République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider Biskra
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
**Département d'Informatique**

**N° d'ordre : GLSD2/M2/1018**

# Mémoire

Présenté en vue de l'obtention du diplôme de master académique en

**Informatique**

Option : **Génie logiciel et système distribués**

---

# *Possibilistic Petri nets analysis*

---

Présenté Par :
**TOUATI HADJER**
Soutenu le : 25 / 06 /2018

Devant le jury composé de :

| | | |
|---|---|---|
| Dr. Kahloul laid | MCA | Président |
| Dr. Bennoui Hammadi | MCA | Rapporteur |
| Mme. Hmidi Zohra | MAA | Examinateur |

**Juin 2018**

# Contents

# List of Figures

# List of Tables

# Acknowledgments

I want to thank Allah Almighty for giving me strength and the will to complete this modest work. I have a lot to be thankful for in my life, and this shared few lines are for some of it, especially regarding the people who helped in the accomplishment of this manuscript.

Specialthanks go to my advisor Dr. Hammadi Bennoui, for his help and support.

I would also like to thank Dr. Laid Kahloul and Mme. Hmidi Zohra for reviewing and evaluating this work.

Without forget my mother,my sister and my two brothers for their contribution, support and patience.

Finally, i extend my sincere thanks to all my close friends, Who have always comforted me during the realization of this work. So, to all these amazing people who went with me along this year, I sincerely thank you.

# Part I

# General Introduction

Approximate reasoning is one of the remarkable human capabilities that manipulate perceptions in a wide variety of physical and mental tasks, whether in fuzzy or uncertain surroundings. To model this remarkable human capability, L.A. Zadeh [1] , [2] proposed a new concept of "computing with words," which is a methodology in which the objects of computation are words and propositions drawn from a natural language. It provides a basis for a computational theory to imitate how humans make perception-based rational decisions in a fuzzy environment. Nevertheless, besides fuzziness, humans also perform perception-based reasoning well under uncertain circumstances. To deal with uncertain information in reasoning methods, several formalisms have been proposed, such as, certainty factor [3] , probabilistic logic [4] , Dempster–Shafer theory of evidence [5] possibilistic logic [6] and possibilistic reasoning [7] , etc. Consequently, an adequate management of uncertainty for reasoning methods has become a significant issue [8] .

To increase the efficiency of rule-based reasoning with uncertain information, two issues are particularly relevant: the possibility of exploiting concurrency and the use of smart control strategies [9]. To achieve these goals, a number of researchers have reported progress toward the modeling of rule-based reasoning with Petri nets [10], [11], [12], [13], [14]. Petri nets are a graphical and mathematical modeling tool to describe and study information processing systems. Petri nets with a powerful modeling and analysis ability are capable of providing a basis for variant purposes, such as knowledge representation, reasoning mechanism [10], [14] , knowledge acquisition [27] , and knowledge verification [17], [18] .

Possibility theory and Petri net theory are combined, leading to a tool for the qualitative representation of uncertain knowledge about a system state. Based on the remark that the membership function of a fuzzy set representing imprecise or vague information can be interpreted as a possibility distribution, we call this model **possibilistic Petri nets**.

The marking of a Petri net describes crude information about the system state whereas the marking of a possibilistic Petri net represents all the propositions which are possibly true at a given step of a reasoning built over the possible current system states

Researchers have made some progress embedding uncertainty models into Petri nets for different purposes. For example, stochastic Petri nets [20],[21] , is a class of Petri nets in which the firing times are considered as random variables, and a probability distribution over all transition firing times is formed. Lin et al [22]. investigated the use of Petri nets to deal with the size problem of the possible world matrix in Nilsson's probabilistic logic [4]. Cardoso et al [23]. proposed a possibilistic Petri nets model that combined possibility theory and Petri net theory to lead to a tool for qualitative representation of uncertain knowledge about a system state. They used possibility distributions over all places and tokens to display the uncertainty about possible locations of a token before receiving certain information

Although the attempt has been made by these researchers to embed the uncertainty model into Petri nets, little emphasis has been put on how to model uncertainty reasoning in rule-based expert systems with Petri nets .Possibilistic reasoning , inspired by Nilsson's probabilistic entailment, is an uncertainty reasoning for classical propositions weighted by the lower bounds of necessity measures and the upper bounds of possibility measures. In **(PPN)** , a possibilistic token carries information to describe a proposition and the corresponding possibility and necessity measures. Four types of possibilistic transitions, inference, aggregation, duplication, and aggregation–duplication transitions, are introduced to fulfill the mechanism of possibilistic reasoning. The inference transitions perform possibilistic reasoning; duplication transitions duplicate a possibilistic token to several tokens representing the same proposition, possibility, and necessity measures; aggregation transitions combine several possibilistic tokens with the same classical proposition; and aggregation–duplication transitions combine aggregation transitions and duplication transitions. A reasoning algorithm based on possibilistic Petri nets is also outlined to improve the efficiency of possibilistic reasoning.

The objective of this master work consists to implement an uncertain diagnosis reasoning mechanism based on PPN models through analysing the T-invariants set of the model. This can be viewed as extending the initial invariant analysis technique proposed initially for classical Petri net models to handle the uncertainty.

The organization of this manuscript is as follows. A general description of Petri nets formalism is recalled in the next chapter. Chapter 2 is devoted to present the association of possibility theory and Petri nets. In particular, it introduces a possibilistic reasoning based on possibilistic entailment and a knowledge representation of possibilistic reasoning. Chapter 3 details the development of a diagnosis prototype by analysing possibilistic Petri nets. Finally, the manuscript is concluded with a summary on the conducted work.

# Part II

# State of the art

# Chapter 1

# Petri Nets

## 1.1 Introduction

Petri nets were introduced in 1962 by Dr. Carl Adam Petri (Petri 1962). They are a powerful modeling formalism in computer science, system engineering and many other disciplines. Petri nets combine a well defined mathematical theory with a graphical representation of the dynamic behavior of systems. The theoretic aspect of Petri nets allow precise modeling and analysis of system behavior, while the graphical representation of Petri nets enable visualization of the modeled system state changes. This combination is the main reason for the great success of Petri nets [25].

More generally, industrial production systems and even general social, ecological, or environmental systems can be modeled by Petri nets. In this chapter we will introduce the basic definitions of Petri net theory, study several examples in detail, and investigate some of the deeper concepts of the theory. The fig 1.1 shows the general method based on Petri net formalism for systems modeling and analysis .

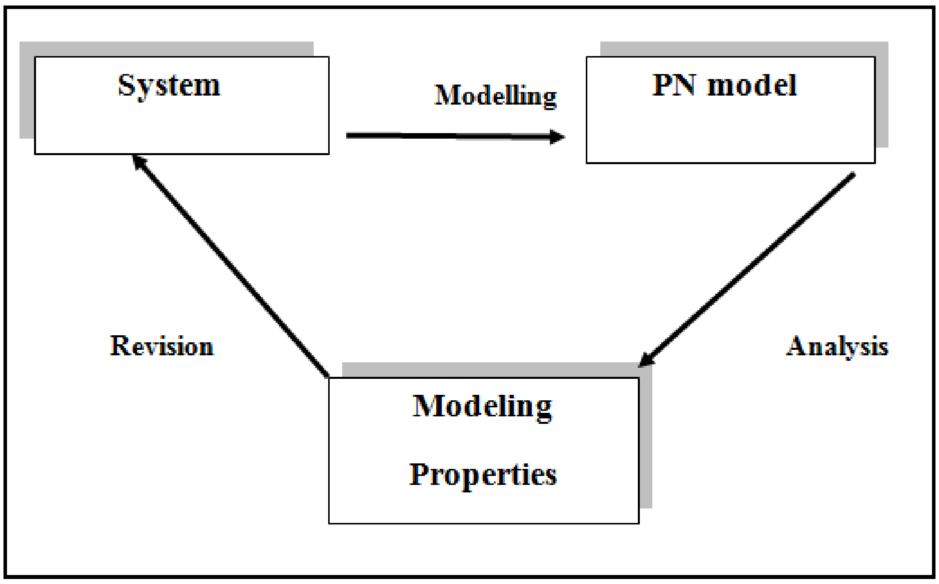


Figure 1.1: General method of modeling and analysis based on Petri nets

## 1.2 Basic concepts of ordinary Petri nets

We will start with some informal definitions.

### 1.2.1 Informal definitions

A Petri net is a particular kind of bipartite directed graphs populated by three types of objects. These objects are places, transitions, and directed arcs. Directed arcs connect places to transitions or transitions to places. In its simplest form, a Petri net can be represented by a transition together with an input place and an output place. This elementary net may be used to represent various aspects of the modeled systems. For example, a transition and its input place and output place can be used to represent a data processing event, its input data and output data, respectively, in a data processing system. In order to study the dynamic behavior of a Petri net modeled system in terms of its states and state changes, each place may potentially hold either none or a positive number of tokens. Tokens are a primitive concept for Petri nets in addition to places and transitions. The presence or absence of a token in a place can indicate whether a condition associated with this place is true or false, for instance. An example of a petri net is illustrated by the fig 1.2.

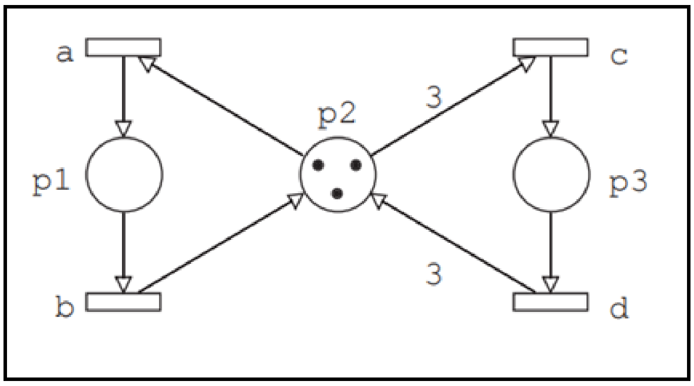


Figure 1.2: Example of a marked Petri net.

**Condition :**
A condition is a predicate or logical description of a system state. A condition is true or false. A state of the system can be described as a set of conditions.

**Event :**
Events are actions taking place in the system. The triggering of an event depends on the state of the system.

### 1.2.2 Formal definition

A Petri net is formally defined as a 5-tuple , $RdP = (P, T, F, W, M_0)$ where :

- $P$ is a finite set of places;
- $T$ is a finite set of transitions.
- $F$ is a set of arcs $F \subseteq (P \times T) \cup (T \times P)$ where :

$(P \times T)$ are the arcs going from $P$ to $T$.

$(T \times P)$ are those going from $T$ to $P$.

- $W : F \to \{1, 2, 3, \cdots\}$ is a weight function where:

  w(P;T) : Pre(p,t) is the weight of the arc going from P to T.

  w(T;P): Post(t,p) is the weight of the arc going from T to P.

- $M_0 : P \to \{0, 1, 2, 3, \cdots\}$ is the initial marking .

A marking in a Petri net is an assignment of tokens to the places of a Petri net. Tokens reside in the places of a Petri net. The number and position of tokens may change during the execution of a Petri net. The tokens are used to define the execution of a Petri net.

### 1.2.3 Matrix representation

The following figure shows a mathematical and formal matrix representation of a PN graph as follows:

$$C = \begin{array}{c} \\ \\ \end{array}\begin{array}{cccc} t_1 & t_2 & t_3 & t_4 \end{array}$$

$$C = \left[\begin{array}{rrrr} 1 & -1 & 0 & 0 \\ -1 & 1 & -3 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right] \begin{array}{c} p_1 \\ p_2 \\ p_3 \end{array}$$

Figure 1.3: Example of the incidence matrix C of a PN

In this matrix or matrix representation, the columns are the transitions $T_i$, and the lines are places $P_i$, and the intersection of these two points (places and transitions) is a function of the weight W.

