Command Interpretation stages.

Constraint R10 states that a variable interaction between two stages t1 and t2 must be coordinated on sources of information that are accessed jointly by the roles that interact. The stage t1 must determine, based on some information it has processed, whether it has to send a message to t2. Similarly, the role that contains t2 must infer from some of the information it has already received, whether or not it must wait for a message from t1 before initiating process t2. Observe that the role containing t2 has received this information at the internal stage preceding t2.

The roles must be interacting based on the same information, which has been generated by the sensor's observations. The roles must thus recognize, from the same combinations of sensor's outputs, the existence or the absence of the interaction. Any variable interaction induces a partition of X into two subsets: The inputs that activate the link, i.e. {x s.t. $\Pi(x)$ contains the fixed interaction L} = AC; and the inputs that do not activate the link, i.e. {x s.t. $\Pi(x)$ does not contain the fixed interaction L} = DC. Proposition 4.9 indicates that $Xi_1,.., Xi_k$ are the effective alphabets if this interaction is based on the outputs of Sensor i_1, .., Sensor i_k. Constraint R10 insures that the information generated by the outputs of Sensor i_1,...,Sensor i_k is accessible in every dataflow structure at the stages where it is needed.

Example 6.3: Figures 6.5 and 6.6 represent two admissible fixed structures with one variable link, the link from the RS stage of role 1 to the IF stage of role 2.

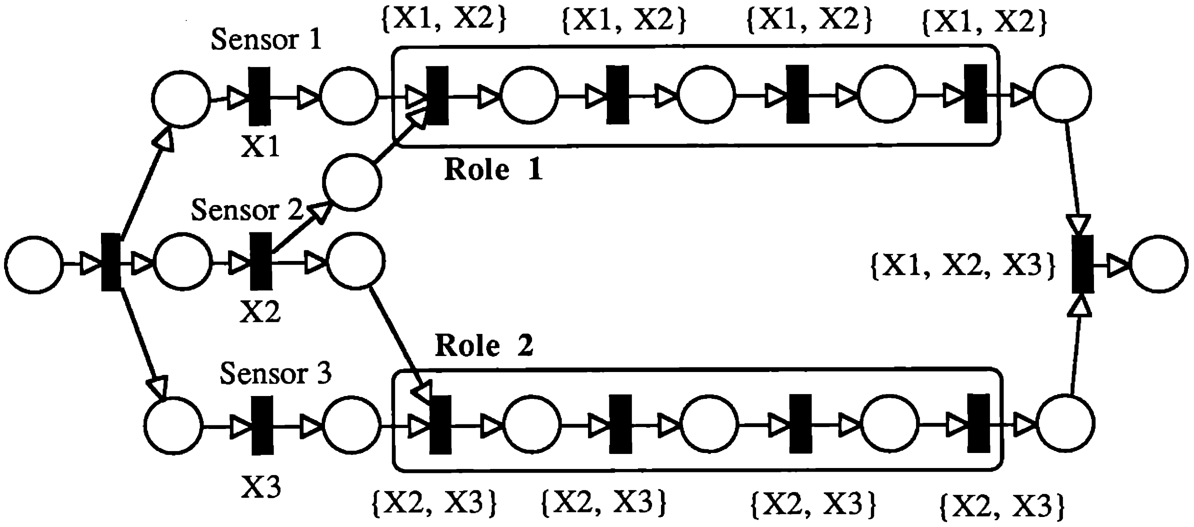


Fig. 6.5 Fixed Structure

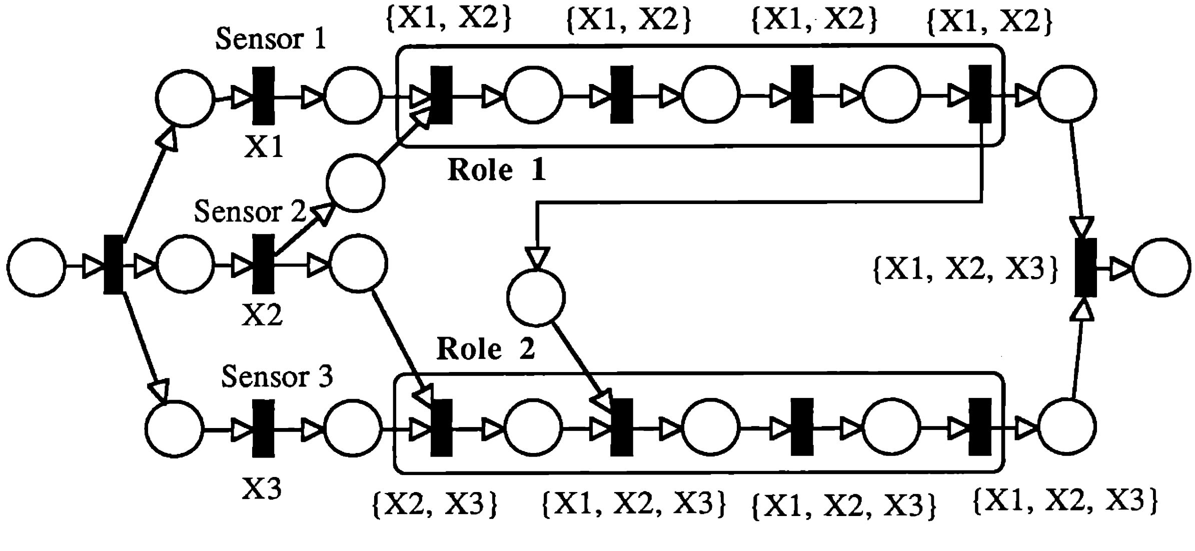


Fig. 6.6 Fixed Structure and Variable Link

The alphabets that are effective at each stage have been represented on each admissible fixed structures. Over both fixed structures, the RS stage of role 1 and the SA stage of role 2 have one and only one common admissible alphabet, X2. The partition of X at the level of the variable link must be of Type X2. In other words, the variability of the link can only be based on observations of Sensor S2, the only source of information that is accessed by both role 1 and role 2.

6.2.4 Mathematical Representation of the Constraints on V

The constraints can be translated into analytical constraints without any difficulty.

Constraint R6

$$\forall x \in X \quad \Pi(x) \in AW$$

Constraint R7

$$(\exists x \in X \text{ s.t. } S_{ij}(x) = 1) \Rightarrow (\forall x\, S_{ij}(x) = 1)$$
$$(\exists x \in X \text{ s.t. } G_{ij}(x) = 1) \Rightarrow (\forall x\, G_{ij}(x) = 1)$$

Constraint R8

$$(\forall i \ \ \forall x \ \ S_{ij}(x) = G_{ij}(x) = 0 \text{ and } \exists\, i_0 \,\exists x \text{ s.t. } F_{i0\,j}(x) = 1) \Rightarrow (\forall x\, F_{i0\,j}(x) = 1)$$
$$(\forall i \ \ \forall x \ \ S_{ij}(x) = G_{ij}(x) = 0 \text{ and } \exists\, i_0 \,\exists x \text{ s.t. } H_{i0\,j}(x) = 1) \Rightarrow (\forall x\, H_{i0\,j}(x) = 1)$$

Constraint R9

$$(\forall i \ \ \forall x \ \ S_{ij}(x) = G_{ij}(x) = F_{ij}(x) = H_{ij}(x) = 0 \text{ and } \exists\, i_0 \,\exists x \text{ s.t. } C_{i_0 j}(x) = 1)$$
$$\Rightarrow (\forall x\, C_{i_0 j}(x) = 1)$$

Finally, constraint R10 is expressed by

Let L be a link. $(\exists x \ \ \exists x' \ L(x) = 1 \text{ and } L(x') = 0) \Rightarrow$
$((\forall x \text{ in } X \ \ \exists \text{ path from Sensor } i \text{ to } L \text{ in } \Pi(x)) \Leftrightarrow (Xi \text{ can be effective}))$.

If a WDVS satisfies constraints R6 to R10 it is said to be an *Admissible Variable Structure* (AVS). The set of all AVSs is called AV.

6.2.5 Accessible Pattern

Constraint R10 is very important, because it relates topological properties of the fixed structures to the properties that must be fulfilled by the assignments. The notation to be introduced in Definition 6.3 provides an elegant tool to deal with those properties.

Definition 6.3

Let Σ be any admissible fixed structure in AW. Its *accessible pattern* is a set of arrays

$$E(\Sigma) = [\, F_e(\Sigma), H_e(\Sigma), C_e(\Sigma), s_e(\Sigma)].$$

- $F_e(\Pi)$, $H_e(\Pi)$, $C_e(\Pi)$ are three $R \times R$ arrays that correspond to the interactions between roles that can be variable.
- $s_e(\Pi)$ is an $R \times 1$ array that corresponds to the links from the roles to the effectors, which can be variable.
- The entry of each array is a subset of $\{X1, ..., XN\}$.
  - $F_e(\Sigma)_{ij}$ contains the alphabets that are accessible at the SA stage of role i and at the SA stage of role j in Σ.
  - $H_e(\Sigma)_{ij}$ contains the alphabets that are accessible at the RS stage of role i and at the SA stage of role j in Σ.
  - $C_e(\Sigma)_{ij}$ contains the alphabets that are accessible at the RS stages of role i and at the IF stage of role j in Σ.
  - $s_e(\Sigma)_{ij}$ contains the alphabets that are accessible at the RS stage of role i in Σ.

Remark : The diagonal entries are filled with $\varnothing$, because they do not correspond to any link in the system.

This notation is explicitly introduced to indicate for each interaction, whether or not it exists, the alphabets that are accessible at both end stages of the interaction. Therefore, if this interaction varies over a range of fixed admissible structures J_1 to J_k, the accessible pattern $E(J_i)$ indicates the constraints on the partition at each link that come from the AFS J_i. See below for a detailed statement. The fact that the interactions between sensors and roles, and between RS stages and SA stages, have been omitted is a result of the constraints R7 and R8, which state that these interactions must be fixed. There is no need to check effective and accessible alphabets for those links.

Example 6.4:

Let us consider the admissible fixed structure Σ of Figure 6.7. The set of alphabets that are accessible at each stage have been indicated on the net.

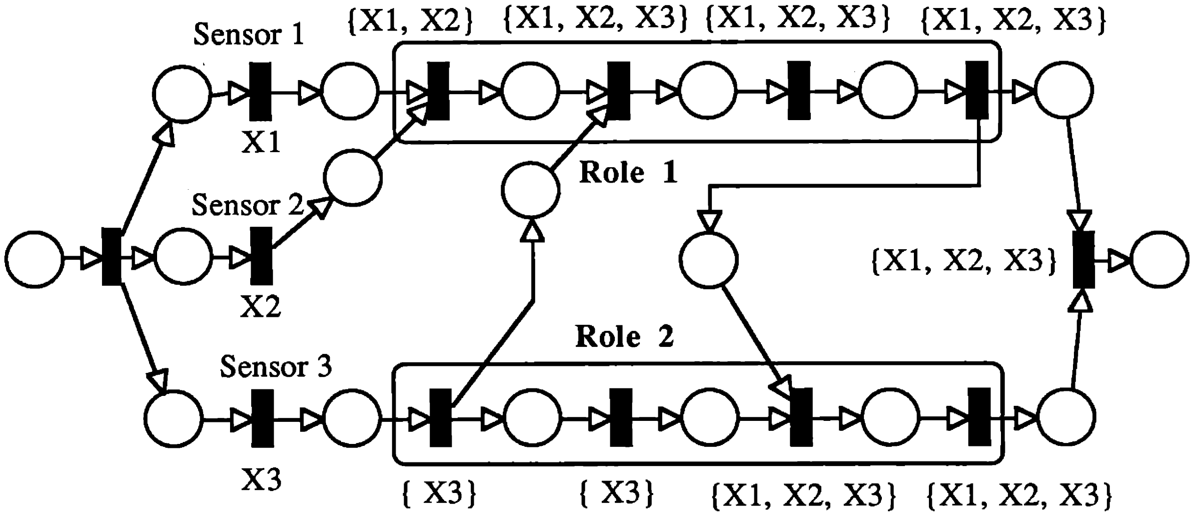


Fig. 6.7 Admissible Fixed structure Σ

Its accessible pattern is given by

$$F_e\,(\Sigma) = \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \varnothing \end{bmatrix},\quad H_e\,(\Sigma) = \begin{bmatrix} \varnothing & \{X3\} \\ \{X2,X3\} & \varnothing \end{bmatrix}$$

$$C_e\,(\Sigma) = \begin{bmatrix} \varnothing & \{X3\} \\ \{X1,\ X2,\ X3\} & \varnothing \end{bmatrix},\quad s_e\,(\Sigma) = \begin{bmatrix} \{X1,\ X2,\ X3\} \\ \{X1,\ X2,\ X3\} \end{bmatrix}.$$

- $F_e(\Sigma)$ indicates that no alphabet is accessible at both RS stages.
- $H_e(\Sigma)$ indicates that X3 is accessible at both the RS stage of Role 1 and the SA stage of role 2 and that X1 and X2 are accessible at both the RS stage of role 2 and the SA stage of role 1.
- $C_e(\Sigma)$ shows that X3 is accessible at both the RS stage of role 1 and the IF stage of role 2, while X1, X2 and X3 are accessible at both the RS stage of role 2 and

the IF stage of role 1.

- $s_e(\Sigma)$ indicates that X1, X2 and X3 are accessible at both RS stages.

6.2.6 Constraint R10 Revisited

As mentioned earlier, the accessible patterns are an elegant tool for providing another mathematical expression of the constraint R10. For that purpose, Definition 6.4 defines the intersection of accessible alphabets.

Definition 6.4

Let E (Σ) and E (Σ') be two accessible patterns. Their intersection E (Σ) INT E (Σ') is defined quite naturally as,

$$[F_e(\Sigma) \cap F_e(\Sigma'),\ H_e(\Sigma) \cap H_e(\Sigma'),\ C_e(\Sigma) \cap C_e(\Sigma'),\ s_e(\Sigma) \cap s_e(\Sigma')]$$

where $\cap$ denotes the usual intersection of subsets of {X1, X2,.., XN}.

Example 6.5:

The accessible pattern of the AFS of Example 6.4 is

$$F_e(\Sigma) = \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \varnothing \end{bmatrix},\quad H_e(\Sigma) = \begin{bmatrix} \varnothing & \{X3\} \\ \{X2,X3\} & \varnothing \end{bmatrix}$$

$$C_e(\Sigma) = \begin{bmatrix} \varnothing & \{X3\} \\ \{X1, X2, X3\} & \varnothing \end{bmatrix},\quad s_e(\Sigma) = \begin{bmatrix} \{X1, X2, X3\} \\ \{X1, X2, X3\} \end{bmatrix}.$$

Let us consider an other accessible pattern E(Σ').

$$F_e(\Sigma') = \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \varnothing \end{bmatrix},\quad H_e(\Sigma') = \begin{bmatrix} \varnothing & \{X3\} \\ \{X3\} & \varnothing \end{bmatrix}$$

$$C_e(\Sigma') = \begin{bmatrix} \varnothing & \{X1, X2, X3\} \\ \{X1, X2, X3\} & \varnothing \end{bmatrix},\quad s_e(\Sigma') = \begin{bmatrix} \{X1, X2, X3\} \\ \{X1, X2, X3\} \end{bmatrix}$$

Then, E (Σ) INT E (Σ') is

$$F_e = \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \varnothing \end{bmatrix}, \; H_e = \begin{bmatrix} \varnothing & \{X3\} \\ \{X3\} & \varnothing \end{bmatrix}$$

$$C_e = \begin{bmatrix} \varnothing & \{X3\} \\ \{X1, X2, X3\} & \varnothing \end{bmatrix}, \; s_e = \begin{bmatrix} \{X1, X2, X3\} \\ \{X1, X2, X3\} \end{bmatrix}$$

Constraint R10 can be restated under the following form, which will be useful in Chapter 7.

> If {J1, .., Jk} is the support of a AVS, then at each variable link L the effective alphabets of the partition of X are included in the entry of $E(J_1)$ INT $E(J_2)$ INT...INT $E(J_k)$ that corresponds to L.

In other words, the intersection of the supports determines the kind of interactions, as indicated by the effective alphabets, that are allowed between two stages.

6.3 USER-DEFINED CONSTRAINTS

The members of the team in charge of Research and Development can introduce constraints that reflect what they know about the structure under study. They may rule in or rule out some links, force a certain pattern of variability, or express hierarchical relationships between the roles. For example, due to a particular expertise, the team might like to indicate that a certain set of observations can only be processed by some roles, while the processing of other sets of observations can be carried out by any role. Within the framework of this thesis, the designer can translate his knowledge by filling 0s and 1s at the appropriate places in the arrays S, s, F, G, H, C. The other elements will remain unspecified and will constitute the degrees of freedom of the design. A designer can impose two types of conditions.

6.3.1 Fixed Constraints

Fixed constraints are the constraints that are valid for any input x in X. They can be of two types, ruling in or ruling out. If a designer wants to rule out some links, he can do so either by putting the $|X| \times |X|$ null matrix O in the appropriate entry of Π = (S, s, F, G, H, C) or by ruling out the link for every Admissible Fixed Structure. If a designer wants to rule in some links, to insure that those links are always activated, he can do so either by putting the $|X| \times |X|$ identity matrix I in the appropriate entry of Π = (S, s, F, G, H, C) or by imposing this link on any Admissible Fixed Structure. It is important to observe at that point that any constraint on a link from a sensor to the input stage of a role, or on a link from the output stage of a role to the input stage of another role, is necessarily a fixed constraint because of R7, R8 and R9. In the rest, the set of fixed constraints is called R_F. This set is a set of links and the indication as to whether they have been ruled in or ruled out.

6.3.2 Colored Constraints

A designer might also want to impose constraints on the variability in the system. He might want to rule out a link for some set of observations and rule it in for some other set of inputs. This can be done by filling the appropriate $|X| \times |X|$ matrix L in Π = (S, s, F, G, H, C). It is assumed that a designer specifies a variable link completely. In other words, the designer specifies a partition of X into two subsets: The set of inputs that activates the link, and the set of inputs that does not activate the link. The thesis rules out the possibility that the designer specifies that the link is activated by the set of inputs A, is not activated by the set of inputs B, and remains unspecified for a non empty set of inputs C. In the sequel, the set of colored constraints is called R_C. This set corresponds to a set of links and a partition of the set of inputs X for each link.

6.4 ILLUSTRATION

In this section, a simple example is developed to illustrate the use of the model as well as the specification of constraints by the designer of a system. This example comes from the experience of a platoon commander in a unit of tanks. A platoon of tanks is a variable structure system with two different roles, that is the role of the platoon leader and the role of the subordinate. The degree of redundancy of the subordinate role in most armies is 2. The platoon receives observation from three sources S1, S2, and S3.

S1 describes the battlefield from 200 m up to 2.5 km. The battlefield is observed by the the platoon leader and both subordinates. The output alphabet of S1 is X1 {Enemy, NoEnemy}, where "Enemy" describes the fact that some enemy, either Armor or Infantry, is in sight. The output "NoEnemy" indicates that no enemy troops can be discerned. If an enemy is present in the battlefield, the platoon leader must issue a command to both subordinates. The platoon leader does not issue a command if the output is "NoEnemy," unless otherwise required.

S2 describes the battlefield in the immediate surroundings. This sector is monitored by at least one of the subordinates, but not by the platoon leader. The output alphabet of S2 is X2 {Infantry, NoInfantry}, where "Infantry" indicates that enemy forces, usually infantry, can be detected within a small shooting distance. The output "NoInfantry" indicates that no enemy groups have been found.

S3 describes the air cover. This sector is monitored at least by the platoon leader and the subordinate which is not in charge of the immediate surroundings. The output alphabet of S3 is {Chopper, NoChopper}, because helicopters are the major air threats for tanks. In case of an air threat, the subordinate informs the platoon leader of its situation assessment and waits for a command.

The set of simultaneous observations is $X1 \times X2 \times X3$. The set X contains eight elements

<Enemy, Infantry, Chopper>, <Enemy, Infantry, NoChopper>,
<Enemy, NoInfantry, Chopper>, <Enemy, NoInfantry, NoChopper>,
<NoEnemy, Infantry, Chopper>, <NoEnemy, Infantry, NoChopper>,
<NoEnemy, NoInfantry, Chopper>, <NoEnemy, NoInfantry, NoChopper>.

Those elements have been ordered by the lexicographic ordering. They have been abbreviated in the sequel by <E, I, C>, <E, I, NC >, <E, NI, C >, <E, NI, NC >, < NE, I, C >, < NE, I, NC >, < NE, NI, C >, < NE, NI, NC >.

Once a response has been selected each role gives appropriate commands to the driver and the shooter in each tank, which are the effectors in this system. The requirements for the system can be translated as follows. Throughout the specifications a "#" denotes that a link has not been specified, i.e. the link is a degree of freedom of the design, a "I" denotes the 8×8 identity matrix, i.e. the link is fixed, and a "0" denotes the 8×8 null matrix, i.e. the link has been ruled out. Platoon leader has been abbreviated by PL, subordinate 1 by Sb1, subordinate 2 by Sb2.

Constraints on S

The matrix S is a 3×3 block matrix. It is given by

$$S = \begin{array}{c@{}l} \begin{array}{ccc} PL & Sb1 & Sb2 \end{array} & \\ \begin{bmatrix} I & I & I \\ 0 & I & \# \\ I & \# & I \end{bmatrix} & \begin{array}{l} \text{Sensor 1} \\ \text{Sensor 2} \\ \text{Sensor 3} \end{array} \end{array}$$

The links are fixed because of constraint R7. The first row indicates that the battlefield is monitored by the three roles. The second row indicates that the immediate surroundings are not monitored by the platoon leader, that they are monitored by subordinate 1, and that subordinate 2 may or may not monitor the immediate surroundings. The third row shows that the air cover is monitored by the platoon leader and subordinate 2, while subordinate 1 might or might not look for choppers.

Constraints on s

Each role has to send a command to some effectors over all set of inputs, therefore s is given by

$$s = \begin{array}{ccc} PL & Sb1 & Sb2 \\ [\,I & I & I\,] \end{array}$$

Constraints on F

Subordinate 2 must inform the platoon leader of its situation assessment in case an air threat is detected. The set of inputs that activates this link is thus

$<C>$ = { <E, I, C>, <E, NI, C >, < NE, I, C >, < NE, NI, C >} and the set of inputs that do not activate the link is

$<NC>$ = { <E, I, NC >, <E, NI, NC >, < NE, I, NC >, < NE, NI, NC >}

Therefore the variability of the link from the SA stage of Subordinate 2 to the IF stage of the Platoon leader is described by the 8 × 8 matrix L1

$$L1 = \begin{array}{c@{}l} \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0 \end{bmatrix} & \begin{array}{l} \text{<E, I, C>} \\ \text{<E, I, NC>} \\ \text{<E, NI, C>} \\ \text{<E, NI, NC>} \\ \text{<NE, I, C>} \\ \text{<NE, I, NC>} \\ \text{<NE, NI, C>} \\ \text{<NE, NI, NC>} \end{array} \end{array}$$

No other constraints on the sharing of situation assessment have been formulated, thus F is

$$F = \begin{array}{cccc} \text{PL} & \text{Sb1} & \text{Sb2} & \\ 0 & \# & \# & \text{PL} \\ \# & 0 & \# & \text{Sb1} \\ \text{L1} & 0 & \# & \text{Sb2} \end{array}$$

Constraints on G

No constraints on G have been stated explicitly. However one can observe that the existence of the links from Sensors to SA stages rules out those links (Constraint R4). Therefore, if the designer has entered its parameters of the design for the matrix s as described above, the parameters on S can be automatically converted into the following matrix G. The constraints on G are thus as follows

$$G = \begin{array}{cccc} \text{PL} & \text{Sb1} & \text{Sb2} & \\ 0 & 0 & 0 & \text{PL} \\ 0 & 0 & 0 & \text{Sb1} \\ 0 & 0 & 0 & \text{Sb2} \end{array}$$

Constraint on H

No constraint on the interactions between the RS stages and the IF stages has been stated. Nevertheless, the designer wants to study structures in which no response can be analyzed by the subordinates. The constraints on H become

$$H = \begin{array}{cccc} \text{PL} & \text{Sb1} & \text{Sb2} & \\ 0 & \# & \# & \text{PL} \\ 0 & 0 & 0 & \text{Sb1} \\ 0 & 0 & 0 & \text{Sb2} \end{array}$$

Constraint on C

Several constraints that have been expressed are translated into the entries of matrix C. First, the platoon leader must issue a command to both subordinates if the output of S1, the main battlefield is "Enemy." Second, the platoon leader must issue a command to Subordinate 2 if some helicopter is detected. Since the platoon leader coordinates the mission assigned to the platoon, it is assumed that both subordinates cannot issue commands to the platoon leader. However each subordinates may issue commands to the other. The constraint on the matrix C can thus be expressed by

$$C = \begin{array}{ccc|c} \text{PL} & \text{Sb1} & \text{Sb2} & \\ 0 & L2 & L3 & \text{PL} \\ 0 & 0 & \# & \text{Sb1} \\ 0 & \# & 0 & \text{Sb2} \end{array}$$

where L2 describes the variability of the link from PL to Sb1. It corresponds to the platoon leader issuing a command to Sb1 if and only if the output of Sensor 1 is E. The partition of X is thus A = <*E*> and B= <*NE*> , the only effective alphabet is X1 and L2 is as follows.

$$L2 = \left[\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \begin{array}{l} \langle E, I, C\rangle \\ \langle E, I, NC\rangle \\ \langle E, NI, C\rangle \\ \langle E, NI, NC\rangle \\ \langle NE, I, C\rangle \\ \langle NE, I, NC\rangle \\ \langle NE, NI, C\rangle \\ \langle NE, NI, NC\rangle \end{array}$$

Matrix L3 incorporates the variability between PL and Sb 2. A command is issued to Sb2 if and only if Sensor 1 outputs "Enemy" or Sensor 3 outputs "Chopper." Thus, the set of inputs that activate the link is <E> ∪ <C> = { <E, I, C>, <E, I, NC >, <E, NI, C >, <E, NI, NC >, < NE, I, C >, < NE, NI, C >} and the set of inputs that do not activate the links is <NE> ∩ <NC> = {< NE, I,N C >, < NE, NI, NC >}. It must be noted that the effective alphabets of this partition are X1 and X3. The matrix L3 is thus as follows

$$L3 = \left[\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \begin{array}{l} \langle E, I, C\rangle \\ \langle E, I, NC\rangle \\ \langle E, NI, C\rangle \\ \langle E, NI, NC\rangle \\ \langle NE, I, C\rangle \\ \langle NE, I, NC\rangle \\ \langle NE, NI, C\rangle \\ \langle NE, NI, NC\rangle \end{array}$$

It is easy to show those constraints on a CPN model of the system. Figure 6.8 depicts a model of the variable structure platoon command and control system. This net is constructed as shown in Chapter V. To facilitate the expression of the constraints on the graph, a link that has been ruled out has not been represented, a link that is fixed is drawn with a bold line and without annotations, and a link that is variable is drawn in bold. A variable link is annotated by the matrix that describes its activation over the set of inputs. Finally, the unspecified links have been indicated and are drawn with plain lines.

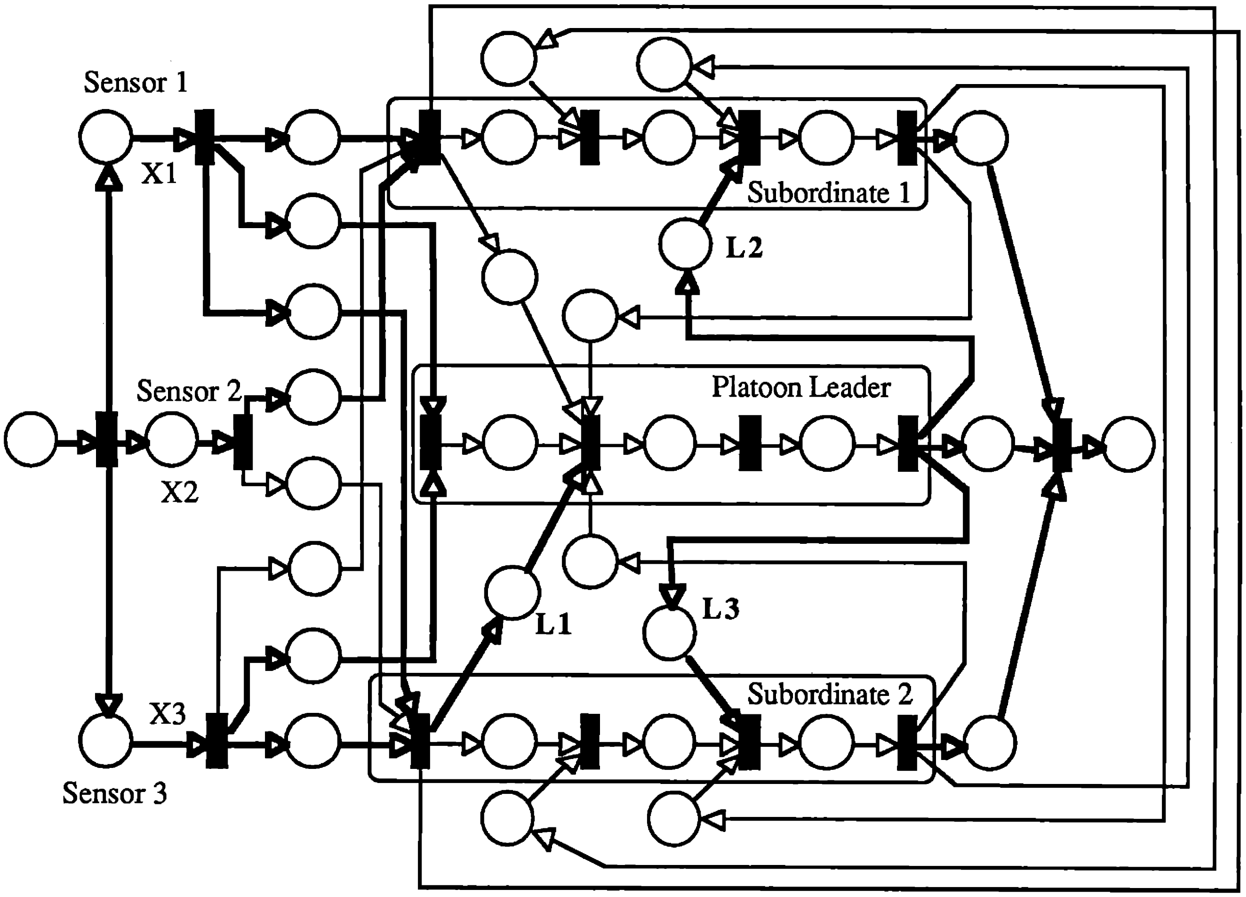


Fig. 6.8 Colored Petri Net of the User's Constraints

Chapter VIII contains two detailed applications of the methodology. In the next chapter, the set of solutions to the design problem is characterized.

CHAPTER VII

SOLUTIONS TO THE DESIGN PROBLEM

7.1 INTRODUCTION

Table 7.1 summarizes the sets and notions that have been introduced. A set of Well Defined Variable Structures, V, has been defined. This is the most generic set as far as variable structures are concerned. It is partially ordered by ACT and (V, ACT) is a lattice. The designers express their knowledge about the system within the framework of V, by imposing the constraints R_F and R_C. Structural constraints have been introduced to characterize the set AV of WDFS that makes physical sense, the Admissible Variable Structures. This set is partially ordered by ACT, and none of its properties have been stated. A solution to the design problem is any AVS that satisfies the user-defined constraints R_F and R_C.

Table 7.1 Mathematical Framework

Sets	Ordering	Properties	Remark
V	ACT	Lattice	Constraints R_F and R_C
AV	ACT	?	
Solutions	ACT	?	Solution = AV and R_F and R_C
W	SUB	Lattice	Constraints R_F
AW	SUB	?	Accessible patterns

In order to use the language and the results from the work on fixed structures, a set W of Well Defined Fixed Structures has been described. This is the most generic set as far as fixed structures are concerned. The connection between V and W is that any element of V can be represented as a mapping from the set of inputs X to the set W. Structural constraints on V restrict the class of possible mappings to those from X to the subset of Admissible Fixed Structures, AW. The fixed constraints R_F rule in or rule out some links in W. This chapter presents some properties of the set of solutions. First, the colored constraints are analyzed in section 7.2. Then, the convexity of a property is defined in section 7.3, and used in section 7.4 to characterize the WDVS that satisfy all constraints except constraint R10. Constraint R10 is studied in detail in section 7.5, and yields a specification of a class of solutions.

7.2 COLORED CONSTRAINTS

7.2.1 Correlation of Activations

Suppose that the designer specified the variability of k links $L_1, ..., L_k$. Each link L_i induces a partition of X into two subsets, AC_i, the inputs that activate the link, and DC_i, the inputs that do not activate the link. The activations of these links are not independent. If a link L is activated by the elements of AC, and another link L' is activated by the elements of AC', then the fixed functional structure associated with each input in AC $\cap$ AC' must have both fixed links L and L', the data flow structure associated with every input in DC $\cap$ AC' must have the fixed link L' but not the fixed link L, the fixed functional structure associated with every input in AC' $\cap$ DC must have the fixed link L but not the fixed link L", and the data flow structure associated with each input in DC $\cap$ DC' must not have the fixed links L and L'. The correlation of the activations is analyzed with the properties introduced in Definition 7.1.

Definition 7.1

Let Σ be any WDFS. For i in [1..k], the properties R_i^0 and R_i^1 are defined by

$R_i^1(\Sigma)$ is true if and only if Σ contains the fixed link L_i.

$R_i^0(\Sigma)$ is true if and only if Σ does not contain the fixed link L_i.

Note that for any i and any Σ, either $R_i^0(\Sigma)$ or $R_i^1(\Sigma)$ holds. R_i^0 and R_i^1 define a partition of W into two subsets, W_i^0, the set of WDFS for which R_i^0 holds, and W_i^1, the set of WDFS for which R_i^1 holds. From the definition of AC_i and DC_i, one infers Proposition 7.1.

Proposition 7.1

For any Π in V, and any i in [1..k], Π describes the variability of the link L_i if and only if

- $\forall$ x in AC_i $\quad$ $\Pi(x)$ belongs to W_i^1
- $\forall$ x in DC_i $\quad$ $\Pi(x)$ belongs to W_i^0

Schematically, Proposition 7.1 is illustrated by Figure 7.1.

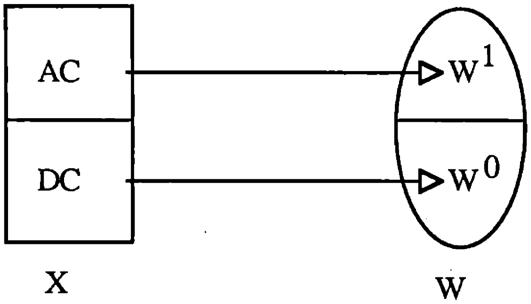


Fig. 7.1 Constraint on Mapping

The set of inputs X is divided into two subsets, as is the set W. The inputs that activate the variable link L (the inputs in AC) must be assigned to fixed structures that contain the fixed link L (fixed structures in W^1), and the inputs that do not activate the variable link L (the inputs in DC) must be assigned to fixed structures that do not contain the fixed link L (fixed structures in W^0). The fact that the activation of links is correlated is illustrated in Figure 7.2. There is no reason to believe that the variability of the links $L_1,..., L_k$ induces the same partitions of X and W. One must thus consider the intersection of these partitions to keep track of the constraints on the mappings, as described in Proposition 7.2.

