Abstract

Currently, complex industrial systems and equipments, performing critical applications, require a very high reliability. Therefore, reliability analysis of these systems becomes very important and helpful especially for evaluating their reliability during their mission time. In studies of system reliability, the information available on the behavior of the system components' reliability usually incorporate epistemic uncertainties. The evidence theory, offering flexible and powerful tools to model uncertain knowledge in the epistemic meaning has become of a great interest for the reliability studies of industriel systems.

Due to the increasing complexity of these industrial systems, the evaluation of their reliability is considered as a very difficult and complex task. The use of graphical models to estimate system reliability is a great solution to overcome this complexity.

The main purpose of this master thesis is to apply two evidential models, based on the theory of belief functions, to the study of complex systems reliability under uncertainty. This dessertation presents a qualitative and quantitative comparison between the two evidential models applied to the reliability assessment of some well-known systems in the reliability field in order to show the power and the capacity of uncertainty representation and reasoning of each one.

Résumé

Actuellement, les systèmes industriels complexes et les équipements utilisés dans les applications critiques requierent une fiabilité importante. L'analyse de la fiabilité de ces systèmes devient très importante et utile pour évaluer leur fiabilité durant leur période d'exploitation. Dans les études de la fiabilité des systèmes, les informations disponibles sur le comportement de la fiabilité des composants du système incorporent généralement des incertitudes épistémiques. La théorie des fonctions de croyance qui offre des outils puisants pour modéliser ce type d'incertitude devient très intéressante pour traiter l'incertitude épistémique dans les études de la fiabilité des systèmes industriels. En raison de la complexité croissante de ces systèmes industriels, l'évaluation de leur fiabilité est considŕúne tâche très difficile et complexe. L'utilisation des modèles graphiques pour évaluer cette fiabilité offre un atout pour contourner le problème de la complexité. L'objectif principal de ce rapport de mastère est l'application de deux modèles graphiques évidentiels à l'étude de la fiabilité des systèmes complexes dans un contexte incertain. Cette dissertation présente une comparaison qualitative et quantitative entre les deux modèles appliqués à l'évaluation de la fiabilté de certains systèmes bien connus dans les études de la fiabilité, et ce afin de montrer la capacité de représentation et de raisonnement de chaque modèle.

Acknowledgements

At the end of this work, I wish to thank all persons who have helped me and contributed to this master report by giving their time, support, knowledge, and experience.

First of all, I would like to express my deepest gratitude to my advisors Dr. Boutheina Ben Yaghlane and Dr. Christophe Simon for their continuous encouragement, motivation, supervision, guidance and advice. I would like to sincerely acknowledge them for helping me in all the stages of research and for providing me the opportunity to work in an interesting and exciting field of research.

Besides my advisors, I gratefully acknowledge all my family members and friends for all their prayers, continuous support, love and good atmosphere they have always provided.

I extend my heartiest thanks to my dear friend Narjes whose encouragement and support, played a vital role during the preparation of this report.

As well, I would like to address special thanks to all professors of ISG Tunis and members of both laboratories LARODEC and CRAN for their help and kindness.

Finally, thanks to all the members of jury who accept to judge my dissertation.

Contents

1 Uncertainty Graphical Models **2**

1.1 Introduction . . . 2

1.2 Belief Functions Theory . . . 3

1.2.1 Frame of Discernment . . . 3

1.2.2 Basic Belief Assignment . . . 4

1.2.3 Belief and Plausibility Measures . . . 5

1.2.4 Commonality and Implicability Functions . . . 7

1.3 Belief Function Operations . . . 8

1.3.1 Combination . . . 8

1.3.2 Marginalization . . . 9

1.4 Graphical Models for Uncertainty representation . . . 9

1.4.1 Basic concepts of Graphical Models . . . 10

1.4.2 Bayesian Network . . . 12

1.4.3 Junction Tree . . . 14

1.4.4 Join Tree . . . 15

1.4.5 Binary join tree (BJT) . . . 17

1.4.6 Evidential Network (EN) . . . 18

1.4.7 Evidential Network with Conditional Belief Functions (ENC) . . . 19

1.4.8 Directed Evidential Network with Conditional Belief Functions (DEVN) . 19

1.5 Conclusion . . . 21

2 Reliability Analysis and Graphical Models **23**

2.1 Introduction . . . 23

2.2 System Reliability . . . 24

2.2.1 Definition of System Reliability . . . 24

2.2.2 System Reliability Analysis . . . 25

2.3 Traditional Techniques for Reliability Analysis . . . 26

2.3.1 Fault trees (FTs) 26
2.3.2 Reliability Block Diagrams (RBDs) 28
2.3.3 Markov Chains 29
2.4 Reliability Analysis and Uncertainty 30
2.5 Typical Systems in the Reliability Field 30
2.5.1 Series Systems 31
2.5.2 Parallel Systems 32
2.5.3 Parallel-Series Systems 33
2.5.4 Complex Systems 34
2.5.5 k-out-of-n Systems 36
2.5.6 Linear Consecutive-k-out-of-n Systems 38
2.5.7 Circular Consecutive-k-out-of-n Systems 39
2.6 Conclusion 40

3 Comparing Evidential Models for Imprecise Reliability 41
3.1 Introduction 41
3.2 Implementation Aspects 42
3.2.1 Implementation Aspects in the EN 42
3.2.2 Implementation Aspects in the DEVN 43
3.3 Comparing Aspects 49
3.3.1 Original Structure 49
3.3.2 Graphical Structures for Message Propagation 51
3.3.3 Message-Passing Schemes 53
3.3.4 Storage Efficiencies 56
3.3.5 Computational Efficiencies 59
3.3.6 Comparison of Models Results 62
3.4 Conclusion 63

Conclusion 65

Bibliography 67

List of Figures

1.1 General graph 10
1.2 Directed and undirected graphs 12
1.3 A Bayesian network model for the valve system : the qualitative part 14
1.4 A junction tree for the valve system 15
1.5 A subtree for the valve system obtained by the fusion algorithm 16
1.6 A binary join tree for the valve system 17
1.7 An evidential network with conditional belief functions 19
1.8 Modified Binary Join Tree for the valve system 21

2.1 Fault trees 27
2.2 A Reliability Block Diagram for a four-Components System 28
2.3 Markov Chain of the valve system 29
2.4 Reliability Block Diagram of a two-Components Series System 31
2.5 A Directed Acyclic Graph for the two-Components Series System 32
2.6 Reliability Block Diagram of a Two-Components Parallel System 33
2.7 A Directed Acyclic Graph for the two-Components Parallel System 33
2.8 Reliability Block Diagram of a Parallel-Series System 34
2.9 Directed Acyclic Graphs for a Parallel-Series System with six Components 35
2.10 Reliability Block Diagram for the Bridge System 35
2.11 Directed Acyclic Graph for the Bridge System 36
2.12 Reliability Block Diagram for a 2oo3 system 37
2.13 Reliability Block Diagram for a 1oo4 system 37
2.14 Two Directed Acyclic Graphs for the $2oo3$ System 38
2.15 Directed Acyclic Graph for the $1oo4$ System 38
2.16 A Directed Acyclic Graph for a Linear Consecutive-1-out-of-4 System 39
2.17 A Directed Acyclic Graph for a Circular Consecutive-1-out-of-3 System 40

3.1 Junction Tree for the two-components Parallel System 42

3.2 Junction Tree for the two-components Series System 42
3.3 Junction Tree for the 2oo3 System 43
3.4 Junction Tree for the 1oo4 System 43
3.5 Junction Tree for the Parallel-Series System 43
3.6 Junction Tree for the Bridge System 44
3.7 A Junction Tree for a Circular Consecutive-1-out-of-3 System 44
3.8 Modified Binary Join Trees for the parallel system and the series system 46
3.9 A modified Binary join tree for the '2oo3' system and the '1oo4' system 46
3.10 A Partial Multiply Connected Structure 48
3.11 The Modified Binary Join Tree for the Bridge System 49
3.12 The Modified Binary Join Tree for the Parallel-Series System 50
3.13 The propagation process in the MBJT using the GBT and the DRC 56
3.14 Evolution of operations of addition in the two structures 60
3.15 Evolution of operations of multiplication in the two structures 61

List of Tables

1.1 Combination of belief functions 9
1.2 Conditional probability functions of the valve system : the quantitative part . . . 14
1.3 A Comparison of Modified Binary Join Tree and Binary Join Tree 22

2.1 The truth table of the logical '*Or*' gate in a fault tree 27
2.2 The truth table of the logic '*And*' gate in a fault tree 27

Introduction

Motivation and Purpose

The reliability analysis of systems interests today many fields, since it is an important element of process control in companies which allows at once to understand the past and to predict the future. In studies of system reliability, the information on the behavior of the system components' reliability usually incorporate uncertainties (Kozine and Utkin, 2002). On one hand, these information are oftentimes collected from different experts that usually have varying assessments of the reliability for the same component. On the other hand, the complexity of the systems sometimes prevents the expert from estimating exactly the values of reliability associated with the different systems' parts.

The ignorance, the inconsistency and the incompleteness of available data are furthermore sources of uncertainty incorporated in systems (Simon et al., 2008). The uncertainty presents various forms. It may be either stochastic or epistemic. The first form of uncertainty occurs when the values of available data are uncertain. Whereas the second type of uncertainty occurs in the presence of incomplete or inconsistent information (Simon and Weber, 2009).

The problem which arises from the uncertainty has been treated by several theories in the literature such as the probability theory, the possibility theory (Dubois and Prade, 1988) and the evidence theory (Shafer, 1976).

The probabilistic formalism provides different tools to handle efficiently the stochastic uncertainty. For the epistemic uncertainty, the probability theory proposes a solution which is not always efficient consisting in using the uniform distribution to represent the total ignorance. In contrary, this type of uncertainty is well treated by the evidence theory which offers flexible and powerful tools to model it (Smets, 1978).

The theory of evidence also called Dempster-Shafer (DS)'s belief functions theory is of a major interest for the reliability studies of complex systems which incorporate uncertain knowledge in the epistemic meaning thanks to its capacity to effeciently represent this type of uncertainty.

Complex systems are in general composed of many parts which are related to each other. Since these systems are too large to easily compute their reliability, graphical models are an important asset which can be used in this context to represent efficiently the relations among the system's parts and to make the computational tasks manageable.

Indeed, graphical models provide a flexible way for describing the conditional dependencies and the connections between the variables of a complex system transforming it into a clear representation. Modelling complex systems for the analysis of their reliability under uncertainty by the means of the evidential network-based approaches is of a great interest.

Evidential networks are very powerful knowledge representations which allow at once graphical modeling and reasoning with the uncertainty in its random and epistemic meanings thanks to the theory of belief functions.