### 1.2.4 Transition Firing

The execution of a Petri net is controlled by the number and distribution of tokens in the Petri net. By changing distribution of tokens in places, which may reflect the occurrence of events or execution of operations, for instance, one can study the dynamic behavior of the modeled system. A Petri net executes by firing transitions. We now introduce the enabling rule and firing rule of a transition, which govern the flow of tokens:

- A transition is said enabled (firable) if each of its input places contains at least one token. An enabled transition can fire such as : $\forall p \in P, M(p) \geq pre(p,t)$.
- When a transition is validated in the marking M0, we note M0[t>.
- Firing a transition removes a token from each input place and adds one token to each ouput place.
- When a transition is validated it does not imply that it will be fired immediately.
- A sequence of transitions that can be fired consecutively starting from the initial marking is said enabled or firable.

- The sequence of firable transitions is not unique.

When several transitions are firable, one of them (in a non-deterministic way) is fired. Its firing consists of removing a token from each of its input places and adding another one to each of its output plaes.

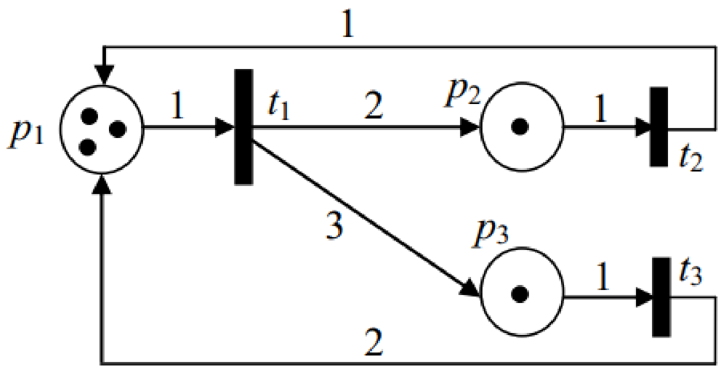


Figure 1.4: Firing of the transition $t_1$ (initial state)

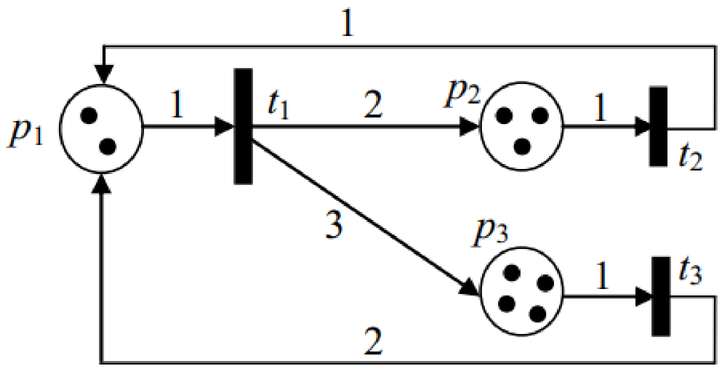


Figure 1.5: Firing of the transition $t_1$ (After firing $t_1$ )

- **Source transition:** A transition without input places is always enabled one: it is a source transition .

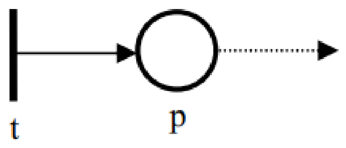


Figure 1.6: Source transition

The firing of a source transition involves adding a token to each of its output places. Note that a source transition is unconditionally enabled.

- **Sink transition:** A transition without output places is a sink transition.

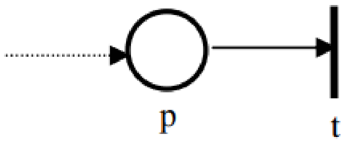


Figure 1.7: Sink transition

Firing a sink transition involves removing one token from each of its input places.

**b) a firing sequence** A a firing sequence from $M_0$ to $M_n$ is a sequence of transitions $t_0 \ldots t_(n-1)$ which can be fired successively from a given marking (or initial). Only one transition can be fired at a time. As there are markings $M_1 \ldots M_(n-1)$ verifying $M_0[t_0 > M_1 \ldots M_(n-1)[t_(n-1) > M_n$.

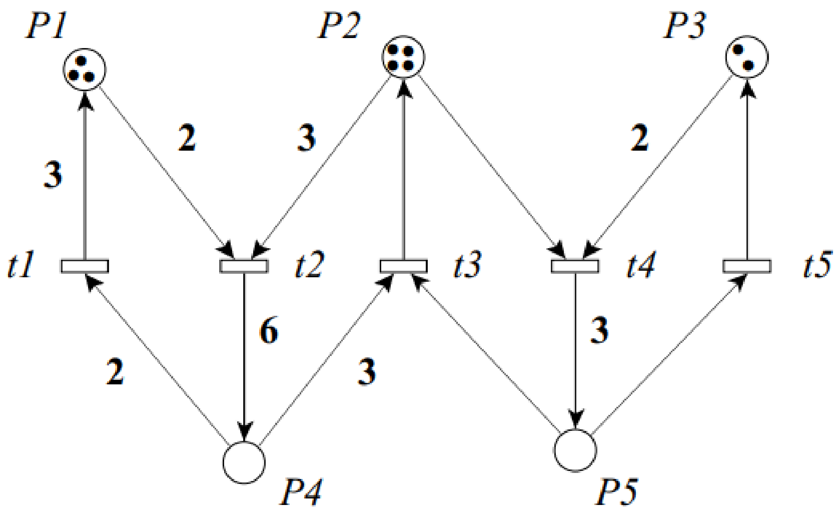


Figure 1.8: Example of firing sequence

$s = t_2 t_4 t_3$ is a firing sequence from $M_0$
$M_0[t_2 > M_1[t_4 > M_3[t_3 > M_4 \; M_3 = M_4$

**c) Marking :** Each place contains a positive or zero number of marks or tokens. The marking M defines the state of the system described by the net at a given moment. It is a column vector of dimension the number of places in the net.

**c) Accessible marking :** The set of accessible markings is the set of markings $M_i$ that can be reached by firing a sequence S from the initial marking $M_0$. We note it $*M_0$. $*M_0 = \{M_i \text{ tel que } M_0[s \rightarrow M_i\}$.

### 1.2.5 Properties of PN model

As a mathematical tool, Petri nets possess a number of properties. These properties, when interpreted in the context of the modeled system, allow the system designer to identify the presence or absence of the application domain specific functional properties of the system under design. Two types of properties can be distinguished, behavioral and structural ones. The behavioral properties are those which depend on the initial state or marking of a Petri net. The structural properties, on the other hand, do not depend on the initial marking of a Petri net. They depend on the topology, or net structure, of a Petri net. Here we provide an overview of some of the most important, from the practical point of view, behavioral and structural properties.

**a) The behavioral properties (Generic properties)**

- **Liveness and blocking :**
  A Petri net with initial marking $M_0$ is live if, no matter what marking has been reached from $M_0$, it is possible to ultimately fire any transition by progressing through some further firing sequence.

  A Petri net modeling a deadlock-free system must be live. This implies that for any reachable marking M, it is ultimately possible to fire any transition in the net by progressing through some firing sequence. This requirement, however, might be too strict to represent some real systems or scenarios that exhibit deadlock-free behavior. For instance, the initialization of a system can be modeled by a transition (or a set of transitions) which fire a finite number of times. After initialization, the system may exhibit a deadlock-free behavior, although the Petri net representing this system is no longer live as specified above. For this reason, different levels of liveness for transition t and marking M0 were defined. Refer to (Murata 1989) for details.

- **Boundedness :**
  A Petri net (N, $M_0$ ) is said to be k-bounded or simply bounded if the number of tokens in each place does not exceed a finite number k for any marking reachable from $M_0$,i.e., M(p) $\leq$ k for every place p and every marking M $\in$ R($M_0$). A Petri net (N,$M_0$) is said to be safe if it is 1-bounded.

- **Conflict :**
  A structural conflict is a set of two or more transitions $t_1$ and $t_2$ with a common place of entry. This is noted as follows : $k =< p, \{t_1, t_2, \ldots, t_n\} >$

  An effective conflict is the existence of a structural conflict k, and a marking M, such that the number of marks in p is smaller than the number of output transitions of p which are validated by M.

**b) Structural properties ( Specific properties)** Specific properties are grouped into four types: accessibility, safety, liveliness and equity properties.

- **accessibility (reachability) :**

  Determines whether a situation is accessible or not from the accessibility graph.

- **safety :**

  Some situations should never be reached.

- **liveness :**
  A situation will sooner or later take place.

- **equity (fairness) :**
  A situation will occur an infinity of times.

**c) Marking graph** The temporal evolution of a PN can be described by a marking graph representing all the accessible markings and arcs corresponding to the firing of the transitions passing from one marking to the other for initial marking $M_0$.

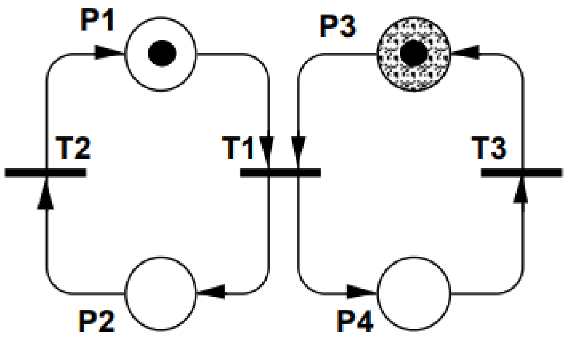


Figure 1.9: Petri net (a)

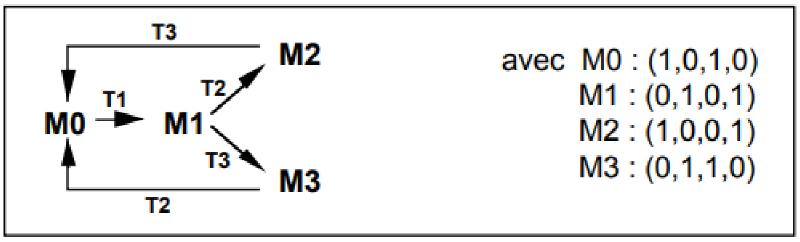


Figure 1.10: Graph of the markings of Petri net (a)

## 1.2.6 Diagnosis by Petri nets

The purpose of monitoring an industrial system is to address all behaviors that deviate from the expected behavior. The purpose of the monitoring function is to increase productivity by better controlling the availability of the means of production. The main elements of a surveillance system are the detection, localization, diagnosis and treatment of errors

The basic mechanism used for detection is to compare the evolutions of the observed system with those of a model that evolves synchronously, that is to say in real time, with the system. Similarly, the current trend is to base the localization and diagnostic phases on a deep model that describes the structure and / or behavior of the system. Petri nets are one of the most used models for discrete event systems. They are, indeed, well adapted to describe the dynamics of such systems (description of the passages from one state to another). On the other hand, they do not correspond to a structural model (the architecture of the system in a given state). We recall that a discrete event system is a system for which time as well as the components of the state vector are discrete variables.