Proposition 7.2

A WDVS Π describes the variability of any link L_i, i in [1..k], if and only if

$$\forall\ (i_1, ..., i_k) \text{ in } \{0, 1\}^k\ \forall\ x \text{ in } \Big(\bigcap_{i_j = 1, j = 1..k} AC_{i_j}\Big) \cap \Big(\bigcap_{i_j = 0, j=1..k} DC_{i_j}\Big)$$

$$\Pi(x) \text{ belongs to } \Big(\bigcap_{i_j = 1, j=1..k} W^1_{i_j}\Big) \cap \Big(\bigcap_{i_j = 0, j=1..k} W^0_{i_j}\Big)$$

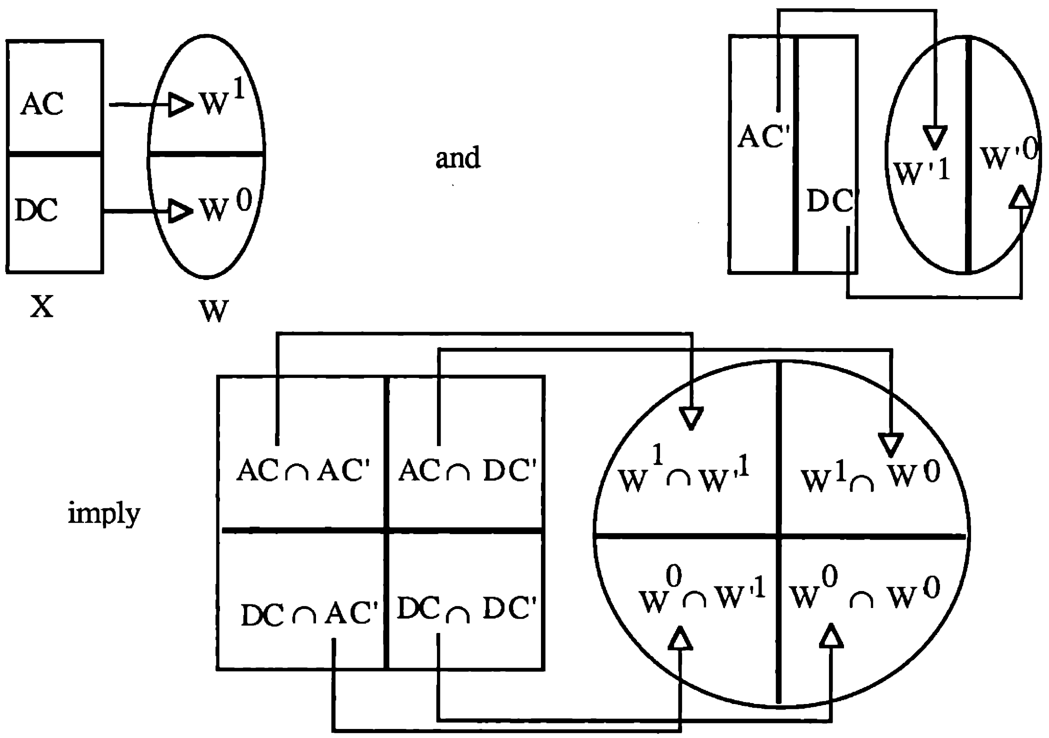


Fig. 7.2 Correlation of Colored Constraints

Proof:

The colored constraints induce k partitions of the set of inputs into AC_i and DC_i, and k partitions of W into W^1_i and W^0_i. Thus,

- the set of inputs is partitioned into the subsets
  $(\bigcap_{i_j = 1, j = 1..k} AC_{i_j}) \cap (\bigcap_{i_j = 0, j=1..k} DC_{i_j})$ with $(i_1, ..., i_k)$ in $\{0, 1\}^k$
- the set of fixed structures is partitioned into the subsets
  $(\bigcap_{i_j = 1, j=1..k} W^1_{i_j}) \cap (\bigcap_{i_j = 0, j=1..k} W^0_{i_j})$ with $(i_1, .., i_k)$ in $\{0, 1\}^k$

  and by Proposition 7.1, any element in $(\bigcap_{i_j = 1} AC_{i_j}) \cap (\bigcap_{i_j = 0} DC_{i_j})$ is assigned to an element of $(\bigcap_{i_j = 1} W^1_{i_j}) \cap (\bigcap_{i_j = 0} W^0_{i_j})$.

Example 7.1:

Suppose that a system has two sensors, whose alphabets are X1 = {A, B, C} and X2 = {U, V}. Suppose that the colored constraints apply to two links L1 and L2, and $AC_1 = \{<A>\}$, $DC_1 = \{<B> \cup <C>\}$, $AC_2 = \{<U>\}$, $DC_2 = \{<V>\}$. Then,

- Any element of $< A > \cap < U > = \{ <A, U> \}$ must be assigned to an element of $W^1{}_1 \cap W^1{}_2$.
- Any element of $< A > \cap < V > = \{<A, V>\}$ must be assigned to an element of $W^1{}_1 \cap W^0{}_2$.
- Any element of $(<B> \cup <C>) \cap < U > = \{<B, U>, <C, U>\}$ must be assigned to an element of $W^0{}_1 \cap W^1{}_2$.
- Any element of $(<B> \cup <C>) \cap < V > = \{<B, V>, <C, V>\}$ must be assigned to an element of $W^0{}_1 \cap W^0{}_2$.

Note, however, that some subsets $(\bigcap_{i_j = 1} AC_{i_j}) \cap (\bigcap_{i_j = 0} DC_{i_j})$ can be empty.

An empty intersection does not impose any constraint on the mapping, and can thus be ignored. In the rest, only the non-empty intersections are considered. To keep the notations simple, it is assumed that there are m such intersections, EX_i i in [1..m], which are called the *elementary sets of inputs*. Each element in EX_i must be assigned to an element of W^i, where W^i is

$$(\bigcap_{i_j = 1} W^1_{i_j}) \cap (\bigcap_{i_j = 0} W^0_{i_j}) \text{ for some appropriate } (i_1, .., i_k) \text{ in } \{0,1\}^k .$$

7.2.2 Solutions of Colored Constraints

Any mapping such that $EX_1 \text{ ----> } W^1, ..., EX_m \text{----> } W^m$ satisfies the colored constraints. However, this thesis characterizes only one class of solutions, as expressed by Assumption 7.1.

Assumption 7.1

Within the scope of this thesis, only mappings that assign all elements of EX_i to the *same fixed structure* in W^i are considered.

Example 7.2:

A solution to the design problem of example 7.1 is thus a mapping such that

- $<A, U>$ is assigned to an element of $W^1{}_1 \cap W^1{}_2$, Σ^1.
- $<A, V>$ is assigned to an element of $W^1{}_1 \cap W^0{}_2$, Σ^2
- $<B, U>$ and $<C, U>$ are both assigned to the same Σ^3 in $W^0{}_1 \cap W^1{}_2$.
- $<B, V>$ and $<C, V>$ are both assigned to the same Σ^4 in $W^0{}_1 \cap W^0{}_2$.

This assumption proceeds from one main consideration, which is the need for a WDVS whose support has a minimal number of elements. By describing the variability of k links, the user imposes certain correlations (different W^i and EX_i). It has been noted that $W^1...W^m$ are distinct subsets of W. Therefore, the ranges in W of EX_1,.., EX_m are distinct, and there are at least m distinct fixed structures in the support of every WDVS that satisfy the colored constraints. Assumption 7.1 restricts the scope of this study exactly to the WDVS with m elements. The hypothesis is that the coordination of a minimal number of fixed structures is easier than the coordination of a large number of fixed structures. (See Proposition 7.20 for a mathematical expression of that statement). This thesis characterizes the solutions that correspond exactly to the degree of correlation, as defined by the size of the support, that is introduced by the designer. Proposition 7.2 and Assumption 7.1 are summed up by:

Proposition 7.3

Any WDVS that satisfies the colored constraints and Assumption 7.1 is characterized by

m Well Defined Fixed Structures Σ^1,.. ,Σ^m,

where m is the number of non empty effective sets of inputs. The effective set EX_i is attached to Σ^i, for any i in [1..m].

In the rest of this subsection, some consequences of Proposition 7.3 are stated. They relate Proposition 7.3 to properties of the fixed structures in the support of a WDVS.

Proposition 7.4

Let Π and Π' be two WDVS that satisfy the colored constraints

Π ACT Π'

if and only if

for any i in [1..m], Σ^i SUB Σ'^i.

Proof:

Subsection 4.4. noted that Π ACT Π' if and only if $\forall$ x in X $\Pi(x)$ SUB $\Pi'(x)$.

This is equivalent to stating that for any i in [1..m] and for every x in EX_i ,

$\Pi(x)$ SUB $\Pi'(x)$, with $\Pi(x) = \Sigma^i$ and $\Pi'(x) = \Sigma'^i$ Q.E.D.

In other words, Π ACT Π' if and only if the data flow structure that describes the processing of the elementary set of inputs EX_i in Π is a subnet of the corresponding data flow structure in Π'. It is equivalent to work with the partial ordering ACT on V, or to work with the partial ordering SUB in the k distinct subsets W^i, i in [1..k]. This result is used later on to characterize the AVS that satisfy the user-defined constraints.

Proposition 7.5

Let Π be a WDVS that satisfies R_C, EX_1, ..., EX_m be the effective alphabets, and L be any link in Π.

The subset of inputs that activates the link L, AC, belongs to the lattice polynomial generated in the lattice of all subsets of X by EX_1, ..., EX_m, by constructing all possible unions and intersections of the disjoint sets EX_1, ., EX_m.

Note that the lattice polynomial generated by a finite number of elements of a lattice is introduced in Definition 3.9. This sublattice is denoted by $L(EX_1, ..., EX_m)$.

Proof:

Let L be a variable link of a WDFS that satisfies the colored constraints. Let I be {i in [1..m] s.t. L is present in Σ^i }, which might be the empty set. By definition, the set of inputs that activate the link is $\cup_{i \text{ in } I} EX_i$, which belongs to $L(EX_1, ., EX_m)$.

Proposition 7.5 is important because the sublattice $L(EX_1, ., EX_m)$ is valid for any link. Therefore, a practical implementation of the methodology need not recompute the values that can be taken by AC for every link. A lot of computing time can be saved by determining the lattice $L(EX_1, ., EX_m)$ once and for all, and translating it into a lattice of matrices that appropriately describe the activation of the links, as discussed in Chapters V and VI.

7.3 CONVEXITY

Section 7.2 described some relationships between the colored constraints and the set of solutions to the design problem. In order to characterize the set of solutions, an extensive investigation of the interactions between the constraints and the partial orderings ACT and SUB must be performed. The notion of convexity is one appropriate mathematical tool to be applied.

Definition 7.2

Let A be a set, R be an ordering of A. If x_1 and x_2 are two elements such that x_1 R x_2, the *interval* $[x_1, x_2]$, if it exists, is defined to be the subset A' of A such that every x in A' satisfies x_1 R x R x_2.

Definition 7.3

Let Y be a subset of the partially ordered set A. Y is said to be *convex* if and only if

$$\forall (y_1, y_2) \in Y^2 \quad (y_1 R y_2) \Rightarrow ([y_1, y_2] \subset Y)$$

For example, the set of real numbers with the ordering ≤ is a convex set. Note that the convexity is a property that is specific to the subset Y. It is possible to have a Y that is a convex subset of a non convex set A. Reciprocally, a convex set can have non convex subsets. Much study has been devoted to convex subsets because they have the unique property of being completely determined by their minimal and maximal elements, as indicated by Proposition 7.6. Minimal and maximal elements of a partially ordered set have been introduced in Chapter III.

Proposition 7.6 (Birkhoff)

Let Y be a convex subset of a set A. Then

- "x belongs to Y" is equivalent to
- "x is a minimal element of Y" or "x is a maximal element of Y" or "$\exists$ (x_1, x_2), where x_1 is a minimal element in Y, and x_2 is a maximal element in Y, such that x_1 R x R x_2".

If a set is convex, its structure can be assessed with three simple sets of tools, a partial ordering, a set of minimal elements and a set of maximal elements. Any element that is below one maximal element and above one minimal element belongs to the set. There is no need for an extensive, and possibly combinatorial, description of all the elements. The issue of finding convex subsets in the set of WDVS, V, appears quite important, because convexity allows us to describe exactly subsets of V without encountering a combinatorial computational problem. In that case the designer can be given the set of solutions within minimal computing requirements. Boolean properties are considered to tackle rigorously this goal. A *Boolean property* on a set A is a proposition P(x) that either holds (is equal to 1) or does not hold (is equal to 0), for any x in A. Let P(A) be the set of elements of A such that P(x) holds. If A is a partially ordered set, there exist P_{max} (A), the set of all maximal elements of P(A), and P_{min} (A), the set of all minimal elements of P(A).

Definition 7.4

A Boolean property P is said to be *convex* with respect to the partial ordering R if and only if P(A) is convex.

Proposition 7.7

If a Boolean property P is convex and verified by at least one element then

$P(A) = \{ x \text{ in } A \text{ s.t. there exist } x_1 \text{ in } P_{min}(A) \text{ and } x_2 \text{ in } P_{max}(A) \text{ and } x_1 R x R x_2 \}$

Proposition 7.7 is a direct consequence of Proposition 7.6, and is important because it indicates that a set of elements that fulfills a convex property is uniquely determined by the boundary elements of that set. Here again, one can gain substantive insights into a set by reducing a combinatorial description of all the elements that verify the property into a description of the minimal and maximal elements of that set. Within the scope of this thesis, elements of A that may simultaneously satisfy several convex properties must also be characterized.

Proposition 7.8

The intersection of two convex sets is a convex set.

Proof:

Let A and B be two convex sets. If A and B have no elements in common, then their intersection is the empty set, which is trivially a convex set.

Let us suppose that their intersection is non empty. Let x be an element of the intersection. If x is minimal or maximal in the intersection, the proof is done. If x is neither maximal nor minimal in the intersection, there exist two elements in the intersection x_1 and x_2 s.t. x_1 R x R x_2. Because of the convexity of A and B, x covers x_1 and x_2 covers x. The proof proceeds inductively on x_1 and x_2.

Therefore, the set of elements of A that simultaneously satisfy several convex properties is uniquely characterized by its boundaries. This result has some drawbacks, however. Suppose that there are j convex properties P_1,.., P_j. Each set $P_j(A)$ is determined by its boundaries, but there are, in general, no direct relationships between minimal and maximal elements of the intersection $\cap_j P_j(A)$ and minimal and maximal elements of each $P_j(A)$. Nevertheless, if (A, R) is a lattice, Proposition 7.9 holds.

Proposition 7.9

Let (A, R) be a lattice. Let P, P' be two convex properties, and $P_{max}(A) = \{x_1,..., x_n\}$, $P_{min}(A) = \{y_1,..., y_p\}$, $P'_{max}(A) = \{z_1, ..,z_q\}$, $P'_{min}(A) = \{w_1,.., w_t\}$.

- $(P \text{ and } P')_{min}(A)$, the set of minimal elements satisfying P and P' is included in the set of the joins of the elements of $P_{min}(A)$ and $P'_{min}(A)$: $\{ y_i \cup w_j\}_{i,j} \supseteq (P \text{ and } P')_{min}(A)$.
- $(P \text{ and } P')_{max}(A)$, the set of maximal elements satisfying P and P' is included in the set of the meets of the elements of $P_{max}(A)$ and $P'_{max}(A)$: $\{x_k \cap z_m\}_{k,m} \supseteq (P \text{ and } P')_{max}(A)$.

Proof:

If x is an element of A that verifies P and P' then,
y_i R x R x_k, with y_i in $P_{min}(A)$, x_k in $P_{max}(A)$
w_j R x R z_m, with w_j in $P'_{min}(A)$, z_m in $P'_{max}(A)$.
Thus, by definition of the meet and the join, $y_i \cup w_j$ R x R $x_k \cap z_m$. Therefore, any element of (P and P') (A) lies above a meet of minimal elements and below a join of maximal elements. Note that all meets and joins might not be minimal and maximal elements of the intersection.

On the other hand, if no y_i, w_j, x_k, z_m verify $(y_i \cup w_j)$ R x R $x_k \cap z_m$, then (P and P')(A) is empty, and if some y_i, w_j, x_k, z_m verify $y_i \cup w_j$ R x R $x_k \cap z_m$ then $y_i \cup w_j$ and $x_k \cap z_m$ belong to (P and P')(A), and are thus boundaries of (P and P')(A).

Proposition 7.9 is more important practically than theoretically. From the theoretical point of view, knowing that an intersection of convex sets is convex is the substantive result. In a practical implementation, however, it is difficult to compute rapidly the minimal and maximal elements of an intersection of several convex sets. Proposition 7.9 indicates that working in a lattice, which is the case in V and W, eases the computation, because of the simple relationship between minimal and maximal elements of the intersection and the boundaries of the convex sets.

7.4 CHARACTERIZATION OF THE CONSTRAINTS

This section applies the results of 7.3 to characterize the constraints. A solution to the design problem is a WDVS such that R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R_F, R_C, and Assumption 7.1 are satisfied. Each of these constraints is a Boolean property, because a constraint is either fulfilled or violated by a WDVS. However, some constraints have been expressed in the language of the set W, while others have been expressed in the language of the set V. The constraints on the set W are analyzed first. They yield a characterization of the set AW. Then, this result is combined with the constraints on V, except R10.

7.4.1 Constraints R2, R3, R4, R5

The constraints R2, R3, R4, R5 have been expressed as constraints in W. They do not pose any problems as far as convexity is concerned, as indicated by Proposition 7.10.

Proposition 7.10

The constraints R2, R3, R4, R5, are convex in W for the partial ordering SUB.

Proof:

R2 If the fixed structure Σ' is depicted by an acyclic Ordinary Petri Net, then any subnet of that Petri Net is acyclic.

R3 If there is at most one link from the RS stage of one role i to another role j in a fixed structure Σ', then each fixed structure Σ such that Σ SUB Σ' has fewer interactions than Σ'. In particular, there is at most one link from the RS stage of one role i to another role j in Σ'. Σ satisfies constraint R3.

R4 If the SA stage of each role in a fixed structure Σ' receives either observations from the sensors or one and only one response sent by some other role, then each fixed structure Σ such that Σ SUB Σ' has fewer interactions than Σ', and receives either sensors' observations or at most one response sent by some other role. Σ satisfies constraint R4.

R5 If a fixed structure Σ' does not have one link from the SA stage of role i to the IF stage of role j and one link from the RS stage of role i to the SA stage of role j, then each fixed structure Σ such that Σ SUB Σ' has fewer interactions than Σ', and cannot have one link from the SA stage of role i to the IF stage of role j and one link from the RS stage of role i to the SA stage of role j. Σ fulfills R5

7.4.2 Constraints R1, R_F

Constraint R1 has to be investigated to characterize AW. The problem is that R1 is not convex. It is possible to break the connectivity of a fixed structure by removing a fixed link as well as by adding a fixed link. This happens, for example, if a link that is added to the fixed structure originates from a transition of the current net but does not terminate at a transition that was previously in the net. In that case, a transition without output place is created, which violates R1. Figure 7.3 describes a sequence in which R1 is fulfilled, violated, and fulfilled again by successively adding two fixed links.

Fortunately, the theory of fixed structure systems has already characterized the fixed structures that satisfy R1 (Remy, 1986). These results can be formulated within the context of this thesis once the Universal Net is introduced. This notion is related to the fixed constraints R_F, the set of links that have been ruled in or ruled out for any AFS. Recall that the designer can rule in or out some fixed links by putting 0s and 1s in (S, s, F, G, H, C), the generic representation of an element of W.

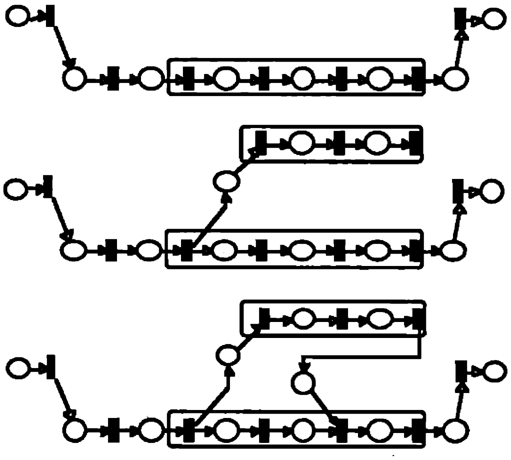


Fig. 7.3 Successive Fixed Structures

Definition 7.5

The *Universal Net* ΣU is the WDFS obtained by replacing the undetermined elements of (S, s, F, G, H, C) by 1.

In other words, ΣU is a fixed structure that contains all the interactions that have not been ruled out. ΣU is associated with an Ordinary Petri Net, which contains all the admissible links in the structure, and which is a marked graph if the external place and the sink are merged in one single place (Propositions 5.9 and subsection 6.2.2). From Theorem 2.4, the simple information flow paths of ΣU are the S-components associated with the minimal support S-invariants of the net that contain the external place. Each simple information flow path is a subnet of ΣU, and is thus an element of W.

Proposition 7.11 (Remy)

A WDFS Σ satisfies constraints R1 and R_F

if and only if

Σ belongs to the lattice polynomial generated by the simple information flow paths of ΣU which is equivalent to

Σ is a meet of simple information flow paths of the Universal Net ΣU.

Recall from Definition 3.9 that the lattice polynomial generated by the simple information flow paths of ΣU is the set of all WDFS that are computed by performing join and meet operations on the simple information flow paths. Proposition 7.11 has several important implications.

First, it shows that one goes from a fixed structure to a fixed structure that is immediately above by adding a simple information flow path instead of a link. The elementary blocks from which connected fixed structures can be built are the simple information flow paths, and not the links.

Second, it applies only to the constraints R1 and R_F. Consequently, this result is valid in W as well as in any subset of W, in the W^i i in [1..m] for example. In each of these subsets, the elements that satisfy constraints R1 and R_F must be the meet of simple information flow paths of the Universal Net ΣU.

Third, the AFS that satisfy R_F can be characterized, as formulated in Proposition 7.12.

Proposition 7.12

Σ is a WDFS that belongs to AW and verifies R_F if and only if

- Σ lies between a maximal solution and a minimal solution:

  $\exists\ \Sigma_1$ and Σ_2 such that Σ_1 SUB Σ SUB Σ_2

  Σ_1 is a minimal element of $W \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$

  Σ_2 is a maximal element of $W \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$
- Σ is a union of simple information flow paths of ΣU, the Universal Net.

Proof:

The first part comes from the fact that R2, R3, R4, R5 are convex constraints.

The second part is Proposition 7.11.

These results must be combined with other constraints to sort out the set of variable structures that are solutions of a given design problem. The relationships between Proposition 7.12 and the colored constraints are first assessed.

7.4.3 Colored Constraints

In section 7.2, the colored constraints are analyzed using the set of Boolean properties R_i^0 and R_i^1 i in [1..k]. These binary relations are defined on W, and verify without much difficulty Proposition 7.13.

Proposition 7.13

For any i in [1..k], R_i^0 and R_i^1 are convex properties in W.

Proof:

If an Ordinary Petri net Σ is such that Σ_1 SUB Σ SUB Σ_2, with $R_i^0(\Sigma_1)$ and $R_i^0(\Sigma_2)$ true, then any link that is absent in both Σ_1 and Σ_2 is absent in Σ.
Thus $R_i^0(\Sigma)$ holds because Σ cannot contain the link L_i.
If an Ordinary Petri net Σ is such that Σ_1 SUB Σ SUB Σ_2, with $R_i^1(\Sigma_1)$ and $R_i^1(\Sigma_2)$ true, then any link that is present in both Σ_1 and Σ_2 is present in Σ.
Thus $R_i^1(\Sigma)$ holds because Σ contains the link L_i.

Remember that each pair of binary relations R_i^0, $R_i^{1\cdot}$ i in [1..k], defines a partition of W into two subsets W^0_i and W^1_i. Proposition 7.13 indicates that both subsets W^0_i and W^1_i are convex, and that any intersection of W^0_i or W^1_i with an other convex subset of W is convex.

Proposition 7.14

The subsets W^i i in [1..m], defined in section 7.3, are convex in W.

Proof:

Each W^i is some $\left(\bigcap_{i_j=1} W^1_{i_j} \right) \cap \left(\bigcap_{i_j=0} W^0_{i_j} \right)$

By Proposition 7.13, each W^0_i and W^1_i is a convex set, and by Proposition 7.8 the intersection of convex sets is a convex set. Thus the claim.

Proposition 7.14 has many implications.

- It shows that any W^i is completely determined by its minimal and maximal elements.
- As R2, R3, R4 and R5 are convex, the intersection of each W^i with the set of WDFS that satisfies R2, R3, R4, R5 is convex. Each intersection is uniquely determined by $\{\Sigma^i_{1,j}\}$, the set of minimal elements and $\{\Sigma^i_{2,k}\}$, the set of maximal elements.
- Finally, it indicates that the intersection of each W^i with the set of all AFS that satisfy R_F verifies Proposition 7.15.

Proposition 7.15

For each i in [1..m], Σ is a WDFS that belongs to AW, W^i and verifies R_F if and only if

- Σ lies between a maximal solution and a minimal solution:
  $\exists\ \Sigma^i_1$ and Σ^i_2 such that Σ^i_1 SUB Σ SUB Σ^i_2
  Σ^i_1 is a minimal element of $W \cap W^i \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$.
  Σ^i_2 is a maximal element of $W \cap W^i \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$.
- Σ is a union of simple information flow paths of ΣU, the Universal Net.

These results on W can be translated into properties on V. The combination of Propositions 7.15 and 7.3 yields a characterization, illustrated in Figure 7.4, of the WDVS that verify R6, R_F, R_C and Assumption 7.1.

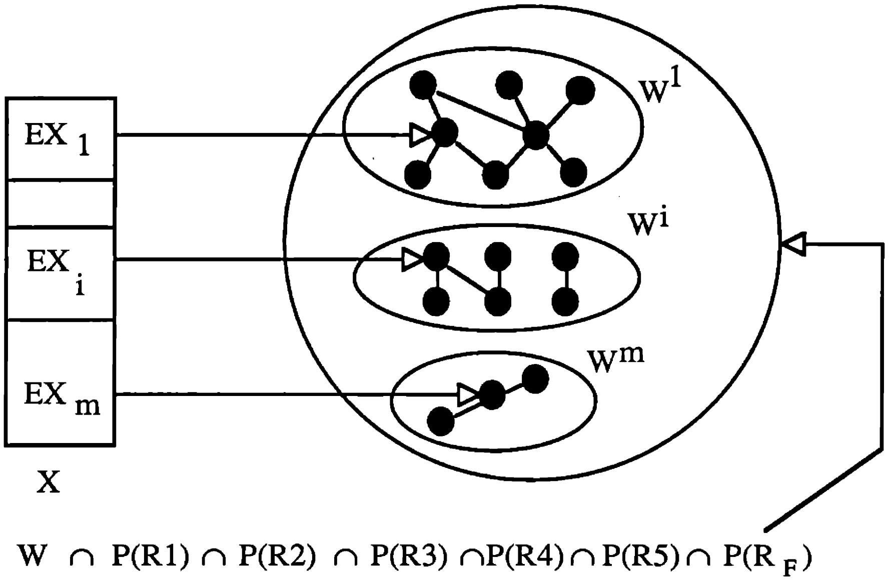

EX_1
EX_i
EX_m
X
W^1
W^i
W^m
$W \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5) \cap P(R_F)$

Fig. 7.4 Assignment

Any WDVS that verifies R6, R_F and R_C and Assumption 7.1 is such that

- The set X is partitioned into $EX_1, ..., EX_m$. (Proposition 7.3)
- The elements of each subset EX_i are assigned to the same unique fixed structure in $W^i \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$, whose ordering can be depicted by a Hasse diagram (Assumption 7.1). In $W^i \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$, this structure lies between one minimal element and one maximal element (Proposition 7.15).

  $EX_1 \text{---->} \Sigma^1{}_1$ with $\Sigma^1{}_{1, j1}$ SUB Σ^1 SUB $\Sigma^1{}_{2, j2}$, ...,

  $EX_m \text{---->} \Sigma^m{}_1$ with $\Sigma^m{}_{1, k1}$ SUB Σ^m SUB $\Sigma^m{}_{2, k2}$.
- In each W^i, one goes from a fixed structure to a fixed structure that is immediately superior by adding a simple information flow path of ΣU (Proposition 7.15).

This thesis now turns to the constraints that have been expressed using the language of the set V exclusively.

7.4.4 Constraints R7, R8, R9

These constraints do pose difficulties as far as convexity is concerned. If Π_1, Π_2 and Π are three WDVS such that Π_1 ACT Π ACT Π_2, if any input link in Π_1 is permanent (Π_1 verifies R7, R8, R9) and any input link in Π_2 is permanent (Π_2 verifies R7, R8, R9), then Π may violate one of the constraints R7, R8, R9. This happens if some input links are present in Π_2 but not in Π_1. One can construct a WDVS Π, that violates one of the constraints R7, R8, R9, by considering Π_1, and by activating an input link that is present in Π_2 but not in Π_1 for some elements of X only.

Example 7.3:

A violation of constraints R7, R8, and R9 is depicted on Figure 7.5. A series of variable structures with two roles and two sensors is presented. In each structure, the links that are permanent have not been annotated. X1 = {A, B} and X2 = {U, V} are the output alphabets of the sensors, L1 is

$$L1 = \begin{array}{c} \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] \end{array} \begin{array}{l} <A, U> \\ <A, V> \\ <B, U> \\ <B, V> \end{array}$$

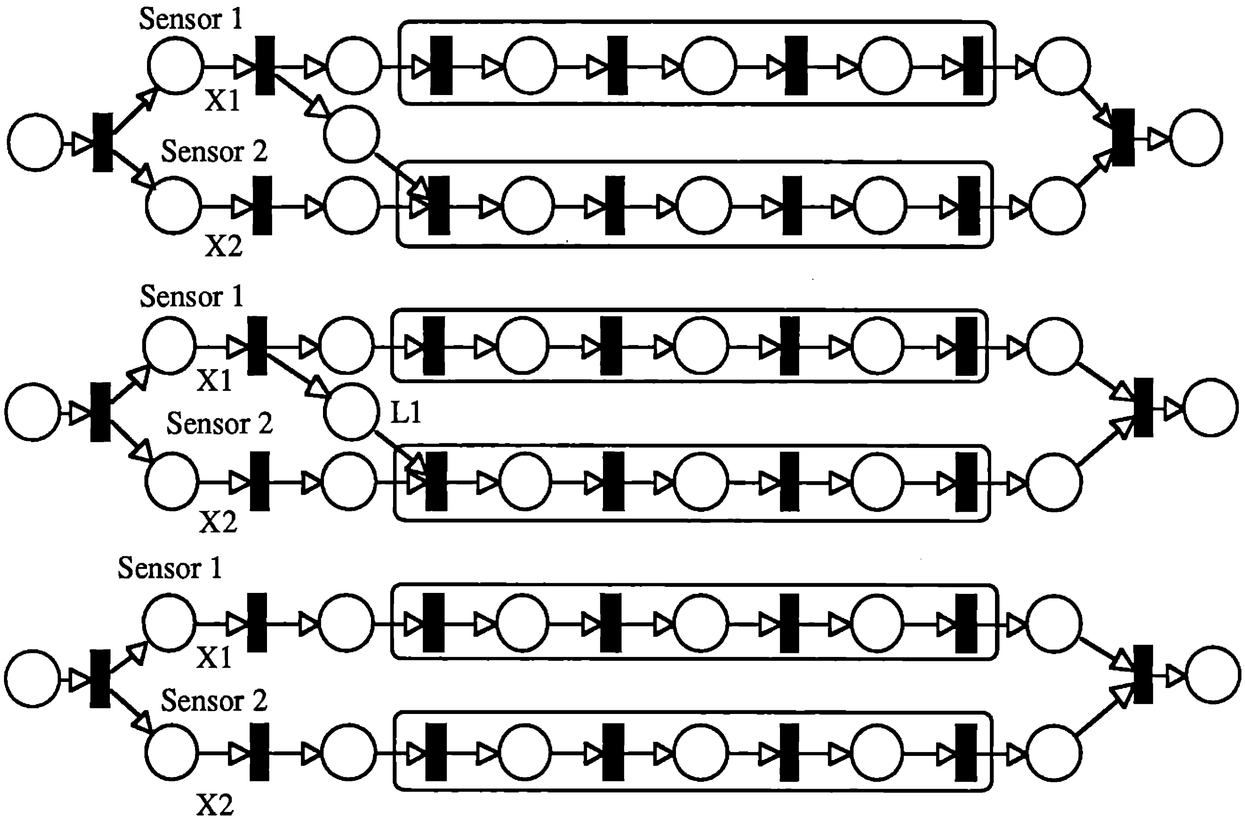


Fig. 7.5 Violation of Constraints R7, R8, R9

> The structures are considered from top to bottom. All the links are permanent in the first structure. This first structure lies above a structure in which one input link is variable. This intermediate structure is itself above one structure in which all links are permanent. The first and the third structures verify constraints R7, R8, R9, whereas the second structure violates constraint R7.

The convexity of the constraints is destroyed between WDVSs that have different input links. However, Proposition 7.16 indicates that the convexity is preserved between variable structures that have the same input links.

Proposition 7.16

Let Π_1, Π_2 and Π be three WDVS such that Π_1 ACT Π ACT Π_2.
If $R7(\Pi_1) = R7(\Pi_2) = R8(\Pi_1) = R8(\Pi_2) = R9(\Pi_1) = R9(\Pi_2)=$ true and
if every input link in Π_2 is activated in Π_1, then $R7(\Pi) = R8(\Pi)= R9(\Pi) =$ true.

Proof:

Π has the same input links as Π_1 and Π_2, which satisfy R7, R8, R9.

Therefore, the input links in Π are permanent and Π satisfies R7, R8, R9.

Note that the input links of a variable structure are permanent if and only if they are present in any fixed structure in the support. Proposition 7.16 shows therefore that the WDVSs that satisfy constraints R7, R8, and R9 are characterized by their input links. Each combination of input links corresponds to one convex set of those WDVS which satisfy R7, R8 and R9.

7.4.5 Partial Characterization

Proposition 7.17

Any WDVS that satisfies all constraints but R10 is such that

- $X = \cup_{i \text{ in } [1..m]} EX_i$ with $(i \neq j) \Rightarrow EX_i \cap EX_j = \emptyset$.
- $\forall$ i in [1..m] , $\forall$ x in EX_i
    - $\exists$ $\Sigma^i{}_1$ and $\Sigma^i{}_2$ in $W^i \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$

      s.t. $\Sigma^i{}_1$ SUB $(\Pi(x) = \Sigma^i)$ SUB $\Sigma^i{}_2$

      and s.t. $\Sigma^i{}_1$ and $\Sigma^i{}_2$ have the same input links.
    - Σ^i is a union of simple information flow paths of ΣU, the Universal Net.

Proof:

The first statement comes from Proposition 7.15. The first part of the second statement comes from the combination of Proposition 7.16 with Proposition 7.15, while the second part comes from Proposition 7.15.

Proposition 7.17 can be translated instantly using the characterization of the WDVS in matrix form, as expressed by Proposition 7.18.

Proposition 7.18

Π is a WDVS that satisfies all constraints but R10 if and only if

- The set of inputs of X that activate a link belongs to the lattice polynomial $L(EX_1,.., EX_m)$.
- Π is bounded by at least one minimal element and one maximal element

$$\Pi_1 \text{ ACT } \Pi \text{ ACT } \Pi_2.$$

- Π_1 and Π_2 have the same input links.
- The support of the CPN representation of Π is a union of simple paths of ΣU.

Proof:

The first statement has been proved already.

One minimal element can be constructed as follows, by assigning to every element of EX_i a minimal fixed structure in W^i, $\Sigma^i{}_1$.

$EX_1 \text{ ----> } \Sigma^1{}_1, \ldots, EX_m \text{----> } \Sigma^m{}_1.$

One maximal element can be constructed as follows, by assigning to every element of EX_i a maximal fixed structure in W^i, $\Sigma^i{}_2$ with the same input links as $\Sigma^i{}_1$.

$EX_1 \text{ ----> } \Sigma^1{}_2, \ldots, EX_m \text{----> } \Sigma^m{}_2.$

For any assignment that satisfies all constraints but R10

$EX_1 \text{ ----> } \Sigma^1$ with $\Sigma^1{}_1$ SUB Σ^1 SUB $\Sigma^1{}_2, \ldots, EX_m \text{----> } \Sigma^m$ with $\Sigma^m{}_1$ SUB Σ^m SUB $\Sigma^m{}_2$. This implies, Proposition 7.4 Π_1 ACT Π ACT Π_2.

Finally, the Ordinary Petri Net $\Pi(x)$ is, for any x, a union of simple information flow paths of ΣU, thus the Support of the representation is a union of simple information flow paths.The converse is obvious.

Proposition 7.18 gives significant insights about the set of solutions.