The main purpose of this dissertation is to apply two evidential networks to the study of complex systems reliability under uncertainty. The first one which is proposed by Simon and Weber (Simon et al., 2008; Simon and Weber, 2009) is based on the junction tree inference algorithm, using an extension of the Bayes' theorem to the representation of DS's belief functions theory. The second one which is proposed by Ben Yaghlane (Ben Yaghlane and Mellouli, 2008) is based on the modified binary join tree (MBJT) algorithm, using the disjunctive rule of combination (DRC) and the generalized Bayesian theorem (GBT), both proposed by Smets (Smets, 1993).
In addition, this dessertation presents a qualitative and a quantitative comparison between these two evidential models used to assessing the reliability of some well-known systems in the reliability field. Furthermore, we propose in this master thesis a comparison between two dynamic evidential networks to model the reliability of time-variant systems.

Master Contribution

In this dissertation, we propose three contributions consisting on:

- Applying two different evidential networks based on the DS's belief functions theory to the study of complex systems reliability with the intent to model and to reason under epistemic uncertainty.
- Presenting a comparison between these two evidential models applied to the study of complex systems reliability with uncertain knowledge in order to show the power and

the capacity of uncertainty representation and reasoning of each one. The comparison is based on different criterion specified progressively.

- Extending the previous comparison to dynamic evidential models.

Master Organization

The outline of this master report is as follows:

In Chapter 1, we first recall basic concepts when using the belief functions theory to model uncertainty incorporated in real world problems. Then we present some elements related to graphical models. Finally, we sketch some uncertainty graphical models using different formalisms to represent the uncertainty in the knowledge, including the evidential graphical models based on the evidence theory.

In Chapter 2, we first define the concept of system reliability. Then we present some traditional techniques used for the reliability analysis of systems. Next we describe some well-known systems in the reliability meaning for modelling their reliability under uncertainty by the means of evidential networks.

Chapter 3 is devoted to a qualitative and quantitative comparison between two directed evidential models described in Chapter 1. The comparison is based on different criteria and is led by the application of the two models on different examples of systems described in Chapter 2.

Finally, we summarize the results achieved in this master report and present some lines for future works.

1

Uncertainty Graphical Models

1.1 Introduction

Over the past several years, available information in the various real-world problems have been characterized by uncertainty, incompleteness, and even inconsistency. A wide variety of tools and hardware platforms have been developed to take into account the imperfection in the information when solving a real-world problem, including knowledge-based systems and expert systems.

Knowledge-based systems allow to represent, to store and to manage knowledge of a specific domain collected from experts or learning examples. They are able to handle uncertainty in the knowledge by various uncertainty theories including probability theory, possibility theory (Dubois and Prade, 1988), belief functions theory and Zadeh's fuzzy set (Zadeh, 1978). But the developing process of knowledge-based systems handling with uncertainty requires a great effort in knowledge acquisition and representation.

Several researches have been concentrated on using graphical models as another alternative for solving problems including uncertainty. These graphical models, also called network-based approaches, provide a general and convenient framework allowing to describe a real-world problem with a graph.

Graphical models have in common the fact that they provide representational and computational aspects making them a powerful and advantageous tool allowing efficiently representing knowledge and reasoning under uncertainty.

The aim of this chapter is to present some graphical models used in the literature for handling uncertainty in knowledge, and to sketch the main evidential networks which are graphs using the Dempster-Shafer's belief functions theory for the uncertainty management.

The rest of this chapter is structured as follows. In Section 2.2, we introduce the evidence theory and remind some basic concepts necessary when using this theory to model real-world problems. Next, in Section 2.3, we present the basic operations of the belief functions theory. Finally, we remind in section 2.4 some key aspects related to graphical models and briefly present well-known network models handling uncertainty.

1.2 Belief Functions Theory

The theory of belief functions, also called Dempster-Shafer (DS) theory, was first introduced by Dempster in 1967 (Dempster, 1967) for quantifying someone's beliefs, and was next developed by Shafer in 1976 (Shafer, 1976) to model the epistemic uncertainty. Later, several other interpretations of this theory were presented, including the Transferable Belief Model (TBM) proposed by Smets (Smets and Kennes, 1994).

The theory of belief functions is a general and flexible framework for dealing with uncertain information in stochastic and epistemic forms. It is a mathematical theory which can be interpreted as a generalization of the probability theory. In belief functions theory, belief masses which are used instead of probability values can be associated to singleton elements as well as to subsets of elements contrary to probabilities which are assigned only to atomic elements.

1.2.1 Frame of Discernment

Let $\Omega = \{h_1, \ldots, h_k, \ldots, h_n\}$ be a set of n mutually exclusive hypotheses related to a given problem. These elementary hypotheses correspond to the possible values of a random variable. Ω is called **frame of discernment or universe of discourse**.

Under a closed world assumption, Ω is considered to be exhaustive. That means that Ω covers all the possible values of the target variable. This assumption is the Dempster-Shafer's interpretation of belief functions theory (Dempster, 1967). Since it was sometimes difficult to enumerate all the possible hypotheses when establishing the frame of discernment related to a given problem, Smets (Smets, 1978) proposed his interpretation of belief functions theory: the open-world assumption.

Under an open-world assumption, Ω is not necessarily exhaustive, therefore we admit that an unknown and possible hypothesis (or more generally several hypotheses) can not be among the elements of the frame of discernment Ω.

The power set of Ω, denoted by 2^{Ω}, represents a set consisting of all the possible subsets of Ω and is defined as follows:

$$2^{\Omega} = \{A : A \subseteq \Omega\} \tag{1.1}$$

Every element of the power set is a proposition, an event or an assumption. Beliefs are expressed on these elements.

Example 1.1 Let us consider a problem of industrial system reliability assessment. An industrial system can have only two operating conditions, either the normal operating state {Up} or the state of failure {Down}.

Thus, the frame of discernment Ω_S related to the variable S representing the industrial system state is constituted of Up and $Down$:
$\Omega_S = \{Up,\ Down\}$.
The corresponding power set is:
$2^{\Omega_S} = \{\emptyset,\ \{Up\},\ \{Down\},\ \{Up, Down\}\}$.

The power set offers the possibility to represent the epistemic uncertainty by the epistemic state $\{Up, Down\}$ whose role is to characterize the ignorance on the real state of the system. In other words, the state $\{Up, Down\}$ denotes that the hypothesis Up or Down is respected but it is not known which one exactly. ■

1.2.2 Basic Belief Assignment

One of the most common ways to represent belief functions is the **basic belief assignment** (bba). The basic belief assignment is based on the allocation of a measure between 0 and 1 to express the degree of belief supplied by a source of information about an event, a proposition or an assumption.
A bba is a mapping $m : 2^{\Omega} \rightarrow [0, 1]$ such as:

$$\sum_{B \in 2^{\Omega}} m(B) = 1 \tag{1.2}$$

where the value $m(B)$, represents the degree of belief given by the source of information to support exactly B.

A **focal element** is an element E of 2^{Ω} for which the bba is strictly positive, $m(E) > 0$. For example $m(\{Up, Down\}) = 0.3$ implies that $\{Up, Down\}$ is a focal element.

The mass assigned to Ω, denoted by $m(\Omega)$, is the part of belief assigned to the frame of discernment. It expresses the uncertainty in the knowledge.

The mass assigned to $\emptyset$, denoted by $m(\emptyset)$, represents the degree of belief to support the fact that the real value of the variable might be not enumerated in the frame of discernment Ω. It translates an open-world assumption.

Example 1.2 Let us consider the previous problem of industrial system reliability assessment. Suppose that a source of information provides the following belief mass distribution:

$m(\{Up\}) = 0.7$
$m(\{Down\}) = 0.2$
$m(\{Up, Down\}) = 0.1$

This bba shows that the source believes that the system is working with a degree of belief 0.7 and that it has failed with a degree of belief 0.2.

As the source assignes a mass to the modality $\{Up, Down\}$, it means that the system can be exclusively in one of the two states {Up} or {Down} without knowing exactly which. This ignorance characterizes the epistemic uncertainty. ■

Several particular belief functions were proposed. We just present the Bayesian belief functions translating a strict Bayesian framework.

A **Bayesian belief function** is a belief function whose bba is defined such that all focal elements are singletons. This belief function is a probability distribution and its corresponding bba is defined as follows:

$$\begin{cases} m(B) \in [0,1] & \text{if } \mid B \mid = 1 \\ m(B) = 0 & \text{otherwise} \end{cases} \tag{1.3}$$

where B is an element of the power set 2^{Ω}.

Example 1.3 Let us take the same example of industrial system reliability assessment and suppose now that the source of information provides these masses:

$m(\{Up\}) = 0.8$
$m(\{Down\}) = 0.2$

Since all the focal elements are singleton, this bba is a Bayesian belief function. Its corresponding probability distribution is:

$p(Up) = 0.8$ and $p(Down) = 0.2$. ■

1.2.3 Belief and Plausibility Measures

Belief function and plausibility function are two non-additive measures which can be deducted from basic belief masses. These functions express the same piece of information as the belief mass distribution (Smets, 1978).

Belief and plausibility measures are the lower and the upper bounds of a probability interval (Shafer, 1976). This interval contains the probability of a set of hypotheses or a focal element.

Belief Function

The measure of belief function of a subset B of Ω, denoted by $\mathbf{bel}(B)$, quantifies the total amount of support given to B.

bel(B) is obtained by summing the belief masses of all subsets C of B (*i.e.* $C \subseteq B$). The belief function is a mapping bel : $2^{\Omega} \rightarrow [0,1]$ such as:

$$B \mapsto \sum_{C \subseteq B, C \neq \emptyset} m(C) \tag{1.4}$$

Example 1.4 Let us consider the same problem described previously and suppose that the source gives the same bba as in example 1.2.

A transformation of this belief function distribution gives us the following belief functions:
$bel(\{Up\}) = 0.7$
$bel(\{Down\}) = 0.2$
$bel(\{Up, Down\}) = 1$

■

Plausibility Function

The measure of plausibility function of a subset B of Ω, denoted by $\mathbf{pl}(B)$, quantifies the maximum amount of support that could be given to B.

pl(B) is measured by summing the belief masses of propositions C compatible with B (*i.e.* $B \cap C \neq \emptyset$). The plausibility function is a mapping pl : $2^{\Omega} \rightarrow [0,1]$ such as:

$$B \mapsto \sum_{B \cap C \neq \emptyset} m(C) \tag{1.5}$$

Or

$$pl(B) = 1 - bel(\bar{B}) \tag{1.6}$$

Example 1.5 Let us consider the same problem described previously and suppose that the source gives the same bba as in example 1.2.