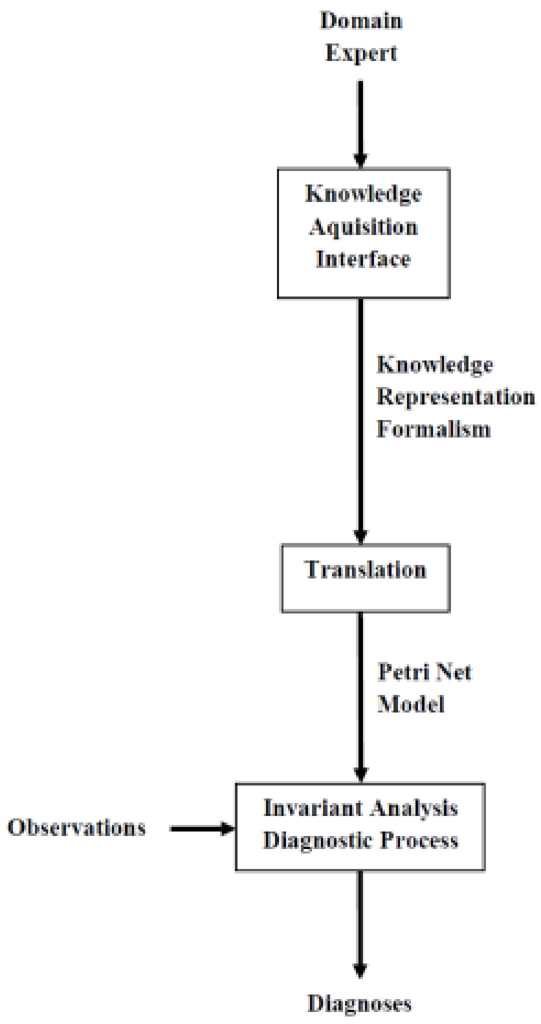

Domain Expert
Knowledge Aquisition Interface
Knowledge Representation Formalism
Translation
Petri Net Model
Observations
Invariant Analysis Diagnostic Process
Diagnoses

Figure 1.11: A diagnostic process architecture

Fig. 1.11 describes an architecture for the diagnostic process, adapted from [33] ,It starts from the expert knowledge about the behavior of a system to a Petri net model representing the system, passing by a formalism for knowledge representation like first order logic. The architecture then depicts the desired solution of the diagnostic process (diagnoses) as the fruit of exploiting an invariant analysis technique, with some observations as parameters..

# Chapter 2

# Diagnosis by Possibility PN Analysis

## 2.1 Introduction

In the field of data analysis and pattern recognition, we manipulate information, most often digital, which is supposed to give as realistic a picture as possible of reality. However, most often, this information is imperfect: imprecise, uncertain, vague, incomplete. We give here a quick presentation and a summary scheme in Figure 2.1. The imperfection in the data can be broken down into three (non-exclusive) categories: uncertainty, inconsistency and imprecision .

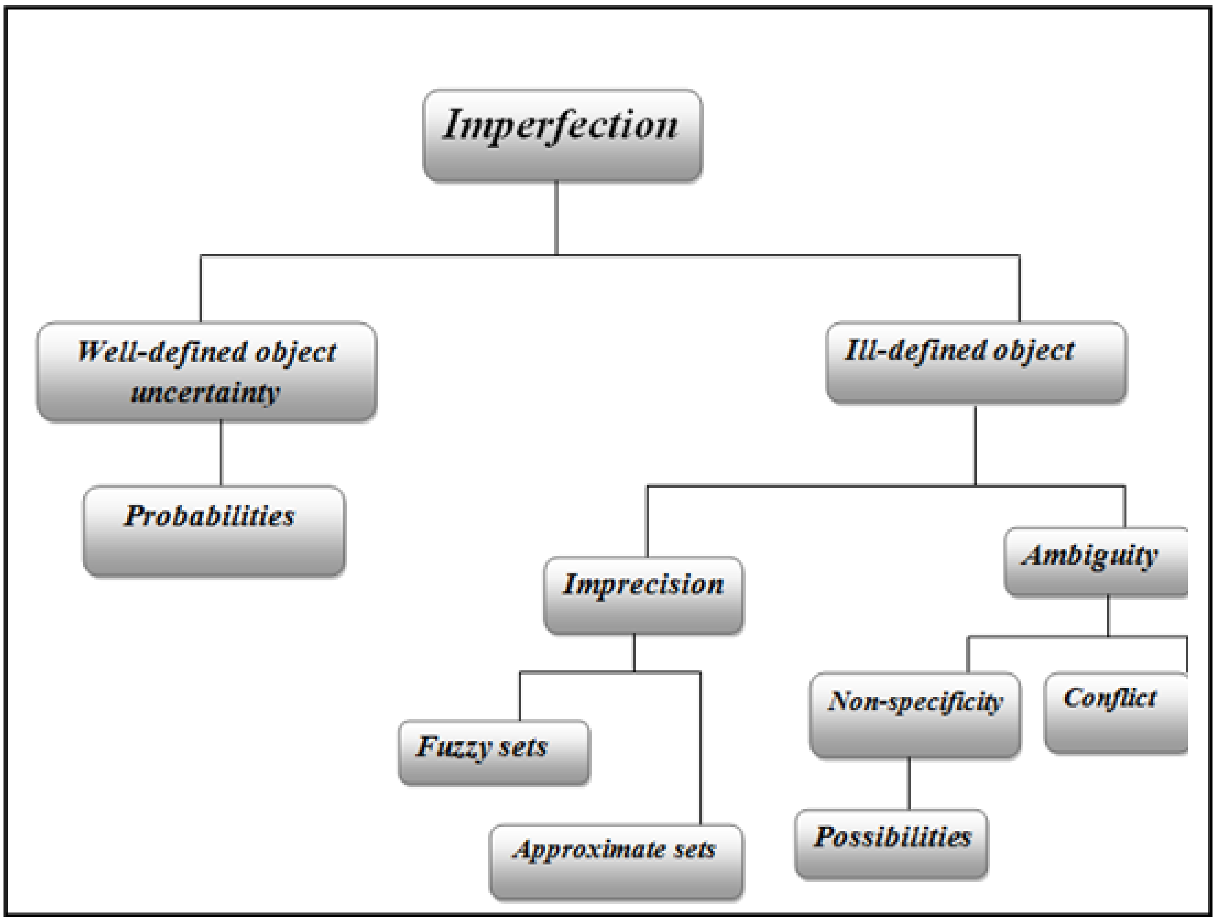


Figure 2.1: Different forms of imperfection

## 2.2 Probabilities :

The notion of probability is linked to that of random experience. An experiment is random if one can not predict with certainty its result. The result of a random experiment is an $\omega$ element of the set $\Omega$ of all the possible outcomes, called universe of possibilities or repository. We denote $P(\Omega)$ the set of parts of $\Omega$. An event, linked to a random experiment, is a logical proposition relative to the result of the experiment, it is chosen in a set of events $A$, subset of $P(\Omega)$. If $A$ and $B$ designates two elements of $A$, then :

- **A $\cup$ B designates the realization of A or B**
- **A $\cap$ B designates the realization of A and B**
- $\bar{A}$ **= $\Omega \setminus$A denotes the opposite of A.**

On the other hand ,

- **$\Omega$ is the sure event.**
- $\theta$ **is the impossible event .**

When the repository **$\Omega$** est finited, **A** groups all the parts of $\Omega$, noted habitually $2^{\Omega}$. When the repository is **R** or an interval of **R**, we use to define **A** to the notion of tribe. **A tribe** is defined as follows :

### 2.2.1 Definition :

A is **a tribe** on $\Omega$ if and only if A is a set of parts of $\Omega$ containing the empty set, stable by passing to the complement and by union and intersection of a finite or countable sequence of elements [31]:

- **A$\subseteq$ P($\Omega$) .**
- $\theta \in$ **A .**
- **If we have an A1, ..., An suite of elements of A, then their union and their intersection $\cup_i A_i$ et $\cap_i A_i$ are also in A.**
- **If A $\in$A then $\bar{\text{A}}$ is also in A**
- $\theta$ **is the impossible event .**

## 2.3 Possibilities :

### History

Possibility theory is an uncertainty theory devoted to the handling of incomplete information.To a large extent, it is comparable to probability theory because it is based on set-functions. It differs from the latter by the use of a pair of dual set functions (possibility and necessity measures) instead of only one. Besides, it is not additive and makes sense on ordinal structures. The name “Theory of Possibility” was coined by Zadeh ,In Zadeh’s view, possibility distributions were meant to provide a graded semantics to natural language statements. However, possibility and necessity measures can also be the basis of a full-fledged representation of partial belief that parallels probability. It can be seen either as a coarse, non-numerical version of probability theory, or a framework for reasoning with extreme probabilities, or yet a simple approach to reasoning with imprecise probabilities .

### 2.3.1 Formal background (theoretical setting)

Given a set of possible worlds, a proposition $r$ is true in some of them and false in the rest. To model the uncertainty associated with the actual world, we define a possibility distribution over all possible worlds. Such description is used to determine the degree of possibility of the actual world being in a possible world. Formally, Dubois et al [30]. defined the possibility and necessity measures as:

$$\prod(r) = \boldsymbol{Sup}\ (\pi(\omega))|\ \omega \models \mathbf{r}$$

$$N(r) = \boldsymbol{Inf}(\ 1 - \pi(\ \omega)\ |\ \omega \models \neg r\ )$$

where :

- **$\prod$ is the possibility measure; .**
- ***r* is a proposition;**
- **$\omega$ is a possible world;**
- **N is the necessity measure;**
- **$\omega$ |= r means that r is true in $\omega$ ($\omega \in \Omega$ );.**
- **$\Omega$ is the set of possible worlds.**

The distinction between imprecise and uncertain information is best explained by the canonical form representation (i.e., a quadruple of attribute, object, value, confidence) proposed by Dubois and Prade . Imprecision implies the absence of a sharp boundary of the value component of the quadruple; whereas, uncertainty is related to the confidence component of the quadruple, which is an indication of our reliance about the information. Dubois and Prade have proposed the possibility and necessity measures as an uncertainty model for classical propositions.

The uncertainty of an event **A**,contrary to probabilities, is therefore characterized by two values: its possibility $\prod(\mathbf{A})$ and its necessity $\mathbf{N(A)}$.The interpretation of a degree of possibility is very different from that of a probability. As an illustration, let's take the famous **Zadeh** example of **Hans breakfast** [32]. The possibility and probability values for the number of eggs that **Hans** eat tomorrow are assumed to be known. They are given in the table **2.1**. We observe that the possibility that **Hans** eats three eggs is 1 while the probability is only 0.1. We see, then, that a high degree of possibility does not imply a high degree of probability, and that a low degree of probability is not synonymous with a low degree of possibility. Only can we say that a degree of zero possibility implies a zero probability.

| $\omega$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $\pi(\ \omega)$ | 1 | 1 | 1 | 1 | 0.8 | 0.6 | 0.4 | 0.2 |
| p($\omega$) | 0.1 | 0.8 | 0.1 | 0 | 0 | 0 | 0 | 0 |

Table 2.1: Possibilities and probabilities associated with egg numbers

The necessity measures satisfy the following relation:

$$\boldsymbol{Min}(\mathbf{N(A)},\mathbf{N}(\bar{A}) = \mathbf{0}$$

In addition, we have:

$$\mathbf{N(A)} > \mathbf{0} \Rightarrow \prod(A) = \mathbf{1} \qquad\qquad \prod(A) \ \mathbf{<1} \Rightarrow \mathbf{N(A)} = \mathbf{0}$$

## 2.4 Possibilistic Petri Nets :

In this section, PPNs are defined for the modeling of possibilistic systems and are used as a representation of knowledge for possibilistic information. The idea was first introduced by Cardoso et al. The simple view of a system focuses on two primitive concepts: **events** and **conditions**. An event is represented as an action in the system. A condition is a predicate or logical description of the state of the systems [34]. A typical interpretation of Petri nets is to consider a place as a condition, a transition as the causal connectivity of conditions (an event), and a token in a place as a fact used to claim the truth of the condition associated with the condition. in law. Confidence about the connectivity of conditions and facts, however, may be uncertain. Taking uncertain situations into account, we formally define a possibilistic Petri net as following:

### 2.4.1 Definition :

A possibilistic Petri net **(PPN)** is 5-tuple, PPN = (P, PT, A, W , $M_0$) where :

- $\mathbf{P} = P_1(r_1)$**,** $P_2(r_2)$**,......**$P_m(r_m)$ is a finite set of places ($p_i$ represents a classical proposition $r_i$).