- A solution can only be found between a minimal variable structure Π_1 and a maximal variable structure Π_2. These variable structures have the same input links. It appears therefore that the input links constitute parameters that characterize the set of solutions to the design problem.
- Between one minimal and one maximal structure, the unit leading from one variable structure Π to one variable structure Π' that covers this structure is composed of a path and one elementary set of inputs EX_i attached to it. One goes from Π to Π' by performing the join of Π with the WDVS whose support is the path, and in which each link is activated by the elementary set of inputs EX_i. This can be translated in Petri Net terms by adding the path to the graphical support of the Colored Petri Net that represents Π, and by checking that all inputs that belong to EX_i activate the matrices attached to the arcs that belong to the path, i.e., that the set of inputs EX_i influences all the processes that belong to the simple information flow path. Unfortunately, all such WDVSs may not be solutions to the design problem because of Constraint R10.

Example 7.4:

Let us illustrate the results that have been obtained so far. Suppose that there are two roles and one sensor. The alphabet of the Sensor is {A, B, C}, and the decomposition of R_C leads to $EX_1 = \{<A>\}$, $EX_2 = \{<B>, <C>\}$.

All solutions correspond to the same permanent input links, one link from the sensor to the SA stage of Role 1, and one link from the sensor to the SA stage of Role 2.

$W^1 \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$ has a unique maximal element, represented on Fig. 7.6, and a unique minimal element represented on Fig. 7.7.

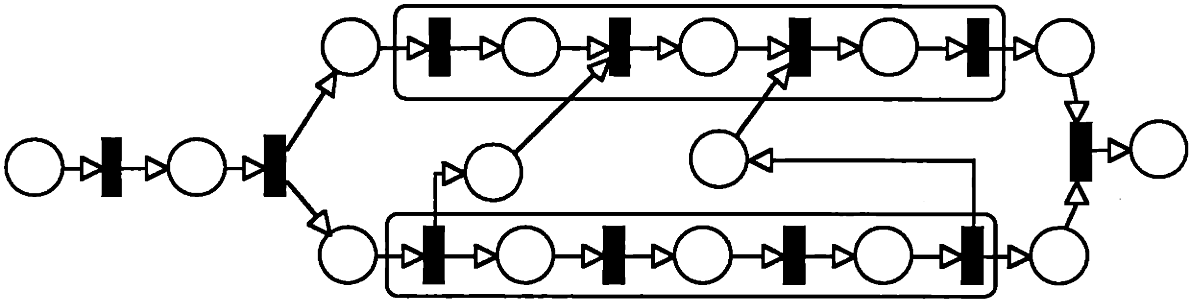


Fig. 7.6 Maximal Element of W^1

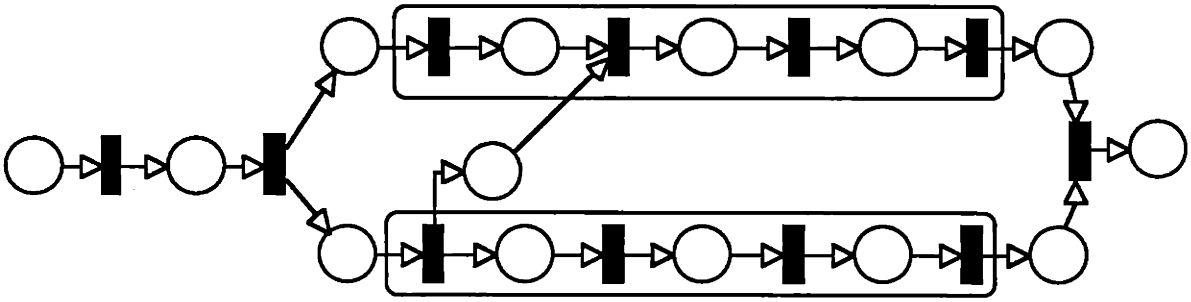


Fig. 7.7 Minimal Element of W^1

$W^2 \cap P(R1) \cap P(R2) \cap P(R3) \cap P(R4) \cap P(R5)$ has a unique maximal element, represented on Fig. 7.8, and a unique minimal element represented on Fig. 7.9.

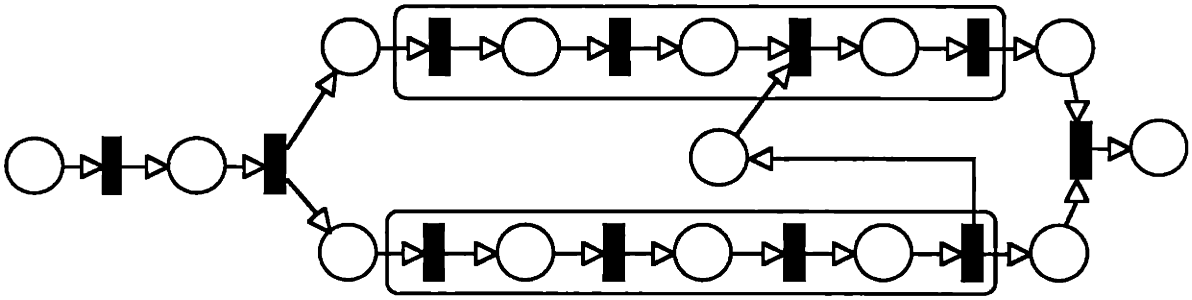


Fig. 7.8 Maximal Element of W^2

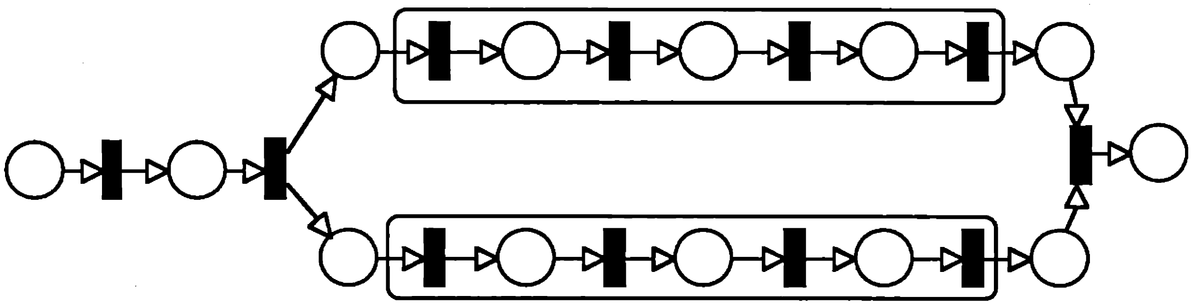


Fig. 7.9 Minimal Element of W^2

There is one and only one minimal variable structure, which is represented on Fig. 7.10. To make the notations simple, a permanent link has not been annotated. The annotation {A} indicates that the link is activated by <A>.

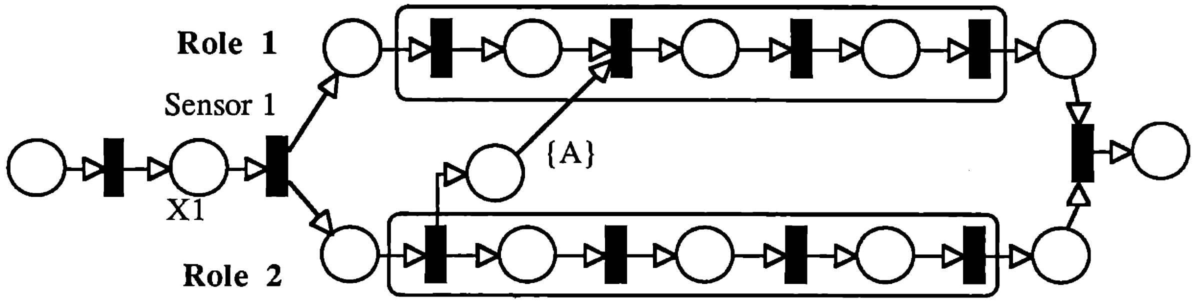


Fig. 7.10 Minimal Variable Structure

There is one and only one maximal variable structure, which is represented on Fig. 7.11. The unique building simple information flow path that allows one to go from the minimal variable structure to the maximal variable structure is the path of Fig. 7.12. One can attach any element of L ({<A>},{B,C}) = {Ø, {<A>}, {B,C}, {A, B, C}} to the path. Fig. 7.13 depicts an intermediate result, in which {B,C} is attached to the path.

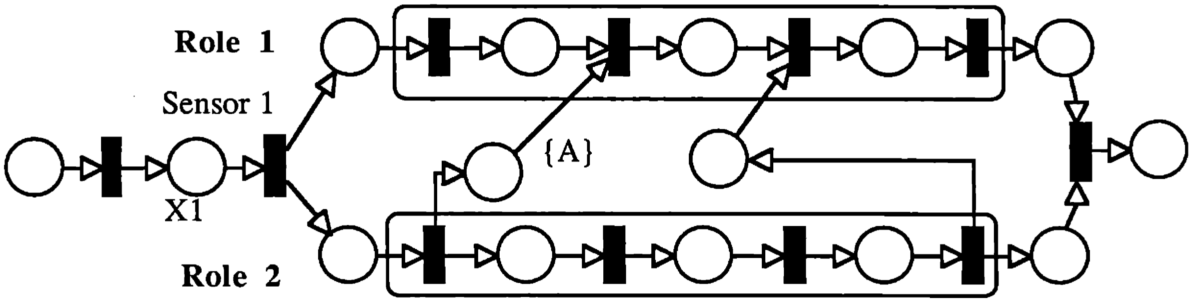


Fig. 7.11 Maximal Variable Structure

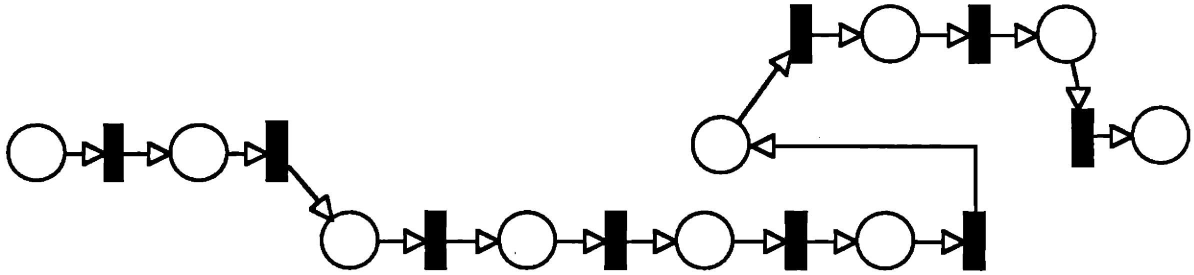


Fig. 7.12 Building Path

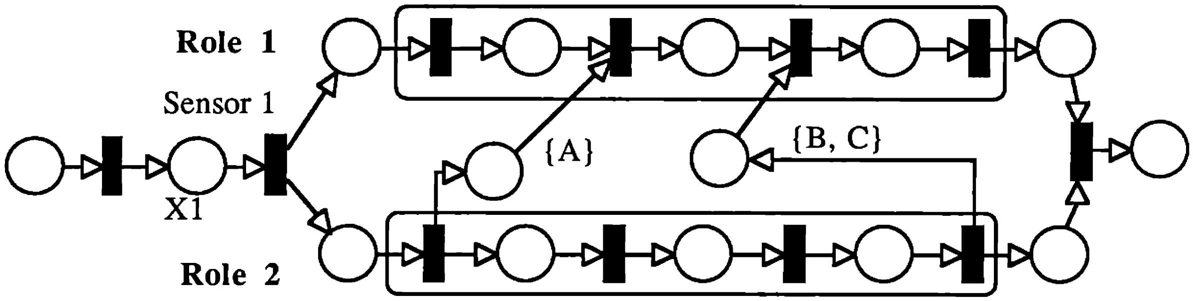


Fig. 7.13 Intermediate Variable Structure

7.5 CONSTRAINT R10

7.5.1 Problem Definition

Consider some minimal variable structure Π_1, some maximal variable structure Π_2, and some variable structure Π that lies between them and verifies Proposition 7.18. The set AC, of inputs that activate the link in Π, contains the set AC1, of inputs that activate the link in Π_1, and is contained in AC2, the set of inputs that activate the link in Π_2. Proposition 7.18 states that, for any element A of $L(EX_1,.., EX_m)$ between AC1 and AC2, there exists some variable structure that satisfies all constraints but R10 such that AC = A. However, constraint R10 indicates that a partition between AC and DC is valid if and only if the effective alphabets of the partition are accessible, i.e., if the stages have enough information to be meaningfully coordinated. There are two problems. The first comes from the fact that the set of effective alphabets vary irregularly when successive elementary sets of inputs EX_i are added to the set of inputs that activate a link. The second is the fact that an alphabet can be effective without being accessible, and vice versa. These two mechanisms are very powerful for eliminating WDVSs that satisfy all constraints except R10, but whose coordination does not make sense.

Example 7.5:

Suppose that there are two alphabets, $X_1 = \{A, B, C\}$ and $X_2 = \{U, V\}$, and that the decomposition of R_C yields $EX_1 = \{ <A, U>\}$, $EX_2 = \{<A, V>\}$, and $EX_3 = <B> \cup <C>$. If a link can vary between a given minimal variable structure and a given maximal structure from AC = Ø to AC = X, the two following sequences are valid sequences for AC.

- Ø, EX_1 , $EX_1 \cup EX_2 = <A>$, $<A> \cup (<B> \cup <C>) = X$
- Ø, $EX_3 = <B> \cup <C>$, $EX_3 \cup EX_2 = <B> \cup <C> \cup <A, V>$, X.

Over those two sequences, the set of effective alphabets of the partition has an irregular pattern.

Over the first sequence, this set is Ø, $\{X_1, X_2\}$, $\{X_1\}$ and finally Ø.

Over the second sequence, this set is Ø, $\{X_1\}$, $\{X_1, X_2\}$ and finally Ø.

Now, if it happens that X_2 is the only accessible alphabet, then none of those sequences are allowed.

This example shows that constraint R10 can be quite stringent, and can restrict dramatically the number of solutions to the design problem, as compared to the size of the set of WDVS that satisfy all constraints but R10. A practical example of this reduction is given in chapter VII. One feels immediately that this reduction in size can have both positive and negative effects. On one hand, this reduction facilitates the computational problem and any implementation of the methodology. Another positive effect is that it may provide the designer with a small set of solutions, which may be analyzed more thoroughly than if the set of solutions were large. On the other hand, this reduction in size may interfere with the probability that a solution exists, because the more drastic the reduction, the less likely the existence of a solution. Under general assumptions, it cannot be proved that there necessarily exists a solution to the design problem. However, some valuable insights can be provided into constraint R10, which are detailed in the next subsection. Finally, note that it is unlikely that this reduction in size is just an artifact of the model. R10 is a constraint that enforces one coordination mechanism. R10 describes one approach in which a variable interaction is valid if this variable interaction is directly related to the information already possessed by the objects of the system. There are many practical and theoretical reasons to believe that coordinating the activity of distributed entities is very difficult. The variable structures that satisfy all constraints but R10 are created by taking all combinations of fixed dataflow structures in m different sets. It would be very surprising if an arbitrary combination were to achieve a valid coordination.

7.5.2 Partial Ordering EFF

The notion of accessible patterns has been introduced in Chapter VI to deal with constraint R10. A partial ordering must be defined on the set of accessible patterns to ease the analysis of constraint R10.

Definition 7.6

Let E, E' be two accessible patterns. The binary relation EFF is defined by

- E EFF E' if and only if
- Every entry of E is included in the corresponding entry of E'.

In other words, E EFF E' if and only if the alphabets that are accessible in E are accessible in E' for each link. If a variable structure Π has an accessible pattern E and Π' has an accessible pattern E', each link in Π' has access to at least as many sources of information as the corresponding link in Π. The partition between AC and DC at each link can be based on more information in Π' than in Π. Note however, that this does not imply anything about the effective alphabets of the partition. A partition is valid if and only if the set of its effective alphabets is included in the set of accessible alphabets, but there are no reasons that force the set of effective alphabets to be exactly the set of accessible alphabets. The binary relation EFF is of theoretical interest because of Proposition 7.19.

Proposition 7.19

The binary relation EFF is a partial ordering of the set of all accessible patterns.

Proof:

The relation is trivially reflexive. It is antisymmetric, because "is included in" is antisymmetric componentwise, it is transitive because "is included in" is transitive component-wise.

The goal of this subsection is to relate properties of WDVS and the ordering of their effective patterns. The first result is Proposition 7.20, which compares the intersection of two accessible patterns E and E', to E and to E'.

Proposition 7.20

Let E and E' be two accessible patterns, then

(E INT E') EFF E and
(E INT E') EFF E'

The proof is obvious from the definition of INT, Definition 6.3.

This simple proposition has many implications.

- First, it shows that the accessible pattern of a variable structure shrinks when its support contains more and more fixed structures. Indeed, suppose that the support of a variable structure contains k fixed structures and corresponds to an accessible pattern E. If a (k+1)-th structure Σ is added to the support, then the accessible pattern becomes E INT E(Σ) , with (E INT E(Σ)) EFF E. The level of information on which the variability can be based gets smaller, and so does the number of structures that verify R10.
- Second, this property validates to a certain extent Assumption 7.1. By choosing to

generate a variable structure with a minimal number of fixed structures, the designer makes sure that he gets the largest accessible pattern possible. The designer maximizes the possibility that a solution exists.

- Finally, because this proposition gives, within this model, a mathematical formulation of the intuition that if a structure is more and more variable (if the support has more and more fixed structures), it becomes more and more difficult to coordinate the activities in a structure (the accessible pattern gets smaller). On the other hand, Proposition 7.21 describes a mechanism that increases effective patterns.

Proposition 7.21

(Σ SUB Σ') implies $E(\Sigma)$ EFF $E(\Sigma')$.

Proof:

Let T be some stage (some transition) in Σ. If an alphabet Xi is accessible in Σ at stage T, there exists a path from Sensor i to T in the Petri Net representation of Σ. This path is also present in Σ', because Σ SUB Σ'. Therefore, the set of alphabets that are accessible at T in Σ' contains the set of alphabets that are accessible at T in Σ. Thus, the set of alphabets that are accessible at both ends of any link in Σ' contains the set of alphabets that are accessible at both ends of the link in Σ.

Proposition 7.21 indicates that the effective pattern associated with a fixed structure gets larger if some connectivity is added to the fixed structure. This proposition counterbalances to a certain extent Proposition 7.20. It shows that the effective pattern of one variable structure can be increased by adding connectivity to the fixed structures that belong to its support. Indeed if each fixed structure is more connected, the effective patterns are larger, and their intersection, the effective pattern of the variable structure is larger. As far as solutions to the design problem are concerned, Propositions 7.17, 7.20 and 7.21 can be combined into Proposition 7.22.

Proposition 7.22

Let Π be a WDVS that satisfies proposition 7.17.

Π ACT Π' is equivalent to

$\forall$ i in [1..m] Σ^i SUB Σ'^i which implies

$\forall$ i in [1..m] $E(\Sigma^i)$ EFF $E(\Sigma'^i)$ which implies

$E(\Pi)$ EFF $E(\Pi')$.

This proposition reflects a trade-off between activation and variability in a variable structure. If each interaction in a variable structure is activated by more and more inputs, each object in the system receives more and more information. The possibility of having a coordinated variable pattern of interaction is thus increased (E(Π) increases). On the other hand, adding more and more inputs in AC reduces the number of partitions such that AC' contains AC. Therefore by moving up from a minimal variable structure to a maximal variable structure, one obtains larger set of observations on which to base a partition of X into AC and DC, while having less and less variable structures left over up to the maximal element.

Note, however, that accessible patterns do not describe all links in the system. By definition, an accessible pattern describes only those links that can be variable, whereas Proposition 7.21 does not establish any distinction between links that can be variable and links that can be permanent. One can also increase the accessible pattern of a variable structure by adding permanent links in the structure, between the sources of information and the input stages. There are thus two mechanisms to increase the amount of information accessible at the stages, increasing the access to the sources of information, or increasing the exchange of information. The problem with the first mechanism is that it may create conflicts with the performance of the roles. The problem with the second mechanism is that it restricts the number of variable structures that are solutions to the design problem. Note, finally, that increasing the sources of information means moving from one set of solutions, characterized by some permanent input links, to some other set of solutions, characterized by more permanent input links. On the other hand, increasing the exchange of information means moving up from the minimal variable structures to the maximal variable structures within a set of solutions characterized by the same permanent input links.

Proposition 7.22 indicates that the effective pattern of a variable structure is increasing from a minimal element to a maximal element. One can thus view the structure of the set of solutions as a layering of partially ordered sets. Each layer corresponds to variable structures that have the same accessible pattern. Proposition 7.23 is important for making the structure of these layers precise.

Proposition 7.23

Let E be some accessible pattern.

The set of all variable structures such that E(Π) EFF E is a sublattice of V.

Proof:

Let L be a link. Even by relabeling the alphabets, suppose that the entry corresponding to L in E is {X1, .., Xk}. X1 = $\{x1_1,..., x1_{|X1|}\}$,.., Xk = $\{xk_1,..., xk_{|Xk|}\}$.
If Π_1 and Π_2 are two WDVS such that
E(Π_1) EFF E and E(Π_2) EFF E, then, by definition of the effective alphabets

$$AC_1 = \bigcup_{i1 \text{ in } I1, \ ..., \ ik \text{ in } Ik} \{<x1_{i1}, ..., xk_{ik}>\}$$

with I1 included in [1..|X1|],..., Ik included in [1..|Xk|], and

$$AC_2 = \bigcup_{i1 \text{ in } I'1, \ ..., \ ik \text{ in } I'k} \{<x1_{i1}, ..., xk_{ik}>\}$$

with I'1 included in [1..|X1|],..., I'k included in [1..|Xk|].

The definition of the meet of two structures implies that a link is activated by any input that activates L in Π_1 and Π_2, thus

$$AC_{\text{meet } \Pi 1 \text{ and } \Pi 2} = \bigcup_{i1 \text{ in } I1 \cap I'1, \ ..., \ ik \text{ in } Ik \cap I'k} \{<x1_{i1}, ..., xk_{ik} > \}$$

The effective alphabets of the meet belong to {X1, .., Xk}, because AC can be written as a union of the equivalence classes of the relation $R_{1,..., k}$, as introduced in Definition 4.13. The definition of the join of two structures implies that a link is activated by every input that activates L in Π_1 or that activates L in Π_2, thus

$$AC_{\text{join } \Pi 1 \text{ and } \Pi 2} = \bigcup_{i1 \text{ in } I1 \cup I'1, \ ..., \ ik \text{ in } Ik \cup I'k} \{<x1_{i1}, ..., xk_{ik} > \}$$

Here again, the effective alphabets of the join belong to {X1, .., Xk}, because AC can be written as a union of the equivalence classes of the relation $R_{1,..., k}$.

Proposition 7.19 indicates therefore that each layer between a minimal element and a maximal element of Proposition 7.18 corresponds to the intersection of a sublattice with a convex set. This intersection is thus characterized by its minimal and maximal elements, and the building block that allows advancing from one structure to a structure that covers it.

7.6 CHARACTERIZATION OF THE SOLUTIONS

The successive propositions that characterized the set of solutions to the design problem can be summarized in a single proposition.

Proposition 7.24

Π is a WDVS that satisfies all constraints if and only if

- The graphical support of the CPN is a union of simple information flow paths of ΣU.
- Π is bounded by at least one minimal solution and one maximal solution

$$\Pi_1 \text{ ACT } \Pi \text{ ACT } \Pi_2.$$

- Π_1 and Π_2 have the same permanent input links.
- $E(\Pi) = E$ if and only if
  - Π is bounded by one minimal solution Π'_1 and one maximal solution Π'_2 such that $\Pi'_1 \text{ ACT } \Pi \text{ ACT } \Pi'_2$
    $E(\Pi'_1) = E(\Pi'_2) = E$
- Let L be a link, and {Xi1, .., Xik} be the entry corresponding to that link in $E(\Pi)$.
  The set, AC, of inputs that activates the link
  - belongs to the lattice polynomial $L(EX_1,.., EX_m)$
  - can be expressed as a union of equivalence classes of $R_{i1,...,ik}$.

Proof:

The first three statements have been proved. The fourth one combines Propositions 7.22 and 7.23. The fifth combines the fact that AC belongs to $L(EX_1,.., EX_m)$ and the need to construct it from the information contained in the accessible alphabets, and thus to construct it by considering the union of the equivalence classes of $R_{i1,...,ik}$.

Definition 7.7

A Minimal Solution to the design problem is called a VMINO.

A Maximal Solution to the design problem is called a VMAXO.

Proposition 7.24 shows that the set of solutions is characterized by several parameters.

- First, the combinations of permanent input links. A set of solutions is attached to each combination.

- Second, the set of solutions attached to one combination can be found between some VMINOs and some VMAXOs. There is a layering of partially ordered sets from a VMINO to a VMAXO. Each layer corresponds to WDVS whose pattern of interaction is E. The partially ordered sets are layered in an increasingly effective pattern of interactions. Each layer is a sublattice of the lattice of the WDVS s.t. E(∏) EFF E. One possible configuration of the set of solutions is depicted on Fig. 7.14.

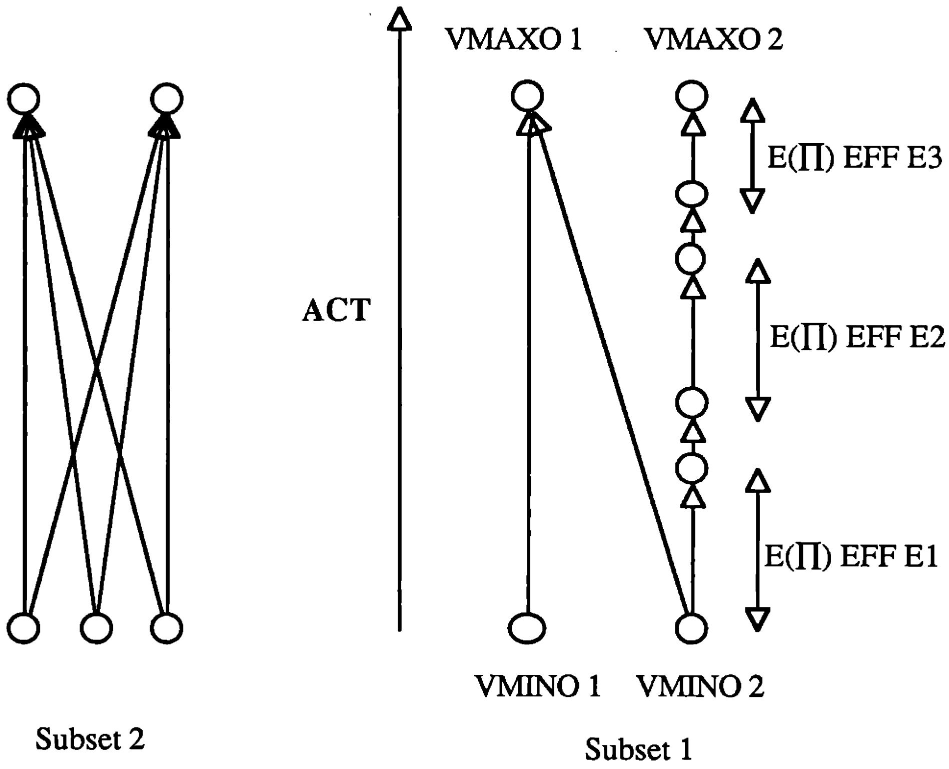


Fig. 7.14 Structure of the Set of Solutions

There are two subsets of solutions. Each subset corresponds to one combination of permanent input links, and is characterized by some VMINOS and VMAXOS. From VMINO 2 to VMAXO 2, three layers have been depicted. They correspond to the effective patterns E1, E2, E3, with E1 EFF E2 EFF E3. The next chapter illustrates these results within the problem of designing two practical systems.

CHAPTER VIII

TWO APPLICATIONS OF THE METHODOLOGY

In this chapter, the methodology is applied to two design problems. These problems have been chosen to illustrate both the modeling issues and the results. The first example describes the problem of coordinating tasks in a submarine. The second one is concerned with one part of the Air Traffic Control system, the Airport Surface Traffic Control system.

8.1 COORDINATION OF TASKS IN A SUBMARINE

8.1.1 The System

This first example describes a simplified version of systems that develop tactical responses in military submarines. The tactical concepts presented here are from the US Navy, and have been illustrated quite convincingly in best sellers such as The Hunt for Red October (Clancy, 1984). The submarine is supposed to be in a sensitive tactical situation and faces two major external threats, torpedoes and depth charges. Torpedoes can be launched by enemy submarines, while depth charges can be dropped from enemy ships or airplanes. Submarines can be detected by passive or active sonars, and surface ships and planes are also detected through acoustical devices. This section focuses on designing a variable structure Command and Control system that adapts its structure of interactions to the tactical parameters.

The submarine has access to two sources of information, the observations from the surface environment and the observations from the deep sea environment. These sources of information can be modeled by two sensors, Sensor 1 and Sensor 2.

- Sensor 1 describes the deep sea environment.
  Its output alphabet is X1 = {NoSubmarine, Submarine, Torpedo}, where NoSubmarine indicates that no submarine is detected by acoustical devices, Submarine indicates that the noise of a submarine vessel has been heard, and Torpedo indicates the detection of a torpedo. To make notations more compact, X1 is abbreviated by X1 = {NS, S, T}.

- Sensor 2 describes the surface environment.
  Its output alphabet is X2 = {NoSurface, Surface, Depth Charge}, where NoSurface indicates that no particular noise coming from the sea surface hints at the presence of a surface ship or a plane, Surface indicates that some characteristic noise has been heard, and Depth Charge models the detection of an offensive weapon. Here again, X2 is abbreviated, and X2 = {NSU, SU, dC}.
- The inputs to the system are given by the cross product of the output alphabets.
  $X = X1 \times X2 = \{ <NS, dC>, <NS, NSU>, <NS, SU >, <S, dC>, <S, NSU >, <S, SU >, <T, dC>, <T, NSU >, <T, SU >\}$. X contains 9 inputs and has been ordered lexicographically.

The Command and Control system is composed of three roles: The Anti Submarine Warfare Commander (ASW), the Anti Surface Warfare Commander (ASUW), and the Officer of the Deck (OOD). The degree of redundancy of each role is 1. The role of the Officer of the Deck (OOD) is at the top of the hierarchical chain. This role has the responsibility for integrating all aspects of the ship's mission, and must be ready to assume the command of the ship at all times. The OOD, however, does not have immediate access to the sensor's observations, which are the responsibilities of ASW and ASUW. The area of competence of ASW is anti submarine warfare. ASW monitors the deep sea environment through different acoustical devices. ASW has been trained to recognize the characteristic noise of foes and friendly ships, as well as their tactical submarine procedures. Similarly, ASUW is the expert in anti-surface warfare. ASUW can recognize the noise of ships, the type of missions to which ships can be assigned, and all relevant aspects of surface warfare. The Command and Control system develops a tactical response that is transmitted to the effectors, the rest of the crew, which includes the ship control party and the combat elements.

Because of the hierarchical relationships, it is assumed that ASW cannot issue a command to ASUW, and conversely that ASUW cannot issue a command to ASW. It is further assumed that the design should characterize structures that optimize the existing competence in the system. For that purpose, the ASW should be the role that formulates the tactical response if the submarine is only threatened in the deep sea environment. Similarly, if a threat is detected in the surface environment only, the ASUW should be the role that issues commands to the effectors. Finally, if no threat has been detected, or if threats are detected in both environments, the Officer of the Deck should formulate the tactics to be followed. Finally, it is assumed that five human resources can be assigned to be the Officer of the Deck. These human resources cannot be assigned to ASW or ASUW. Four human resources can be assigned to ASW, while four other human resources have the training required to perform the role of the Anti Surface Warfare Commander. Each resource can perform one and only one role.

8.1.2 Constraints

The informal description of the system done in subsection 8.1.1 can be translated into mathematical constraints using the methodology of this thesis. Three roles have already been identified, as well as two sources of information. The cross product of the sensor's alphabets is X, which contains nine elements. The knowledge that has been acquired about the system allows us to formulate the constraints described in the sequel. Throughout the specifications, a "#" denotes that a link has not been specified, i.e., this link is a degree of freedom in the design. A "I" denotes the 9×9 identity matrix, i.e., the link is activated by all inputs, and a "0" denotes the 9×9 null matrix, i.e., the link has been ruled out and cannot be activated by any input.

Constraints on S:

The matrix S is a 2×3 block matrix. It is given by:

$$S = \begin{array}{cccl} \text{ASW} & \text{OOD} & \text{ASUW} & \\ I & 0 & 0 & \text{Sensor 1} \\ 0 & 0 & I & \text{Sensor 2} \end{array}$$

The matrix S is completely specified. S indicates that the outputs of Sensor 1, the state of the deep sea environment, can only be monitored by ASW, and that the outputs of Sensor 2, the state of the surface environment, can only be accessed by ASUW.

Constraints on s:

The objective of the design is to obtain variable structures that optimize the competence of the different roles. Thus

- ASW must select a response to be sent to the effectors if and only if the inputs to the system are <S, NSU>, <T, NSU>. These inputs describe the situations in which there is only a submarine threat.
- ASUW must select a response to be sent to the effectors if and only if the inputs to the system are <NS, dC>, <NS, SU>. These inputs describe the situations in

which there is only a surface threat.

- OOD must select the response in all other cases, which are <NS, NSU>, <S, dC>, <S, SU>, <T, dC>, <T, SU>. These inputs describe the situations in which there are either no threats, or threats in both the submarine and the surface environments. This is translated by:

$$\begin{array}{cccc} & \text{ASW} & \text{OOD} & \text{ASUW} \\ s = [& L1 & L3 & L2] \end{array}$$

where,

$$L1 = \left[\begin{array}{ccccccccc} 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \begin{array}{l} \langle NS, dC\rangle \\ \langle NS, NSU\rangle \\ \langle NS, SU\rangle \\ \langle S, dC\rangle \\ \langle S, NSU\rangle \\ \langle S, SU\rangle \\ \langle T, dC\rangle \\ \langle T, NSU\rangle \\ \langle T, SU\rangle \end{array}$$

L1 indicates that ASW selects the response if and only if the inputs to the system are <S, NSU> or <T, NSU>. Similarly,

$$L2 = \left[\begin{array}{ccccccccc} 1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \begin{array}{l} \langle NS, dC\rangle \\ \langle NS, NSU\rangle \\ \langle NS, SU\rangle \\ \langle S, dC\rangle \\ \langle S, NSU\rangle \\ \langle S, SU\rangle \\ \langle T, dC\rangle \\ \langle T, NSU\rangle \\ \langle T, SU\rangle \end{array}$$

L2 indicates that ASUW selects the response if and only if the inputs to the system are <NS, dC> or <NS, SU>. Finally,

$$L3=\begin{array}{l}\left[\begin{array}{ccccccccc}0&0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&1&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&1\end{array}\right]\end{array}\begin{array}{l}<NS, dC>\\<NS, NSU>\\<NS, SU>\\<S, dC>\\<S, NSU>\\<S, SU>\\<T, dC>\\<T, NSU>\\<T, SU>\end{array}$$

L3 indicates that OOD selects the response if and only if the inputs to the system are <NS, NSU>, <S, dC>, <S, SU>, <T, dC>, <T, SU>.

Constraints on F:

Both ASW and ASUW must inform OOD of their situation assessment. OOD does not receive information at its SA stage, therefore it cannot perform situation assessment. No other constraints have been stated on the sharing of information. Thus F is

$$F=\begin{array}{cccc}\text{ASW}&\text{OOD}&\text{ASUW}&\\0&I&\#&\text{ASW}\\0&0&0&\text{OOD}\\\#&I&0&\text{ASUW}\end{array}$$

Constraints on G:

No role can receive as an input the response of the other role, therefore

$$G=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}$$

Constraints on H:

No constraint has been expressed on the sharing of responses, therefore

$$H=\begin{bmatrix}0&\#&\#\\\#&0&\#\\\#&\#&0\end{bmatrix}$$

Constraints on C:

The only constraint on the hierarchical structure is that both ASW and ASUW cannot issue commands or advisories regarding the tactical response to each other. However, they can issue advisories to OOD, or can receive commands or advisories from it.

$$C=\begin{array}{cl} \begin{matrix}\text{ASW} & \text{OD} & \text{ASUW}\end{matrix} & \\ \begin{bmatrix}0 & \# & 0\\ \# & 0 & \#\\ 0 & \# & 0\end{bmatrix} & \begin{matrix}\text{ASW}\\ \text{OD}\\ \text{ASUW}\end{matrix}\end{array}$$

Constraint on RD

Three classes of resources can be identified from the informal description of the system. Each class corresponds to a pool of human decisionmakers, all of whom have received the same training, and can be assigned to one and only one role. To make the notations simple, each class is represented by a unique resource place. The initial marking of the resource place indicates how many persons belong to that class. Thus

$$RD=\begin{array}{cl} \begin{matrix}\text{ASW} & \text{OOD} & \text{ASUW}\end{matrix} & \\ \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} & \begin{matrix}\text{CLASS 1}\\ \text{CLASS 2}\\ \text{CLASS 3}\end{matrix}\end{array} \text{ with } M(\text{Class } 1) = M(\text{Class } 3) = 4,\ M(\text{Class } 2) = 5$$

These constraints are shown on Figures 8.1 and 8.2.

