ON THE GENERATION OF ORGANIZATIONAL ARCHITECTURES USING PETRI NETS

by

PASCAL A. REMY

Ingénieur de l'Ecole Polytechnique
(1983)
Ingénieur des Ponts et Chaussées
(1985)

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

December 1986

Signature of Author _____
Department of Electrical Engineering and Computer Science
December 19, 1986

Certified by _____
Dr. Alexander H. Levis
Thesis Supervisor

Accepted by _____
Professor Richard de Neufville, Chairman
Technology and Policy Program

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

ON THE GENERATION OF ORGANIZATIONAL ARCHITECTURES USING PETRI NETS

by

PASCAL A. REMY

Submitted to the Department of Electrical Engineering and Computer Science
on December 19, 1986
in partial fulfillment of the requirements for the degree of
Master of Science in Technology and Policy.

ABSTRACT

A methodology is presented for generating decisionmaking organizational architectures that satisfy some generic structural properties, as well as more specific designer's requirements. Petri Nets are used as the basic technique to represent organizational architectures. The allowable set of interactions among the organization members is first defined, and a mathematical framework is developed to represent the interactions between organization members. The set of organizational architectures satisfying both the structural and the designer's requirements is then analyzed. This set is delimited by its minimal and maximal elements and a technique is given to generate the entire set from its boundaries. Simple paths are used as the incremental unit leading from one organizational form in the set to its neighboring ones. The internal structure of the set is then investigated using results from Lattice theory. The methodology has been implemented on a personal computer; a description of the different modules of the program is provided.

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

ACKNOWLEDGEMENTS

I wish to express my gratitude to:

Dr. Alexander Levis, for his guidance and support throughout the work on this thesis. Working with him has been a broadening as well as enjoyable experience.

Lisa Babine, for her help and kindness.

Scott and Hervé, for their comradeship and the many hours of discussion we had together.

Dr. Robert Shapiro, who developed the software "Design" with which all the Petri Nets shown in this thesis have been drawn.

Dr. John Brode, for his helpful suggestions at the early stage of this thesis.

My family and Pascale.

This research was carried out at the MIT Laboratory for Information and Decision Systems, with support provided primarily by the Office of Naval Research under Contract No. N00014-84-K-0519 and also by the Joint Directors of Laboratories under Contract No. N00014-85-K-0782.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 PROBLEM DEFINITION

Most of the theoretical developments in decision and control theory have addressed the problem of analyzing the performance of a given organizational form. In this case, the organizational structure is fixed and well defined. Some changes in the topology of the organization may occasionally be made in order to improve its performance, but they always remain incremental. There is a need for a methodology to generate in some orderly manner organizational architectures that are not just variants of the same structure. Two main problems need to be addressed to implement successfully such a design methodology.

First, a framework needs to be defined that will specify the class of organizations under consideration. This framework will allow for a mathematical formulation of the design problem and will thus give the organization designer a means to translate into hard numbers the practical problem he - or she - is trying to solve.

Second, compromises have to be made in order to keep the organizational form problem computationally feasible. There is always a trade-off to be made between the explanatory power of a model - its universality - and its usefulness as an analysis tool. Too broad a model may be able to reflect faithfully reality, but might be unusable because of computational limitations. On the other hand, a model that is too restrictive will be able to account only for oversimplified examples and will be useless for complex real-life applications.

1.2 THEORETICAL BACKGROUND

A quantitative methodology for the analysis and evaluation of information processing and decisionmaking organizations has been developed by Levis and his

co-workers [1]. In this model, organization members have a four stage internal structure that allows for the differentiation of the different interactions between two organization members.

This methodology has been used primarily, up to now, for analysis purposes [2][3]. It is, however, very appropriate for addressing the problem of organization design precisely because of the possibility of differentiating between the interactions. The four stage model of the single interacting decisionmaker will be, therefore, the starting point of the theoretical development presented in this thesis.

The mathematical formulation of the problem will be based on Petri Net theory. Petri Nets have been introduced [4] to represent organizational forms because they show explicitly the interactive structure between decisionmakers and the sequence of operations within an organization. Petri Nets appeared to be a very efficient tool not only for the modeling but also for the analysis of decisionmaking organizations [5]. In this thesis, Petri Nets will be primarily used as a design tool, but the dual role of Petri Nets both as a modeling and as an analysis technique should be kept in mind.

1.3 GOALS AND CONTRIBUTIONS

In this thesis, a mathematical model of interactions between decisionmakers is defined. This model allows the organization designer to characterize with an arbitrary level of precision the class of organizations he - or she - is considering. The specificity of the designer's requirements will determine the degrees of freedom left. If the designer's requirement are loose, the dimensionality of the combinatorial problem can be very large. A technique will thus be developed to reduce this dimensionality. The main idea is to characterize the set of allowable organizations by its boundaries only. To give a meaning to the notion of boundaries, a partial order will be defined allowing for a classification of organizations. Lattice theoretic results are then used to gain deeper insight into the internal structure of the set of all allowable organizations.

The different organizational architectures generated by the above procedure can be translated into the conventional Petri Net representation of organizations. These organizations can then be analyzed and their performance, with respect to different criteria,

can be compared. A link is then achieved between the design methodology presented in this thesis and the existing analysis tools that have already been developed by other members of the research team at the MIT Laboratory for Information and Decision Systems.

The overall procedure has been implemented on a personal computer and a program with a user interface is available. It allows the organization designer to go step by step through the entire design methodology.

1.4 THE THESIS IN OUTLINE

The thesis is organized as follows. Chapter II is a review of Petri Net theory: the basic notions are introduced together with some more advanced topics that will be used throughout the thesis. Chapter III is a review of Lattice Theory: it presents the formalism used in subsequent chapters to formulate and analyze the results obtained. In Chapter IV, the basic methodology for defining and representing organizational classes is introduced and Chapter V goes through the translation of this methodology into the language of Petri Net theory. In Chapter VI, additional constraints are introduced that will define the concept of valid organizational form. Chapter VII gives a characterization of the set of all valid organizational forms as well as some properties of the internal structure of this set. The algorithmic implementation of the overall methodology is presented in Chapter VIII and two application examples are given in Chapter IX. Finally, Chapter X concludes the thesis and suggests some developments for further research.

CHAPTER II

REVIEW OF PETRI NET THEORY

Petri Nets will be used throughout this thesis both as an analysis and as a modeling tool. Because they show explicitly the structure of interactions between decisionmakers, Petri Nets are very appropriate for modeling decisionmaking organizations. They lead to a mathematical description of the organization structure that can then be investigated analytically. This chapter presents an introduction to Petri Net theory. The basic notions are introduced together with some more advanced topics that will be used in this thesis, such as the concept of invariant which is discussed in some detail. References related to specific topics are given throughout the chapter. Elements and definitions of general net theory are given in [6] and [7]. Introductory material about Petri Nets may be found in [8],[9],[10] or [11]. Most proofs of the results stated in this chapter can be found in [5] and will therefore be omitted here.

2.1 FUNDAMENTALS

2.1.1 Basic Definitions

Petri net

A **Petri net** - denoted by PN - is a bipartite directed graph represented by a quadruple $PN = (P, T, I, O)$.

$P = \{p_1,...,p_n\}$ is a finite set of **places.**

$T = \{t_1,...,t_m\}$ is a finite set of **transitions.**

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

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

A **node** will refer to either a place or a transition of PN.

A function that takes values from the set of all positive integers may be associated with the arcs of the net. This is equivalent to having the mappings I and O take values from the set of all positive integers. The nets under consideration in this thesis, where I and O take values from {0,1} are called ordinary Petri Nets.

An example of a Petri Net is shown in Fig.2.1; let it be denoted PN_1. Places are represented by circles and transitions by bars.

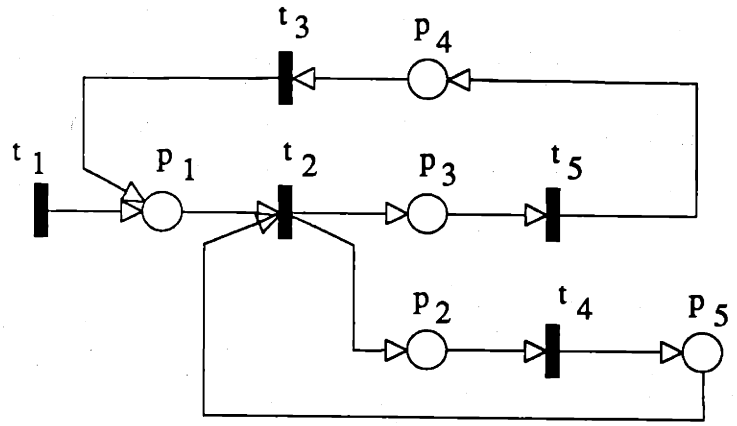


Figure 2.1: Petri Net PN_1.

Postset and Preset

We denote by $t^\bullet = \{p \in P / O(t,p) = 1\}$ the set of all output places of transition t.

Similarly, ${}^\bullet t = \{p \in P / I(p,t) = 1\}$ denotes the set of all input places of transition t.

The same notation applies for places as well.

${}^\bullet p = \{t \in T / O(t,p) = 1\}$ denotes the set of all input transitions of place p.

$p^\bullet = \{t \in T / I(p,t) = 1\}$ denotes the set of all output transitions of place p.

If X is a subset of $P \cup T$, i.e. X is a set of nodes of PN, we define $^\bullet X$ and $X^\bullet$ as follows:

$$X^\bullet = \{ x^\bullet / x \in X \}$$

$$^\bullet X = \{ {}^\bullet x / x \in X \}$$

$X^\bullet$ (resp. $^\bullet X$) will be called the **postset** (resp. **preset**) of X. The definition applies to an element x as well.

As an illustration, let us consider the Petri Net PN_1 of Figure 2.1. We have the following equalities:

$^\bullet p_1 = \{t_1, t_3\}$

$t_2^\bullet = \{p_2, p_3\}$

If $X=\{p_3, t_4\}$ then $X^\bullet = \{t_5, p_5\}$ and $^\bullet X = \{t_2, p_2\}$.

Marking

A **marking** of PN - denoted by M - is a mapping: $P \rightarrow \{0,1,2,...\}$ which assigns a non-negative integer number of tokens to each place of the net. A marking can be represented by a n-dimensional integer vector, whose components correspond to the places of the net.

In Figure 2.1, no tokens are shown. The marking of the net is therefore the following vector:

$$M^0 = (0, 0, 0, 0, 0)$$

Firing

A transition t is **enabled** by a given marking M if and only if for each input place p of t, $M(p) \geq I(p,t)$.

When a transition is enabled it can **fire**. The new marking M' reached after the firing of t is defined as follows :

$$(\forall p \in P) \quad M'(p) = M(p) + O(t,p) - I(p,t) \qquad (2.1)$$

We will write $M \xrightarrow{t} M'$ to indicate that t is enabled by the marking M and that the firing of t yields the new marking M'.

In Figure 2.1, t_1 is the only transition that is enabled. It can actually fire an infinite number of times.

2.1.2 Complementary Definitions

Firing sequence

The sequential firing of transitions $t_{j1},t_{j2},...t_{js}$ will be denoted

$\sigma_s = t_{js} \bullet t_{js-1} \bullet ... \bullet t_{j2} \bullet t_{j1}$.

We will write $M \xrightarrow{\sigma_s} M'$ to indicate that the firing of σ_s yields the marking M'.

The set of all firing sequences of PN will be denoted by T^*. In Figure 2.1, the only possible firing sequences are the power series of t_1, denoted t_1^p, where p is a positive integer.

Parikh mapping - Firing vector

With a firing sequence σ_s can be associated a **firing vector** N_s, also called the **Parikh mapping** [12] of the sequence σ_s: N_s is a $m \times 1$ non-negative integer vector whose j-th component corresponds to the number of occurrences of transition t_j in the sequence σ_s.

Note that the relationship between firing vector and firing sequence is not a one by one correspondence. If $N=[n_1,n_2,...n_m]$ is a firing vector, there will be Σ associated firing sequences with

$$\Sigma = [n_1+n_2+...+n_m]! \,/\, [n_1!n_2!...n_m!].$$

Reachability Set

Given an initial marking M^o of the Petri Net PN, we will call **reachability set** or **forward marking class** [11]- denoted $\Re(M^o)$ - the set of all possible reachable

markings. In other words, a marking M belongs to $\Re(M^o)$ if there exists a firing sequence σ leading from M^o to M.

$$\Re(M^o) = \{M / \exists\, \sigma \in T^*\ \ M^o \xrightarrow{\sigma} M\} \qquad (2.2)$$

The reachability set of PN_1, given the initial marking M^o is obviously

$$\Re_1(M^o) = \{\ (n, 0, 0, 0, 0) / n \text{ positive integer }\}.$$

If we denote by ω the cardinal of the set of positive integers, we can write the reachability set of PN_1 as follows:

$$\Re_1(Mo) = \{(\omega, 0, 0, 0, 0)\}$$

Self-loop and pure Petri Nets

A place p and a transition t are on a **self-loop** if p is both an input and an output place of t. A Petri Net will be **pure** if it does not contain self loops. Petri Nets under consideration in this thesis will all be pure.

Dual of a Petri Net

We will call **dual** of a Petri Net PN, the Petri Net PN' obtained from PN by exchanging places and transitions, and by reversing the direction of the links. Formally, if PN = (P,T,O,I) and PN' = (P',T',O',I'), we have:

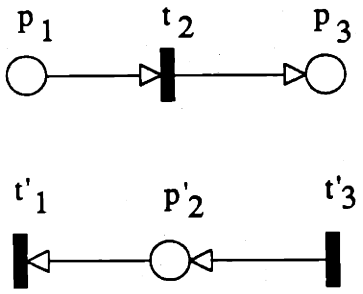
$P' \equiv T$,i.e., transition t_j of T corresponds to place p'_j of P'.

$T' \equiv P$,i.e., place p_i of P corresponds to transition t'_i of T'.

$I'(p'_j,t'_i) = I(p_i,t_j)$ for all $t_j \in T$ and all $p_i \in P$

$O'(t'_i,p'_j) = O(t_j,p_i)$ for all $t_j \in T$ and all $p_i \in P$

The correspondence between the mappings I' and O' and the mappings I and O is illustrated in below.



The following relations hold:

$I(p_1,t_2) = 1$ $\quad$ $I'(p'_2,t'_1) = 1$

$I(p_3,t_2) = 0$ $\quad$ $I'(p'_2,t'_3) = 0$

$O(t_2,p_1) = 0$ $\quad$ $O'(t'_1,p'_2) = 0$

$O(t_2,p_3) = 1$ $\quad$ $O'(t'_3,p'_2) = 1$

Figure 2.2 represents the dual PN_1' of the Petri Net PN_1.

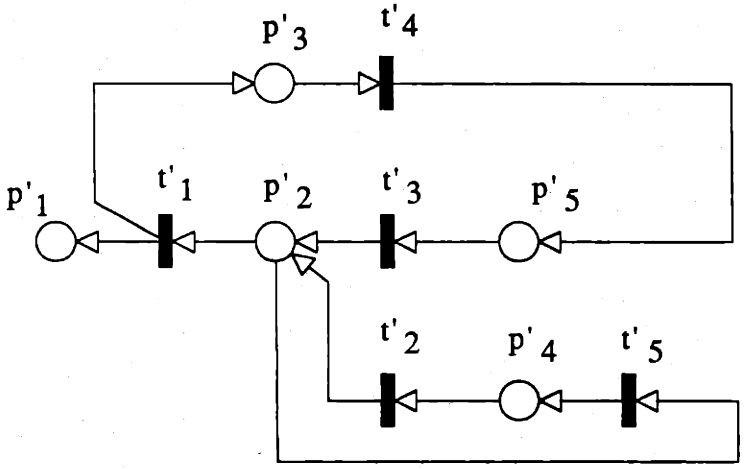


Figure 2.2: Petri Net PN_1', dual of PN_1.

Subnet of a Petri Net

A subnet [5] of a Petri Net PN = (P, T, O, I) is a Petri Net $PN_S = (P_S, T_S, O_S, I_S)$ such that:

$P \supset P_S$; $T \supset T_S$

I_S and O_S are the restrictions of I and O to $P_S \times T_S$ and $T_S \times P_S$, respectively.

2.1.3 Graph Theoretic Definitions

Connectivity

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

Strong Connectivity

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

PN_1 is connected but not strongly connected: there is no directed path from t_1 to p_1, for instance.

Directed Circuit

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

Directed Elementary Circuit

A **directed elementary circuit** is a directed circuit in which no node appears more than once.

PN_1 has two directed elementary circuits:

- $p_1 - t_2 - p_3 - t_5 - p_4 - t_3 - p_1$
- $t_2 - p_3 - t_5 - p_5 - t_4 - p_2 - t_2$

Directed elementary circuits will play a key role in the theory developed in this thesis. An algebraic characterization of those circuits will be given in subsection 2.5 for a specific class of Petri Nets.

2.2 LINEAR ALGEBRA APPROACH

In section 2.1, Petri Nets were introduced using a graph theoretic approach. Petri Nets can also be described in terms of integer arithmetic [13]. In that case Petri nets are referred to as Vector Replacement Systems (VRS) [14] or Vector Addition Systems (VAS) [15].

Incidence Matrix

The topological structure of a pure Petri Net can be represented by an integer matrix C - called **incidence** or **flow matrix**. C is a $n \times m$ matrix whose columns correspond to the transitions and whose rows correspond to the places of the net. C is defined as follows

$$C_{ij} = O(t_j,p_i) - I(p_i,t_j) \quad 1 \le i \le n \quad 1 \le j \le m$$

Note that the definition is restricted to pure Petri Nets. There is actually a problem with non-pure Petri Nets in the sense that self-loops cannot be represented in the incidence matrix: a 1 and a -1 cancel each other to yield a zero in the matrix, losing therefore track of the existence of the self-loop. Since we are considering pure Petri Nets only, we will not have to face this problem.

The mappings O and I can be reconstructed from the matrix C in the following trivial way :

$$O(t_j,p_i) = \max \{ C_{ij} , 0 \}$$

$$I(p_i,t_j) = \min \{ C_{ij} , 0 \}$$

The incidence matrix of the Petri Net PN_1 is given below.

$$C_1 = \begin{bmatrix} 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 \end{bmatrix}$$

The marking of a Petri Net will be represented by a n-dimensional integer vector. The same notation - M - will be kept for both the marking and its vector representation.

If σ_s is a firing sequence associated with a firing vector N_s, the marking M' reached from the initial marking M after the firing of σ_s is given by:

$$M' = M + C \cdot N_s \qquad (2.3)$$

The previous algebraic relation should be used carefully, bearing in mind that some information has been lost in the process leading from a firing sequence to a firing vector. To illustrate the point, let us consider the Petri Net PN_1 of Figure 2.1.

Let us apply relation (2.3) to the following initial marking and firing sequence:

$$M^o = (0, 0, 0, 0, 0)^T$$

$$N_s = (1, 1, 1, 1, 1)^T$$

$$M^1 = M^0 + C_1 . N_s = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 \end{bmatrix}$$

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

M^1 is a positive integer vector corresponding to a valid marking of the Petri Net PN_1. It is however impossible to find a firing sequence associated with the firing vector N_s that will take the marking of PN_1 from M^o to M^1. N_s corresponds to the firing of transitions t_1, t_2, t_3, t_4 and t_5. Whatever the firing order, there will be a point where either transition t_2 or transition t_4 is not enabled. This fact is however hidden in the algebraic relation (2.3).

Duality

The incidence matrix of a Petri Net is the transpose of the incidence matrix of its dual.

The proof is straightforward. If C' is the incidence matrix of the dual PN' of a Petri Net PN, we have:

$$C'_{ij} = O'(t'_j,p'_i) - I'(p'_i,t'_j)$$
$$= O(t_i,p_j) - I(p_j,t_i)$$
$$= C_{ji} \text{ , and therefore } C' = C^T.$$

2.3 SOME PROPERTIES OF PETRI NETS

In this section definitions and results relevant to the subsequent developments will be given.

Boundedness

A marking M^o is **bounded** if there exists a positive integer k such that for every reachable marking M - element of the reachability set $\Re(M^o)$ - the number of tokens in each place is bounded by k. If k equals one, the marking is said to be **safe**. A Petri Net PN is **structurally bounded** if any initial marking of PN is bounded.

PN_1 is not bounded under M^o - or any other marking - since p_1 can have an arbitrarily high number of tokens.

Liveness

A marking M^o is **live** if for any transition t and for every reachable marking M there exists a firing sequence from M that includes t. In other words every transition of the net can fire an infinite number of times.

A Petri Net PN is **structurally live** if any initial marking of PN is live.

Consistency

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

- σ brings the marking M of the net back to itself, i.e. $M \xrightarrow{\sigma} M$
- σ fires each transition at least once.

σ is called a **cyclic firing sequence**.

PN_1 is obviously not consistent.

Persistency

A Petri Net is **persistent** [16] if for all transitions t_1 and t_2 with $t_1 \neq t_2$ and for any reachable marking M so that t_1 and t_2 are both enabled, the firing of one of the transitions cannot disable the other.

Conflict-free Petri Net

A Petri Net is **conflict-free** if every place which is an input of more than one transition is an a self-loop with each such transition

Conflict-free Petri Nets are persistent but the converse needs not be true. PN_1 is a conflict-free Petri Net.

Event graph

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

Pure event graphs will be the only class of Petri nets under consideration in this thesis. Event graphs will be analyzed in more detail in section 2.5.

2.4 INVARIANTS

2.4.1 Definitions

S- and T-invariants

We will call **S-invariant** [13] a $n \times 1$ non-negative integer vector X element of the kernel of C^T, i.e. verifying the relation

$$C^T \cdot X = 0 \qquad (2.4)$$

Similarly, we will call **T-invariant** a $m \times 1$ non-negative integer vector Y element of the kernel of C, i.e. verifying the relation

$$C \cdot Y = 0 \qquad (2.5)$$

Support

The set of places (resp. transitions) whose corresponding components in X (resp.Y) is strictly positive is called the **support** of the invariant and is denoted <X> (resp.<Y>).

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

S- and T-components

Let X be a S-invariant of a Petri Net PN and let <X> be its support. <X> is a set of places of PN, i.e. a subset of P. We call **S-component** [17] associated with X - denoted [X] - the subnet of PN whose set of places is <X> and whose transitions are the input and output transitions of the places of <X> in PN.

$[X] = (P_X, T_X, I_X, O_X)$

with

$P_X = \langle X \rangle$

$T_X = \{p^\bullet / p \in P_X\} \cup \{{}^\bullet p / p \in P_X\}$

I_X (resp. O_X) is the restriction of I (resp. O) to $P_X \times T_X$ (resp. $T_X \times P_X$).

T-components are defined in a similar way. If Y is a T-invariant, we will call **T-component** associated with Y - denoted [Y] - the subnet of PN whose set of transitions is <Y> and whose places are the input and output places of the transitions of <Y> in PN.

S- and T-invariant nets

A **T-invariant net** is a Petri Net whose set of transitions is the support of a T-invariant. In other words, there is a T-invariant whose components are all strictly positive.

Similarly, an **S-invariant net** is a Petri Net whose set of places is the support of an S-invariant. In other words, there is an S-invariant whose components are all strictly positive.

2.4.2 Example

Let us consider the Petri Net PN_2 of Figure 2.3.

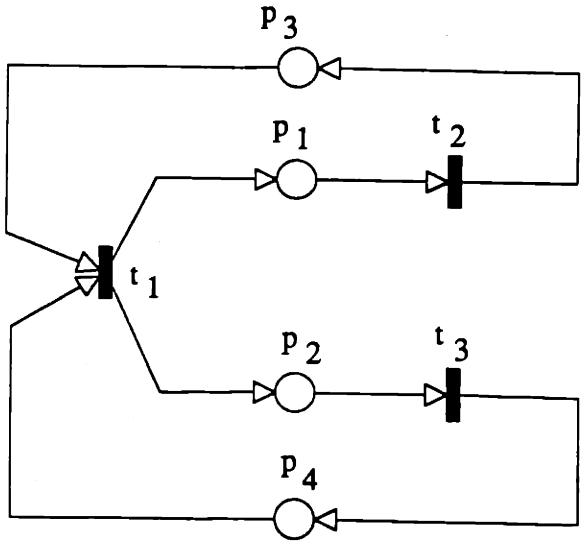


Figure 2.3: Petri Net PN_2.

The incidence matrix of the Petri Net PN_2 represented in Figure 2.3 is

$$C_2 = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

S-invariants

$X = [x_1, x_2, x_3, x_4]^T$ will be a S-invariant if and only if:

$$X^T \cdot C_2 = 0$$

which yields $x_1=x_3$ and $x_2=x_4$. Therefore $X = [x_1, x_2, x_1, x_2]$.

The reader should be easily convinced that there are two minimal support S-invariants:

$X_1 = [1, 0, 1, 0]$ with $\langle X_1 \rangle = \{p_1,p_2\}$

and

$X_2 = [0, 1, 0, 1]$ with $\langle X_2 \rangle = \{p_3,p_4\}$

T-invariants

Similarly, $Y = [y_1, y_2, y_3]$ will be a T-invariant if and only if

$$C_2 \cdot Y = 0$$

which yields $y_1=y_2=y_3$. Once again, it is easily seen that there is a single minimal support T-invariant:

$Y_1 = [1, 1, 1]$ with $\langle Y_1 \rangle = \{t_1,t_2,t_3\}$.

Note that PN_2 is both an S-invariant and a T-invariant net. $X_1 + X_2$ is indeed an S-invariant that includes all the places of PN_2. Similarly Y_1 is a T-invariant that includes all the transitions of PN_2.

2.4.3 Duality

Theorem 2.1

The S-invariants (resp. T-invariants) of a Petri Net PN are exactly the T-invariants (resp. S-invariants) of its dual PN'.

The proof is straightforward. If C is the incidence matrix of PN, C^T will be the incidence matrix of PN' (2.1.1). Let X be a S-invariant of PN. Then $X^T.C = 0$, which can also be written $C^T.X = 0$. X is therefore a T-invariant of PN'. The proof is identical to

show that a T-invariant of PN is an S-invariant of PN'.

2.4.4 Properties of S- and T-invariants

Theorem 2.2

X is a S-invariant of PN if and only if for any initial marking M^o of PN and for any reachable marking M (i.e. $M \in \Re(M^o)$),

$$X^T \cdot M = X^T \cdot M^o \qquad (2.6)$$

The proof is straightforward: (2.6) is obtained by premultiplying equation (2.3) by X^T and by using (2.4) to eliminate the term X^T.C.Ns. Equation (2.6) establishes the conservation of the tokens belonging to the support <X>.

Theorem 2.3

If a Petri Net has a T-invariant, it is possible to construct a marking M and a firing sequence σ such that $M \xrightarrow{\sigma} M$.

Proof:

Let $X=[x_1,x_2,...,x_m]$ be a T-invariant of PN.

For each place p of PN, we choose:

$$M(p) = \sum_{1 \le j \le m} x_j.O(t_j,p) = \sum_{1 \le j \le m} x_j.I(p,t_j).$$

σ is given by $\sigma = t_1^{x1} \bullet t_2^{x2} \bullet ... \bullet t_m^{xm}$.

The following result, due to Memmi and Roucairol [13], justifies a posteriori the introduction of minimal support invariants.

Theorem 2.4

Let $I_1,I_2,...,I_s$ be the minimal supports of the S-invariants of a Petri Net. Let X_i be an invariant whose support is I_i, i.e. $I_i = <X_i>$. The set $\{X_1,X_2,...,X_s\}$ is a **base** of the set of all S-invariants, i.e. a minimal set of generators.

In other words, every S-invariant X can be written as a linear combination of the X_i, with positive rational coefficients λ_i: $X = \sum_{1 \le i \le s} \lambda_i . X_i$

Note that the set of minimal supports of a Petri Net is necessarily finite, since the number of places is finite.

2.5 EVENT GRAPHS

Event graphs will play a key role in this thesis since all the Petri Nets under consideration in the sequel will be event graphs. The definition, already given in section 2.1.1 is recalled here and some important results are stated.

Definition

An **event graph** [9] is a connected Petri Net, in which each place has exactly one input and one output.

The following two theorems, due to Commoner and Holt [18], are of primary importance.

Theorem 2.5

In an event graph, the number of tokens in any elementary directed circuit - the token **content** of the circuit - remains invariant by transition firings.

Theorem 2.6

A marking of an event graph is live if and only if the token content of every directed elementary circuit is strictly positive.

The following result, due to Hillion [5], relates directed circuits and S-components of an event graph. It gives an algebraic characterization of a topological concept and will be extensively used in the sequel.

Theorem 2.7

The minimal S-components of an event graph are exactly its elementary directed circuits.

2.6 SWITCHES

Unlike all the notions introduced in the previous sections of this chapter, switches are not yet part of the current definition of a Petri Net as found in the general literature. Some authors however have already defined the concept [4].

Switches need to be introduced to automate the resolution of conflicts. If we consider the situation presented in Figure 2.4, the token which appears in place p_1 can go either in place p_2 after firing of transition t_1, or in place p_3 after firing of transition t_2. The choice is arbitrary and there is no way in the standard Petri Net theory to automate this choice. This can be a serious shortcoming. Switches are specifically introduced to resolve this problem.

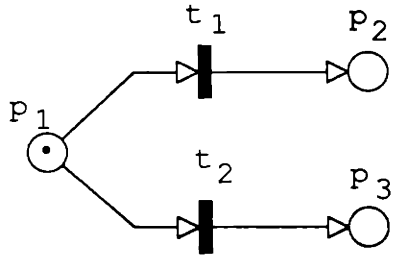


Figure 2.4: Example of conflict.

Different kind of switches can be thought of. The following definition has been retained and will be used in the sequel.

Definition

A **switch** is a transition with multiple output places and some **decision rule**, which directs the output flow of information toward one and only one of its output places.

Since a switch is a kind of transition, the firing rules for a switch will be identical to the firing rules for a transition: a switch will fire if all its input places contain at least one token. Unlike regular transitions however, all the output places of a switch will not receive a token. Only one of them will. This place will be chosen by the internal decision rule associated with the switch. Figure 2.5 gives a example of a switch with two input places and three output places. The output places of the switch will also be called the **branches** of the switch.

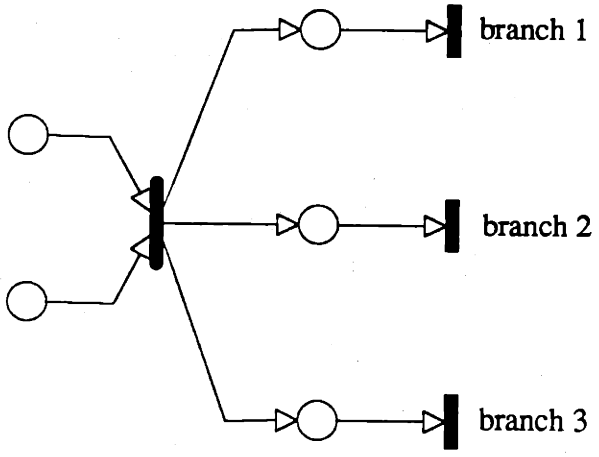


Figure 2.5: Example of a 3-branch switch.

The decision rules associated with the switch can be deterministic or not. They can take the input information explicitly into account or not. They can take the state of the entire Petri Net (the marking of the net) into account or not. Decision rules may be represented by algorithms but they can also involve more sophisticated techniques derived from artificial intelligence. There is virtually no limitations on the kind of decision rules to be associated with a switch, which makes the model rather powerful. The more sophisticated the decision rules, however, the more involved the analysis of the resulting Petri Net. The trade-off has to be made by the designer of the net. Examples of decision rules associated with switches can be found in [2],[19].

CHAPTER III

REVIEW OF LATTICE THEORY

This chapter reviews the basic notions of lattice theory. The emphasis is on definitions and concepts. Few results are actually stated. The formalism of lattice theory will be used in Chapters IV and VII. It will provide a mathematical framework that will help in articulating the problem under consideration and in stating rigorously the results obtained. The reader already familiar with the concept of a lattice may skip this chapter. Introductory material about lattice theory may be found in [20].

3.1 PARTIALLY ORDERED SETS

3.1.1 Fundamental Definitions

Partially ordered set (Poset)

A set X will be **partially ordered** or **partly ordered** by the binary relation $\leq$, if and only if the following properties are satisfied:

(P1) $\forall x \in X \quad x \leq x$

(P2) $\forall (x,y) \in X^2 \quad (x \leq y) \text{ and } (y \leq x) \Rightarrow (x = y)$

(P3) $\forall (x,y,z) \in X^3 \quad (x \leq y) \text{ and } (y \leq z) \Rightarrow (x \leq z)$

We will note $x < y$ whenever $x \leq y$ and $x \neq y$.

If Y is a subset of X we can define a binary relation on Y by restricting the binary relation $\leq$ defined on X to Y. We will keep the same notation $\leq$. If (P1) - (P3) are satisfied by $\leq$ on X, then they are satisfied a fortiori by $\leq$ on Y, since Y is included in X. We have therefore the following result.

Theorem 3.1

Any subset of a partially ordered set (Poset) is itself partially ordered by the same binary relation.

Isomorphism

An **isomorphism** between two partially ordered sets X and Y, is a one to one correspondence φ between X and Y such that:

$$\forall\ (x,y) \in X^2\ (x \leq y) \Leftrightarrow (\varphi(x) \leq \varphi(y))$$

3.1.2 Diagrams

In a hierarchy, it is important to know when one man is another's immediate superior. The notion of immediate superior or immediate superordinate can be defined abstractly in any partially ordered set as follows.

Definition

By **b covers a** is meant that $a < b$ and that $a < x < b$ is not satisfied by any x.

The previous definition leads to a graphical representation of any partially ordered set X. Circles will be drawn to represent the elements of X. A directed arc from b to a will then be drawn whenever b covers a. Any figure so obtained is called a **diagram** of X. It is easily shown that any partially ordered set is defined up to an isomorphism by its diagram.

Example

Let us consider the following set $X_1 = \{0,1\}^3$. X_1 is obviously a partially ordered set under the following order: if $x=(x_1,x_2,x_3)$ and $y=(y_1,y_2,y_3)$ are two elements of X, then $x \leq y$ means $x_i \leq y_i$ for all i between 1 and 3. The 8 elements of X_1 are listed below in Figure 3.1, while the diagram of X_1 is represented in Figure 3.2.

(0,0,0)

(1,0,0) (0,1,0) (0,0,1)

(1,1,0) (1,0,1) (0,1,1)

(1,1,1)

Figure 3.1: Elements of X_1.

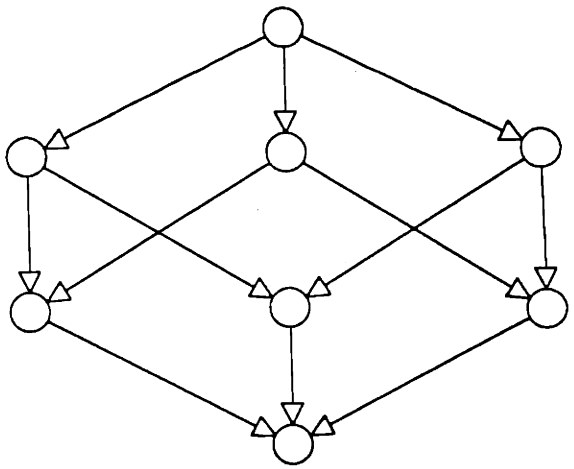


Figure 3.2: Diagram of X_1.

3.1.3 Greatest and Least Elements

Any partially ordered set X can contain at most one element a such that $a \leq x$ for all x element of X. Indeed, if a and a' are two such elements then $a \leq a'$ and $a' \leq a$ by hypothesis. Therefore, $a = a'$, by (P2). Such an element, if it exists, is called the **least element** of X and will be denoted by ω.

Similarly, there is at most one element b verifying $x \leq b$ for all x in X. Such an element, if it exists, is called the **greatest element** of X and will be denoted Ω.

More generally, if Y is a subset of X, we will define the least element of Y, a, and the greatest element of Y, b, as follows.

$$\forall\, y \in Y \quad a \leq y \leq b \quad \text{with } a \in Y \text{ and } b \in Y.$$

It is important to note that a given subset of a partially ordered set needs not have a least or a greatest element. Those notions are not to be confused with the concepts of minimal and maximal elements as defined below.

Definitions

A **minimal element** of a subset Y of a partially ordered set X is an element a of Y such that $a > y$ for no y in Y. Similarly, a **maximal element** of Y is an element b of Y such that $b < y$ for no y in Y.

Clearly a least element is minimal and a greatest element is maximal but the converse need not be true.

Theorem 3.2

Any finite subset Y of a partially ordered set X has minimal and maximal members.

Proof

Let the elements of Y be $y_1,...y_n$. Define the sequences m_k and M_k as follows:

$$m_1 = M_1 = y_1$$

$$m_k = y_k \text{ if } y_k < m_{k-1} \text{ and } m_k = m_{k-1} \text{ otherwise}$$

$$M_k = y_k \text{ if } y_k > M_{k-1} \text{ and } M_k = M_{k-1} \text{ otherwise}$$

Then m_n is a minimal element of Y and M_n is a maximal element of Y.

Atom

An element which covers ω in a partially ordered set X is called an **atom** of X.

If we denote X^* the subset of X obtained by excluding θ, an atom of X is a minimal element of X^*.

3.1.4 Chains

Chain

A partially ordered set X satisfying the condition

(P4) $(\forall (x,y) \in X^2 \ \ (x \leq y) \text{ or } (y \leq x))$,

is said to be simply ordered and called a **chain**.

Chains are convenient to deal with because of the following property.

Theorem 3.3

With chains, the notions minimal and least (resp. maximal and greatest) are equivalent.

Proof

If a is minimal then for no x element of the chain X, $x < a$. By (P4) we have then $a \leq x$ for all x in X. a is therefore the least element of X. The reasoning is similar for maximal elements.

Connected chain

A chain $x_0 < x_1 <...< x_i <...$ will be **connected** if x_i covers x_{i-1} for all i.

Dimension

The **dimension** d[x] of an element x of a partially ordered set X is the maximum length d of chains $x_0 < x_1 <...<x_d=x$ in X having x for greatest element - in case d is finite. Similarly, by the dimension d[X] of X is meant the maximum length of a chain in X.

It is clear that in determining dimension, one only needs to consider connected chains. The notion of dimension is of particular importance when the following condition is satisfied.

Jordan-Dedekind chain condition

All finite connected chains between fixed end points have the same length.

The following theorem [21] gives a characterization of the Jordan-Dedekind condition.

Theorem 3.4

Let X be a partially ordered set which has a ω and a Ω and in which all chains are finite. Then X satisfies the Jordan-Dedekind chain condition if and only if there exists an integer-valued function f[x] such that

$$(x \text{ covers } y) \Leftrightarrow (x > y \text{ and } f[x] = f[y] + 1)$$

3.2 LATTICES

3.2.1 Definitions

Least upper bound

Let Y be a subset of a partially ordered set X. An **upper bound** of Y is an element a of X such that $y \leq a$ for all y in Y. The **least upper bound** - l.u.b. - of Y is the least element - if it exists - of the set of all upper bounds of Y.

The concepts of **lower bound** and **greatest lower bound** - g.l.b. - are defined by analogy.

Lattice

A **lattice** is a partially ordered set X, any two of whose elements have a g.l.b or **meet** - denoted $x \cap y$ - and a l.u.b. or **join** - denoted $x \cup y$ -.

Partially ordered sets in which every subset has a g.l.b. and a l.u.b. are called complete lattices.

3.2.2 Sublattices

Definition

A **sublattice** of a lattice L is a subset of L which contains with any two elements their join and their meet.

It is important to remark that a subset of a lattice L may be a lattice with respect to the binary relation defined on L, without being a sublattice. The crucial point in the definition of a sublattice is that the joint and meet of two elements have to be in the sublattice. We finally define the concept of lattice polynomial that will be used in the following chapters.

Lattice polynomial

If $x_1,x_2,...,x_n$ are elements of a lattice L, we will call **lattice polynomial** $L(x_1,x_2,...,x_n)$ the sublattice of L generated by the elements $x_1,x_2,...,x_n$, i.e.in performing join and meet operations on the x_i.

3.3 EXAMPLE

The lattice formalism presented in this chapter will be used in Chapter IV and VII. An application example will however be given at this point to illustrate the different notions that have been introduced. A more sophisticated example can be found in the Petri Net literature [22], where the properties of the set of all slices of a Petri Net are investigated using the framework of lattice theory.

Let us consider the set X of all 1×4 vectors whose elements take value in {0,1}:

$$X = \{\underline{x} = (x_1,x_2,x_3,x_4) / \forall i \in [1,4] \;\; x_i \in \{0,1\} \}$$

The cardinal of X is 2^4=16. An order can be defined on the set X as follows. If $\underline{x}$ and $\underline{x}'$ are two elements of X,

$$(\underline{x} \leq \underline{x}') \Leftrightarrow (\forall i \in [1,4] \;\; x_i \leq x_i')$$

X with this order is a partially ordered set. The join and meet operators are defined on the set {0,1} as follows.

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

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

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

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

The operators $\cup$ and $\cap$ are then extended to the set X on an element by element basis:

$$\underline{x} \cup \underline{x}' = (x_1 \cup x_1' , x_2 \cup x_2' , x_3 \cup x_3' , x_4 \cup x_4')$$

$$\underline{x} \cap \underline{x}' = (x_1 \cap x_1' , x_2 \cap x_2' , x_3 \cap x_3' , x_4 \cap x_4')$$

Each element of X has a meet and a join within the set X. X is therefore a lattice. X has a least element ω and a greatest element Ω: $\omega = (0,0,0,0)$ and $\Omega = (1,1,1,1)$.

The dimension of an element $\underline{x}$ of X (as defined in 3.1.4) is the sum of all its components: $d[\underline{x}] = x_1+x_2+x_3+x_4$. The following equivalence hold:

$$(\underline{x} \text{ covers } \underline{x}') \Leftrightarrow (\underline{x}' < \underline{x} \text{ and } d[\underline{x}] = d[\underline{x}'] + 1).$$

According to Theorem 3.4, X satisfies the Jordan-Dedekind chain condition. The diagram of X is represented in Figure 3.3. Note that all chains between fixed end points have the same length.

Let us now consider the subset Y of X defined as follows:

$$Y = \{\underline{y} = (y_1,y_2,y_3,y_4) \in X / y_2=y_3 \}.$$

The cardinal of Y is $2^3=8$. Let $\underline{y}$ and $\underline{y}'$ be two elements of Y: $y_2=y_3$ and $y_2'=y_3'$. By definition of the join and meet operations, $y_2 \cup y_2' = y_3 \cup y_3'$ and $y_2 \cap y_2' = y_3 \cap y_3'$. Therefore $\underline{y} \cup \underline{y}'$ and $\underline{y} \cap \underline{y}'$ are elements of Y. Y is a sublattice of X. To show that Y satisfies the Jordan-Dedekind chain condition let us define the following function on Y:

$$f[\underline{y}] = y_1+y_3+y_4 = d[\underline{y}] - y_2.$$

The following equivalence holds:

$$(\underline{y} \text{ covers } \underline{y}') \Leftrightarrow (\underline{y}' < \underline{y} \text{ and } f[\underline{y}] = f[\underline{y}'] + 1).$$

Note that the previous equivalence is not true if the dimension d is used instead of the function f. The function f has been chosen by taking into account the constraint imposed on the elements of Y. Figure 3.4 represents the diagram of Y. Note that the Jordan-Dedekind chain condition is satisfied.