A transformation of this belief mass distribution gives us the following plausibilities:
$pl(\{Up\}) = 0.8$
$pl(\{Down\}) = 0.3$
$pl(\{Up, Down\}) = 1$

■

The **bounding property** (Shafer, 1976) is defined by the following equation:

$$bel(B) \leq Pr(B) \leq pl(B) \tag{1.7}$$

where $Pr(B)$ defines the occurrence probability of a focal set B which is unknown. It can take any value in [$bel(B)$,$pl(B)$].

Example 1.6 After obtaining both of belief measure (example 1.4) and plausibility measure (example 1.5), we can for example define the interval containing the probability of B=$\{Down\}$ whose lower and upper bounds are respectively $bel(\{Down\})$ and $pl(\{Down\})$ as follows:
$Pr(Down) \in [0.2, 0.3]$ ■

1.2.4 Commonality and Implicability Functions

Commonatily and implicability functions are also transformations of belief functions. These functions have no signification but they are useful to simplify computation.

The **implicability function** is defined as follows:

$$\begin{aligned} & b : 2^{\Omega} \rightarrow [0,1] \\ & A \mapsto \sum_{B \subseteq A} m(B) = bel(A) + m(\emptyset) \end{aligned} \tag{1.8}$$

The **commonality function** is defined as follows:

$$\begin{aligned} & q : 2^{\Omega} \rightarrow [0,1] \\ & A \mapsto \sum_{A \subseteq B} m(B) \end{aligned} \tag{1.9}$$

Example 1.7 Let us consider the same problem of industrial system reliability assessment and suppose that the source gives the following bba:
$m(\emptyset) = 0.1$
$m(\{Up\}) = 0.6$
$m(\{Down\}) = 0.2$
$m(\{Up, Down\}) = 0.1$
The corresponding implicability function is as follows:
$b(\emptyset) = 0.1$
$b(\{Up\}) = 0.7$
$b(\{Down\}) = 0.3$
$b(\{Up, Down\}) = 1$
The corresponding commonality function is as follows:
$q(\emptyset) = 1$
$q(\{Up\}) = 0.7$
$q(\{Down\}) = 0.3$
$q(\{Up, Down\}) = 0.1$

■

1.3 Belief Function Operations

In this section, we present some basic operations of the belief functions theory: the combination and the marginalization. These two operations are going to be very useful when propagating beliefs in graphical models.

1.3.1 Combination

Combination corresponds intuitively to the aggregation of knowledge (Shafer, 1976).
Several combination rules were proposed in the literature to combine the belief functions, including the Dempster's rule of combination (Dempster, 1967) and the conjunctive rule of combination of Smets (Smets, 1978).

Dempster's Rule of Combination

The Dempster's rule of combination is used under a closed world assumption (Dempster, 1967; Shafer, 1976), where the frames of discernment on which are defined the bba's to be combined are considered to be exhaustive.

Let $m_1^{D_1}$ be a bba defined on a domain D_1 under a frame of discernment Ω_{D_1}. Similarly let $m_2^{D_2}$ be another bba defined on a domain D_2 under a frame of discernment Ω_{D_2}.

- If $D_1 \equiv D_2$, then the bba m_{12}^D resulting from the application of the Dempster's rule of combination is defined on the frame of discernment $\Omega_D \equiv \Omega_{D_1} \equiv \Omega_{D_2}$ as follows:

$$m_{12}^D = m_1^{D_1} \oplus m_2^{D_2}(A) = \begin{cases} \dfrac{\sum\limits_{B \cap C = A} m_1^{D_1}(B) \times m_2^{D_2}(C)}{1 - \sum\limits_{B \cap C = \emptyset} m_1^{D_1}(B) \times m_2^{D_2}(C)} & \forall A \subseteq \Omega_D,\ A \neq \emptyset \\ 0 & \text{if } A = \emptyset \end{cases} \tag{1.10}$$

- If $D_1 \neq D_2$, then before applying the Dempster's rule of combination, the two bba's must be extended to the domain $D = D_1 \cup D_2$ as follows:

$$m_1^{D_1 \uparrow (D_1 \cup D_2)}(C) = \begin{cases} m_1^{D_1}(A) & \text{if } C = A \times \Omega_{D_2}, A \subseteq \Omega_{D_1} \\ 0 & \text{otherwise} \end{cases} \tag{1.11}$$

$$m_2^{D_2 \uparrow (D_2 \cup D_1)}(C) = \begin{cases} m_2^{D_2}(A) & \text{if } C = A \times \Omega_{D_1}, A \subseteq \Omega_{D_2} \\ 0 & \text{otherwise} \end{cases} \tag{1.12}$$

Conjunctive Rule of Combination (CRC) of Smets

The conjunctive rule of combination of Smets is used under an open world assumption (Smets, 1978), where a strictly positive belief mass can be assigned to the empty set. It corresponds to

a non-normalized version of the Dempster's rule described previously.

Let $m_1^{D_1}$ and $m_2^{D_2}$ be two different bba's defined on two domains D_1 and D_2 under two frames of discernment Ω_{D_1} and Ω_{D_2}.

If $D_1 \equiv D_2$, then the bba m_{12}^D resulting from the application of the conjunctive rule of combination is defined on the frame of discernment $\Omega_D \equiv \Omega_{D_1} \equiv \Omega_{D_2}$ as follows:

$$m_{12}^D(A) = (m_1^{D_1} \textcircled{\cap} m_2^{D_2})(A) = \sum_{B,C \subseteq \Omega_D, B \cap C = A} m_1^{D_1}(B) \times m_2^{D_2}(C) \tag{1.13}$$

If $D_1 \neq D_2$, then before applying the conjunctive rule of combination, the two bba's must be extended to the domain $D = D_1 \cup D_2$ as defined previously.

Example 1.8 In table 1.1, we present two bba's given by two different sources of information and related to the same variable S of the problem described previously in the example 2.1. The table shows the bba's resulting from the application of the Dempster's Rule of Combination and the application of the Conjunctive Rule of Combination. ■

Table 1.1: Combination of belief functions

	$\mathbf{m_1}$	$\mathbf{m_2}$	$\mathbf{m_1 \oplus m_2}$	$\mathbf{m_1 \textcircled{\cap} m_2}$
$\emptyset$	0	0.1	0	0.29
Up	0.7	0.6	0.87	0.62
$Down$	0.2	0.1	0.1	0.07
$Up \cup Down$	0.1	0.2	0.03	0.02

1.3.2 Marginalization

Marginalization corresponds intuitively to the coarsening of the knowledge (Shenoy, 1994). The marginalization of a bba m^D defined on a domain D produces a new bba $m^{D'}$ defined on a new domain $D' \subseteq D$. Let m^D be a bba defined on a domain D. The bba $m^{D \downarrow D'}$ resulting from the marginalization of this bba on a domain D' is defined as follows:

$$m^{D \downarrow D'}(d') = m^D(d', \Theta_D) \tag{1.14}$$

where $D' \subseteq D$ and $D' \neq \emptyset$.

1.4 Graphical Models for Uncertainty representation

Graphical models, also called network-based approaches, combine the graph theory with any theory dealing with uncertainty, to provide a general and convenient framework for an intuitive

description of the real-world problems. They are a good tool for representing and manipulating the dependencies among the variables of interest describing these real-world problems.

1.4.1 Basic concepts of Graphical Models

Some basic concepts related to graphical models are recalled in this section. For more information on graphical models, the reader is referred to (Pearl, 1988; Lauritzen, 1996; Jensen, 2001).

A **graphical model** consists of two distinct parts: the **qualitative** part and the **quantitative** part.

The qualitative part of a graphical model is represented by the graphical structure or the **graph** modeling a given problem. While the quantitative part is expressed by the set of functions representing the **parameters** of the graphical model.

- **The qualitative part: the graph**
  A graph **G** is a mathematical structure defined as a couple (**V**, **E**), where V=$\{X_1, ..., X_n\}$ is a non empty finite set of vertices or nodes, and E is a set of links (X_i, X_j) such that $1 \leq i, j \leq n$, called edges. For example, Figure 2.1 shows a general graph in which V=$\{X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$ and E= $\{(X_1, X_2)$,(X_1, X_5),(X_2, X_5),(X_2, X_7),(X_3, X_6), (X_4, X_5),(X_5, X_3),$(X_6, X_5)\}$.

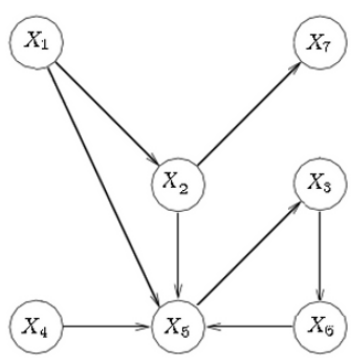

Figure 1.1: General graph

The edges in the graph encode the qualitative dependencies among the variables of the problem. They can be either directed (in directed graphs) or undirected (in undirected graphs). If (X_i, X_j) is undirected, then X_i and X_j are two adjacent nodes. If (X_i, X_j)

is directed, then X_i is the direct predecessor or the parent of X_j and X_j is the direct successor or the child of X_i. In Figure 2.1, the directed edge (X_4, X_5) points from the parent node X_4 to the child node X_5. X_4 is the direct predecessor of X_5 and X_5 is the direct successor of X_4.

Each node in G is associated with a variable of interest (or more generally a set of variables can be associated with a node). A root node is a node with no parents. A child node is a node with parents. A leaf is a node with no child. For example, the root nodes in the graph shown in figure 2.1 are X_1 and X_4. The child nodes are X_2,X_3,X_5,X_6 and X_7. X_7 is a leaf.

A **path** in G between two nodes X_i and X_j, denoted by path(X_i, X_j), represents a sequence of nodes such that there is an edge that connects each node and the next node in the sequence. In figure 2.1, $X_1X_2X_5X_3$ is a path between nodes X_1 and X_3.

A **cycle** in G is a path(X_i, X_i) such that the sequence of nodes starts and ends with the same node. $X_5X_3X_6X_5$ is a cycle in figure 2.1.

- **The quantitative part: the parameters of the graphical model**
  The parameters of the graphical model are the local functions specified for each node in the graph. These functions express the quantitative dependencies among the variables of the model.

The graphical model is considered as a simple way that describes the different connections between the variables of a problem transforming it into a clear and intuitive graphical representation. Graphical models are very useful because on one hand they allow to visualize the structure of the model describing a given problem, and on the other hand they provide a framework for efficient and easy computations.

Several types of graphical models were proposed in the literature by many authors for representing uncertainty. We can distinguish the directed graphical models and the undirected graphical models.

- A **directed graphical model** is a model in which the graph is directed, that means that all the edges of the graph have direction. Formally, a graph G is directed implies that if $(X_i, X_j) \in E$, then $(X_j, X_i) \notin E$ for all $X_i, X_j \in V$. The left side of figure 2.2 shows a directed graph (a). Directed graphs represent conditional dependencies between random variables (Pearl, 1988).