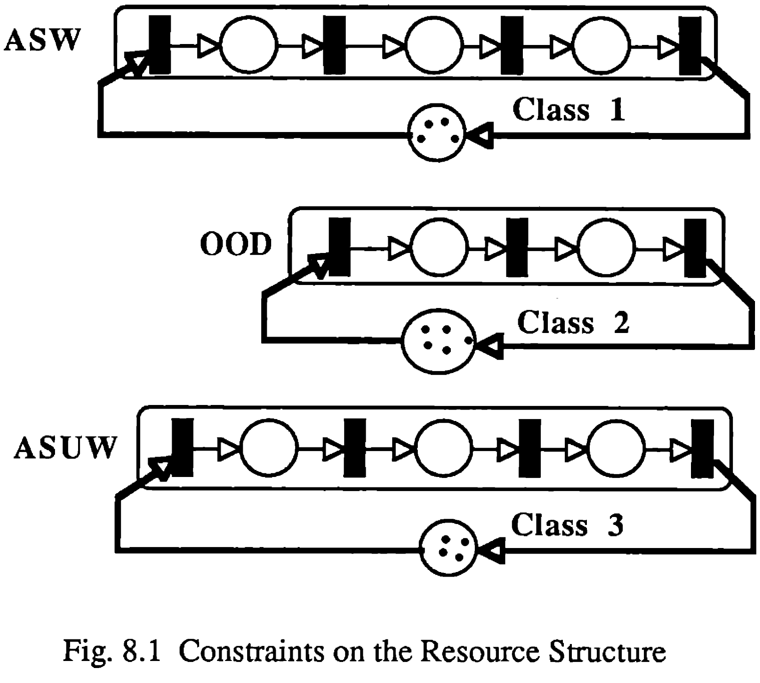


Fig. 8.1 Constraints on the Resource Structure

Figure 8.1 describes the constraints on the resource structure, while Figure 8.2 describes the constraints on the functional structure. The expression of the constraints are as follows: a link that has been ruled out has not been represented, a link that is permanent is drawn with a bold line and without annotations, and a link that is variable is drawn in bold and annotated by the matrix that describes its activation over the set of inputs. Finally, the unspecified links have been indicated and are drawn with plain lines.

8.1.3 Decomposition of the Constraints

Constraint R_F imposes that the link from Sensor 1 to ASW be fixed, that links from Sensor 1 to OOD and ASUW be ruled out, that the link from Sensor 2 to ASUW be fixed, that links from Sensor 2 to OOD and ASW be ruled out, and lastly that all links to the SA stage of OOD must be ruled out. The Universal Net ΣU, the Ordinary Petri Net that contains all links that have not been ruled out by the the design is the Ordinary Petri Net depicted in Figure 8.3.

Constraint R_C applies to three links, the links from the RS stages of the roles to the effectors. These constraints determine a natural partition of X into three elementary sets of inputs, EX_1, EX_2, and EX_3.

- EX_1 = {<S, NSU>, <T, NSU>}, and W^1 corresponds to the subset of W that contains the WDFS in which the fixed link from the RS stage of the ASW to the effectors is present, and in which the fixed links from the RS stages of the OOD and the ASUW to the effectors are not present. These inputs corresponds to a threat in the submarine environment only.
- EX_2 = {<NS, dC>, <NS, SU>}, and W^2 corresponds to the subset of W that contains the WDFS in which the fixed link from the RS stage of the ASUW to the effectors is present, and in which the fixed links from the RS stages of the OOD and the ASW to the effectors are not present. This subset corresponds to a threat in the surface environment only.
- EX_3 = {<NS, NSU>,<S, dC>,<S, SU>,<T, dC>, <T, SU>}, and W^3 corresponds to those WDFS in which the fixed link from the RS stage of the OOD to the effectors is present, and in which the fixed links from the RS stages of the ASW and the ASUW are not present. These are the cases in which the OOD formulates the tactical response.

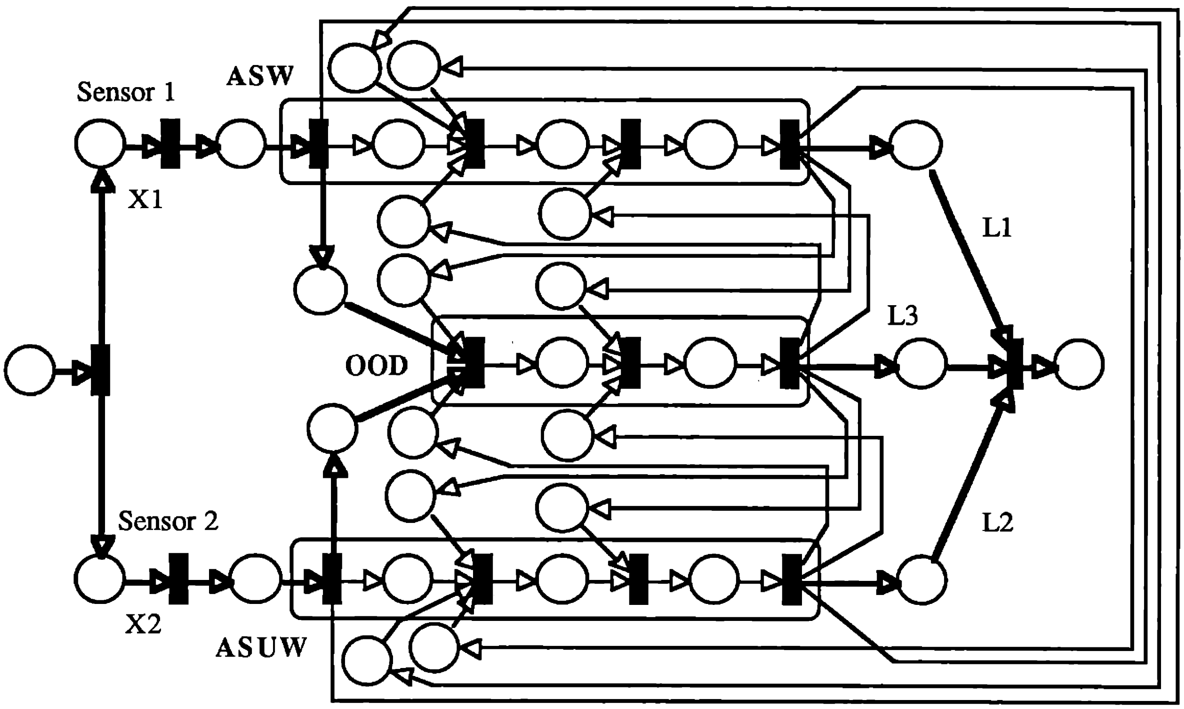


Fig. 8.2 Constraints on the Functional Structure

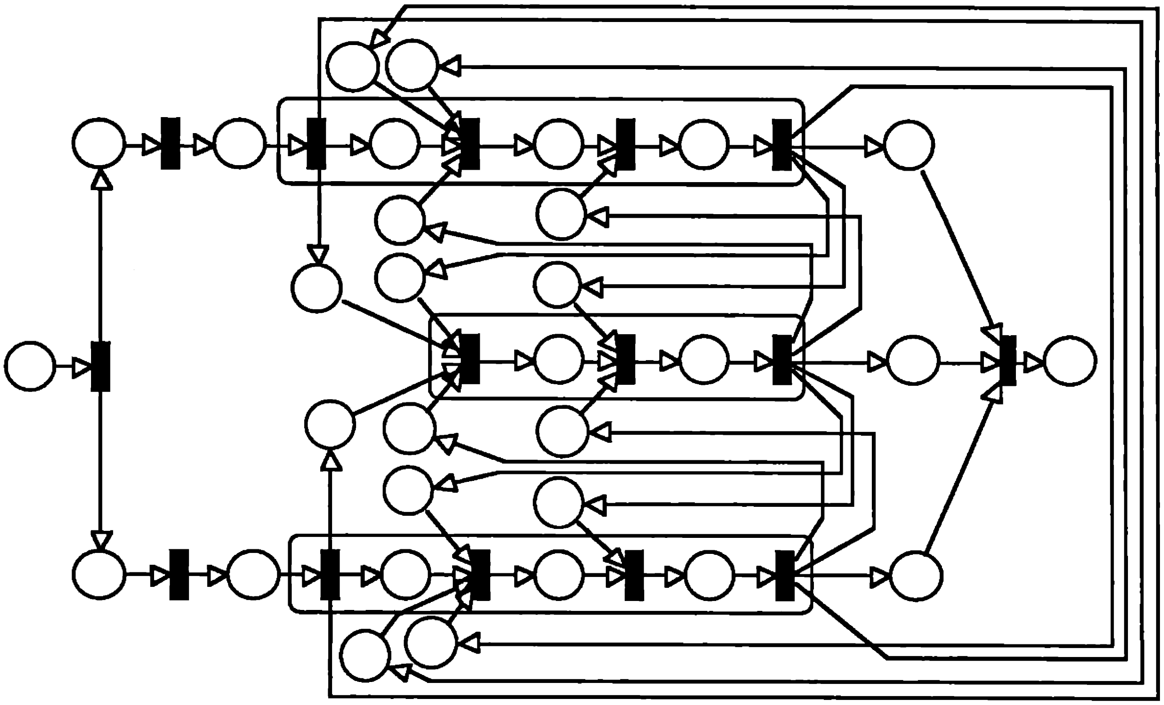


Fig. 8.3 Universal Net for Submarine Example

8.1.4 Minimal and Maximal WDFS

Very few constraints are imposed on the WDFS in W^1, W^2, W^3. Consequently, one finds many minimal and maximal elements, which are presented in Figures 8.4 to 8.9.

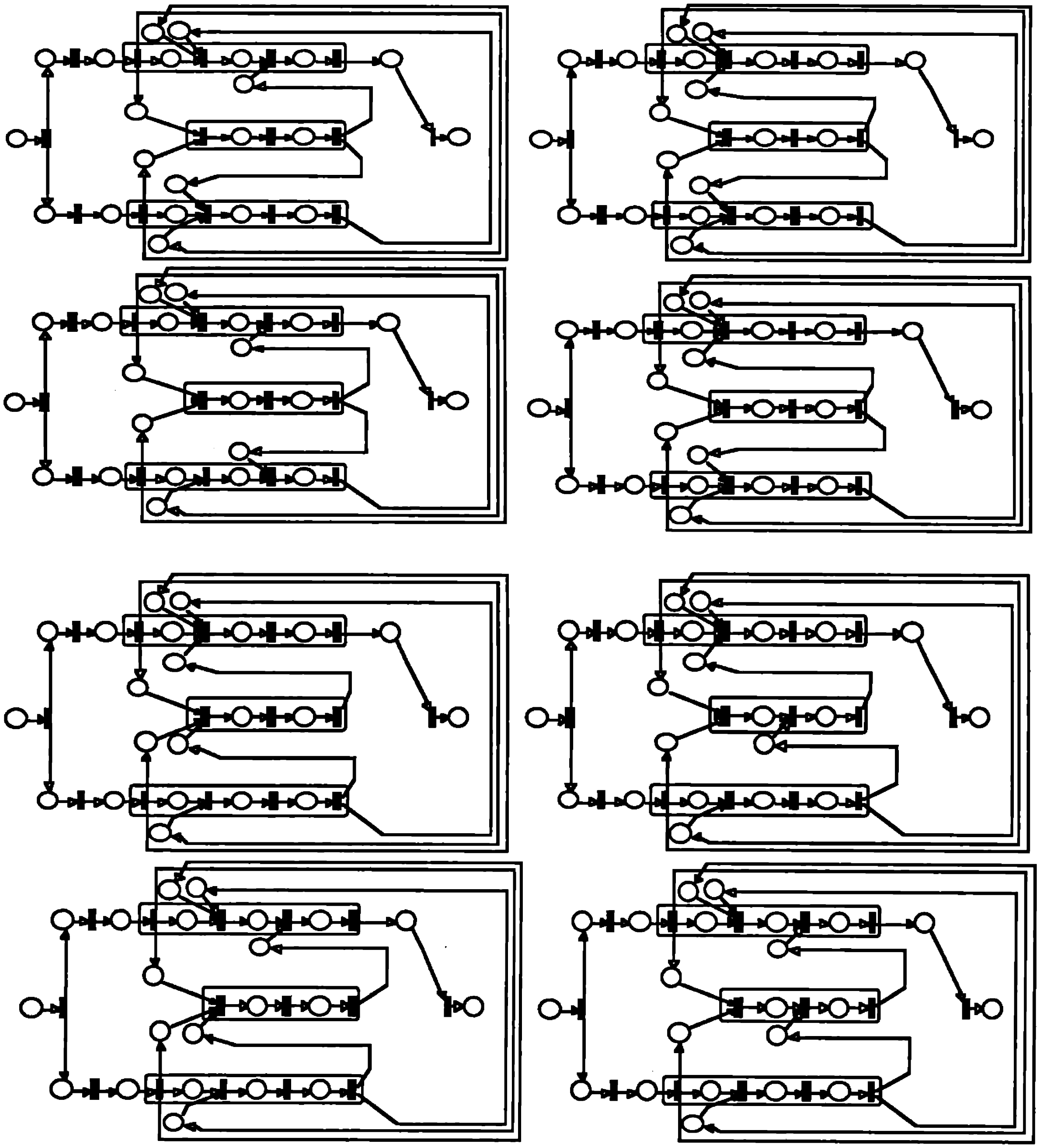


Fig. 8.4 Maximal Elements in W^1

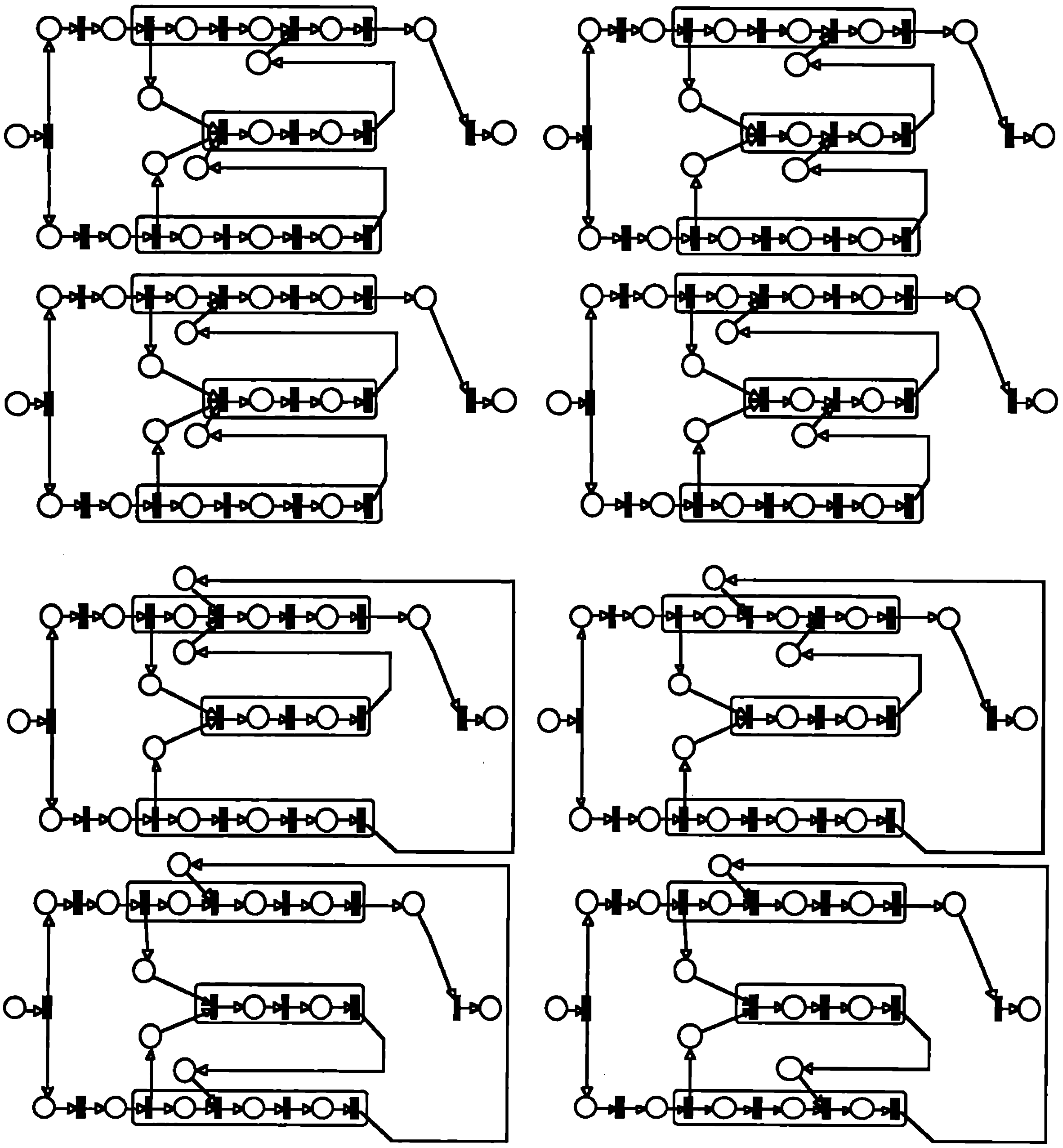


Fig. 8.5 Minimal Elements in W^1

Note that since there are 8 minimal and 8 maximal elements, there are at least 16 elements in W^1 that can be the dataflow structure associated with the elementary set of inputs EX_1.

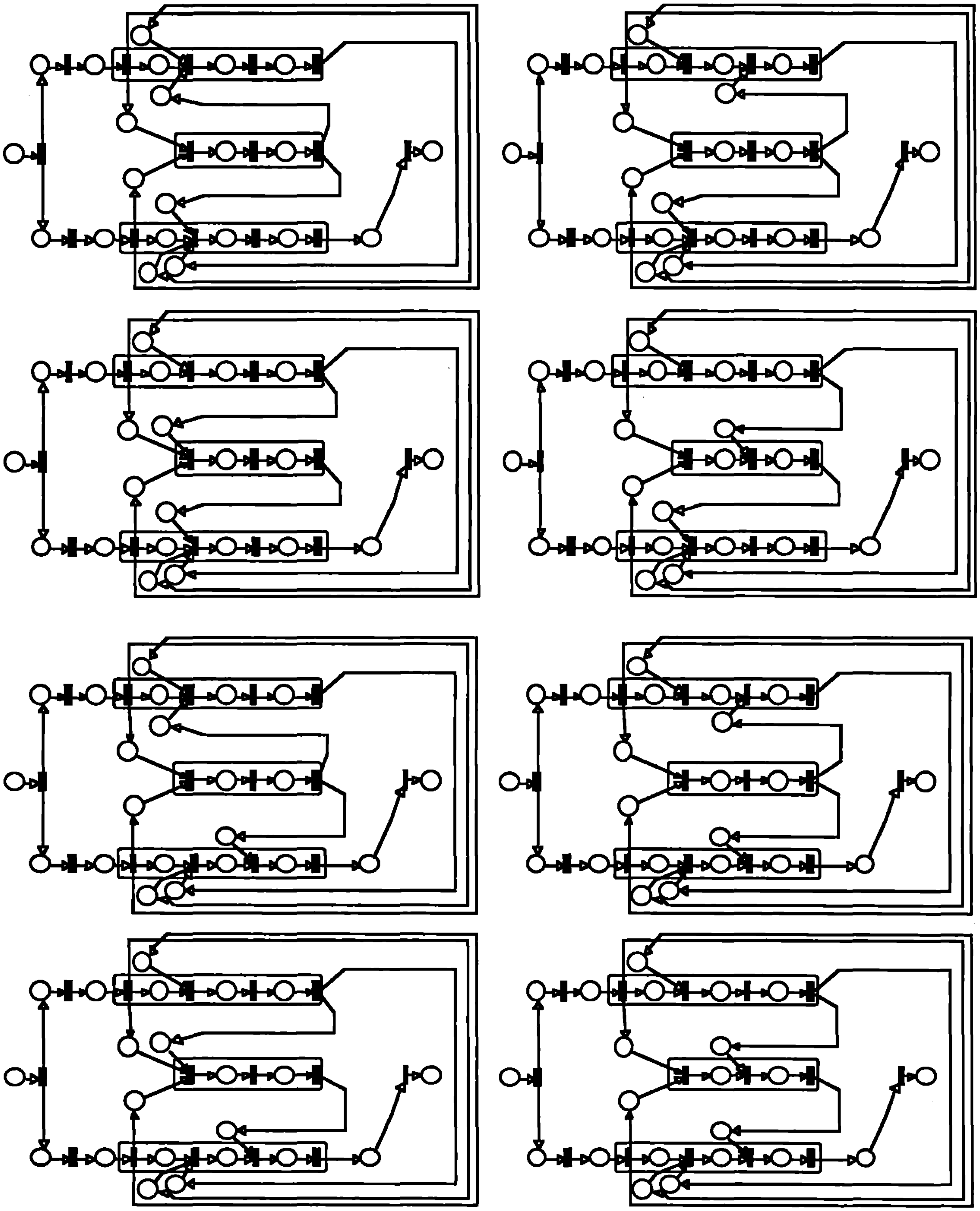


Fig. 8.6 Maximal Elements in W^2

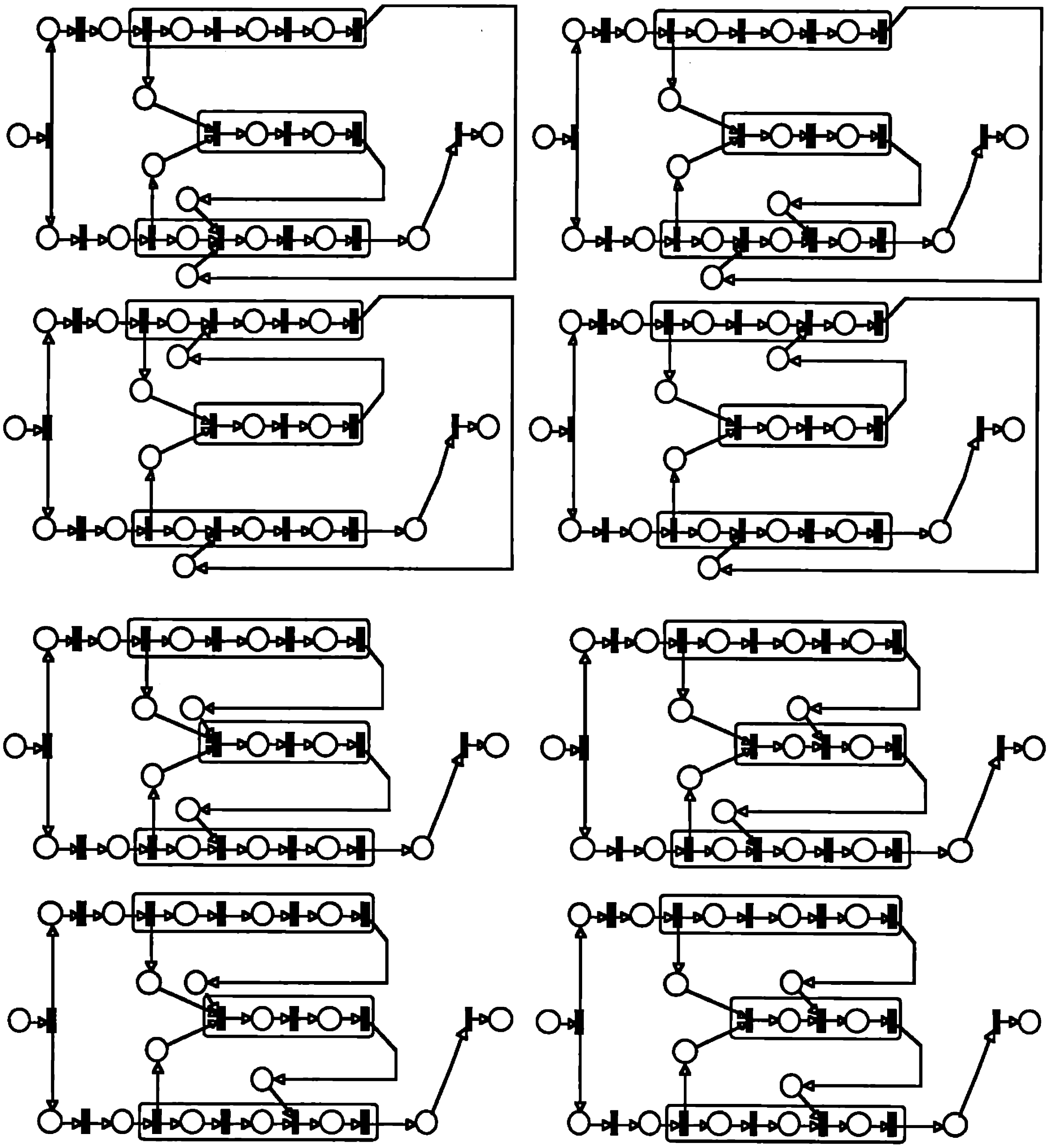


Fig. 8.7 Minimal Elements in W^2

Note that W^2 also contains at least 16 elements, which can be associated with the elementary set of inputs EX_2.

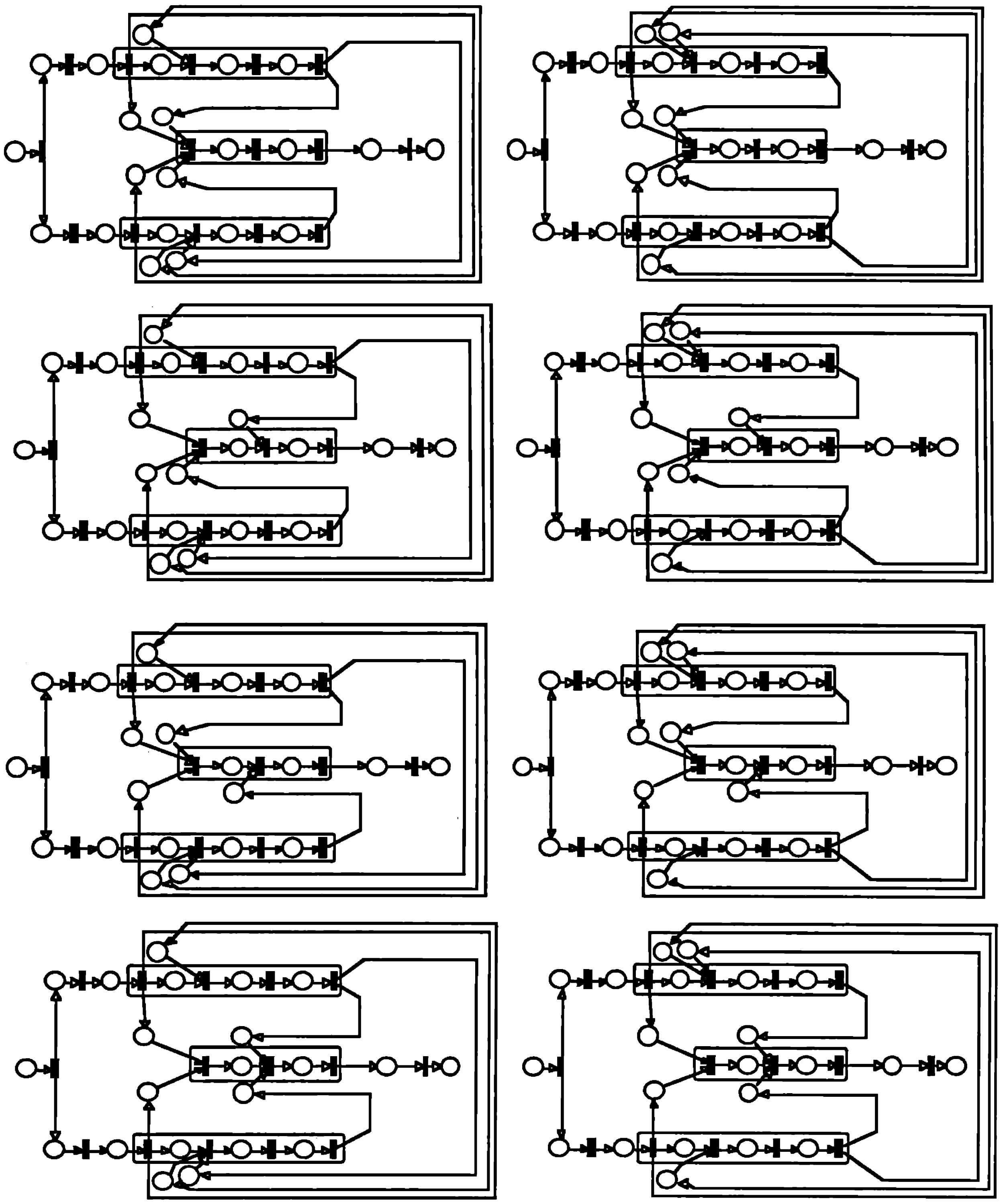


Fig. 8.8 Maximal Elements in W^3

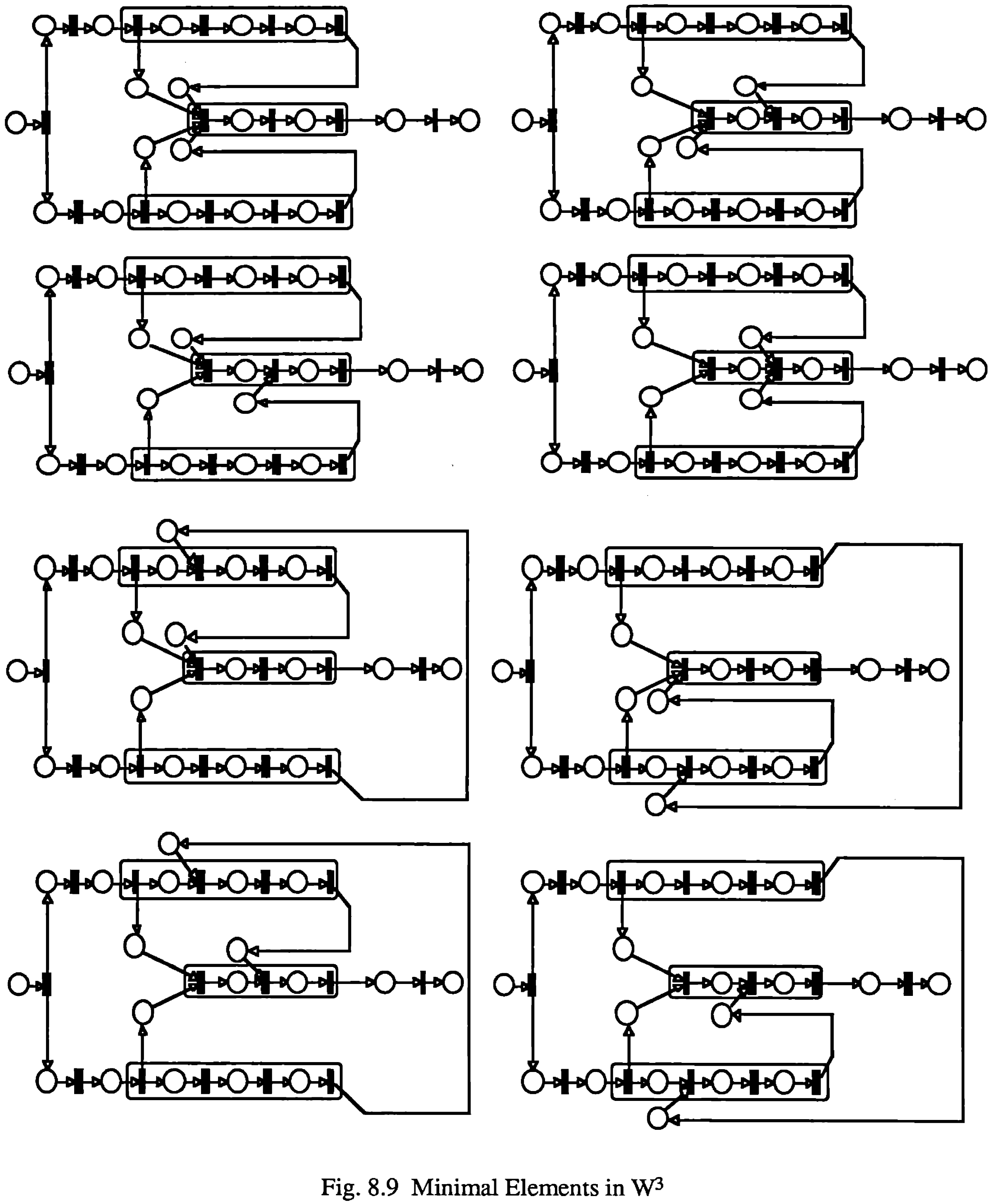


Fig. 8.9 Minimal Elements in W^3

Note finally, that here again W^3 contains at least 16 elements, which can be the fixed functional structure associated with the elementary set of inputs EX_3.

8.1.5 Solutions to the Design Problem

A WDVS that satisfies all constraints of the design except R10 is a mapping from X to W, in which all the elements of EX_1 (EX_2 and EX_3 respectively) are assigned to the same functional structure in W^1 (W^2 and W^3 respectively). There are 16 boundaries in the intersection of each subset W^1, W^2, W^3 with AW and the set of fixed structures that verifies the constraints R_F. By consequence, there are at least $16 * 16 * 16 = 4096$ WDVS that satisfy all constraints but R10 .

The decomposition of R_C distinguished three elementary sets of inputs,
$EX_1 = \{<S,NSU>, <T, NSU>\}$, $EX_2 = \{<NS, dC>, <NS, SU>\}$,
$EX_3 = \{<NS, NSU>, <S, DC>, <S, SU>, <T, dC>, <T, SU>\}$.
Therefore, for any link in the structure, the set AC of inputs that activate the link belongs to the lattice polynomial generated by EX_1, EX_2, and EX_3,
$L(EX_1, EX_2, EX_3) = \{\emptyset, EX_1, EX_2, EX_3, EX_1\cup EX_2, EX_1 \cup EX_3, EX_2 \cup EX_3, X\}$.

- If $AC = \emptyset$ or X, the set of effective alphabets of the partition is the empty set.
- If AC belongs to $\{EX_1, EX_2, EX_3, EX_1\cup EX_2, EX_1 \cup EX_3, EX_2\cup EX_3\}$, X1 and X2 are the effective alphabets of the partition. A variable link describes a valid interaction in a structure Π if and only if the entry of $E(\Pi)$ that corresponds to the link indicates that X1 and X2 are accessible.

The computation of a solution is detailed in Appendix A. This example illustrates how drastic constraint R10 can be, because there is one and only one solution to the design problem. A CPN model of the functional structure is depicted in Figure 8.10. This WDVS has been drawn using the convention that a link that is not activated is not represented, and that a permanent link is not annotated. This WDVS corresponds to the folding of three data flow structures depicted in Figures 8.11, 8.12, and 8.13.

The first WDFS, Σ^1 represents the exchange of information when there is a threat in the submarine environment only. In this pattern of interaction, the situation assessments of ASW and ASUW are sent to every role in the system. ASUW chooses a response which is communicated to the OOD. The OOD incorporates the situation assessments of ASW and ASUW to the response selected by ASUW, and selects a command to be sent to ASW. Finally, ASW interprets this command and chooses an appropriate course of action which is communicated to the effectors.