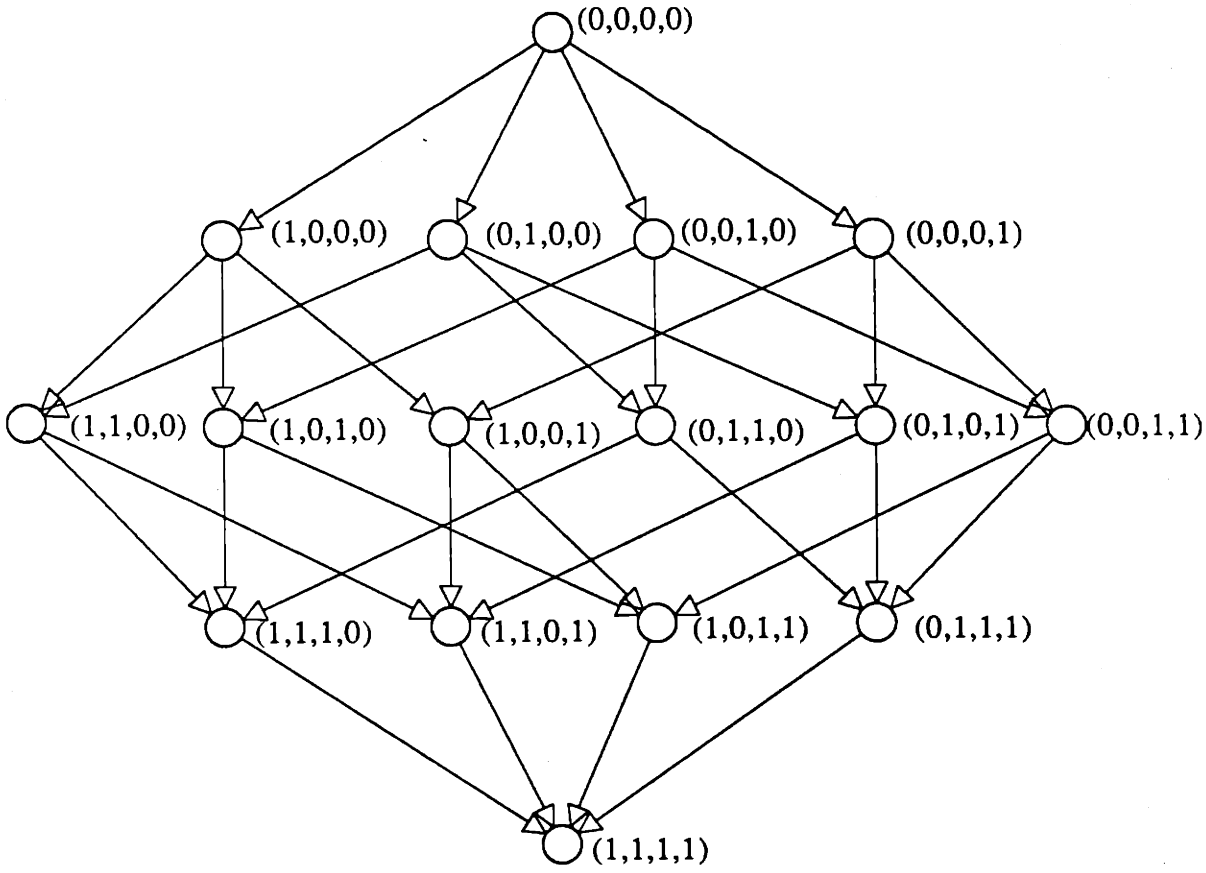


Figure 3.3: Diagram of X.

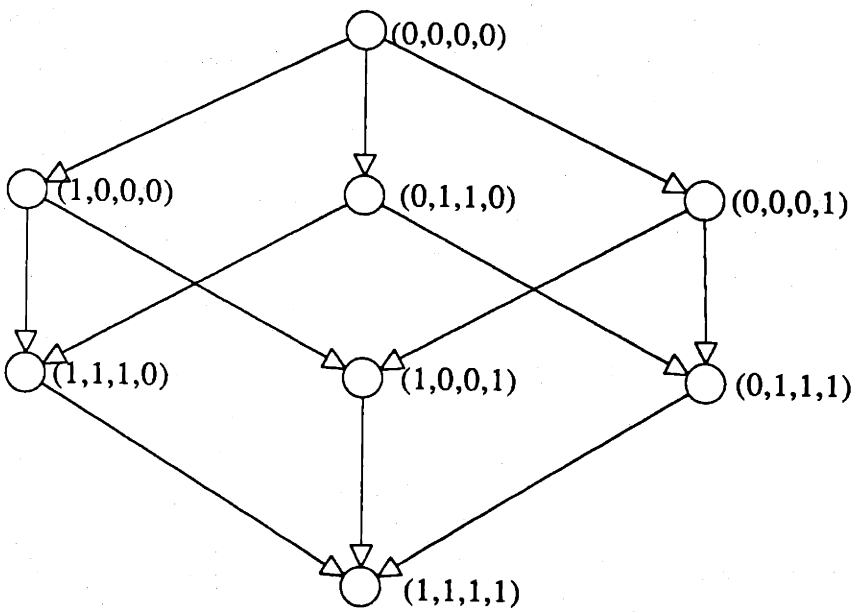


Figure 3.4: Diagram of Y.

Lastly, let us consider the complement Z of Y in X:

$$Z = \{ \underline{z} = (z_1,z_2,z_3,z_4) / z_2 \neq z_3 \}.$$

The following equalities between sets hold: $X = Y \cup Z$ and $Y \cap Z = \varnothing$. Note that in this case, the operators $\cup$ and $\cap$ refer respectively to the union and the intersection of two sets. In the sequel, these symbols will be exclusively used to designate the join and meet operators and no confusion should arise.

The subset Z is not a lattice. Indeed, let us consider the two following elements of Z: $\underline{z} = (0,1,0,0)$ and $\underline{z}' = (0,0,1,0)$. $\underline{z} \cup \underline{z}' = (0,1,1,0) \in Y$ and $\underline{z} \cap \underline{z}' = (0,0,0,0) \in Y$. Therefore, neither the meet nor the join of $\underline{z}$ and $\underline{z}'$ is an element of Z. The diagram of Z is represented is Figure 3.5. Note that this diagram is not connected.

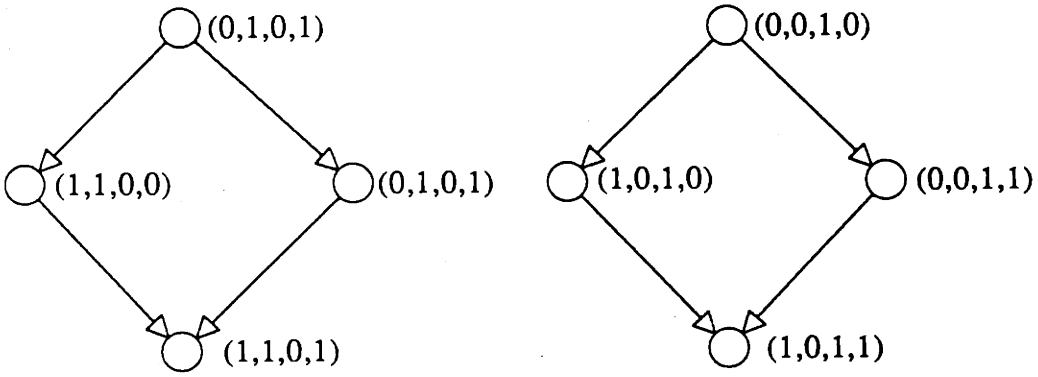


Figure 3.5: Diagram of Z.

CHAPTER IV

ORGANIZATIONAL CLASSES

4.1 OVERVIEW OF THE METHODOLOGY

This chapter introduces a methodology for defining and representing organizational classes. Before entering into the details of the methodology itself, the need for such a methodology must be justified.

Up to now, information processing and decisionmaking organizations have been modeled and analyzed using Petri Nets [2],[3],[4],[5]. Petri Nets are indeed a powerful and convenient tool to represent and study a given organizational structure. When it comes to the problem of designing organizational forms - i.e. when no structure is given a priori -, the general Petri Net theory alone is of little use because of its generality. The scope of organizational forms that can be modeled by Petri Nets is only limited by the imagination of the designer. To make the problem tractable, the class of organizations of interest must be specified with more precision. The proposed methodology consists of three stages. First, the basic constituents of a decisionmaking organization are defined. Then, the allowable interactions between those basic units are introduced, and lastly a mathematical model is derived.

The first step of a methodology for designing decisionmaking organizations is the modeling of a single decisionmaker. The degree of refinement of this basic model will impact the entire design procedure and will be a primary determinant of the kind of results to be expected from the complete methodology. In section 4.2, three basic models of a single decisionmaker, with increasing levels of complexity, are presented. Only one of these models will however be retained in the following developments.

The definition of all allowable interactions between decisionmakers will be a crucial step in the design of our model. This is where assumptions and choices are to be made, that will eventually determine the usefulness, the flexibility, and the power of the model.

Section 4.3 will investigate this matter in some detail.

Finally, a methodology that would just include the first two steps might be useful as a descriptive tool, but would be of little use for analysis. It is necessary to synthesize those ideas into a mathematical model. This model will allow for an analytical representation of decisionmaking organizations. Section 4.4 presents such a model.

4.2 SINGLE INTERACTING DECISIONMAKER

4.2.1 The Black Box Model

The easiest way to represent a decisionmaking unit is to consider it as a black box, with inputs and outputs but without any explicit internal structure (see Figure 4.1). This model is the first that comes to mind and, before developing more sophisticated ones, one should investigate why this simple model is not good enough. The major shortcoming of such a representation is that there is no differentiation among inputs or among outputs, e.g., whether they represent information sharing or commands. A model of decisionmaking organizations based on this black box representation of a single decisionmaker would necessarily be very limited in its analytic capacity. To overcome this shortcoming, the internal structure of the decisionmaker has to be more explicit. The following subsection introduces such an improvement.

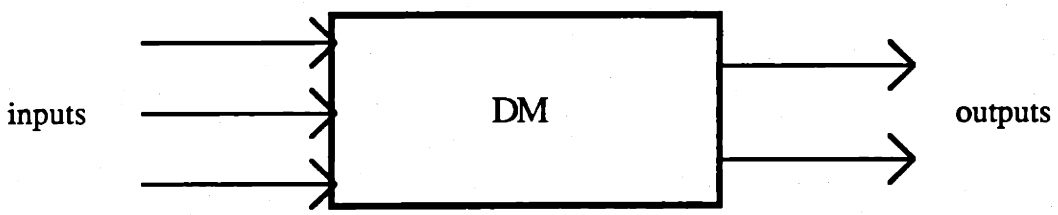


Figure 4.1: Black Box model of a decisionmaker.

4.2.2 The Four Stage Model

This model has been proposed by Levis [1]. A somewhat simplified version of it is reproduced in Figure 4.2. Note that the notation of Petri Net theory - as defined in Chapter II - is used.

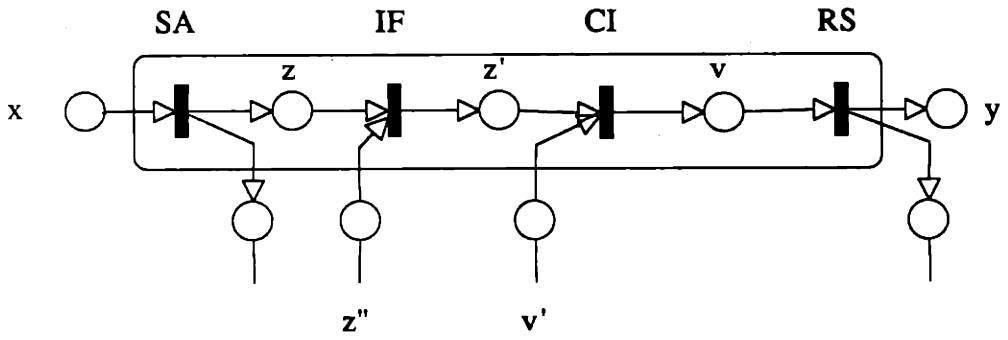


Figure 4.2: Four stage model of a decisionmaker

The decisionmaker receives a signal x - from the external environment or from another organization member. The situation assessment (SA) stage contains algorithms that process the incoming signal to obtain the assessed situation z. The assessed situation z may be shared with other members. Concurrently, the decisionmaker can receive a signal z" from another part of the organization; z" and z are then merged together in the information fusion (IF) stage to produce z'. The possibility of receiving commands from other organization members is reflected in the variable v'. The command interpretation (CI) stage will combine z' and v' to produce the variable v representing the appropriate strategy to use in the response selection (RS) stage. Finally, the RS stage contains algorithms that will produce the output y.

This model explicitly shows at which stage a decisionmaker can interact either with the external environment or with other organization members. A decisionmaker can receive inputs at three different stages: SA (x), IF (z"), and CI (v'). The inputs can be multiple and originate from different organization members. Note, however, that a decisionmaker can receive inputs from the external environment, at the SA stage only. Conversely, a decisionmaker can send outputs at two different stages: SA (z) and RS (y). In both cases

several outputs can be sent. An output to the external environment can however only be sent from the RS stage.

A decisionmaker need not have all four stages. If two given stages are present, however, any intermediate stage must also be present. As an illustration, if a decisionmaker has both SA and RS stages, he must also have the IF and CI stages. Moreover, a decisionmaker must be able to receive inputs at his first stage and to send output from his last stage. The first stage must therefore be the SA, IF or CI stage and the last stage must be either the SA or the RS stage. This yields four different allowable configurations for the internal structure of a decisionmaker:

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

Although the division into four stages proceeds from a certain logic - as explained in the first paragraph of this subsection -, it should be noted that such a division keeps an artificial flavor. One can hardly pretend that the actual cognitive process of a human being can be divided into sequential stages as it is suggested in this model. The four stage model however, has the advantage of differentiating explicitly among the inputs and outputs of a decisionmaker. It allows, furthermore, the estimation of the internal activity of a decisionmaker using information theoretic tools [1],[19],[23]. Lastly, its complexity is high enough to account for realistic examples and small enough to generate tractable models (see Chapters V and VI).

The following subsection will go one step further in the refinement of the single decisionmaker model.

4.2.3 The Four Stage Model with Switches

In subsection 4.2.2, the four stages of the proposed model of a decisionmaker are themselves black boxes. A further refinement would consist in explicitly defining the internal structure of the stages. This has been done by Levis [1] for the SA and RS stages. The model is shown in Figure 4.3.

The SA and RS stages both consist of a set of, respectively, U and V algorithms. Each of these algorithms is represented in Figure 4.3 by a single transition. The choice among those algorithms is achieved in accordance with the situation assessment or the response selection strategy. Switches - as defined in section 2.5 of Chapter II- are used to indicate that a choice has to be made.

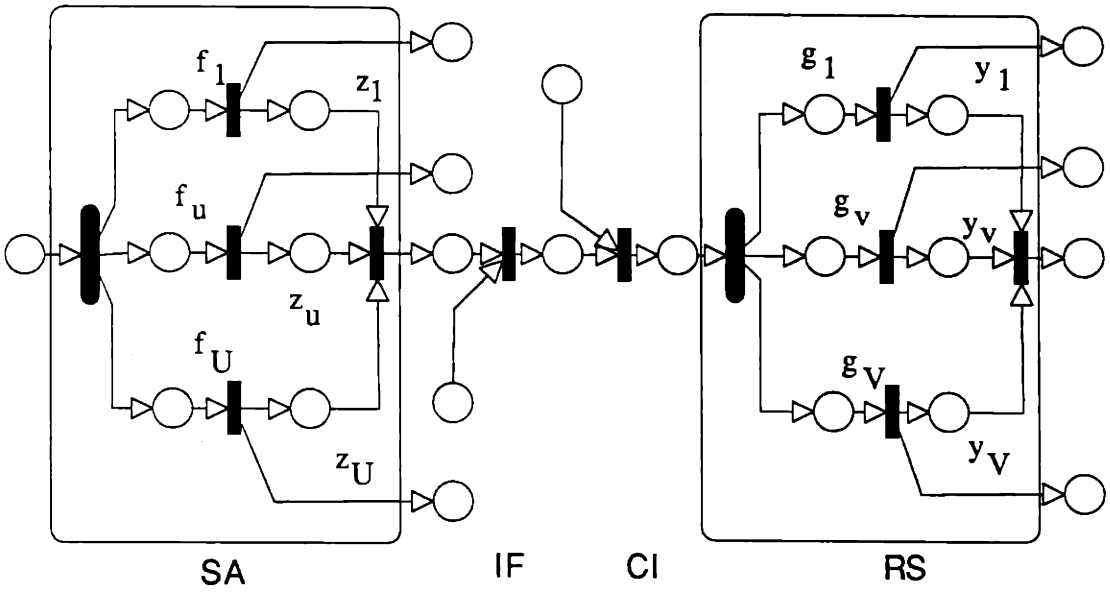


Figure 4.3: Four stage model with switches.

Let us investigate the consequences of introducing switches, as far as organizational structures are concerned. Since the SA and RS stages have similar internal structures, let us concentrate on the SA stage only. The situation assessed by the algorithm f_u - z_u - may be shared with other decisionmakers, as mentioned in 4.2.2. The sharing of situation assessment information may however very well depend on the algorithm used, i.e. on the branch of the switch under consideration. This leads to **variable organizational structures**, i.e. organizations where the structure of interactions among decisionmakers will depend on the kind of information the organization is processing. This is a major change compared to the model described in 4.2.2 which yields fixed structures.

The following example illustrates how the introduction of switches can generate variable structures. Figure 4.4 represents a two person organization with two switches.

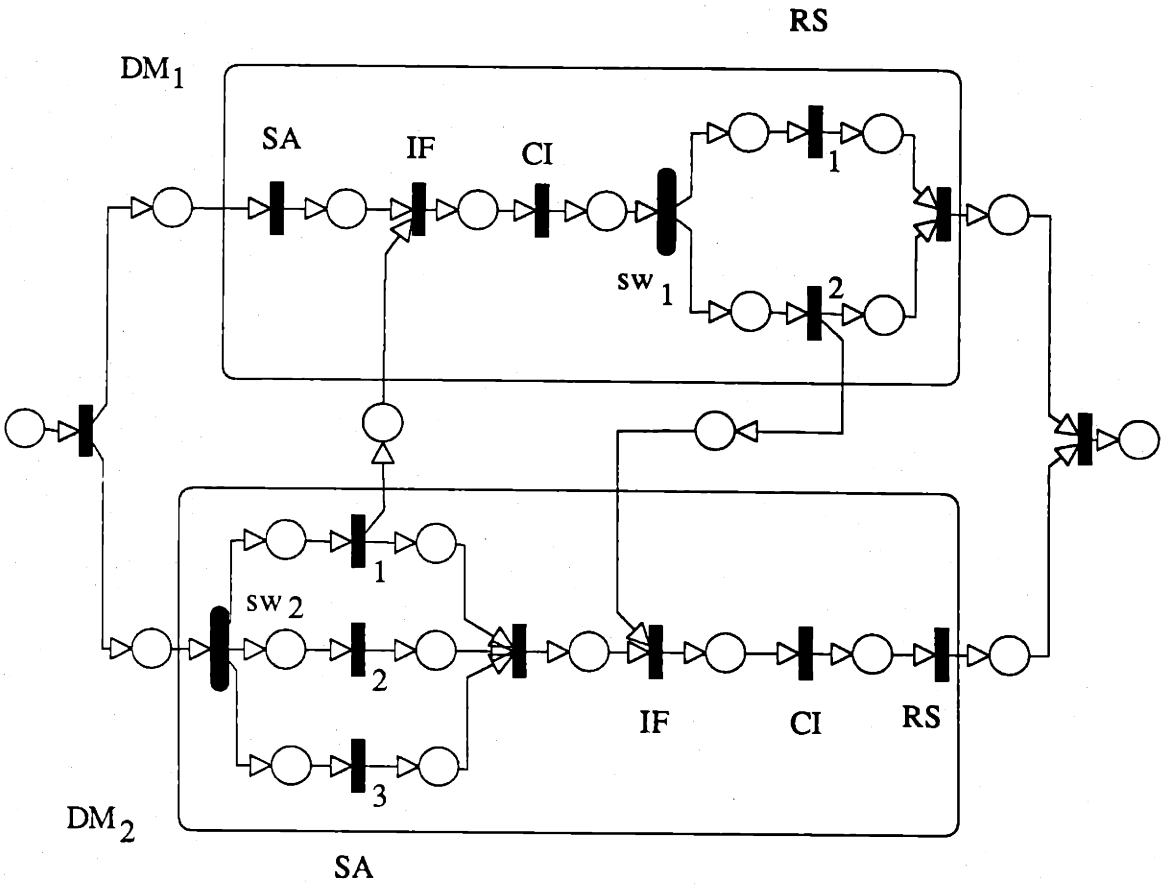
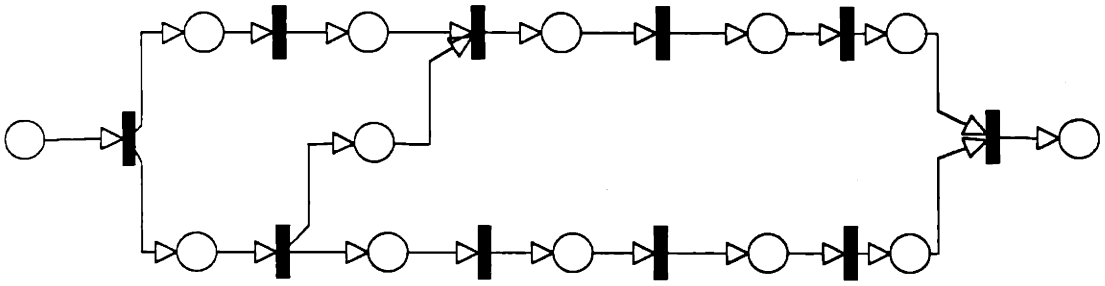
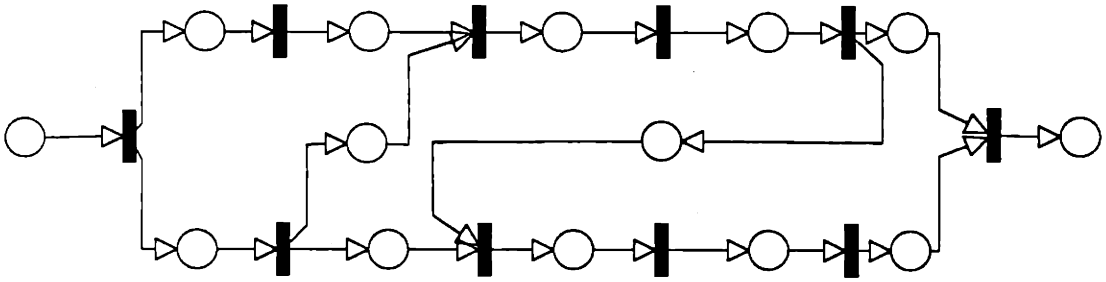


Figure 4.4: Example of a 2-DM organization with switches.

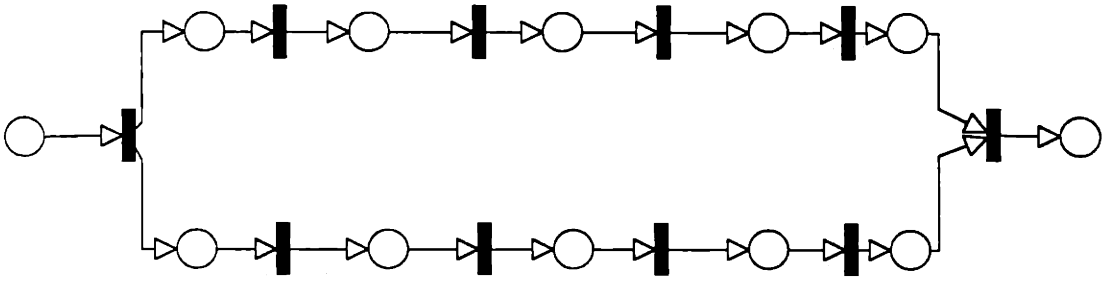
The switch sw_1, corresponding to the RS stage of DM^1, has two branches, while the switch sw_2, corresponding to the SA stage of DM^2, has three branches. We have altogether six different settings of the two switches. Note that the branches 2 and 3 of sw_2 are equivalent, as far as interactions among the decisionmakers are concerned. The six different settings therefore yield four different organizational structures. Those structures are represented in Figure 4.5.



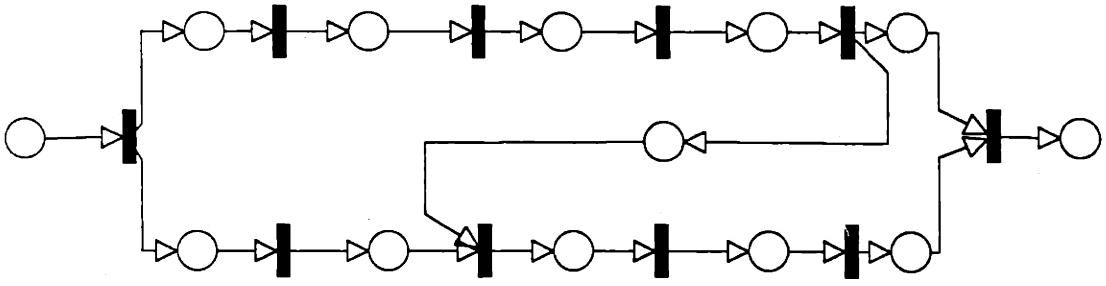
Structure No.1



Structure No.2



Structure No.3



Structure No.4

Figure 4.5: Example of variable structures.

The correspondence between the four different structures and the setting of the switches sw_1 and sw_2 is given in Table 4.1 below.

TABLE 4.1 CORRESPONDENCE BETWEEN STRUCTURES AND SWITCHES.

Structure No.	sw_1	sw_2
1	1	1
2	2	1
3	1	2
	1	3
4	2	2
	2	3

The setting of the two switches will depend on the incoming signal through the decision strategies associated with the switches. The structure of interactions between the two decisionmakers will then depend on the nature of information processed by the organization.

Variable structures will not be considered any further in this thesis. The four stage model of section 4.2.2., which leads to fixed organizational forms, will be retained in the sequel. The intent of this section was to give the reader some hints for further development oriented towards variable structures.

4.3 INTERACTIONS AMONG DECISIONMAKERS

The four stage model of a single decisionmaker described in subsection 4.2.2 will be, from now on, the only one to be considered.

4.3.1 General Case

Since the four stage model of section 4.2.2 allows a decisionmaker to receive three different kinds of inputs and to send two different kinds of outputs, there are six different

ways a decisionmaker DM^i can send information to another decisionmaker DM^j. Figure 4.6 represents those six interactions from DM^i to DM^j. Symmetrical links from DM^j to DM^i are of course also allowable. They have however not been represented in Figure 4.6 for the sake of clarity.

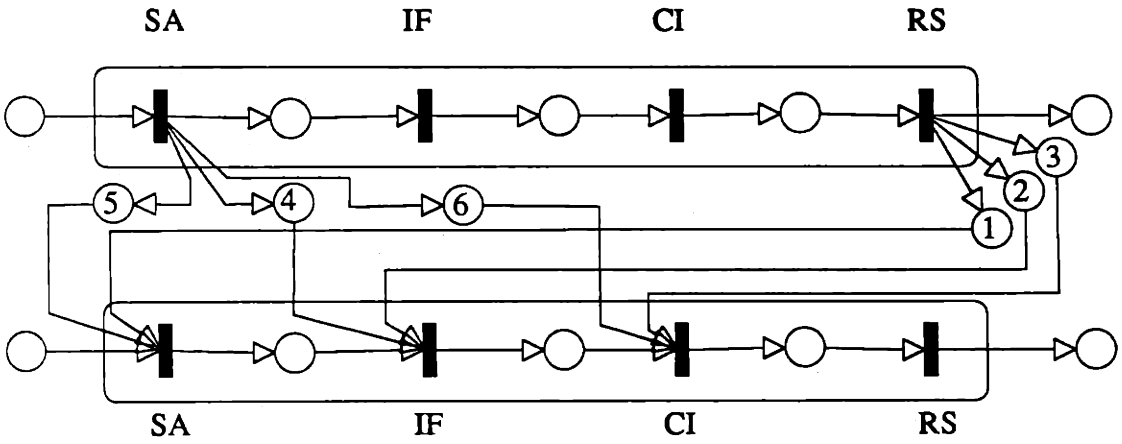


Figure 4.6: General set of interactions between two DMs.

If we consider a n-decisionmaker organization (where n is a positive integer), the maximum number of internal links in the organization will be 6n(n-1). Finally, if the external environment is taken into account, the previous number increases by 2n to reach

$$l_{max} = 6n^2 - 4n.$$

All those interactions will not however be considered, as explained in the following section.

4.3.2 Allowable Interactions

To reduce the dimensionality of the design problem, two of the six possible links between two decisionmakers - as presented in subsection 4.3.1 - will be ruled out. The set of all allowable interactions is represented in Figure 4.7. As in Figure 4.6, links from DM^i to DM^j only have been represented. Symmetrical links from DM^j to DM^i are of course valid

interactions.

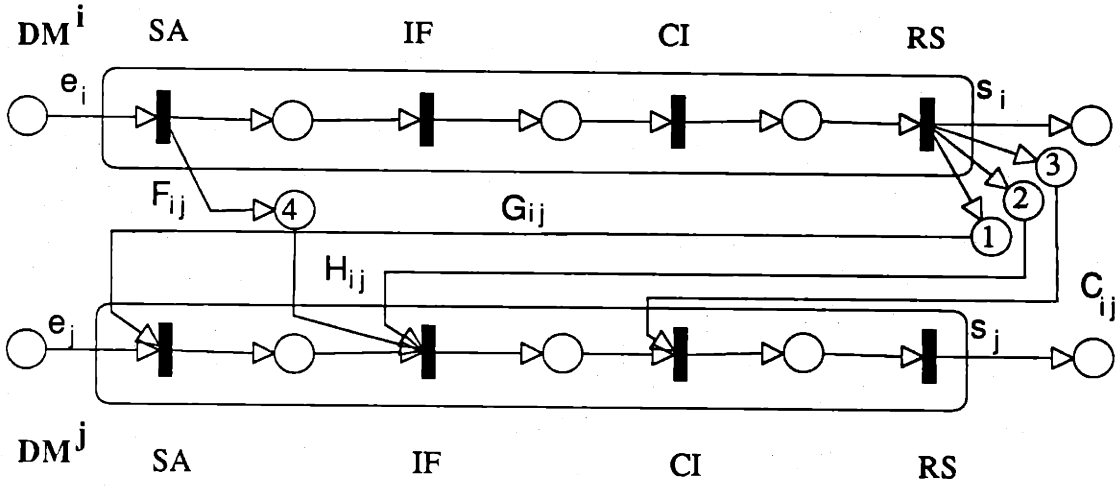


Figure 4.7: Allowable interactions

The prohibited links are the links from the SA stage of a decisionmaker to the SA and CI stages of another decisionmaker (#5 and #6, in Fig. 4.6). There is some degree of arbitrariness in ruling out those two links. From the six possible kinds of interactions between two decisionmakers, they are however the two that have the smallest physical relevance. Indeed, it does not make a lot of sense to send situation assessment information -which is an intermediate result - as a command to another decisionmaker. As far as SA to SA links are concerned, they may be replaced by SA to IF links without significantly altering the meaning of the interaction. Moreover, it will be shown in Chapter VI, how very specific cases not covered by the interaction model of Figure 4.3.2 can be nevertheless accounted for. Those specific cases will include SA to SA links as well as SA to CI links. The loss of generality in ruling out those two interactions, seems therefore relatively small compared to the advantage presented by a reduction in the dimensionality of the problem.

There are now only four possible links from a decisionmaker to another one and the maximum number of links in a n-decisionmaker organization is therefore

$$k_{max} = 4n^2 - 2n.$$

The following section describes the physical significance of the interactions represented in Figure 4.7.

4.3.3 Physical Significance of the Interactions

The four kinds of interactions between two decisionmakers as well as the two different interactions between a decisionmaker and the external environment, are described below. The notation of Figure 4.7 is used.

- External environment to SA of i-th DM: e_i.

This link represents the input a decisionmaker can receive from the external environment. Since the focus of this thesis is on the topological structure of interactions between decisionmakers, the physical content of the input will not be discussed here. The relevant fact is whether or not a decisionmaker will receive information from the external environment. For a discussion of the nature of such information, see [24][25] or the application example of Chapter IX.

- RS of i-th DM to external environment: s_i.

This link models the output a decisionmaker can send to the external environment.

- SA of i-th DM to IF of j-th DM: F_{ij}.

This link models the transmission of situation assessment information from DM^i to DM^j. Once again, the focus here is not on the nature of the information transmitted, but on the fact that an interaction between the two decisionmakers occurs at this stage.

- RS of i-th DM to SA of j-th DM: G_{ij}.

This interaction represents the case where the output of DM^i is the input of DM^j, e.g., a serial or tandem arrangement. As an example, workers on an assembly line present that kind of interactions.

- RS of i-th DM to CI of j-th DM: H_{ij}.

This is the result sharing type of interaction: DM^i sends his final decision to DM^j for informational purpose only. DM^j may or may not take it into account in formulating

his own response. This interaction does not introduce a hierarchical relationship between the two decisionmakers.

• RS of i-th DM to CI of j-th DM: C_{ij}.

This interaction introduces explicitly a hierarchy between the decisionmakers, since it models the issuing of a command from DM^i to DM^j. DM^j must take into account DM^i's orders.

The above description of the interactions between decisionmakers underlines the advantage of breaking down the internal structure of a decisionmaker into four stages. Interactions between two decisionmakers can have completely different meaning depending on the stages to which they are related.

4.4 MATHEMATICAL MODEL

4.4.1 Representation of Interactions

The previous section leads to a mathematical representation of interactions between decisionmakers. The labels $e_i, s_i, F_{ij}, G_{ij}, H_{ij}, C_{ij}$ of Figure 4.7 will be integer variables taking values in {0,1} where 1 will indicate that the corresponding directed link is actually present in the organization, while 0 will reflect the absence of the link.

These variables will be aggregated into two vectors $\underline{e}$ and $\underline{s}$, and four matrices F,G,H, and C. The interaction structure of a n-decisionmaker organization will therefore be represented by the following six arrays.

• Two $n \times 1$ vectors $\underline{e}$ and $\underline{s}$, representing the interactions between the external environment and the organization.

$$\underline{e} \equiv [e_i] \qquad i = 1,2,...,n$$

$$\underline{s} \equiv [s_i] \qquad i = 1,2,...,n$$

$e_i = 1$: DM^i receives an input from the external environment.

$s_i = 1$: DM^i sends an output to the external environment.

• Four $n \times n$ matrices F, G, H, C representing the interactions between decisionmakers inside the organization.

$F \equiv [F_{ij}]$ $\quad G \equiv [G_{ij}]$ $\quad i = 1,2,...,n$

$H \equiv [H_{ij}]$ $\quad C \equiv [C_{ij}]$ $\quad j = 1,2,...,n$

As an example $F_{ij} = 1$ means that DM^i sends an output from his SA stage to the IF stage of DM^j. Similarly $C_{ji} = 1$ means that DM^j sends an output from his RS stage to the CI stage of DM^i. Since the diagonal elements of the matrices F,G,H and C have no meaning, they will be arbitrarily set to 0:

$$F_{ii} = G_{ii} = H_{ii} = C_{ii} = 0 \quad \text{for} \quad i = 1,2,...,n.$$

4.4.2 Well Defined Net

In the framework of subsection 4.4.1, the structure of interactions of an organization is defined by the set of vectors and matrices $\underline{e}$, $\underline{s}$, F, G, H, and C, whose elements take value in {0,1}. The six-tuple {e,s,F,G,H,C} will be called a **Well Defined Net** (WDN) of dimension n, where n is the number of decisionmakers in the organization.

The set of all Well Defined Nets of dimension n will be denoted Ψ^n. It is clear that Ψ^n is isomorphic to the set $\{0,1\}^{k_{max}}$, where k_{max} has been defined in subsection 4.3.2 and is equal to $4n^2 - 2n$. The dimension of Ψ^n is therefore

$$2^{k_{max}} = 2^{4n^2 - 2n}.$$

4.4.3 Lattice Structure of Ψ^n

The formalism of lattice theory, as reviewed in Chapter III, will be very useful for formulating rigorously the results presented in Chapter VII. This formalism is applied in this subsection to the set Ψ^n of all WDNs of dimension n. An order can be defined on this set as follows.

Definition

Let

$\Pi \equiv (\underline{e},\underline{s},F,G,H,C)$ and $\Pi' \equiv (\underline{e}',\underline{s}',F',G',H',C')$
be two WDNs.

We will say that Π' is a **subnet** of Π - denoted $\Pi' \leq \Pi$ - if and only if

$$\underline{e}' \leq \underline{e} \quad F' \leq F \quad G' \leq G$$
$$\underline{s}' \leq \underline{s} \quad H' \leq H \quad C' \leq C$$

The inequality between arrays means that

$$(A' \leq A) \Leftrightarrow (\forall i \in [1,n] \quad \forall j \in [1,n] \quad A'_{ij} \leq A_{ij})$$

This means that Π' is a subnet of Π, if every interaction in Π' is also an interaction in Π. The relation "$\leq$", defined on Ψ^n, verifies the properties (P1)-(P3) of subsection 3.1.1. The set Ψ^n with the relation "$\leq$" is therefore a partially ordered set. The following properties are just translations of some definitions given in Chapter III and are therefore given without further proofs.

The WDN whose arrays have all their elements equal to 0 is the least element of Ψ^n. It will be denoted ω^n. Similarly, the greatest element of Ψ^n is the WDN whose arrays have all their elements equal to 1. It will be denoted Ω^n.

A WDN Π will **cover** a WDN Π' if and only if Π' is a subnet of Π, i.e. $\Pi' \leq \Pi$, and Π has exactly one more link than Π'.

The notion of *dimension* on a partially ordered set has been defined in Chapter III (3.1.4). Because *dimension* has already a meaning for WDNs - this is the number n of decisionmakers- the term *size* will be used to refer to the concept introduced in Chapter III. The size of a WDN Π is its number of links, i.e. the total number of 1s in the arrays defining Π. It will be denoted $d[\Pi]$. We have the following property:

$$(\Pi \text{ covers } \Pi') \Leftrightarrow (\Pi' < \Pi \text{ and } d[\Pi] = d[\Pi'] + 1)$$

According to Theorem 3.4 of subsection 3.1.4, Ψ^n satisfies the Jordan-Dedekind chain condition. The following proposition gives a characterization of the least upper bound (l.u.b.) of two WDNs of dimension n.

Proposition 4.1

Let $\Pi^1 \equiv (\underline{e}^1,\underline{s}^1,F^1,G^1,H^1,C^1)$ and $\Pi^2 \equiv (\underline{e}^2,\underline{s}^2,F^2,G^2,H^2,C^2)$ be two WDNs of dimension n.

The l.u.b. (or join) of Π^1 and Π^2, $\Pi = \Pi^1 \cup \Pi^2$, will be the WDN represented by the arrays $\underline{e},\underline{s},F,G,H$,and C with

$$\underline{e} = \underline{e}^1 \cup \underline{e}^2 \qquad F = F^1 \cup F^2 \qquad C = C^1 \cup C^2$$

$$\underline{s} = \underline{s}^1 \cup \underline{s}^2 \qquad G = G^1 \cup G^2 \qquad H = H^1 \cup H^2$$

The binary operator $\cup$ is an internal composition law defined on the set {0,1} as follows:

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

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

The operator $\cup$ is then extended to arrays taking values on the set {0,1}, on an element to element basis. Note that for the sake of simplicity, the same notation "$\cup$" has been used for three different operations: the composition law defined on the set {0,1}, the extension of this law to arrays, and the join operation between two WDNs.

Proposition 4.1 follows from a two step extension of the binary operator $\cup$: first, from the set {0,1} to arrays taking values in {0,1}, and then to a set of six of these arrays.

Intuitively, the l.u.b. of two WDNs Π^1 and Π^2 is a new net that contains all the interactions that appear in either Π^1 or Π^2 or both. The greatest lower bound of two WDNs is defined in a very similar way by Proposition 4.2.

Proposition 4.2

Let $\Pi^1 \equiv (\underline{e}^1,\underline{s}^1,F^1,G^1,H^1,C^1)$ and $\Pi^2 \equiv (\underline{e}^2,\underline{s}^2,F^2,G^2,H^2,C^2)$ be two WDNs of dimension n.

The g.l.b. (or meet) of $\Pi 1$ and $\Pi 2$, $\Pi = \Pi^1 \cap \Pi^2$,will be the WDN represented by the arrays $\underline{e}$,$\underline{s}$,F,G,H,and C with

$$\underline{e} = \underline{e}^1 \cap \underline{e}^2 \qquad F = F^1 \cap F^2 \qquad C = C^1 \cap C^2$$

$$\underline{s} = \underline{s}^1 \cap \underline{s}^2 \qquad G = G^1 \cap G^2 \qquad H = H^1 \cap H^2$$

The binary operator $\cap$ is an internal composition law defined on the set {0,1} as follows:

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

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

The operator $\cap$ is then extended to arrays taking values on the set {0,1}, on an element to element basis. As it was the case for the join operator "$\cup$", the same notation "$\cap$" has been used for three different operations: the composition law defined on the set {0,1}, the extension of this law to arrays, and the meet operation between two WDNs.

Intuitively, the g.l.b. of two WDNs Π^1 and Π^2 is a new net that contains all the interactions that appear in both Π^1 and Π^2.

From Propositions 4.1 and 4.2 it is clear that the g.l.b. and the l.u.b. of any two WDNs can always be defined and are within the set of WDNs. We have therefore the folowing proposition.

Proposition 4.3

The set Ψ^n of all WDNs of dimension n is a lattice.

CHAPTER V

PETRI NET REPRESENTATION OF ORGANIZATIONAL CLASSES

As mentioned earlier, Petri Nets have been extensively used to study decisionmaking organizations. This is one of the justifications for translating the mathematical model described in the previous chapter into a Petri Net. Perhaps more fundamental, however, is the use that will be made in the sequel of the Petri Net representation of an organization as the starting point from which alternative organizational forms will be generated (see Chapter VI).