- **PT** = $t_1(N_1, \prod_1)$ , $t_2(N_2, \prod_2)$ , ........, $t_n(N_n, \prod_n)$
  is a finite set of possibilistic transitions, with $t_i$ representing the connectivity between places, $N_j$ denoting the lower bounds of necessity measures and $\prod_i$ denoting the upper bounds of possibility measures.

- **F** ⊆ (P × T) ∪ (T × P) is a set of arcs,

- $M_0$ : P → M($P_1$ ),M($P_2$ ),........,M($p_m$) is the initial marking, with M($P_i$ ) standing for the number of tokens in $P_i$ .

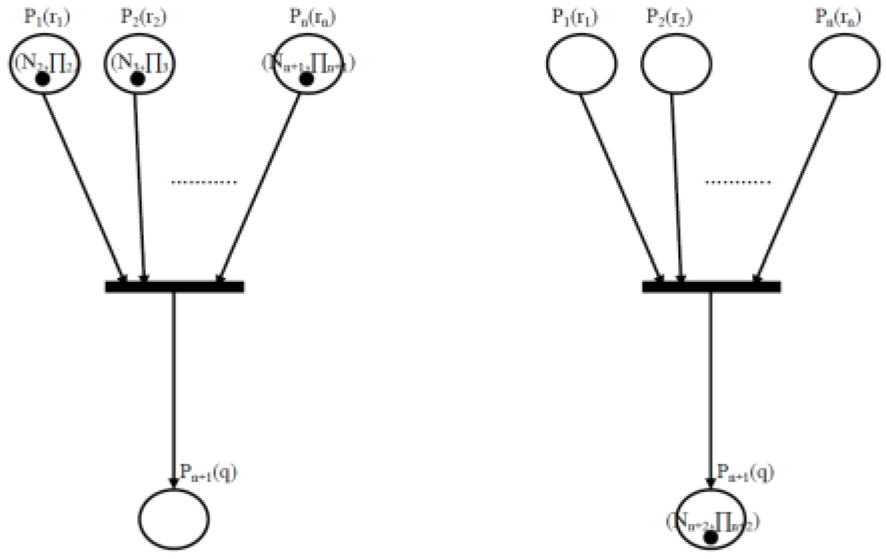


Figure 2.2: A simple example of a possibilist Petri net

Many researchers have devoted themselves to modeling rulebased reasoning via Petri nets . There are several reasons behind the computational paradigm for rulebased reasoning on Petri net theory:

- Petri nets achieve the structuring of knowledge within rule bases which express the relationships among rules, and also help experts to construct and modify rule bases.

- Petri net's graphic nature provide visualization of dynamic behavior of rule-based reasoning.
- Petri nets make it easier to design an efficient reasoning algorithm.

- Petri net's analytic capability provides a basis for developing knowledge verification technique.

- Petri nets can model the underlying relationship of concurrency among rules activation, an important aspect where real-time performance is crucial.

To simulate the dynamic behavior of a possibilistic system, a marking in a PPN is changed according to the firing rule: a firing of an enabled possibilistic transition $t_j$ , removes the possibilistic tokens from each input place $P_i$ of $t_j$ , adds a new token to each output place $p_k$ of $t_j$ and the necessity and possibility measures attached to the new token will be computed based on possibilistic systems. ( the possibilistic system is considered as possibilistic reasoning.) A simple example of is illustrated in Fig. 2.3

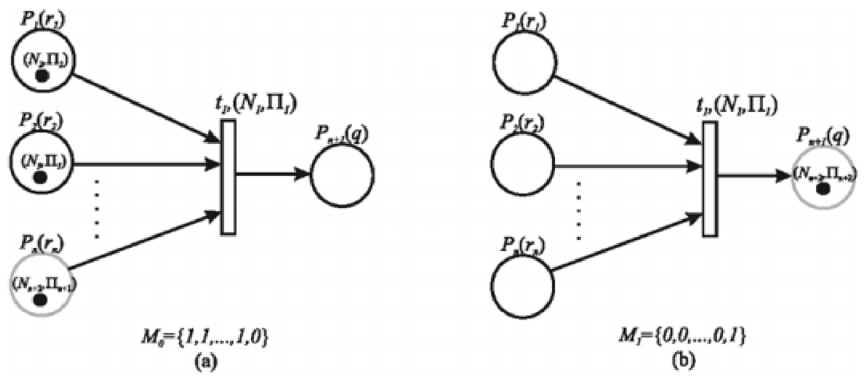


Figure 2.3: Illustration of a possibilistic Petri net : (a) Before firing $t_1$ . (b) After firing $t_1$

To represent the uncertain information, we proposed a representation of the possible propositions as follows

$$r, \ (N_r, \Pi_r)$$

where $\boldsymbol{r}$ denotes a classic proposition, $N_r$ refers to the lower limits of necessity and $\prod_r$ the upper limits of the possible measures

Cardoso et al. have proposed a possibilistic Petri net model that combines the possibility of theory and the theory of Petri nets to lead to a qualitative representation tool of uncertain knowledge about a state of the system. They used opportunity distributions across all places and tokens to display uncertainty about the possible locations of a token before receiving certain information.

## ***Knowledge Representation***

The three key components in uncertain rule-based reasoning : propositions,uncertain rules,and uncertain facts,can be formulated as places,possibilistic transitions,and possibilistic tokens,respectively.The mapping between possibilistic reasoning and ***PPN*** is described as follows.

- **Places :** Places correspond to classical propositions.The classical propositions that attached to places represent conditions .

- **Possibilistic tokens :** A possibilistic token represents an uncertain fact .A pair of necessity and possibility measures are attached to possibilistic tokens to represent our confidence level about the observed facts .
- **Possibilistic transitions :** Possibilistic transitions are classified into four types: inference,aggregation,duplication ,and aggregation-duplication transitions. The inference transitions represent the uncertain rules, the aggregation transitions are designed to aggregate the conclusion parts of rules which have the same classical propositions,the duplication transitions are used to duplicate possibilistic tokens to avoid the conflict problem and the aggregation-duplication transitions link the same classical propositions. These are formally defined below :

### *Type 1 : inference transition* $t^i$

An inference transition is used to model an uncertain rule. An uncertain rule with multiple antecedents is represented as :

$$(r_1 \wedge r_2 \wedge \cdots \wedge r_n) \rightarrow q, (N_1, \Pi_1)$$

$$\begin{array}{cl}
(r_1 \wedge r_2 \wedge \cdots \wedge r_n) \rightarrow q & , \left(N_{(r_1 \wedge r_2 \wedge \cdots \wedge r_n) \rightarrow q}, \Pi_{(r_1 \wedge r_2 \wedge \cdots \wedge r_n) \rightarrow q}\right) \\
r_1 & , (N_{r_1}, \Pi_{r_1}) \\
r_2 & , (N_{r_2}, \Pi_{r_2}) \\
\vdots & \vdots \\
r_n & , (N_{r_n}, \Pi_{r_n}) \\
\hline
q & , (N_q, \Pi_q)
\end{array}$$

where $r_i$ and $q$ are classical propositions. In Fig. 2.4,after firing the inference transition $t_i^1$ the tokens will be removed from the input places of $t_i^1$ , a new token will be deposited into the output place of $t_i^1$ and a pair of necessity and possibility measures attached to the new token are derived by the possibilistic reasoning

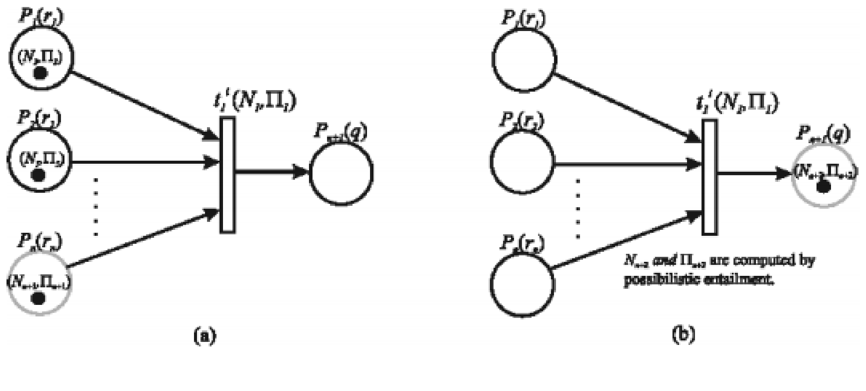


Figure 2.4: Modeling possibilistic reasoning through; (a) Before firing inference transition $t_i^1$ , (b) After firing $t_i^1$.

### *Type 2 : Aggregation Transition ( $t^a$ ):*

An aggregation transition is used to aggregate the conclusions of several uncertain rules which have a same classical proposition and to link the antecedent of an uncertain rule which also have the same classical proposition. For example, there are $m$ uncertain rules with the same classical proposition **q** in the conclusions, denoted as :

$$(r_1 \rightarrow q, (N_1, \Pi_1)), (r_2 \rightarrow q, (N_2, \Pi_2)), \ldots$$
$$(r_m \rightarrow q, (N_m, \Pi_m)).$$

In figure 2.5 ,after firing the aggregation transition ($t^a_{m+1}$), the tokens in input places of $t^a_{m+1}$ will be removed, a new token will be deposited into the output place of $t^a_{m+1}$ and a pair of necessity, and possibility measures attached to the new token are derived by disjunction. It should be noticed that $t^a_{m+1}$ is dead if one of its input places never received a token. To avoid deadlock in aggregation transitions, we assume that for each source place $P_i$ , a token will be inserted into $P_i$ and a pair of necessity and possibility (0,1) is attached to represent ignorance if no fact refers to the proposition in place $P_i$ .

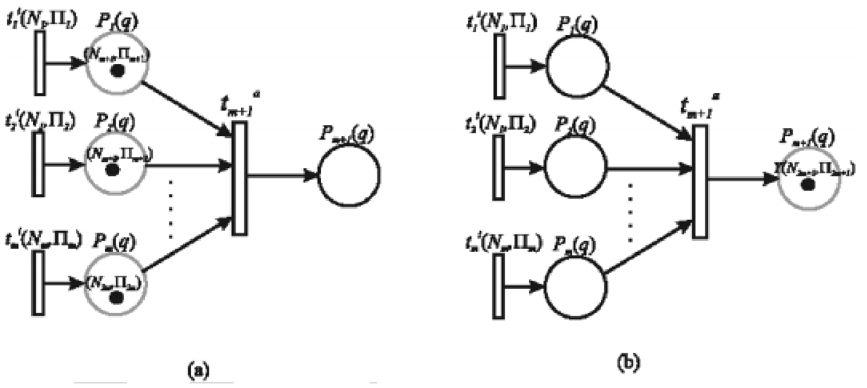


Figure 2.5: Modeling the agregation of conclusion by an agregation transition : (a) Before firing $t^a_{m+1}$ . (b) After firing $t^a_{m+1}$ .