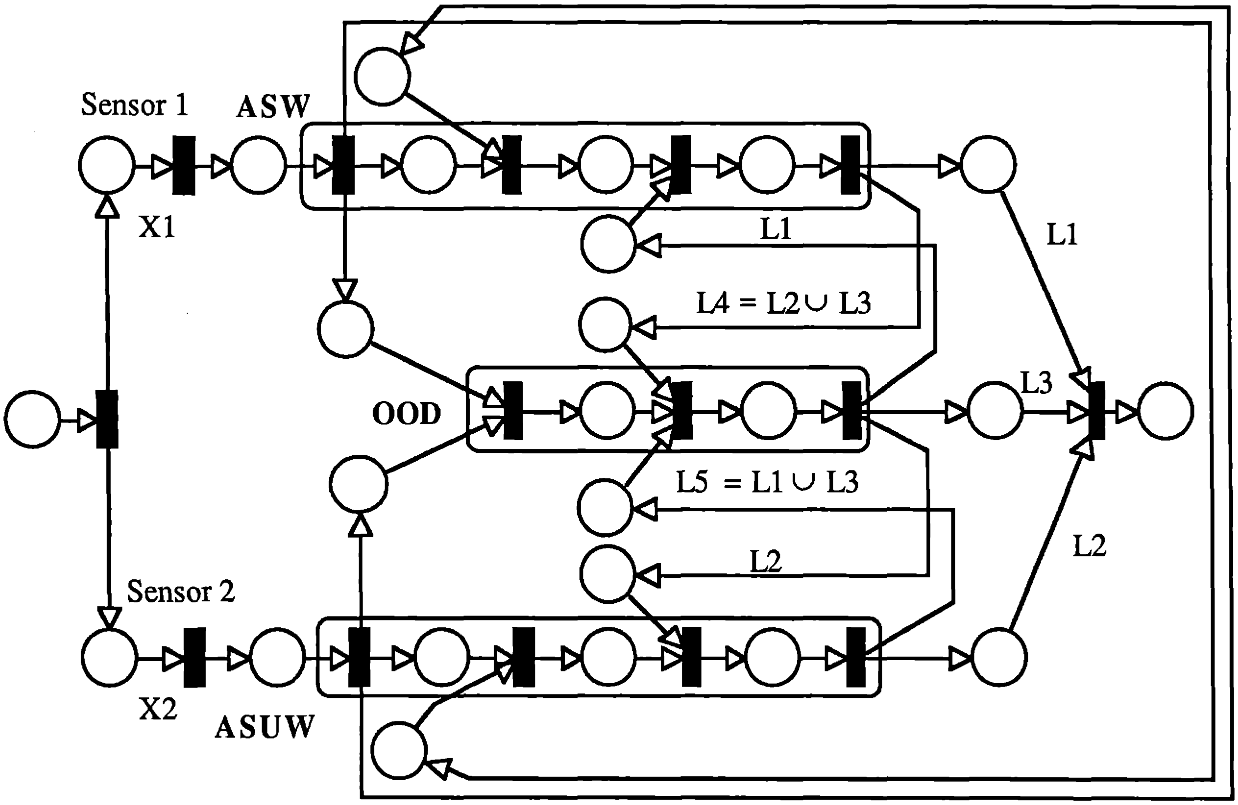


Fig. 8.10 WDVS Solution to the Design Problem

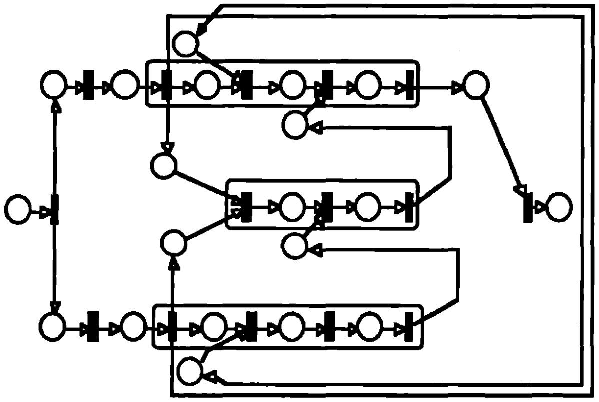


Fig. 8.11 WDFS Σ^1

The WDFS Σ^2 represents the interactions in case of surface threat. The interactions are symmetrical to those in Σ^1. The situation assessments are sent to every role in the system. ASW communicates its response to the Officer of the Deck. OOD incorporates the situation assessments and the response of ASW so as to select a command to be sent to ASUW. Finally, ASUW interprets this command and chooses an appropriate course of action which is communicated to the effectors.

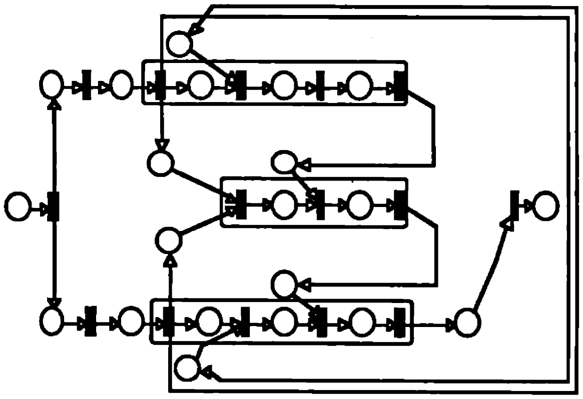


Fig. 8.12 WDFS Σ^2

Finally, if the tactical response is formulated by the Officer of the Deck, the exchange of information between the roles is depicted by Fig. 8.13. The OOD receives the situation assessment of ASW and ASUW, waits for their expert responses, and interprets their expert advisories with his assessment of the mission, so as to produce the response to be communicated to the effectors.

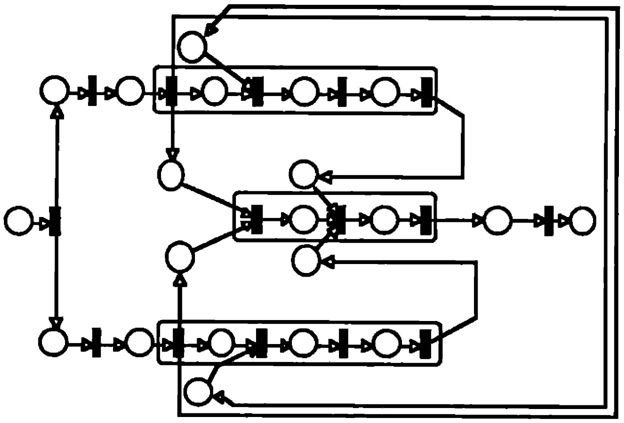


Fig. 8.13 WDFS Σ^3

It must be noted that the variable structure is fairly simple and robust. Each expert subordinate has two modes of interactions overall. Each role always communicates its situation assessment to the other expert subordinate and the OOD. Every expert expects to receive the situation assessment of the other role, irrespective of the tactical parameters. Then, if an expert role has to formulate the response, it waits for a command from OOD. Otherwise, it sends its expert advisory to the Officer of the Deck. Similarly, OOD receives the situation assessments of both subordinates, and infers from them the state of the environment. Then, if OOD has to formulate the tactical response, he waits for the expert responses. Otherwise, he integrates the response from the expert who has not detected a threat with its own assessment, and issues a command to the subordinate who issues the tactical response. The solution to the design problem, including the resource structure, is depicted in Figure 8.14.

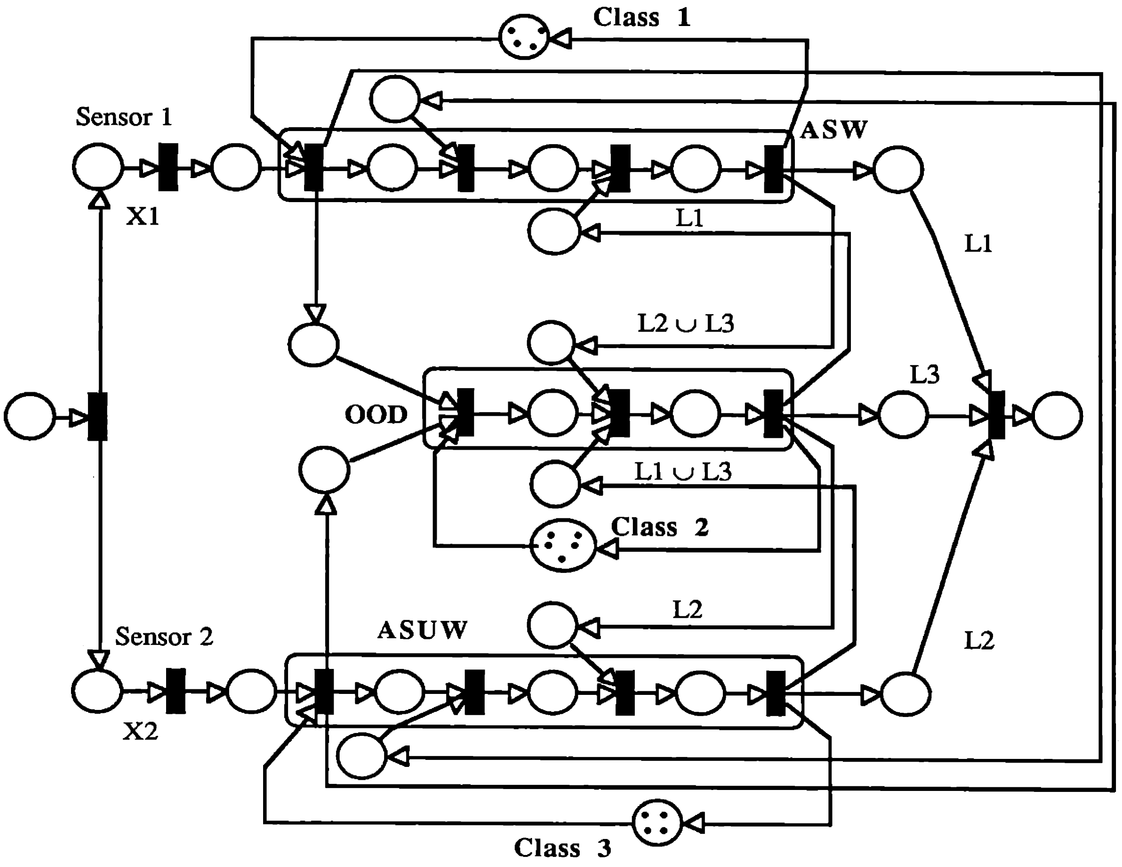


Fig. 8.14 Solution

One can gain in Appendix A some insight into the drastic reduction from at least 4096 candidate structures to one solution. The principal constraint is the fact that one and only one role directly receives the observations from Sensors 1 and 2. Thus, there cannot be variable links for which X1 and X2 are effective until all roles are provided with the situations in both the surface and the submarine environment. This is resolved of by having permanent links from the SA stages of ASW and ASUW to the IF stages of every roles. At that point, the only links that can be variable are the links from the RS stages to the CI stages, and the links from the RS stages to the effectors. The colored constraints and the need for acyclical fixed structures interact so as to yield only one solution.

8.2 THE AIRPORT SURFACE TRAFFIC CONTROL SYSTEM

8.2.1 Problem Definition

This section applies the methodology to the design of a civilian system, the Airport Surface Traffic Control system (ASTC). ASTC is broadly defined as the portion of the Air Traffic Control system that is responsible for traffic on the runways and taxiways of an airport. The system encompasses people, procedures, and equipment. In major US airports, the system consists of two control positions, Local Control and Ground Control, which are stationed in the tower cab using visual surveillance, voice radio, and ground surveillance radars wherever available. Local Control handles the traffic on the runways and in the airspace in the immediate vicinity of the airport, while Ground Control handles the traffic on the taxiways, and, at some airports, issues advisories regarding airplane movements at the ramps. Local Control monitors landings and takeoffs by interacting with the portion of the Air Traffic Control system that is in charge of arrivals and departures in the flight corridors. Local Control can also communicate with Ground Control and with all planes on the runways and taxiways. Ground Control exchanges information with Local Control and with each plane on the ground.

As with all aspects of the Air Traffic Control system, the continual growth in air traffic is placing increasing demands on ASTC. It has been recognized for some time (Federal Aviation Administration, 1979) that Ground Control is a part of ASTC that can significantly slow down the timing of operations and reduce the capacity of the largest airports. Particularly in the case of bad visibility or at peak hours, a single Ground Controller (GC) saturates, and cannot manage the traffic without causing substantial delays. One solution to that problem is to partition the workload of Ground Control between two Ground Controllers. Such a partition is difficult to design and implement, because of the problem of coordinating the tasks, and of resolving conflicts among Ground Controllers, or between Local Control and Ground Control. This example describes how to design a variable ASTC system that encompasses one Local Controller (LC) and two Ground Controllers. This example has been based on an hypothetical design of an ASTC system for Logan Airport in Boston, Massachusetts. The plan of Logan Airport has been simplified, as depicted in Fig. 8.15, in order to illustrate the methodology and the issues associated with the design of an ASTC system without providing too many details, which would be beyond the scope of this thesis.

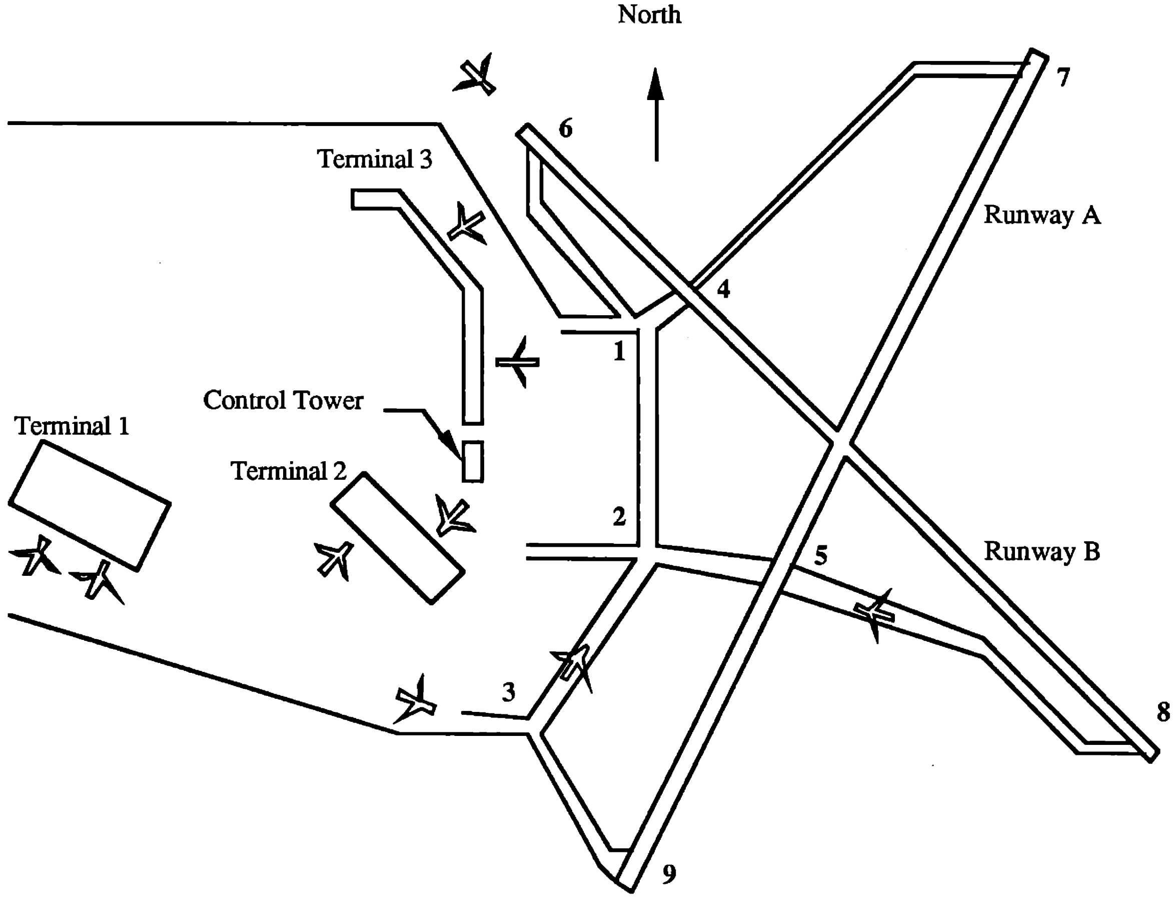


Fig. 8.15 Plan of Logan Airport

8.2.2 System Requirements

The airport of Figure 8.15 has three terminals and two runways, Runways A and B. Planes land and take off on runways, and move to and from the terminals on the taxiways. Local Control monitors the movements on the runways, while Ground Control surveys the taxiways, the crossings of taxiways and runways, and the terminals. The utilization of the runways depends on the direction of the wind, the guiding principle being that landings and takeoffs are done "against the wind." If the wind blows from the North or from the South, Runway A is used. If the wind blows from the East or the West, planes land and take off on Runway B. Finally, both runways can be used if the speed of the wind is below a certain limit. Under low wind conditions the runways are not used in the same way however. Because of noise abatement concerns in the communities around Boston Harbor, large planes land and take off from Runway A exclusively. Runway B is used for general aviation, which generates less noise.

The assumption of the design is that the division of Ground Control between two Ground Controllers is done on a geographic basis. One GC, hereafter called GC 1, is responsible for the southern sector of the airport, while the other GC, GC 2, monitors the northern sector. Terminals 1 and 2 are monitored by Ground Controller 1, while Terminal 3 is monitored by Ground Controller 2. Nine areas have been labeled on Figure 8.15 which indicate critical spots of the ASTC system. Crossings 1, 2, and 3 designate dangerous crossings between taxiways. Crossing 1 lies within the jurisdiction of GC 2 while Crossing 3 is under GC 1's. Crossing 2, however, stands on the boundary between the northern and southern sectors, and is monitored by both Ground Controllers.

Crossings 4 and 5 indicate crossings between a taxiway and a runway. Crossing 4 is monitored by GC 2, while Crossing 5 is monitored by GC 1. If a plane on a taxiway approaches 4 or 5, a Ground Controller must interact with the Local Controller to get the status of the runway, and can authorize or deny the crossing. Finally, Crossings 6, 7, 8, and 9 designate the ends of the runways, the point at which a plane changes jurisdiction between Ground Control and Local Control.

8.2.3 Parameters of the Model.

Two roles, Local Control and Ground Control have already been defined. The degree of redundancy of Local Control is 1, while the degree of redundancy of Ground Control is 2. Five sources of information can trigger variable patterns of interaction in the system.

- Sensor 1 : Wind. Wind is a parameter that influences the direction of landings and takeoffs. This source can take five values: X1 = {0, E, N, S, W}. N (S, E, W respectively) indicates that the wind comes from the North (South, East, West respectively). 0 models low wind conditions in which both runways can be used.
- Sensor 2 : Runway Status. This source of information models the movements on the runways. Its output alphabet is X2 = {LAA, LAB, TOA, TOB, CL}, where LAA indicates that a plane is landing on Runway A, LAB indicates that a plane is landing on Runway B, TOA the fact that a plane is taking off from runway A, TOB, the fact that a plane is taking off from Runway B, and CL the possibility of the runways being clear of any movement. Note that one and only one runway is used if the output of Sensor 1 is 0, E, N, S or W. Both runways can be used under low wind conditions but the safety standards require that there cannot be simultaneous use of both runways, i.e., there cannot be at the same time one movement on the Runways A and B.
- Sensor 3 : Terminals 1 and 2. This source models a plane leaving Terminal 1 or Terminal 2. Its output alphabet is X3 = {DP1, NDP1} where DP1 models a departure from Terminal 1 or Terminal 2, and NDP1 indicates that no plane is departing. Note that a departing plane must be directed to a runway. If several runways are used at the same time, Ground Controller 1 must ask the Local Controller where to direct the plane.
- Sensor 4 : Terminal 3. Similarly, this source models a plane leaving Terminal 3. Its output alphabet is X4 = {DP3, NDP3}, where DP3 models the case in which one plane is leaving, and NDP3 the case in which no plane leaves. Note that GC2 must ask the Local Controller where to direct the plane in case of low wind conditions.
- Sensor 5 : Conflicts. This source of information models the existence of conflicts at the boundaries of the northern and the southern sectors. Its output alphabet is X5 = {C, NC} where C indicates that there is some conflict in the area that is monitored by both Ground Controllers, whereas NC models the absence of conflict.

The set of inputs to the system is $X = X1 \times X2 \times X3 \times X4 \times X5$. The set of inputs contains $5 * 5 * 2 * 2 * 2 = 200$ elements. Recall that these inputs model only the parameters that necessitate or influence coordination between roles. They do not describe extensively the parameters that are processed by one role only.

Finally, suppose that one and only one human decisionmaker can be assigned to each role corresponding to a situation in which no back-up controllers are present in the tower cab. In a Petri Net model of the structure, one and only one resource loop is associated with each role. The resource place in the resource loop initially contains one and only one token. The resource structure is very simple and can be omitted below. Functional structures are thus used to represent the structure of interactions in the system.

8.2.4 Constraints

Constraints are introduced to restrict the solutions to WDVS that make sense as far as an ASTC system for Logan airport is concerned. These constraints are expressed in plain English in this subsection and can be translated without any difficulty into mathematical terms, as done in section 8.1.2 and in section 6.4. They are represented, with the usual conventions, on the CPN of Figure 8.16.

- Constraints on S

It is assumed that each role knows the direction of the wind, the output of Sensor 1. The output of Sensor 2, the status of the runway(s), is known by LC only. The output of Sensor 3, the status of Terminals 1 and 2, is monitored by GC 1, but not by GC 2. LC may have access to that source of information, for example, by looking at departure strips on a monitor or by knowing in advance the planning of departures. Similarly, the output of Sensor 4, that is the status of Terminal 3, is monitored by GC 2, may be known by LC, and is not surveyed by GC 1. Finally, the conflicts at the boundaries of the northern and southern sectors, the output of Sensor 5 are only known to the GCs.

- Constraint on s

Every role has to produce a response to be the sent to the planes under its jurisdiction, and each role must have a fixed link from its RS stage to the effectors.

- Constraints on F

No constraints are imposed on sharing information between Ground Controllers or from a GC to LC. Some constraints are imposed on the exchange of information from LC to the Ground Controllers, because LC must inform the Ground Controllers of the Runway Status. There is no need for fixed interactions because the utilization of the runways does not always create conflicts or dangers in both the northern and the southern sectors. If the wind is N or W, the landing planes leave the runway at 6 or 7, under the jurisdiction of GC 2, and head for the terminals through the crossings 1 and 4, all monitored by GC 2. The departing planes take off from 8 or 9. The way they follow on the taxiways to crossings 8 and 9 is mainly under the jurisdiction of GC 1, and they cannot cross the runway that is used. Therefore, GC 2 only needs to be informed by LC. Symmetrically, if the wind is E or S, GC 1 needs to be informed only by LC. If Runways A and B are in use however, both Ground Controllers must be informed, because each GC monitors a crossing (4 or 5) between a runway and a taxiway.

These constraints can be summed up by stating that AC = $<E> \cup <S> \cup <0>$ is the set of inputs that activates the link from the SA stage of the Local Controller to the IF stage of Ground Controller 1. This variability is depicted by the matrix L1 on Figure 8.16. Similarly, the set of inputs that activate the link from the SA stage of the Local Controller to the IF stage of Ground Controller 2 is AC = $<N> \cup <W> \cup <0>$. This variable pattern of interaction is indicated by matrix L2 on Figure 8.16.

- Constraint on G

Every SA stage receives some sensors' observations, therefore all links between RS stages and SA stages must be ruled out.

- Constraint on H

All links from the RS stages to the IF stages have been ruled out. The rationale is that a response of a role is an executory command, a resolution of a conflict or an answer to a request. Therefore, the communications of responses to roles are most appropriately modeled by the links from the RS stages to the CI stages.

- Constraint on H

First, it is assumed that GCs cannot issue commands to LC because the movements on the runways have priority as far as safety is concerned. The number of alternative courses of action that LC can select shall never be restricted. Second, GC 2 cannot issue a command to GC1, whereas GC 1 may issue one to GC 2. GC 1 covers a larger geographical area than GC 2, and may need to restrict the courses of action of GC2 to solve the conflicts in its own jurisdiction. Finally, in the case of low wind conditions, LC must instruct the Ground Controllers when a plane leaves a terminal. The link from the RS stage of LC to the CI stage of GC 1 is thus activated for AC = $<0, DP3>$. This variability has been indicated by the matrix L3. Similarly, the link from the RS stage of LC to the CI stage of GC 2 is activated by the set AC = $<0, DP1>$. This variability has been represented by matrix L4.

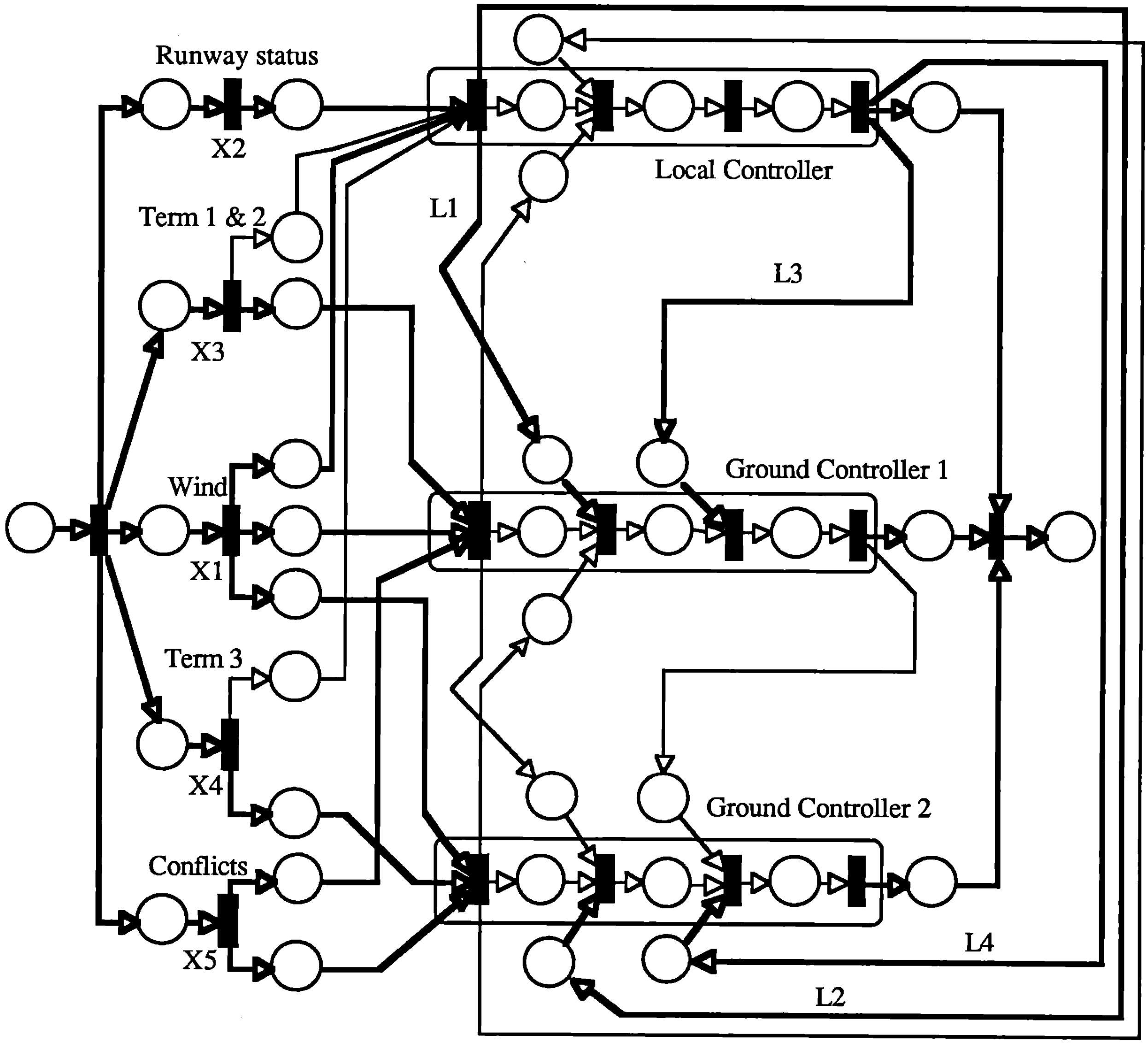


Fig. 8.16 Colored Petri Net model of the Constraints

8.2.5 Decomposition of the Constraints

The elementary sets of inputs can be determined from the colored constraints expressed in L1, L2, L3, and L4 : $EX_1 = <N> \cup <W>$, $EX_2 = <E> \cup <S>$, $EX_3 = < 0, DP1, DP3>$, $EX_4 = <0, DP1, NDP3>$, $EX_5 = <0, NDP1, DP3>$, $EX_6 = <0, NDP1, NDP3>$. The set of inputs that activate one link belongs to the lattice polynomial generated by $EX_1, ..., EX_6$. Note that any elementary set of inputs corresponds to a union of equivalence classes of R_1, R_3, R_4, because they correspond to partitions based on the observations carried out by Sensors 1, 3, and 4. Therefore, the effective alphabets of any partition of X between AC and DC are at most X1, X3, and X4, and the model of this thesis does not characterize solutions in which X2 and X5 may be effective alphabets of the partition.

Finally, it can be seen that

- There are no effective alphabets if AC = Ø or X.
- X1 is the only effective alphabet if $AC = EX_1$ or EX_2 or $EX_1 \cup EX_2$ or $EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$.
- X1 and X3 are the effective alphabets if $AC = EX_3 \cup EX_4$ or $EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4$ or $EX_2 \cup EX_3 \cup EX_4$ or $EX_1 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_5 \cup EX_6$.
- X1 and X4 are the effective alphabets if $AC = EX_3 \cup EX_5$ or $EX_4 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_5$ or $EX_2 \cup EX_3 \cup EX_5$ or $EX_1 \cup EX_4 \cup EX_6$ or $EX_2 \cup EX_4 \cup EX_6$.
- Finally, X1, X3, X4 are the effective alphabets for all other AC in $L\ (EX_1, ..., EX_6)$.

8.2.6 Solutions

There are four subsets of solutions to the design problem. As indicated by Proposition 7.24, each subset of solutions corresponds to a particular combination of input links to the system. In addition, each subset of solutions in the example has the property that there exists a unique minimal element and a unique maximal element in each subset. These minimal and maximal elements are depicted in the following figures. The usual convention that a permanent link is not annotated is respected throughout Figures 8.17 to 8.28.

- Subset 1

The first subset of solutions corresponds to the variable structures whose input links are exactly the inputs links that have been imposed by the constraints. The VMINO is the WDVS of Figure 8.17. In this variable structure, all links except the ones whose variability has been defined are permanent. LC does not have access to direct knowledge of the departures at the terminals, and receives this information over the permanent links from the SA stages of both GCs to its IF stage. Besides these two links, all links are permanent or variable as imposed by the user-defined constraints. The VMAXO of the subset of solutions is depicted on Figure 8.18.

As compared to the VMINO, three new links are activated, i.e., the links from the SA stages to the IF stages between Ground Controllers, and the link between the RS stage of GC 1 to the CI stage of GC2. This VMAXO describes a structure in which, in addition to the interactions that are required, all roles share their situation assessments, and GC 1 always issues a command to GC 2.

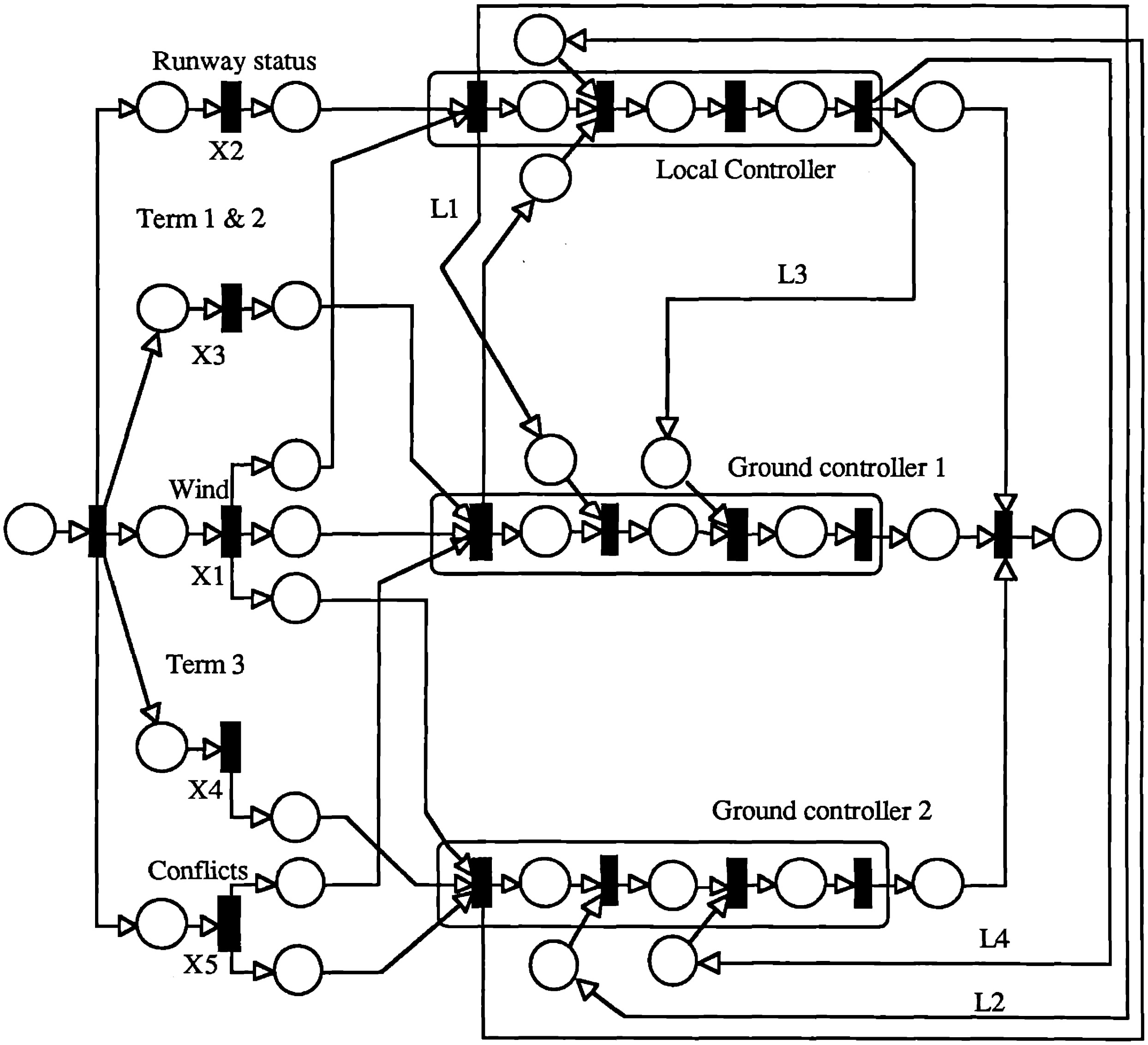


Fig. 8.17 VMINO 1

There are two layers of solutions from VMINO 1 to VMAXO 1. These layers correspond to different accessible alphabets for Link A, the link from the SA stage of GC 1 to the IF stage of GC 2, Link B, the link from the SA stage of GC 2 to the IF stage of GC 1, Link C, the link from the SA stage of GC 1 to the IF stage of GC 2, the links that vary between VMINO 1 and VMAXO 1. The accessible pattern of the solutions can be expressed in a reduced form by [{Accessible alphabets Link A},{Accessible alphabets Link B},{Accessible alphabets Link C}].