This chapter describes how the translation is made between the matrix representation of a WDN, i.e. the mathematical representation described in section 4.4, and the Petri Net representation of the same net. It will be shown that there is a one to one correspondence between both representations.

5.1 TRANSITIONS

As mentioned in subsection 4.2.2, each stage of the four stage model of the single interacting decisionmaker will be represented by a single transition. A decisionmaker will therefore have at most four transitions and a n-decisionmaker organization will contain a maximum of 4n internal transitions. Two supplementary transitions are necessary to represent the external environment acting at both ends of the organization, so that the maximum number of transitions in the Petri Net representation of a WDN will be 4n+2.

Labeling

The internal transitions will be labeled to reflect both the decisionmaker they belong to and the stage they represent. This labeling technique is introduced primarily for computational purposes. This point will be clarified in Chapter VIII, where the algorithmic implementation of the methodology is presented. The labeling will also be used in section 5.4 . Table 5.1 gives the labels associated with all possible transitions of a WDN.

TABLE 5.1 LABELING OF THE TRANSITIONS OF A WDN.

Description	Label
Input transition	t_0
Output transition	t_5
SA of decisionmaker DM^i	t_{1i}
IF of decisionmaker DM^i	t_{2i}
CI of decisionmaker DM^i	t_{3i}
RS of decisionmaker DM^i	t_{4i}

The generic label of an internal transition will then be t_{ki} with $1 \leq k \leq 4$ and $1 \leq i \leq n$. The index k corresponds to the stage and i to the decisionmaker.

5.2 PLACES

A distinction has to be made among the places of the Petri Net representation of a WDN. **Interactional** places will refer to those places that correspond to interactions between two different decisionmakers or between a decisionmaker and the external environment. **Internal** places will correspond to connections that remain within the boundaries of a single decisionmaker. Finally two places, representing the external environment, will be given a special status: the **source** and the **sink** of the organization.

Let us open a brief parenthesis at this point about input and output places. The implicit choice made here to have a single input place (source) and a single output place (sink) may seem a restrictive assumption. It is not. Multiple sources can indeed be represented by a single place associated with a transition that will partition the input information and distribute it to the appropriate organization members [1],[23]. Transition t_0 has been introduced to specifically model such a partitioning process. Similarly, transition t_5 reflects the aggregation process of individual responses from different organization

members into a single response. The advantage of having a single input and a single output places will be demonstrated in Chapter VII, where information paths will be introduced.

5.2.1 Interactional Places

There is a direct one to one correspondence between interactional places and the non zero elements of the matrix representation of a WDN. Each 1 of the arrays representing a WDN corresponds to a link between two stages of two different decisionmakers or to a link between a decisionmaker and the external environment. In the Petri Net representation of the WDN, there will be a place connecting the two considered stages or connecting the external environment and the appropriate stage. Table 5.2. lists all possible links and gives for each of them the correspondence between the matrix and the Petri Net representations. The Petri Net representation is given in terms of input and output transitions for the place under consideration.

TABLE 5.2 CORRESPONDENCE BETWEEN MATRIX AND PETRI NET REPRESENTATIONS

Matrix representation	Corresponding transitions		
	Input		Output
$e_i = 1$	t_0	$\rightarrow$	t_{1i}
$s_i = 1$	t_{4i}	$\rightarrow$	t_5
$F_{ij} = 1$	t_{1i}	$\rightarrow$	t_{2j}
$G_{ij} = 1$	t_{4i}	$\rightarrow$	t_{1j}
$H_{ij} = 1$	t_{4i}	$\rightarrow$	t_{2j}
$C_{ij} = 1$	t_{4i}	$\rightarrow$	t_{3j}

5.2.2 Internal Places

Once interactional places have been defined, internal places are uniquely determined according to the procedure presented in this subsection.

There are three types of internal places characterized by the stages they are related to: SA → IF, IF → CI, and CI → RS. Internal places correspond to the transfer of information within a decisionmaker. Since a given decisionmaker need not have all four stages (4.2.2), he need not have all three internal places. As an example, a decisionmaker may very well perform situation assessments only, in which case he will just have the SA stage with no internal places at all. The internal places of a given decisionmaker will be determined according to the following rules.

- SA → IF

A place will exist between the SA and IF stages of a decisionmaker if and only if

SA has at least one interactional input place.

and

IF has at least one interactional input place.

or

CI has at least one interactional input place.

or

RS has at least one interactional output place.

- IF → CI

A place will exist between the IF and CI stages of a decisionmaker if and only if IF has at least one input place - interactional or internal.

- CI → RS

A place will exist between the CI and RS stages of a decisionmaker if and only if CI has at least one input place - interactional or internal.

The rules concerning IF → CI and CI → RS places are self-explanatory; they just ensure that the flow of information within a decisionmaker is not discontinuous.

The SA → IF rule needs however more detailed explanation. For an internal place to be present between the SA and IF stages of a decisionmaker, the first requirement is that some information be fed into the SA stage. This justifies the first part of the rule. The second part - composed of three propositions related with the predicate "or" - is motivated by the following argument. There is no need to transfer information from the SA to the IF stage of a decisionmaker, if the decisionmaker is not going to use this information in subsequent stages. A characterization of the fact that the decisionmaker is going to use the information in subsequent stages is given by the existence of interactional places related to the subsequent stages. Unless at least one of those interactional places is present, no information need be transferred from the SA to the IF stage and therefore no internal place is necessary.

Note that the three rules presented above insure that a decisionmaker cannot be partitioned into two separate pieces, which is one of the basic assumption made in Chapter IV (subsection 4.1.1). The rules also guarantee the compliance with subsection 4.2.2, where it is stated that only four internal configurations of a decisionmaker are allowable: SA alone, SA-IF-CI-RS, IF-CI-RS, and CI-RS.

More fundamental is the fact that internal places are uniquely determined once interactional places are given. Since a one to one correspondence exists between interactional places and the non null elements of the matrix representation of a WDN, there will be a one to one correspondence between the set of places of a WDN and its matrix representation. Since the set of places of a WDN characterizes the net - given the fixed structure of the set of transitions - there is a one to one correspondence between the Petri Net and the matrix representations of a WDN. From now on, the same terminology will be used for both representations and the term WDN will apply to either of them.

5.2.3 Labeling of the Places

The labeling of places, like the labeling of transitions, is introduced for computational purposes. Its use will be illustrated in section 5.4. The rationale behind the

labeling technique is the following. A place will be labeled with a minimum of two and a maximum of four digits. The minimum number of digits necessary to completely characterize a place will be used. Two digits are sufficient to characterize internal places: the first one will refer to the **input stage of the place**, while the second one will correspond to the **decisionmaker**. Similarly, interactional places related to the external environment at either end of the organization, will be labeled with two digits. Interactional places representing the sharing of situation assessment between two decisionmakers will require three digits. The first digit will characterize the type of place under consideration - namely a SA $\rightarrow$ IF interactional place -, while the other two will refer to the decisionmakers sharing the assessed situation. Lastly, interactional places of the type RS $\rightarrow$ SA , IF, or CI will be labeled with four digits. The first one will characterize the type of place, the second and third ones will refer to the decisionmakers exchanging information and the last one will allow to differentiate between the SA, IF, and CI stages of the decisionmaker receiving information.

In summary, a place will be labeled $p_{i1\ i2\ i3\ i4}$ where i1, i2, i3, and i4 are determined as follows.

- i1 is the first digit of the input transition of the place augmented by 1.

- i2 is the second digit of the input transition. In the case of an interactional place between the external environment and the SA stage, the input transition,t_0, has no second digit and the second digit of the output transition is used instead. This digit will correspond to the decisionmaker to which the place belongs.

- i3 is the second digit of the output transition of the place

- i4 is the first digit of the output transition of the place.

Last of all, the source of the net will be denoted p_0 and the sink will be denoted p_6. Note that the presence of the source place is the justification for the addition of 1 in the definition of i1. Table 5.3 indicates how interactional places are labeled following the rules defined above.

TABLE 5.3 LABELING OF PLACES.

Transitions Input		Output	Corresponding place label
	→	t_0	P_0
t_0	→	t_{1i}	P_{1i}
t_{1i}	→	t_{2i}	P_{2i}
t_{1i}	→	t_{2j}	P_{2ij}
t_{2i}	→	t_{3i}	P_{3i}
t_{3i}	→	t_{4i}	P_{4i}
t_{4i}	→	t_5	P_{5i}
t_{4i}	→	t_{1j}	P_{5ij1}
t_{4i}	→	t_{2j}	P_{5ij2}
t_{4i}	→	t_{3j}	P_{5ij3}

5.2.4 Maximum Number of Nodes

The maximum number of transitions in a n-dimensional WDN will be

$$M_{max} = 4n + 2$$

The maximum number of places can be determined as presented in Table 5.4. Places are listed according to their input transition, which is equivalent to a listing according to the first digit of the numerical part of their label.

TABLE 5.4 MAXIMUM NUMBER OF PLACES OF A WDN.

Description	Label	Maximum number
Source place	p_0	1
Source → SA	p_{1i}	$N_1 = n$
SA → IF	p_{2i}, p_{2ij}	$N_2 = n^2$
IF → CI	p_{3i}	$N_3 = n$
CI → RS	p_{4i}	$N_4 = n$
RS → SA	p_{5ij1} $i \neq j$	$n^2 - n$
→ IF	p_{5ij2} $i \neq j$	$n^2 - n$
→ CI	p_{5ij3} $i \neq j$	$n^2 - n$
→ Sink	p_{5i}	n
Subtotal RS →		$N_5 = 3n^2 - 2n$
Sink place	p_6	1

The maximum number of places of a WDN will therefore be:

$$N_{max} = 1 + N_1 + N_2 + N_3 + N_4 + N_5 + 1,$$

i.e.

$$N_{max} = 4n^2 + n + 2 .$$

Note that N_{max} differs from the number k_{max} as defined in subsection 4.3.2; in fact, N_{max} can be decomposed as follows:

$$N_{max} = k_{max} + N_{int} + 2$$

with $k_{max} = N_1 - n + N_2 + N_5 = 4n^2 - 2n$

$N_{int} = n + N_2 + N_3 = 3n$

The maximum number of interactional places is k_{max}, while N_{int} is the maximum number of internal places. The number 2 in the decomposition of N_{max} accounts for the sink and the source of the net.

5.3 EXAMPLES

The development in the preceding sections will be illustrated with two examples. In both cases the matrix representation of a WDN is given first. The interactional places of the Petri Net representing the WDN are then defined, and ,finally, the internal places are added to complete the picture. The labeling of the places and transitions is given on the figures.

5.3.1 Example 1

Figure 5.1 gives the matrix representation of Π_1, a 3-dimensional WDN. Figure 5.2 presents the interactional structure of Π_1, i.e. the interactional places in the Petri Net representation of Π_1. Lastly, Figure 5.3 represents the entire Petri Net, where internal places as well as sink and source have been added.

$$e = [\,0 \quad 1 \quad 1\,] \qquad F = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$s = [\,0 \quad 1 \quad 1\,] \qquad H = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 5.1: Matrix representation of Π_1

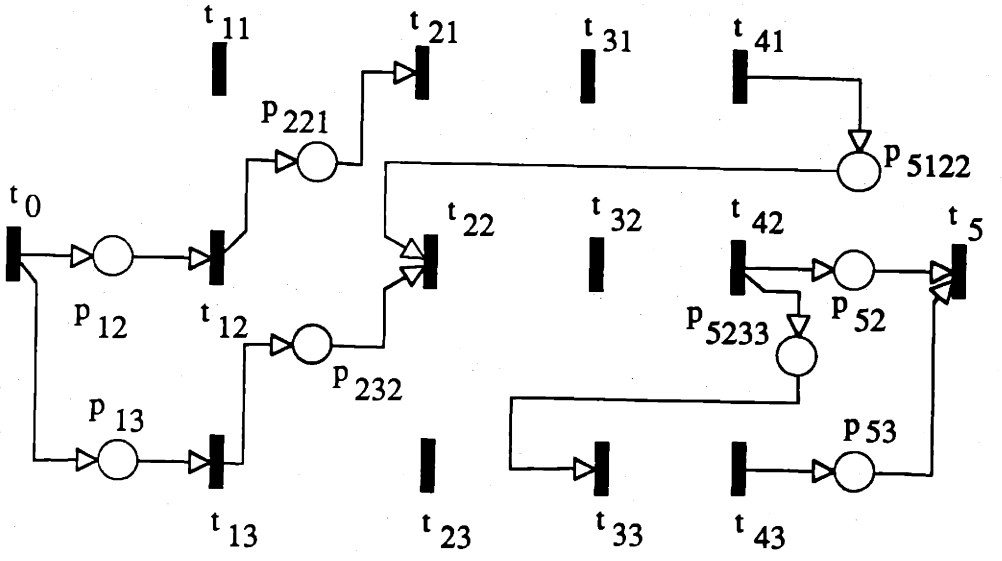


Figure 5.2: Interactional places of Π_1.

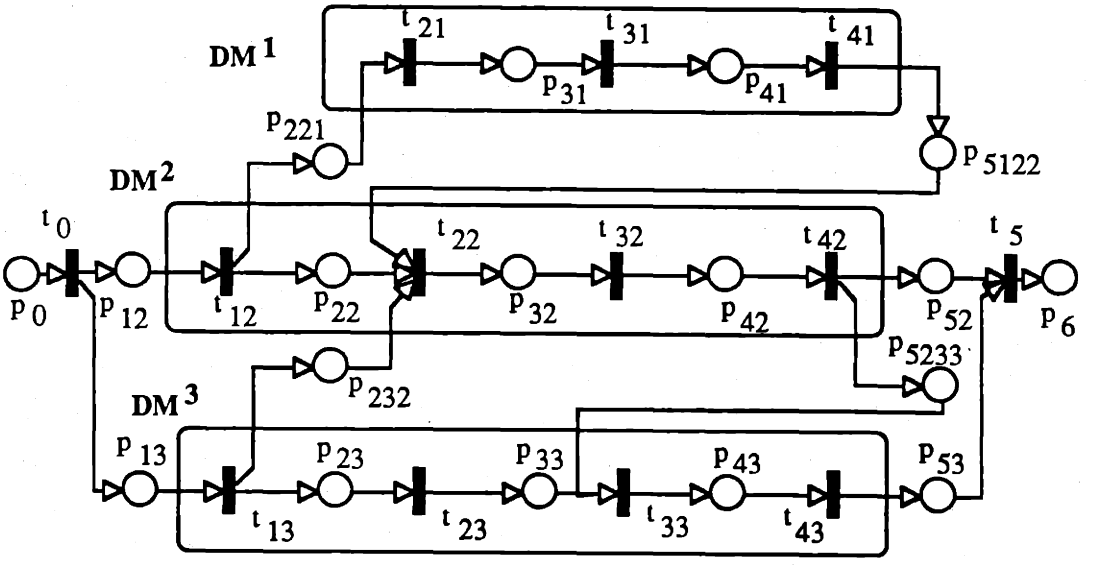


Figure 5.3: Petri Net representation of Π_1.

5.3.2 Example 2

Figure 5.4 gives the matrix representation of Π_2, a 5-dimensional WDN. Figure 5.5 presents the interactional structure of Π_2. Lastly, Figure 5.6 represents the entire Petri Net.

$$e = [\ 1\ 0\ 0\ 1\ 1\ ] \qquad F = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$s = [\ 0\ 1\ 0\ 0\ 1\ ] \qquad H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5.4: Matrix representation of Π_2.

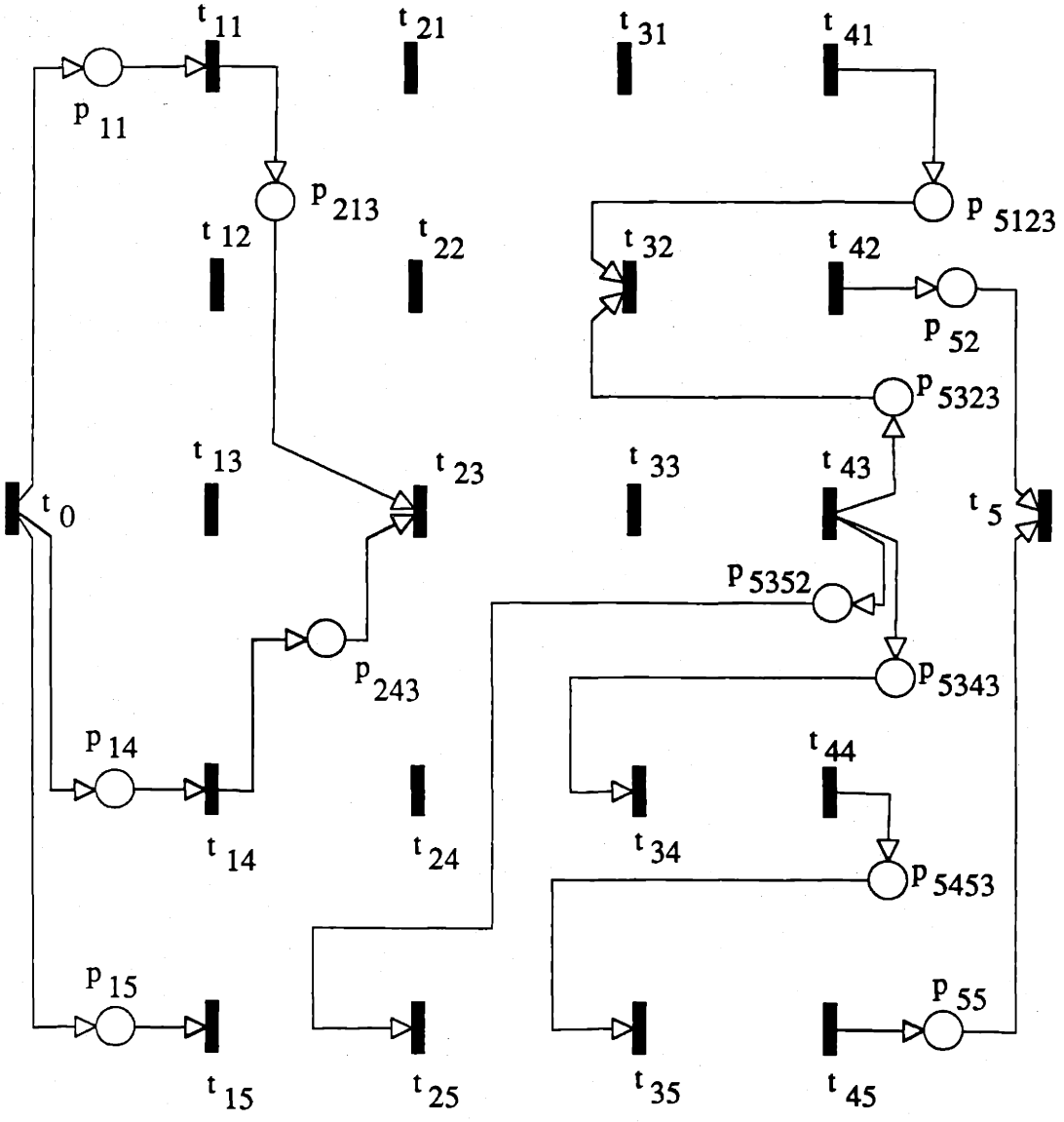


Figure 5.5: Interactional places of Π_2.

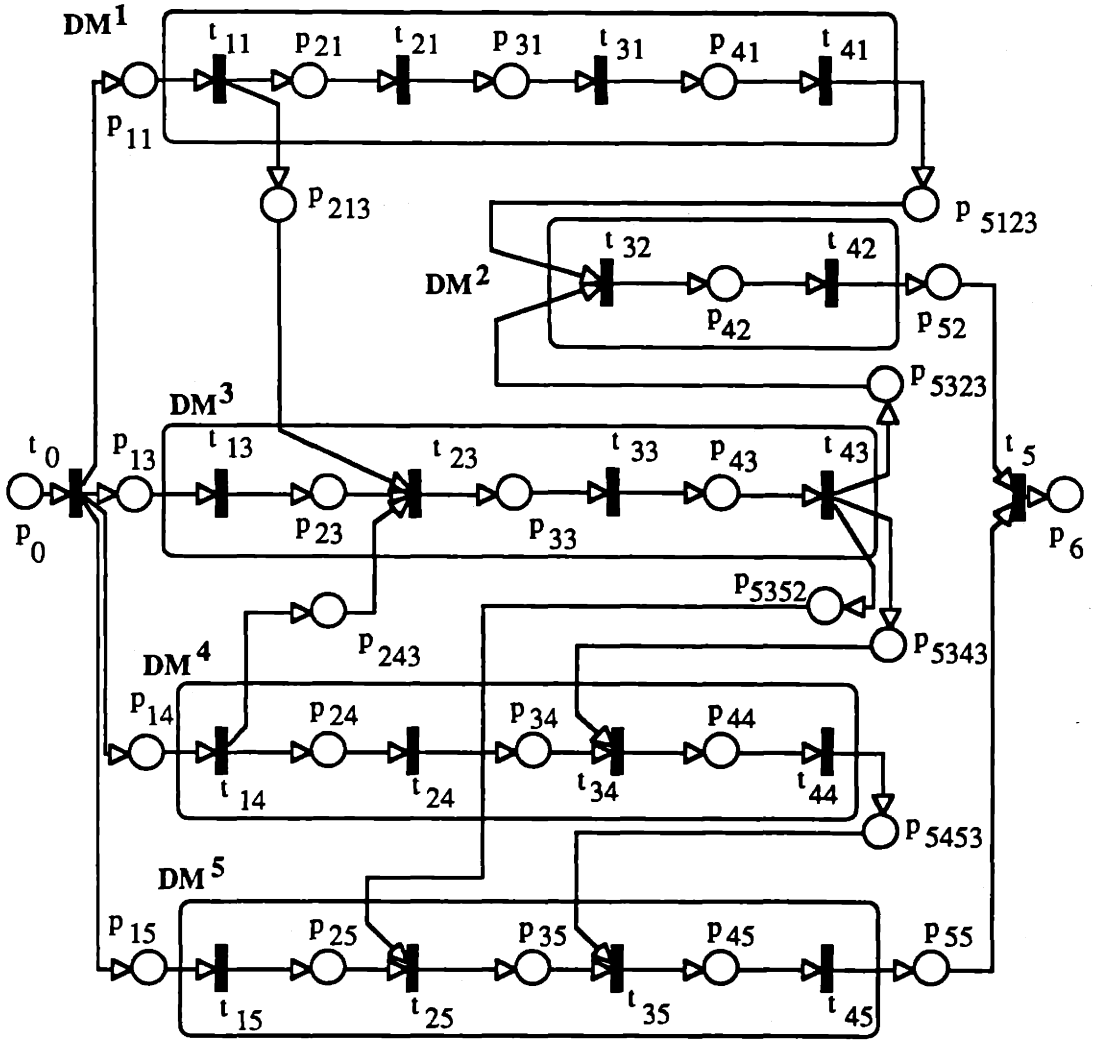


Figure 5.6: Petri Net representation of Π_2.

5.4 INCIDENCE MATRIX

As described in Chapter II, a Petri Net can be represented by an integer matrix reflecting its topological structure. This matrix, called incidence matrix, is the basis of all algebraic computations that can be made on a Petri Net to analyze its properties. This section shows how the incidence matrix of a WDN is constructed.

5.4.1 Regrouping of Places and Transitions

To underline the block structure of the incidence matrix of a WDN, transitions and places will be combined as follows.

Transitions

For each of the four stages SA, IF, CI, and RS (identified by an integer k respectively equal to 1,2,3, and 4), a vector $\underline{t}_k$ will be defined. It is obtained in combining together all the transitions corresponding to the stage k, i.e., all the transitions whose label starts with k. The total number of such transitions will be denoted m_k. Those transitions correspond to the decisionmakers that actually have the stage k in their representation. As pointed out in 5.2.2, a decisionmaker need not have all four stages. Note that m_k is bounded by n, the total number of decisionmakers. The complete set of transitions of a WDN will therefore be:

$$t_0 , \underline{t}_1 , \underline{t}_2 , \underline{t}_3 , \underline{t}_4 , t_5.$$

Since t_0 and t_5 are single transitions, m_0 and m_5 will be equal to 1.We will denote by M the total number of transitions of a WDN. Since every decisionmaker has at most four internal transitions, we have

$$M = m_0 + m_1 + m_2 + m_3 + m_4 + m_5 \leq M_{max} = 4n + 2$$

Places

Similarly, the places will be combined together according to the stages they are related to. The proposed labeling of the places introduced in section 5.2 has been designed to make this classification easy. Places will be regrouped according to the first digit of their

label. Within the same group they will be classified according to the lexicographic order. There are five different groups that will be denoted $\underline{p}_1,\underline{p}_2,\underline{p}_3,\underline{p}_4$, and $\underline{p}_5$. The number of places in each of these groups will be respectively n_1,n_2,n_3,n_4, and n_5. Note that these numbers are not fixed and that their actual value depend upon the WDN under consideration. The following upper bounds have, however, been obtained in subsection 5.2.4:

$n_1 \leq N_1 = n$ $\quad$ $n_2 \leq N_2 = n^2$ $\quad$ $n_3 \leq N_3 = n$

$n_4 \leq N_4 = n$ $\quad$ $n_5 \leq N_5 = 3n^2 - 2n$

N will denote the total number of places of a WDN.

$$N = n_0 + n_1 + n_2 + n_3 + n_4 + n_5 + n_6 \leq N_{max} = 4n^2+n+2$$

In the preceding equality, n_0 and n_6 account for the source (p_0) and the sink (p_6) of the net: they both are equal to 1.

Examples

Let us illustrate the classification of places presented above with the two examples of section 5.3.

Example 1 of subsection 5.3.1

p_0	$n_0 = 1$
$\underline{p}_1 = (p_{12},p_{13})$	$n_1 = 2$
$\underline{p}_2 = (p_{221},p_{222},p_{232},p_{233})$	$n_2 = 4$
$\underline{p}_3 = (p_{31},p_{32},p_{33})$	$n_3 = 3$
$\underline{p}_4 = (p_{41},p_{42},p_{43})$	$n_4 = 3$
$\underline{p}_5 = (p_{5122},p_{522},p_{5233},p_{533})$	$n_5 = 4$
p_6	$n_6 = 1$
	$N = 18$

Example 2 of subsection 5.3.2

p_0	$n_0 = 1$
$\underline{p}_1 = (p_{11}, p_{14}, p_{15})$	$n_1 = 3$
$\underline{p}_2 = (p_{211}, p_{213}, p_{243}, p_{244}, p_{255})$	$n_2 = 5$
$\underline{p}_3 = (p_{31}, p_{33}, p_{34}, p_{35})$	$n_3 = 4$
$\underline{p}_4 = (p_{41}, p_{42}, p_{43}, p_{44}, p_{45})$	$n_4 = 5$
$\underline{p}_5 = (p_{5123}, p_{522}, p_{5323}, p_{5343},$ $p_{5352}, p_{5453}, p_{555})$	$n_5 = 7$
p_6	$n_6 = 1$
	$N = 26$

5.4.2 Construction of the Matrix

The incidence matrix Δ of a WDN Π is defined as follows. Δ is a $N \times M$ integer matrix whose columns correspond to the transitions of Π and whose rows correspond to the places of Π. The matrix Δ is represented in a block format in Figure 5.7. The regrouping of places and transitions into vectors induces the block structure of Δ. The block element B^{ij} will correspond to the place vector $\underline{p}_i$ and to the transition vector $\underline{t}_j$ (if i is equal to 0 or 6 or if j is equal to 0 or 4, the corresponding vector has only one element). B^{ij} is a $n_i \times m_j$ integer matrix whose elements take value in {-1, 0, 1}. B^{00} and B^{65} are therefore scalars. Note that the null elements of the matrix Δ, represented by the symbol 0 in Figure 5.7, have not all the same dimension. For instance, 0 denotes a scalar for the couple (p_0, t_5), while 0 refers to the $n_3 \times m_2$ null matrix for the couple $(\underline{p}_3, \underline{t}_2)$.

	t_0	t_1	t_2	t_3	t_4	t_5
p_0	B^{00}	0	0	0	0	0
p_1^T	B^{10}	B^{11}	0	0	0	0
p_2^T	0	B^{21}	B^{22}	0	0	0
p_3^T	0	0	B^{32}	B^{33}	0	0
p_4^T	0	0	0	B^{43}	B^{44}	0
p_5^T	0	B^{51}	B^{52}	B^{53}	B^{54}	B^{55}
p_6	0	0	0	0	0	B^{65}

Figure 5.7: Block representation of Δ.

The different block elements of Δ are interpreted below.

- B^{10} and B^{11} (resp B^{55} and B^{65}) account for the places located between the input (resp. output) transition of the organization and the SA (resp. RS) stages of the decisionmakers.
- B^{32} and B^{33} (resp. B^{43} and B^{44}) account for the internal places located between the IF and CI (resp. CI and RS) stages.
- B^{21} and B^{22} accounts for all the places located between the SA and IF stages. Those places may be internal as well as interactional.
- Lastly, B^{51}, B^{52}, B^{53} and B^{54} account for all the output places of the RS stages of the decisionmakers.

The determination of the non-zero elements of Δ is done as follows. The rows of Δ will be scanned one by one. Since every place of a WDN - but the source and the sink - has exactly one input and one output transition, each row of Δ - but the first and the last ones - will have exactly one "-1" and one "1". The location of these non-zero elements will be known, if the input and output transitions of the place corresponding to the row are known. The labeling of the places has been designed in such a way that one can identify the input and output transitions of a given place by inspection of its label. The labeling of the places will therefore be the only information used to determine explicitly the elements of Δ. In other words, the complete information concerning the structure of a WDN is included in the labeling of its places. This point will be developed in subsection 5.4.4.

In the following development, the transition corresponding to the l-th column of Δ will be denoted by $t_{i'j'}$. To characterize Δ_{kl} (the element of Δ at the intersection of the k-th row and the l-column), the Kronecker delta will be used. It provides a mechanism for writing integer equations involving 0 and 1 only. This makes the translation into a computer language straightforward. The Kronecker delta is defined as follows:

$$\delta_{ij} = 1 \quad \text{if } i=j \quad \text{and} \quad \delta_{ij} = 0 \quad \text{if } i \neq j.$$

Let us consider the k-th row of Δ ; this row will correspond to a place p. Four cases will be distinguished according to the number of digits of the label of p.

• p has a one digit label: $p \equiv p_0$ or $p \equiv p_6$.

This case is straightforward: p_0 is the source of the organization and has only an output transition, t_0, while p_6 is the sink of the organization and has only an input transition, t_5. Therefore $B^{00} = -1$ and $B^{65} = 1$. All other elements of the first and last rows of Δ are equal to zero.

• p has a two digit label: $p \equiv p_{ij}$.

This case covers the block elements B^{10}, B^{11}, B^{32}, B^{33}, B^{43}, B^{44}, and include some of the rows of the block elements B^{21}, B^{22}, B^{54}, and B^{55}. From Table 5.3, it can be seen that the input transition of p_{ij} is $t_{i-1\,j}$ if i is greater than 1, and t_0 if i is equal to 1.

Similarly, the output transition of p_{ij} is t_{ij} if i is less than 5, and t_5 if i is equal to 5. Therefore Δ_{kl} will be +1 if it corresponds to transition $t_{i-1\,j}$ (or t_0), -1 if it corresponds to transition t_{ij} (or t_5), and 0 in all other cases. This can be formally written using the Kronecker delta. As mentioned earlier ,the l-th transition of Δ will be denoted $t_{i'j'}$.

$$\Delta_{kl} = (\delta_{i\,i'+1} - \delta_{ii'}) * \delta_{jj'} \qquad \text{if } 1 \le i' \le 4 \qquad (5.1)$$

$$\Delta_{kl} = (\delta_{i\,i'+1} - \delta_{ii'}) \qquad \text{if } i'=0 \text{ or } i'=5 \qquad (5.2)$$

• p has a three digit label: $p \equiv p_{2ij}$.

This case covers the rows of $B^{21}$and B^{22} that were left out from the previous case. According to Table 5.3, the input transition of p_{2ij} is t_{1i} and the output transition of p_{2ij} is t_{2j}. Therefore,

$$\Delta_{kl} = \delta_{1i'} * \delta_{ij'} - \delta_{2i'} * \delta_{jj'} \qquad (5.3)$$

• p has a four digit label: $p \equiv p_{qijr}$.

Up to now, the only four digit label places encountered are the interactional places of the type RS $\rightarrow$ SA, IF or CI. The first digit of their label is 5. A generic index q is used, however, because another kind of interactional places with a four digit label will be introduced in Chapter VI. The general case, applying to any kind of four digit label places, is treated here, so that no further development will be needed in Chapter VI. The case q=5 covers the matrices B^{51}, B^{52}, B^{53}, B^{54}, and B^{55}. The input transition of p_{qijr} will be $t_{q-1\,i}$ and the output transition will be t_{rj}. Therefore,

$$\Delta_{kl} = \delta_{q-1\,i'} * \delta_{ij'} - \delta_{ri'} * \delta_{jj'} \qquad (5.4)$$

Equations (5.1),(5.2), (5.3) and (5.4) completely characterize the matrix Δ.

5.4.3 Example

Let us take the two examples of section 5.3 and construct the incidence matrix for each of them.

Example 1

The incidence matrix Δ_1 of Π_1 is represented in Figure 5.8 below.

	t_0	t_1	t_2	t_3	t_4	t_5
p_0	-1	0 0	0 0 0	0 0 0	0 0 0	0
p_{12}	1	-1 0	0 0 0	0 0 0	0 0 0	0
p_{13}	1	0 -1	0 0 0	0 0 0	0 0 0	0
p_{221}	0	1 0	-1 0 0	0 0 0	0 0 0	0
p_{22}	0	1 0	0 -1 0	0 0 0	0 0 0	0
p_{232}	0	0 1	0 -1 0	0 0 0	0 0 0	0
p_{23}	0	0 1	0 0 -1	0 0 0	0 0 0	0
p_{31}	0	0 0	1 0 0	-1 0 0	0 0 0	0
p_{32}	0	0 0	0 1 0	0 -1 0	0 0 0	0
p_{33}	0	0 0	0 0 1	0 0 -1	0 0 0	0
p_{41}	0	0 0	0 0 0	1 0 0	-1 0 0	0
p_{42}	0	0 0	0 0 0	0 1 0	0 -1 0	0
p_{43}	0	0 0	0 0 0	0 0 1	0 0 -1	0
p_{5122}	0	0 0	0 -1 0	0 0 0	1 0 0	0
p_{52}	0	0 0	0 0 0	0 0 0	0 1 0	-1
p_{5232}	0	0 0	0 0 0	0 0 -1	0 1 0	0
p_{53}	0	0 0	0 0 0	0 0 0	0 0 1	-1
p_6	0	0 0	0 0 0	0 0 0	0 0 0	1

Figure 5.8: Incidence matrix Δ_1 of Π_1.

Example 2

Figure 5.9 gives the incidence matrix Δ_2 of Π_2.

	t_o	t_1	t_2	t_3	t_4	t_5
p_0	-1	00000	0000	00000	00000	0
p_{11}	1	-10000	0000	00000	00000	0
p_{13}	1	00-100	0000	00000	00000	0
p_{14}	1	000-10	0000	00000	00000	0
p_{15}	1	0000-1	0000	00000	00000	0
p_{21}	0	00000	1000	-10000	00000	0
p_{212}	0	00000	1000	00-100	00000	0
p_{23}	0	00000	0100	00-100	00000	0
p_{243}	0	00000	0010	00-100	00000	0
p_{24}	0	00000	0010	000-10	00000	0
p_{25}	0	00000	0001	0000-1	00000	0
p_{31}	0	00000	0000	10000	-10000	0
p_{33}	0	00000	0000	00100	00-100	0
p_{34}	0	00000	0000	00010	000-10	0
p_{35}	0	00000	0000	00001	0000-1	0
p_{41}	0	00000	0000	10000	-10000	0
p_{42}	0	00000	0000	01000	0-1000	0
p_{43}	0	00000	0000	00100	00-100	0
p_{44}	0	00000	0000	00010	000-10	0
p_{45}	0	00000	0000	00001	0000-1	0
p_{5123}	0	00000	0000	0-1000	10000	0
p_{52}	0	00000	0000	00000	01000	-1
p_{5323}	0	00000	0000	0-1000	00100	0
p_{5343}	0	00000	0000	000-10	00100	0
p_{5352}	0	00000	000-1	00000	00100	0
p_{5453}	0	00000	0000	0000-1	00010	0
p_{55}	0	00000	0000	00000	00001	-1
p_6	0	00000	0000	00000	0000 0	1

Figure 5.9: Incidence matrix Δ_2 of Π_2.

5.4.4 Equivalence Between the Representations of a WDN.

A WDN can be represented in three different ways:

(1) The matrix representation, i.e. the arrays e,s,F,G,H,C, as presented in section 4.4 of Chapter IV.

(2) The Petri Net representation, given by the graph or the incidence matrix of the net, with the associated labeling of the transitions.

(3) The Petri Net representation, given by the labeling of the places.

The three different representations of a WDN listed above are equivalent, i.e. a one to one correspondence exists between any two of them. The proof of the previous statement is implied in the definitions of the different representations and in the way they are derived from each other. The logic leading from one representation to another is recalled below.

• (1) $\Rightarrow$ (3) : section 5.2 shows how the labeling of the places of a WDN is uniquely determined from the matrix representation of the net.

• (3) $\Rightarrow$ (2) : subsection 5.4.2 explains how the incidence matrix of a WDN with the associated labeling of the transitions can be obtained once the labeling of the places is known. One can also go directly from the labeling of the places to the graph representation of the net. The procedure to follow is outlined below and illustrated on an example.

1. Determine the number of decisionmakers in the organization, i.e., the dimension of the WDN. This is the highest number appearing in second position in the labels of the places.
2. Find all the transitions of the net with their labeling: they are all the input and output transitions of the places of the net and are obtained in reversing the procedure described in subsection 5.2.3 (Table 5.3).
3. Find all the internal places and draw the boundaries of each decisionmaker.
4. Connect all the interactional places with the appropriate transitions.

The four step procedure leading from the labeling of the places of a WDN to the graph representation of the same WDN is illustrated on the following example. Let the set of all places of a WDN Π_3 be

$$P_3 = \{ p_0, p_{11}, p_{13}, p_{212}, p_{232}, p_{23}, p_{32}, p_{33}, p_{42}, p_{43}, p_{5233}, p_{52}, p_6 \}.$$

The largest number appearing in second position in the labels is 3; the WDN Π_3 is therefore 3-dimensional. The input and output transitions of the places of P_3 are obtained using Table 5.3 and are given below.

TABLE 5.5 INPUT AND OUTPUT TRANSITIONS OF THE PLACES OF P_3.

Place	Transition	
	Input	Output
p_0		t_0
p_{11}	t_0	t_{11}
p_{13}	t_0	t_{13}
p_{212}	t_{11}	t_{22}
p_{232}	t_{13}	t_{22}
p_{23}	t_{13}	t_{23}
p_{32}	t_{22}	t_{32}
p_{33}	t_{23}	t_{33}
p_{42}	t_{32}	t_{42}
p_{43}	t_{33}	t_{43}
p_{5233}	t_{42}	t_{33}
p_{52}	t_{42}	t_5
p_6	t_5	

The set of all transitions of Π_3 is, therefore:

$$T_3 = \{t_0, t_{11}, t_{13}, t_{22}, t_{23}, t_{32}, t_{33}, t_{42}, t_{43}, t_5 \}.$$

There are five internal places in P_3: p_{23}, p_{32}, p_{33}, p_{42}, and p_{43}. Figure 5.10 summarizes the information gathered up to now about the graph of Π_3. The three decisionmakers are represented with their boundaries.

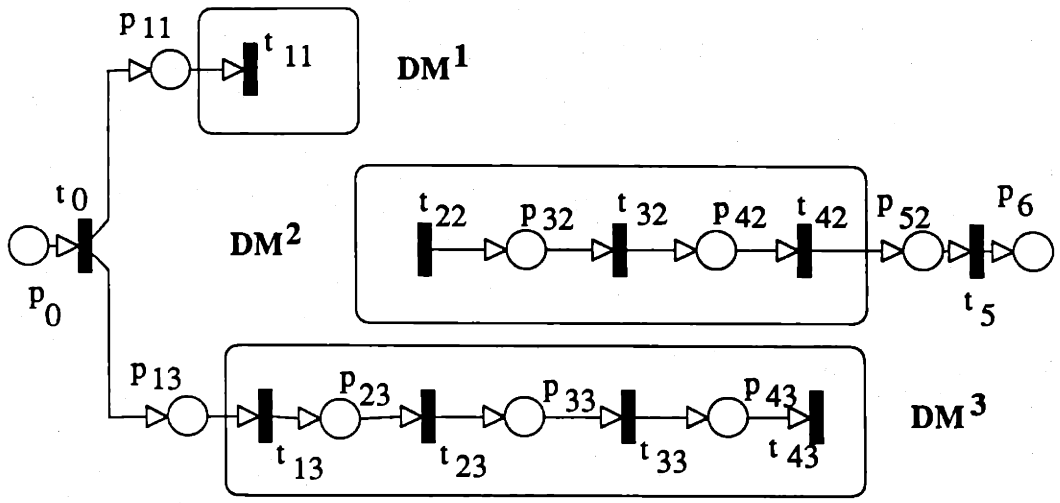


Figure 5.10: Decisionmakers with their boundaries.

The complete graph, obtained by incorporating the interactional places, is represented in Figure 5.11. Note that the graph is connected, but that t_{43} has no output place. This situation will be ruled out in Chapter VI when additional constraints are introduced.

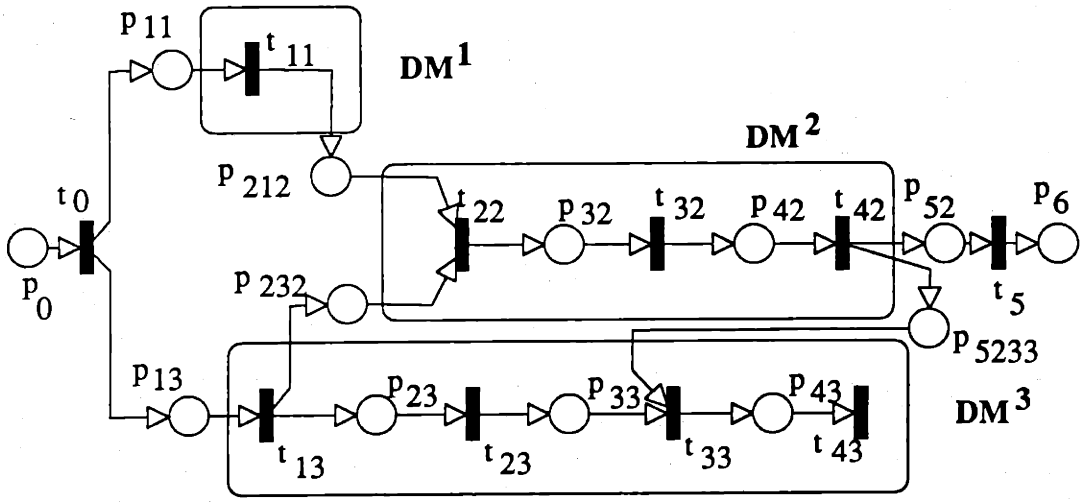


Figure 5.11: Complete graph of Π_3.