### *Type 3 : Duplication Transition ( $t^d$ ):*

The purpose of the duplication transitions is to avoid the conflict by duplicating the token.For example, there are uncertain rules ( $\imath$ ) with a same classical proposition **r** in the antecedents, denotes as

$$(r \rightarrow q_1, (N_1, \Pi_1)), (r \rightarrow q_2, (N_2, \Pi_2)), \ldots$$

$$(\mathrm{r} \rightarrow q_\imath \ , \ ( \ N_\imath, \Pi_\imath))$$

They are linked by a duplication transition shown in Fig. 2.6. After firing the duplication transition $t_1^d$. The token in the input place of $t_1^d$ will be removed , new token will be added into the output places of $t_1^d$ and a pair of necessity and possibility measures attached to the new tokens are not changed .

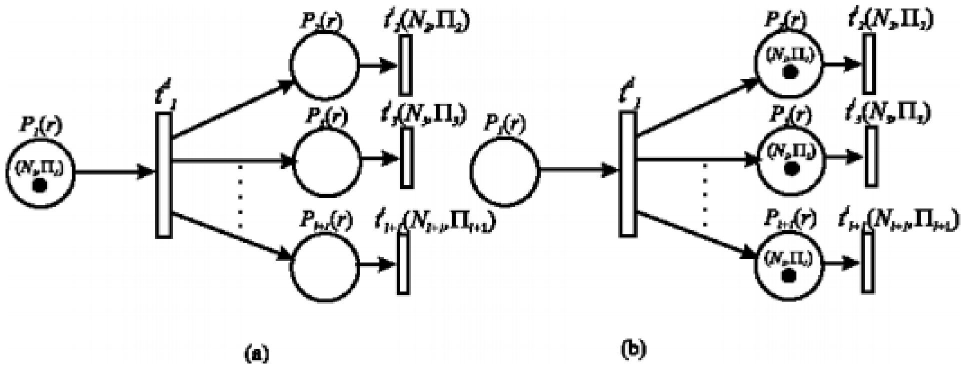


Figure 2.6: Modeling the duplication of possibilistic token through P P N : (a)Before firing the duplication transition $t_1^d$ , (b) After firing $t_1^d$

### ***Type 4 : Aggregation-duplication Transition*** $(t^{ad})$

An Aggregation-duplication transition is a combination of an aggregation transition and a duplication transition ( Fig. 2.6). It is used to link all the same classical propositions. For example, there are uncertain rules **m** with a same classical proposition **q** in the conclusions and uncertain rules ( $\imath$ ) with the same classical proposition **q** in the antecedents, denoted as

$$(r_1 \rightarrow q, (N_1, \Pi_1)), (r_2 \rightarrow q, (N_2, \Pi_2)), \ldots$$
$$(r_m \rightarrow q, (N_m, \Pi_m)), (q \rightarrow s_1, (N_{m+1}, \Pi_{m+1}))$$
$$(q \rightarrow s_2, (N_{m+2}, \Pi_{m+2})), \ldots, (q \rightarrow s_l, (N_{m+l}, \Pi_{m+l})).$$

They are linked by an aggregation-duplication transition shown in Fig. 2.7. After firing the aggregation-duplication transition , the tokens in the input places of $t^{ad}_1$ will be removed, new tokens will be deposited into the output places of $t^{ad}_1$ and the pair of necessity and possibility measures attached to the new tokens are derived by disjunction .

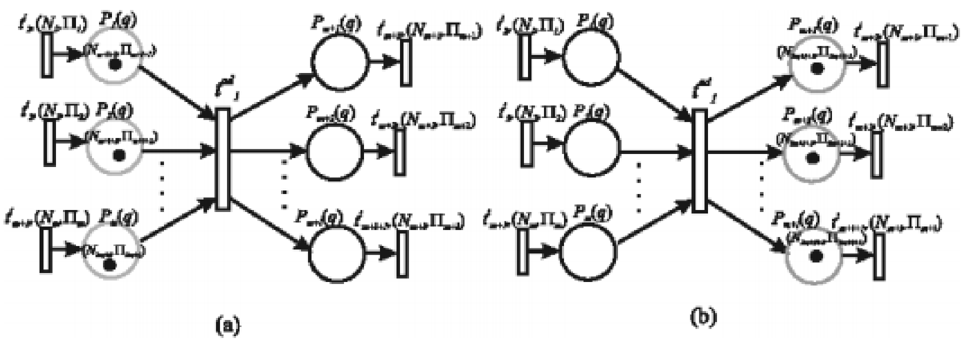


Figure 2.7: Modeling the aggregation-duplication of possibilistic tokens through P P N . (a) Before firing the aggregation-duplication transition $t^{ad}_1$ , (b) After firing $t^{ad}_1$

## 2.4.2 *Reasoning Algorithm :*

To consider the efficiency of ***P P N*** , an algorithm of transition firing is proposed based the notion of extended markings . An extended marking , denoted as $M^E$, is a $k$-vector , where $k$ is the number of possibilistic tokens.

$M^E(\mathrm{P}_i) =$

$$\Big[P_i(r_a), (N_i, \Pi_i), P_i(r_a)\bullet, \bullet(P_i(r_a)\bullet)\backslash\{P_i(r_a)\}, (P_i(r_a)\bullet)\bullet\Big]$$

where P represents a place; $r_i$ is a classical proposition;( $N_i$,$\prod_i$ ) is a pair of necessity and possibility measure s attached to the token in place $P_i$,$P_i(r_a)\bullet$ is the output transition of $P_i(r_a)$ .From an extended places $M^E(P_i)$ we know that:(a) The pair of necessity and possibility measures attached to the token in $P_i(r_a)$ (i.e.,( $N_i$,$\prod_i$ )),(b) The other tokens needed to fire $P_i(r_a)\bullet$ (i.e., $\bullet P_i(r_a)\bullet \setminus P_i(r_a)$ ).(c) What kind of computation to carry out after firing (i.e., the type of $P_i(r_a)\bullet$) and (d) where to go for the new tokens after firing (i.e.,$(P_i(r_a)\bullet)\bullet$). The algorithm is described as follows:

Algorithm 1: **(Possibilistic Petri Nets)**

**1**. Get the initial extended marking $M^E_0$, which consists of all source places.

**2.** For each $i$, set a current extended marking $M^E_c = M^E_i$ and the next extended marking $M^E_{i+1} =$ .

**3.** Select an element of the current extended marking,

$$M^E_c(P_j) = \Big[P_j(r_b), (N_j, \Pi_j),\ \ P_j(r_b)\bullet, \bullet(P_j(r_b)\bullet)\backslash\{P_j(r_b)\}, (P_j(r_b)\bullet)\bullet\Big]$$

**4.** **(a)** If $P_j(r_b)\bullet$ is an inference transition and the extended place of each $P_\imath(r_d) \in \bullet P_j(r_b)\bullet \setminus P_j(r_b)$ exists in $M_c^E$, then infer the extended place $M_i^E + 1(P_k)$ of $P_k(r_c) = P_j(r_b)\bullet)\bullet$ by possibilistic reasoning .

**(b)** Else if $P_j(r_b)$ is a duplication transition, then infer the extended place $M_i^E + 1(P_k)$ of each $P_k(r_c) \in P_j(r_b)\bullet)\bullet$ by duplication .

**(c)** Else if $P_j(r_b)\bullet$ is an aggregation transition and the extended place of each $P_\imath(r_d) \in \bullet P_j(r_b)\bullet \setminus P_j(r_b)$ exists in $M_i^E + 1(P_k)$ of $P_k(r_c) = P_j(r_b)\bullet)\bullet$ by disjunction . **(d)** Else if $P_j(r_b)\bullet$ is an aggregation-duplication transition and the extended place of each $P_\imath(r_d) \in \bullet P_j(r_b)\bullet \setminus P_j(r_b)$ exists in $M_c^E$ then infer the extended place $M_i^E + 1(P_k)$ of each $P_k(r_c) = P_j(r_b)\bullet)\bullet$ by disjunction .

**5.(a)** If the output transition of $P_j$ is fired, then insert the inferred extended place into the next extended place $M_i^E + 1(P_k)$ into the next extended marking $M_i^E + 1$ .

**(b)** Else if the output transition of $P_j$ is a hierarchy, then insert the extended place $M_c^E(P_j)$ into the hierarchy and wait for the final extended marking of the hierarchy to be inserted into the current extended marking .

**(c)** Else insert this element $M_c^E(P_j)$ and each $M_c^E(\mathrm{P}\imath)(\mathrm{P}\imath\ (r_d) \in \bullet P_j(r_b)\bullet)$ into the next extended marking $M_i^E{}_{+1}$ .

**6.** Delete the element $M_c^E(P_j)$ and each $M_c^E(\mathrm{P}\imath)(\mathrm{P}\imath\ (r_d) \in \bullet P_j(r_b)\bullet)$ from the current extended marking .

**7.** Repeat step 3 to step 6 until no element is in the current extended marking.

**8.** Repeat step *2* to step *7* until all output transitions in the current extended marking are not fired .

**9.** Send the final extended marking to the upper-leveled hierarchy.

Basically, the algorithm is used to manage the evolution of extended marking. First, it defines the current and next extended markings (step 2). Second, it determines which type of the transitions in the current extended marking are and decides whether the transitions can be fired or not (step 4). Finally, it deletes the extended places from the current extended marking if their output transitions are fired and inserts the output extended places of these fired transitions into next extended marking or next hirarchies. The procedure is repeated until no distinction can be made between the current and next extended markings .

## 2.5 Possibilistic Entailment

Inspired by Nilson's work on probabilistic logic [35] , more specifically probabilistic entailment, the notion of possibilistic entailment has been outlined in [36] as:

$$\frac{\begin{array}{l} r \rightarrow q, (N_{r\rightarrow q}, \Pi_{r\rightarrow q}) \\ r \qquad , (N_r, \Pi_r) \end{array}}{q, (N_q, \Pi_q)}$$

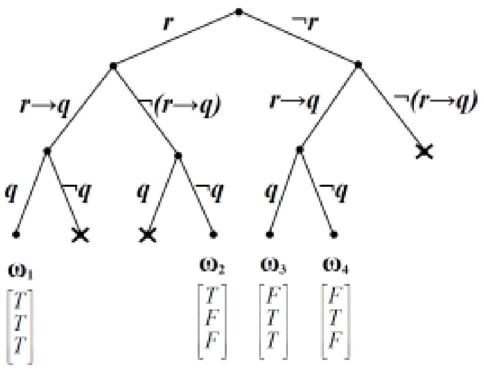


Figure 2.8: Possible worlds for possible information

The goal of this entailment is to infer a new proposition with its associated necessity and possibility. The possible worlds, in which the actual world might be (Fig. 2-8), are determined using a semantic tree. Hence, a consistent path in the semantic tree is considered to represent a possible world. Moreover, Table 2.2 shows the truth values of these possible worlds

| $w$ | $w_1$ | $w_2$ | $w_3$ | $w_4$ |
|---|---|---|---|---|
| **r** | true | true | false | false |
| $r \rightarrow q$ | true | false | true | true |
| q | true | false | true | false |

Table 2.2: Truth values of possible worlds.

So, given the set of propositions **S** $= (r\ ,\ r \rightarrow q\ ,\ q)$ , its possible worlds could be deduced using a semantic tree ( Fig. 2-8).