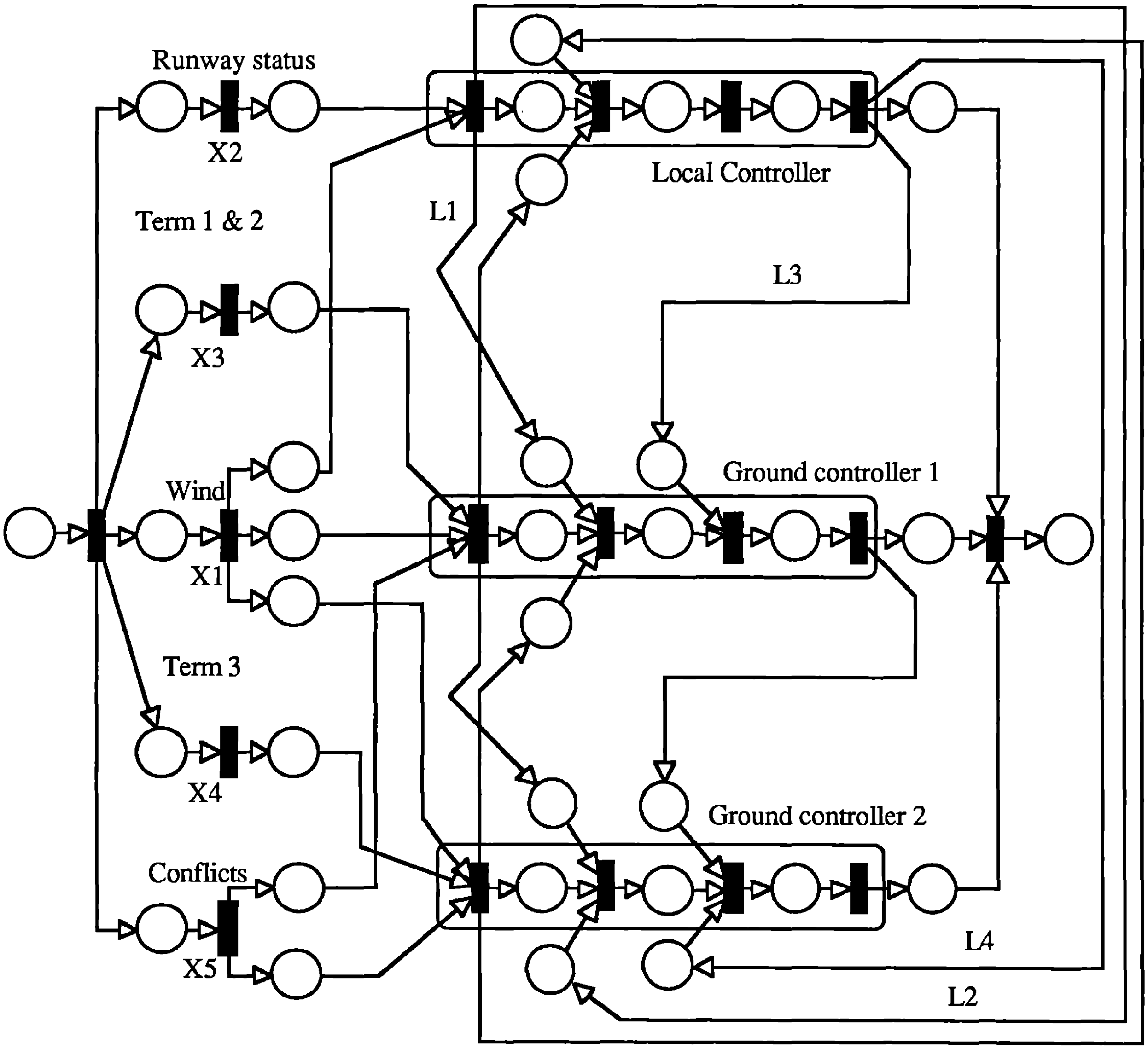


Fig. 8.18 VMAXO 1

VMINO 1 has an accessible pattern that corresponds to [{X1}, {X1}, {X1}]. Therefore any WDVS that lies between VMINO 1 and VMAXO 1 and whose links A, B, and C are activated by AC $= EX_1$ or EX_2 or $EX_1 \cup EX_2$ or $EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ is a valid solution to the problem. These solutions are the variable structures for which all changes are only based on the direction of the wind, i.e., on the way runways are utilized for landings and takeoffs.

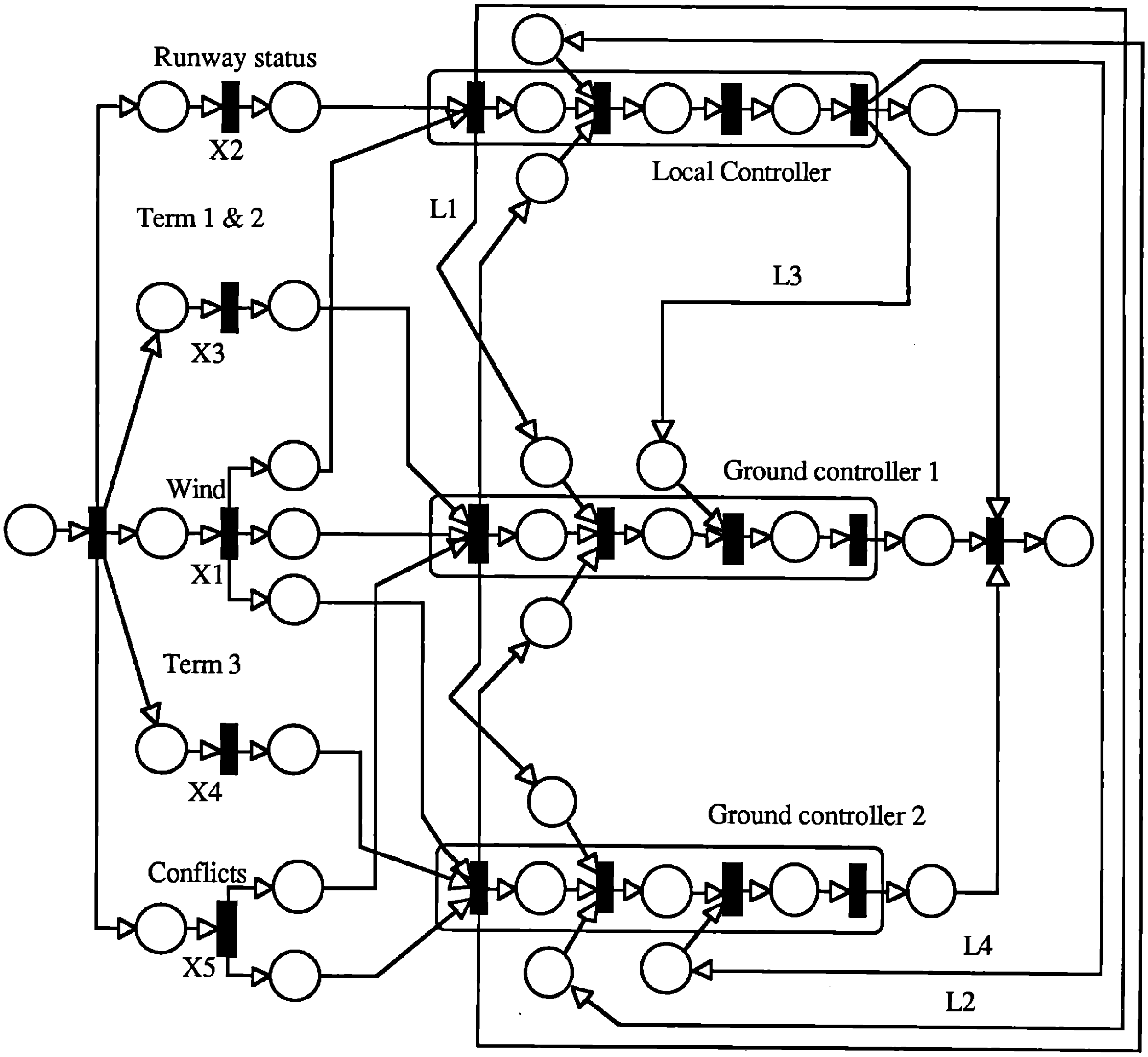


Fig. 8.19 Intermediate Solution

The second layer is made of the WDVSs that lie between VMINO 1 and VMAXO 1 and above the solution in Fig. 8.19, whose effective pattern is [{X1},{X1},{X1, X3, X4}]. These solutions are the ones in which roles always exchange their situation assessments. The link between the RS stage of GC 1 and the CI stage of GC 2 can be activated by any set of inputs in L(EX_1, .., EX_6), i.e. GC 1 can issue a command to GC 2 for any combination of cases.

- Subset 2

This set of solutions corresponds to the structures in which the Local Controller has direct knowledge of planes departing from terminals 1 and 2, but not of the departures from terminal 3. There exists a unique minimal solution that is represented in Figure 8.20. In this structure, GC 2 must send its situation assessment to the Local Controller, while all other links had been determined by the constraints on the design.

There is one and only one maximal solution, which is depicted in Figure 8.21. As compared to VMINO 2, four links are activated, that is the links A, B, and C described for the first subset, and the link D from the RS stage of GC 1 to the IF stage of LC. There are two layers from VMINO 2 to VMAXO 2. Here again, the accessible pattern of the solutions can be expressed in a reduced form by:
[{Accessible alphabets Link A}, {Accessible alphabets Link B}, {Accessible alphabets Link C}, {Accessible alphabets Link D}].

VMINO 2 has an accessible pattern that corresponds to [{X1}, {X1}, {X1}, {X1, X3}]. Therefore a solution to the design lies between VMINO 2 and VMAXO 2 and

- Its links A, B, and C are activated by AC = EX_1 or EX_2 or $EX_1 \cup EX_2$ or $EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$.
- Its link D is activated by AC = $EX_3 \cup EX_4$ or $EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4$ or $EX_2 \cup EX_3 \cup EX_4$ or $EX_1 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_5 \cup EX_6$.

The second layer is made of the WDVSs between VMINO 2 and VMAXO 2 that are above the structure of Fig. 8.22, whose effective pattern of interactions is:

[{X1}, {X1}, {X1, X3, X4}, {X1, X3}].

These solutions correspond to structures in which GC1 and GC2 always exchange their situation assessments, in which Link C can be activated by any set in L(EX_1,..., EX_6), and Link D can be activated by AC = $EX_3 \cup EX_4$ or $EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4$ or $EX_2 \cup EX_3 \cup EX_4$ or $EX_1 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_5 \cup EX_6$.

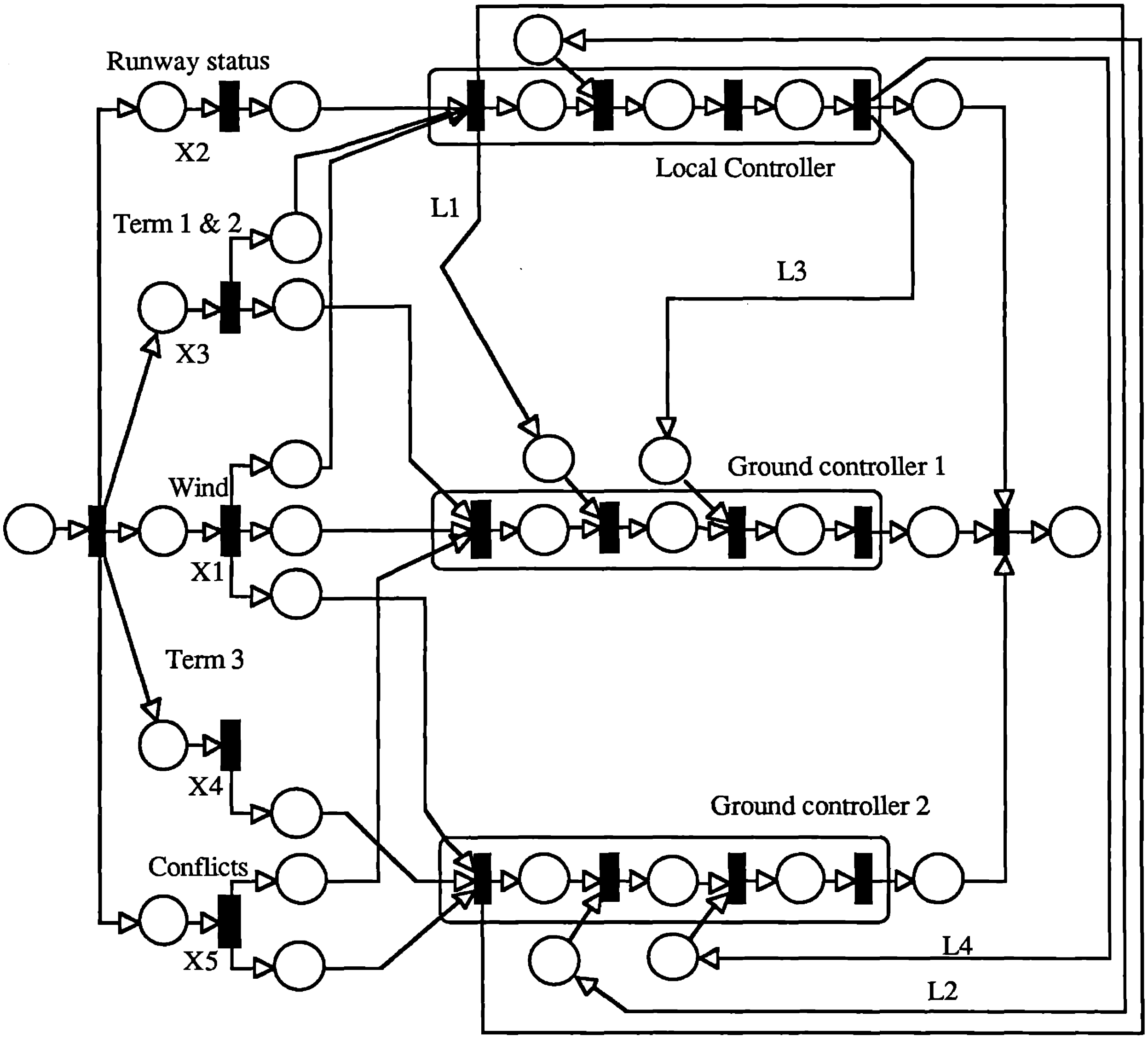


Fig. 8.20 VMINO 2

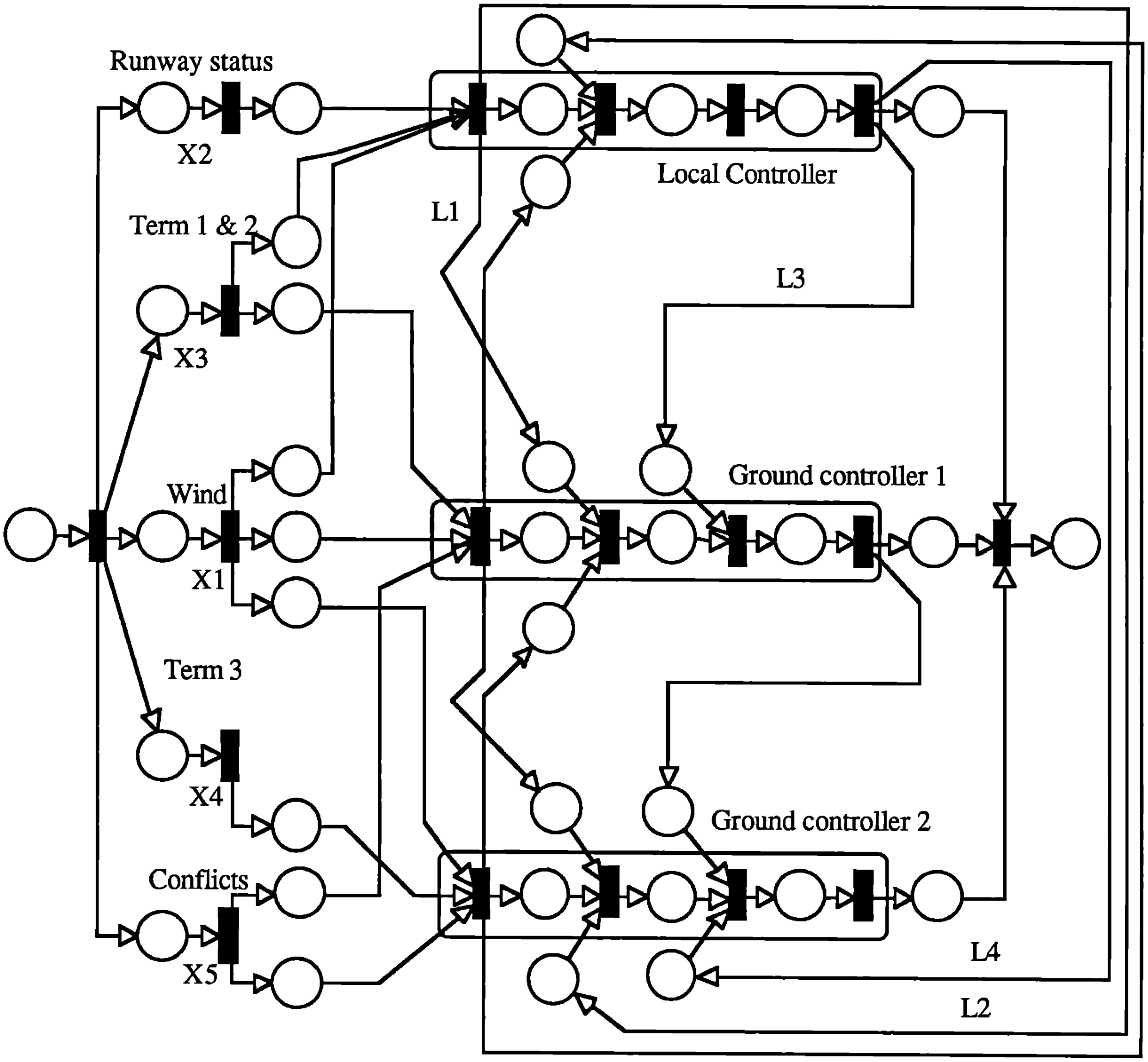


Fig. 8.21 VMAXO 2

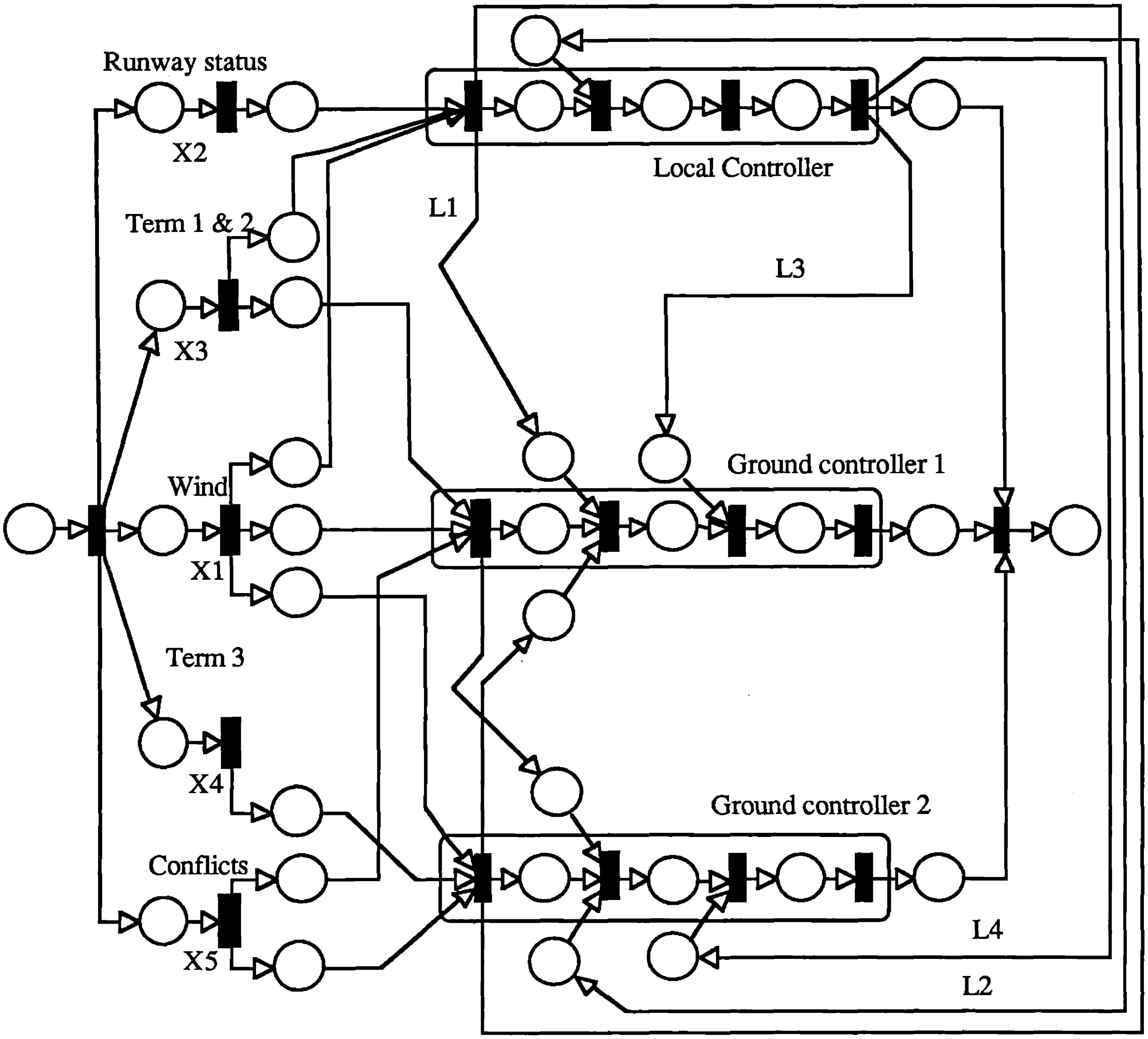


Fig. 8.22 Intermediate Solution

- Subset 3

This set of solutions corresponds to the structures in which the Local Controller has direct knowledge of planes departing from terminal 3, but not of the departures from terminals 1 and 2. There exists a unique minimal solution that is represented in Figure 8.23. In this structure, GC 1 must send its situation assessment to the Local Controller, while all other links have been determined by the constraints on the design.

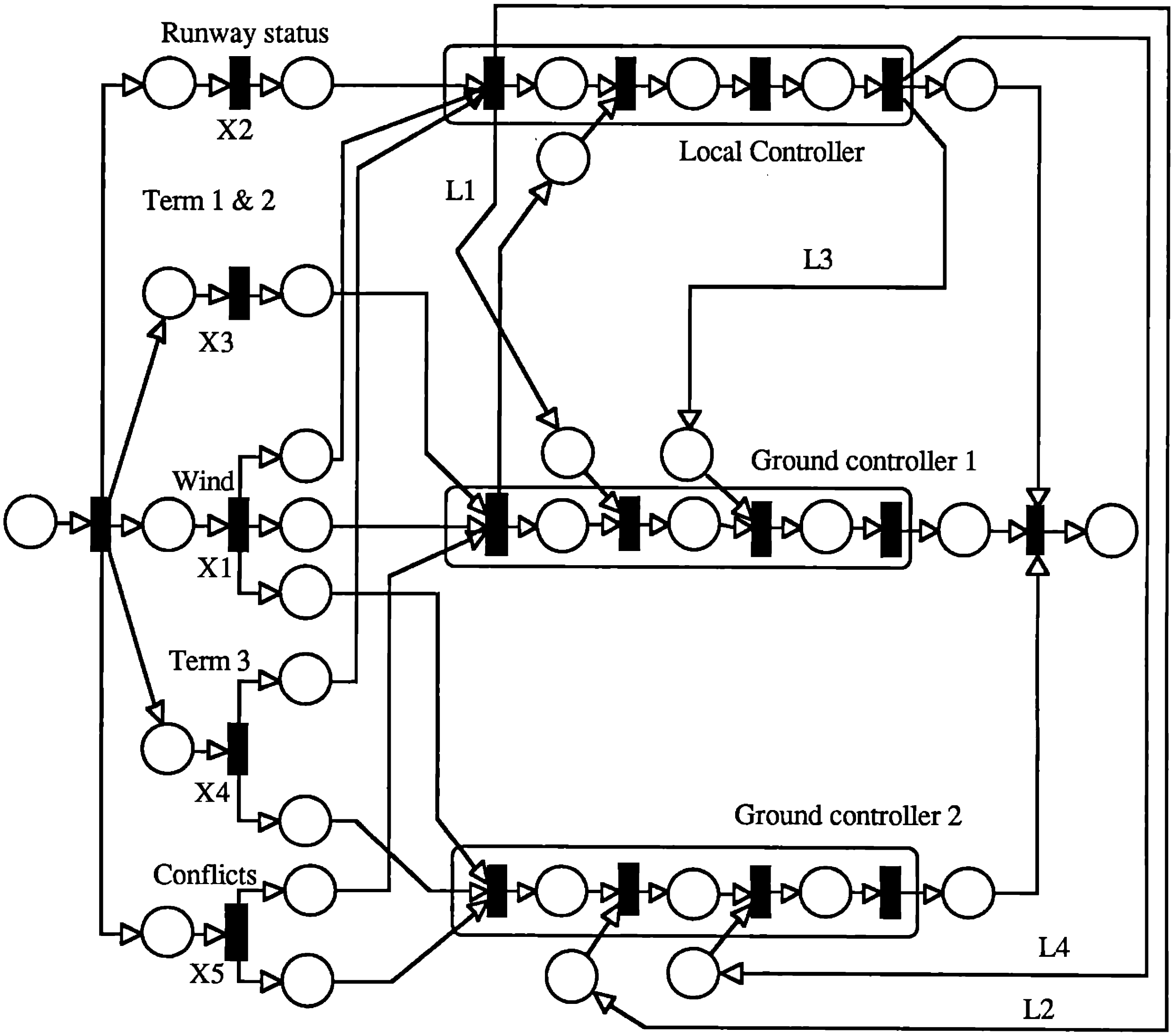


Fig. 8.23 VMINO 3

There is one and only one maximal solution, which is depicted in Figure 8.24. As compared to VMINO 3, four links are activated: The links A, B, and C described for the first subset, and the link E from the RS stage of GC 2 to the IF stage of LC. There are two layers from VMINO 3 to VMAXO 3. Here again, the accessible pattern of the solutions can be expressed in a reduced form by:
[{Accessible alphabets Link A}, {Accessible alphabets Link B}, {Accessible alphabets Link C}, {Accessible alphabets Link E}].

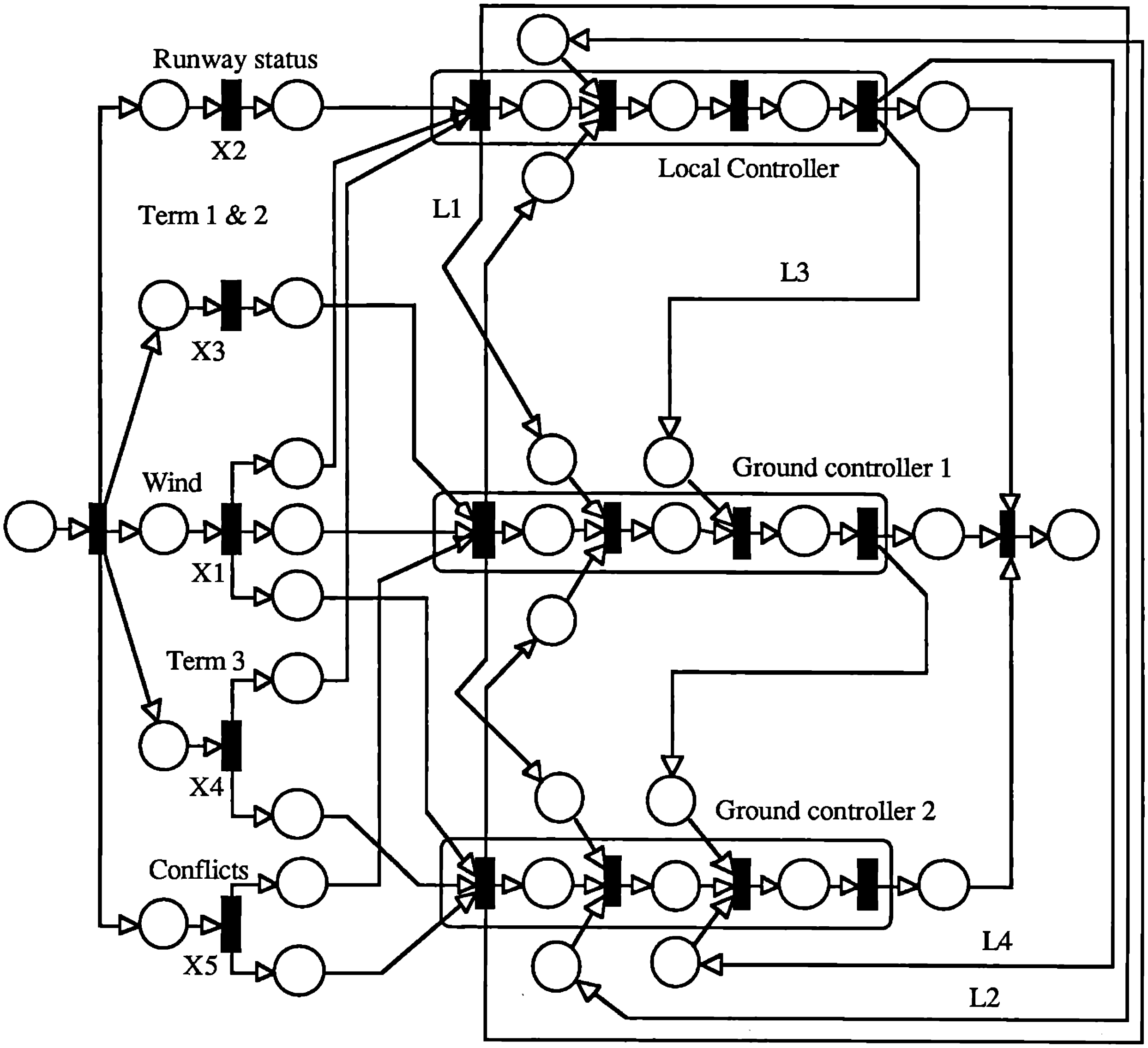


Fig. 8.24 VMAXO 3

VMINO 3 has an accessible pattern [{X1}, {X1}, {X1}, {X1, X4}]. Therefore, a valid solution is a WDVS that lies between VMINO 3 and VMAXO 3,

- whose links A, B, C are activated by AC = EX_1 or EX_2 or $EX_1 \cup EX_2$ or $EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$
- whose link E is activated by AC = $EX_3 \cup EX_5$ or $EX_4 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_5$ or $EX_2 \cup EX_3 \cup EX_5$ or $EX_1 \cup EX_4 \cup EX_6$ or $EX_2 \cup EX_4 \cup EX_6$.

Finally, the accessible pattern of any WDVS between VMINO 3 and VMAXO 3 that lies above the solution depicted on Fig. 8.25 is [{X1}, {X1}, {X1, X3, X4}, {X1, X4}]. The second layer of solutions is the structures in which GC1 and GC2 always exchange their situation assessments, in which Link C can be activated by any set of inputs in $L(EX_1, ..., EX_6)$ and Link E can be activated by AC = $EX_3 \cup EX_5$ or $EX_4 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_5$ or $EX_2 \cup EX_3 \cup EX_5$ or $EX_1 \cup EX_4 \cup EX_6$ or $EX_2 \cup EX_4 \cup EX_6$.

- Subset 4

This final set of solutions corresponds to the structures in which the Local Controller has direct knowledge of planes departing from terminals 1, 2, and 3. There exists a unique minimal solution that is represented in Figure 8.26. In this structure, all links excepts the links from Sensor 3 and Sensor 4 to LC have been defined as constraints to the design.

There is one and only one maximal solution, which is depicted in Figure 8.27. As compared to VMINO 4, five links are activated, the links A, B, C, D, and E . Once again, there are two layers from VMINO 4 to VMAXO 4, and the accessible patterns of the solutions can be expressed in reduced form by [{Accessible alphabets Link A}, {Accessible alphabets Link B}, {Accessible alphabets Link C}, {Accessible alphabets Link D}, {Accessible alphabets Link E}].

The accessible pattern of VMINO 4 is [{X1}, {X1}, {X1}, {X1, X3}, {X1, X4}]. Therefore, a solution lies between VMINO 4 and VMAXO 4, and

- Its links A, B, C are activated by AC = EX_1 or EX_2 or $EX_1 \cup EX_2$ or $EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_3 \cup EX_4 \cup EX_5 \cup EX_6$.
- Its link D can be activated by AC = $EX_3 \cup EX_4$ or $EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4$ or $EX_2 \cup EX_3 \cup EX_4$ or $EX_1 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_5 \cup EX_6$.

- Its link E is activated by AC = $EX_3 \cup EX_5$ or $EX_4 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_5$ or $EX_2 \cup EX_3 \cup EX_5$ or $EX_1 \cup EX_4 \cup EX_6$ or $EX_2 \cup EX_4 \cup EX_6$.

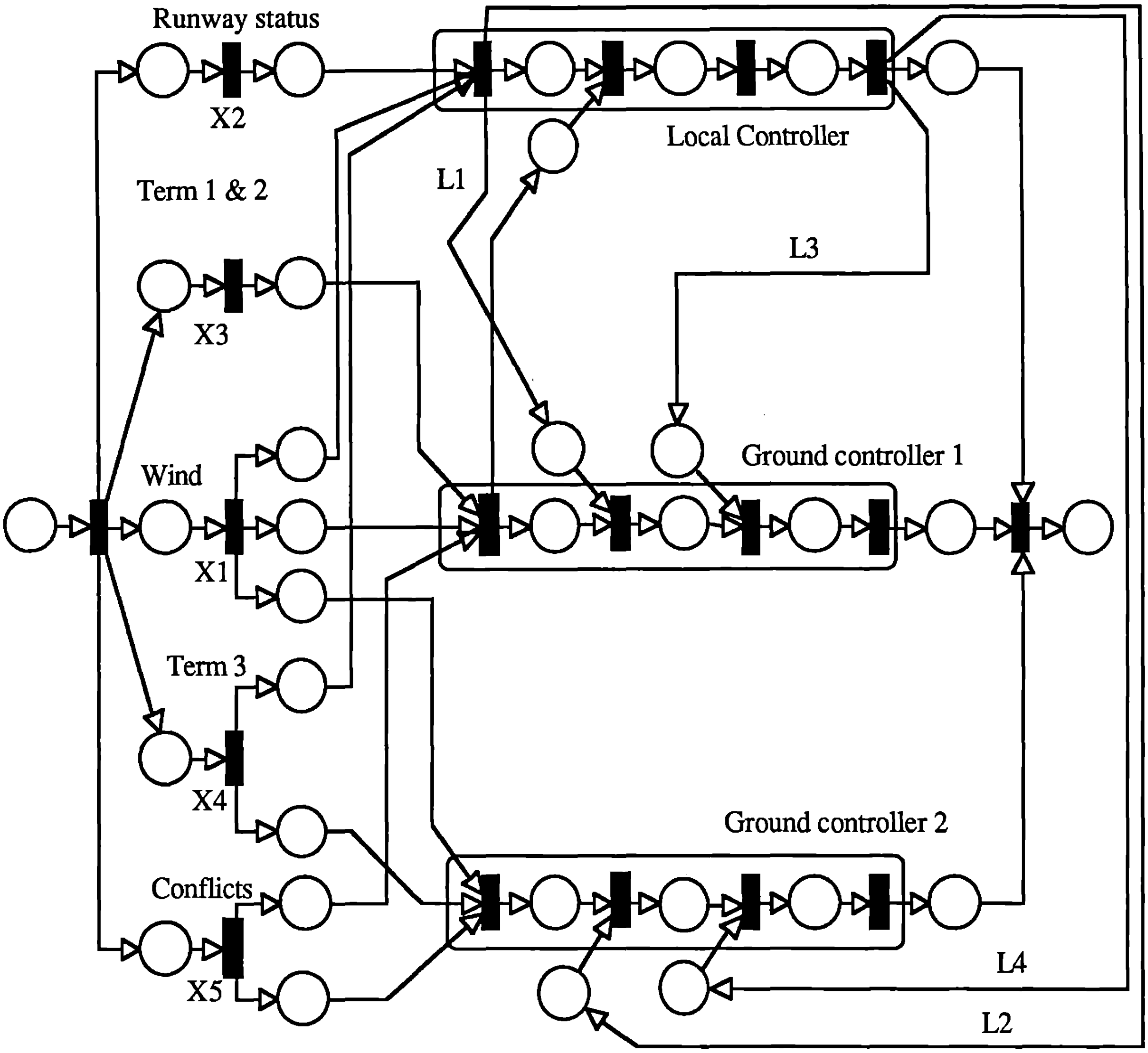


Fig. 8.25 Intermediate Structure

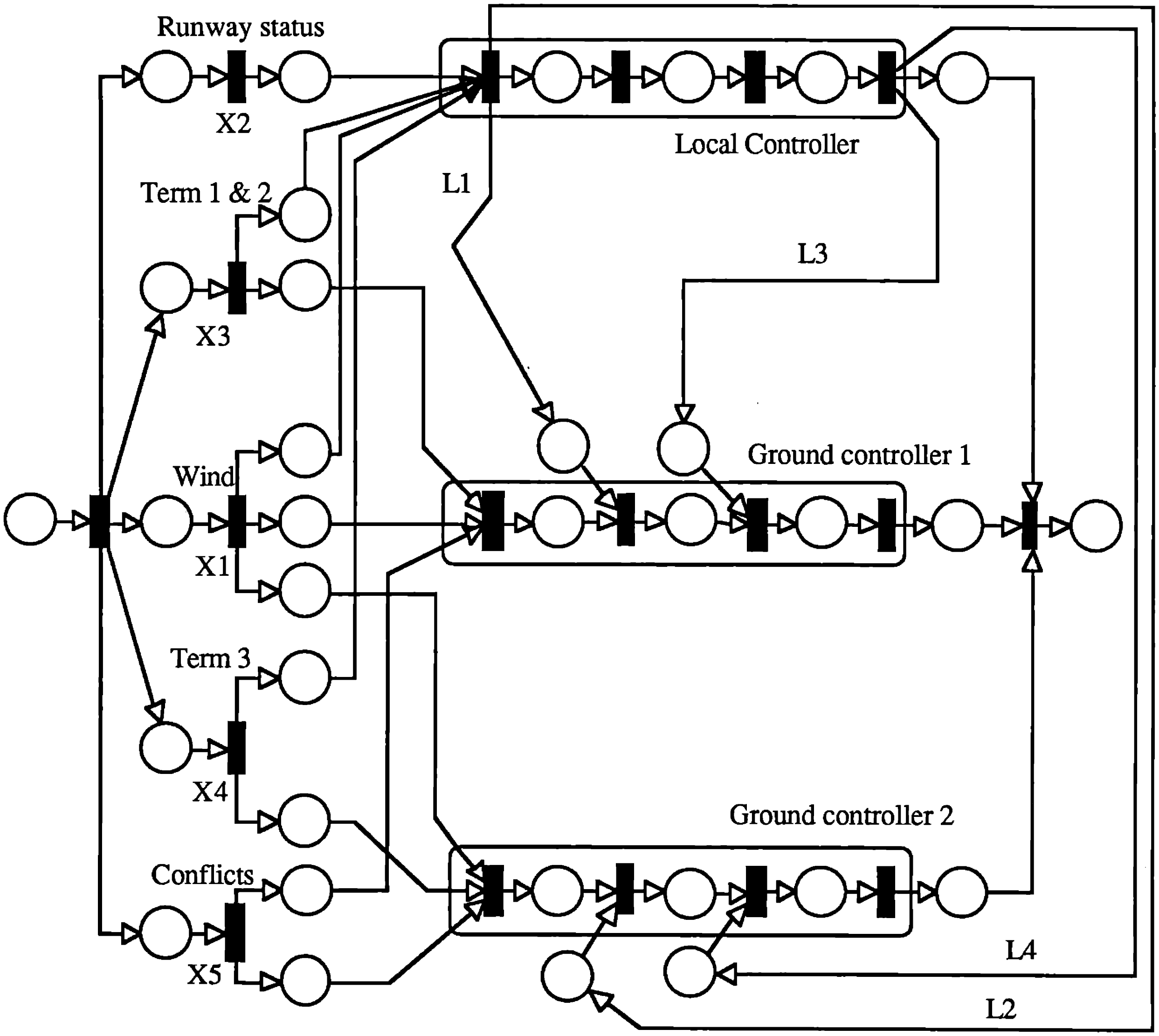


Fig. 8.26 VMINO 4

Finally, the accessible pattern of the solution depicted on Fig. 8.28 is [{X1}, {X1}, {X1, X3, X4}, {X1, X3}, {X1, X4}]. The second layer of solutions is the structures in which GC1 and GC2 always exchange their situation assessments, and

- in which Link C can be activated by any set of inputs in L(EX_1, ..., EX_6)
- in which link D can be activated by any set of inputs AC = $EX_3 \cup EX_4$ or $EX_5 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_4$ or $EX_2 \cup EX_3 \cup EX_4$ or $EX_1 \cup EX_5 \cup EX_6$ or $EX_2 \cup EX_5 \cup EX_6$.
- in which link E can be activated by any set of inputs AC = $EX_3 \cup EX_5$ or $EX_4 \cup EX_6$ or $EX_1 \cup EX_3 \cup EX_5$ or $EX_2 \cup EX_3 \cup EX_5$ or $EX_1 \cup EX_4 \cup EX_6$ or $EX_2 \cup EX_4 \cup EX_6$.