• (2) ⇒ (1) : the arrays e,s,F,G,H, and C can be easily retrieved from the incidence matrix or from the graph, provided that each transition is labeled to identify the decisionmaker it belongs to, as well as the stage it represents. The labeling technique described in section 5.1 has been chosen to achieve this result. The decisionmakers and their internal transitions are identified thanks to the labeling of the transitions. Borders can be drawn around the different decisionmakers and the internal places can then be distinguished from the interactional places. Once the interactional places are identified, the arrays e,s,F,G,H, and C can be constructed, since each interactional place correspond to a 1 in these arrays. As an illustration, Figure 5.12 gives the matrix representation of the WDN Π_3 of Figure 5.11. In Figure 5.11, there are six places located outside the boundaries of the decisionmakers (the source and the sink are note included). Π_3 has therefore six interactional

places - p_{11}, p_{13}, p_{212}, p_{232}, p_{52}, p_{5233} - yielding six non-zero elements in the arrays representing the net.

$$e = [\,1 \quad 0 \quad 1\,] \qquad F = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$s = [\,0 \quad 1 \quad 0\,] \qquad H = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 5.12: Matrix representation of Π_3.

The following implications have been proved: (1) ⇒ (3), (3) ⇒ (2), (2) ⇒ (1). The three representation are therefore equivalent in the sense that any one of them completely characterizes a WDN. It should however be noted that the three representations are not equivalent in terms of storage space (memory) they require. This point is just mentioned here and will be developed in Chapter VIII.

5.5 GENERIC PROPERTIES OF WDNs.

The following proposition gives a theoretical explanation to the fact that all the rows of the incidence matrix of a WDN (with the exception of the first and the last ones) have exactly one "-1" and one "1".

Proposition 5.1

Let the source and the sink places of the Petri Net representing a WDN be combined into a unique place, i.e. $p_0 \equiv p_6$. If the resulting Petri Net is strongly connected, it is an event graph.

The proof of Proposition 5.1 is straightforward. Each internal or interactional place of a WDN has exactly one input and one output transition. The sink of a WDN has one input but no output transitions, while the opposite stands for the source. If source and sink are merged into one place, every place in the net will therefore have one input and one output transition. Since the net is furthermore strongly connected, it is an event graph (see definition in section 2.3 of Chapter II).

The three theorems stated in section 2.5 of Chapter II will therefore apply to WDNs and will be used in the following chapters.

Note that considering the source and the sink of a WDN as the same place has no bearing on the internal topology of the net, which is the focus of this thesis. The assumption becomes however important when the dynamic behavior of a WDN is studied. The merging of source and sink limits indeed the amount of information a given organization can process simultaneously. The initial marking of the place representing the external environment will define this bound (see [5] for a detailed discussion of these issues). However, those considerations are not within the scope of this thesis.

CHAPTER VI

DECISIONMAKING ORGANIZATIONS

In Chapter IV, the notion of Well Defined Net (WDN) has been introduced to characterize the class of organizations under consideration in this thesis. While WDNs constitute the framework within which organizations will be designed, each WDN is not a valid organizational structure. This chapter defines additional constraints that will restrict and characterize the concepts of organizational form and organization.

6.1 DEFINITION OF THE CONSTRAINTS

The introduction of additional constraints to restrict the set of WDNs proceeds from two different considerations.

First, there are some WDNs corresponding to combinations of interactions between decisionmakers that do not make any sense. Those WDNs should be eliminated, if realistic organizational forms are to be generated. The **structural constraints** define what kinds of combinations of interactions need to be ruled out.

Second, any realistic design procedure should allow the designer to restrict the scope of the set of organizations he is considering for a specific problem. All possible allowable interactions between decisionmakers, as defined in the generic model, may not be of interest to the designer for a given application. **User-defined constraints** are introduced to address this issue.

As an important side effect of the introduction of constraints, the dimensionality of the problem will be reduced, thus enhancing its computational tractability.

6.1.1 Structural Constraints

Structural constraints refer to the set of conditions that any kind of organization must fulfill. They are contrasted to user-defined constraints which are a set of specific conditions defined by the organization designer for a particular application. Four different structural constraints are formulated that apply to all organizational structures being considered.

- (R_1) a) The structure should be connected, i.e., there should be at least one undirected path between any two nodes in the structure.
  b) A directed path should exist from the source to every node of the structure and from every node to the sink.

- (R_2) The structure should have no loops, i.e., the organizational structures are acyclical.

- (R_3) There can be at most one link from the RS stage of a DM to each one of the other DMs, i.e., for each i and j, only one element of the triplet $\{G_{ij},H_{ij},C_{ij}\}$ can be nonzero.

- (R_4) Information fusion can take place only at the IF and CI stages. Consequently, the SA stage of each DM can have only one input.

The set of structural constraints is defined as

$$R_s = \{R_{1a}, R_{1b}, R_2, R_3, R_4\} \tag{6.1}$$

The interpretation of the structural constraints is given below. The constraints R_{1a} and R_{1b} define connectivity as it pertains to this problem. Constraint R_{1a} corresponds to the notion of connectivity as presented in subsection 2.1.3. It eliminates structures that do not represent a single organization. Constraint R_{1b} insures that the flow of information is continuous within an organization. It eliminates internal input or output places (internal

sources or sinks). Since we are only considering organizations with a single source and a single sink, constraint R_{1b} implies constraint R_{1a}: if R_{1b} is satisfied any two nodes of the organization will be connected through the source or the sink of the organization and R_{1a} follows. Constraint R_{1a} has nevertheless been explicitly stated to accommodate the more general case where several output or several input places are present. If multiple sources or sinks are allowed, a connectivity constraint such as R_{1a} is necessary to ensure the unity of the organization.

Constraint R_2 allows acyclical organizations only. The acyclical hypothesis has been first formulated by Levis [1]. This restriction is made to avoid deadlock and 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. This point will be clarified in section 6.2 when the acyclical hypothesis will be translated into the language of Petri Net theory.

Constraint R_3 states that a decisionmaker can send the output of the RS stage to another given decisionmaker only once. It does not make much sense to send the same output to the same decisionmaker at several different stages. This restriction is therefore rather natural and reasonable. There is, however, the case where decisionmaker DM^i sends a command to decisionmaker DM^j as well as informs him of what his own decision is. This case is illustrated in Figure 6.1: decisionmaker DM^i sends his output to both the IF and CI stages of DM^j.

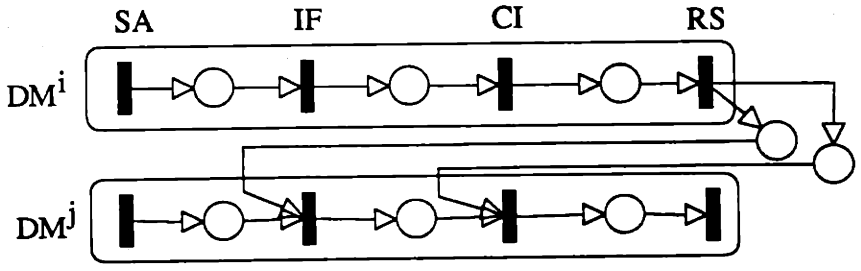


Figure 6.1: A violation of constraint R_3.

The problem is that the interactional structure represented in Figure 6.1 violates constraint R_3 and is therefore not allowed. The situation under consideration can, however, be modeled without violating R_3: a single link between the RS stage of DM^i and the IF stage of DM^j is sufficient. One can suppose that the link has two separate channels: one of them will transfer the command of DM^i to DM^j while the other will transfer the information concerning DM^i's decision. DM^j will then internally transfer the command from his own IF stage to his own CI stage without altering it.

Constraint R_4 prevents a decisionmaker from receiving more than one input at the SA stage. In fact, a decisionmaker can theoretically receive a maximum of n inputs - one from each other organization members and one from the external environment. The logic behind this limitation is that information cannot be merged at the SA stage. The IF stage has been specifically introduced to perform such a fusion. This condition may seem restrictive and arbitrary; it is meant to avoid the problem of multiple competing tasks arriving simultaneously. This case is outside the scope of this thesis. For the modeling of such a situation, see [26]. Subsection 6.1.5 will present a way of circumventing this restriction. In subsection 6.2.3, the effect of relaxing the constraint will be quantified.

Lastly, an implicit constraint has been omitted: the fact that a n-decisionmaker organization has exactly n members. This requirement has not been stated per se, because it is considered embedded in the very definition of a n-person organization. Attention should however be paid to it when it comes to computer implementation.

6.1.2 User-defined Constraints

The organization designer may want to introduce constraints that will reflect the specific application he is considering. For example, there may be a hierarchical relationship between the decisionmakers that must be maintained in the organizational structure. Therefore some links will be imposed and others will be ruled out.

These restrictions and specifications will be denoted **user-defined constraints**. They can be introduced in two different ways.

The organization designer can place the appropriate 0's and 1's in the arrays {e,s,F,G,H,C} defining a WDN. The other elements will remain unspecified and will constitute the degrees of freedom of the design. This type of constraints will be referred to as R_f.

To accommodate some very specific kind of interactions, the organization designer may want to create links between decisionmakers that are not modeled by the arrays mentioned above. In other words, those links are not among the allowable interactions presented in Figure 4.5. The links are, however, fixed and therefore do not increase the dimensionality of the design problem. They will be referred to as special constraints and denoted R_p.

The rationale behind the introduction of special constraints is the following. The generic model of a WDN presented in Chapter IV results from a trade-off between explanatory power and computational tractability. Some interactional links between decisionmakers have been ruled out from the generic case because they present little physical relevance, while substantially increasing the dimensionality of the problem. The designer may however absolutely need one of those links to handle a specific application. Special constraints have been introduced to address this need. Because they are not part of the generic model of a WDN, they do not increase the dimensionality of the design problem. They just introduce more flexibility.

The set of user-defined constraints will be denoted R_u, while the complete set of constraints will be denoted R.

$$R_u = R_f \cup R_p$$

$$R = R_u \cup R_s \qquad (6.2)$$

6.1.3 Well Defined Structure

By **Well Defined Structure** (WDS) is meant the association of a Well Defined Net (WDN) with a set of special constraints R_p:

$$WDS \equiv (WDN, R_p)$$

In the sequel, the set of special constraints will always be given and will therefore be fixed throughout the design procedure. When the set of special constraints is fixed, there is an isomorphism between the set of WDSs and the set of WDNs. It is trivially defined by

$$WDS \equiv (WDN, R_p) \rightarrow WDN$$

The set of WDSs associated with given structural constraints will thus be identified with the set of WDNs. The same notation Ψ^n will be used.

6.1.4 Terminology

Admissible Organizational Form

A WDS that fulfills the set of user-defined constraints R_u will be called an **Admissible Organizational Form** (AOF). The set of all AOFs will be denoted $\Phi(R_u)$.

Feasible organization

An AOF that fulfills the set of constraints R_s will be called a **Feasible Organization** (FO). Note that a Feasible Organization is a WDS that fulfills the complete set of constraints R. The set of all Feasible Organizations will be denoted $\Phi(R)$.

To avoid cumbersome notation, the dimension n - the number of decisionmakers - has been omitted. If the set of special constraints is given, the following inclusions hold.

$$\Psi^n \supset \Phi(R_u) \supset \Phi(R) \qquad (6.3)$$

6.1.5 Conflict Among the Constraints.

In the general case, no conflict is allowed between the structural and the user-defined constraints. In other words, the designer cannot overrule the structural constraints with user-defined constraints. If he does so, the set of Feasible Organizations will be empty. As an example, the user cannot specify that both C_{ij} and C_{ji} (with $i \neq j$) be equal to 1. This

would indeed create a loop in the organization thus violating constraint R_2.

There is a single exception to the generic rule stated above. This exception has been introduced to alleviate the somewhat arbitrary restriction imposed by constraint R_4, which limits the number of inputs a decisionmaker can receive at the SA stage. The exception is the following: R_4 will not apply to the special constraints. In other words, a decisionmaker can have more than one input at his SA stage, provided that all those inputs but at most one be special constraints. An illustrative example where this exception applies is given in subsection 6.3.1.

6.2 MATHEMATICAL REPRESENTATION OF THE CONSTRAINTS

In this section, the constraints previously introduced are reviewed and analytically characterized.

6.2.1 Structural Constraints

Constraint R_1

The connectivity requirement can be reformulated as follows. If we merge the source and the sink of an organization together, i.e., if we represent the external environment with a single place connected to both ends of the organization, then constraints R_{1a} and R_{1b} can be aggregated into R_1':

(R_1') The Petri Net representing a structure should be strongly connected.

Proposition 5.1 of Chapter V then induces the following proposition.

Proposition 6.1

A WDN whose sink and source are merged into a single place and which fulfills constraint R_1 is an event graph.

Constraint R_2

Let us suppose that R_1 is fulfilled and that source and sink are merged into a single place. This place will be referred to as the **external place**. The acyclical assumption states that no internal loop - or internal directed circuit - is allowed within the organization. By **internal loop** is meant a directed circuit that does not include the external place. The next paragraph makes this point explicit.

We will only consider directed elementary circuits, i.e., directed circuits in which no node appears more than once (see 2.1.3). A directed elementary circuit which contains the external place among its nodes will be called a **simple path**. A simple path is therefore a loop going from the external place back to itself. If the external place is partitioned into source and sink, a simple path becomes a directed path between the source and the sink of the organization. Conversely, a directed elementary circuit which does not contain the external place will be called an **internal loop**. Constraint R_2 rules out internal loops. In other words, constraint R_2 can be stated as follows:

(R_2') All directed elementary circuits of the Petri Net representing the structure should be simple paths.

Constraint R_3

The mathematical translation of this constraint is straightforward and is given below without further comments.

$$\forall (i,j) \in [1, n]^2 \quad G_{ij} + H_{ij} + C_{ij} \leq 1 \tag{6.4}$$

Constraint R_4

Like R_3, the translation of R_4 is straightforward.

$$\forall j \in [1, n] \qquad e_j + \sum_{1 \leq i \leq n} G_{ij} \leq 1 \tag{6.5}$$

6.2.2 User-defined Constraints

Constraints R_f

As mentioned in section 6.1.2, the constraints R_f are defined by assigning the value 0 or 1 to some elements of the arrays $\underline{e}$, $\underline{s}$, F, G, H, and C. This is already an analytic characterization and no further development is needed.

Constraints R_p

A special constraint is an interactional link between two different decisionmakers, that cannot be represented in a WDN. Such a link will be characterized by its input and output transitions. It will thus be designated by the pair (t_{ki}, t_{mj}). t_{ki} is the input transition - k refers to the stage and i to the decisionmaker- while t_{mj} is the output transition - m is for the stage, j for the decisionmaker. The following restrictions apply:

- $i \neq j$: the two decisionmakers should be different.
- If k=1 then $m \neq 2$: a link between SA and IF can be represented in a WDN.
- If k=4 then m=4 : links between RS and SA,IF, or CI can be represented in a WDN.

In the Petri Net representation, each special constraint will be represented by an interactional place. The labeling of this place will be determined from its input and output transitions as follows.

(t_{ki}, t_{li}) will correspond to $p_{k+1,ijl}$.

All properties of WDNs will apply to WDSs. For example, the construction of the incidence matrix of a WDS will follow the rules stated in subsection 5.4.2 for WDNs.

6.2.3 Reduction in the Dimensionality

The introduction of constraints significantly reduces the dimension of the design

problem. The exact determination of the reduction coefficient will however not be undertaken for the following reasons. First, the reduction effects of the structural constraints R_1 and R_2 are difficult to evaluate in the general case. Internal loops within an organization may be very intricate and the implication of connectivity is not easy to evaluate quantitatively. Going into sophisticated combinatorial analysis would be rather cumbersome and would bring little supplementary insight. Second, the most drastic reduction in dimensionality originates from the user-defined constraints and is therefore entirely problem dependent.

The reduction in dimensionality induced by the constraints R_3 and R_4, when taken separately, is nevertheless given below. The reduction coefficients are easy to compute in these cases and the effect of relaxing the constraints will be quantified.

Constraint R_3

Since Gij + Hij + Cij ≤ 1 for all i and j, the number of degrees of freedom in the matrices G,H, and C shrinks from $3n^2$-3n to $2n^2$-2n. The dimension has therefore been divided by the reduction coefficient

$$\rho_3 = (2^{n-1})^n = 2^{20} \approx 10^6 \text{ for } n = 5.$$

Constraint R_4

Without constraint R_4, the dimension of the set of all allowable arrays e,G is 2^{n*n}. With the constraint, this dimension is reduced to $(n+1)^n$. The reduction coefficient is therefore

$$\rho_4 = (2^n/(n+1))^n.$$

Relaxing constraint R_4 would therefore significantly increase the dimension of the problem. As an example, n=5 yields

$$\rho_4 = (16/5)^5 \approx 336.$$

6.3 ON THE SELECTION OF USER-DEFINED CONSTRAINTS

The next chapter will present an automatic procedure to generate all possible Feasible Organizations fulfilling a given set of constraints R. It will be assumed that the user-defined constraints R_u are given. The goal of this section is to address the problem of selecting the user-defined constraints. The challenge is to find a set of constraints R_u that will reflect, as best as possible, the reality of the situation, without leaving an unmanageable number of degrees of freedom.

The best way to present the actual reasoning method that will reduce a real application into the specification format used in this thesis, is to go through an example. Although an application example will be thoroughly studied in Chapter IX, the reader may find it useful to see at this point how the model so far developed can be applied.

Example: The Ship Control System of a Submarine.

This example has been developed and analyzed by Weingaertner [19]. The situation under consideration pertains to the ship control system of a submarine. A crew of five members is in charge of this task and their roles and functional relationships are described below.

At the top of the hierarchy is the Officer of the Deck (DM^1) with responsibility for all ship control matters pertaining to the conduct of the submarine mission. He receives information both from the external environment and from the Diving Officer of the Watch (DM^2). He issues command to DM^2.

The Diving Officer of the Watch is responsible for the bulk of the control decision process. He receives information from and sends information to the remaining members of the organization: the Chief of the Watch (DM^3), the Lee Helm (DM^4), the Helm (DM^5). DM^3, DM^4, and DM^5 can be considered the sensors and the actuators of the organization. They receive information from the external environment (ship control panels,...) and can act on the external environment (stern planes, fairwater planes,...).

From the above description of the situation, it appears that two different kinds of real life constraints are imposed on the organization: physical and hierarchical constraints.

Physical constraints are due to those physical devices that organization members use to interact with the external environment. The Helm (DM^4) and the Lee Helm (DM^5) actually sit in front of the ship control panels. They directly receive information from the external environment of the submarine. Note that the other decisionmakers may also receive information from the external environment. Nothing, for instance, prevents the Officer of the Deck to look at the control screens above the Helm's shoulders. This is however not a constraint and should be left as a degree of freedom in the design. A different set of hardware constraints applies to the other end of the organization, where actions are taken upon the external environment. Decisionmakers DM^3, DM^4 and DM^5 have actual physical control over the ship. They directly interact with the external environment. On the contrary, DM^1 and DM^2 have no such direct control. These hardware constraints are reflected in the specification of the arrays $\underline{e}$ and $\underline{s}$.

$$e = [\, x\ x\ x\ 1\ 1 \,] \qquad s = [\, 0\ 0\ 1\ 1\ 1 \,]$$

A "x" denotes that this link has not been specified, i.e. a degree of freedom.

Hierarchical constraints refer to the underlying hierarchy existing between the different organization members. Subordinate members (DM^4 and DM^5) will communicate their situation assessments to superordinate members (DM^1 and DM^2) who in turn will issue commands to the former. DM^3 has an intermediate status. This hierarchical structure is reflected, as follows, in the matrices F and C representing respectively the sharing of situation assessments and the issuance of commands.

$$F = \begin{bmatrix} \# & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 \\ x & x & x & \# & x \\ x & x & x & x & \# \end{bmatrix} \qquad C = \begin{bmatrix} \# & 1 & x & x & x \\ 0 & \# & 1 & 1 & 1 \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & \# \end{bmatrix}$$

The symbol "#" indicates that the designer has no control over the diagonal elements of the matrices (see 4.4.1).

The matrix G will be identically zero since no decisionmaker is sending his output as an input to another organization member.

A result sharing type of interaction may nevertheless exist between DM^4 and DM^5. This will be reflected in the matrix H.

$$G = \begin{bmatrix} \# & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & \# \end{bmatrix} \qquad H = \begin{bmatrix} \# & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & \# & x \\ 0 & 0 & 0 & x & \# \end{bmatrix}$$

Last of all, the situation as described by Weingaertner [19], requires the use of special constraints. First, the Diving Officer of the Watch (DM^2) will use the situation assessment of the Chief of the Watch (DM^3) to formulate his own situation assessment. To model this case, a link is necessary between the SA stages of both decisionmakers. Second, the Officer of the Deck (DM^1) will merge the result coming from the IF stage of DM^2 with the information he receives from other members. This interaction will be translated by a link between the IF stages of DM^2 and DM^1. We will therefore have two special constraints:

(t_{22}, t_{21}) : from IF of DM^2 to IF of DM^1.

(t_{13}, t_{12}) : from SA of DM^3 to SA of DM^2.

Note that there are 16 degrees of freedom left.

The Petri Net Π_4 representing the WDS defined above is given in Figure 6.2. Boldface connectors and places correspond to the 1's of the arrays e,s,F,G,H,C and to the special constraints. Regular connectors and places correspond to the unspecified elements of the same arrays (the x's).

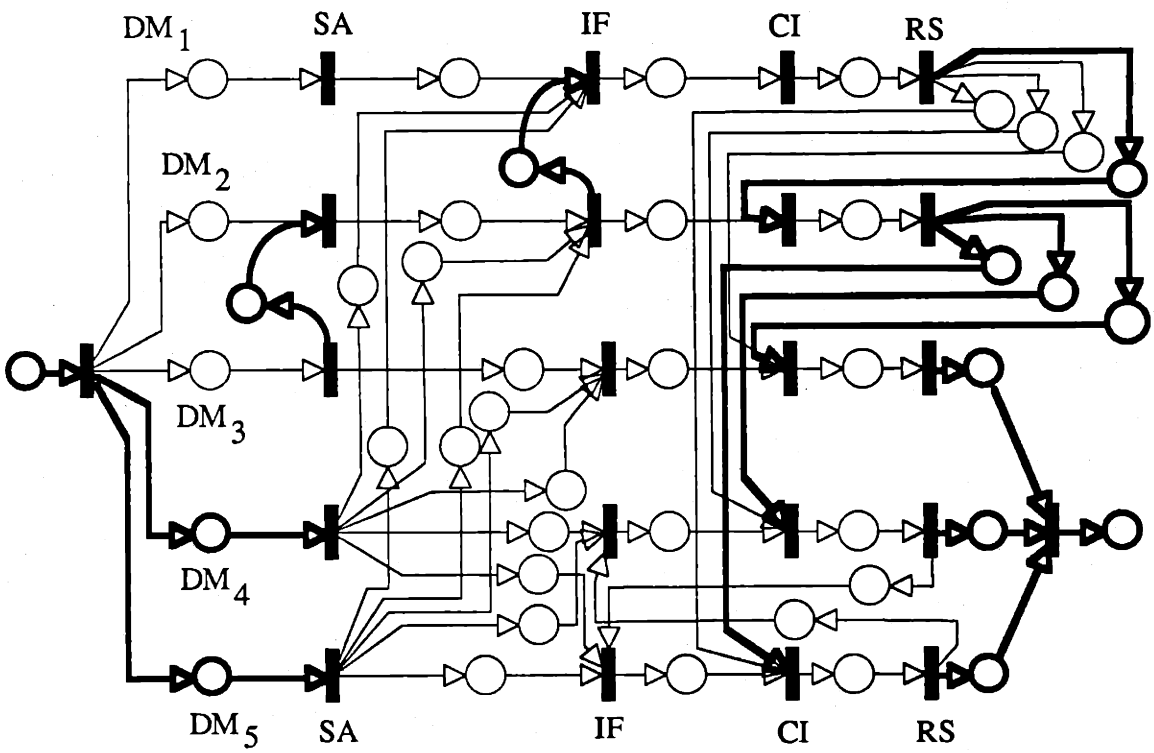


Figure 6.2: Petri Net Π_4.

A few comments can be made on the Petri Net Π_4. Note first that the SA stage of DM_2 has two inputs. This however does not violate constraint R_4 because one of the input is a special constraint (see 6.1.5). A violation of the structural constraints would occur if both result sharing type of links between DM^4 and DM^5 were implemented simultaneously: a loop would indeed be created. Lastly, it should be emphasized that the introduction of a special constraint between the SA stages of DM^2 and DM^3 could have been avoided. Replacing the special constraint by a link between the SA stage of DM^3 and the IF stage of DM^2 would have induced little change in the topological structure of the net. The same remark does not apply however to the other special constraint between the IF stages of DM^2 and DM^3. Replacing this interaction by a link from the SA stage of DM^2 to the IF stage of DM^1 would have eliminated numerous information paths, while replacing it with a link originating from the RS stage of DM_2 would have created a loop.

CHAPTER VII

CHARACTERIZATION OF FEASIBLE ORGANIZATIONS

In the previous chapter, the concept of Feasible Organization (FO) has been defined: a FO is a Well Defined Structure (WDS) that satisfies both the structural and the user-defined constraints. The design problem is to determine the set of all Feasible Organizations corresponding to a specific set of constraints. It is assumed throughout this chapter that the user-defined constraints R_u are given.

7.1 MAXIMALLY AND MINIMALLY CONNECTED ORGANIZATIONS

Since the set of special constraints is given, the notions of WDN and WDS are interchangeable (6.1.3). From now on, the term WDS only will be used. In Chapter IV, an order is defined on the set Ψ^n of all WDSs. It is shown that Ψ^n, associated with this order, is a lattice (Proposition 4.3). The set of all Feasible Organizations $\Phi(R)$ is a subset of Ψ^n and is therefore also partially ordered (Theorem 3.1). From Theorem 3.2, we conclude that $\Phi(R)$ has at least one minimal and one maximal elements.

Definition

A maximal element of the set $\Phi(R)$ of all Feasible Organizations will be called a **Maximally Connected Organization** (MAXO).

Similarly, a minimal element of $\Phi(R)$ will be called a **Minimally Connected Organization** (MINO).

The set of all MAXOs (resp. MINOs) will be denoted $\Phi_{max}(R)$ (resp. $\Phi_{min}(R)$).

Maximally and minimally connected organizations can be interpreted as follows. It was shown in Chapter IV (4.4.3) that the size of a WDS is its number of links. By

definition, a maximal element has no immediate superordinate: its size is thus maximal in the sense that no elements with a higher size covers it. We have, therefore, the following interpretation of a MAXO.

A MAXO is a WDS such that it is not possible to add a single link (i.e. to increase the size) without violating the set of constraints R (i.e. without crossing the boundaries of the subset $\Phi(R)$). Similarly, a MINO is a WDS such that it is not possible to remove a single link (i.e. to decrease the size) without violating the set of constraints R.

By the definition of minimality and maximality, every element of a subset Y of a partially ordered set X is bounded by at least one minimal element and at least one maximal element of Y. We have, therefore, the following result.

Proposition 7.1

For any given Feasible Organization Π, there is at least one MINO Π_{min} and at least one MAXO Π_{max} such that

$$\Pi_{min} \leq \Pi \leq \Pi_{max}$$

Alternatively,

$$\{\Pi \in \Psi^n / \exists(\Pi_{min},\Pi_{max}) \in \Phi_{min}(R)\times\Phi_{max}(R) \quad \Pi_{min} \leq \Pi \leq \Pi_{max}\} \supset \Phi(R)$$

Note that the previous inclusion is not an equality in the general case. There is indeed no guarantee that a WDS located between a MAXO and a MINO will fulfill the constraints R. To address this problem, we need to study the behavior of the order "$\leq$" with respect to the constraints R. The notion of convexity is introduced for that purpose in the next section.

7.2 CONVEXITY

7.2.1 Definitions

Interval

Let us consider a partially ordered set X. If x_1 and x_2 are two elements of X satisfying $x_1 \leq x_2$, the **interval** $[x_1,x_2]$ is defined to be $\{x \in X \mid x_1 \leq x \leq x_2\}$ [27].

Convex subset

Let Y be a subset of the partially ordered set X. The subset Y will be **convex** [27] if and only if the following implication holds:

$$(\forall (y_1,y_2) \in Y^2) \ (y_1 \leq y_2) \ \Rightarrow \ (Y \supset [y_1,y_2])$$

7.2.2 Convexity of a Property

Let X be a partially ordered set. By **property** defined on X, is meant a set of conditions P that an element of X may or may not fulfill. The key point is the binary nature of P: for every element x of X, the property P is either true or false. If x fulfills P we will write P[x]=1. If x violates P, we will write P[x]=0. One may think of a property as a binary mapping from X to the set {0,1} (0 for false and 1 for true). Let K(P) be the set of all elements of X fulfilling the property P:

$$K(P) = \{x \in X \mid P[x]=1 \}.$$

Let $K_{max}(P)$ (resp. $K_{min}(P)$) denote the set of all maximal (resp. minimal) elements of K(P).

The notion of convexity introduced in 7.2.1 for a subset of X, can be extended to a property P defined on X as follows.

Definition

A property P defined on X will be **convex** if and only if every element x of X located between two elements x_1 and x_2 that satisfy P, will also satisfy P:

$$(\forall\ (x_1,x_2) \in X^2)\quad (P[x_1] = P[x_2] = 1 \text{ and } x_1 \leq x_2) \Rightarrow (\forall\ x \in [x_1,x_2]\ \ P[x] = 1))$$

The notion of convexity of a property is closely related to the notion of convexity of a subset. The previous definition can, indeed, be rephrased as follows: a property P is convex if and only if the subset K(P) of all elements of X satisfying P is convex. The notion of convexity is introduced to help us characterize the set K(P) of all elements satisfying the property P. The following proposition achieves this goal; it is a direct consequence of the definition of the convexity.

Proposition 7.2

If P is convex on X, K(P) is characterized by its minimal and maximal elements as follows:

$$K(P) = \{x \in X \mid \exists\ (a,b) \in K_{min}(P) \times K_{max}(P)\ \ a \leq x \leq b\}$$

7.2.3 Convexity of the Constraints

The constraints R are properties defined on the set Ψ^n since a constraint is either satisfied or violated by a given WDS. We can therefore apply the concept of convexity defined in the previous subsection to the different constraints R.

The advantage of having convex constraints is becoming clear now. Suppose that all constraints R are convex. Then an element Π of Ψ^n located between a MINO and a MAXO would necessarily fulfill the constraints R. The set $\Phi(R)$ would therefore be characterized by the sets of MAXOs and MINOs (Proposition 7.2). Unfortunately, we are not in the simple case where all constraints are convex. As mentioned in section 7.1, there is no guarantee that a WDS located between a MAXO and a MINO will satisfy the structural constraints. The reason why Proposition 7.2 does not apply is that the constraint R_1 is not convex. However, R_1 is the only structural constraint that poses a problem as the following proposition attests.

Proposition 7.3

The constraints R_2, R_3, R_4 defined on the set Ψ^n are convex.

The proof of the proposition is direct. Let us consider the constraint R_2. If a WDS is acyclical, i.e. fulfills R_2, then any WDS obtained by removing links from the initial WDS will also be acyclical. Loops cannot be created in a loop-free structure by removing links. The same argument applies to the constraints R_3 and R_4.

The difficulty with the constraint R_1 is that one can break the connectivity of a structure by removing a link (this should be obvious), but also by **adding a link**. Indeed, adding a link that originates from a transition of the current net but does not terminate at another transition of the net will create a transition without output place, and will thus violate R_1. Before this point is illustrated on an example, the notion of link need be defined without ambiguity.

According to the language of Chapter IV, a link in a WDS refers to a 1 in the arrays representing the structure, i.e., to an interaction between two stages of two different decisionmakers or between a decisionmaker and the external environment. There is, therefore, a one to one correspondence between links and *interactional* places. Up to now the word link has been exclusively used with this meaning and it will keep this meaning throughout the thesis. A link will, therefore, always cross the boundary of at least one decisionmaker: an **internal connection** between two stages of the same decisionmaker is **not a link**. Lastly, an another possible confusion need be avoided. In the Petri Net representation of a WDS, an interactional place is related to its input and output transitions by two connectors. Those connectors should not be mistaken for links: it is the association of the interactional place and the two connectors that will constitute a link, not the connectors themselves.

Let us now illustrate on an example the non-convexity of the constraint R_1.

The matrix representation of a 2-dimensional WDS Π_1 that satisfies the constraint R_1, is given below.

$$e_1 = [\ 0\ 1\ ] \qquad F_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad G_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$s_1 = [\ 0\ 1\ ] \qquad H_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad C_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The Petri Net representation of Π_1 is reproduced in Figure 7.1. Let us add to the WDS Π_1, a link between the external environment and the SA stage of DM^1: the vector e_1 is replaced by the vector $e_2 = [\ 1\ 1\ ]$. The resulting structure Π_2 has an internal transition without output place and therefore violates the connectivity constraint. Lastly, another link is added from the SA stage of DM^1 to the IF stage of DM^2 yielding the WDS Π_3. The WDS Π_3 now fulfills the connectivity constraint R_1. The situation is, therefore, the following:

$$\Pi_1 \leq \Pi_2 \leq \Pi_3 ,$$

the nets Π_1 and Π_3 satisfy R_1 but Π_2 does not: the constraint R_1 is not convex. Note that to go directly from Π_1 to Π_3 one needs to add two links. In doing so, a new simple path is created in the WDS: one can go from Π_1 to Π_3 by adding a simple path to Π_1. This consideration will lead us to use the notion of simple path instead of link, as the incremental unit leading from a WDS to its immediate superordinate. In replacing links by simple paths, the difficulty posed by R_1 vanishes. This idea is developed and defined precisely in section 7.4.

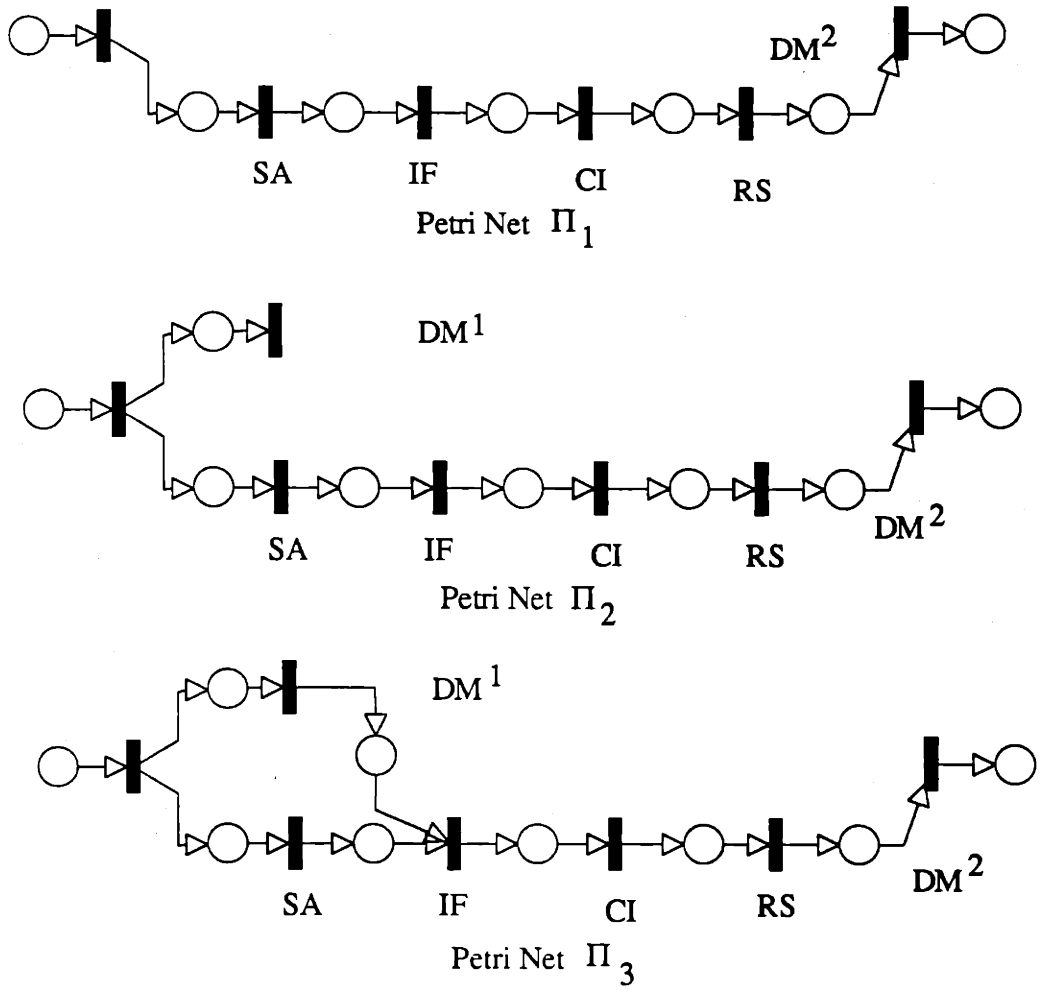


Figure 7.1: Example of the non-convexity of the constraint R_1.

We have concentrated so far only on structural constraints. As far as the user-defined constraints R_u are concerned, the situation is fairly simple. The user-defined constraints are convex and the set $\Phi(R_u)$ can, therefore, be characterized by its minimal and maximal elements, using Proposition 7.2.

The above discussion provides us the line of reasoning that we will follow in this chapter. In section 7.3, the scope of the problem will be reduced by considering the set $\Phi(R_u)$ of all Admissible Organizational Forms only. This set will be characterized using Proposition 7.2. In section 7.4, we will then use the concept of simple path to eliminate the problem posed by the constraint R_1 and obtain a characterization of $\Phi(R)$, the ultimate goal of this chapter.

7.3 CHARACTERIZATION OF $\Phi(R_u)$.

7.3.1 Universal Net

In this section, we concentrate on the user-defined constraints R_u and, more precisely, on the constraint R_f. As mentioned in 6.1.2, R_f is defined by the organization designer who specifies some elements of the arrays $\underline{e}$,$\underline{s}$,F,G,H, and C, leaving the remaining elements undetermined. A given element of $\Phi(R_u)$ will be characterized by the specification of all the undetermined elements. If there are m undetermined elements, i.e., if there are m degrees of freedom left in the design, $\Phi(R_u)$ will be isomorphic to $\{0,1\}^m$ and will therefore have 2^m elements.

Definition

> The **Universal Net** associated with the constraints R_u - $\Omega(R_u)$ - is the WDS obtained by replacing all undetermined elements of $\underline{e}$, $\underline{s}$, F, G, H, and C by 1. Similarly the **Kernel Net** - $\omega(R_u)$ - is the WDS obtained by replacing the same undetermined elements by 0.

It is easy to see that $\omega(R_u)$ is the **least element** of $\Phi(R_u)$, while $\Omega(R_u)$ is the **greatest element** of $\Phi(R_u)$. Using the notation of section 7.1, we have:

$$\Phi_{max}(R_u) = \{ \Omega(R_u) \} \qquad \text{and} \qquad \Phi_{min}(R_u) = \{ \omega(R_u) \}$$

Proposition 7.4 summarizes the analysis of this subsection and provides a characterization of the set $\Phi(R_u)$ of all Admissible Organizational Forms.

Proposition 7.4

The set $\Phi(R_u)$ is the subset of Ψ^n that satisfies the two following conditions:

- any element of $\Phi(R_u)$ is a subnet of the Universal Net $\Omega(R_u)$.
- the Kernel Net $\omega(R_u)$ is a subnet of any element of $\Phi(R_u)$.

Alternatively,

$$\Phi(R_u) = \{ \Pi \in \Psi^n / \omega(R_u) \leq \Pi \leq \Omega(R_u) \} = [\omega(R_u), \Omega(R_u)]$$

Proof

Proposition 7.4 is a direct consequence of Proposition 7.2. Since the user-defined constraint R_u is convex, Proposition 7.1 applies. The proof is completed, if we note that $\Phi(R_u)$ has a single maximal element (the Universal Net) and a single minimal element (the Kernel Net).

Corollary

$\Phi(R_u)$ is a sublattice of Ψ^n.

Proof

Let Π and Π' be two elements of $\Phi(R_u)$. According to Proposition 7.4, $\Omega(R_u)$ is an upper bound to Π and Π'. The lowest upper bound (l.u.b.) of Π and Π' - which exists since Ψ^n is a lattice - is therefore necessarily smaller than or equal to $\Omega(R_u)$. Applying Proposition 7.4 again, we conclude that the l.u.b. of Π and Π' is an element of $\Phi(R_u)$. The reasoning is completely symmetrical to prove that the g.l.b. of Π and Π' is also an element of $\Phi(R_u)$. QED.

It is important to understand the fundamental difference between Proposition 7.4 and Proposition 7.1. Proposition 7.4 gives a characterization of the set $\Phi(R_u)$ of all FOFs: to be located between the Kernel and the Universal Nets is a *necessary and sufficient* condition for a WDS to be an element of $\Phi(R_u)$. The set $\Phi(R_u)$ is therefore completely defined once

$\omega(R_u)$ and $\Omega(R_u)$ are known. Conversely, Proposition 7.1 does not characterize the set $\Phi(R)$ of all FOs. It just gives a *necessary* condition that a WDS must satisfy to be a FO. To be located between a MINO and a MAXO is *not sufficient* for a WDS to be a FO. As pointed out earlier, the difference in nature between the sets $\Phi(R_u)$ and $\Phi(R)$ arises from the fact that R_u is stable by interval while R is not. The goal of the remaining part of this chapter is to find a necessary and sufficient condition characterizing a Feasible Organization.

7.3.2 Simple Paths of the Universal Net

The notion of simple path has been introduced in 6.2.1. Given the importance of the concept, the definition is recalled below.

Simple path

> Let Π be a WDS whose source and sink have been merged together into a single external place. A **simple path** of Π is a directed elementary circuit which includes the external place.

According to Proposition 5.1 of Chapter V, the Petri Net representing Π is an event-graph. We can therefore use Theorem 2.5 of Chapter II to find all the simple paths of Π: a simple path is a minimal support S-invariant of Π whose component corresponding to the external place is equal to 1. Note that if the latter property is not satisfied, the S-invariant is an internal loop of the net. Subsection 2.4.1 of Chapter II defines the concept of S-component associated with an S-invariant: it is the subnet of the initial Petri Net whose places are exactly the places of the support of the S-invariant. An S-component of a WDS is therefore itself a WDS. Consequently, the simple paths of a given WDS are themselves WDSs. We will denote by $Sp(R_u)$ the set of all simple paths of the Universal Net $\Omega(R_u)$. We will write

$$Sp(Ru) = \{sp_1, ..., sp_r\}.$$

The sp_i $(1 \leq i \leq r)$ are WDSs satisfying $sp_i \leq \Omega(R_u)$.