## 2.6 Centralized Diagnosis with Possibilistic Petri Nets

### 2.6.1 Causal Models

Causality [38] is one of the key concepts associated with a diagnostic process. It describes *the cause* $\rightarrow$ *effect* relation among a system components. The basic mathematical model used to describe such a relation is a directed graph (digraph). Composed of a set of nodes with arcs relating them, a digraph captures this relation in its simplest way. Console Torasso [39] suggest distinguishing at least among three types of nodes:

- *Initial nodes*: the first nodes on a causality chain. They correspond to the initial perturbations leading the system to fail, in case of a faulty model. They have no cause, so no other node leads to them.

- *Internal nodes:* they correspond to the consequences of initial nodes and they have at least one parent and one child node. Since internal nodes can be explained by means of initial nodes, they do not make part of diagnoses.

- *Manifestations:* they correspond to the observable part of the system in which the symptoms of a malfunction are expected to be observed. Since they are the last nodes on a causality chain, they lead nowhere.

Following a causal scheme to diagnosis corresponds to explaining a manifestation by means of initial states. Moreover, in the case of Petri nets, the nodes are represented by places and cause-effect relationships are represented by transitions between the corresponding places.

Fig. 2.9 adapted from [40] shows a causal network demonstrating the causal relation between four nodes. For instance, “Cold” causes both “Fever” and “RunnyNose”, while “Allergy” causes “RunnyNose”. On the other hand, it shows how diagnosis works in the other direction of causality. For instance, given that we observe “RunnyNose”, a diagnosis shall give us either “Cold” or “Allergy”. As one can easily see, both causes are legitimately plausible, and without additional measures it is not possible to exclude any of them. Thus, having a quantifiable basis to distinguish between how likely each of which is the actual cause offers a more realistic representation. That basis is probability. An example of an additional measure is further observing “Fever” reinforce the belief that “Cold” is the actual cause of “RunnyNose”.

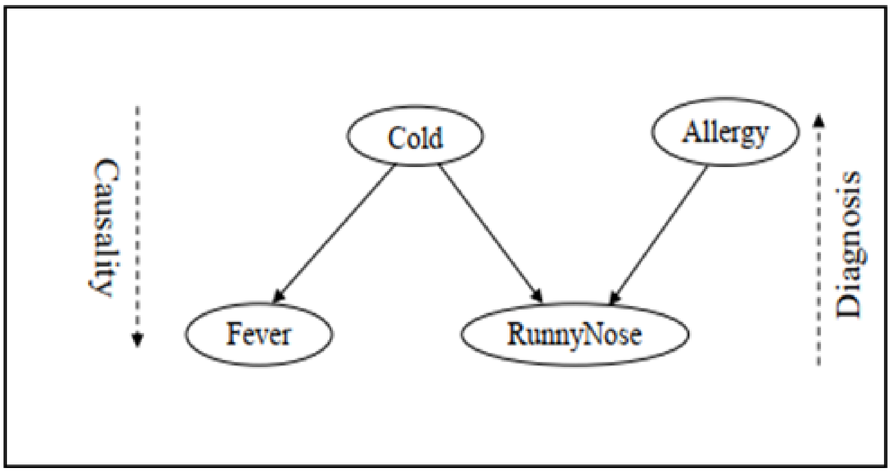


Figure 2.9: A causal network model

The use of possibility theory instead of the well known probability theory is because,in real life applications when not enough data is collected to set accurate probabilities1, possibility offers a better alternative [41]. Moreover, possibility is less sensitive to uncertainty measurement errors [42]. In fact, it has already been used in diagnostic approaches as an uncertainty model in both single [43] and multiple [41] fault diagnosis in centralized contexts.

### 2.6.2 Centralized Diagnosis (Formalization)

It is important to show a diagnostic process in the formalized framework to be used later.2 When following a causal scheme to diagnosis using a Petri net model, it is important to make the projection of a causal model on a PoPN framework, which would be as following:

- A place corresponds to state of a causal model, hence three types of places could be distinguished accordingly;
- A transition represents the cause-effect relationship;
- A source place corresponds to an initial state;
- A sink place corresponds to either an internal state or a manifestation.

We follow the diagnostic problem definition presented in [44] as $DP = (BM, Ctx, \langle \Psi^+ , \Psi^- \rangle$ .Where *DP* stands for the diagnostic problem, *BM* represent the behavioral model of the system to be diagnosed, *Ctx* is the set of possible fault hypotheses (the observations have to be explained by means of elements of *Ctx*), $\langle \Psi^+ , \Psi^- \rangle$ represent the made observation such that $\Psi^+$ is for manifestations to be entailed by a diagnosis and $\Psi^-$ is for manifestations not to be entailed (they are in conflict with the first ones). Such a problem has been reformulated to the context of Petri nets where, following the causal model scheme, a diagnostic solution is given in terms of source places (corresponding to initial nodes) that should have an initial marking $\mu^{ini}$ that is consistent with the made observations.

The illustrated definition of a diagnostic problem is considered to be an *"abduction problem with consistency constraints,"* in which a diagnosis could be seen logically as a set of assumptions ($\Delta \subseteq Ctx$) about the presence of a fault such that:

$$\forall\ m \in \Psi^+ : BM_i \bigcup \Delta \vdash\ m\ ;$$

$$\forall\ n \in \Psi^- : BM_i \bigcup \Delta \nvdash\ n\ ;$$

**Assumption 1 :** The PPN model we use is safe and irreflexive.

**Definition :**
A marking $\mu$ of a PPN is said to be final if there is no transitions to be fired at $\mu$ .
**Theorem 1**. In a marked PPN there is exactly one final marking.
*Proof.* This theorem can be proven in the same manner as theorem IV-B of [45], where the author used the properties: safeness, irreflexitivity and the absence of source transitions to sketch the final marking uniqueness from determinism.

The assumption that the net model is safe and irreflexive is a common one for diagnostic models. Furthermore, the projection of the diagnostic problem definition on a PPN framework would result the following definition. A possibilistic Petri net diagnostic problem is defined as $PPNDP = (N, P^{ini}, \langle$ P$^{+}$ , $P^{-} \rangle)$, such that: an initial marking $\mu^{ini}$ is a solution to $PPNDP$ if and only if the final marking $\mu$ of $N$ covers $P^{+}$ and zero-covers $P^{-}$ according to the following definition.

**Definition :** Let $Q \subseteq P$ and let $N = P,\ T,\ F$ be a Petri net, a marking $\mu$ of a $N$ is said to cover $Q$ if and only if $\forall$ p $\in Q \rightarrow \mu$ (p) = 1; while it is said to *zero-cover* $Q$ if and only if $\forall$ p $\in Q \rightarrow \mu$ (p) = 0 .

To explain the modeling methodology, we suggest the following example of a faulty PPN model.

| | State | Faulty value |
|---|---|---|
| Initial states | piston–state | worn |
| | ground –clearance | low |
| | oil–sump–state | worn |
| | spark–plague–mileage | high |
| | carbur–tuning | severe |
| Internal states | oil–consumption | high |
| | oil–sump | holed |
| | oil–lack | intense |
| | engine–temp | high |
| | incr–cool–temp | high |
| | cool–leakage | high |
| | spark–ign | irreg |
| | mixt | irreg |
| | mixt–ign | irreg |
| Manifestations | exhaust–smoke | black |
| | hole–in–oil–sump | yes |
| | oil–light | on |
| | temp–indic | red |
| | smoke–from–ing | yes |
| | acc–resp | irreg |

Table 2.3: States and their faulty values

**Example .** As a centralized example, let's re-use the one presented in [46]. We illustrate first how a diagnosis is performed in a general manner, then we demystify how PPNs

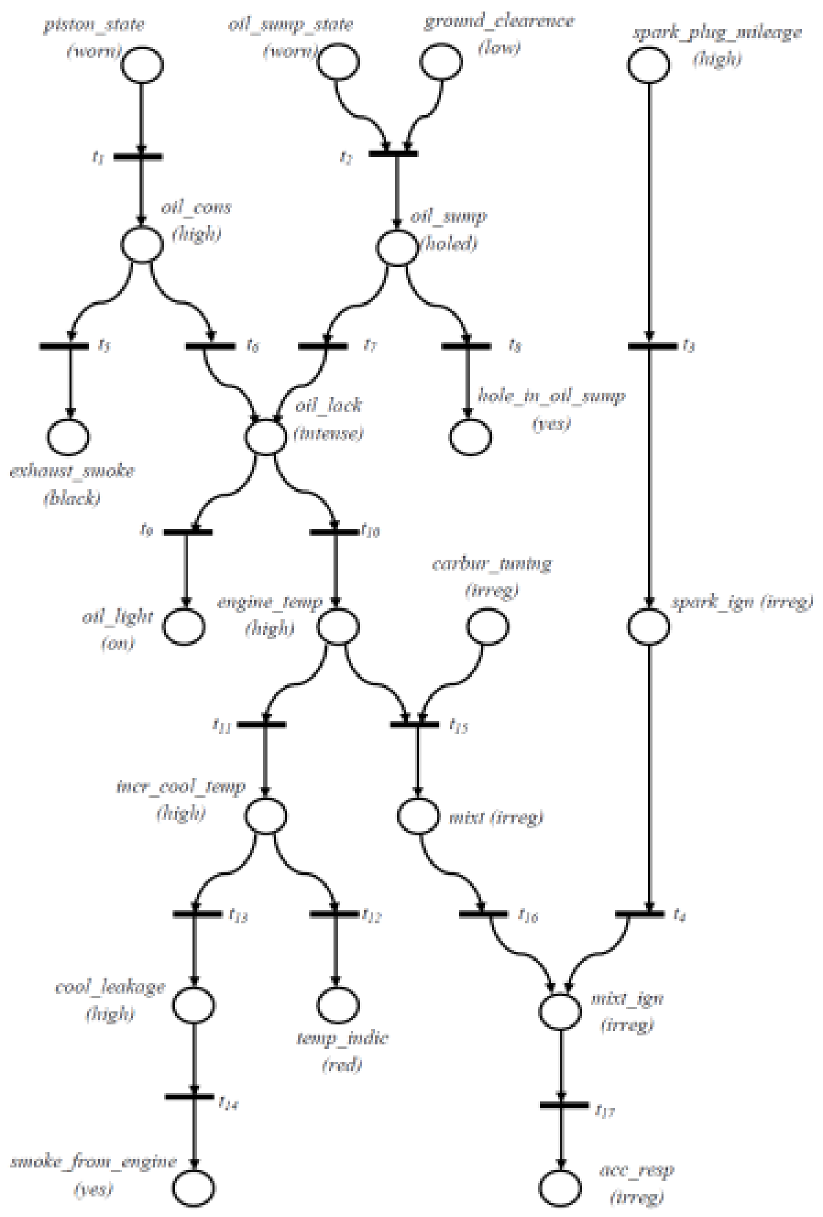


Figure 2.10: A simple example of a faulty behavior of a car

| t | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ |
|---|---|---|---|---|---|---|
| p | (0.9 , 1.0) | (0.5 , 1.0) | (0.4 , 1.0) | (0.5 , 1.0) | (0.8 , 1.0) | (0.6 , 1.0) |
| t | $t_7$ | $t_8$ | $t_9$ | $t_{10}$ | $t_{11}$ | $t_{12}$ |
| p | (0.8 , 1.0) | (0.9 , 1.0) | (0.9 , 1.0) | (0.8 , 1.0) | (0.9 , 1.0) | (0.9 , 1.0) |
| t | $t_{13}$ | $t_{14}$ | $t_{15}$ | $t_{16}$ | $t_{17}$ | |
| p | (0.9 , 1.0) | (0.9 , 1.0) | (0.8 , 1.0) | (0.7 , 1.0) | (0.7 , 1.0) | |

Table 2.4: Possibilities associated to transitions of Example above .

could be used in this context to model the uncertainty related to the diagnostic process. The three types of places are distinguishable as in Table 2.4 and the possibility measures associated to each transition are illustrated in Table 2.4. A diagnosis could be performed using a reachability graph or invariant analysis [47] [48], albeit the invariant analysis approach has been empirically shown to provide better results in terms of time needed to accomplish a diagnostic process in both centralized and distributed contexts [49] ,[46]. It is to be noted that the shown example is originally represented logically in terms of definite clauses without recursion of the form

$$\textit{piston–state}\ (\textit{worn}) \longrightarrow \textit{oil–cons}(\textit{high})\ .$$

### 2.6.3 T-invariant analysis

**Algorithm 1** : *T-invariant analysis*

```
Input : PPN model + OBS.
Output : minimal supports $\sigma$ with its possibility values.
Compute T-invariants :
for each pt do
if pt $\in PT^-$ then eliminate all source transitions which presented with pt in $PT^-$ .
else
using possibilistic values attached to transitions in $PT^+$ to compute possibilistic diagnoses
.
end if
end for
```

# Conclusion

This chapter was dedicated to intrducing key concepts of a less known uncertainty formalism which is the theory of possibility. It offers a different perception on the uncertain information with its two measures: possibility and necessity. Possibilistic Petri nets extend normal Petri nets to capture uncertainty according to this theory. Some basic properties and characteristics are demonstrated within this chapter as they will be helpful for causal model-based diagnosis. In particular, for uncertain reasoning to accomplish such a diagnosis.