- An **undirected graphical model**, also called Markov Random Field or Markov network, is a model in which the graph is undirected, that means that all the edges have no direction. Formally, a graph G is undirected implies that if $(X_i, X_j) \in E$, then $(X_j, X_i) \in E$ for all

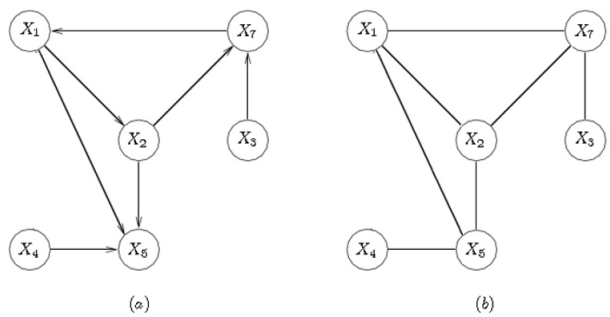

Figure 1.2: Directed and undirected graphs

$X_i, X_j \in V$. The right side of figure 2.2 shows an undirected graph (b).

Undirected graphs are used to represent the correlative relationships among the variables.

The main difference between directed and undirected graphical models is the manner in which the edges are defined and so the semantic significance of those edges in the two types of models. Edges in directed graphical models represent the causal relationships among the linked variables, while edges in undirected graphical models express interactions between these variables.

We present some well-known directed and undirected models for uncertainty representation and reasoning.

1.4.2 Bayesian Network

Bayesian networks (Pearl, 1988), called also Bayes nets or causal networks (Lauritzen and Spiegelhalter, 1988), are directed acyclic graphs (**DAG**), *i.e.* directed graphs that have no cycle. They are probabilistic graphical models. It means that Bayesian networks combine the graph theory with the probability theory for representing and handling uncertainty in the knowledge.

Formally, a Bayesian network (**BN**) is considered as a couple (**G**, **P**) where:

- G is the qualitative part of the BN. It is a DAG in which nodes represent the random variables of a given problem, and edges represent the conditional dependencies between these variables.

  Each node X_i in G has a set of possible instances or values represented by a frame Θ_{X_i}.

Each edge in G represents the causal relationship among the two connected nodes.

- P $= \{P(X_i|Pa(X_i))\}$ is the quantitative part of the BN. It is a set of local conditional probability functions specified for each node X_i in the graph given its parents, denoted by $Pa(X_i)$ such as:

$$\begin{cases} \sum_{x_i} P(x_i|Pa(X_i)) = 1 & \text{if } Pa(X_i) \ = \emptyset \\ \sum_{x_i} P(x_i) = 1 & \text{otherwise} \end{cases} \tag{1.15}$$

  where x_i denotes the instances of the random variable X_i .

The conditional probability functions are the basic concept for representing uncertainty in the Bayesian network.

The joint probability distribution over all the variables in the Bayesian network is expressed as the product of the conditional probability functions specified for each of those variables. This joint distribution, denoted by $P(V)$ is unique and is given by the **probabilistic chain rule** defined as follows:

$$P(V) = P(X_1, ..., X_n) = \prod_i P(X_i|Pa(X_i)) \tag{1.16}$$

where $V = \{X_1, ..., X_n\}$ is the set all the variables in the Bayesian network and $Pa(X_i)$ is the parent set of the variable X_i.

Figure 2.3 shows the graphical structure (DAG) of the Bayesian network modeling the valve system which is a parallel-series system (will be explained later in section 2.5.3) classically used in theoretical reliability analysis (Weber and Simon, 2008; Satoh et al., 1993; Tavakkoli-Moghaddam et al., 2008). The valve system's description is as follows:
The system consists of three valves used to distribute a fluid. Each valve Vi has two disjoint states $(\{Up\}, \{Down\})$: Up is the state of a valve when it is working, and $Down$ is its state when it has failed. The state of the system depends on its components' states and its reliability is computed using the probability distributions over the states of the three valves.

The variables of interest of the valve system are $V1$ (valve 1), $V2$ (valve 2), $V3$ (valve 3), $A1$ (logical And gate), $A2$ (logical And gate) and $O1$ (logical Or gate).

The qualitative dependencies between the variables of the valve system are represented by the edges shown in figure 2.3.

Table 1.2: Conditional probability functions of the valve system : the quantitative part

α:	p($V1$={Up})	= .8	ε:	p($A1$={Up}|$V1$={Up},$V2$={Up})	= 1
	p($V1$={$Down$})	= .2		p($A1$={$Down$}|$V1$={Up},$V2$={$Down$})	= 1
				p($A1$={$Down$}|$V1$={$Down$},$V2$={Up})	= 1
				p($A1$={$Down$}|$V1$={$Down$},$V2$={$Down$})	= 1
σ:	p($V2$={Up})	= .7	ξ:	p($A2$={Up}|$V1$={Up},$V3$={Up})	= 1
	p($V2$={$Down$})	= .3		p($A2$={$Down$}|$V1$={Up},$V3$={$Down$})	= 1
				p($A2$={$Down$}|$V1$={$Down$},$V3$={Up})	= 1
				p($A2$={$Down$}|$V1$={$Down$},$V3$={$Down$})	= 1
λ:	p($V3$={Up})	= .9	δ:	p($O1$={Up}|$A1$={Up},$A2$={Up})	= 1
	p($V3$={$Down$})	= .1		p($O1$={Up}|$A1$={Up},$A2$={$Down$})	= 1
				p($O1$={Up}|$A1$={$Down$},$A2$={Up})	= 1
				p($O1$={$Down$}|$A1$={$Down$},$A2$={$Down$})	= 1

The quantitative dependencies between the variables of the valve system are represented by the conditional probability functions given in table 2.2.

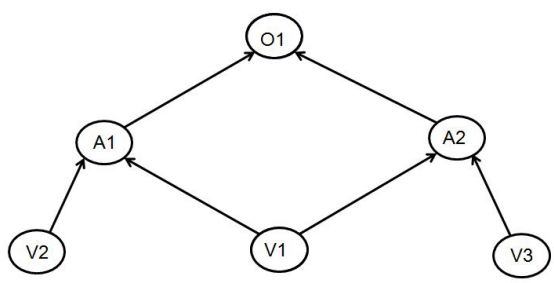

Figure 1.3: A Bayesian network model for the valve system : the qualitative part

1.4.3 Junction Tree

The junction tree (Jensen et al., 1990) is a data structure which enables local computations with potentials on small domains avoiding explicitly the computation of the marginal of the joint distribution for the random variables of interest describing a given problem.

The junction tree is a singly-connected and undirected graph whose nodes are subsets of random variables. It satisfies the running intersection property (Lauritzen and Spiegelhalter, 1988). It means that if a variable appears in two different nodes, it also appears on all nodes on the path linking them.

The nodes are called the cliques of the junction tree. Each separator between two adjacent nodes is labeled by the intersection of the two nodes.

A junction tree can be obtained from either a directed graph or an undirected graph.
To construct a junction tree from a DAG G, we first create a moral graph UG, then we triangulate UG. Next we identify the cliques of the triangulated graph, and finally we organize them into a junction tree. For further information, the reader is referred to (Jensen et al., 1990).

Figure 1.4 shows a junction tree for the valve system. This junction tree is obtained from the directed acyclic graph shown in figure 2.3.

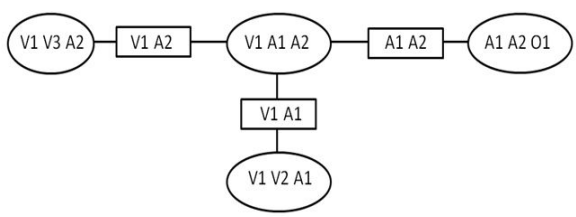

Figure 1.4: A junction tree for the valve system

1.4.4 Join Tree

The join tree, called also the Shenoy-Shafer (SS) architecture (Shenoy and Shafer, 1990), is another well-known data structure for local computations giving a solution to avoid computing the joint valuation over all the random variables of interest.

The join tree is an undirected and singly-connected graph consisting of a set of nodes V=$\{N_1, ..., N_n\}$, where each node represents a subset of random variables, such that if a variable is in two distinct nodes, then the variable must be in every node on the path between them.

Every node in the join tree has a domain or a label which corresponds to the set of random variables belonging to it. The join tree is initialized by assigning initial potentials to the corre-

sponding nodes. Every potential is assigned to a node whose domain corresponds to its domain.

To construct a join tree, an hypergraph H containing subsets of variables, a deletion sequence containing the random variables to be eliminated, and a set of valuations or potentials for subsets in H are needed as parameters.

The join tree construction process is based on **the fusion algorithm** (Shenoy, 1997) which is considered as a successive variables deletion: the key idea of this algorithm is that to delete a variable, we start by combining all the valuations whose domain contain it and then we marginalize the result of combination by the remaining variables. If we eliminate just one variable, we get a partial join tree.

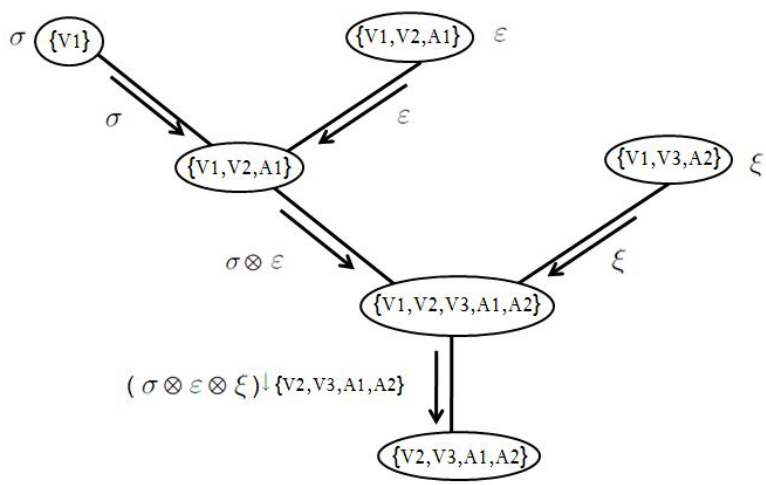

Figure 1.5: A subtree for the valve system obtained by the fusion algorithm

The reader who is interested in the construction process of a join tree can find the corresponding formal description in (Shenoy, 1991; Lauritzen and Shenoy, 1996).

The example shown in figure 2.6 illustrates the fusion algorithm applied to delete the variable V1 in the valve system. All the valuations containing V1 in their domains are combined as shown in the figure. The variable is then marginalized out of the combination. The obtained tree is a subtree for the valve system. To get a join tree for this system, the fusion algorithm is recursively applied using an elimination sequence.