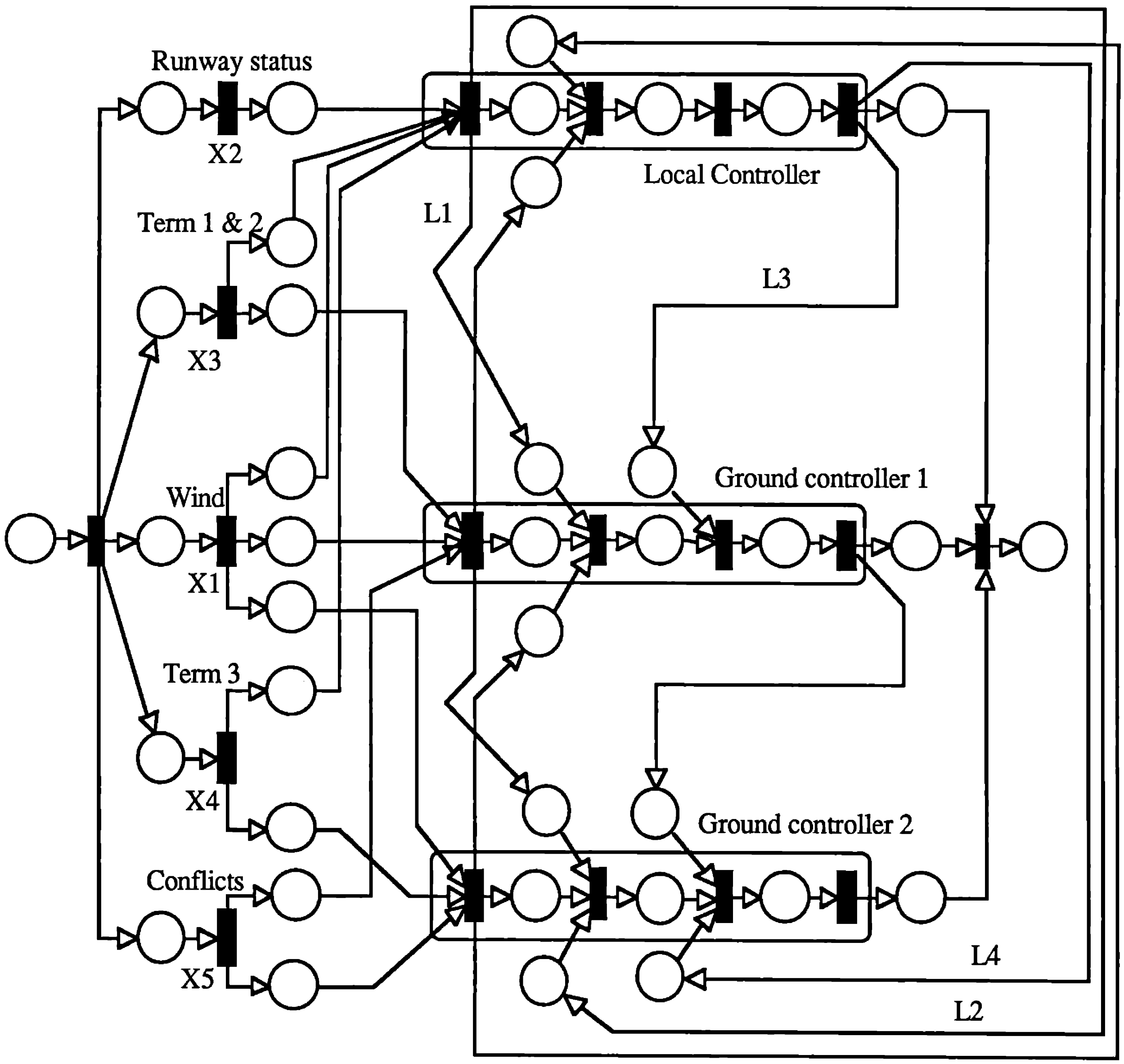


Fig. 8.27 VMAXO 4

The four subsets of results correspond to four different degrees of coordination between the roles of the ASTC system. The first subset corresponds to the case in which the roles only have access to the sources of information that they monitor because of the constraints. Local Control exclusively monitors the landings, and each Ground Controller monitors its geographic sector. Note that the pattern of interaction of the Local Controller with the Ground Controllers is the same over all solutions in subset 1. There is one and only one way to achieve a meaningful coordination between Local Control and Ground Control. On the other hand, there are many variable structures between VMINO 1 and VMAXO 1, which correspond to different patterns of interaction between Ground Controllers. The first layer of solutions corresponds to the case in which the variability between Ground Controllers is based exclusively on their common sources of information, X1 and X5. Finally, the second layer of solutions corresponds to a series of solutions in which both Ground Controllers exchange at all times their situation assessment. and in which GC 1 can issue a command or an advisory to GC 2, based on the observations of Sensors 1, 3, 4, and 5.

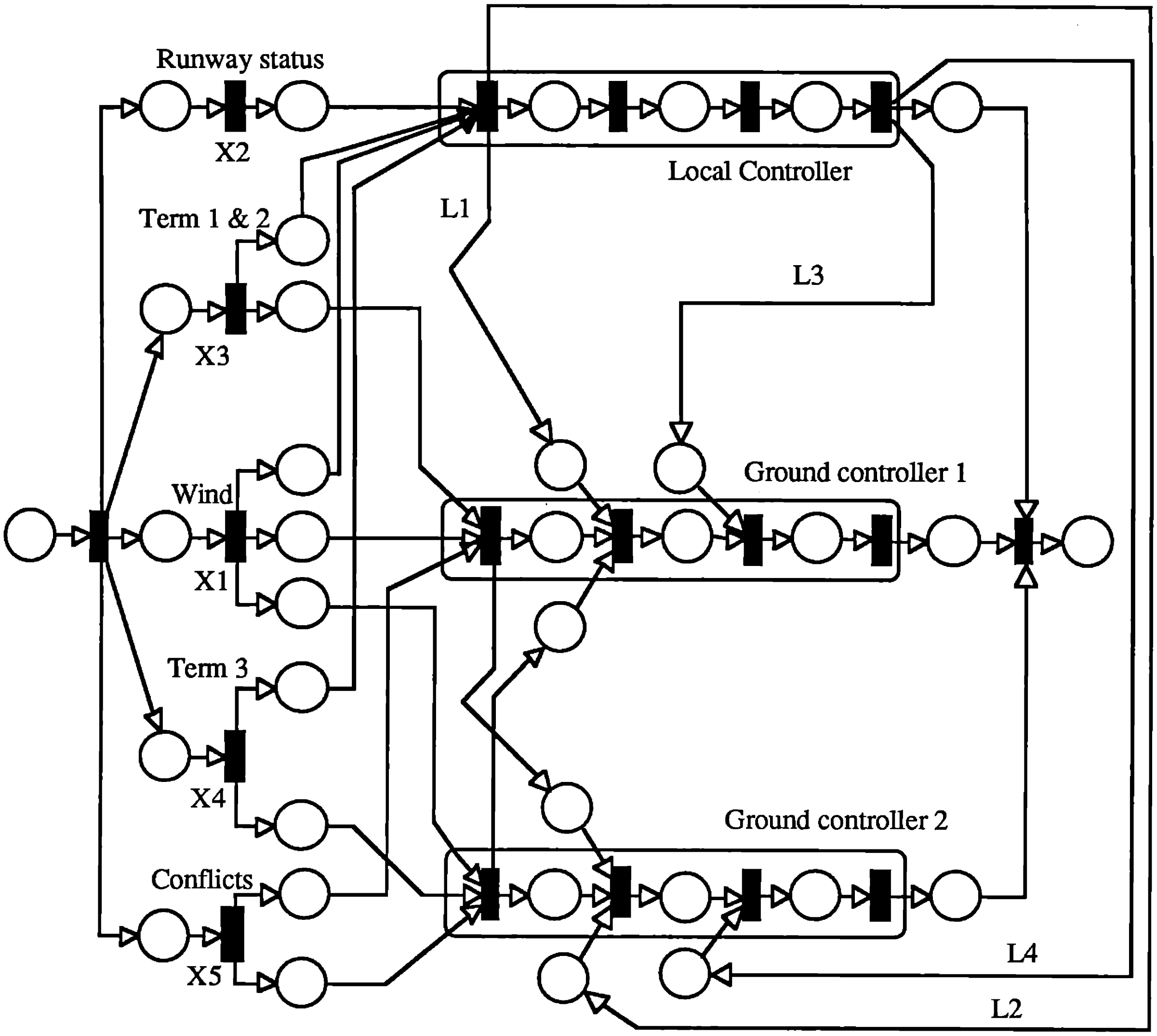


Fig. 8.28 Intermediate Structure

The other subsets present larger ranges of solutions. This is achieved by allowing roles to have access to more sources of information. Note however, that the possible pattern of interactions between Ground Controllers is identical over the subsets 1, 2, 3, and 4. The VMINOs always contain the same links between Ground Controllers, and the VMAXOs also have the same links between GCs. On the other hand, subsets 2, 3, and 4 describe the cases in which more and more varied patterns of coordination between Local Control and Ground Control can be instituted. Subset 2 indicates the case in which there is more than one way to achieve coordination between GC 1 and LC, while there is only one solution to the coordination between GC 2 and LC. Subset 3 indicates the converse case, in which there is more than one way to enforce coordination between GC 2 and LC, while there is only one solution to the coordination between GC 1 and LC. Finally, subset 4 describes the case in which both Ground Controllers can have different patterns of variable interactions with LC.

From a practical point of view, these subsets indicate a clear trade-off between access to sources of information and the number of coordination mechanisms between roles. The number of ways can be increased if the roles have access to more sources of information. On the other hand, this increase in the number of sources that are monitored can increase the workload of the controllers, and may decrease their reliability. In order to tackle these issues quantitatively, the Colored Petri Nets generated in this example can be used with the System Effectiveness Analysis methodology to assess the impact of these tradeoffs on performance. In particular, the workload associated with the configurations, as well as the timeliness and accuracy of the system can be computed.

CHAPTER IX

POLICY ANALYSIS

Because this thesis intends to contribute to the theory of distributed intelligence systems, this chapter analyzes the socioeconomic and political environment within which the need for research in distributed intelligence systems has been formulated. The interactions between the research issues and the political needs are identified. Based on those findings, a strategy for implementing the methodology of this thesis is devised.

9.1 LARGE SCALE SYSTEMS: AN AREA OF CONCERN

The federal government devotes larger and larger amounts of resources, in terms of time and money, to design, analyze, and run large-scale distributed systems. At the same time, there seems to be a growing dissatisfaction with such systems. The press regularly reports problems with the Air Traffic Control system, the large delays experienced by passengers at airports due to traffic congestion, and the safety issues, such as near misses and collisions, controllers' failures, etc. The specialized press insists that most problems are rooted in the obsolescence of the technological rationales which led in the 50s to the design of the actual system, and that the FAA failed to keep up with technological changes and the boom in traffic that came from deregulation. Furthermore, by focusing primarily on the enroute Air Traffic Control, the FAA made the Airport Surface Traffic Control the most saturated and dangerous part of the ATC system today. The working environment of airport controllers has not been adapted to the increase in traffic, causing many problems of workload management and coordination between controllers.

Similarly, Anger (1988) describes how some systems of the Department of Defense fall short of expectations because they can neither manage the complexity of their environment nor coordinate rapidly and efficiently their missions. The dramatic accident involving the USS Vincennes illustrates the difficulty of managing a distributed intelligence system, if it has to react rapidly under time and tactical pressures. The Vincennes was cruising at sea, the intelligence support was distributed throughout the Persian Gulf and could not identify rapidly the nature of the radar signature, the Commander in Chief with responsibility for the Persian Gulf was not physically present, but was thousands of miles away.

Under its mandate to address the needs of the United States, Congress is responsible for monitoring the development of systems under federal jurisdiction. Over the past twenty five years, Congress has ordered that federal systems should become more efficient, and should address new needs, under the general constraint that financial resources are scarce. To enhance the performance of the interaction between Congress, the federal agencies and the private sector, Systems Analysis has been developed as a quantitative aid to certify only those programs that will be cost effective (Anger, 1988). While performing a system analysis, developers consider the costs and benefits of the programs, explore a variety of options, and define the combination that will be most cost effective.

The federal authorities have recognized that the government should meet the increasing needs for analysis and design of distributed intelligence systems by launching comprehensive research efforts in this field where little was known. As formulated in a recommendation of the Defense Science Board (1978) for the systems of the Department of Defense, "DoD should develop a coordinated program of research and testing on command and control concepts, design and system performance, to provide the intellectual base to guide the evolution of improved command and control systems." Similarly, the FAA has supported many research efforts to devise ways to improve or change the actual Air Traffic Control system.

9.2 OPERATIONAL ISSUES

The formulation of operational needs is strongly influenced by the classical works on organization theory of Herbert Simon et al. (Simon, 1981; Crecine and Salomone, 1988), whose work relies on the assumption that the cognitive processes of human beings are constrained. Therefore, the limit on the information that can be absorbed forces people to respond only to limited aspects of their environment, and to rely on relatively simple mental strategies. In an operational context such as the ATC system or a DoD system, teams of decisionmakers are created as a solution for overcoming these individual cognitive limits. Through specialization, individuals acquire the capacity to apply relatively complex cognitive strategies to narrowly defined tasks environments. Through division of labor, substantial cognitive resources can be simultaneously brought to bear on many tasks or information sources at a time.

At first sight, the potential accomplishments of the division of labor seem to be limitless. In practice however, systems that mix human beings, computerized decision aids, and multiple communications channels must overcome a fundamental difficulty, the need for the pattern of activities carried out by individuals in various subunits of the system to fit together in a coherent fashion. Achieving such coherence defines what is termed the coordination problem, but this structuring can easily become so complex as to overwhelm the cognitive capacities of those who must carry it out. Indeed the high frequency of coordination failures suggest that such problems commonly occur. Typically, near misses originate from coordination failures between controllers. In response, the FAA imposes strict rules on ground movements and landings and takeoffs, which cause greater delays at peak hours or bad weather. Those delays put pressure on the controllers and further deteriorate their ability to coordinate their tasks optimally!

Direct supervision, mutual adjustment, standardization and planning are the four basic mechanisms for achieving coordination in a distributed intelligence system. Direct supervision is the most commonly used mechanism. It corresponds to the introduction of hierarchical layers, or supervisor-subordinate types of relationships. Mutual adjustment occurs when two or more actors agree to share resources and to confer with one another concerning decisions that affect their respective tasks. Standardization is the creation of standard operating procedures, and planning corresponds to a direct supervision done ahead of time. Unfortunately, all these strategies are very costly, and practical systems are often designed so as to limit the amount of coordination. A consequence that can be sensed in everyday life is that most systems fail to exploit the potential that would theoretically be generated by division of labor.

Another problem is that most distributed intelligence systems are designed using a functional decomposition. Functional decomposition exploits the economies of scale that are inherent in high degrees of specialization, but tends to induce heavy coordination costs in the system regardless of the means employed to achieve coordination. Therefore, functionally structured organizations tend toward inflexibility and inability to adapt to rapidly changing situations.

9.3 A RESEARCH AGENDA

While committing financial and human resources to the development of basic research on distributed intelligence systems, FAA and DoD have indicated four research problems that are most crucial to their needs: processing capability, reaction time, flexibility and coordination capability.

The issue in processing capability is that systems developers or operational people tend to formulate needs for as much cognitive, computational, and communications capacity as possible, which is not feasible given political and budgetary constraints. This body of research should thus address how timely, accurate, and complete an information should be, what decision aids are really needed to adequately perform the task at hand, and a general formulation of the tradeoffs that must be made while developing or upgrading systems.

The second area of concern is the study of reaction times, that is, the amount of time that elapses between the onset of an input to the system and the initiation of the response versus the amount of time within which a response should be initiated. This area of research is closely related to safety vs. congestion concerns for the ATC, and has a dramatic impact on the effectiveness of military systems.

The third domain of studies is flexibility, which is the capacity to adapt to rapidly changing conditions, to changes of interactions, or to courses of actions that can be followed. Indeed it is widely believed that flexible systems may perform more accurately than rigid systems, because they can adapt their pattern of interactions to the optimal configuration for each input.

Finally, the last area of research focuses on coordination capacity, the determination of the most cost-effective procedures to coordinate tasks in a system, which is, as described before, the most basic and difficult problem encountered in actual implementations.

9.4 A RESEARCH COMMUNITY

The research community can be roughly divided into four groups with partial overlaps.

- One group is interested in theories and models of distributed intelligence systems. The members of this group are trying to establish a theoretical foundation for this emerging field by relying on organization theory, but extending it to include findings from the fields of computer science, control theory and communications theory. The theoretical analyses try, in particular, to incorporate all four crucial characteristics.

- The second group studies primarily the physical components of the system. Such components very often involve state-of-the-art technology because of the huge data processing needs formulated by the users of the systems. Radars, communications equipment, communication protocols, and parallel computers are just a few examples of the key technologies that were developed in this area. This group focuses on the processing capability and the development of tools to enhance communications and coordination.
- A third group analyzes the behavioral aspects of distributed intelligence systems so as to develop decision support systems and expert systems. This is the realm of research on ergonomics and man-machine interactions. These studies focus on the parameters that affect the response time, and the constraints on flexibility due to limited human processing capabilities.
- Finally, a fourth group is interested in testing and evaluating actual or prospective systems. This includes the development of distributed architectures, and the applications of evaluation techniques (quantitative analysis, analysis by computer simulation, testbeds, etc). This fourth area tries to encompass all four research agendas.

9.5 IMPLEMENTATION CHANNELS

The federal agencies expect that the research efforts will influence four different areas that are under administrative jurisdiction: planning and budgeting; requirement generation; systems development; and training.

- Planning and budgeting.

  The science boards of FAA and DoD estimate that the results of the research will include critical new aspects into long range planning and architectural developments. The group that develops theoretical foundations and the group that is more concerned with performance measurement can help evaluate the relative contributions of different parameters to the effectiveness of actual and prospective systems. They can highlight the areas in which actual achievements are adequate, should be improved, or stand at the limit of technological capacities. Basic research about distributed intelligence systems helps to perform cost analyses, so that the cost-effectiveness tradeoffs imposed by the policymakers can be as rational as possible.

- Requirements generation.
  Many of the actual requirements procedures are formulated based on anecdotal evidence. This evidence is generally gathered during exercises, simulation, or from an event of unprecedented nature such as an accident. For most systems, this means that modernization or development programs do not incorporate all the knowledge that has been gained from actual systems. There is thus a need to generate credible lessons from the present experience which can be used to formulate new and improved operational requirements.
- Development and acquisition.
  The scientific community and developers and acquisition people have difficulty communicating despite the fact that a collaborative effort could bring significant improvements in the development and acquisition processes. The Science Advisory Panels of both FAA and DoD expect that the research programs and their educational counterparts (Seminars, books, etc) will give operational people the opportunity to acquire major sets of new skills, and that future upgradings of the systems will be oriented more toward general problem solving than toward incremental fixing of the latest detected deficiencies. Test bed experimentation or the development of simulators (ATC simulators) where scientists and engineers can exchange ideas, test concepts, and get immediate feedback from people with field experience seem to offer the best tools to that respect.
- Training.
  Training is no longer the exclusive area of people who run the systems. The science boards believe that simulators and testbeds can here again contribute significantly to many improvements, both on the theoretical front and on the practical front, by testing in a safe environment new ideas and procedures, and by significantly reducing the cost of training new controllers.

9.6 AN INTERIM ASSESSMENT

Most of the research programs on distributed intelligence systems have been initiated in the last decade. Ten years after the emergence of this field, federal authorities judge very differently the progress that has been achieved on the basic research agendas. Some of the issues discussed in the sequel have been raised by the Science Advisory Panel of DoD (1987).

- The panel noted first that some progress has been made in formulating an embryonic theory for distributed intelligence systems. However the panel also noticed that theoretically oriented groups were the least involved in practical applications, and that a lot of effort should be expended in educating the operational community to the first significant results. In its final word, the panel insisted that this area of research should be expanded , because theorists are starting to understand the important issues, and should be able to provide an acceptable framework for assessing the effects of the various systems' parameters on performance.
- The second segment of the research community has generated the most interactions and progress. Agencies such as FAA, FCC, DARPA, and NOSC have sponsored programs that have been translated into developments and acquisitions. To cite just a few, the FAA has sponsored much research on radars and instrument landing capabilities; DARPA and the FCC have co-sponsored studies and development of advanced packet switching networks; and in addition DARPA financed many research efforts on parallel computing. As a whole, the Science Advisory Panel estimated that this area of research is much too successful, because advances on technological issues effectively make the coordination problem even more complicated.
- The behavioral groups have had isolated successes. In particular, in the area of decision support systems, more is known about the design of workstations and about the use of expert decision aids to enhance the effectiveness of systems. On the other hand, the Science Panel found that the overall research is much too focused on components, or on improvements of a particular application than on formulating concepts that could be applied universally.
- Finally, the fourth part of the research community has had some success in aiding in the planning and budgeting areas, as well as in the area of development specifications. Their most important contribution was to educate and convince people that much gain in effectiveness can be obtained by optimizing over the structures rather than by focusing primarily on efficiency improvements at the level of physical components.

9.7 CONTRIBUTION

This thesis falls within the activities of the first segment of the research community, that is the development of theoretical models to understand distributed intelligence systems. For the first time in the literature, the coordination problem for variable structure systems is addressed. This thesis extends a model that had been successfully developed for fixed structure systems, the four stage model of a boundedly rational decisionmaker. This model incorporates indeed three out of four coordination mechanisms presented in 9.2: direct supervision, mutual adjustment, and standardization. Direct supervision and mutual adjustment are modeled by the allowable interactions between roles. Standardization is modeled by the fact that one can create a Colored Petri Net model of the system, i.e., that the activities of the roles are globally coordinated at the system level, and that this coordination is deterministic.

Two major illustrative examples have been developed. They are the coordination of tasks in a submarine, and the coordination of air traffic controllers at Logan Airport. They illustrate the usefulness of the methodology in at least two domains. First, these examples show that the framework is able to address the coordination issues for a practical problem. The exposition of Chapter VIII is flexible and practical enough to easily incorporate input from non technically oriented people. Second, the examples prove that significant insights into the variable structures that satisfy some well defined constraints can be obtained without a computationally expensive, and practically infeasible, exhaustive enumeration. On the other hand, a drawback is that any design process must involve at least one person with a good knowledge of the model, who can translate the various constraints into mathematical terms. Furthermore, the results of the methodology are more easily understandable if the reader has had some knowledge of Petri Nets.

This thesis is closely related to work in the fourth area of research, on the evaluation of performance. The performance of any variable structure can be assessed by the extension of the System's Effectiveness Analysis as described in Monguillet (1988), using a Colored Petri Net model of the structure. This thesis can thus be related to other tools developed at MIT/LIDS, which answer the needs of a Research & Development team that has to study all feasible configurations of a system, given the constraints of the mission and of technology. This thesis describes how to generate all variable structures that satisfy the constraints of the design. The other tools evaluate the performance of the structures. The team in charge of research and development can thus answer unambiguously the concerns of policymakers and contractors. The team can demonstrate that a system can be implemented or upgraded in the presence of technological and financial constraints, and will yield substantial advantages. The political authorities, and expert panels that are consulted, can base their decision on such quantitative measures instead of making an educated guess. The predictions furnish criteria by which the actual performance of the system will be assessed by policymakers, and by which decisions about the outcome of the project will be made.

One sees that this research effort addresses precisely the needs of the research contractors, who want theoretical tools to help them make decisions. This thesis makes a contribution that can be integrated into a complete set of analytical tools to address the effect of the structure of the system on the performance of the system. This thesis tried to narrow the gap between the organizational structures that can be modeled and analyzed, and the real, complex organizations that exist and function.

9.8 RECOMMENDATIONS

The policy analysis suggests a natural strategy for implementing the methodology of this thesis. The constraints are that this research has been done on the theoretical front, which fails to generate enough communication with the operational community. The goal of these recommendations is to devise a strategy which will prove to non-theory oriented people that they can acquire new skills from this thesis.

- First, this thesis should generate papers to convince the theoretical community of the merits of this model, i.e., that this model is a first and promising step toward an understanding of variable structures. It will be stressed that this model incorporates many concepts from organization theory à la Simon into a mathematical framework, and offers brand new ways to tackle coordination problems. It is hoped in particular that some of the ideas introduced in this thesis, the use of Colored Petri Nets to model variable distributed intelligence systems, and the four stage model of the variably interacting role, will be adopted by other studies. If other groups develop the insights of this thesis, these studies will generate a critical mass of knowledge that will be more and more appropriate for operational needs.
- Second, the following two projects will aim at translating the results into a useful form for the people who interact in both the research and the operational worlds.

  A) The design methodology advocated in this thesis will be implemented on a personal computer and tied to the CAESAR package, which is under development at MIT/LIDS. This package is a Macintosh II-based prototype of a workstation that provides Computer Assisted Design of distributed intelligence systems. The actual package contains modules to evaluate performance of system designs, and a module that solves the design problem for fixed structures. The latter module will be updated to account for the results of this thesis, and will bring into CAESAR the possibility to generate and analyze variable structure systems. This workstation is regularly demonstrated to people who are involved in planning, requirements specification, and development, and serves as a demonstration forum of the work done at MIT/LIDS. The new platform will be particularly attractive to people who are developing or upgrading new systems, as it will provide a complete set of tools to investigate rapidly the set of designs that can be chosen, and their performance. The hope is that in this way, the dialogue between basic research and developers will be enhanced.

B) Papers based on the examples of this thesis will be submitted at the appropriate applied research forums. The choice of illustrative topics that are the subject of much discussion in their operational context should convince the respective communities that this model can both address currently important issues, and open ways to overcome the problems encountered during the course of the operations.

CHAPTER X

CONCLUSIONS AND DIRECTIONS FOR FURTHER RESEARCH

10.1 CONCLUSIONS

This thesis described a methodology to model and generate variable structure distributed intelligence systems. The first step has been carried out in Chapter IV, where the objects from which the system may be built are described. In this model, a decisionmaking process is described by a four stage process, and internal interactions between the decisionmaking processes are identified. Chapter IV restricts the scope of this thesis to one class of variable structure systems, those that are functionally deterministic and temporally consistent. It is shown that these variable structure systems can be represented in matrix form. In Chapter V, the matrix description of a variable structure system is translated into the language of Colored Petri Net theory. The processes of folding and unfolding allow one to integrate the results obtained on fixed structure systems into a theory of variable structure systems, and to formulate the problem of generating variable structure distributed intelligence systems using the language of Colored Petri Nets.

In Chapter VI, the class of structures that must be considered given a design problem is described. Ten structural constraints have been imposed to define the set of variable structures that make physical sense. The designer of a system can also introduce the constraints that correspond to his knowledge of the specific application. Chapter VII analyzes the properties of the different constraints using the notion of convexity. It is shown, in particular, that a class of solutions can be decomposed into several distinct subsets. Each subset of solutions has minimal and maximal elements. The solutions between a minimal element and a maximal element can be sorted into layers of solutions with the same effective pattern, and each layer is a lattice.

In Chapter VIII, the new results are illustrated through two examples. It appears, in these examples, that the major contribution of the model is a quantitative formulation of the coordination problem. From the theoretical and practical points of view, one major limitation of this model developed in this thesis is that it characterizes only one subclass of solutions. The user-defined colored constraints partition the set of inputs into k subsets, EX_1,..., EX_k. The subclass of solutions corresponds to the Well Defined Variable Structures that assign one and only one fixed data flow structure to each EX_i. Another difficulty is that the properties of this set of solutions are less obvious than the properties of the set of fixed structures.

Finally, Chapter IX assesses the contribution of this thesis within the body of research on distributed intelligence systems. The chapter describes an implementation strategy for this thesis, based on an analysis of the rationales that led the federal government to initiate research programs on distributed intelligence systems.

10.2 DIRECTIONS FOR FURTHER RESEARCH

Research can be pursued in many directions to improve and extend the methodology developed in this thesis.

- The first natural extension would be to achieve the characterization of the set of variable structures. This work should describe the set of solutions that have an arbitrary number of fixed structures in their support, and should provide an algorithm to generate them. It is suspected that convexity is here again the right tool to be applied.
- The present model should include new protocols of interactions between decisionmaking processes. The application of the methodology to several real-scale examples has shown that the coordination problem has too few solutions because the model permits at most three interactional links between two roles. This paucity of interactions prevents the roles from having elaborate protocols of interaction at several stages of their processing. For example, a case in which one role A would ask for more information from another role B, if A has already received a message from B, cannot be modeled.

  One possibility is to transform the four stage model into a five stage model, in which the additional stage is equivalent to the middle processing of Andreadakis and Levis (1988). New interactions should be defined and would yield more realistic structures.

  The present model should also be extended to incorporate classes of variable structures that are not temporally consistent, and process observations induced by events that do not have the same temporal origin.
- It would be particularly interesting to start an investigation of the relationships between

the performance of a system, and properties of a Colored Petri Net model of the structure. This can be achieved for example by introducing Stochastic Timed Colored Petri Nets, if one is interested in the dynamical properties of a structure. For that purpose, the new release of a software product that allows the design and simulation of Colored Petri Nets seems to offer a promising tool.

On the same vein, there is a need to have an extensive investigation of the impact of the resource structure. One should look at the properties that are characteristic of the resource structure, and at the protocols that control the assignment of physical resources to the functional entities.

- Finally, there is an urgent need to have new analytical tools to investigate the properties of Colored Petri Nets. In particular, the concept of S-invariants in Colored Petri Net should be clarified, and one should be able to write an algorithm to compute colored S-invariants, if they can provide meaningful insights into variable structures.

REFERENCES

Andreadakis, S. K., and Levis A.H. (1988) "Synthesis of Distributed Command and Control for the Outer Air Battle" *Proc.1988 Symposium on Command and Control Research.* Science Applications International Corporation, McLean, Virginia.

Anger, T.E. Editor. (1988) *Analysis and National Security Policy.* Center for Naval Analyses, Alexandria, Virginia.

Bertsekas, D.P., and Tsitsiklis, J.N. (1988). *Parallel and Distributed Computation.* Prentice-Hall, New-York.

Birkhoff, G. (1948). *Lattice Theory.* American Mathematical Society, Providence, RI.

Boettcher, K.L. and Levis, A. H. (1982). "Modeling the Interacting Decisionmaker with Bounded Rationality." *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-12,No.3.

Bouthonnier, V. and Levis, A. H. (1984). " Effectiveness Analysis for C^3 Systems. " *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-14.

Brams, G.W. (1983). *Réseaux de Petri: Théorie et Pratique Tomes 1 et 2.* Masson, Paris.

Carré, B. (1979). *Graphs and Networks.* Oxford University Press, New York.

Clancy, T. (1984). *The Hunt for Red October.* Berkley Books, New York.

Cothier, P.H., and Levis, A. H. (1986). " Timeliness and Measures of Effectiveness in Command and Control." *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-16,No.6.

Crecine, J.P., and Salomone, M.D. (1988). "Organization theory and C^3". *Proc.1988 Symposium on Command and Control Research.* Science Applications International Corporation, McLean, Virginia.

Defense Science Board (1978). *Report of the Defense Science Board Task Force on Command and Control Systems Management.* Office of the Undersecretary of Defense Research and Engineering.

Defense Science Board (1987). *Report of the Defense Science Board Task Force on Command and Control Systems Management.* Office of the Undersecretary of Defense Research and Engineering.

Dersin, P., and Levis, A. H. (1981) "Large Scale System Effectiveness Analysis." LIDS-FR-1072, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Federal Aviation Administration. (1979). *Airport Surface Traffic Control System Requirements Analysis.* Report N0. FAA-EM-79-5. Office of Systems Engineering Management. Washington D.C.

Genrich, H. and Lautenbach, K. (1981) "Systems Modeling with High-Level Petri Nets." *Theoretical Computer Science*, No.13, pp.109-136.

Genrich, H. (1987). " Predicate Transition Nets " in *Advanced Course on Petri Nets 1986*, Lecture Notes in Computer Science, Springer Verlag, Berlin, Germany.

Grätzer, G. (1971). *Lattice theory, First Concepts and Distributive Lattices.* W.H. Freeman and Company, San Francisco.

Grevet, J.-L, and Levis, A. H. (1988). "Coordination in Organizations with Decision Support Systems". *Proc.1988 Symposium on Command and Control Research.* Science Applications International Corporation, McLean, Virginia.