Before proceeding to the next section where the main result of this chapter is presented, we will illustrate the concept of simple path and its link with minimal support S-invariants with an example.

7.3.3 Example

We will consider the Petri Net Π_1 presented in Figure 5.3 of Chapter V. The incidence matrix Δ_1 of Π_1 is recalled below.

$$\Delta_1 = \begin{bmatrix}
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

If sink and source are merged together, the first and the last rows of Δ_1 are replaced by the sum of the two rows. This yields the matrix Δ_{1b} presented below. The first row accounts for the external place.

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

Let us find the S-invariants of Δ_{1b}. An S-invariant will be a 17-dimensional integer vector $X=(x_i)$, with $1 \leq i \leq 17$, which satisfies:

$$\Delta_{1b}{}^{T} . X = 0.$$

The previous equation yields the following system.

$$\begin{array}{l}
-x_1+x_2+x_3 = 0 \\
-x_2+x_4+x_5 = 0 \\
-x_3+x_6+x_7 = 0 \\
-x_4+x_8 = 0 \\
-x_5-x_6+x_9-x_{14} = 0 \\
-x_7+x_{10} = 0 \\
-x_8+x_{11} = 0 \\
-x_9+x_{12} = 0 \\
-x_{10}+x_{13}-x_{16} = 0 \\
-x_{11}+x_{14} = 0 \\
-x_{12}+x_{15}+x_{16} = 0 \\
-x_{13}+x_{17} = 0 \\
x_1-x_{15}-x_{17} = 0
\end{array}$$

Note that the labeling of the places and transitions of Π_1 is such that the above system is quasi-diagonal, which makes its solution fairly simple. The rank of the system is 5. We will choose x_1, x_2, x_4, x_6, and x_{13} as independent variables. The system becomes:

$$\begin{array}{l}
x_3 = x_1 - x_2 \\
x_5 = x_2 - x_4 \\
x_7 = x_1 - x_2 - x_6 \\
x_8 = x_4 \\
x_9 = x_2 + x_6 \\
x_{10} = x_1 - x_2 - x_6 \\
x_{11} = x_4 \\
x_{12} = x_2 + x_6 \\
x_{14} = x_4 \\
x_{15} = x_1 - x_{13} \\
x_{16} = -x_1 + x_2 + x_6 + x_{13} \\
x_{17} = x_{13}
\end{array}$$

We can therefore write any element X of the kernel of Δ_{1b} as follows:

$X = x_1.X_1 + x_2.X_2 + x_4.X_4 + x_6.X_6 + x_{13}.X_{13}$, where the vectors X_1, X_2, X_4, X_6, and X_{13} are given in Table 7.1 below.

TABLE 7.1 BASE OF THE KERNEL OF Δ_{1b}.

	X_1	X_2	X_4	X_6	X_{13}
x_1	1	0	0	0	0
x_2	0	1	0	0	0
x_3	1	-1	0	0	0
x_4	0	0	1	0	0
x_5	0	1	-1	0	0
x_6	0	0	0	1	0
x_7	1	-1	0	-1	0
x_8	0	0	1	0	0
x_9	0	1	0	1	0
x_{10}	1	-1	0	-1	0
x_{11}	0	0	1	0	0
x_{12}	0	1	0	1	0
x_{13}	0	0	0	0	1
x_{14}	0	0	1	0	0
x_{15}	1	0	0	0	-1
x_{16}	-1	1	0	1	1
x_{17}	0	0	0	0	1

The family $\{X_1, X_2, X_4, X_6, X_{13}\}$ is a base of the kernel of the matrix Δ_{1b}. The dimension of the kernel is therefore 5. However, those vectors are not S-invariants since their components are not all positive. There are seven minimal support S-invariants, obtained as follows from the vectors X_i.

$$X_{s1} = X_1 + X_2$$

$$X_{s2} = X_1 + X_2 + X_4$$

$$X_{s3} = X_1 + X_2 + X_4 + X_{13}$$

$$X_{s4} = X_1 + X_2 + X_{13}$$

$$X_{s5} = X_1 + X_6$$

$$X_{s6} = X_1 + X_6 + X_{13}$$

$$X_{s7} = X_1 + X_{13}$$

The family $\{X_{s1},X_{s2},X_{s3},X_{s4},X_{s5},X_{s6},X_{s7}\}$ is a base of the set of all S-invariants of Δ_{1b} (2.1.1). Any S-invariant of Δ_{1b} will be obtained as a linear combination with positive coefficient of the vectors X_{si} (Theorem 2.4). Table 7.2 shows the six minimal support S-invariants by giving the places of their support.

TABLE 7.2 MINIMAL SUPPORT S-INVARIANTS OF Π_1.

		X_{s1}	X_{s2}	X_{s3}	X_{s4}	X_{s5}	X_{s6}	X_{s7}
x_1	p_0 or p_6	1	1	1	1	1	1	1
x_2	p_{12}	1	1	1	1	0	0	0
x_3	p_{13}	0	0	0	0	1	1	1
x_4	p_{221}	0	1	1	0	0	0	0
x_5	p_{222}	1	0	0	1	0	0	0
x_6	p_{232}	0	0	0	0	1	1	0
x_7	p_{233}	0	0	0	0	0	0	1
x_8	p_{31}	0	1	1	0	0	0	0
x_9	p_{32}	1	1	1	1	1	1	0
x_{10}	p_{33}	0	0	0	0	0	0	1
x_{11}	p_{41}	0	1	1	0	0	0	0
x_{12}	p_{42}	1	1	1	1	1	1	0
x_{13}	p_{43}	0	0	1	1	0	1	1
x_{14}	p_{5122}	0	1	1	0	0	0	0
x_{15}	p_{522}	1	1	0	0	1	0	0
x_{16}	p_{5233}	0	0	1	1	0	1	0
x_{17}	p_{533}	0	0	1	1	0	1	1

The labeling of the places corresponds to Figure 5.3.3 of chapter V.Note that the component of the external place p_0 is positive for all invariants : Π_1 is therefore acyclical. The S-invariants X_{si} are all simple paths of Π_1. Figure 7.2 shows those simple paths.

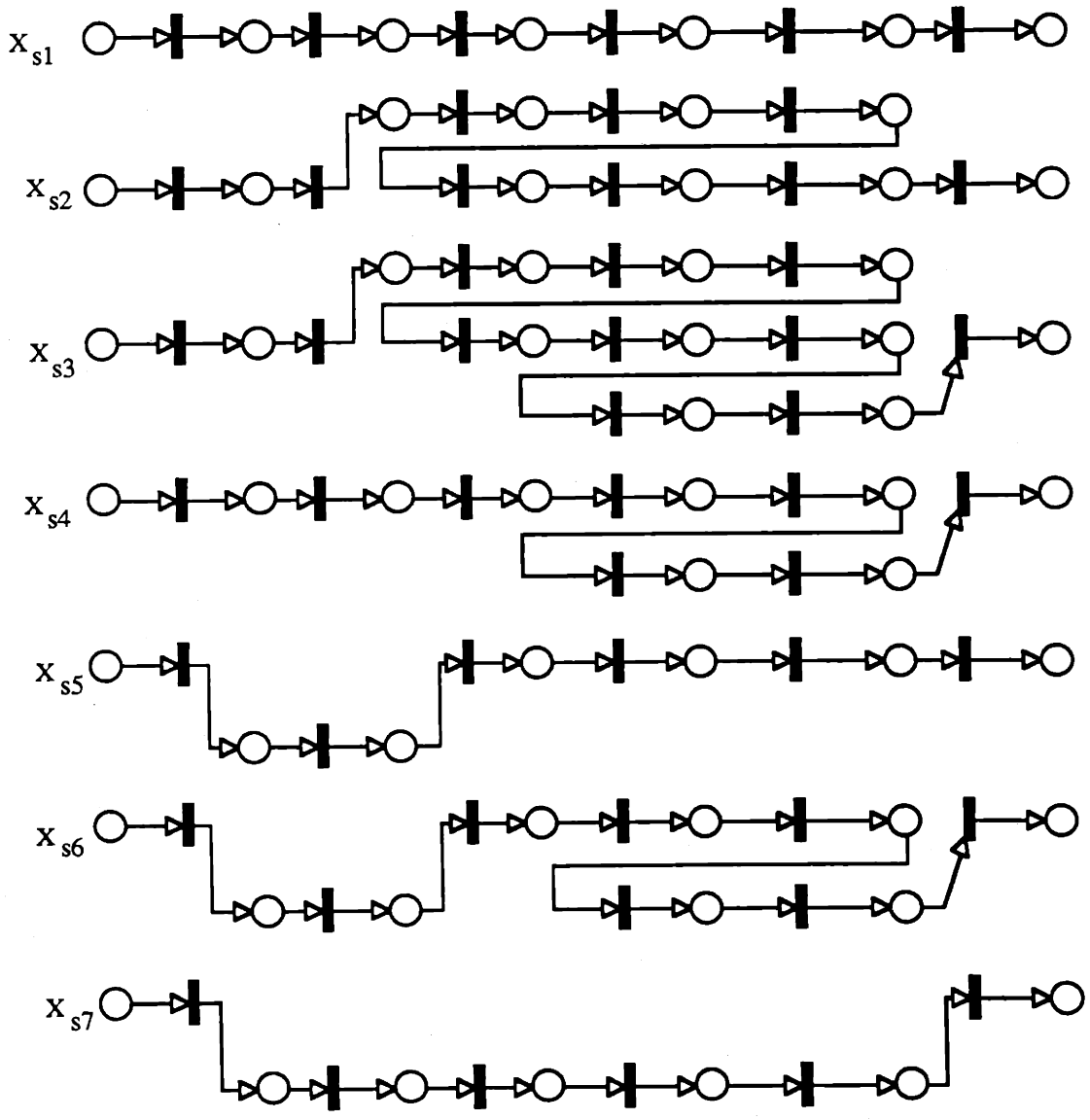


Figure 7.2: Simple paths of Π_1.

In Chapter VIII an algorithmic procedure will be presented to find all minimal supports S-invariants of a given Petri Net. The example of this subsection illustrates the kind of computations which are needed in the search for S-invariants.

7.4 CHARACTERIZATION OF $\Phi(R)$

7.4.1 Union of Simple Paths: the Set $USp(R_u)$

We defined in subsection 7.3.2 $Sp(R_u)$ as the set of all simple paths of $\Omega(R_u)$. Its cardinal is r and we can write $Sp(R_u) = \{sp_i , 1 \leq i \leq r\}$. Since simple paths are WDSs, the set $Sp(R_u)$ is included in the set of all WDSs Ψ^n:

$$\Psi^n \supset Sp(R_u).$$

Definition

We will denote $USp(R_u)$ the set of all possible unions (or meet) of elements of $Sp(R_u)$, augmented with the null element ω of Ψ^n.

$$USp(R_u) = \{\Pi \in \Psi^n / \exists (sp_{i1},...sp_{1q}\} \in Sp(Ru)^q \ \Pi = sp_{i1} \cup ... \cup sp_{iq}\} \cup \{\omega\}$$

$USp(R_u)$ is the set of all combinations of simple paths of the Universal Net $\Omega(R_u)$. The joint (or union) of two elements of $USp(R_u)$ will be the WDS composed of all the simple paths included in **either one** of the two considered elements. Similarly, the meet (or intersection) of two elements of $USp(R_u)$ will be the WDS composed of all simple paths included in **both** considered elements. The meet of two simple paths sp_i and sp_j will be the null element ω of Ψ^n, i.e., the WDS of dimension n without any link. This justifies the inclusion of ω into the set Ψ^n. The meet and the joint of two elements of $USp(R_u)$ are therefore elements of $USp(R_u)$, which induces the following proposition.

Proposition 7.5

The set USp(Ru) is a lattice.

It is easy to see that $USp(R_u)$ has at most 2^r elements and that any element of $USp(R_u)$ is included in $\Omega(R_u)$. In the general case, the number of elements of $USp(R_u)$ will be much less than 2^r, because the same WDS can be obtained from different combinations of different simple paths. To illustrate the point, let us consider the example of subsection 7.3.3. The following combinations of simple paths all yield the same WDS, as the reader can check by looking at Figure 7.2.

$$X_{s1} \cup X_{s3} \cup X_{s5} \equiv X_{s1} \cup X_{s3} \cup X_{s6} \equiv$$
$$X_{s1} \cup X_{s2} \cup X_{s6} \equiv X_{s2} \cup X_{s4} \cup X_{s5} \equiv$$
$$X_{s2} \cup X_{s4} \cup X_{s6} \equiv X_{s3} \cup X_{s4} \cup X_{s5}$$

Proposition 7.6 justifies the introduction of the set $USp(R_u)$.

Proposition 7.6

Every WDS, element of the set $USp(R_u)$, satisfies the connectivity constraint R_1. In formal language:

$$\{\Pi \in \Psi^n / R_1[\Pi] = 1\} \supset USp(R_u)$$

Proof

We will capitalize on the building rules of the elements of $USp(R_u)$ and prove Proposition 7.6 by induction.

Any simple path sp_i obviously satisfies R_1: this is the very definition of the constraint R_1.

Let us now consider a simple path sp_i and an element usp_1 of $USp(R_u)$ that we will assume satisfies the constraint R_1. There are two possible cases.

• $sp_i \leq usp_1$

In this case, $usp_1 \cap sp_i = sp_i$ and $usp_1 \cup sp_i = usp_1$. Consequently, both joint and meet of usp_1 and sp_i fulfill R_1.

• sp_i is not included in usp_1

We have $usp_1 \cap sp_i = \omega$ and $usp_1 \cup sp_i = usp_2$. For reason of consistency, we will consider that the null element ω fulfills the constraint R_1. Let us now consider a place p of usp_2. If p belongs to usp_1, there is a simple path of the Universal Net including p, since usp_1 fulfills R_1 by hypothesis. If p belongs to sp_i, sp_i itself is a path going from the source to the sink of $\Omega(R_u)$. Consequently, usp_2 satisfies R_1.

Since any element of $USp(R_u)$ is obtained by the joint or meet operation with a simple path, Proposition 7.6 follows.

Corollary

Let Π be a WDS and let the simple paths of Π be $sp_1, sp_2, \ldots sp_q$. If Π is not connected, i.e., violates the constraint R_1, the following inclusion is strict.

$$\Pi \supset \Pi' = sp_1 \cup sp_2 \cup \ldots \cup sp_q.$$

The above corollary is a direct consequence of Proposition 7.6. First, the WDS Π', obtained as the join of all the simple paths of Π, is necessarily included in Π by the very definition of the join operator $\cup$. The WDS Π' is furthermore an element of $USp(R_u)$ and is connected (Proposition 7.6). The WDS Π' cannot, therefore, be equal to Π which is not connected: it is then strictly included in Π.

Proposition 7.7 is a kind of reciprocal to Proposition 7.6.

Proposition 7.7

A Feasible Organizational Form that fulfills the constraint R_1 is an element of $USp(R_u)$, i.e.,

$$USp(R_u) \supset \{\Pi \in \Phi(R_u) / R_1[\Pi] = 1\}$$

$R_1[\Pi] = 1$ means that Π satisfies the constraint R_1 (see 7.2.1)

Proof

Let us consider a FOF Π which satisfies the constraint R_1. Let the simple paths of Π be $\{sp_1, sp_2, ...sp_q\}$ and let Π' be the joint of those simple paths: $\Pi' = sp_1 \cup ... \cup sp_q$. We have $\Pi' \leq \Pi$ (corollary of Proposition 7.6). Let us suppose that $\Pi' \neq \Pi$. This implies that there is at least one place p which belongs to Π but not to Π'. Since Π fulfills R_1, there is a simple path of Π which includes p. p is therefore included in the joint of all simple paths of Π, i.e. Π'. We reach a contradiction and therefore $\Pi = \Pi'$. Since a simple path of Π is a simple path of the universal net, Π' is an element of $USp(R_u)$.QED.

From Propositions 7.6 and 7.7 taken together, we have the following double inclusion:

$$\{\Pi \in \Psi^n / R_1[\Pi] = 1\} \supset USp(R_u) \supset \{\Pi \in \Phi(R_u) / R_1[\Pi] = 1\}$$

7.4.2 Characterization of $\Phi(R)$

We are now ready to put all the pieces together and state the following proposition characterizing the set $\Phi(R)$ of all feasible organizations.

Proposition 7.8

Let Π be a WDS of dimension n. Π will be a Feasible Organization if and only if

- Π is a union of simple paths of the Universal Net $\Omega(Ru)$, i.e., $\Pi \in USp(R_u)$.
- Π is bounded by at least one MINO and one MAXO.

In formal language:

$$\Phi(R) = \{\Pi \in USp(R_u) / \exists (\Pi_{min}, \Pi_{max}) \in \Phi_{min}(R) \times \Phi_{max}(R) \ \Pi_{min} \leq \Pi \leq \Pi_{max}\}$$

Proof

Let the set defined in the right-hand side of the above equation be called Z. From Proposition 7.6 we have:

$USp(R_u) \supset \{\Pi \in \Phi(R_u) / R_1[\Pi] = 1\} \supset \Phi(R)$.

From Proposition 7.1 we have:

$\{\Pi \in \Psi^n / \exists (\Pi_{min}, \Pi_{max}) \in \Phi_{min}(R) \times \Phi_{max}(R) \ \ \Pi_{min} \leq \Pi \leq \Pi_{max}\} \supset \Phi(R)$

Consequently, $Z \supset \Phi(R)$. Let Π be an element of Z. Π belongs to $USp(R_u)$. Π satisfies the constraints R_u by definition of $USP(R_u)$ and the constraint R_1 by Proposition 7.4. Furthermore, Π is smaller than at least one MAXO Π_{max} and greater than at least one MINO Π_{min}. The WDS Π_{max} and Π_{min} fulfill the constraints R_2,R_3, and R_4 by definition. Since those three constraints are convex (Proposition 7.3), they are satisfied by Π. Consequently, Π satisfies the entire set of constraints R and is a Feasible Organization. It follows that $\Phi(R) \supset Z$ and consequently $\Phi(R) = Z$. QED.

Proposition 7.8 gives a characterization of the set $\Phi(R)$ just like Proposition 7.4 gives a characterization of the set $\Phi(R_u)$. While Ψ^n is used in the equality characterizing $\Phi(R_u)$, $USp(R_u)$ is used to characterize $\Phi(R)$. In the former case, the link is the incremental unit leading from a WDS to its immediate superordinate, while in the latter the simple path plays the role of the building unit.

7.4.3 Structure of the Set $\Phi(R)$

Proposition 7.8 defines the "boundaries" of the set $\Phi(R)$. The next step of the analysis is to understand the internal structure of $\Phi(R)$. This subsection addresses this issue and gives a few results about the structure of $\Phi(R)$.

According to Proposition 7.8, $\Phi(R)$ is included in $USp(R_u)$. $USp(R_u)$ is a lattice, but, in the general case, $\Phi(R)$ will not be a lattice as the following proposition shows.

Proposition 7.9

The set $\Phi(R)$ of all Feasible Organizations is a lattice, if and only if $\Phi(R)$ has exactly one MAXO and one MINO.

Proof

If the number of MINOs or MAXOs is greater than one, it is easy to see that $\Phi(R)$ is not a lattice. Indeed, if Π and Π' are two MAXOs, their joint $\Pi \cup \Pi'$ cannot be within the set $\Phi(R)$, since this would violate the fact that Π and Π' are maximal elements of $\Phi(R)$. The same reasoning applies for MINOs.

Let us now suppose that $\Phi(R)$ has exactly one MINO and one MAXO. The proof that $\Phi(R)$ is a lattice in this case is completely similar to the proof that $\Phi(R_u)$ is a lattice (see the corollary of Proposition 7.4).

Proposition 7.9 is actually a rather negative result since, in most cases, there will be several MAXOs and MINOs. More work is needed to gain a better understanding of the structure of the set $\Phi(R)$. The following development constitutes only a step in that direction.

Any element of $\Phi(R)$ belongs to a chain whose extremities are a MINO and a MAXO. A characterization technique of all the chains leading from a given MINO to a given MAXO would therefore bring deeper insight into the structure of the set $\Phi(R)$. The first step is to determine the length of those chains. Chapter IX will provide examples where the set $\Phi(R)$ violates the Jordan-Dedekind chain condition. The chains leading from a given MINO to a given MAXO have, therefore, not necessarily the same length: the goal of the remaining part of this section is to define a lower bound to this length. To do so, the notion of minimal decomposition of an element of $\Phi(R)$ is introduced.

Minimal Decomposition of a Feasible Organization

Let Π be an element of $\Phi(R)$ and let $Sp(\Pi)=\{sp_1,sp_2,...,sp_q\}$ be the set of all the simple paths of Π. Since the Feasible Organization Π is equal to the join of all its simple paths, i.e., $\Pi=sp_1 \cup sp_2 \cup ... \cup sp_q$, $Sp(\Pi)$ will be called a **generating family** of Π (the word family is taken from the language of Vector Space Theory). The family $Sp(\Pi)$ may however not be the smallest family able to generate Π. Let $USp(\Pi)$ denote the set of all unions of elements of $Sp(\Pi)$, augmented with the null element ω. Note that $Sp(\Pi)$ and $USp(\Pi)$ are defined in the way that $Sp(R_u)$ and $USp(R_u)$ have been defined in subsections 7.3.2 and 7.4.1: Π plays now the role of the net $\Omega(R_u)$. Therefore, Proposition 7.5 applies to $USp(\Pi)$ and this set is a lattice: its least element is ω and its greatest element is Π. The

minimal decompositions of Π will be characterized by the minimal length chains in the lattice USp(Π). The following definition summarizes this analysis.

Definition

Let Π be an element of $\Phi(R)$ and let USp(Π) be the lattice generated by all the simple paths of Π. A **minimal decomposition** of Π will be a family of simple paths of Π, that constitutes a **minimal length chain** leading from ω to Π in the lattice USp(Π).

In other words, a minimal decomposition of Π is a family $A = \{sp_{i1},...,sp_{is}\}$ of simple paths of Π, satisfying the following conditions:

- $\Pi = sp_{i1} \cup ... \cup sp_{is}$
- $\omega < sp_{i1} < sp_{i1} \cup sp_{i2} < sp_{i1} \cup sp_{i2} \cup sp_{i3} < \ldots < sp_{i1} \cup ... \cup sp_{is} = \Pi$
- The length of any chain leading from ω to Π is at least equal to s.

Note that Π may have several minimal decompositions. The number of elements of all the minimal decompositions of Π is, however, the same: this number will be called the complexity of Π.

Definition

Let Π be an element of $\Phi(R)$. The **complexity** of Π, denoted $c(\Pi)$, will be the minimum number of simple paths whose join (or union) is equal to Π.

The complexity is a monotone function of its argument. In other words, if Π and Π' are two FOs, the following implication holds:

$$(\Pi \geq \Pi') \Rightarrow (c(\Pi) \geq c(\Pi')).$$

The two following propositions are key properties of minimal decomposition and complexity.

Proposition 7.10

Let Π be a FO and let $\{sp_1, sp_2, ... sp_q\}$ be a minimal decomposition of Π. Let Π_i be defined as follows: $\Pi_i = sp_1 \cup sp_2 \cup ... \cup sp_i$.

The family $\{sp_1,...,sp_i\}$ is a minimal decomposition of Π_i and $c(\Pi_i) = i$.

Proof

The family $\{sp_1,...,sp_i\}$ is a generating family of Π_i. Let us suppose that the complexity of Π_i is smaller than i. There is, therefore, a minimal decomposition of Π_i with less than i elements: $\Pi_i = sp_1' \cup sp_2' \cup ... \cup sp_s'$ with $s < i$.

Since $\Pi = \Pi_i \cup sp_{i+1} \cup sp_{i+2} \cup ... \cup sp_q$, the complexity of Π is at most equal to s+q-i, which is smaller than q. This is a contradiction and therefore the complexity of Π_i is exactly equal to i. It follows that $\{sp_1,...,sp_i\}$ is a minimal decomposition of Π_i.

Proposition 7.11

Let Π and Π' be two FOs, elements of $\Phi(R)$. The following inequality holds:

$$c(\Pi \cup \Pi') \leq c(\Pi) + c(\Pi') - c(\Pi \cap \Pi') \qquad (7.1)$$

Proof

An inductive method of proof will be used.

Let $\{sp_1, sp_2, ... sp_q\}$ be a minimal generating family of Π. Let Π_i be defined as follows: $\Pi_i = sp_1 \cup sp_2 \cup ... \cup sp_i$. Note that $\Pi = \Pi_q$. Let us prove by induction on i that the following property, denoted (Pi), holds for all i between 1 and q.

$$(Pi) : c(\Pi' \cup \Pi_i) \leq c(\Pi') + c(\Pi_i) - c(\Pi' \cap \Pi_i)$$

• i=1. There are two cases. Either $sp_1 \leq \Pi'$ or not. If $sp_1 \leq \Pi'$, then $\Pi' \cup sp_1 = \Pi'$ and $\Pi' \cap sp_1 = sp_1$. Therefore,

$$c(\Pi' \cup sp_1) = c(\Pi') = c(\Pi') + c(sp_1) - c(sp_1)$$

$$= c(\Pi') + c(sp_1) - c(\Pi' \cap sp_1).$$

(P1) holds in this case. Let us consider the other case where sp_1 is not included in Π'. The union of a minimal generating family of Π' with sp_1 will be a generating family of $\Pi' \cup sp_1$. Consequently, we have $c(\Pi' \cup sp_1) \leq c(\Pi') + 1 = c(\Pi') + c(sp_1)$.

Lastly $\Pi' \cap sp_1 = \omega$ and $c(\Pi' \cap sp_1) = c(\omega) = 0$. We have therefore,

$c(\Pi' \cup sp_1) \leq c(\Pi') + c(sp_1) - c(\Pi' \cap sp_1)$.

We have proved that (P1) holds in both cases.

• Let us assume that (Pi-1) is true and let us prove that (Pi) is also true ($i > 1$).

$$c(\Pi' \cup \Pi_i) = c(\Pi' \cup (\Pi_{i-1} \cup sp_i))$$

$$= c((\Pi' \cup \Pi_{i-1}) \cup sp_i)$$

$$\leq c(\Pi' \cup \Pi_{i-1}) + 1$$

Let us apply (Pi-1) to the right hand of the previous inequality:

$$c(\Pi' \cup \Pi_i) \leq c(\Pi') + c(\Pi_{i-1}) - c(\Pi' \cap \Pi_{i-1}) + 1$$

Since $\{sp_1, sp_2, \ldots sp_q\}$ is a minimal generating family of Π, Proposition 7.10 applies and $c(\Pi_i) = i = c(\Pi_{i-1}) + 1$. Therefore,

$$c(\Pi' \cup \Pi_i) \leq c(\Pi') + c(\Pi_i) - c(\Pi' \cap \Pi_{i-1})$$

Lastly, $\Pi' \cap \Pi_i \leq \Pi' \cap \Pi_{i-1}$ and since c is a monotone function,

$$c(\Pi' \cup \Pi_i) \leq c(\Pi') + c(\Pi_i) - c(\Pi' \cap \Pi_i)$$

We have proved that (Pi) is true, given that (Pi-1) is true.

Since (P1) is true and since (Pi) is a recursive property,i.e., (Pi-1) $\Rightarrow$ (Pi), then (Pq) will be true and therefore,

$$c(\Pi' \cup \Pi) \leq c(\Pi') + c(\Pi) - c(\Pi' \cap \Pi).$$ QED.

Let us show on an example that (7.1) can actually be a strict inequality. Figure 7.3 represents a 2-dimensional WDS Π_1 whose complexity is equal to 3: $\Pi_1 = sp_1 \cup sp2 \cup sp_3$. The family $\{sp_1, sp_2, sp_3\}$ is a minimal decomposition of Π_1. When the simple path sp_4 is added to Π_1, the resulting WDS Π_2 has still a complexity of 3: $\Pi_2 = sp_1 \cup sp_2 \cup sp_5$.

Therefore,

$$3 = c(\Pi_2) = c(\Pi_1 \cup sp_4) < c(\Pi_2) + c(sp4) - c(\Pi_2 \cap sp_4) = 4$$

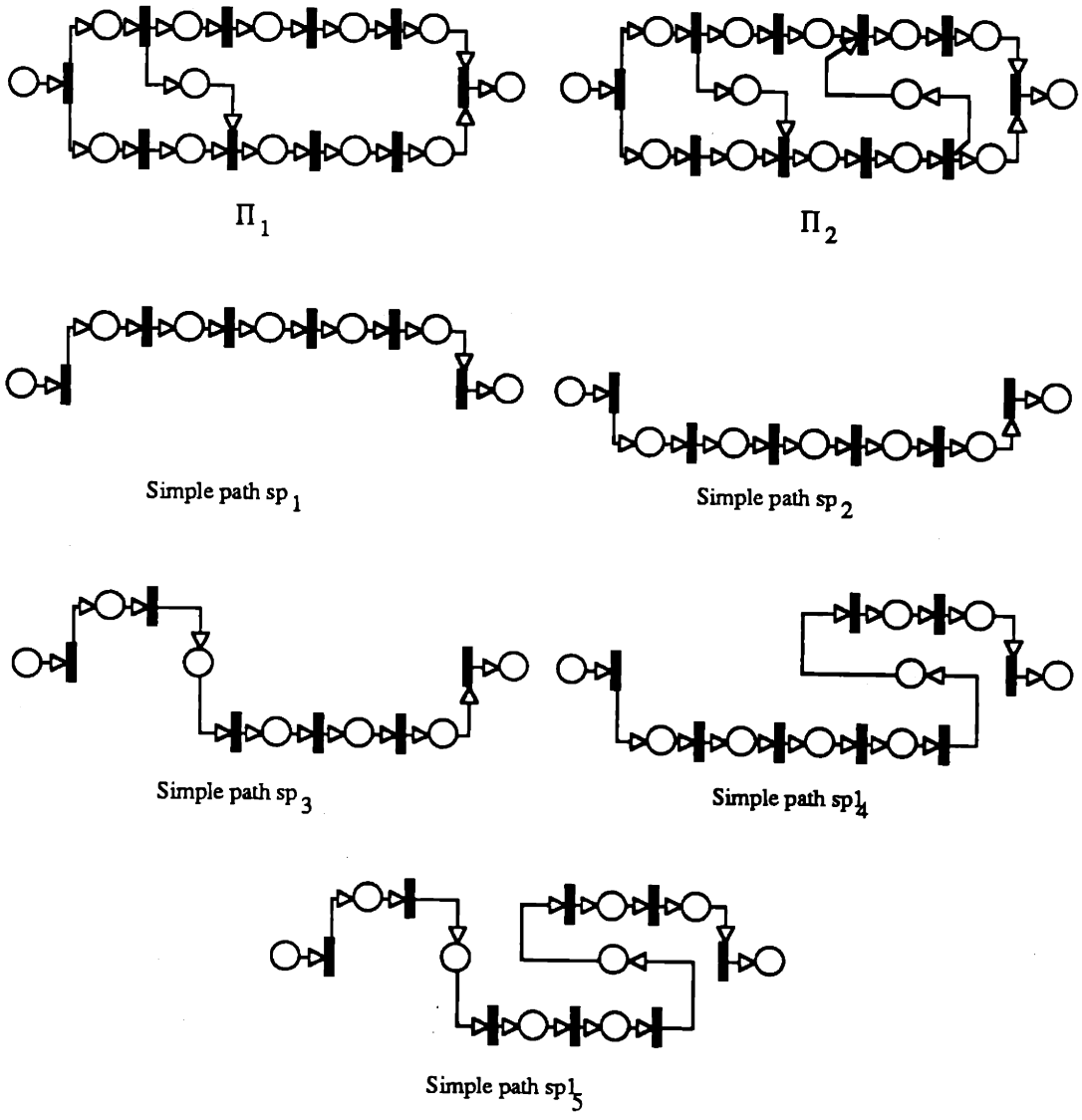


Figure 7.3: Example of a strict inequality in (7.1).

The following proposition justifies the introduction of the notion of complexity.

Proposition 7.12

Let Π and Π' be two Feasible Organizations such that $\Pi < \Pi'$. Any chain between Π and Π' has at least a length equal to $c(\Pi') - c(\Pi) + 1$.

Proof

Let $c(\Pi) = q$ and let $\{sp_1, sp_2, \ldots sp_q\}$ be a minimal generating family of Π. Let us consider a chain of length r between Π and Π':

$$\Pi \leq \Pi \cup sp_{q+1} \leq \ldots \leq \Pi \cup sp_{q+1} \cup \ldots \cup sp_{q+r-1} = \Pi'.$$

The family $\{sp_1, sp_2, \ldots sp_{q+r-1}\}$ is a generating family of Π' and therefore, $c(\Pi') \leq q + r - 1 = c(\Pi) + r - 1$. In other words, $c(\Pi') - c(\Pi) + 1 \leq r$. QED.

It should be noted that Proposition 7.12 does not guarantee that a chain whose length is exactly equal to $c(\Pi')-c(\Pi)+1$ exists.

In this subsection, some directions for the analysis of the internal structure of the set $\Phi(R)$ have been explored. The fact that $\Phi(R)$ violates the Jordan-Dedekind chain condition makes things fairly complicated. More work is required to gain deeper insight into $\Phi(R)$. The ultimate goal would be to define some "categories" of organizations within the set $\Phi(R)$ and to give those categories physical interpretation.

CHAPTER VIII

ALGORITHMIC IMPLEMENTATION

8.1 OVERALL STRUCTURE OF THE ALGORITHM

The design methodology presented in the previous chapters has been implemented on a personal computer. The specifications of the system are given in Table 8.1. The complete algorithm is composed of four programs, written in Turbo Pascal ©: ARCGEN, MAIN, PORGA, and DORGA. These programs have been compiled independently but they all have the same memory map to ensure proper chaining between them. The Pascal command "execute" is used to transfer control from one program to another.

TABLE 8.1 DESIGN WORKSTATION SPECIFICATIONS

Hardware:	IBM AT with - 512 k RAM.
	- 20 M Hard Disk.
	IBM Professional Graphics Display Monitor.
	EPSON FX-100 Dot Matrix Printer.
Software:	DOS 3.0. © IBM.
	Turbo Pascal 3.01A. © Borland International.
	Screen Sculptor. © Software Bottling Company.

Since ARCGEN initiates the procedure, the complete package will be denoted ARCGEN. To begin a design session, the user invokes ARCGEN. The title screen (screen#1) with the description of the program appears. Whenever a key is struck, control is transferred to the program MAIN: screen#2, shown in Figure 8.1, replaces screen#1. This screen presents to the user a menu with three options: "Input or modify CONSTRAINT DATA" invokes the program PORGA, "Run DESIGN program" invokes the program DORGA, and "Exit to DOS" transfers control back to the operating system thus terminating the session. If PORGA or DORGA is invoked, control is given back to MAIN after

execution. These two programs can therefore be run as many times as desired in the same session. The default for all three options is "N"; the user has to type "Y" and escape for one of the three options to continue.

The above description of the overall architecture of the package is summarized in Figure 8.2. A Petri Net has been used instead of the classical flow chart representation. The Petri Net representation is indeed more accurate and more powerful than the flow chart representation. As it will be seen, it allows for different levels of refinement in the description of the algorithm structure. Figure 8.2 represents the highest level, where each algorithm is aggregated into a single transition. By refining those transitions, one can obtain more and more detailed descriptions, while preserving the same basic structure. The places of Figure 8.2 represent the actual state of the computer screen. The place corresponding to screen#2 (program MAIN) has three output transitions representing the three options available to the user.

GENERATION OF ORGANIZATIONAL ARCHITECTURES

written by PASCAL A. REMY

M.I.T. Laboratory for Information and Decision Systems

MAIN MENU

	(Y / N)
Input or modify CONSTRAINT DATA	**N**
Run DESIGN program	**Y**
Exit to DOS	**N**

Figure 8.1: Screen#2 (program MAIN).

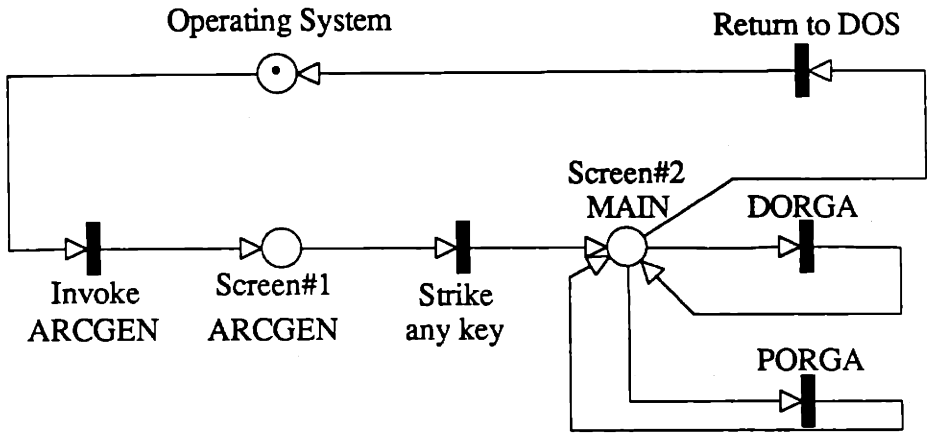


Figure 8.2: Petri Net representation of the overall structure.

Before the description of the program DORGA (the core of the algorithm) is given, the program PORGA is examined. This program allows the user to enter or modify user-defined constraints. A database of user-defined constraints has been created and is stored in the file CONSTRAINT.DAT. The user can retrieve, use, and modify existing records from the database. He can also create new ones that will be added to the database. In brief, PORGA allows the user to update the database CONSTRAINT.DAT. A pointer indicates how many records are currently stored in the database. The internal structure of the program PORGA is represented in Figure 8.3. The Petri Net of Figure 8.3 is the refinement of the transition labeled PORGA in Figure 8.2.

As in Figure 8.2, each place of Figure 8.3 represents a screen. The transitions represent the actual processing done when going from one screen to another. The place corresponding to the Menu Screen (screen#3 in Figure 8.4) has six output transitions corresponding to the six options available to the user at this stage. The user can retrieve constraint data from the database CONSTRAINT.DAT and can also enter new data. In the later case, he can choose the number of decisionmakers (between 2 and 5, the maximum 5 being the default value). Once the data are entered or modified, the user can store them in the database either in adding a new record or in overwriting an existing record.

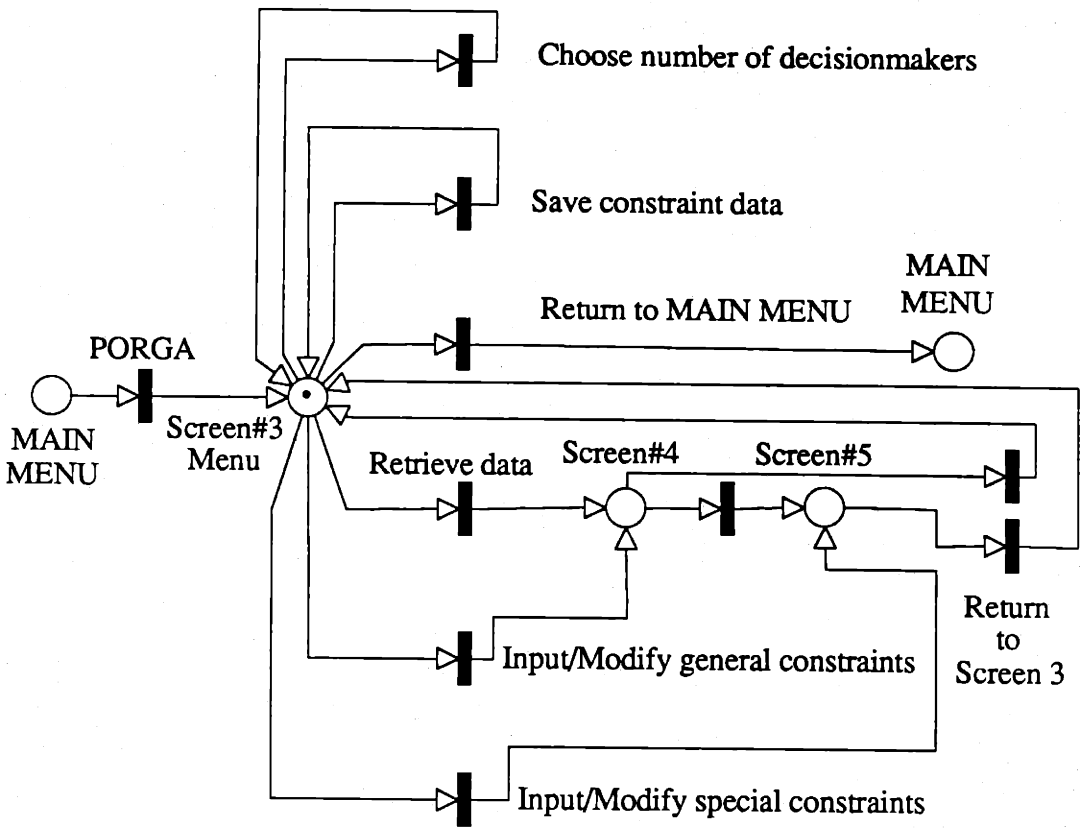


Figure 8.3: Petri Net representation of the program PORGA.

When data are retrieved or entered screen#4 shown in Figure 8.5, appears: it provides a graphic representation of the arrays defining the constraint R_f. The dimension of the arrays will correspond to the number of decisionmakers requested by the user. When data are retrieved from the database, screen#5 - corresponding to the special constraints and shown in Figure 8.6 -, appears automatically after screen#4. When new data are entered, however, screen#5 will appear only if requested by the user. Eventually, the menu screen will reappear. The user will exit the program by answering "Y" to the prompt "Return to MAIN MENU".