1.4.5 Binary join tree (BJT)

The join tree, described previously in section 1.4.4, presents computational inefficiencies if a node has many neighbors (Shenoy, 1997). It requires a lot of unnecessary computations resulting from the redundant combinations done when computing marginals.

The concept of binary join trees (**BJTs**) was introduced by Shenoy (Shenoy, 1997) as a significant improvement of the Shenoy-Shafer (SS) architecture.

A binary join tree is a join tree in which each node has at most three neighbors (Lepar and Shenoy, 1998). The main idea of a binary join tree is that all combinations are done two at time, *i.e.* a binary fusion algorithm is performed to construct the binary join tree.

A binary join tree is computationally more efficient than the corresponding join tree since it reduces the number of combinations performed by storing intermediate results when inferring beliefs. In general in a join tree, m^2 combinations are done in each node with m neighbor nodes. For a corresponding binary join tree, constructed by the addition of some nodes, only $3(m-1)$ combinations are needed.

As for the join tree, the construction of the binary join tree is based on the fusion algorithm. Figure 2.7 gives the binary join tree corresponding to the Bayesian network of the valve system shown in figure 2.3.

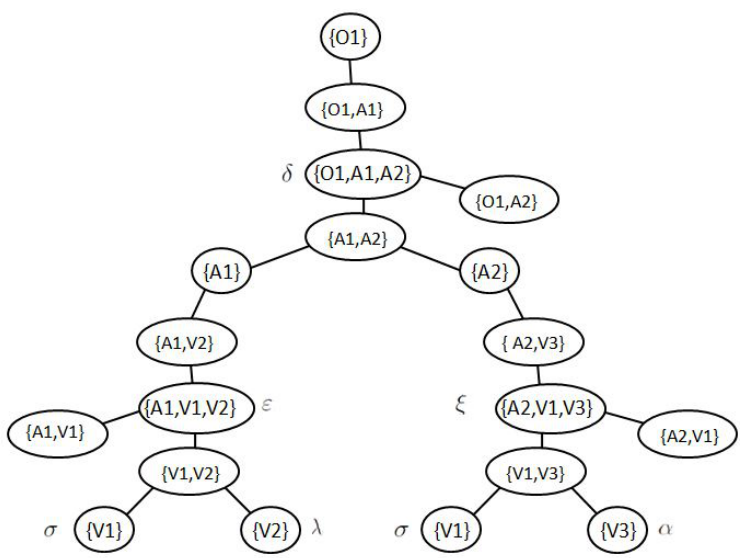

Figure 1.6: A binary join tree for the valve system

1.4.6 Evidential Network (EN)

The evidential network (**EN**), proposed by Simon and Weber (Simon et al., 2008; Simon and Weber, 2009), combines the Dempster-Shafer theory with Bayesian network to take random and epistemic uncertainties into account when modeling and reasoning. Thus, to represent the conditional dependencies between the variables in a description space integrating uncertainty, the evidential network uses conditional belief masses instead of using conditional probability functions as in BN.

Conditional belief masses are defined in the evidential network like conditional probability functions in the BN, that means, when a node X is not a root node, its conditional belief mass distribution is required for every possible combination of its parents' states.

The qualitative structure of an evidential network is represented by a DAG defined as a couple $G = (N, E)$, where N represents the set of nodes, and E represents the set of edges.

The quantitative level of an evidential network is represented by the set of belief mass distributions M associated to each node in the graph.

For each root node X, having a frame of discernment Ω constituted by q mutually exhaustive and exclusive hypotheses, an a priori belief mass distribution $M(X)$ has to be defined over the 2^q focal sets A_i^X by the following equation:

$$M(X) = [m(X \subseteq \emptyset)\ \ m(X \subseteq A_1^X)...m(X \subseteq A_i^X)...m(X \subseteq A_{2q-1}^X)] \tag{1.17}$$

with

$$m(X \subseteq A_i^X) \geq 0 \quad and \sum_{A_i^X, A_i^X \in 2^\Omega} m(X \subseteq A_i^X) = 1 \tag{1.18}$$

where $m(X \subseteq A_i^X)$ is the belief that variable X verifies the hypotheses of the focal element A_i^X.

For other nodes (*i.e.* child nodes), a conditional belief mass distribution $M[Pa(X)](X)$ is specified for each possible hypothesis A_i^X knowing the focal sets of the parents of X defined by $Pa(X)$.

To compute the marginal belief mass distributions of the variables in the evidential network, the exact inference algorithm proposed by Jensen (Jensen et al., 1990) is performed. The computation mechanism is based on an extension of the Bayes' theorem to the representation of uncertain information according to the framework of Dempster-Shafer theory (Simon and Weber, 2009). The algorithm used for inferring the beliefs relies on the construction of a secondary structure which is the junction tree described previously in section 1.4.3.

1.4.7 Evidential Network with Conditional Belief Functions (ENC)

Evidential networks with conditional belief functions, called **ENCs**, were initially proposed by Smets for the propagation of beliefs (Smets, 1993). Then, these networks have been studied by Xu and Smets (Xu and Smets, 1994).

An ENC has the same structure as Bayesian network, since it is a DAG. Nevertheless, the manner in which the conditional beliefs are defined is different from that one in which the conditional probabilities are defined in Bayesian networks: each edge in the graph represents a conditional relation between the two nodes it connects. For example, figure 1.7 shows an ENC with three variables $V1$, $V2$ and $A1$, where $V1$ and $V2$ are the parent nodes of $A1$. The edges (V1,A1) and (V2,A1) mean that we have $\{bel^{A1}[v1_i] : v1_i \in \Omega_{V1}\}$ and $\{bel^{A1}[v2_i] : v2_i \in \Omega_{V2}\}$, but not $\{bel^{A1}[v1_i, v2_i] : v1_i \in \Omega_{V1}, v2_i \in \Omega_{V2}\}$ as in a BN.
However, if conditional beliefs such as $\{bel^{A1}[v1_i, v2_i] : v1_i \in \Omega_{V1}, v2_i \in \Omega_{V2}\}$ are given, nodes

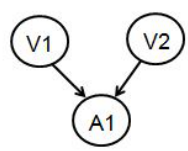

Figure 1.7: An evidential network with conditional belief functions

V1 and V2 are merged as one node in the ENC built by Xu' method. The belief function for the merged node is obtained by combining the ballooning extension of each conditional belief function.

1.4.8 Directed Evidential Network with Conditional Belief Functions (DEVN)

The graphical representation and propagation algorithm proposed by Xu are related to the evidential networks which have only binary relations between the nodes (Xu and Smets, 1994). Thus, in order to generalize ENC to the case where relations are given for any number of nodes, Ben Yaghlane proposed a new representation (Ben Yaghlane, 1992), called **DEVN** which means a directed evidential network with conditional belief functions.

The directed evidential network model combines the DAG and the evidence theory. As the Bayesian network, the DEVN is represented by two different parts which are:

1. The qualitative level is defined by the DAG in which nodes represent random variables and directed edges describe the conditional dependencies in the model.

2. The quantitative level is defined by a set of parameters for the DAG represented by conditional belief functions for each variable given its parents, expressing the dependence relations.
   A set of possible values, called frame of discernment, is defined for each variable X in the DEVN. The uncertainty is represented by an a priori belief function bel_0^x for each root node X, and by conditional belief function $bel^x[Pa_x]$ on X given the value taken by its parents denoted by Pa_x for the child nodes.

Directed evidential networks are graphically similar to Bayesian networks. They use conditional belief functions instead of using conditional probability functions.

As in the BN, the main objective of the reasoning process in DEVN is to compute the marginals of the global belief functions for all or some variables. Thus, Ben Yaghlane proposed a propagation algorithm for the DEVN based on the use of a secondary computational structure, called (modified) binary join tree (MBJT), which permits to maintain the (in)dependence relations of the original DEVN and to perform the propagation directly with the conditional belief functions (Ben Yaghlane and Mellouli, 2008).

Modified Binary Join Tree (MBJT)

The computational data structure, proposed by Ben Yaghlane to perform the propagation process (Ben Yaghlane and Mellouli, 2008), is an adaptation of the binary join tree (Shenoy, 1997), called the modified binary join tree (**MBJT**).

The MBJT was proposed as a refinement of the binary join tree. It was designed to avoid the major drawback of binary join tree which is the lost of useful information about the relationship among the variables arising when transforming the original directed evidential network into a binary join tree.

As for the binary join tree construction (Shenoy, 1997), the modified binary join tree construction process is based on the fusion algorithm with some modifications used for emphasizing explicitly the conditional relations in the initial DEVN.

The modified binary join tree integrates rectangles containing these conditional relations between the variables instead of circles containing just the set of these variables (Ben Yaghlane and Mellouli, 2008). Thus, by profiting efficiently of the original network's structure, the MBJT maintains the available independence relations between the variables in order to make them useful when inferring beliefs. The modified binary join tree for the valve system is shown in figure 1.8.

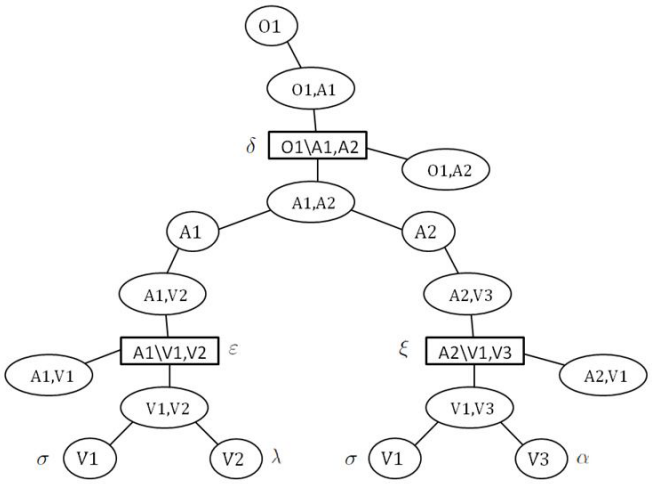

Figure 1.8: Modified Binary Join Tree for the valve system

Modified Binary Join Tree vs Binary Join Tree

Both binary join tree and modified binary join tree are data structure for local computations. The construction process of a BJT needs as parameters the subsets containing in an hypergraph and a set of potentials, while the construction process of a corresponding MBJT takes as parameters a directed evidential network initialized by joint and conditional belief functions added to an hypergraph and a set of valuations. A comparison of the BJT and the MBJT is shown in table 1.3. More detailed comparison can be found in (Ben Yaghlane, 1992).

1.5 Conclusion

In this chapter, we have recalled basic concepts necessary when using belief functions to model uncertainty in real world problems. We have presented some elements related to graphical models and we have described several uncertainty directed and undirected graphical models. These graphical models providing a flexible and intuitive framework for representing and handling imperfect data and uncertainty in knowledge have become very popular in several applications dealing with uncertain information. They provide a support for modeling this uncertainty and reasoning efficiently with it.