Hillion, H.P. (1986) "Performance Evaluation of Decisionmaking Organizations using Timed Petri Nets." LIDS-TH-1590, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Hillion, H.P. and Levis, A.H. (1987). "Timed Event-Graph and Performance Evaluation of Systems." *Proc. 8th European Workshop on Applications and Theory of Petri Nets.* Zaragoza, Spain.

Hommel, G. (1985) *Distributed Systems. Methods and tools for specification*, Springer Verlag, Berlin, Germany.

Jensen, K. (1987). " Colored Petri Nets " in *Advanced Course on Petri Nets 1986*, Lecture Notes in Computer Science, Springer Verlag, Berlin, Germany.

U.S. Joint Chiefs of Staff (1986). *Dictionary of Military and Associated Terms.* Publications 1, Washington D.C.

Levis, A.H. (1984). "Information Processing and Decision-Making Organizations: a Mathematical Description." *Large Scale Systems*, No. 7, pp. 151-163.

Levis, A. H. (1988) ."Human Organizations as Distributed Intelligence Systems." *Proc. IFAC Symposium on Distributed Intelligence Systems.* Pergamon Press, Oxford, England.

Memmi, G. and Roucairol, G. (1980). "Linear Algebra in Net Theory" in *Net Theory and Applications, Proceedings of the Advanced Course on General Net Theory of Processes and Systems, Hamburg 1979.* W.Brauer Ed. Springer-Verlag, Berlin, Germany.

Minsky, M. (1986). *The Society of Mind.* Simon and Schuster, New York.

Monguillet, J.-M., and Levis A. H. (1988). " Modeling and Evaluation of Variable Structure Command and Control Organizations." *Proc.1988 Symposium on Command and Control Research.* Science Applications International Corporation, McLean, Virginia.

Perdu, D. (1988) "Evaluation of Expert Systems in Decisionmaking Organizations." in *Science of Command and Control: Coping with Uncertainty. Vol.2* AFCEA International Press, Washington D.C.

Peterson, J. L. (1981). *Petri Net theory and the Modeling of Systems*, Prentice Hall, Englewood Cliffs, NJ.

Reisig, W. (1985). *Petri Nets, An Introduction.* Springer Verlag, Berlin, Germany.

Remy, P.(1986). "On the Generation Of Organizational Architectures Using Petri Nets." LIDS-TH-1630, Laboratory for Information and Decision Systems, MIT, Cambridge, MA

Remy, P. and Levis, A. H. (1988). " On the Generation Of Organizational Architectures Using Petri Nets. " in in *Advances in Petri Nets 1988*, Lecture Notes in Computer Science, G.Rozenberg Ed. Springer Verlag, Berlin, Germany

Simon, H.A. (1981). *The Sciences of the Artificial*, The MIT Press, Cambridge, MA.

Stabile, D.A. and Levis, A.H. (1984) "The Design of Information Structures: Basic Allocations Strategies for Organizations." in *Large Scale Systems*, Vol.6, No 2, North-Holland.

Valraud, F. (1989). "On the Evaluation of Functionality in Distributed Systems", Master's Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA.

APPENDIX A

COMPUTATION OF SOLUTIONS IN SUBMARINE EXAMPLE

The procedure for determining the solutions to the design problem for variable structures consists of three steps. In the case of the submarine example presented in section 8.1, the steps take the following form.

Step A: The colored constraints R_C specify the variability of some links. Three variable links have been specified in 8.1, the links from the RS stages to the effectors. The effective alphabets of the partition of the links are X1, X2. The first step is to determine the convex subset of all fixed structures such that X1, X2, and X3 are accessible at the RS stages of the three roles. The output of this computation in each W^i, i = 1, 2, 3 is a convex subset of W^i. In order to avoid unnecessary and lengthy listings, the minimal and maximal elements in W^1 only are represented in Figures A.1, A.2, and A.3.

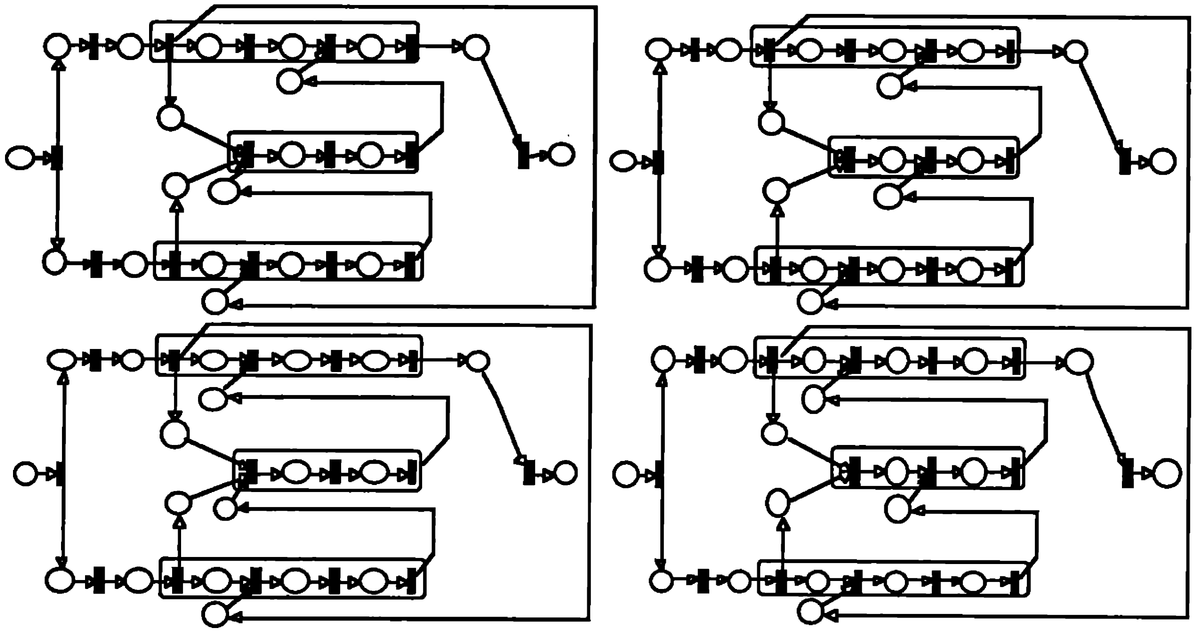


Fig. A.1 Minimal Elements in W^1

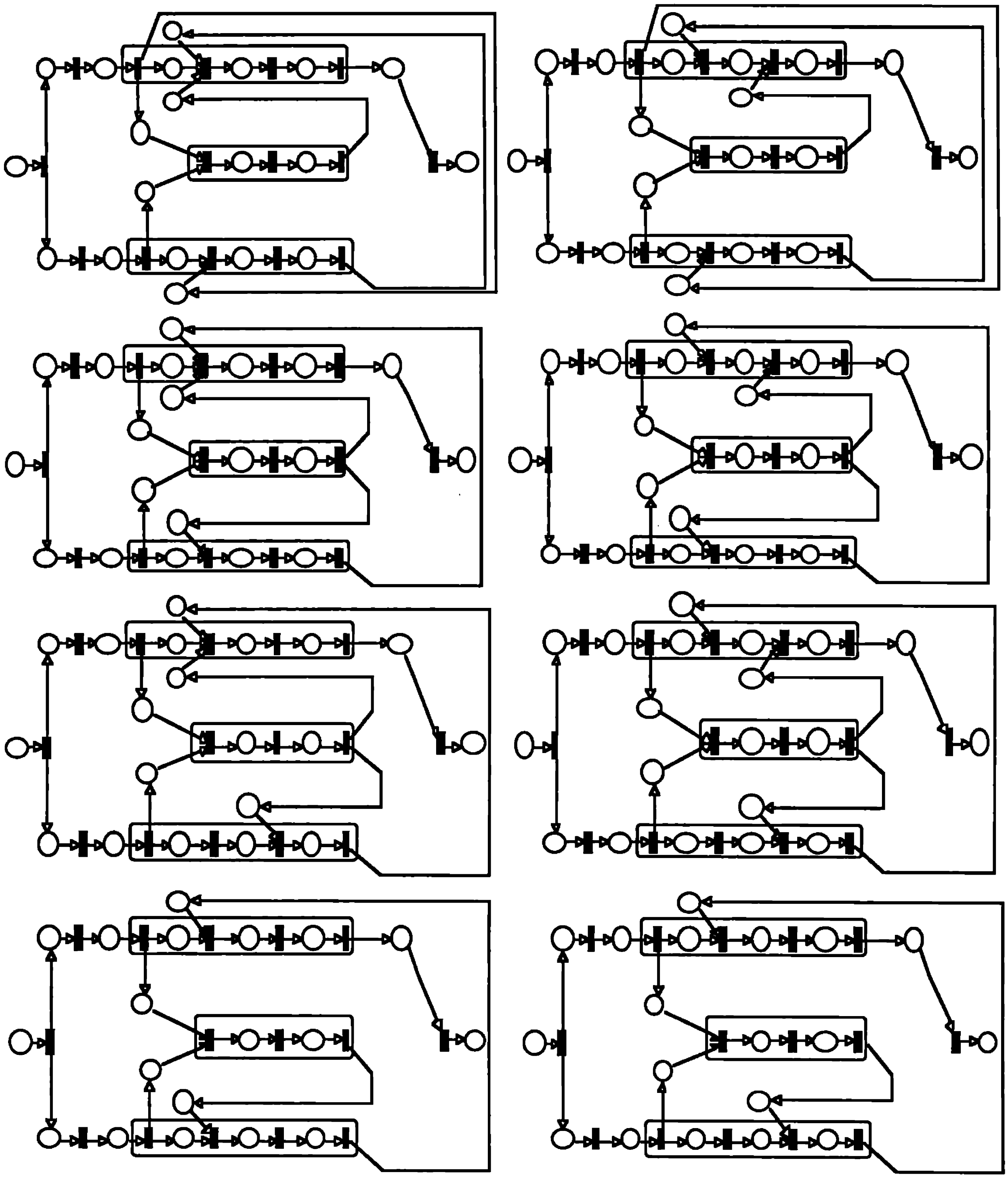


Fig. A.2 Minimal Elements of W^1 continued

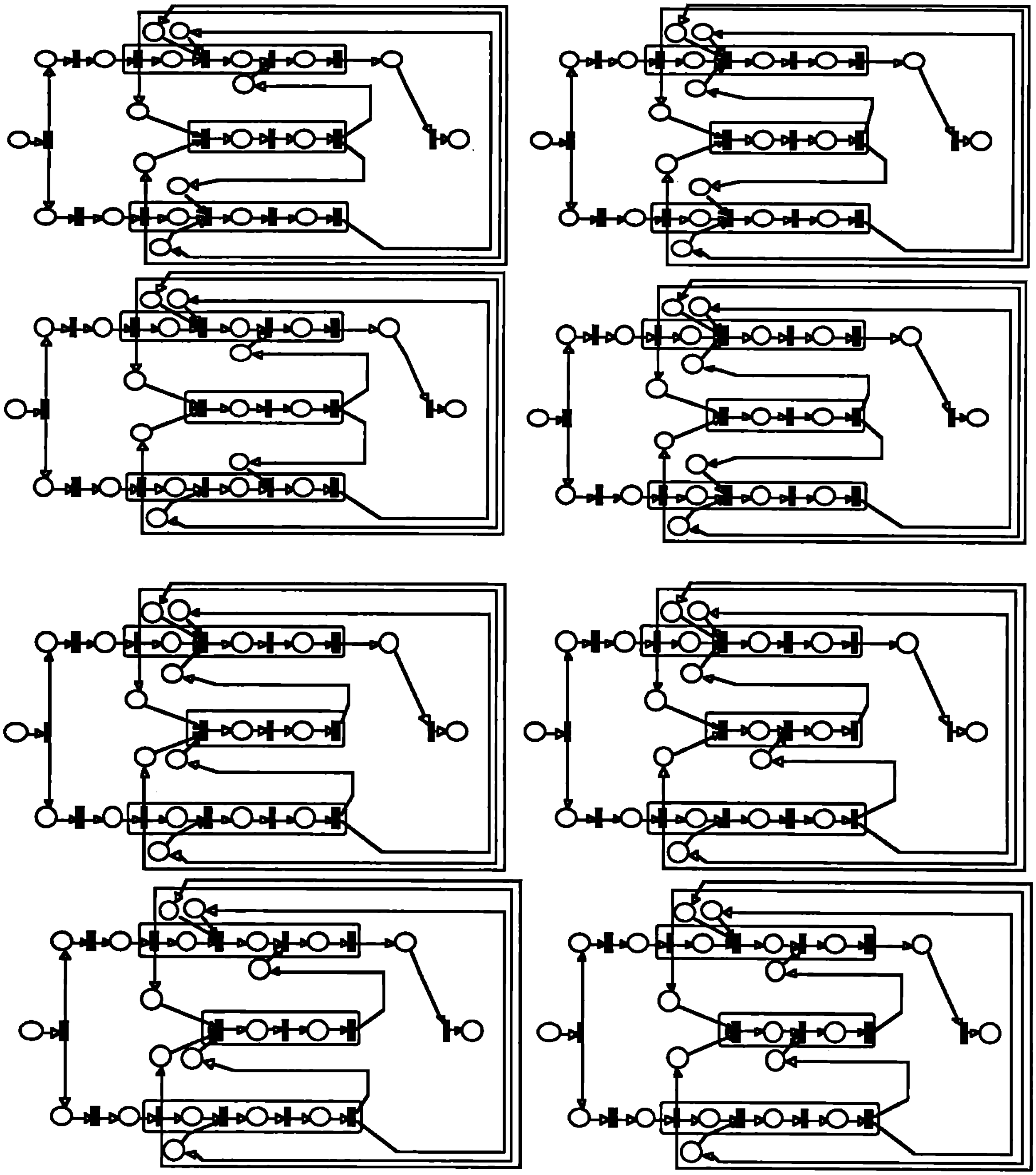


Fig. A.3 Maximal Elements in W^1

Note that the minimal elements obtained after Step A have been obtained by moving up the lattice from the minimal elements that are represented in Figure 8.5, whereas the maximal elements that are the outputs of Step A are exactly the maximal elements in Figure 8.4.

Step B:

At this step, one checks if there exists a minimal solution such that

- EX_1 is assigned to a minimal element Σ^1 in W^1.
- EX_2 is assigned to a minimal element Σ^2 in W^2.
- EX_3 is assigned to a minimal element Σ^3 in W^3.

One checks thus if there exists a VMINO which corresponds to a folding of one minimal fixed structure in W^1, a minimal WDFS in W^2, and a WDFS in W^3 that have been characterized after Step A. If one does a systematic survey in this example, no candidate VMINOs can be found. Let us illustrate the reason for it with one example.

Suppose that EX_1 is assigned to the WDFS Σ^1 depicted in Figure A.4, EX_2 is assigned to the WDFS Σ^2 of Figure A.5, and EX^3 is assigned to the WDFS Σ^3 represented in Fig. A.6.

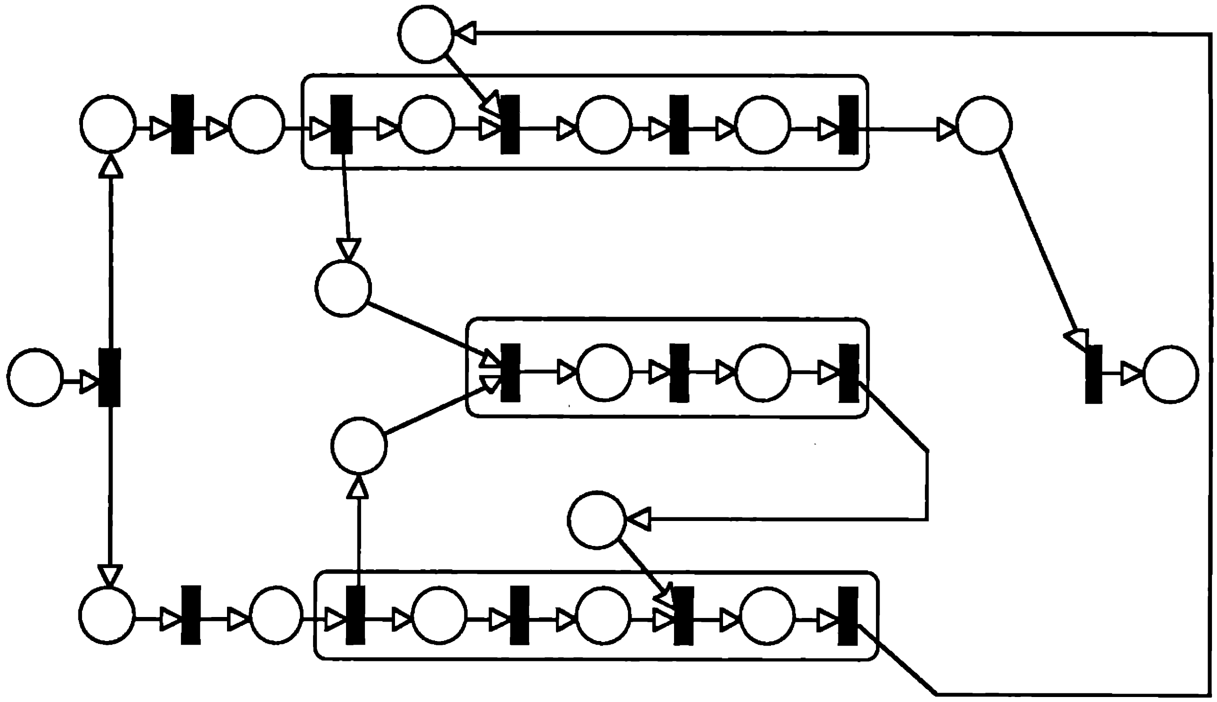


Fig. A.4 WDFS Σ^1 in W^1

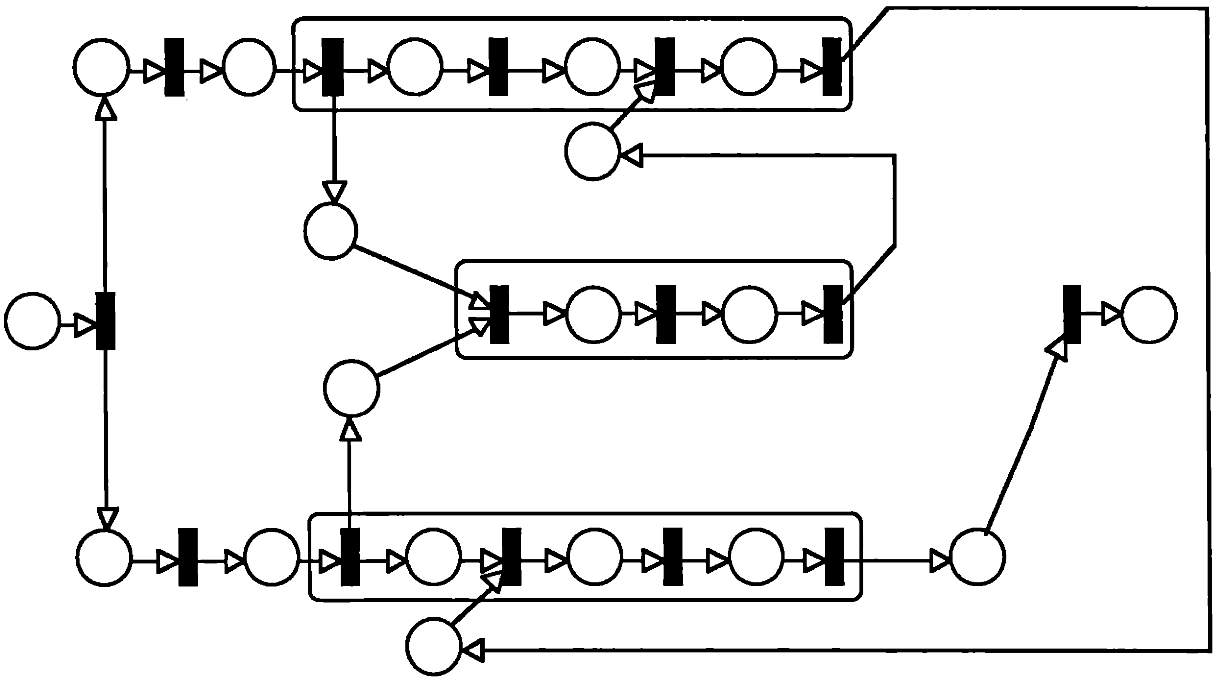


Fig. A.5 WDFS Σ^2 in W^2

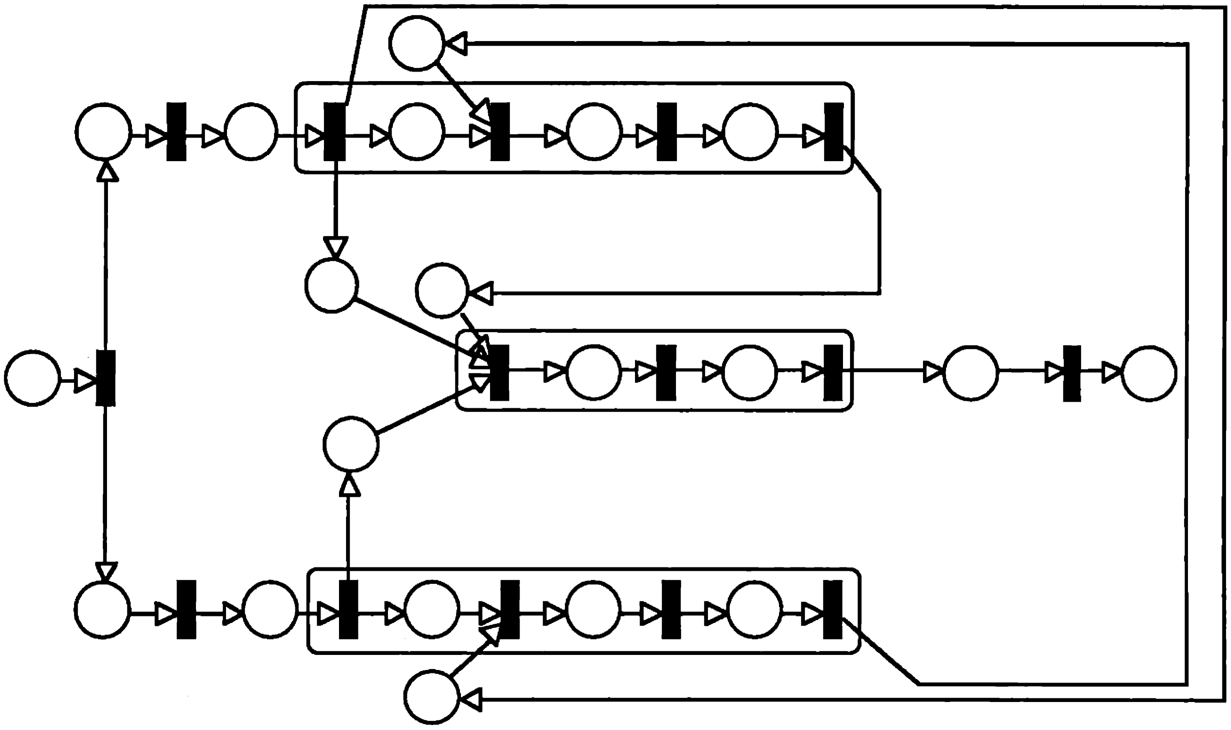


Fig. A.6 WDFS Σ^3 in W^3

The WDVS that corresponds to the folding of the three WDFS Σ^1, Σ^2, and Σ^3 does not fulfill constraint R10. Consider for example the link from the RS stage of ASUW (the lower role) to the IF stage of ASW (the upper role). This fixed link is present in Σ^1 and Σ^3. In the variable structure, the link from the RS stage of ASUW to the IF stage of ASW is thus activated by AC = $EX_1 \cup EX_3$. The effective alphabets of the partition of X into AC = $EX_1 \cup EX_3$ and DC = EX_2 are X1 and X2. One sees however that X2 is not an accessible alphabet at the IF stage of ASW, because ASW monitors only the outputs of Sensor 1 at its SA stage, the internal stage that precedes IF.

Similarly, one checks at Step B, if a VMAXO can be constructed by considering a WDVS which is such that

- EX_1 is assigned to a maximal element Σ^1 in W^1.
- EX_2 is assigned to a maximal element Σ^2 in W^2.
- EX_3 is assigned to a maximal element Σ^3 in W^3.

If one does a systematic survey in this example, no candidate VMAXOS are found at this step. Here again, let us present only one example. Figures A.7, A.8, and A.9 represent three WDFS that have been assigned to the elementary sets of inputs, EX_1, EX_2, and EX_3.

The WDVS that corresponds to the folding of the three WDFS Σ'^1, Σ'^2, and Σ'^3 does not fulfill constraint R10. Consider for example the link from the RS stage of ASUW (the lower role) to the IF stage of ASW (the upper role). This fixed link is present in Σ'^1 and Σ'^3. In the variable structure, the link from the RS stage of ASUW to the IF stage of ASW is thus activated by AC = $EX_1 \cup EX_3$. The effective alphabets of the partition of X into AC = $EX_1 \cup EX_3$ and DC = EX_2 are X1 and X2. One sees, however, that X2 is not an accessible alphabet at the IF stage of ASW, because ASW monitors only the outputs of Sensor 1 at its SA stage, the internal stage that precedes IF.

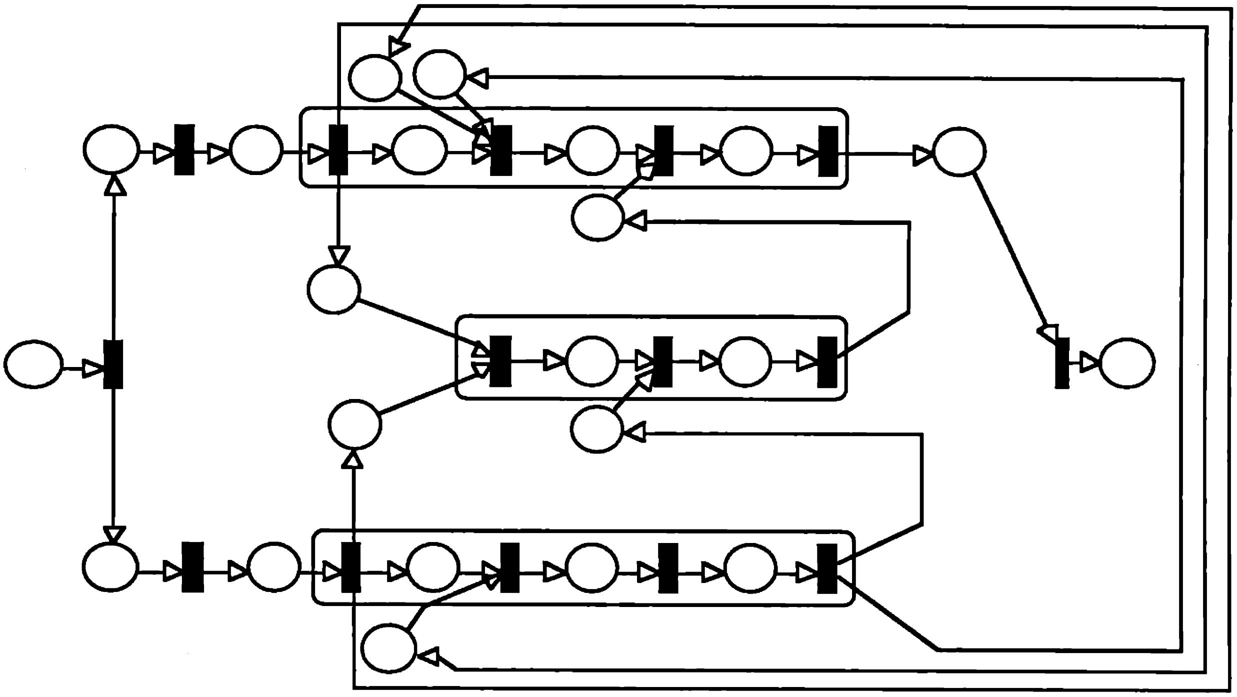


Fig. A.7 WDFS Σ'^1 in W^1

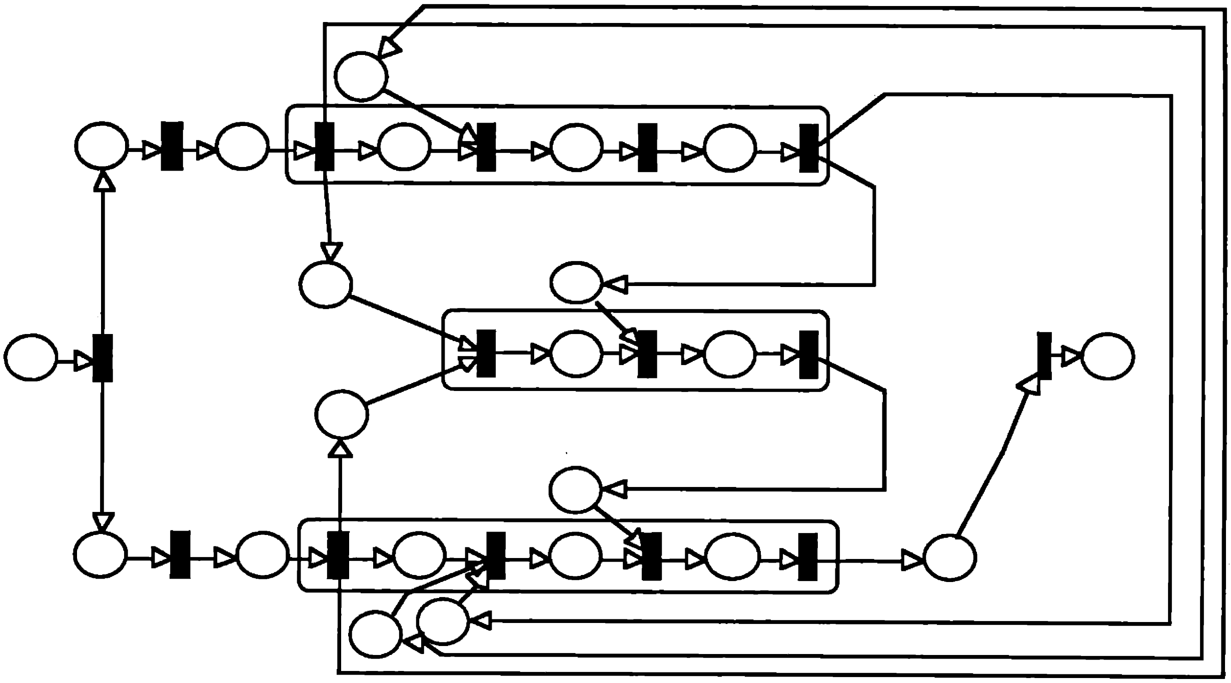


Fig. A.8 WDFS Σ'^2 in W^2

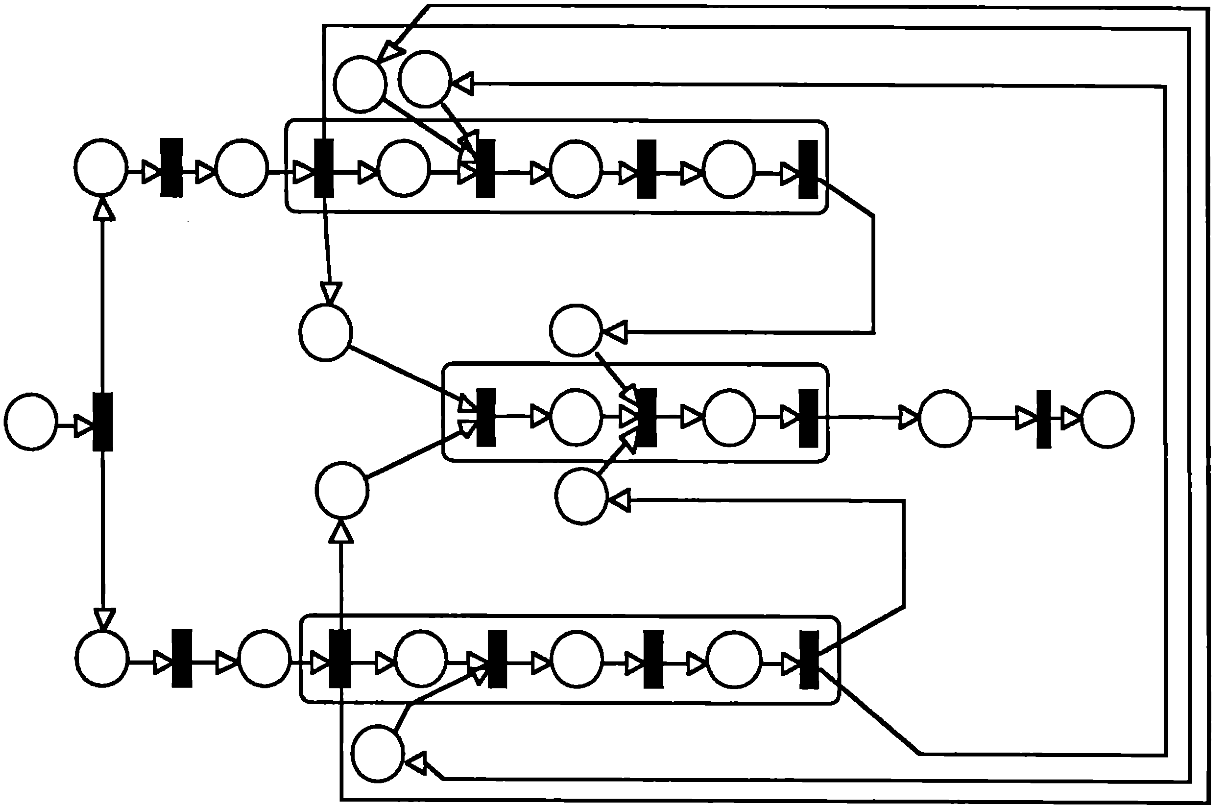


Fig. A.9 WDFS Σ'^3 in W^3

Step C

If no candidate VMINOs or VMAXOs have been obtained at Step B, then the search is continued, by moving up in from the minimal elements of Step B, to investigate new candidate VMINOs, and down from the maximal elements from Step B, to investigate new candidate VMAXOs. This search can be done optimally, because Chapter VII characterizes the building blocks that must be added to the previous candidate VMINOs, and that must be removed from the previous candidate VMAXOs, to obtain the new candidate structures. This building path is made from a simple information flow path and a elementary set of inputs attached to it.

If one applies this technique to the submarine example, then the computation yields one and only one solution, which is depicted in Figure 8.10.

APPENDIX B

NOMENCLATURE

Sets

X: The set of inputs to the system. X is a cross product of the output alphabets of the sensors:

$$X = X_1 \times X_2 \times ... \times X_N$$

where N is the number of sensors.

W: The set of Well Defined Fixed Structures. Each WDFS can be represented:

- In matrix form.
- By a unique Ordinary Petri Net.

V: The set of Well Defined Variable Structures. Each AVS can be represented:

- In matrix form.
- As a mapping from X to W. The range of the mapping is the support of the WDVS.
- By a unique Colored Petri Net.

AW: The subset of W that contains the WDFSs which satisfy Constraints R1, R2, R3, R4, R5. These WDFSs are called the Admissible Fixed Structures.

AV: The subset of V that contains the WDVSs which satisfy Constraints R6, R7, R8, R9, R10. These WDVSs are called the Admissible Variable Structures.

AP: The set of Accessible Patterns. One accessible pattern is associated with every WDFS. The accessible pattern of a WDFS Σ is computed on the Ordinary Petri Net associated with Σ. The accessible pattern of a WDVS Π is computed as the intersection (see operator INT) of the accessible patterns of the WDFSs in the support of Π.

Relations

R_i: Equivalence relation defined on X. Two inputs x and x' verify x R_i x' if and only if they correspond to the same observation of Sensor i, i = 1...N.

SUB: Partial ordering defined on W. Σ SUB Σ' if and only if the Ordinary Petri Net associated with Σ is a subnet of the Ordinary Petri Net associated with Σ'.

ACT: Partial ordering defined on V. Π ACT Π' if and only if every interaction in Π' is activated by at least as many inputs as it is in Π.

EFF: Partial ordering defined on AP. E EFF E' if and only if every entry in E is included in the corresponding entry in E', i.e., for every interaction the set of accessible alphabets in E is included in the set of accessible alphabets in E'.

Operator

INT: This operator is defined on AP. Every entry of E INT E' corresponds to the intersection of the corresponding entries in E and E'.