GENERATION OF ORGANIZATIONAL ARCHITECTURES

(Y / N)

Retrieve — Y

Save — organization data — N — Constraint set # 1 Total 6

Return to MAIN MENU — N

(Y / N)

Input / Modify general constraints — N

Input / Modify special constraints — N

Number of decisionmakers (2 - 5) — 5

Figure 8.4: Screen#3 (program PORGA)

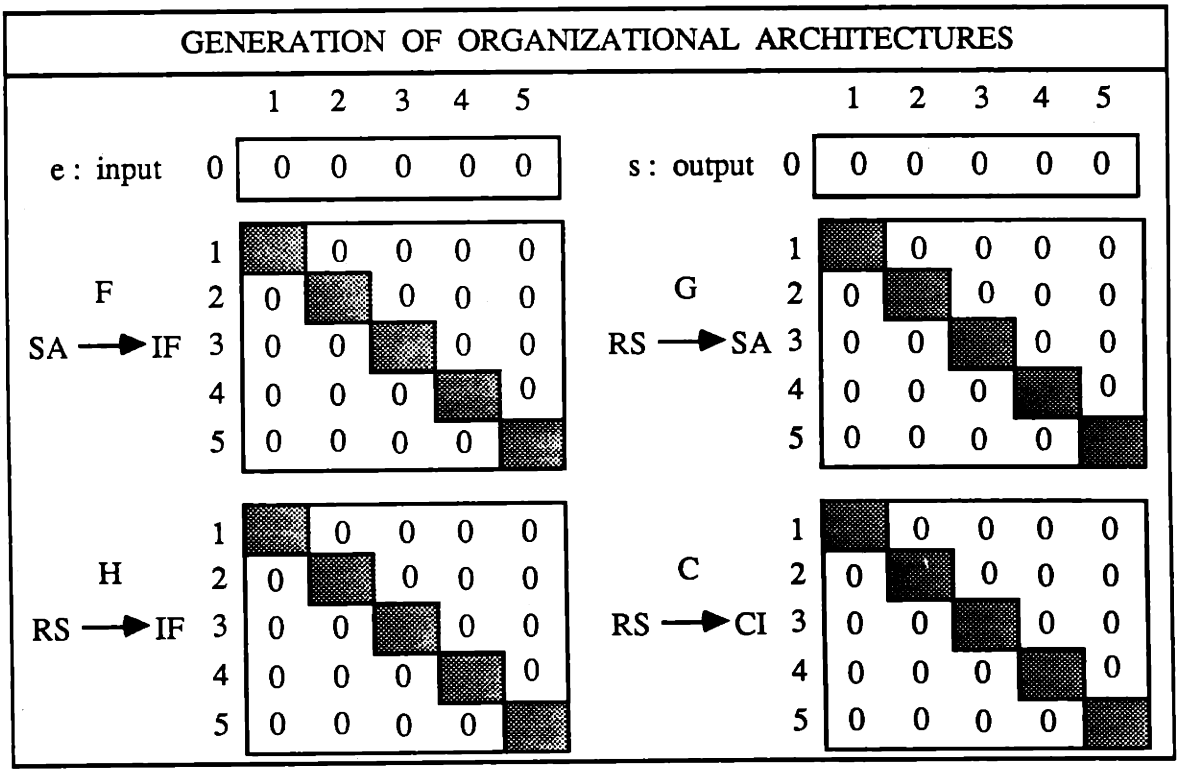


Figure 8.5: Screen#4 (program PORGA)

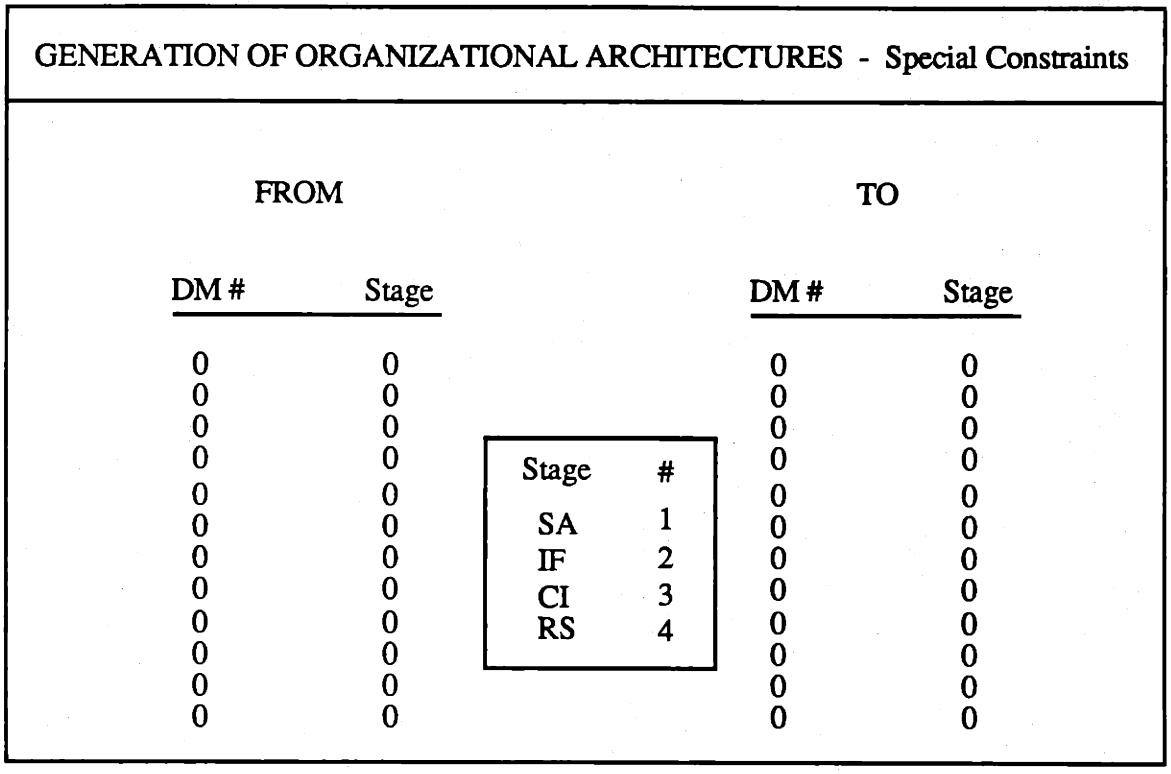


Figure 8.6: Screen#5 (program PORGA)

The screens used in ARCGEN, MAIN, and PORGA have been designed using the commercial software SCREEN SCULPTOR © by the Software Bottling Company.

8.2 PROGRAM DORGA: THE SEARCH FOR MINOS AND MAXOS.

8.2.1 Overview

Figure 8.7 represents the structure of the program DORGA.

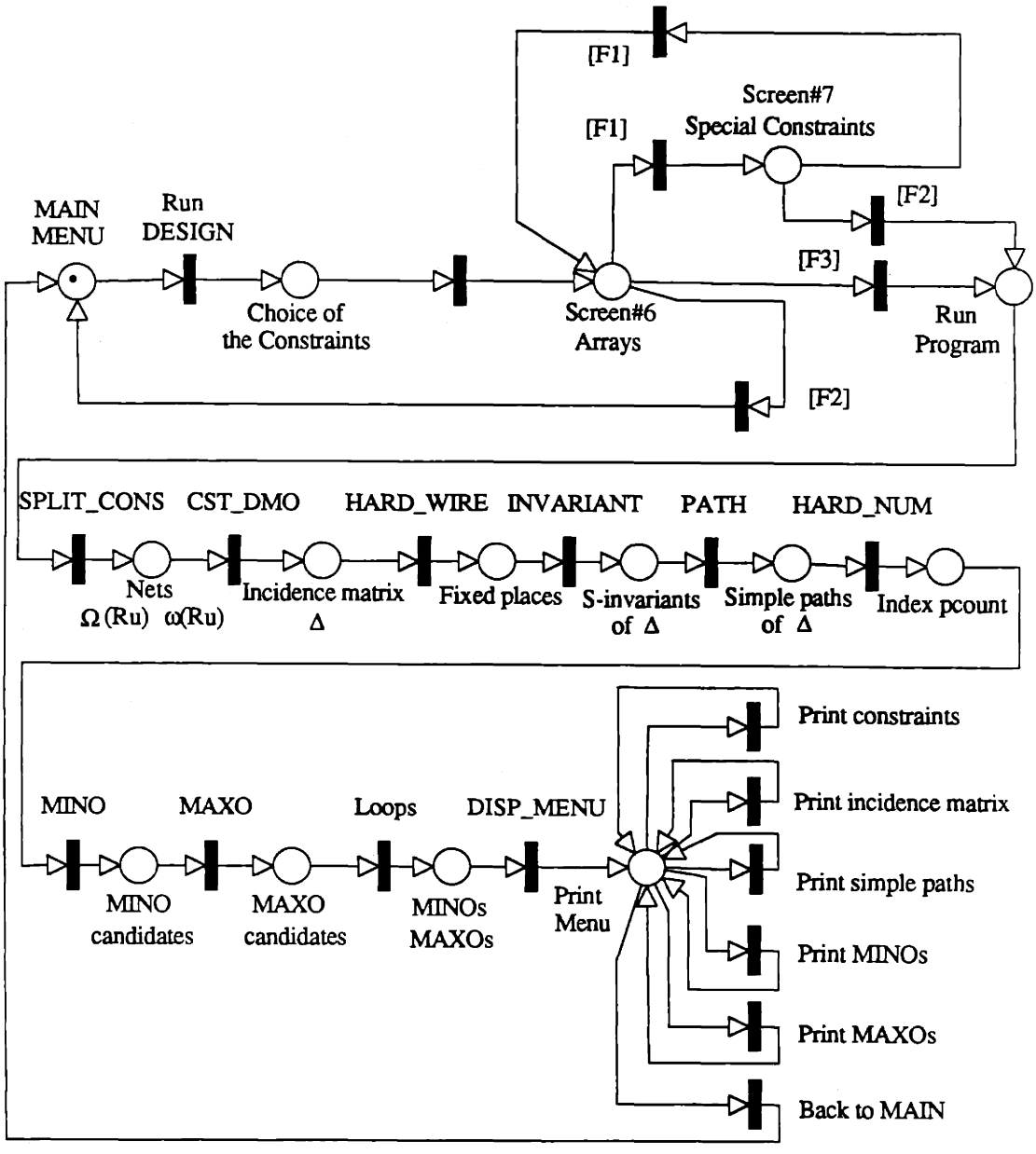


Figure 8.7: Structure of the program DORGA.

The program DORGA is the core of the complete package ARCGEN: it implements the methodology described in Chapter VII. The program first retrieves a set R_u of user-defined constraints from the database CONSTRAINT.DAT and then generates the

corresponding MINOs and MAXOs . As intermediate results, the incidence matrix of the net Ω(Ru) as well as the simple paths of this net are generated. The diagram in Figure 8.7 is divided into three levels corresponding to three different phases of the program. The upper level is the data acquisition stage, the intermediate level generates the simple paths of the set Ω(Ru), and the lower level corresponds to the computation of the MINOs and MAXOs and to the printing of the results. The three levels are analyzed below.

8.2.2 Data Acquisition Stage.

This stage corresponds to the **upper level** of the Petri Net of Figure 8.7. When the "Run DESIGN" option is invoked from the MAIN MENU (Screen#2), the user is prompted for the record number of the set of user-defined constraints he wishes to retrieve from the database. The arrays corresponding to the chosen set of constraints are displayed on the screen (Screen#6). This enables the user to check whether the retrieved set of constraints actually corresponds to the one he wants to use. Would this not be the case, the user, by hitting the function key [F2], can return to the MAIN MENU and either start again with a different set of constraints or run the program PORGA to scan the database. The special constraints (Screen#7) can also be checked by hitting the function key [F1] from Screen#6. Once the user is satisfied that he has the correct set of user-defined constraints, he hits [F3] (or [F2] from Screen#7) to go to the next stage of the algorithm.

8.2.3 Generation of the Set $Sp(R_u)$.

The **intermediate level** in Figure 8.7 represents the different steps necessary to generate all the simple paths of $\Omega(R_u)$. Each transition of this level corresponds to a specific subroutine and has been named accordingly. SPLIT_CONS differentiates between the fixed and the unspecified elements of the arrays defining the constraints R_u: the matrix representations of $\Omega(R_u)$ and $\omega(R_u)$ are generated at this stage. CST_DMO transforms those matrix representations into the representations (2) and (3) of subsection 5.4.2: the incidence matrices with the labeling of the places and transitions of both nets are produced. The incidence matrix of $\Omega(R_u)$ will be denoted Δ. HARD_WIRE matches the places of

$\Omega(R_u)$ with the places of $\omega(R_u)$ to identify the fixed places. INVARIANT computes the S-invariants of the incidence matrix Δ. This procedure has been taken from a program written by Hillion [5] and is based on an algorithm first proposed by Martinez and Silva [28] and improved by Alaiwan and Toudic [29]. Since the procedure generates all the S-invariants of the net $\Omega(R_u)$, simple paths need be sorted out and distinguished from internal loops. This operation is achieved by the subroutine PATH. A different version of the algorithm ARCGEN uses the procedure SIMPATH, developed by Jin [2], instead of the procedures INVARIANT and PATH. The procedure SIMPATH directly generates the simple paths of the net $\Omega(R_u)$. HARD_NUM assigns to each place of the net $\Omega(R_u)$ an index, **pcount**, representing the number of simple paths containing this place. The usefulness of this index will be clarified when the algorithm MINO is described. At this stage, the set $Sp(R_u)$ of all the simple paths of the net $\Omega(R_u)$ is found. Before describing the last stage of the algorithm, it is necessary to explain the technique used to store WDS.

8.2.4 Internal Representation of WDSs.

In subsection 5.4.4, three representations of a WDS are described: representation (1) is the matrix representation, representation (2) consists of the incidence matrix with the associated labeling of the transitions , while representation (3) refers to the labeling technique of the places. The three representations are used to describe the net $\Omega(R_u)$. The matrix representation is directly derived from the user-defined constraints and is used to compute the incidence matrix of the net. The incidence matrix is necessary to do the actual computation but is very inefficient as far as memory space is concerned. The representation (3), i.e. the labeling technique of the places, has been introduced to remedy this situation: it is compact and requires much less memory space than the incidence matrix representation. A N-dimensional vector $\underline{L}$, where the labels of the places of $\Omega(R_u)$ are stored, will thus characterize the net $\Omega(R_u)$. Any subnet of $\Omega(R_u)$, and a fortiori any Feasible Organization , will be represented by a N-dimensional vector whose elements correspond to the places of $\Omega(R_u)$ and take value in {0,1}. A 1 indicates that the corresponding place is present, while

a 0 denotes its absence. In the sequel, this representation of a subnet of Ω(Ru) will be referred to as the vector representation. A simple path of $\Omega(R_u)$ will thus be represented by a N-dimensional integer vector. The vector representation is extremely efficient to perform meet and join operations: the meet and join are done element by element following the rules described in 4.4.3. Lastly, the incidence matrix of any subnet of $\Omega(R_u)$ can be obtained from the vector representation of the subnet by extracting from the matrix Δ the rows corresponding to the 1's of the vector. The procedure EXTRACT accomplishes this function.

8.2.5 The Search for MINOs

MINOs and MAXOs are obtained as combinations of simple paths of $Sp(R_u)$: they are elements of the set $USp(R_u)$. The vector representation is used to characterize any element of $USp(R_u)$. The search for MINOs is done by scanning the set $USp(R_u)$, depth first, as illustrated in Figure 8.8. The depth (or vertical) index will be denoted i and, for a given depth, the lateral (or horizontal) index will be denoted j[i].

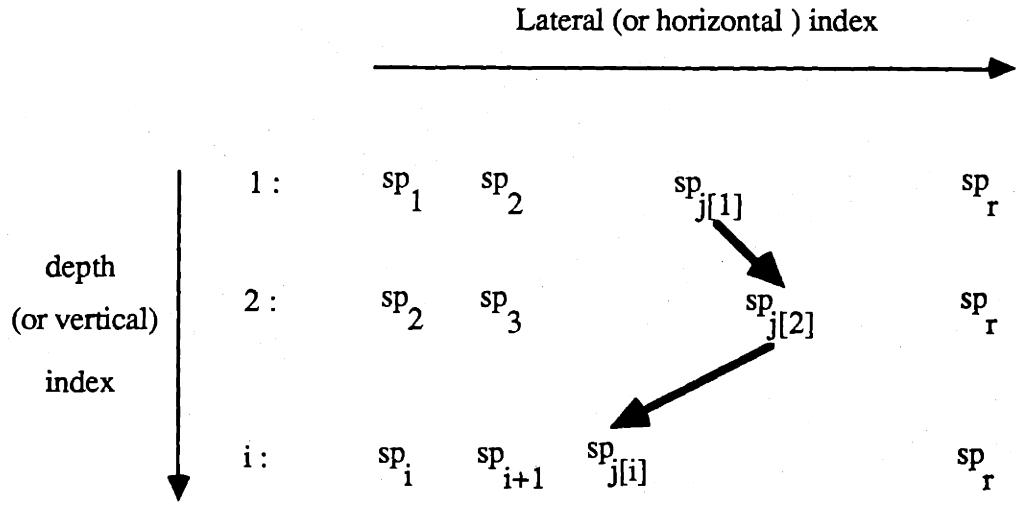


Figure 8.8: Scanning technique of $USp(R_u)$.

A given point in the scanning process of $USp(R_u)$ will be characterized by its depth index i and by the lateral indices j[i], j[i-1],...,j[1]. Such a point will correspond to the element $\Pi = sp_{j[1]} \cup sp_{j[2]} \cup \ldots \cup sp_{j[i]}$ .To eliminate redundancy, the lateral index j[i] is a strictly monotone function of i: j[i] > j[i-1] > ... > j[1]. In Figure 8.8, the first element corresponding to depth i is thus sp_i. This scanning process is exhaustive and ensures that no element of $USp(R_u)$ will be omitted. The scanning can, however, be improved. Since only minimally connected organizations are searched, every single element of $USp(R_u)$ need not be considered. More specifically, as soon as an element of $USp(R_u)$ is found that includes all the fixed places, the search can stop: the element is a MINO candidate and there is no need to keep adding simple paths. Consequently, simple paths will be chosen in such a way that the number of fixed places that are left out is strictly decreasing as new simple paths are added. This technique is implemented as follows. Let suppose that the depth index is i and that the corresponding element of $USp(R_u)$ is:

$$\Pi_i = sp_{j[1]} \cup sp_{j[2]} \cup \ldots \cup sp_{j[i]} .$$

Let us, furthermore, assume that this element satisfies the structural constraints, but violates the user-defined constraints because it does not include all fixed places. The fixed places that are not included in Π are scanned and the one with the smallest index, pcount, is chosen: let p_{min} denote this place. Let $j_{max}[i]$ denote the number of simple paths that include the place p_{min}. The selected simple paths are added one by one toΠ. The process is repeated until no fixed place is left out. At each step, the structural constraints are checked, using the procedure CHECK. Figure 8.9 summarizes this procedure. In the scanning process, when a net Π_i is found that both satisfies the structural constraints and includes all fixed places, such a net is a MINO candidate. It is then compared to the previously found MINOs to eliminate possible redundancy. A given MINO can, indeed, be obtained as a combination of two different sets of simple paths. When the place labeled "end" in Figure 8.9 is reached the search stops. A set of MINO candidates has been generated. One last check need be done. It comes from the fact that the subroutine CHECK does not eliminate **multi-level loops**, when checking the structural constraints. Multi-level loops refer to those loops involving more than two decisionmakers. Because those loops may be very

intricate, they are not accounted for in the procedure CHECK. The procedure INVARIANT is indeed used to locate such loops and eliminate the corresponding MINO candidates. Because the same problem arises for MAXOs, this last check is done simultaneously on MINO and MAXO candidates: it is represented in Figure 8.7 by the transition labeled "loops" in the lower level of the figure.

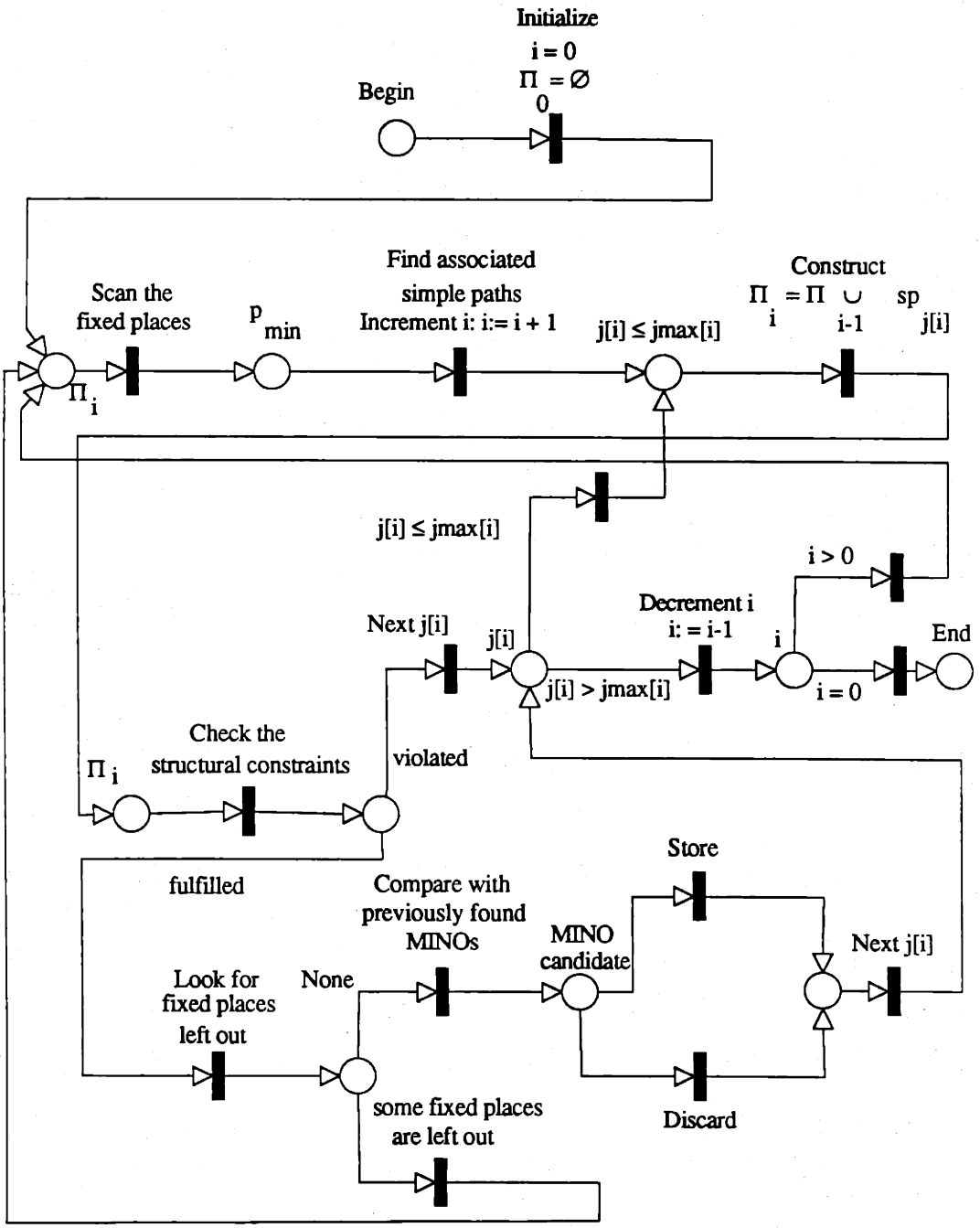


Figure 8.9: The search for MINOs.

8.2.6 The Search for MAXOs

The MAXOs and the MINOs are found in a similar way. In the case of MAXOs, however, the scanning process of $USp(R_u)$ is done starting from the net consisting of the union of all the simple paths of $Sp(R_u)$, i.e., the greatest element of $USp(R_u)$. Simple paths are then removed one by one until structural constraints are satisfied. At each step, it is necessary to ensure that no fixed place is removed, since this would induce a violation of the user-defined constraints. The order in which simple paths are removed proceeds from the following rationale. When a violation of the structural constraints is encountered, the subroutine CHECK identifies two of the places responsible for the violation. The simple paths that include those two places are then singled out and removed one by one from the initial structure. This step is the counterpart of the step using the index pcount in the search for MINOs. The procedure is repeated until the structural constraints are satisfied - in which case a MAXO candidate is found - or until the user-defined constraints are violated - in which case the organization is discarded. Once the set of all MAXO candidates is found, the procedure INVARIANT is used to eliminate multi-level loops.

8.2.7 Presentation of the Results

Once MAXOs and MINOs have been found, a print menu is displayed on the screen: it corresponds to the transition labeled DISP_MENU in Figure 8.7. This menu allows the user to get a hard copy of the user-defined constraints (key [F1]), the incidence matrix of $\Omega(Ru)$ (key ([F2]), the simple paths of $\Omega(Ru)$ (key [F3]), and the MAXOs and MINOs (keys [F4] and [F5]). Finally, by hitting the key [F6], the user comes back to the MAIN MENU and can start another session or leave the program. Simple paths, MINOs and MAXOs are printed using their vector representations. The algorithmic labeling of the places is given to help the user interpret the results. Examples of computer printouts are given in Chapter IX.

8.3 USER MANUAL

The different steps necessary to run a design session are summarized below.

- Start the session by invoking ARCGEN. Hit any key to continue.
- The MAIN MENU is displayed. Select the first option: "Input or modify CONSTRAINT DATA"
- The menu screen of PORGA is displayed. Enter user-defined constraints, either from scratch or by first retrieving existing data. Do not forget to save the data you just entered! Return to MAIN MENU once you are satisfied with the user-defined constraints.
- Select the second option from the MAIN MENU: "Run DESIGN program".
- Enter the record number corresponding to the set of user-defined constraints you just entered and run the design procedure.
- Print the results you are interested in and go back to the MAIN MENU.
- Start another session or return to the operating system in selecting the third option of the menu: "Exit to DOS".

CHAPTER IX

APPLICATIONS

Two examples are analyzed in this chapter. The first one is an example given to illustrate the mathematical concepts that have been used throughout the foregoing developments. The second one is an application illustrating how the methodology can be used to tackle concrete problems.

9.1 THE GENERAL CASE OF TWO MEMBER ORGANIZATIONS.

The most general case of a 2-member organization will be considered in this section. This example has been chosen because its complexity is high enough to generate insightful results while remaining within tractable limits.

9.1.1 User-defined Constraints

The first step of the design methodology consists in specifying the user-defined constraints. All interactions are allowable. The symmetry between the two decisionmakers will however be broken to eliminate some unnecessary redundancy but also to analyze the implications of having asymmetrical constraints. The symmetry is broken in specifying that DM^1 **must** receive an input from the external environment and that DM^2 **must** send an output to the external environment. All other interactions are permissible but not compulsory. The arrays represented in Figure 9.1 reflect those specifications.

A "x" indicates that the corresponding link may or may not be present. The dimension of the set $\Phi(R_{u1})$ of all WDSs satisfying the constraint R_{u1} is $2^{10} = 1{,}024$.

$$e = [\ 1\ x\ ] \qquad F = \begin{bmatrix} \# & x \\ x & \# \end{bmatrix} \qquad G = \begin{bmatrix} \# & x \\ x & \# \end{bmatrix}$$

$$s = [\ x\ 1\ ] \qquad H = \begin{bmatrix} \# & x \\ x & \# \end{bmatrix} \qquad C = \begin{bmatrix} \# & x \\ x & \# \end{bmatrix}$$

Figure 9.1: User-defined constraints R_{u1}.

9.1.2 Nets $\Omega(R_{u1})$ and $\omega(R_{u1})$.

The Net $\Omega(R_{u1})$, obtained by replacing by 1 all unspecified elements of the arrays in Figure 9.1, is shown in Figure 9.2. Note that this net contains all allowable interactions between two decisionmakers. Places have been labeled sequentially, thus avoiding clustering the picture with four digit labels, but transitions have been labeled according to the labeling scheme described in Chapter V. A WDN is indeed completely characterized by its graph with the associated labeling of the transitions: this is representation (2) of subsection 5.4.4. Fixed links and places are in boldface characters.

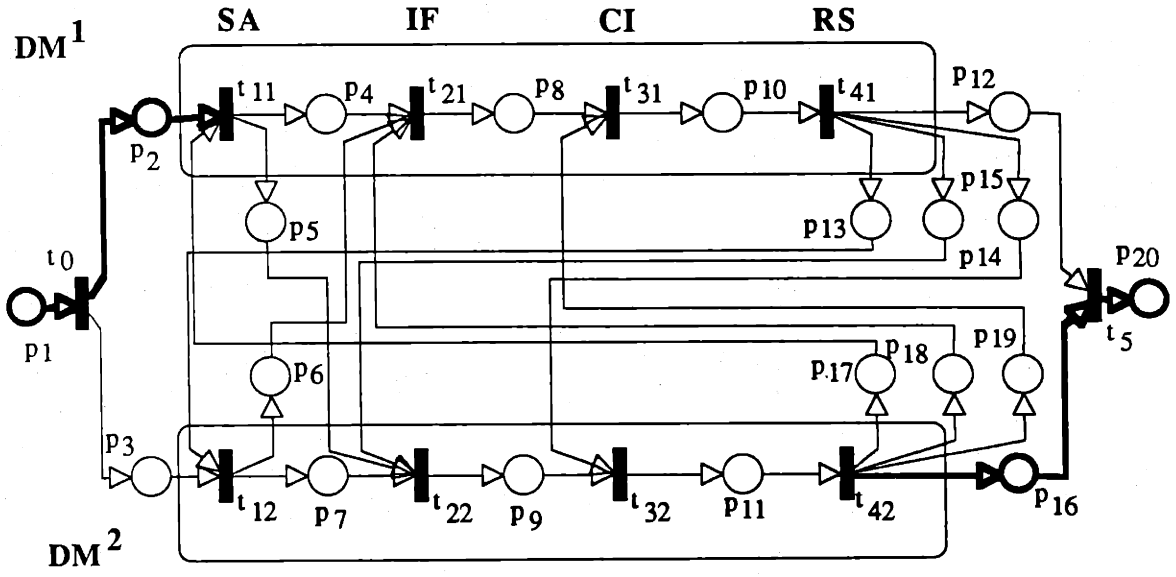


Figure 9.2: Universal Net $\Omega(R_{u1})$.

Table 9.1 gives the correspondence between the sequential labeling of the places of $\Omega(R_{u1})$ and the internal labeling technique used by the computer algorithm and described in Chapter V. In the sequel, the latter will be referred to as the algorithmic labeling.

TABLE 9.1 SEQUENTIAL VS ALGORITHMIC LABELING OF THE PLACES.

sequential label → algorithmic label

$p_1 \rightarrow p_0$	$p_2 \rightarrow p_{11}$	$p_3 \rightarrow p_{12}$	$p_4 \rightarrow p_{21}$	$p_5 \rightarrow p_{212}$
$p_6 \rightarrow p_{221}$	$p_7 \rightarrow p_{22}$	$p_8 \rightarrow p_{31}$	$p_9 \rightarrow p_{32}$	$p_{10} \rightarrow p_{41}$
$p_{11} \rightarrow p_{42}$	$p_{12} \rightarrow p_{51}$	$p_{13} \rightarrow p_{5121}$	$p_{14} \rightarrow p_{5122}$	$p_{15} \rightarrow p_{5123}$
$p_{16} \rightarrow p_{52}$	$p_{17} \rightarrow p_{5211}$	$p_{18} \rightarrow p_{5212}$	$p_{19} \rightarrow p_{5213}$	$p_{20} \rightarrow p_6$

The incidence matrix of $\Omega(R_{u1})$, computed from the constraints R_{u1} by the program ARCGEN, is reproduced in Table 9.2 below.

TABLE 9.2 INCIDENCE MATRIX Δ_1 OF $\Omega(R_{u1})$.

	t_0	t_{11}	t_{12}	t_{21}	t_{22}	t_{31}	t_{32}	t_{41}	t_{42}	t_5
p_1	-1	0	0	0	0	0	0	0	0	0
p_2	1	-1	0	0	0	0	0	0	0	0
p_3	1	0	-1	0	0	0	0	0	0	0
p_4	0	1	0	-1	0	0	0	0	0	0
p_5	0	1	0	0	-1	0	0	0	0	0
p_6	0	0	1	-1	0	0	0	0	0	0
p_7	0	0	1	0	-1	0	0	0	0	0
p_8	0	0	0	1	0	-1	0	0	0	0
p_9	0	0	0	0	1	0	-1	0	0	0
p_{10}	0	0	0	0	0	1	0	-1	0	0
p_{11}	0	0	0	0	0	0	1	0	-1	0
p_{12}	0	0	0	0	0	0	0	1	0	-1
p_{13}	0	0	-1	0	0	0	0	1	0	0
p_{14}	0	0	0	0	-1	0	0	1	0	0
p_{15}	0	0	0	0	0	0	-1	1	0	0
p_{16}	0	0	0	0	0	0	0	0	1	-1
p_{17}	0	-1	0	0	0	0	0	0	1	0
p_{18}	0	0	0	-1	0	0	0	0	1	0
p_{19}	0	0	0	0	0	-1	0	0	1	0
p_{20}	0	0	0	0	0	0	0	0	0	1

Net $\omega(R_{u1})$

This net is obtained in setting to 0 all unspecified elements of the arrays of Figure 9.1. The net is represented in Figure 9.3. Note that it is not connected, as it will generally be the case. By definition, the places of this net are the fixed places.

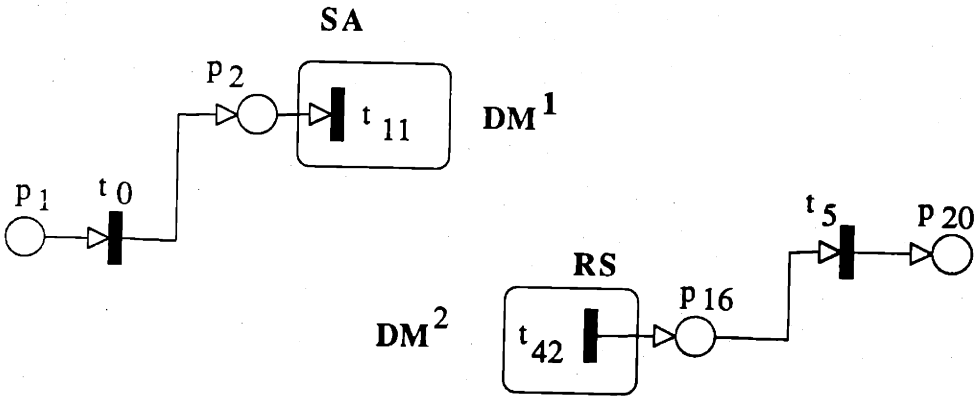


Figure 9.3: Net $\omega(R_{u1})$.

Simple paths of $\Omega(R_{u1})$

The net $\Omega(R_{u1})$ has 14 simple paths. They have been generated by the algorithm ARCGEN and are represented below. Table 9.3 gives the vector representations of these simple paths and Figure 9.4 gives the corresponding Petri Net representations. Fixed places are identified with an asterisk in Table 9.3 and with boldface characters in Figure 9.4. Note that the fixed places are exactly the places of the net $\omega(R_{u1})$.

The fixed places p_1 and p_{20} correspond respectively to the source and the sink of the net $\Omega(R_{u1})$: every simple path must include those two places by definition. Places p_2 and p_{16} refer respectively to the link between the external environment and DM^1 and to the link between DM^2 and the external environment. Those two places have to be included in every feasible organization.

TABLE 9.3 SIMPLE PATHS OF $\Omega(R_{u1})$.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
p_1 *	1	1	1	1	1	1	1	1	1	1	1	1	1	1
p_2 *	1	0	1	0	1	0	1	0	1	1	0	1	0	0
p_3	0	1	0	1	0	1	0	1	0	0	1	0	1	1
p_4	1	0	0	0	1	0	1	0	1	0	0	0	0	1
p_5	0	0	1	0	0	0	0	0	0	1	0	1	0	0
p_6	0	1	0	0	0	1	0	1	0	0	0	0	0	0
p_7	0	0	0	1	0	0	0	0	1	0	1	0	1	1
p_8	1	1	0	0	1	1	1	1	1	0	0	1	1	1
p_9	0	0	1	1	0	0	1	1	1	1	1	1	1	1
p_{10}	1	1	0	0	1	1	1	1	1	1	1	1	1	1
p_{11}	0	0	1	1	1	1	1	1	1	1	1	1	1	1
p_{12}	1	1	0	0	0	0	0	0	0	1	1	1	1	1
p_{13}	0	0	0	0	0	0	0	0	1	0	0	0	0	0
p_{14}	0	0	0	0	0	0	1	1	0	0	0	0	0	0
p_{15}	0	0	0	0	1	1	0	0	0	0	0	0	0	0
p_{16} *	0	0	1	1	1	1	1	1	1	0	0	0	0	0
p_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	1
p_{18}	0	0	0	0	0	0	0	0	0	0	0	0	1	0
p_{19}	0	0	0	0	0	0	0	0	0	1	1	0	0	0
p_{20} *	1	1	1	1	1	1	1	1	1	1	1	1	1	1

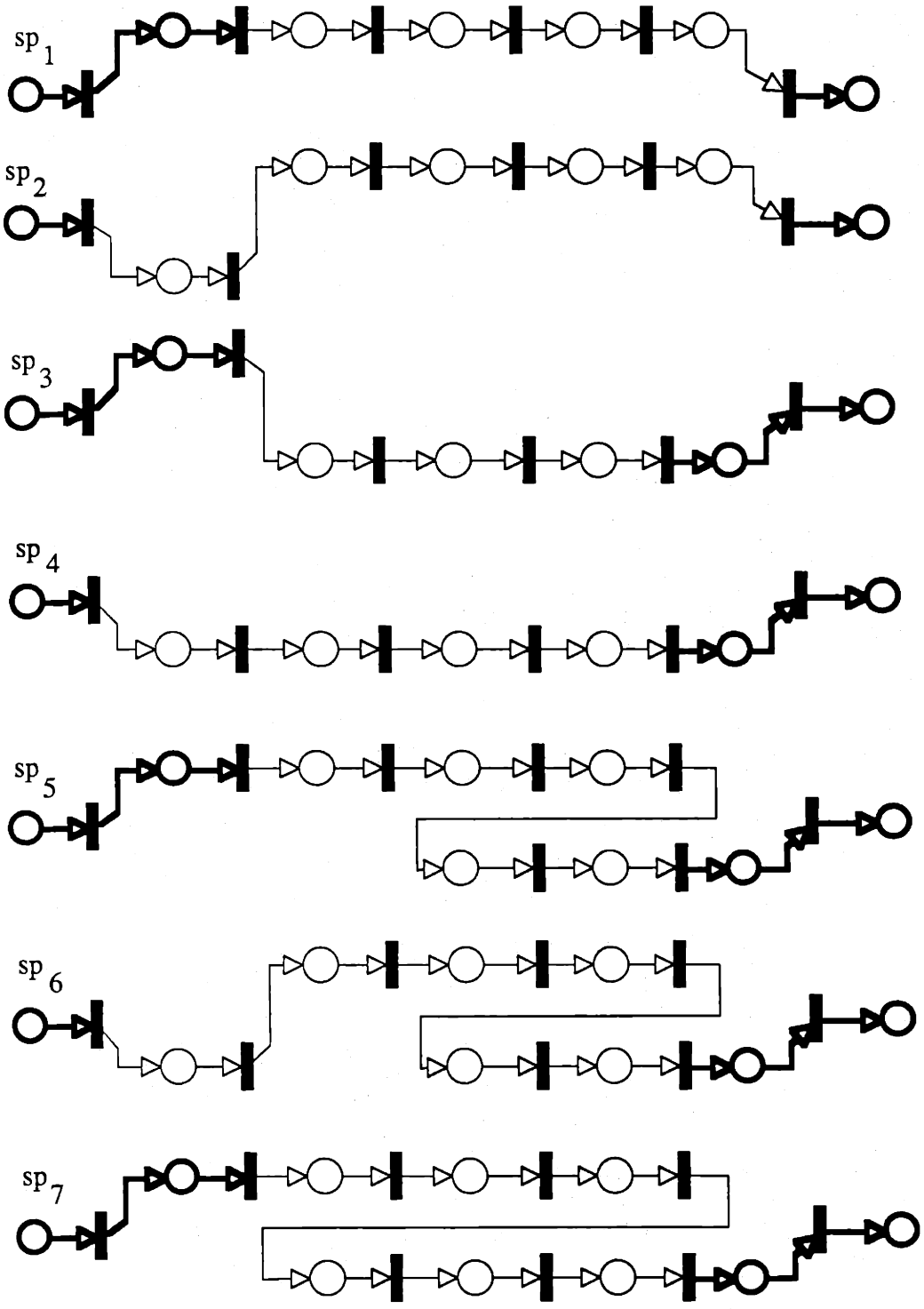


Figure 9.4: Petri Net representation of the simple paths of $\Omega(R_{u1})$.

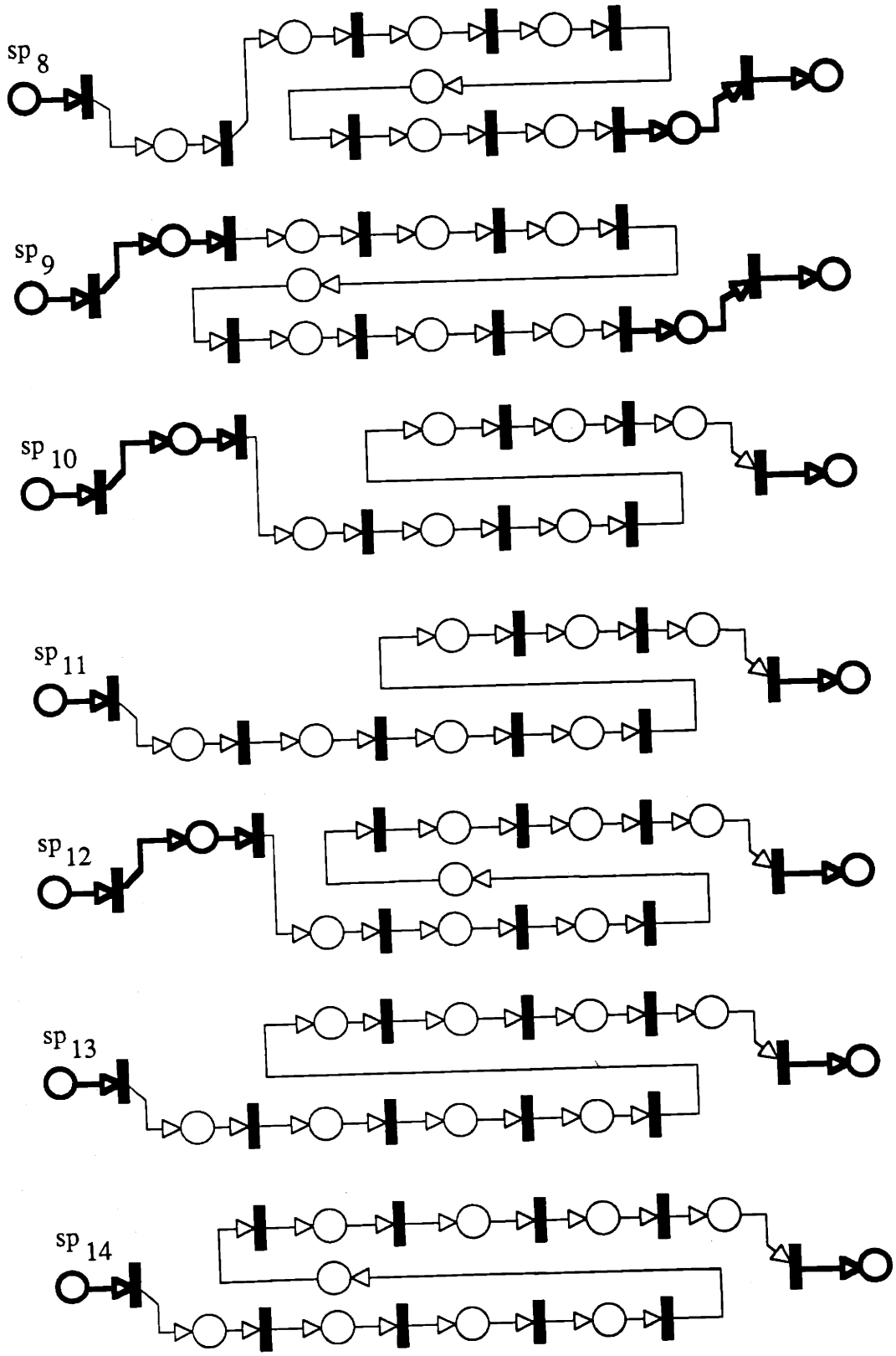


Figure 9.4 (continue): Petri Net representation of the simple paths of $\Omega(R_{u1})$.