# Chapter 3

# Development of diagnostic tool based on *PPN*

In the previous chapter, we presented a diagnostic technique based on PPNs. Our work consists in developing a tool for the analysis based on this technique. In order to achieve the reliability and operational safety of our tool, we will follow the classic software life cycle where the technical part of development be seen as the establishment of a sequence of descriptions more and more precise and closer to an executable program.

We start with requirments definition, followed by a well described architectural design and a detailed description of such a design and finally the implementation.

## 3.1 Requirments definition

This is the essential activity at the beginning of the software development process. Its purpose is to avoid developing a software that is not adequate. As we mentioned before, our objective is the realization of a diagnostic tool based on PPNs providing :

- A simple graphical user interface to edit a PPN model;
- A simple way to enter all manifestations so that the task of the diagnosis can begins;
- The diagnostic task must be passed through:
  - Computing all T-invriants minimal supports of the net model;
  - Pruning such supports by applying the described analysis technique;
  - Using possibilistic values attached to transitions to compute possibilistic diagnoses.

### 3.1.1 Design

The user does the graphical creation and editing of the possibilistic Petri nets, and at the same time these graphs will be automatically transformed into data structures. Data structures can be transformed into files (PNML) so that the user can export them or reimport them.

In our work, we are interested in diagnostic of a *PPN* model by a well-known algorithm base on a backward reachability analysis by exploiting T-invariants of the net model. We can schematize the general architecture of our system by figure 3.1.1.

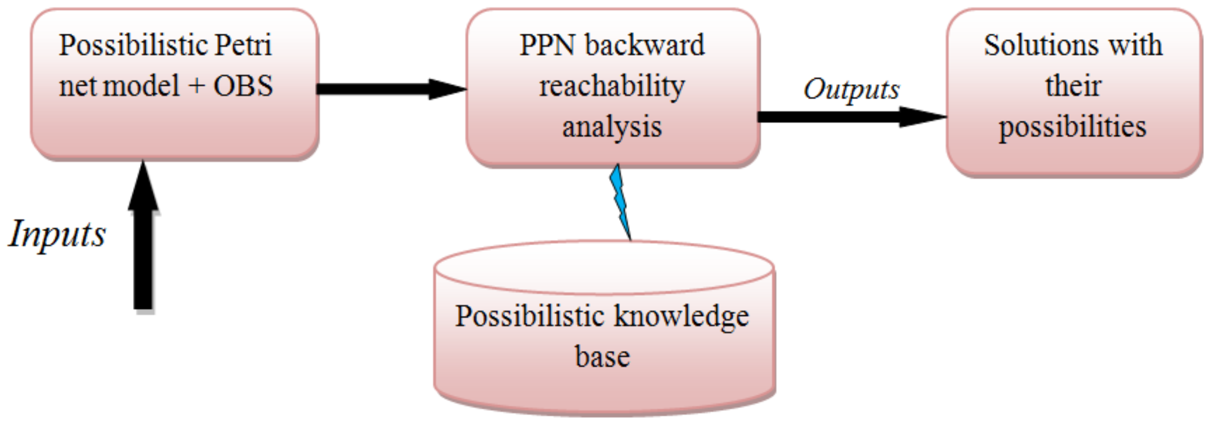

Possibilistic Petri net model + OBS
PPN backward reachability analysis
Outputs
Solutions with their possibilities
Inputs
Possibilistic knowledge base

Figure 3.1: The general architecture of the system

## 3.2 Detailed design

The detailed design provides for each component a description of how the functions of our diagnostic system are implemented in terms of algorithms and data structures .

- *Editing:* the editing module takes as input the system model in the form of a PPN graph representing the causal behavior of the system to be diagnosed.
- *Computation:* the computation module computes the set of all minimal supports of the T-invariants of the PPN model.
- *Analysis:* this module applies the technique presented in chapter 2.

The backward reachability analysis starts from the set of sink transitions that correspond to the abscent manifestations to discard all supports that contains source transtions that are present with those sink ones in the same support. From the remaining supports, the technique combines sources transtions to explain the manifestations in $\Psi^+$. Then, it uses the possibilistic values of all transitions contained in the used supports to compute the possibility of each solution.

Thus, rather than using the backward reachability graph, the implemented technique has the advantage to exploit the supports that are computed in an off-line manner.

## 3.3 Implementation :

After the analysis and design stages, we start the realization of our tool. This step allowed us to develop everything we studied in the previous chapter. This chapter represents the different computer tools and development languages that we have chosen to begin this work.

### 3.3.1 XML :

EXtensible Markup Language (XML) is a markup language that has been designed to store and transport data. The most important thing about this language is that it has designed to be both human and machine. We used this language to store our files (P / T, morphisms), from our tool.

### 3.3.2 PNML :

The Petri Net Markup Language (PNML) is a proposal of an XML-based interchange format for Petri nets. Originally, the PNML was intended to serve as a file format for the Java version of the Petri Net Kernel . But, it turned out that currently several other groups are developing an XML-based interchange format too. So, the PNML is only one contribution to the ongoing discussion and to the standardization efforts of an XML-based format .We used PNML as a format for the description of Petri nets imported or exported from our tool to give the possibility to exchange models with other external tools.

### 3.3.3 NetBeans IDE 8.0.2 :

NetBeans is an integrated development environment (IDE) for Java. NetBeans allows applications to be developed from a set of modular software components called modules. NetBeans runs on Microsoft Windows, macOS, Linux and Solaris. In addition to Java development, it has extensions for other languages like PHP, C, C++, HTML5, Javadoc, and Javascript. Applications based on NetBeans, including the NetBeans IDE, can be extended by third party developers. NetBeans is also a platform that allows the development of specific applications (Swing library (Java)). The NetBeans IDE builds on this platform.

NetBeans IDE and NetBeans Platform are based on software from netbeans.org, which has been dual licensed under the Common Development and Distribution License (CDDL) and the GNU General Public License version 2 with Classpath exception. For more information, please visit www.netbeans.org.

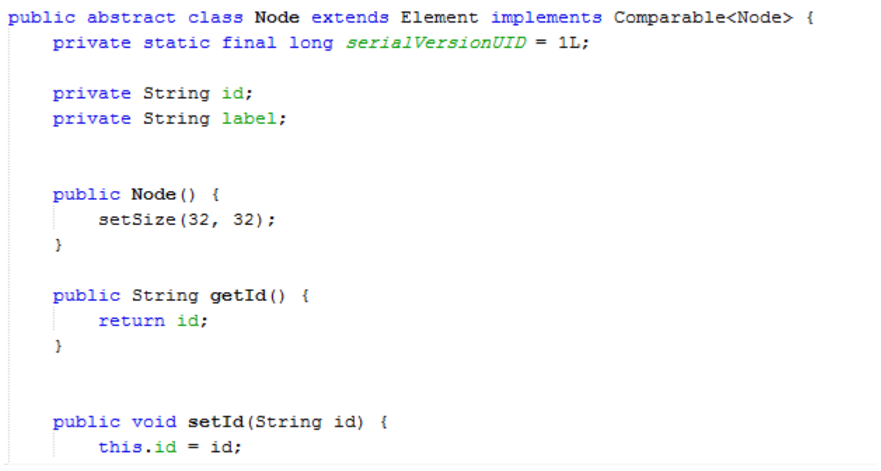
Figure 3.2: The NetBeans logo

### 3.3.4 LATEX :

LaTeX is widely used in academia for the communication and publication of scientific documents in many fields, including mathematics, statistics, computer science, engineering, chemistry, physics, economics, linguistics, quantitative psychology, philosophy, and political science. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials, such as Tamil, Sanskrit and Greek. LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language.

## The main classes of our tool :

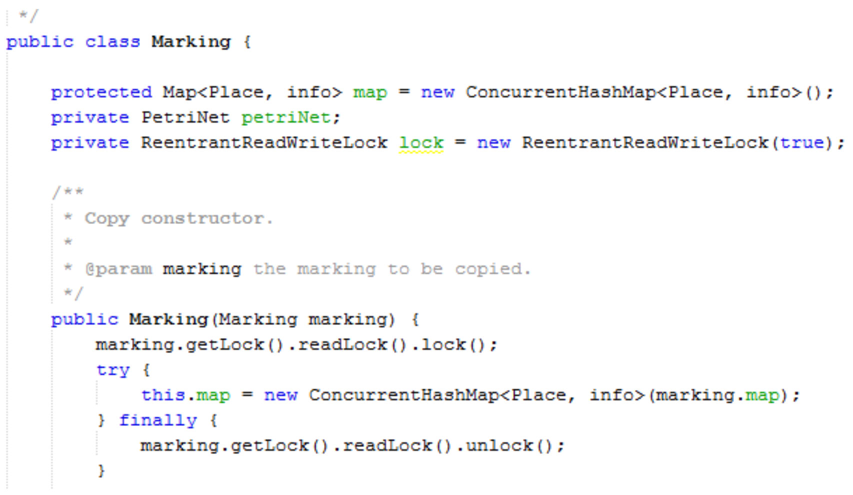
Starts with the class 1 which is used to represent the places of each classical proposition $r_i$ .



```
public abstract class Node extends Element implements Comparable<Node> {
    private static final long serialVersionUID = 1L;

    private String id;
    private String label;


    public Node() {
        setSize(32, 32);
    }

    public String getId() {
        return id;
    }


    public void setId(String id) {
        this.id = id;
```

Figure 3.3: Class represent place

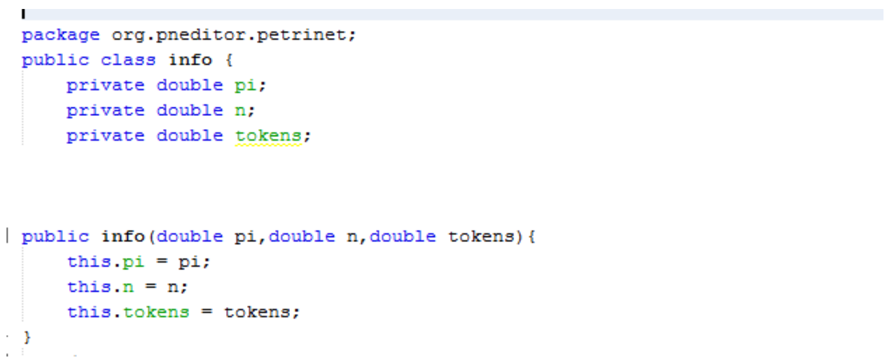
The second results is for possibilistic token :



```
 */
public class Marking {

    protected Map<Place, info> map = new ConcurrentHashMap<Place, info>();
    private PetriNet petriNet;
    private ReentrantReadWriteLock lock = new ReentrantReadWriteLock(true);

    /**
     * Copy constructor.
     *
     * @param marking the marking to be copied.
     */
    public Marking(Marking marking) {
        marking.getLock().readLock().lock();
        try {
            this.map = new ConcurrentHashMap<Place, info>(marking.map);
        } finally {
            marking.getLock().readLock().unlock();
        }
```

Figure 3.4: Class represent possibilistic token

With class *info* whitch is resresent the possibility and necessity measures :



```java
package org.pneditor.petrinet;
public class info {
    private double pi;
    private double n;
    private double tokens;


public info(double pi,double n,double tokens){
    this.pi = pi;
    this.n = n;
    this.tokens = tokens;
}
```

## 3.4 Conclusion

We have seen in this chapter two part, the first part is the tools and languages of development, i e the different computer tools and languages of development that we have chosen to start this work, and the second part is part of implementation, ie the step of producing a tool for the specification .