Table 1.3: A Comparison of Modified Binary Join Tree and Binary Join Tree

Features	**BJT**	**MBJT**
Graphical Structure		
1.**Type of graph**	hypergraph	hypergraph
2.**Representation of conditional relations**	joint form	joint and conditional forms
3.**Nodes**	joint nodes	joint and conditional nodes
Inference Procedure		
4. **Inference process**	Quantitative based on fusion algorithm	Quantitative based on fusion algorithm and GBT and DRC rules of Smets
5. **Nodes involved**	all the created nodes	only joint nodes

2

Reliability Analysis and Graphical Models

2.1 Introduction

Currently, complex industrial systems and equipments performing critical applications require a very high reliability or a very low probability of failure. Therefore, reliability analysis of these complex systems has become very important and helpful especially for evaluating their reliability during their mission time. Reliability analysis provides important information given for example as inputs to decision support systems for optimizing the maintenance operations (Donat et al., 2009).

A wide range of quantitative techniques have been developed for predicting the reliability of a system based on its structure and the individual reliability of each of its components. Some of these techniques, including the fault trees, the reliability block diagrams and the Markov chains, are well referred in different international standards like IEC 61025 (IEC61025, 1990) and IEC 61511 (IEC61511, 2000). Despite their capacities to model system reliability, these traditional techniques used in the reliability analysis have different important limitations (Simon et al., 2007).

In realistic applications, it is sometimes difficult to exactly predict the individual system components reliability on which the system reliability depends. As a result, the information available on the reliability of each component and then on the system reliability may incorporate uncertainty. The uncertainty incorporated in systems may be either stochastic or epistemic. The problematic arising from this uncertainty was an important reason which has led to the important developments of many reliability analysis tools involving the use of graphical models which allow at once to model complex system dependabilities and to quantitatively and qualitatively represent the uncertainty.

Recently, Bayesian networks have been applied in the reliability field for quantifying complex systems reliability (Torres-Toledano and Sucar, 1998; Langseth and Portinale, 2007). Bayesian networks are very interesting tools allowing to model the reliability of a complex system whatever its complexity is (Simon et al., 2008). They rely to the probabilistic formalism to model and to manage uncertainty in the knowledge.

The stochastic uncertainty incorporated in systems is well solved by the probabilistic formalism used in Bayesian networks. In contrary, the epistemic uncertainty can not be efficiently handled by the probability theory. The evidence theory, which allows to suitably handle this type of uncertainty, has become of a great interest for the reliability studies of complex systems which incorporate uncertain knowledge in the epistemic meaning.

The rest of this chapter is structured as follows. In section 3.2, we define the concept of system reliability. Then, we sketch in section 3.3 some quantitative methods for system reliability assessment and briefly present their limits. In section 3.4, we focus on the problem of uncertainty in the reliability field which has made the graphical models based on the DS's belief function theory a tool of interest and has led to their application in this field. Section 3.5 is devoted to describe several well-known configurations in the reliability field and to present their corresponding graphical representations.

2.2 System Reliability

In this section, we briefly define some concepts related to system reliability. For more information, the reader is referred to (Lindqvist and Doksum, 2003; IEC60870, 1990).

2.2.1 Definition of System Reliability

The concept of **system reliability** has been variously defined in the literature by several authors. Originally, reliability was considered as a qualitative property characterizing a system, an equipment or a component (Lindqvist and Doksum, 2003). It refereed to the ability of a system (or a component) to properly perform the task for which it was designed.

Today, system reliability is usually used as a quantitative concept (Lindqvist and Doksum, 2003). According to the IEC 60870 standard (IEC60870, 1990), the reliability denotes a measure of a system or an equipment to carry out adequately its required function when operating under specified operating conditions for at least a specified period of time. Torres-Toledano and Sucar define the reliability of a system as its probability to perform its intended functions satisfactorily or without failure, for a mission time under the specific design and environmental conditions encountered (Torres-Toledano and Sucar, 1998).

In common, an equipment, a system or a component carries out adequately its assigned functions means that it does not fail during the mission time. In other words, there is no failure or breakdown during the time of service specified to perform its tasks.

The reliability can be expressed in different manners. One of the most common ways to calculate the reliability of a system or a component is to use its associated failure rate.

A **failure rate**, denoted by λ, is defined in (Kuo and Zuo, 2003) to be the frequency with which a system or a component fails. It is expressed in number of failures per unit of time.

The reliability R of a system or a component under a period of time T is obtained from its failure rate λ using this equation:

$$R = e^{-\lambda T} \tag{2.1}$$

Example 2.1 As a simple example, let $C1$ be a component whose associated failure rate λ_{C1} is of 18 failures per 1000 hours. The reliability R_{C1} of this component over 10 hours is obtained by the equation (3.1):
$\lambda_{C1} = 18/1000 \ hrs$
$R_{C1}(10) = e^{-\lambda_{C1} 10} = 83.5\%$.
There is 83.5 % of chance that $C1$ is functioning after 10 working hours. ■

2.2.2 System Reliability Analysis

A system usually consists of two or more components linked together and organized in a certain way. The reliability of each system depends on the individual reliability of its components.

The system reliability analysis aims at using an abstract model of a system to quantify its reliability. Several techniques have been proposed for assessing the reliability of systems. We distinguish the empirical techniques and the analytical techniques (Lindqvist and Doksum, 2003).

The empirical techniques

The main idea of the empirical techniques is to determine the reliability experimentally by using N systems and operating them over a long period of time. The percentage of failed systems during that time is used as an indication of the reliability of the systems (Lindqvist and Doksum, 2003).

The main drawback of these techniques comes from the large number of systems required to achieve the experimentations. In addition, the time required for the experiment may be too long to afford.

The analytical techniques

The analytical techniques aim to quantify the reliability of a system using the reliability of each of its components. The reliability of each system as quantified by these techniques depends on the individual reliability of its components, as well as to the configuration in which the components are organized (Lindqvist and Doksum, 2003).

2.3 Traditional Techniques for Reliability Analysis

In this section, we present some traditional techniques commonly used in the reliability field for modeling the system reliability.

2.3.1 Fault trees (FTs)

Fault Trees, initially proposed in 1961 (Watson, 1961) by Bell Telephone Laboratories, are one of the most important and widely used techniques to conduct the system reliability analysis.

A fault tree (**FT**) allows to graphically model the reliability of a system and to describe the logic of the failures propagation through it (Weber and Simon, 2008). In other words, a FT is a deductive method for the failure analysis aiming at determining successively the causes contributing to a specific undesired event occurring, like a system failure (Rae and Lindsay, 2004). Therefore, a FT represents the boolean relationship between a particular event, called the top event, and the events leading to its occurrence.

Fault trees are logic models consisting of faults and logical gates. Each fault, representing an event, is linked to either zero or one input gate and either zero or one output gate. Each logical gate has one or more input events and one output event. It defines the relationship between its input events and its output event (Rae and Lindsay, 2004).

Figure 2.1 shows two simple fault trees. The presence of the logical '*And*' gate in the first tree (at the right part of the figure) means that to ensure the occurrence of the top event $E3$ which is the output event of the 'And' gate, both events $E1$ and $E2$ must occur. While the presence of the logical '*Or*' gate in the second fault tree (at the left part of the figure) means that the output event $E3$ of this logical gate occurs if at least one of the input events occurs.

The behavior of the output of each logical gate in the fault tree depends on the behaviors of its inputs. The relationship between the input events and the output event of the '*Or*' gate shown in figure 2.1 is given in the truth table 2.1. The value 1 is assigned to an event when it occurs, while the value 0 is assigned to it when it doesn't occur.

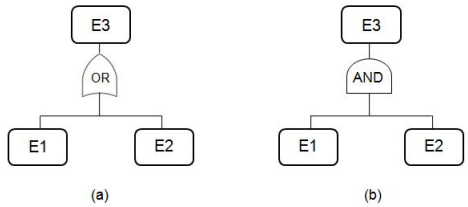

Figure 2.1: Fault trees

Table 2.1: The truth table of the logical '*Or*' gate in a fault tree

C1	**C2**	**Or**
0	0	0
0	1	1
1	0	1
1	1	1

The behavior of the output event of the logical '*And*' gate in the fault tree shown in figure 2.1 depends on the behaviors of the two input events according to the truth table 3.2.

Despite their powerful graphical representation, fault trees present different limitations. In

Table 2.2: The truth table of the logic '*And*' gate in a fault tree

C1	**C2**	**And**
0	0	0
0	1	0
1	0	0
1	1	1

fact, it is so difficult to model large or complex systems by fault trees. Moreover, fault trees are not able to take into account common causes and present difficulties to model dependency between events (Simon et al., 2007).

2.3.2 Reliability Block Diagrams (RBDs)

The reliability block diagram (RBD) (Kleinerman and Weiss, 1954; Villemeur, 1998), sometimes called success paths model, is another well-known practical reliability modeling method for evaluating system reliability. It is used to represent the logic relationship between the functioning of a system and the functioning of its constituant components (Kuo and Zuo, 2003). This model allows to estimate the system reliability given the reliability of each individual component.

The reliability block diagram is a logical model which provides a graphical representation of the system's structure in which every block or rectangle corresponds to a component connected functionally to other components in series or parallel form (Torres-Toledano and Sucar, 1998).

As shown in figure 2.2, a RBD consists of an input node (in the left side of the diagram), an output node (in the right side of the diagram) and at least one path describing successful system operation. Each path in the reliability block diagram represents a sequence of blocks joined from the input node to the output node. It expresses a possible system success combination. In other words, if all the components of any path operate successfully, then the system is able to perform successfully its functions. For example, to ensure the success of the system whose reliability block diagram is given in figure 2.2, the components C3, C4 and either C1 or C2 must be functional.

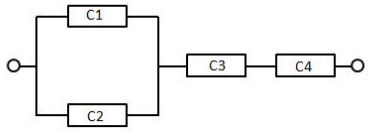

Figure 2.2: A Reliability Block Diagram for a four-Components System

The graphical representation of the reliability block diagram makes it a simple method to describe the physical arrangement of components in a system. Once the RBD corresponding to a system is constructed, the reliability of the whole system can be easily evaluated based on the reliability assigned to each of its components.

Reliability block diagrams have been widely used to model several structures, including series structures, parallel structures, series-parallel structures and parallel-series structures. But not all systems can be modeled by reliability block diagrams (Kuo and Zuo, 2003). Moreover, RBD does not allow to model the sequence of failures in systems contrary to the FT.

Reliability Block Diagrams vs Fault Trees Diagrams

In general, a FT can be easily converted to a RBD. However, it is generally more difficult to convert an RBD into a fault tree, especially for highly complex systems.
The most fundamental difference between a FT and a RBD is that the RBD describes the "success space", and looks at system success combinations, whereas the fault tree describes the "failure space" and looks at system failure combinations.