9.1.3 MINOs and MAXOs.

The algorithm ARCGEN generates 5 MINOs and 5 MAXOs. Table 9.4 gives the vector representation of both MINOs and MAXOs. Figure 9.5 (resp. 9.6) reproduces the Petri Net representations of the MINOs (resp. MAXOs).

TABLE 9.4 VECTOR REPRESENTATION OF THE MINOS AND MAXOS.

	MINO	MAXO
	1 2 3 4 5	1 2 3 4 5
p_1 *	1 1 1 1 1	1 1 1 1 1
p_2 *	1 1 1 1 1	1 1 1 1 1
p_3	1 0 0 0 0	1 1 1 1 0
p_4	1 0 1 1 1	1 1 1 1 1
p_5	0 1 0 0 0	1 1 1 1 1
p_6	0 0 0 0 0	1 1 1 1 0
p_7	1 0 0 0 1	1 1 1 1 1
p_8	1 0 1 1 1	1 1 1 1 1
p_9	1 1 0 1 1	1 1 1 1 1
p_{10}	1 0 1 1 1	1 1 1 1 1
p_{11}	1 1 1 1 1	1 1 1 1 1
p_{12}	1 0 0 0 0	1 1 1 1 1
p_{13}	0 0 0 0 1	0 0 0 0 1
p_{14}	0 0 0 1 0	0 0 0 1 0
p_{15}	0 0 1 0 0	0 0 1 0 0
p_{16} *	1 1 1 1 1	1 1 1 1 1
p_{17}	0 0 0 0 0	0 0 0 0 0
p_{18}	0 0 0 0 0	0 1 0 0 0
p_{19}	0 0 0 0 0	1 0 0 0 0
p_{20} *	1 1 1 1 1	1 1 1 1 1

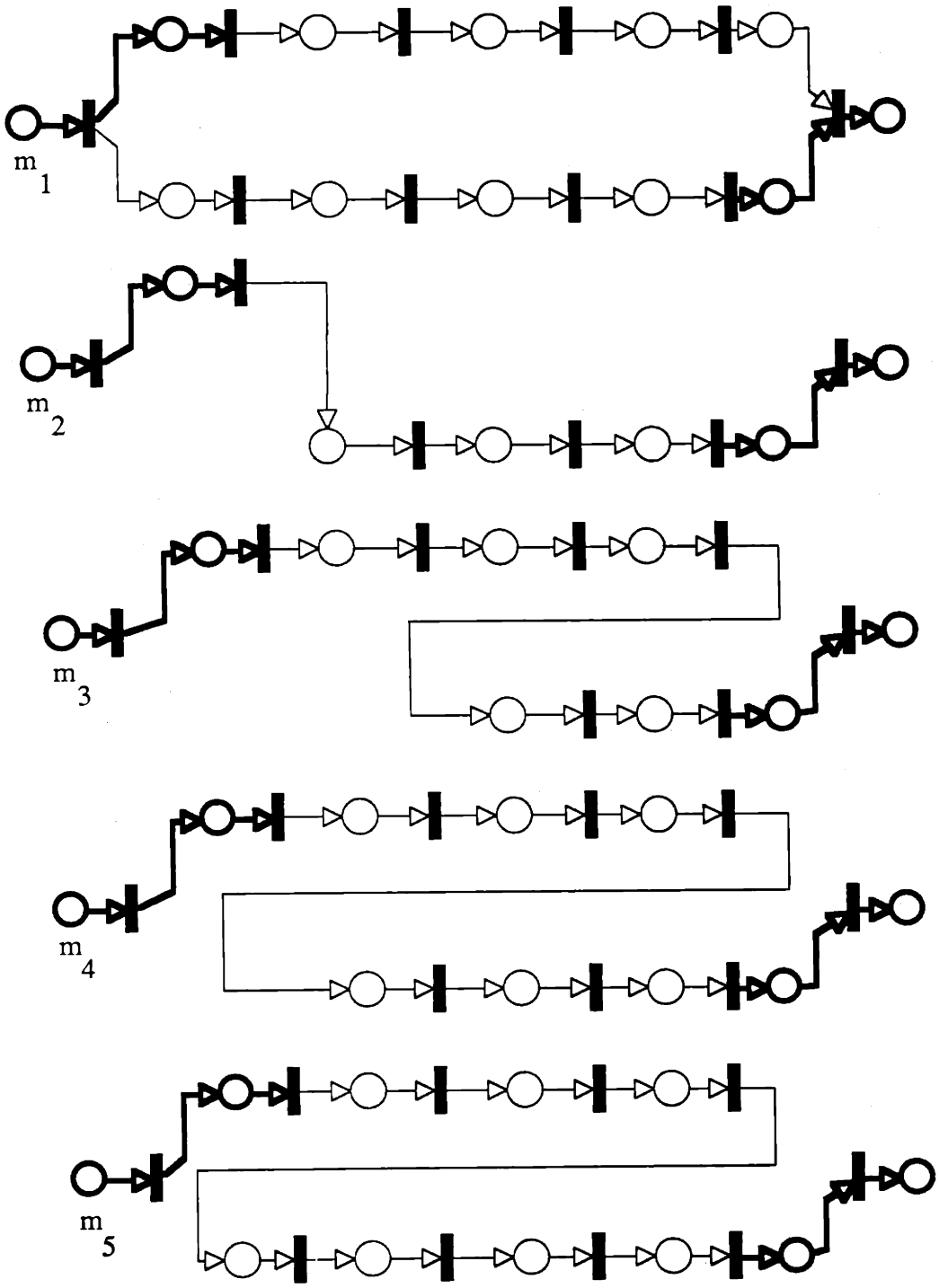


Figure 9.5: Petri Net representation of the MINOs.

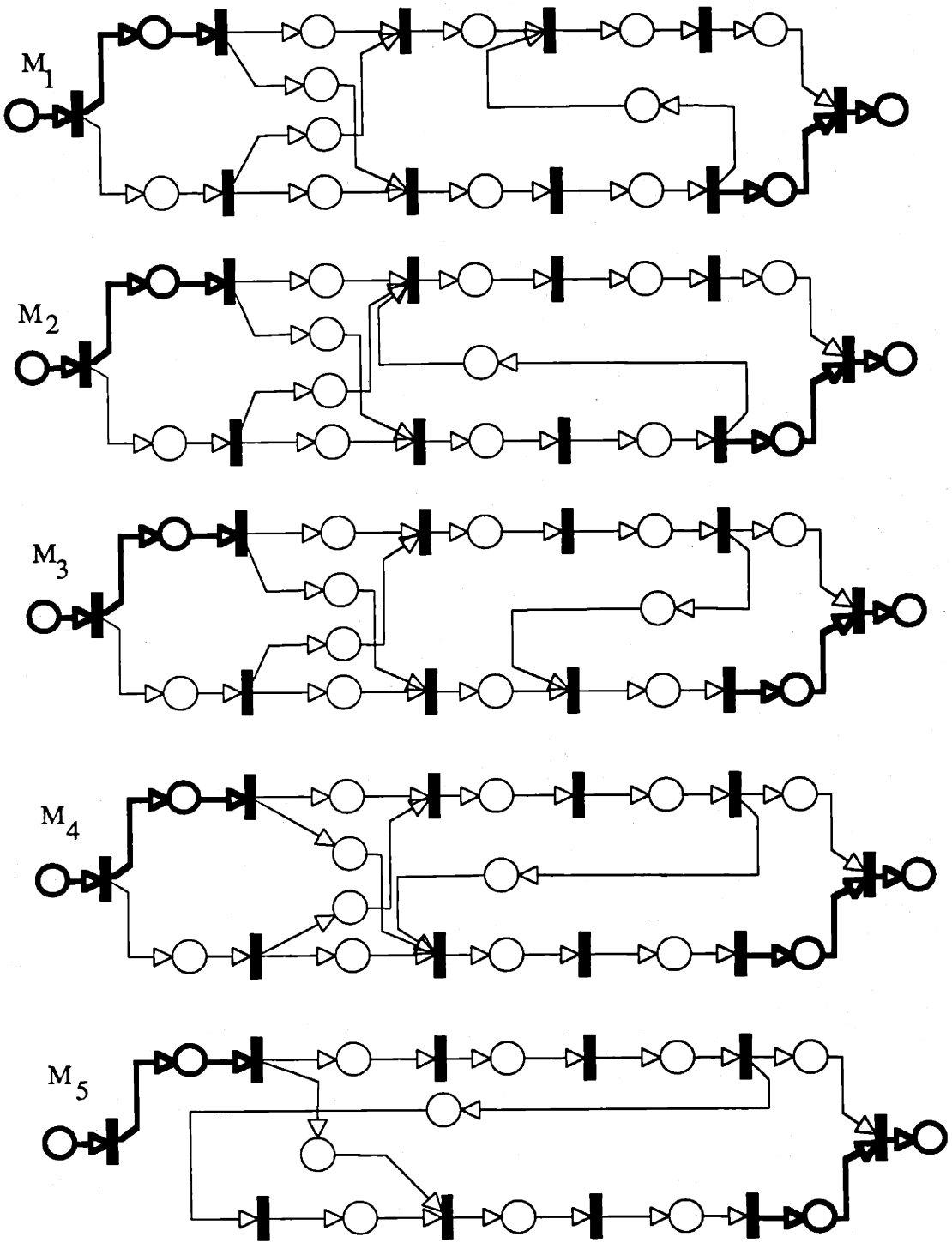


Figure 9.6: Petri Net representation of the MAXOs.

9.1.4 Considerations About Symmetry

The effect of imposing asymmetrical constraints is reflected both in the sets of MINOs and MAXOs, at different degrees however. Let σ_{12} denote the permutation operator between DM^1 and DM^2.

MINOs

Without the asymmetry in the constraints, the four MINOs m_2,m_3,m_4, and m_5 would have had counterparts obtained by reversing the roles of DM^1 and DM^2 (i.e., by application of the operator σ_{12}). Because of the asymmetry in the constraints, the set of MINOs is highly asymmetrical: m_1 is the only MINO invariant under the operator σ_{12} ($m_1 = \sigma_{12}(m_1)$).

MAXOs

M_1 and M_3, as well as M_2 and M_4, correspond to each other with respect to the permutation σ_{12} : $M_3 = \sigma_{12}(M_1)$ and $M_4 = \sigma_{12}(M_2)$. M_5 has no counterpart in the set $\Phi_{max}(R)$ of all MAXOs.

The way in which symmetries in the set of user-defined constraints are reflected in the sets of all MINOs and MAXOs is an interesting subject that would require further investigation. The above example suggests that the set of MINOs may be more sensitive to the effect of having asymmetrical constraints than the set of MAXOs.

9.1.5 Structure of the Set $\Phi(R)$.

The diagram of Figure 9.7 outlines the inclusion relation existing between the MINOs and the MAXOs. This diagram is the skeleton of the set $\Phi(R)$. Even in this very simple case the diagram of the entire set $\Phi(R)$ is very large. In Figure 9.8, all the Feasible Organizations containing the MINO m_1 are listed. Figure 9.8 represents the subset of the set $\Phi(R)$ composed of all the superordinates of m_1 .

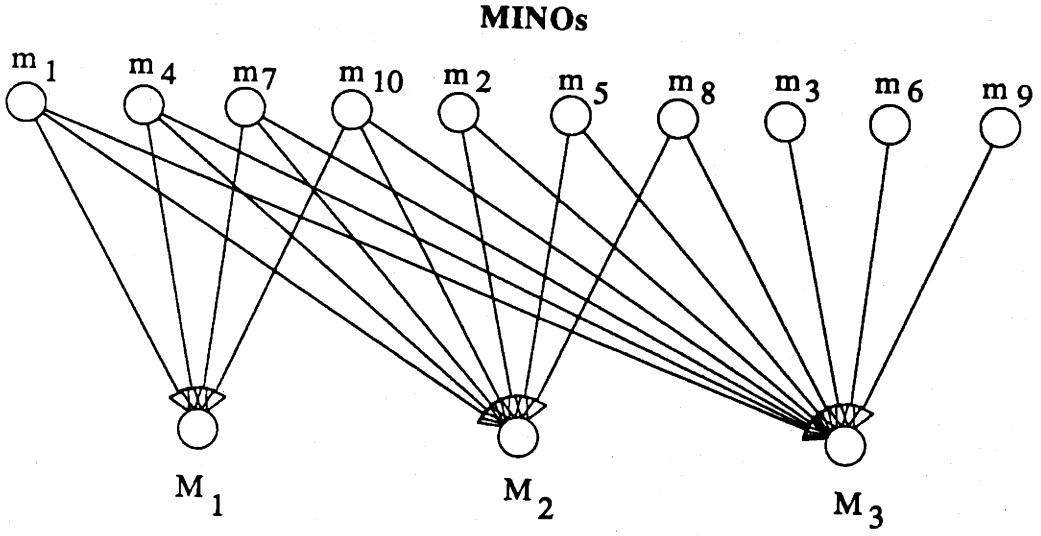


Figure 9.7: Inclusion relations between the MINOs and the MAXOs.

The entire set $\Phi(R)$ would be obtained in connecting together the five partial diagrams corresponding to the subsets of superordinates of the five MINOs. In Figure 9.8, every Feasible Organization is designated by one of its minimal generating families. Note that all the paths leading from m_1 to a given MAXO have not the same length: the set $\Phi(R)$ violates the Jordan-Dedekind chain condition (7.5.2). The bold face links in Figure 9.8 represent the minimal length paths between any two nodes (whenever paths from different length are present).

At the outset, the total number of WDSs satisfying the user-defined constraints was 1,024. Althougth the net $\Omega(R_{u1})$ has 14 simple paths, the number of elements of $USp(R_{u1})$ is much less than 2^{14}. Indeed, the MAXOs have all a complexity less than 4. Consequently, any element of $USp(R_{u1})$ can be obtained as the join of at most 4 simple paths. The cardinal of $USp(R_{u1})$ is therefore bounded by:

$$C_{14}^{4} = \frac{14!}{10! * 4!} = 13 * 11 * 7 = 1{,}001.$$

The exact number of elements of $USp(R_{u1})$ is very likely to be much less than 1,001, since redundancy is still present among the combinations of at most four simple paths. A rough estimate of the number of elements of the set $\Phi(R)$ can be infered from the

diagram of Figure 9.8. Since this diagram has 20 elements and since the set $\Phi(R)$ is obtained by juxtaposition of five such diagrams, one can expect $\Phi(R)$ to have around 100 elements. Once again, the exact number of elements of $\Phi(R)$ is most probably well below 100, because the same organization can be obtained from two different MINOs, thus creating redundancy.

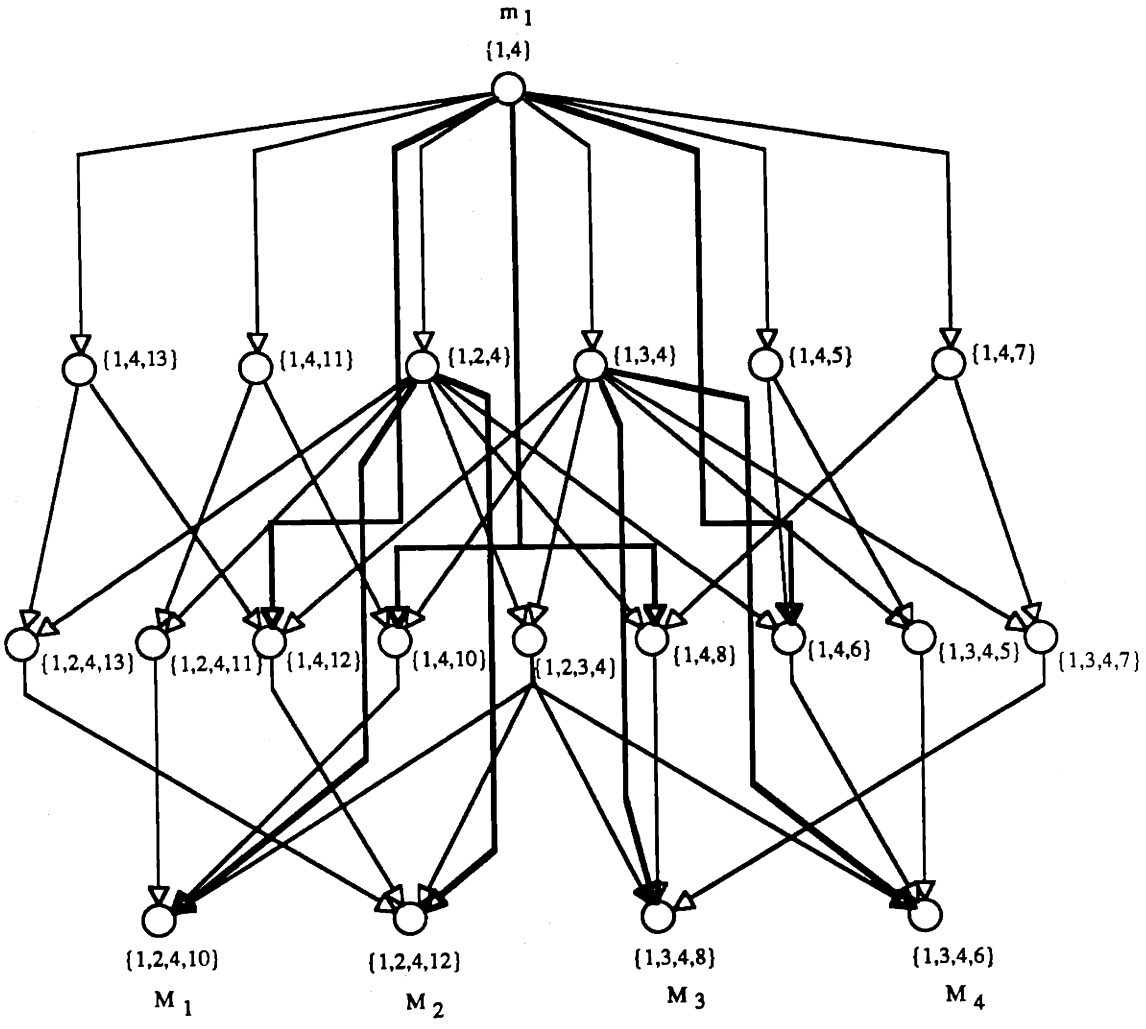


Figure 9.8: Diagram of all the superordinates of the MINO m_1.

9.2 THE WARFARE COMMANDER PROBLEM

9.2.1 Description of the Problem

Let us consider a surface ship designed to handle three different kinds of threats: air, surface and underwater attacks. The ship disposes of three different types of weapons: missiles, torpedoes and depth charges. To identify the threats, two detection devices are available: a radar and a sonar. Each of these devices will be monitored by a single operator. The Sonar Operator (SO) will be able to detect both underwater and surface threats, while the Radar Operator (RO) will be able to detect surface as well as air threats. Each operator has therefore his reserved area of competence: an air attack can be detected by the radar operator only , while a submarine attack can be identified by the sonar operator only. The two operators share, however, a common area of competence: a surface attack can be detected be either of them. The weapon system of the ship is controlled in a similar way by two operators : the Anti-Air Warfare Commander (AAW) is in charge of the missiles, while the Anti-Submarine Warfare Commander (ASW) controls the torpedoes and the depth charges. As it was the case for the SO and the RO, AAW and ASW have their own reserved intervention area: an air attack can be handled by AAW only, while a submarine attack can be handled by ASW only. A surface attack, however, is a common area of intervention for the two weapon operators: either a missile or a torpedoe can be used to engage a surface ship.

Once a threat is detected, the relevant information and commands need to be sent to the weapon operators. The Executive Coordinator (EXCO) could play this role. He could receive information from RO and SO and send his commands to AAW and ASW. The executive coordinator **has to** receive information from either RO or SO and has to send orders to either AAW or ASW. However, he **need not** receive information from both RO and SO, nor need he send commands to both AAW and ASW. If the ship is facing an attack from an individual airplane for instance, information coming from the sonar operator is of little interest to the coordinator. Similarly, the coordinator will just send orders to AAW , since torpedoes and depth charges are of little use against an airplane.

The organization under consideration in this section has therefore five members: RO, SO, EXCO, AAW, and ASW.

Decisionmakers RO and SO act as the sensors of the organization. They will be denoted by DM^1 and DM^2. They both receive information from the external environment. They may or may not share this information with each other and with the other members of the organization. One of them however, has to send this information to the coordinator. At the other end of the organization, AAW and ASW act as actuators. They will be denoted by DM^4 and DM^5.They are both directly related to the external environment. They will receive orders from the coordinator and they may receive information from RO and SO. They may also share their results, i.e., the concrete action they are taking, with each other and with the coordinator. The coordinator will be denoted DM^3.

The problem is to design interactional structures between the five members of the organization, that will faithfully reflect the situation described above. The first step consists in translating the above description into user-defined constraints. This is done in detail in the following subsection.

9.2.2 User-defined Constraints.

One needs to specify the links that must be present in all cases, i.e., the fixed links, as well as the links that are permissible, but not compulsory. The latter will constitute the degrees of freedom of the design.

Fixed links

Since DM^1 and DM^2 acts as the sensors of the organization, they must receive information from the external environment. This constraint is reflected in the vector $\underline{e}$:

$$e_1 = e_2 = 1.$$

Similarly, DM^4 and DM^5 act as the actuators of the organization and must send information to the external environment. The vector $\underline{s}$ will reflect this constraint:

$$s_4 = s_5 = 1.$$

Lastly, it was stated in 9.2.1 that the coordinator DM^3 must receive information from DM^1 or DM^2 and must send information to either DM^4 or DM^5. It will be assumed that DM^1 always sends information to DM^3 and that DM^3 always sends commands to DM^4. In doing so, the symmetry between DM^1 and DM^2 and between DM^4 and DM^5 is somewhat broken. As pointed out in section 9.1 where the first example is studied, to brake

the symmetry between the decisionmakers in the user-defined constraints has two beneficial effects: it eliminates some redundancy in the sets of MINOs and MAXOs, and it allows for a study of the way asymmetry diffuses through the methodology. As subsection 9.2.3 will show, organizations symmetrical with respect to DM^1 and DM^2 as well as DM^4 and DM^5 will still be generated. Moreover DM^1 has not been specifically defined: DM^1 can be either the sonar or the radar operator, so that a given organization can be interpreted either way. Similarly, DM^4 can be either AAW or ASW. This point will be made more explicit in subsection 9.2.5, when results are interpreted.

The fixed link between DM^3 and DM^4 will be represented by $C_{34} = 1$. As far as the fixed link between DM^1 and DM^3 is concerned, two cases are possible: either a link from the SA stage of DM^1 to the IF stage of DM^3, or a link from the RS stage of DM^1 to the IF stage of DM^3. Both links express the sharing of information between DM^1 and DM^3. The latter will be used because it increases the flexibility of the design in allowing for the following configuration: DM^2 shares his situation assessment with DM^1 and DM^1 sends the fused information to DM^3. Figure 9.9 illustrates this configuration.

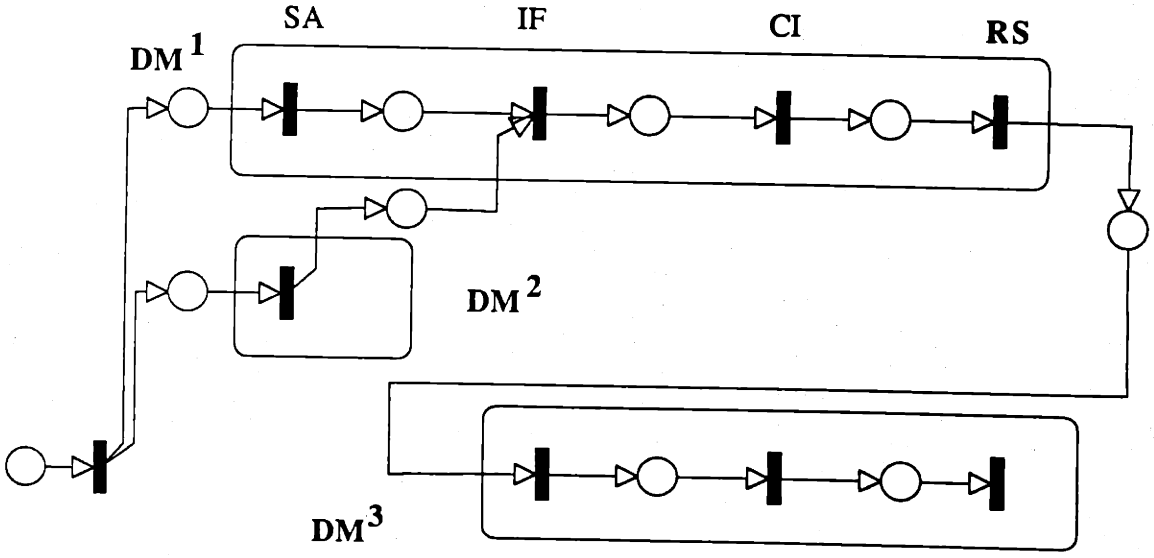


Figure 9.9: DM^1 sends information to DM^3 from his RS stage.

This configuration is not possible if DM^1 sends information to DM^3 directly from his SA stage: DM^1 is denied the opportunity to take into account the situation assessed by DM^2 and to send an aggregate signal to DM^3. As illustrated in Figure 9.9, DM^1 will have four stages. It will be seen that DM^2 may also have four stages. Since DM^1 and DM^2 act as sensors only, the CI and RS stages of these two decisionmakers will be dummy stages in the sense that they perform no action whatsoever on the incoming signal, but provide a mere transmission of this signal. In other words, the signal issued by the RS stage of DM^1 or DM^2 is the same that the signal issued by the IF stage of the decisionmaker. The fixed link between DM^1 and DM^3 will therefore be represented by $H_{13} = 1$.

Unspecified elements

• DM^1 and DM^2 are allowed to exchange their situation assessments: F_{12} and F_{21} will be unspecified.

• DM^1 and DM^2 are allowed to send information to all other members of the organization. The same reasoning used to justify the choice of the fixed link between DM^1 and DM^3 applies here. More flexibility is obtained if DM^1 and DM^2 issue information to the other members from their RS stages rather than their SA stages. Once again, this allows DM^1 and DM^2 to first share their situation assessments, and then send an aggregate signal to the other members. Therefore H_{14}, H_{15}, H_{23}, H_{24}, and H_{25} will be unspecified.

• DM^4 and DM^5 can share their results with each other and with DM^3. Therefore, H_{43}, H_{45}, H_{53}, and H_{54} will be unspecified.

• DM^3 can send commands to DM^5: C_{35} will be unspecified.

All the elements that are neither fixed nor unspecified are set to 0. The arrays defining the user-defined constraints R_{u2} are represented in Figure 9.10.

$$e = [\,1\ 1\ 0\ 0\ 0\,] \qquad F = \begin{bmatrix} \# & x & 0 & 0 & 0 \\ x & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & \# \end{bmatrix} \qquad G = \begin{bmatrix} \# & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & \# \end{bmatrix}$$

$$s = [\,0\ 0\ 0\ 1\ 1\,] \qquad H = \begin{bmatrix} \# & 0 & 1 & x & x \\ 0 & \# & x & x & x \\ 0 & 0 & \# & 0 & 0 \\ 0 & 0 & x & \# & x \\ 0 & 0 & x & x & \# \end{bmatrix} \qquad C = \begin{bmatrix} \# & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 \\ 0 & 0 & \# & 1 & x \\ 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & \# \end{bmatrix}$$

Figure 9.10: User-defined constraints R_{u2}.

The number of undetermined elements is equal to 12. Note that if all undetermined elements are set to 1, the above arrays are symmetrical with respect to DM^1 and DM^2, as well as DM^4 and DM^5. In the sequel, the transposition operator between DM^i and DM^j will be denoted σ_{ij}. The next section analyzes the nets $\Omega(R_{u2})$ and $\omega(R_{u2})$.

9.2.3 Nets $\Omega(R_{u2})$ and $\omega(R_{u2})$.

The net $\Omega(R_{u2})$ is obtained in setting to 1 all the unspecified elements of the arrays defining the user-defined constraints R_{u2}. Note that $\Omega(R_{u2})$ is invariant under the operators σ_{12} and σ_{45}. The net $\Omega(R_{u2})$ is represented in Figure 9.11. Places are labeled sequentially but transitions are labeled according to the algorithmic technique. Figure 9.11 corresponds to representation (2) of a WDN as presented in 5.4.4: a WDN is characterized by the graph of its Petri Net representation with the associated labeling of the transitions. Table 9.5 gives the correspondence between the sequential and the algorithmic labelings of the places.

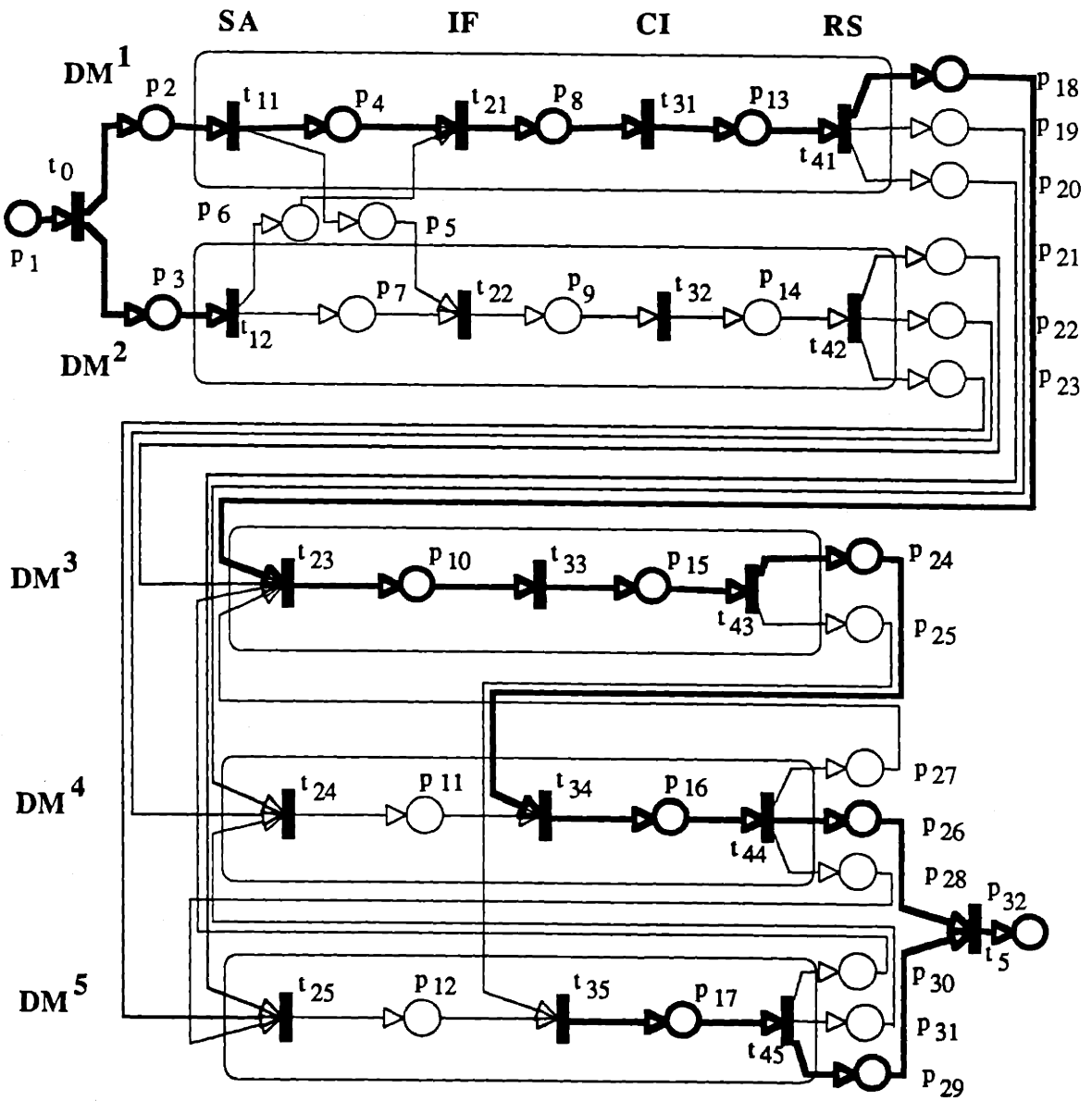


Figure 9.11: Net $\Omega(R_{u2})$.

TABLE 9.5 SEQUENTIAL VS ALGORITHMIC LABELING OF THE PLACES

sequential label → algorithmic label

$P_1 \rightarrow P_0$	$P_2 \rightarrow P_{11}$	$P_3 \rightarrow P_{12}$
$P_4 \rightarrow P_{21}$	$P_5 \rightarrow P_{212}$	$P_6 \rightarrow P_{221}$
$P_7 \rightarrow P_{22}$	$P_8 \rightarrow P_{31}$	$P_9 \rightarrow P_{32}$
$P_{10} \rightarrow P_{33}$	$P_{11} \rightarrow P_{34}$	$P_{12} \rightarrow P_{35}$
$P_{13} \rightarrow P_{41}$	$P_{14} \rightarrow P_{42}$	$P_{15} \rightarrow P_{43}$
$P_{16} \rightarrow P_{44}$	$P_{17} \rightarrow P_{45}$	$P_{18} \rightarrow P_{5132}$
$P_{19} \rightarrow P_{5142}$	$P_{20} \rightarrow P_{5152}$	$P_{21} \rightarrow P_{5232}$
$P_{22} \rightarrow P_{5242}$	$P_{23} \rightarrow P_{5252}$	$P_{24} \rightarrow P_{5343}$
$P_{25} \rightarrow P_{5353}$	$P_{26} \rightarrow P_{54}$	$P_{27} \rightarrow P_{5432}$
$P_{28} \rightarrow P_{5452}$	$P_{29} \rightarrow P_{55}$	$P_{30} \rightarrow P_{5532}$
$P_{31} \rightarrow P_{5542}$	$P_{32} \rightarrow P_6$	

The incidence matrix, as computed by the program ARCGEN, is reproduced in Figure 9.12. As it was the case for the graph representation of $\Omega(R_{u2})$, places are labeled sequentially and transitions are labeled according to the internal technique of the algorithm.

Net $\omega(R_{u2})$

This net is obtained in replacing by 0 all unspecified elements of the arrays of Figure 9.10. It is represented in Figure 9.13. By definition, its places are all fixed places. Note that there are six interactional fixed places corresponding to the six fixed links of subsection 9.2.2, i.e. the six 1's in the arrays of Figure 9.8. Those places are p_2, p_3, p_{18}, p_{24}, p_{26}, and p_{29}. The remaining fixed places are internal places induced by the presence of the interactional fixed places. As an example, p_4, p_8 and p_{10} are all internal places of DM^1. The presence of these places is made compulsory because DM^1 receives information from the external environment (p_2) and sends information from his RS stage (p_{18}). Since p_2 and

p_{18} are fixed places, p_4, p_8 and p_{10} are also fixed places (see the rules presented in 5.2.2 to construct internal places from interactional places).

	t_0	t_{11}	t_{12}	t_{21}	t_{22}	t_{23}	t_{24}	t_{25}	t_{31}	t_{32}	t_{33}	t_{34}	t_{35}	t_{41}	t_{42}	t_{43}	t_{44}	t_{45}	t_5
p_1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_2	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_3	1	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_4	0	1	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_5	0	1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_6	0	0	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_7	0	0	1	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_8	0	0	0	1	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0
p_9	0	0	0	0	1	0	0	0	0	-1	0	0	0	0	0	0	0	0	0
p_{10}	0	0	0	0	0	1	0	0	0	0	-1	0	0	0	0	0	0	0	0
p_{11}	0	0	0	0	0	0	1	0	0	0	0	-1	0	0	0	0	0	0	0
p_{12}	0	0	0	0	0	0	0	1	0	0	0	0	-1	0	0	0	0	0	0
p_{13}	0	0	0	0	0	0	0	0	1	0	0	0	0	-1	0	0	0	0	0
p_{14}	0	0	0	0	0	0	0	0	0	1	0	0	0	0	-1	0	0	0	0
p_{15}	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	-1	0	0	0
p_{16}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	-1	0	0
p_{17}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	-1	0
p_{18}	0	0	0	0	0	-1	0	0	0	0	0	0	0	1	0	0	0	0	0
p_{19}	0	0	0	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0	0
p_{20}	0	0	0	0	0	0	0	-1	0	0	0	0	0	1	0	0	0	0	0
p_{21}	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	1	0	0	0	0
p_{22}	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	1	0	0	0	0
p_{23}	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0
p_{24}	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	1	0	0	0
p_{25}	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	1	0	0	0
p_{26}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	-1
p_{27}	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	1	0	0
p_{28}	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	1	0	0
p_{29}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-1
p_{30}	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	1	0
p_{31}	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	1	0
p_{32}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 9.12: Incidence matrix of $\Omega(R_{u2})$.

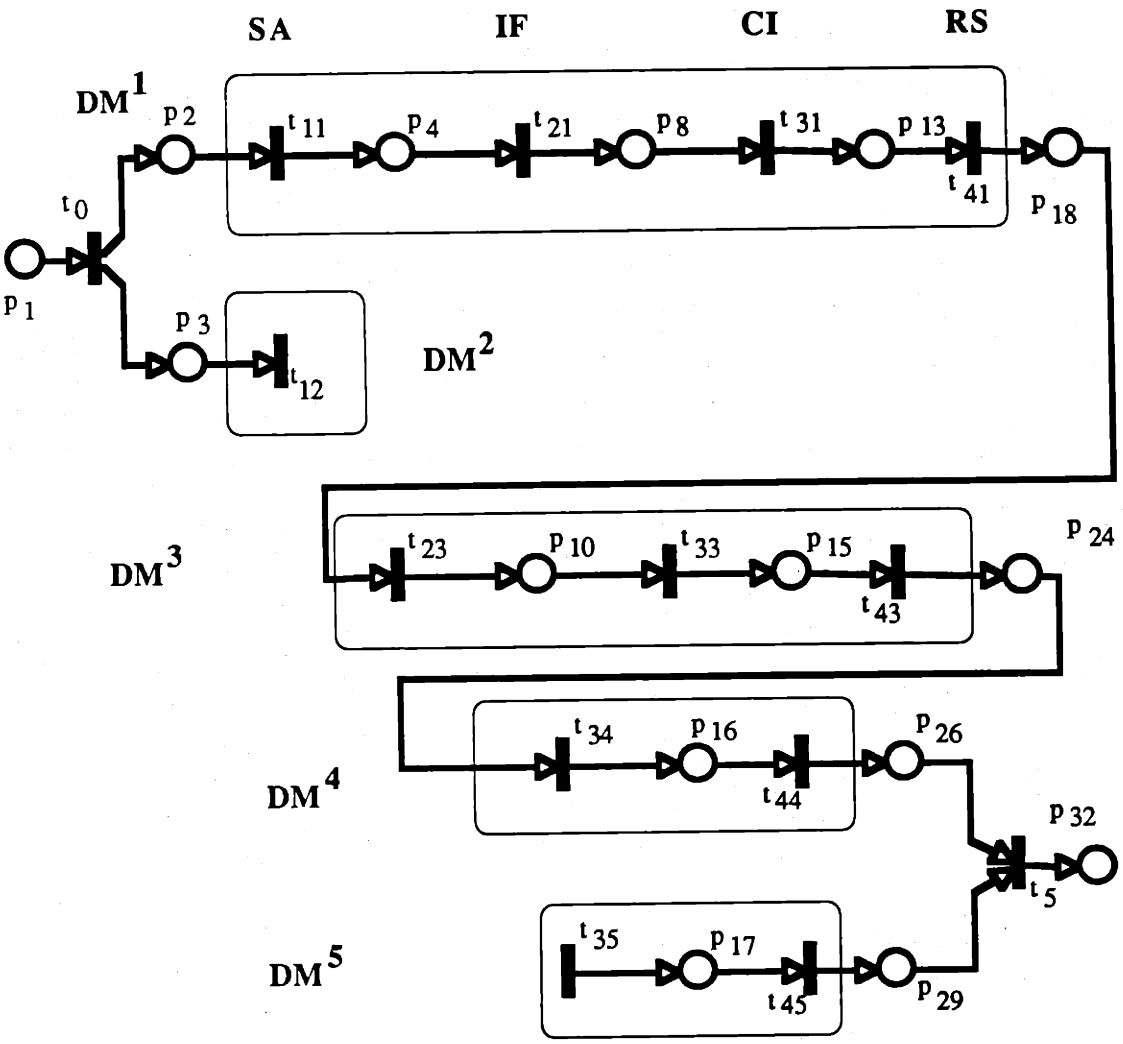


Figure 9.13: Net $\omega(R_{u2})$.

Simple paths of $\Omega(R_{u2})$.

The algorithm ARCGEN generates 40 simple paths for the net $\Omega(R_{u2})$. They are reproduced in their vector representation in Table 9.6. Fixed places, corresponding to the places of $\omega(R_{u2})$, are singled out with an asterisk. Note that simple path number 5 includes only fixed places. This simple path need therefore be included in every Admissible Organizational Form and a fortiori in every Feasible Organizations.