# Part III

# General conclusion

In this manuscript, a model combining the Petri net and the possibility theory has been presented. The aim of this approach is to allow reasoning on an imperfect knowledge about a system state.

The main issue treated was also the quantification of the uncertainty associated with diagnosis, particularly in the centrelized context. To do so, this time, a class of high-level Petri nets called Possibilistic Petri nets has been used to capture uncertainty on the ground of possibility theory.

We have seen how the transition firings (or pseudo-firings) can update and revise the knowledge about the system state represented by the marking. We must point out that these notions are completely consistent with the theory of Petri nets, because they do not change the set of actual reachable markings. A possibilisic analysis algorithm for diagnosing PPN was presented in this work by explaining the reasoning of backward reachability analysis in possibilistic Petri net model. The analysis itself exploits the structural properties of the model (the T-invariants). An implementation of such a technique has been realized.

# Bibliography

[1] L. A. Zadeh, “Fuzzy logic = computing with words,” IEEE Trans. Syst.,Man, Cybern., vol. 4, pp. 103–111, May 1996

[2] “From computing with numbers to computing with words—From manipulation of measurements to manipulation of perceptions,” IEEE Trans. Circuits Syst. I, vol. 45, pp. 105–119, Jan. 1999

[3] E. H. Shortliffe, Computer Based Medical Consultations: MYCIN. New York: Elsevier, 1976.

[4] N. J. Nilsson, “Probabilistic logic,” Artif. Intell., vol. 28, no. 1, pp.71–87, AU: WHAT MONTH? 1986.

[5] G. Shafer, “Mathematical theory of evidence,” in Mathematical Theory of Evidence. Princeton, NJ: Princeton Univ. Press, 1976.

[6] D. Dubois, J. Lang, and H. Prade, “Possibilistic logic,” in Handbook of Logic in Artificial Intelligence and Logic Programming, D. M. Gabbay,C. J. Hogger, and J. A. Robinson, Eds. Oxford, U.K.: Clarendon, 1994, pp. 439–513..

[7] “ A possibilistic-logic-based approach to integrating imprecise and incertain information ,” Fuzzy Sets syst .,Vol 13.no.2,pp.309-322, AU : WHAT MONTH? 2000 .

[8] P. P. Bonissone, “Editorial: Reasoning with uncertainty in expert systems,” Int. J. Man-Mach. Stud., vol. 22, pp. 241–250, AU: WHAT MONTH? 1985.

[9] A Giordana and L . Satta,“ Modeling production rules by means of predicate transition networks,” Inf. Sci., vol. 35, pp. 1–41, AU: WHAT MONTH? 1985..

[10] A. J. Bugarin and S. Barro, “ Fuzzy reasoning supported by Petri nets,” IEEE Trans. Fuzzy Syst., vol. 2, pp. 135–150, May 1994.

[11] S. M. Chen, J. M. Ke, and J. F. Chang, “Knowledge representation using fuzzy Petri nets,” IEEE Trans. Knowledge Data Eng., vol. 2, pp. 311–319, Sept. 1990..

[12] A. Konar and A. K. Mandal ,“ Uncertainty management in expert systems using fuzzy Petri nets,” IEEE Trans. Knowledge Data Eng., vol. 8, pp. 96–105, Feb. 1996.

[13] C.G .Looney,“ Fuzzy Petri nets for rule-based decision making ”, IEEE Trans. Syst.,Man , Cybern.,vol .18,pp.178-183,Jan/Feb .1988 .

[14] H. Scarpelli, F. Gomide, and R. Yager, “A reasoning algorithm for high level fuzzy Petri nets,” IEEE Trans. Fuzzy Syst., vol. 4, pp. 282–294, Aug. 1996.

[15] T. Murata and D. Zhang, “A predicate-transition net model for parallel interpretation of logic programs,” IEEE Trans. Software Eng., vol. 14, pp. 481–497, Apr. 1988.

[16] T. Shimura, J. Lobo, and T. Murata, “An extended Petri net model for normal logic programs,” IEEE Trans. Knowledge Data Eng., vol. 7, pp. 150–162, Feb. 1995

[17] C. Wu and S. Lee, “Enhanced high-level Petri nets with multiple colors for knowledge verification/validation of rule-based expert systems,” IEEE Trans. Systems, Man, Cybern. B, vol. 27, pp. 760–773, Sept. 1997.

[18] A. K. Zaidi and A. H. Levis, “Validation and verification of decision making rules,” Automatica, vol. 33, no. 2, pp. 115–169, 1996.

[19] L. A. Zadeh, “Fuzzy Sets as a basis for a theory of possibility,” Fuzzy Sets Syst., vol. 1, pp. 3–28, 1978.

[20] P. Kemper, “ Transient analysis of superposed GSPN’s,” IEEE Trans.Software Eng ., vol. 25, pp. 182-193, Mar./Apr. 1999 .

[21] C. Lin and D.C.Marinescu,“Stochastic high-level Petri nets and applications,” IEEE Trans.Comput.,vol.37,pp.815-825,July 1988 .

[22] C. Lin, Y. T Wu, and B. Li,“ Modeling probabilistic logic using Petri nets ,” in Proc.IEEE Systems,Man Cybernetics Conf., 1996,pp.864-869 .

[23] J. Cardoso, R. Valette, and D. Dubois, “Possibilistic Petri nets,” IEEE Trans. Syst., Man, Cybern. B, pp. 573–582, Oct. 1999.

[24] C. J. Chao and F. P. Cheng, “Fuzzy pattern recognition model for diagnosing cracks in RC structures,” J. Computing Civil Eng., ASCE, vol. 12, no. 2, pp. 111–119, AU: WHAT MONTH? 1998.

[25] R. Valette, "Les Réseaux de Petri", LAAS-CNRS Toulouse, septembre2002

[26] J.-M. Proth and X. Xie, Petri nets: a tool for design and management of manufacturing systems, John Wiley  Sons, 1996

[27] C. Cassandras and S. Lafortune, Introduction to Discrete Event Systems, Springer, 2007

[28] Mandrioli, D., A. Morzenti, M. Pezze, P. Pietro S. and S. Silva. 1996. A Petri net and logic approach to the specification and verification of real time systems. In: Formal Methods for Real Time Computing (C.Heitmeyer and D. Mandrioli eds), John Wiley Sons Ltd.

[29] Petri Nets for Dynamic Event-Driven System Modeling Jiacun Wang .Department of Software Engineering .Monmouth University West Long Branch, NJ 07764.

[30] Didier Dubois, Jérôme Lang, and Henri Prade. Possibilistic logic 1. 1994.

[31] Apports de la théorie des possibilités et des fonctions de croyance à l’analyse de données imprécises Marie-Hélène Masson Mémoire présenté en vue de l’obtention du diplôme d’Habilitation à Diriger des Recherches Habilitation à diriger des Recherches soutenue le 2 décembre 2005.

[32] C Negoita, L Zadeh, and H Zimmermann. Fuzzy sets as a basis for a theory of possibility. Fuzzy sets and systems, 1(3-28):61–72, 1978.

[33] Luigi Portinale. Exploiting t-invariant analysis in diagnostic reasoning on a petri net model. In Application and Theory of Petri Nets 1993, pages 339–356. Springer,1993.

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

[35] Nils J Nilsson. Probabilistic logic. Artificial intelligence, 28(1):71–87, 1986 .

[36] Jonathan Lee, Kevin FR Liu, and Weiling Chiang. A possibilistic-logic-based approach to integrating imprecise and uncertain information. Fuzzy Sets and Systems, 113(2):309–322, 2000.

[37] Jonathan Lee, Kevin FR Liu, and Weiling Chiang. Modeling uncertainty reasoning with possibilistic petri nets. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 33(2):214–224, 2003.

[38] Judea Pearl. Causality. Cambridge university press, 2009 .

[39] Luca Console and Pietro Torasso. Hypothetical reasoning in causal models. International Journal of Intelligent Systems, 5(1):83–124, 1990.

[40] Uffe B Kjærulff and Anders L Madsen. Probabilistic networks: an introduction to bayesian networks and influence diagrams. Recuperado em, 30, 2005.

[41] Koichi Yamada. Diagnosis under compound effects and multiple causes by means of the conditional causal possibility approach. Fuzzy sets and systems, 145(2):183–212, 2004.

[42] Didier Dubois and Henri Prade. Epistemic entrenchment and possibilistic logic.Artificial Intelligence, 50(2):223–239, 1991.

[43] Didier Dubois, Michel Grabisch, Olivier De Mouzon, and Henri Prade. A possibilistic framework for single-fault causal diagnosis under uncertainty*. INTERNATIONAL JOURNAL OF GENERAL SYSTEM, 30(2):167–192, 2001.

[44] Luca Console and Pietro Torasso. A spectrum of logical definitions of model-based diagnosis1. Computational intelligence, 7(3):133–141, 1991 .

[45] Luigi Portinale. Behavioral petri nets: a model for diagnostic knowledge representation and reasoning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 27(2):184–195, 1997.

[46] Luigi Portinale. Exploiting t-invariant analysis in diagnostic reasoning on a petri net model. In Application and Theory of Petri Nets 1993, pages 339–356. Springer, 1993.

[47] Kurt Lautenbach. Linear algebraic techniques for place/transition nets. In Petri nets: central models and their properties, pages 142–167. Springer, 1987 .

[48] Gérard Memmi and Gérard Roucairol. Linear algebra in net theory. In Net Theory and Applications, pages 213–223. Springer, 1980.

[49] Hammadi Bennoui. Interacting behavioral petri nets analysis for distributed causal model-based diagnosis. Autonomous agents and multi-agent systems, 28(2):155–181, 2014.

[50] C. Anglano and L. Portinale, B-W Analysis: A Backward Reachability Analysis for Diagnostic Problem Solving Suitable to Parallel Implementation, Application and Theory of Petri Nets, LNCS, vol.815, pp.39-58, 1994