2.3.3 Markov Chains

Markov chains (Dugan et al., 1992; Villemeur, 1987), called also state space methods, are commonly used in system reliability modeling. A markov chain is defined by a set of states and a set of transitions emanating from a given state over the possible next states.

A markov chain presents the different system's states and the possible transitions between these states. Each of these states is connected to other states by transition probabilities.

Figure 2.3 depicts a Markov chain given in (Simon et al., 2008) for the valve system previously described in section 1.4.2. The passage from one state to another is defined by a transition probability λ_i.

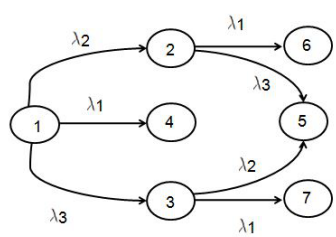

Figure 2.3: Markov Chain of the valve system

In spite of their capacity to supply a good estimation for the system reliability, Markov chains present a major drawback which comes from the combinatory explosion of the states induced when modeling large systems. This combinatory explosion makes modeling large systems by Markov chains very difficult (Simon et al., 2007). In fact, the Markov chain generates all the possible system's states. Therefore, the size of a Markov chain grows exponentially with the number of components in a system. Generally, the number of states is relatively large for large or complex systems. As a result, a Markov model can be infeasible. For instance, for a system with 6 components, if each component has only two states, then the total number of states generated is 2^6 =64.

2.4 Reliability Analysis and Uncertainty

In the reliability analysis, the system reliability depends on its components reliabilities. The information on the reliability of each component is oftentimes collected from experts. When giving these information, experts can not be always precise and sure. Therefore, the assessments of the components reliabilities are usually characterized by uncertainty (Kozine and Utkin, 2002).

The problem of uncertainty comes also from the ignorance, the inconsistency and the incompleteness of available data (Simon et al., 2007). In fact, it is sometimes necessary to inspect the behavior of the components when the system is operating. However, the complexity of real systems, especially developed for performing critical applications, makes getting the exact state of each part in the system at any moment a very difficult task.

The uncertainty incorporated in systems presents several forms. It can be either stochastic or epistemic. Several developments of interesting tools based on the probabilistic formalism have been developed to deal with this uncertainty, including Bayesian networks. The probability theory solves partially this problematic because it provides different tools to handle efficiently the stochastic uncertainty. But it does not treat suitably the epistemic uncertainty.

The epistemic uncertainty has been well taken into account in the literature by various mathematical theories such as the possibility theory and the evidence theory (Shafer, 1976).

The evidence theory, also called Dempster-Shafer (DS)'s belief functions theory, offers flexible and powerful tools to model uncertain knowledge in the epistemic meaning incorporated in complex systems.

To deal with the complexity of large systems, graphical models are an important asset which can be used in this context. They allow representing efficiently the relations among the system's parts and make the computational tasks manageable.

Evidential graphical models relying on both the evidence theory and the graph theory have found a major interest in the reliability field. They allow modeling complex systems and representing epistemic uncertainty in the knowledge related to them.

2.5 Typical Systems in the Reliability Field

In this section, we present some well-known systems commonly used in the reliability analysis.

Throughout this master report, we assume that each system S and each component Ci in S has two disjoint states ($\{Up\}$,$\{Down\}$): Up is the state of a component (or a system)

when it is working and *Down* is its state when it has failed. Thus the frame of discernment of each component and each system is $\Omega = \{Up, Down\}$ and its corresponding power set is $2^{\Omega} = \{\emptyset, \{Up\}, \{Down\}, \{Up \cup Down\}\}$.

This power set 2^{Ω} offers the possibility to assign a quantity to the modality $\{Up \cup Down\}$ whose role is to characterize our ignorance on the real state of the component or the system (Simon et al., 2008). It means that the component (or the system) may only be exclusively in one of the two possible states $\{Up\}$ or $\{Down\}$ without knowing exactly which. This ignorance characterizes the **epistemic uncertainty** and $\{Up \cup Down\}$ is the **epistemic state**.

We suppse that all systems are homogeneous and no repair is considered. The elementary events on all system components are also supposed to be independent.

2.5.1 Series Systems

The series configuration is one of the most common structures. In this configuration, to ensure the success of a series system consisting of n components, all n components must operate adequately. Thus, if any one of the n components is not operating then the system would not be able to perform its function.

The reliability block diagram, given in figure 2.4, shows a series system consisting of 2 components. Both components C1 and C2 must be operating to ensure system operation.

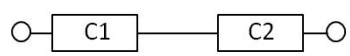

Figure 2.4: Reliability Block Diagram of a two-Components Series System

The reliability $R(t)$ of a series system, consisting of n components, is expressed as the product of the n reliabilities specified for its components.
It is evaluated by the following equation:

$$R(t) = \prod_{i=1}^{n} R_i(t) \tag{2.2}$$

where $R_i(t)$ is the probability of component Ci to be in the state $\{Up\}$ and t is the time of mission.

The reliability $R_S(t)$ of the two-components series system S shown in figure 2.4 conveys its probability to be in the state $\{Up\}$ at time t. This reliability could be evaluated applying the

following equation:

$$R_S(t) = R_1(t) * R_2(t) \tag{2.3}$$

where $R_i(t)$ is the probability of component Ci to be in the state $\{Up\}$ and t is the time of mission.

The directed acyclic graph modeling the two-components series system is shown in figure 2.5. Each node Ci represents the i-th component in the system and node And represents the logical '*And*' gate. The system states are assigned to the result node R. In the DAG modeling this system, the conditional distribution of the node And is defined equivalent to the logical '*And*' gate.

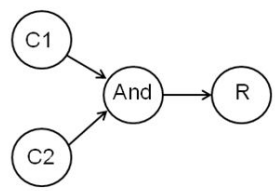

Figure 2.5: A Directed Acyclic Graph for the two-Components Series System

2.5.2 Parallel Systems

The parallel configuration is another well-known structure. In this configuration, to ensure the success of a parallel system consisting of n components, one component at least must be operating. Thus, a failure of all the n components leads to the system failure.

A reliability block diagram for a parallel system with two components is shown in figure 2.6. Either C1 or C2 must be operating to ensure the system operation. If both of them fail, then the system would not be able to carry out its assigned function.

The reliability $R(t)$ of a parallel system with n components at time t is evaluated by the following equation:

$$R(t) = 1 - \prod_{i=1}^{n}(1 - R_i(t)) \tag{2.4}$$

where $R_i(t)$ is the probability of component Ci to be in the state $\{Up\}$ and t is the time of mission.

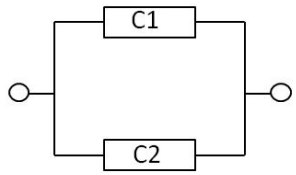

Figure 2.6: Reliability Block Diagram of a Two-Components Parallel System

Thus, the reliability $R_S(t)$ at time t of the two-components parallel system S shown in figure 2.6 is obtained as follows:

$$R_S(t) = 1 - [(1 - R_1(t)) * (1 - R_2(t))] \tag{2.5}$$

where $R_1(t)$ is the the reliability of component C1 at time t, $R_2(t)$ is the the reliability of component C2 at time t and t is the time of mission.

Figure 2.7 shows the directed acyclic graph modeling the two-components parallel system, in which each node Ci represents the i-th component in the system and node Or represents the logical 'Or' gate. The system states are assigned to the result node R. In the DAG modeling this system, the conditional distribution of the node Or is defined equivalent to the logical 'Or' gate.

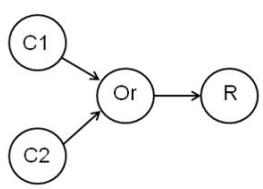

Figure 2.7: A Directed Acyclic Graph for the two-Components Parallel System

2.5.3 Parallel-Series Systems

A parallel-series system is composed of k subsystems connected in a series form, where each subsystem i consists of n_i components connected in a parallel form (Kuo and Zuo, 2003).

A reliability block diagram for a parallel-series system is shown in figure 2.8. This system includes 3 subsystems linked together in a series form, where the first subsystem consists of 2 components organised in a parallel form, the second subsystem consists of one component and

the third subsystem consists of 3 components organised in a parallel form. The RBD of this system shows 6 paths which are C1C3C4, C1C3C5, C1C3C6, C2C3C4, C2C3C5 and C2C3C6. To make the system able to perform its function, one path at least must be such as all the components belonging to it are working adequately.

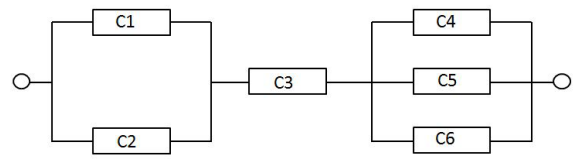

Figure 2.8: Reliability Block Diagram of a Parallel-Series System

The reliability of a parallel-series system with k subsystems is expressed as the product of the k reliabilities of its k subsystems, where the reliability of each subsystem can be evaluated using the reliabilities of its components via equation (3.4).

The reliability $R_S(t)$ of the parallel-series system S whose RBD is given in figure 2.8 could be evaluated applying the following equation:

$$R_S(t) = (R_1(t) + R_2(t) - R_1(t) * R_2(t)) * R_3(t) * [1 - (1 - R_4(t))(1 - R_5(t))(1 - R_6(t))] \quad (2.6)$$

where $R_i(t)$ is the probability of component Ci to be in the state $\{Up\}$ and t is the time of mission.

Figure 2.9 shows two different DAGs modeling the parallel-series system with six components shown in figure 2.8. In each DAG, Ci represents the i-th component, Aj represents the j-th '*And*' gate, and Oz represents the z-th '*Or*' gate. All the conditional distributions of the nodes Aj and Oz are defined equivalent to the logical '*And*' and '*Or*' gates. The first graph is used for computing the reliability of the system, while the second is used for computing its unreliability. These two DAGs supply the same estimation of reliability.

2.5.4 Complex Systems

In addition to systems which are combinations of series and parallel configurations, there are systems in which combinations of components are neither. These systems are called complex systems or non parallel-series systems (Villemeur, 1987).