TABLE 9.6 VECTOR REPRESENTATION OF THE SIMPLE PATHS OF $\Omega(R_{u2})$.

places	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	*	*	*	*				*		*			*		*	*		*						*		*			*			*
path #																																
1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1
2	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1
3	1	1	0	0	1	0	0	0	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1
4	1	0	1	0	0	0	1	0	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1
5	1	1	0	1	0	0	0	1	0	1	0	0	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1
6	1	0	1	0	0	1	0	1	0	1	0	0	1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1
7	1	1	0	0	1	0	0	0	1	1	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	1
8	1	0	1	0	0	0	1	0	1	1	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	1
9	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1
10	1	0	1	0	0	1	0	1	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1
11	1	1	0	0	1	0	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
12	1	0	1	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
13	1	1	0	1	0	0	0	1	0	1	0	0	1	0	1	0	1	1	0	0	0	0	0	1	0	0	0	1	0	0	1	
14	1	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	0	0	0	0	0	1	0	0	0	1	0	0	1	
15	1	1	0	0	1	0	0	0	1	1	0	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1
16	1	0	1	0	0	0	1	0	1	1	0	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1
17	1	1	0	1	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1
18	1	0	1	0	0	1	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1
19	1	1	0	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1
20	1	0	1	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1
21	1	1	0	1	0	0	0	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	0	0	1
22	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	0	0	1
23	1	1	0	0	1	0	0	0	1	1	0	1	0	1	1	1	1	1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1
24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1
25	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	0	1	0	0	0	0	0	1	0	1	0	1	0	0	1
26	1	0	1	0	0	1	0	1	0	1	1	0	1	0	1	1	1	1	0	1	0	0	0	0	0	1	0	1	0	1	0	0	1
27	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1
28	1	0	1	0	0	0	1	0	1	1	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1
29	1	1	0	1	0	0	0	1	0	0	1	1	1	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1
30	1	0	1	0	0	1	0	1	0	0	1	1	1	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1
31	1	1	0	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1	1
32	1	0	1	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1	1
33	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1
34	1	0	1	0	0	1	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1
35	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1
36	1	0	1	0	0	0	1	0	1	1	1	0	0	1	1	1	1	1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1
37	1	1	0	1	0	0	0	1	0	1	0	1	1	0	1	1	1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	1	
38	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	1	1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	1	
39	1	1	0	0	1	0	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	1	0	1	
40	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	1	0	1	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	*	*	*	*				*		*			*		*	*		*						*		*			*			*

9.2.4 MINOs and MAXOs

The algorithm ARCGEN generates 10 MINOs and 3 MAXOs. Their vector representation is given in Table 9.7. Figure 9.14 (resp. 9.15) reproduces the graph representations of the MINOs (resp. MAXOs).

TABLE 9.7 VECTOR REPRESENTATION OF THE MINOS AND MAXOS.

	MINOs										MAXOs		
	1	2	3	4	5	6	7	8	9	10	1	2	3
p_1 *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_2 *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_3 *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_4 *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_5	0	0	0	0	0	0	0	0	0	0	1	1	1
p_6	0	0	0	1	1	1	0	0	0	0	1	1	1
p_7	1	1	1	0	0	0	1	1	1	1	1	1	1
p_8 *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_9	1	1	1	0	0	0	1	1	1	1	1	1	1
p_{10} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{11}	1	1	1	0	0	0	0	0	0	0	1	1	1
p_{12}	1	0	1	1	0	1	1	0	1	1	1	1	1
p_{13} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{14}	1	1	1	0	0	0	1	1	1	1	1	1	1
p_{15} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{16} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{17}	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{18} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{19}	0	0	0	0	0	0	0	0	0	0	1	1	1
p_{20}	1	0	0	1	0	0	1	0	0	0	1	1	1
p_{21}	0	0	0	0	0	0	1	1	1	0	1	1	1
p_{22}	1	1	1	0	0	0	0	0	0	0	1	1	1
p_{23}	0	0	0	0	0	0	0	0	0	1	1	1	1
p_{24} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{25}	0	1	0	0	1	0	0	1	0	0	0	1	1
p_{26} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{27}	0	0	0	0	0	0	0	0	0	0	0	0	0
p_{28}	0	0	1	0	0	1	0	0	1	0	0	0	1
p_{29} *	1	1	1	1	1	1	1	1	1	1	1	1	1
p_{30}	0	0	0	0	0	0	0	0	0	0	1	0	0
p_{31}	0	0	0	0	0	0	0	0	0	0	1	1	0
p_{32} *	1	1	1	1	1	1	1	1	1	1	1	1	1

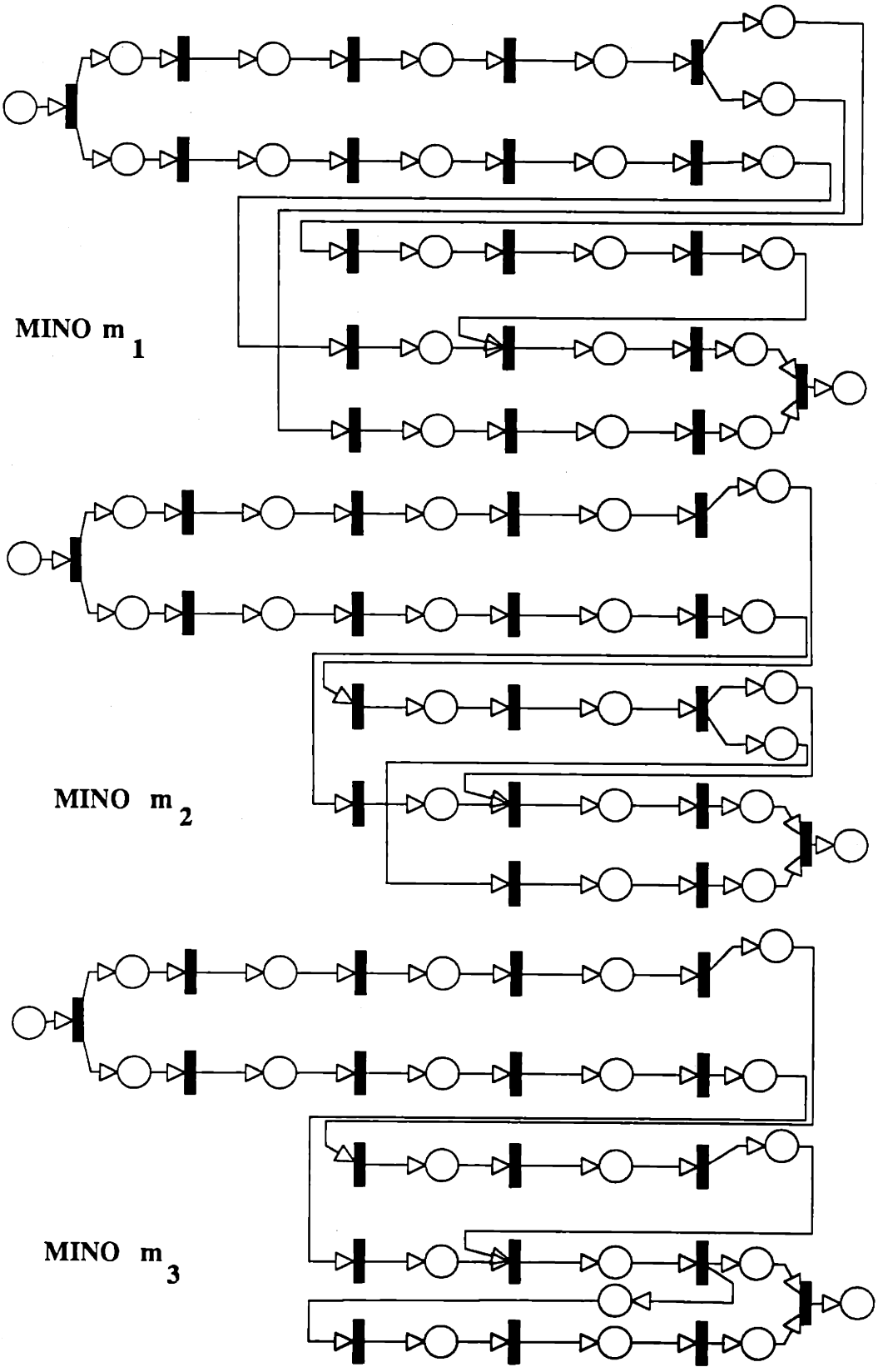


Figure 9.14: Graph representation of the MINOs.

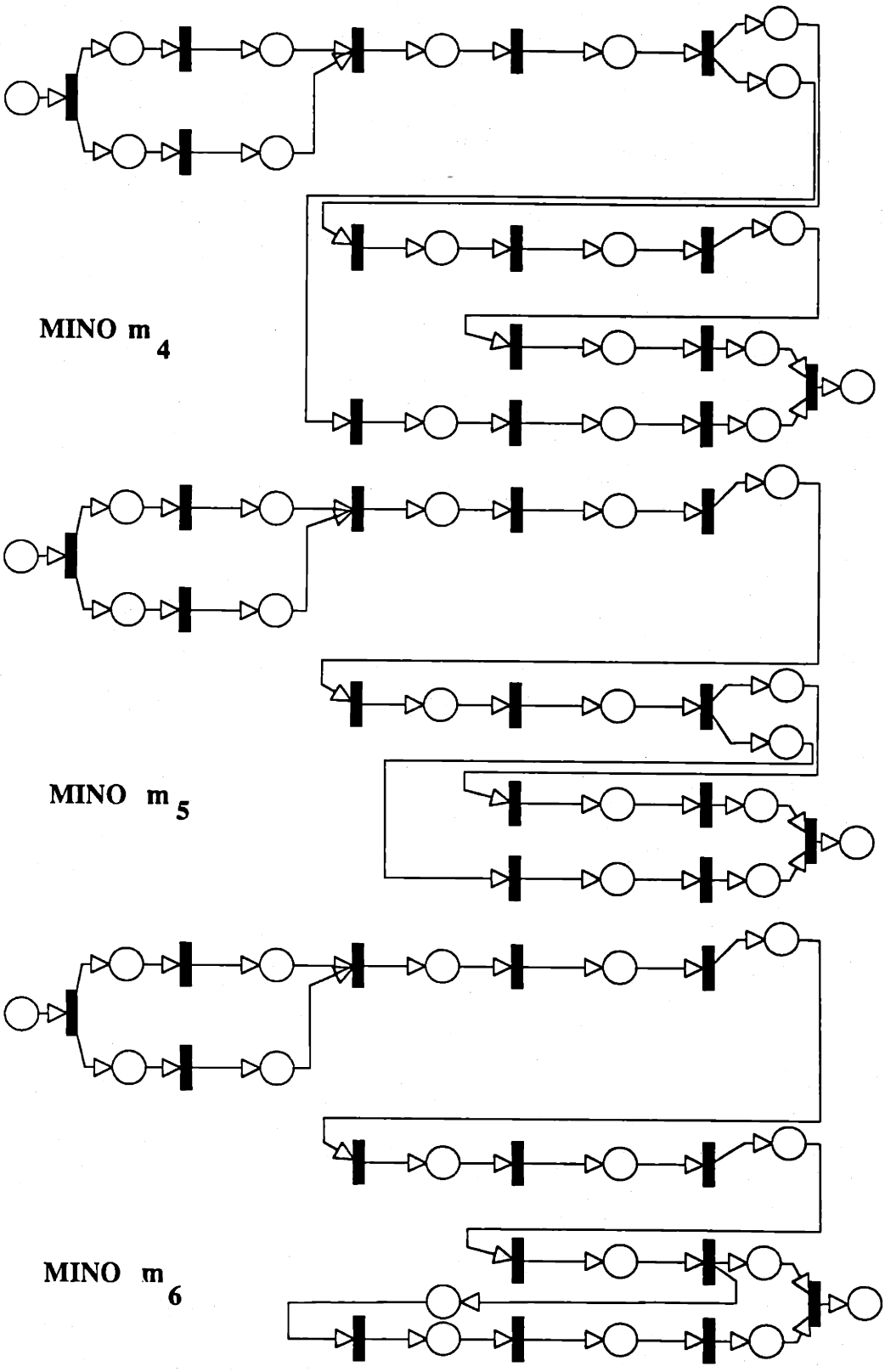


Figure 9.14 (continue): Graph representation of the MINOs.

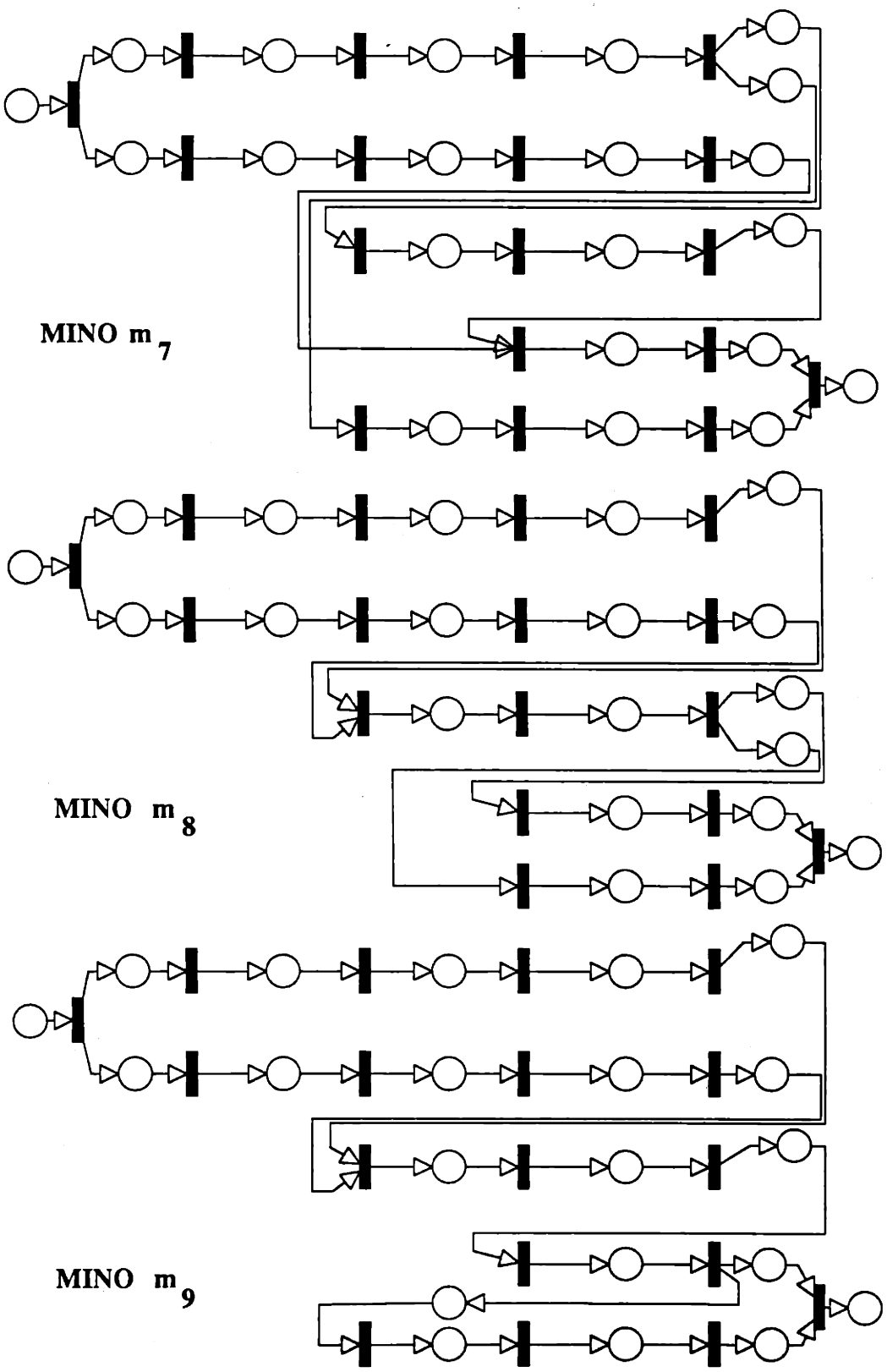


Figure 9.14 (continue): Graph representation of the MINOs.

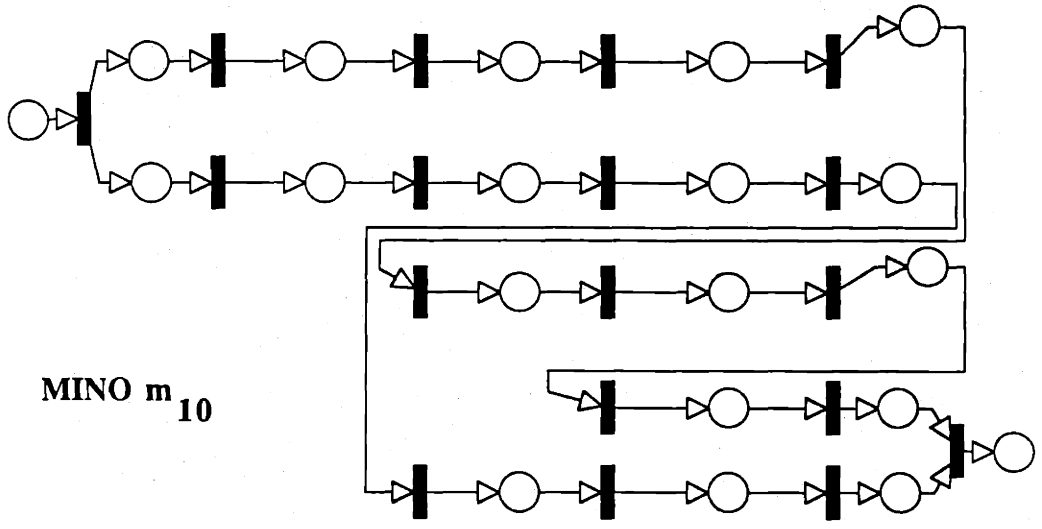


Figure 9.14 (continue): Graph representation of the MINOs.

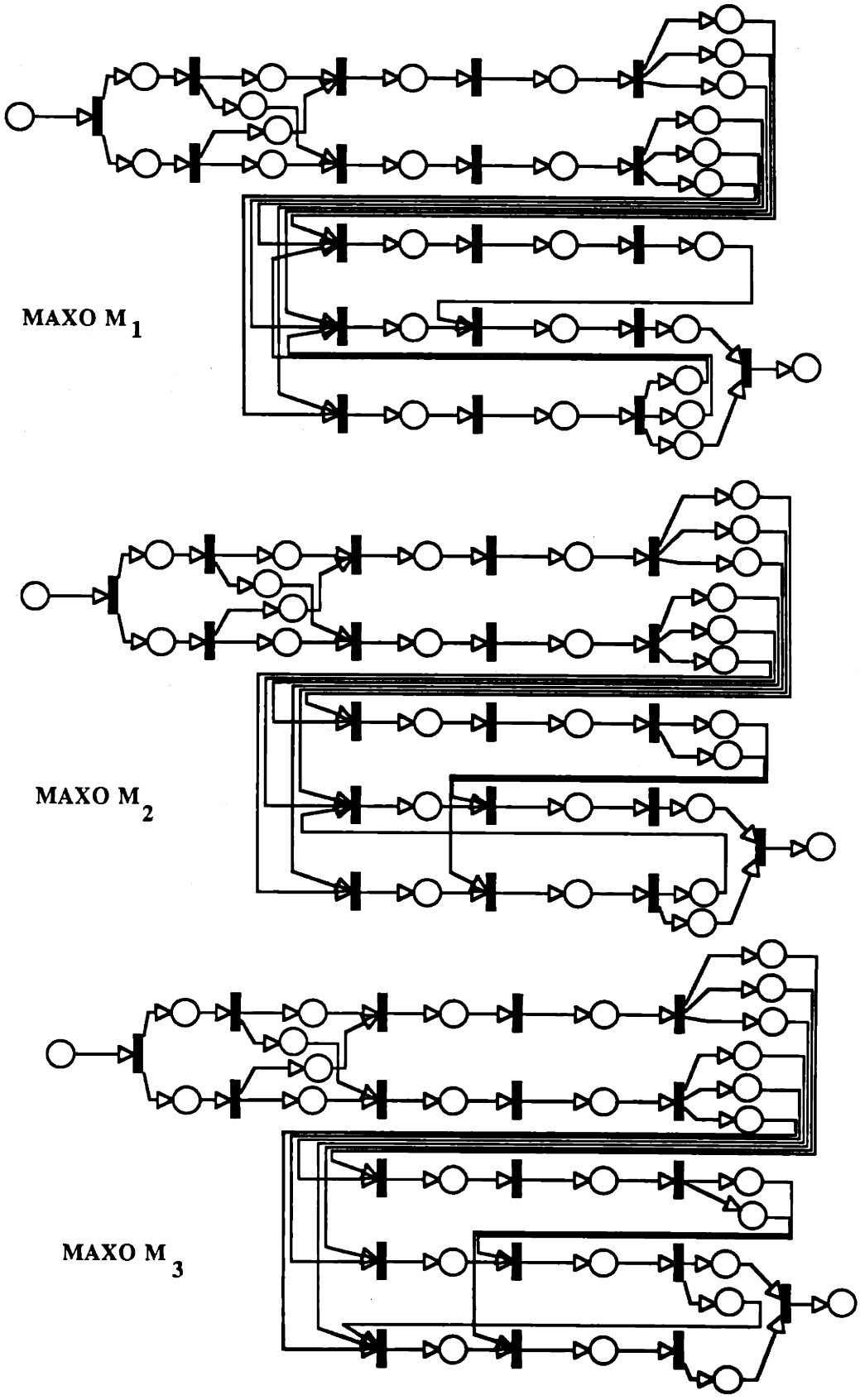


Figure 9.15: Graph representation of the MAXOs.

9.2.5 Structure of the Set $\Phi(R)$.

The inclusion relation between MINOs and MAXOs is represented in Figure 9.16. This graph is the skeleton of the diagram of $\Phi(R)$. Note that the set $\Phi(R)$ seems to be divided into three groups of organizations. One group is related to the MAXO M_3 only and originates from the MINOs m_3, m_6 and m_9. Another group is related to the MAXOs M_3 and M_2 and originates from the MINOs m_2, m_5, and m_8. Lastly, a third group is related to the three MAXOs and originates from the MINOs m_1, m_4, m_7, and m_{10}. Therefore, MAXO M_3 can be reached from every MINO, while MAXOs M_1 and M_2 can only be reached from specific MINOs. This division of the set $\Phi(R)$ into categories would require further theoretical development before any meaningful result could be derived. It will not be investigated further here.

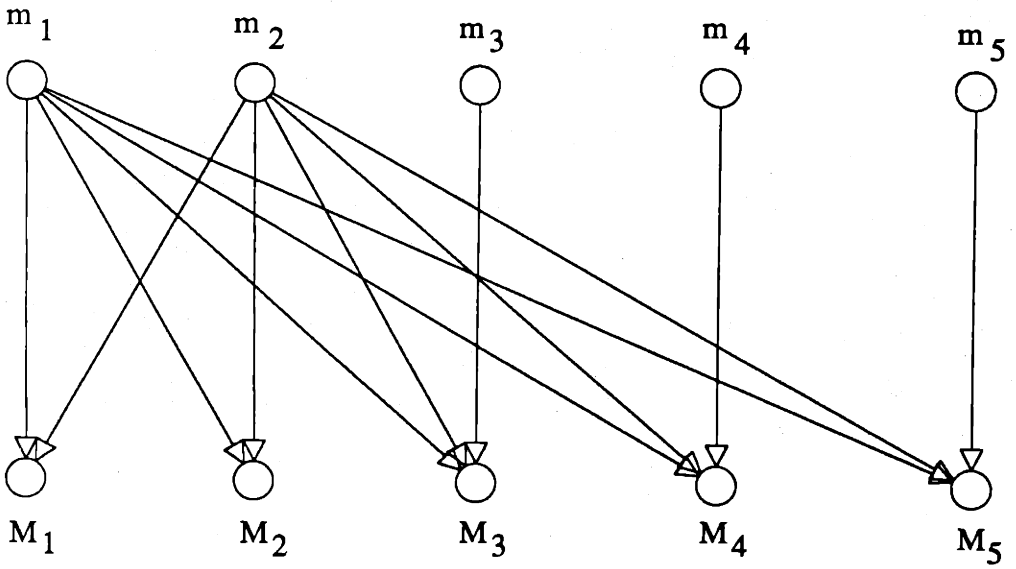


Figure 9.16: Skeleton of the diagram of the set $\Phi(R)$.

The complete diagram of $\Phi(R)$ would be obtained in making explicit the links connecting every pair (MINO,MAXO) of Figure 9.16. In other words, one would need to determine all the organization chains existing between every pair (MINO,MAXO). Those chains may have different lengths since $\Phi(R)$ violates the Jordan-Dedekind chain condition. The minimal length of a chain between a given MINO and a given MAXO is obtained by looking at the complexity (as defined in 7.4.3) of the two organizations. The complexity of all the MINOs is equal to 2: any of the ten MINOs can be obtained as the meet of two simple paths. The MAXOs M_1, M_2, and M_3 have respectively a complexity of 7,6,and 6. The minimum lenth of a chain between any MINO and M_1 is therefore 6 (7-2+1), while the minimum length of a chain between any MINO and M_2 or M_3 is 5. At the opposite end of the spectrum, the longest chain leading from a MINO to a MAXO will be obtained in adding to the MINO simple paths such that the minimum number of new interactional links is created at each step. Let us for instance consider the MINO m_{10} and the MAXO M_2. M_2 has 8 more interactional places than m_{10}. In this case, it is possible to add simple paths in such a way that a single new interactional link is created at each step. Consequently, the maximum length of a chain between m_{10} and M_2 is 10 (one intermediate organization for each additional link). All the chains between m_{10} and M_2 will, therefore, have a length between 5 and 10. In the absence of further theoretical developments about the structure of the set $\Phi(R)$, one cannot go much further in the analysis of $\Phi(R)$ without explicitly constructing its diagram.

9.2.6 Interpretation of the MINOs and MAXOs.

Besides the fact that the MINOs and MAXOs bound the set $\Phi(R)$ of all Feasible Organizations, one may want to gain a physical understanding of what they represent. This subsection presents an attempt to give MINOs and MAXOs an interpretative meaning. The interpretation presented is by no way definitive. For a given MINO or MAXO other interpretations that fit the structure of the organization and the context within which it operates may be possible.

MINOs

The MINOs represent organizations with a minimal number of interactions between decisionmakers. The MINOs will most likely not represent very robust organizational structures since no redundancy is present to double-check and confirm information or to back-up a deffective link. They will, however, certainly represent efficient configurations as far as response time is concerned. The MINOs could therefore be interpreted as organizational structures designed to respond to emergency situations, where time is critical and where redundancy is a luxury.

Surface attack

In a surface attack, information from both the radar and the sonar operators is relevant. There will be two cases: either the sonar and radar operators share their assessed situations and one of them sends an aggregate signal to the coordinator, or they both send their own situation assessments to the coordinator who will do the fusion himself.

The MINOs m_4, m_5, and m_6 correspond to the first case. To reduce the coordinator workload, DM^1 will fuse DM^2's situation assessment with his own situation assessment and send the combined information to the coordinator DM^3. DM^3 can then issue a command to one of the two weapon operators (case represented by the MINOs m_4 and m_6) or to both of them (MINO m_5). If only one of the two weapon operators receive a command from the coordinator, the other one will be informed of what is going on, either by the weapon operator who received directives from the coordinator (MINO m_6), or by DM^1 (MINO m_4).

The second case is reflected in the MINOs m_7, m_8, and m_9. They correspond respectively to m_4, m_5, and m_6. In this case, the workload of the coordinator is increased, since he has to fuse the situation assessments of the radar and the sonar operators.

Air or submarine attack

Since both cases are completely symmetrical, an air attack only will be considered. A submarine attack would be analyzed in reversing the roles of RO and SO, and the roles of AAW and ASW. In the case of an air attack, DM^1 will be the radar operator (RO) and DM^4 will be the missile system operator (AAW). The MINOs m_1, m_2, and m_3 represent configurations where the sonar operator (SO) sends information to the AAW. The nature of this information may be to instruct AAW to get ready to fire a missile, while the coordinator is evaluating the situation. In all cases, AAW will not act until he receives a command from the coordinator. As far as ASW is concerned, he has little role to play and will just be informed of what the situation is either by RO (MINO m_1), EXCO (MINO m_2) or AAW (MINO m_3). Lastly, the MINO m_{10} represents a case where SO and ASW play no role at all: the air attack is handled exclusively by RO, EXCO and AAW who do not interfere in any way with SO and ASW.

MAXOs

Unlike the MINOs, the MAXOs represent configurations with a lot of redundancy and double-checking capability. In all MAXOs, for example, RO and SO share their situation assessments together and both send their fused information to the coordinator. The latter therefore receives redundant information and can review the assessments of his subordinates. The three MAXOs only differ in the interactions between DM^3, DM^4 and DM^5. The set of all allowable interactions creates a potential for loops. Loops are indeed present in the net $\Omega(R_u)$. The three MAXOs are derived from the net $\Omega(R_u)$ by removing the minimum number of paths that will yield loop-free structures. As an example, the link between the RS stage of DM^4 and the IF stage of DM^3 (p_{27}) is removed from all MAXOs since it is in conflict with the fixed link going from the RS stage of DM^3 to the CI stage of DM^4 (p_{24}). In the MAXO M_1, DM^5 shares his result with DM^4 and DM^3. In the MAXO M_2, DM^3 sends command to both DM^4 and DM^5 and DM^5 share his result with DM^4. The roles of DM^4 and DM^5 are inverted in the MAXO M_3.

The MAXO M_1 is best fit for an air or a submarine attack, since the coordinator sends a command to DM^4 only. In MAXOs M_2 and M_3, the coordinator issues orders to both DM^4 and DM^5: this case could correspond to a surface attack, where missiles and torpedoes are used simultaneously to counter the attack. Note that MAXOs M_2 and M_3 are symmetrical to each other with respect to the operator σ_{45}.

9.2.7 Concluding remarks

This example shows how a concrete problem can be formulated within the framework developed in this thesis. The user-defined constraints are first chosen in analyzing carefully the actual situation. It is important to eliminate unnecessary redundancy that would increase the dimensionality of the design problem without bringing further insight. Once the user-defined constraints are specified, the algorithm ARCGEN generates the MINOs and the MAXOs. The set of all organizational structures that will satisfy the designer's requirements is, therefore, defined. Instead of looking at the entire set, the designer will concentrate on the MINOs and the MAXOs. Note that, althought the original problem is fairly sophisticated, there are only 10 MINOs and 3 MAXOs. The organization designer can, therefore, concentrate his analysis on those 13 organizational structures. His task is thus very much alleviated, if compared to the original one. The first step of the analysis consists in putting the MINOs and the MAXOs in their actual context, to give them a physical interpretation. If the organization designer is more specifically interested in a given pair of MINO and MAXO, he can further investigate the chains connecting those two organizations within the set $\Phi(R)$.

In summary, the methodology presented provides the organization designer with a rational way to handle a problem whose combinatorial complexity is very large.

CHAPTER X

CONCLUSIONS AND DIRECTIONS FOR FURTHER RESEARCH

10.1 CONCLUSIONS

In this thesis, a methodology is presented for generating organizational architectures that satisfy some generic structural properties, as well as more specific designer's requirements. An analytical framework is developed to formulate first and then analyze the problem.

The first step of the methodology is the characterization of the class of organizations under consideration. This step is carried out in Chapter IV. The starting point is the basic model of the interacting decisionmaker. From this model, the allowable set of interactions between the different organization members is derived, and a mathematical framework is developed to represent the interactional structure of an organization. More specifically, two binary vectors and four binary matrices will completely characterize the topological structure of an organization. In Chapter V, the above matrix description of an organization is converted into a Petri Net representation, thus making available the powerful analytical tools of Petri Net theory.

In Chapter VI, the class of organizations under consideration is further restricted by imposing a set of structural constraints on the organizational structures to be generated. Connectivity and acyclicity are among the structural properties that an organization must satisfy. Moreover, the organization designer will impose his own requirements, thus restricting even further the class of organizational structures under consideration. These requirements will be specified in using the mathematical framework presented in Chapter IV.

Given the designer's requirements and the set of pre-established structural constraints, the design problem consists in finding the set $\Phi(R)$ of all Feasible Organizations, i.e., the set of organizational structures that satisfy both the designer's

requirements and the stuctural constraints. In Chapter VII, the set $\Phi(R)$ is characterized by its boundaries, i.e., by its minimal and maximal elements. Those elements, the MINOs and the MAXOs, will correspond to the organizational structures with the minimum and the maximum number of interactional links among organization members. The complete set of Feasible Organizations is then generated by considering all the organizational structures that lie between the MINOs and the MAXOs. The notion of simple path is introduced as the incremental unit leading from an organization to another. By adding simple paths to every MINO until a MAXO is found, one will scan the complete set of Feasible Organizations.

The internal structure of the set $\Phi(R)$ of all Feasible Organizations is then investigated, using lattice theoretic techniques. The notion of subnet allows, indeed, to define an order on the set $\Phi(R)$. With this order, $\Phi(R)$ is a partially ordered set. Although $\Phi(R)$ is not, in the general case, a lattice, it is shown that $\Phi(R)$ is embeded in a lattice (the lattice $USp(R_u)$, generated by all the simple paths of the universal net). Lattice theoretic concepts and results can, therefore, be used: the notion of chain leading from a given MINO to a given MAXO is investigated.

The main contribution of this thesis is to set up a framework within which the organization design problem can be articulated. Asumptions about the model are clearly and precisely defined, thus making the scope and the limitations of the methodology easily identifiable. In relaxing some of those asumptions, the generality of the model can be extended. The organization designer is provided with a rational way to translate his requirements into specifications. This is a major feature of the methodology, since in most design procedures, the step leading from a physical to an analytic description of a concrete application, is the hardest to make. Lastly, the methodology introduces a technique to reduce considerably the number of organizational classes that the designer will eventually have to investigate. Starting from a high level of complexity, the design problem is therefore brought down to a tractable level.

10.2 DIRECTIONS FOR FURTHER RESEARCH

Research could be pursued in three different directions to improve and extend the methodology described in this thesis. Those three directions are presented in the following subsections according to an increasing degree of generalization with respect to the present model.

10.2.1 Improvement of the Present Model

In this direction, the basic assumptions made about the class of organizations under consideration are not challenged. The proposed improvements are, therefore, strictly within the scope of this thesis.

From an analytical viewpoint, the internal structure of the set $\Phi(R)$ of all Feasible Organizations needs to be investigated in much more detail than what has been done in Chapter VII. Lattice theory seems to be the right analytical tool to gain deeper insight into the structure of $\Phi(R)$. The goal is to define within the set $\Phi(R)$, categories of organizations and to select among each category a representative element, that would then be analyzed by the organization designer.

From an interpretative viewpoint, a better understanding of the analytical results obtained, is needed. An attempt to give physical significance to the MINOs and the MAXOs has been made in Chapter IX: this line of reasoning should be explored further. Classical concepts, such as hierarchical and parallel organizational structures, need to be defined within the framework of this thesis. How could, for instance, hierarchical structures be identified, by looking at paterns in the arrays defining an organization ? The concept of **slice** [22] seems to be appropriate to investigate questions related to hierarchy and parallelism. Slices, indeed, account for concurrency in the Petri Net representation of an organization, and concurrency and parallelism are related notions. An index reflecting the *degree of parallelism* of an organization could be defined from the number of maximal slices [5] of the Petri Net representing the organization. The definition of such an index would help classify Feasible Organizations, by assigning a number to each element of $\Phi(R)$.

10.2.2 Extension of the Model

In this subsection, the relaxation of the structural constraints is investigated.

Relaxing the structural constraints R_3 and R_4 would change neither the methodology nor the results presented in this thesis: it would just substantially increase the dimensionality of the design problem. More interesting is the relaxation of the acyclicity constraint. As mentioned in Chapter VI, this constraint has been introduced to avoid deadlocks and situations where messages are circulating indefinitely within the organization. The Petri Net theory, however, presents a way to avoid such a situation. According to Theorem 2.6, an event-graph will be safe under a given initial marking if and only if the token content of all its directed elementary circuits is strictly positive. In other words, loops will not create deadlock in an organization, if tokens are appropriately allocated among the different loops. Since the token content of a circuit is invariant (Theorem 2.5), those tokens can be thought of as part of the organization structure. Loops need to be introduced to model the kind of situation described below. Let us consider a two-person organization. Decisionmaker DM^1 is the commander and can instruct DM^2 to fire a weapon. Because of the stakes involved, DM^2 would need a confirmation of DM^1's order to actually fire. Decisionmaker DM^2 will, therefore, issue a querry to DM^1, requesting the latter to confirm his order. A close-loop double-check capacity is then created. Note, however, that if the protocol for the sequence of events is defined clearly, then no deadlock will occur . There is no way to model that kind of situation without information feed-back, i.e., with acyclical organizational structures. This situation is, however, very realistic, especially in a strategic environment where the stakes are high.

10.2.3 Variable Structures

A completely new class of organizations is obtained, if the basic model of the interacting decisionmaker is replaced with the four stage model with switches. As pointed out in subsection 4.3.1, this leads to variable organizational structures. The classical Petri Net theory is not the right analytical tool to handle variable structures. Since the topological structure of an organization will depend on the kind of information the organization processes, one needs to take explicitly into account the nature of information flowing from one place to another. Colored Petri Nets [30] provide the approprite analytical framework to study variable structures. In a Colored Petri Net, each token is identified by a certain color,

so that it is possible to differentiate the tokens. Colored Petri Nets account, therefore, for the diffferent types of inputs an organization receives and processes. In [30], it is shown how invariants can be defined and computed in a Colored Petri Net. Since invariants are at the root of the methodology presented in this thesis, there is some hope that the methodology may be extendable to Colored Petri Nets without fundamental changes.

Lastly, Colored Petri Nets are just a specific case of a more general class of Petri Nets: Predicate-Transition Nets [6]. In a Predicate-Transition Net, condition schemes (or predicates) are associated with the nodes and the arcs of the Petri Net. Those schemes account for the nature of information that a place can hold or that a connector can transmit, as well as for the way information is tranfered through a transition. The step leading from ordinary Petri Nets to Predicate-Transition Nets is comparable - quantitatively and qualitatively - to that of going from propositional logic to first order predicate logic [6]. In Predicate-Transitions Nets, formal integer polynomials will play the same role that integers play in ordinary Petri Nets. As an example, the elements of the incidence matrix of a Predicate-Transition Net will be formal integer polynomials. Since formal integer polynomials have properties very similar to integers, most linear algebra results obtained for ordinary Petri Nets are transferable to Predicate-Transition Nets.

In summary, Predicate-Transition Net Theory seems to offer a promising framework to analyze both the topological structure of an organization, and the information theoretic concepts accounting for the nature of information processed by the organization. Those two paths have been, up to now, explored rather independently.

REFERENCES

[1] A.H. Levis, "Information processing and decisionmaking organizations: a mathematical description", Large Scale Systems, 7., 155-163, 1984.

[2] V.Y-Y. Jin, "Delays for distributed decisionmaking organizations", MS Thesis, Report LIDS-TH-1459, Laboratory for Information and Decision Systems, MIT, 1985.

[3] G. Bejjani and A.H.Levis, "Information storage and access in decisionmaking organizations", Report LIDS-P-1466, Laboratory for Information and Decision Systems, MIT, 1985.

[4] D.Tabak and A.H.Levis, "Petri Net Representation of Decision Models", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-15, No.6, 812-818, November/December 1985.

[5] H.Hillion, "Performance Evaluation of Decisionmaking Organizations using Timed Petri Nets", MS Thesis, Report LIDS-TH-1590, Laboratory for Information and Decision Systems, MIT, 1986.

[6] H.J.Genrich, K.Lautenbach, and P.S.Thiagarajan, "Elements of General 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, 1980.

[7] H.J.Genrich and E.Stankiewicz-Wiechno, "A Dictionnary of some Basic Notions of 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, 1980.

[8] J.L.Peterson, Petri Net Theory and the Modeling of Systems, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[9] G.W.Brams, Réseaux de Petri: Théorie et Pratique, Masson, Paris, 1983.

[10] W.Reisig, Petri Nets. An Introduction, Springer-Verlag, Berlin, 1985.

[11] M.Jantzen, R.Valk, "Formal properties of Place/Transition Nets", 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, 1980.

[12] R.J.Parikh, "On Context-free Langages", Journal of the Association for Computing Machinery, Vol.13, No.4, pp.570-581, Oct.1966.

[13] G.Memmi and G.Roucairol, "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, 1980.

[14] R.M.Keller, "Vector Replacement Systems: a Formalism for Modelling Asynchronous Systems", Report IR117,CSL, Princeton University, Dec.1972.

[15] H.Müller, "The Reachability Problem for VAS", in Advances in Petri Nets,1984, G.Rozenberg, Ed., Springer-Verlag, Berlin, 1985.

[16] L.H.Landweber and E.L.Robertson, "Properties of Conflict-free and Persistent Petri Nets", Journal of the Association for Computing Machinery, Vol.25, No.3, pp.352-364, Jul.1978.

[17] J.Sifakis, "Structural Properties of Petri Nets",in Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1978.

[18] F.Commoner and A.Holt, "Marked Directed Graphs", Journal of Computer and System Science, Vol.5, No.5, pp. 511-523, 1971.

[19] S.T.Weingaertner, "A Model of Submarine Emergency Decisionmaking and Decision Aiding", MS Thesis, Laboratory for Information and Decision Systems, MIT, Cambridge, MA., 1986.

[20] G.Grätzer, Lattice Theory, First Concepts and Distributive Lattices, W.H.Freeman and Company, San Francisco, 1971.

[21] G.Birkhoff, Lattice Theory, American Mathematical Society, Providence, 1948.

[22] C.Fernandez and P.S.Thiagarajan, "A Lattice Theoretic View of K-density",in Advances in Petri Nets, G.Rozenberg, Ed., Springer-Verlag, Berlin, 1985.

[23] K.L.Boettcher and A.H.Levis, "Modeling and Analysis of Teams of Interacting Decisionmakers with Bounded Rationality", Automatica, No.6, pp.703-709, 1983.

[24] D.A.Stabile and A.H.Levis, "The Design of Information Structures: Basic Allocation Strategies for Organizations", Large Scale Systems, Vol.6, pp.123-132,1981.

[25] S.A.Hall, "Information Theoretic Models of Storage and Memory", SM Thesis, LIDS-TH-1232, Laboratory for Information and Decision Systems, MIT, Cambridge, MA., 1982.

[26] R.Balbes and P.Dwinger, Distributive Lattices, University of Missouri Press, Columbia, Missouri, 1974.

[27] G.H.-L.Chyen and A.H.Levis, "Analysis of Preprocessors and Decision Aids", Proc. 2nd IFAC/IIP/IFORS/IEA Conference on Analysis, Design and Evaluation of Man-Machine Systems, Varese, Italy, pp.81-86, 1984.

[28] J.Martinez and M.Silva, "A Simple and Fast Algorithm to Obtain all Invariants of a Generalized Petri Net", in Application and Theory of Petri Nets, C.Giraud, W.Reisig, Eds., Springer-Verlag, Berlin, 1980.

[29] H.Alaiwan and J.-M.Toudic, "Recherche des Semi-Flots, des Verrous et des Trappes dans les Réseaux de Petri", Technique et Science Informatiques, Vol.4, No.1, Dunod, France, pp.103-112, 1985.

[30] K.Jensen, "Colored Petri Nets and the Invariant Method", Theoretical Computer Science, No.14, North-Holland, pp. 317-336, 1981.