Figure 2.10 gives the reliability block diagram for a complex system in the reliability meaning as written by Villemeur (Villemeur, 1992): the bridge system which is a well known system that

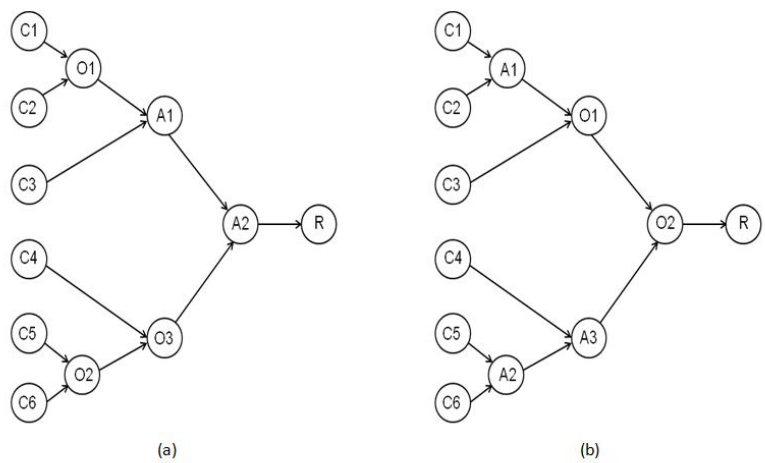

Figure 2.9: Directed Acyclic Graphs for a Parallel-Series System with six Components

has been deeply studied in the literature by several authors. To evaluate the bridge system reliability, Torres-Toledano (Torres-Toledano and Sucar, 1998) modeled it with Bayesian networks.

The bridge system consists of 5 components. The RBD modeling it in figure 2.10 has 4 success paths which are C1C3, C2C4, C1C5C4 and C2C5C3. If all the components belonging to any path of them are operating adequately then the system would operate.

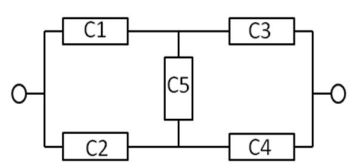

Figure 2.10: Reliability Block Diagram for the Bridge System

The bridge system reliability $R_S(t)$ at time t is given by the following expression:

$$\begin{aligned} R_S(t) &= (1 - R_5(t))[1 - (1 - R_1(t)R_3(t)) * (1 - R_2(t)R_4(t))] \\ &+ R_5(t)[1 - (1 - R_1(t))(1 - R_2(t))][1 - (1 - R_3(t))(1 - R_4(t))] \end{aligned} \tag{2.7}$$

where $R_i(t)$ is the probability of component Ci to be in the state $\{Up\}$ and t is the time of mission.

The bridge system is modeled by the directed acyclic graph shown in figure 2.11, where Ci represents the i-th component, Aj represents the j-th '*And*' gate, and Oz represents the z-th

'*Or*' gate. The system states are assigned to the result node R.

In the bridge system, all the conditional distributions of the nodes Aj and Oz are defined equivalent to the logical '*And*' and '*Or*' gates.

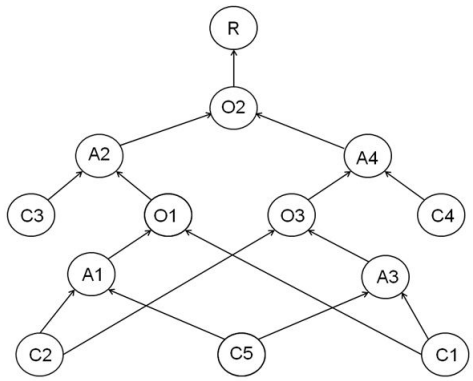

Figure 2.11: Directed Acyclic Graph for the Bridge System

2.5.5 k-out-of-n Systems

The k-out-of-n configuration, denoted by *koon*, is a special configuration which is widely used in real applications. To ensure the ability of a k-out-of-n system to perform its function, k components at least out of the n components are required to operate adequately.

The series configuration, described previously in section 2.5.1, is a particular case of the k-out-of-n configuration when k=n. The parallel configuration, described above in section 2.5.2, represents also a special case of this configuration when k=1.

Figure 2.12 shows a reliability block diagram for a $2oo3$ system. For this k-out-of-n system, $k = 2$ and $n = 3$. Thus, two components at least must be operating to make the system able to perform its function. A reliability block diagram for a $1oo4$ system is given in figure 2.13. For the 1-out-of-4 system, one component at least must be operating to make the system able to perform its function.

The reliability $R_S(t)$ of a k-out-of-n system whose components have the same reliability is given by the following equation:

$$R_S(t) = \sum_{i=k}^{n} C_n^i R(t)^i (1 - R(t))^{n-i} \text{ where } C_n^i = \frac{n!}{i!(n-i)!} \tag{2.8}$$

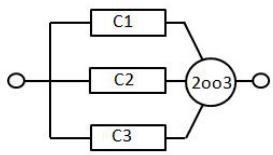

Figure 2.12: Reliability Block Diagram for a 2oo3 system

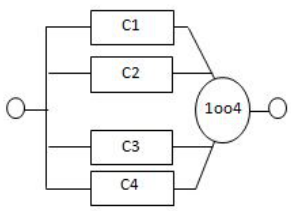

Figure 2.13: Reliability Block Diagram for a 1oo4 system

where $R(t)$ is the probability of each component Ci to be in the state $\{Up\}$ and t is the time of mission.

The reliability $R_S(t)$ of the $2oo3$ system is given theoretically by the following equation:

$$R_S(t) = [R_1(t)R_2(t) + R_1(t)R_3(t) + R_2(t)R_3(t)] - [2 * R_1(t)R_2(t)R_3(t)] \tag{2.9}$$

where $R_i(t)$ is the probability of each component Ci to be in the state $\{Up\}$ and t is the time of mission.

The reliability $R_S(t)$ of the $1oo4$ system is given theoretically by the following equation:

$$R_S(t) = 1 - [(1 - R_1(t)) * (1 - R_2(t)) * (1 - R_3(t)) * (1 - R_4(t))] \tag{2.10}$$

where $R_i(t)$ is the probability of each component Ci to be in the state $\{Up\}$ and t is the time of mission.

Figure 2.14 gives two directed acyclic graphs modeling the $2oo3$ system. In the two DAGs corresponding to this system, Ci represents the i-th component. The node $2oo3$ in the first DAG (at the left part of the figure) represents the logical '$2oo3$' gate. Its conditional belief distribution is defined equivalent to this gate. In the second DAG (at the right part of the figure), each node Aj models the j-th 'And' gate and each node Oz represents the z-th 'Or'

gate. All the conditional distributions of nodes Aj and Oz are defined equivalent to the logical '*And*' and '*Or*' gates. The system states are assigned to the result node R.
As seen previously, to ensure the operation of a 2*oo*3 system, two components must be operating. This means that if both (C1 and C2) or both (C1 and C3) or both (C2 and C3) operate adequately, then the system also operate.

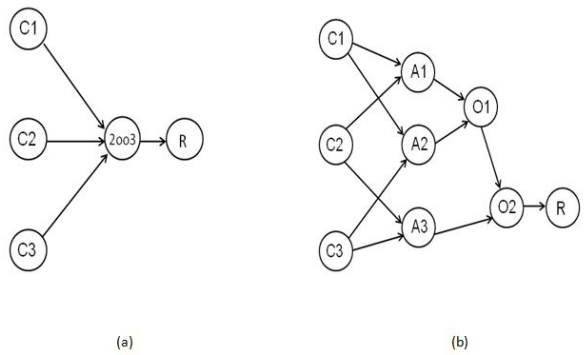

Figure 2.14: Two Directed Acyclic Graphs for the 2*oo*3 System

Figure 2.15 shows a DAG for the 1*oo*4 system, where Ci represents the i-th component and 1oo4 represents the '1*oo*4' logical gate. The system states are assigned to the result node R.

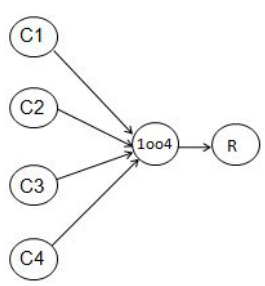

Figure 2.15: Directed Acyclic Graph for the 1*oo*4 System

2.5.6 Linear Consecutive-k-out-of-n Systems

Linear consecutive-k-out-of-n systems, denoted by Lin/Con/koon, are characterized by physical connections among components in lines (Kuo and Zuo, 2003). A Lin/Con/koon consists of n components linearly connected in such a way that the system works if and only if at least k

components out of the n consecutive components work adequately.

The parallel configuration, described previously in section 2.5.2, is a particular case of the Lin/Con/koon configuration when k=1. The series configuration, presented above in section 2.5.1, represents also a special case of this configuration when $k = n$.

Figure 2.16 illustrates a DAG for a Lin/Con/1oo4 system. This model has been proposed for representing for example oil pipeline systems in which successful system operation is ensured if and only if at least 1 pump station out of 4 consecutive pump stations works adequately (Kuo and Zuo, 2003). As shown in figure 2.16, the pipeline system designed for transporting oil from one point to another point consists of 10 pump stations denoted by Ci. The pump stations are equally spaced between the two points. Each pump station is able to transport oil to a distance including 4 other pump stations. If one pump station out of 4 consecutive pump stations is down, the flow of oil is not interrupted, because the following 4-1=3 stations can still perform the load. But, if at least 4 consecutive pump stations fail, the oil flow is interrupted and the system fails.

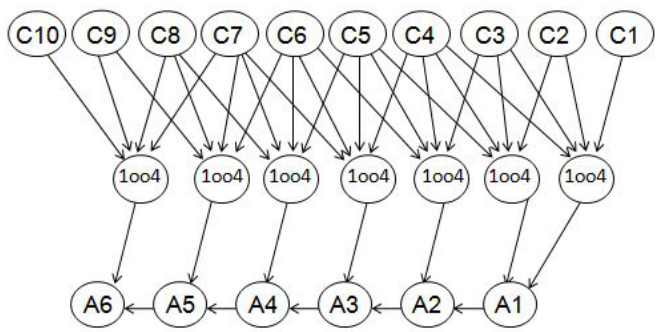

Figure 2.16: A Directed Acyclic Graph for a Linear Consecutive-1-out-of-4 System

2.5.7 Circular Consecutive-k-out-of-n Systems

Circular consecutive-k-out-of-n systems, denoted by Cir/Con/koon, are characterized by physical connections among components in circles (Kuo and Zuo, 2003). A Cir/Con/koon consists of n components circularly connected in such a way that the system works if and only if at least k components out of the n consecutive components work adequately. The parallel configuration, described in section 2.5.2, is a special case of the Cir/Con/koon configuration when k=1. The series configuration, presented in section 2.5.1, represents also a particular case of this configuration when k=n.

Figure 2.17 illustrates a DAG for a Cir/Con/1oo3 system. This model represents for example a photographing system whose description is in (Kuo and Zuo, 2003). This system is used to take pictures of the acceleration activities occurring in a nuclear accelerator. Because of the high speed of the activities and the high cost involved in implementing such an experiment, the photographing system must be reliable and accurate. As shown in figure 2.17, the system consists of a set of 10 camera installed around the accelerator. The photographing system works properly if and only if at least 1 camera out of 3 consecutive cameras works properly.

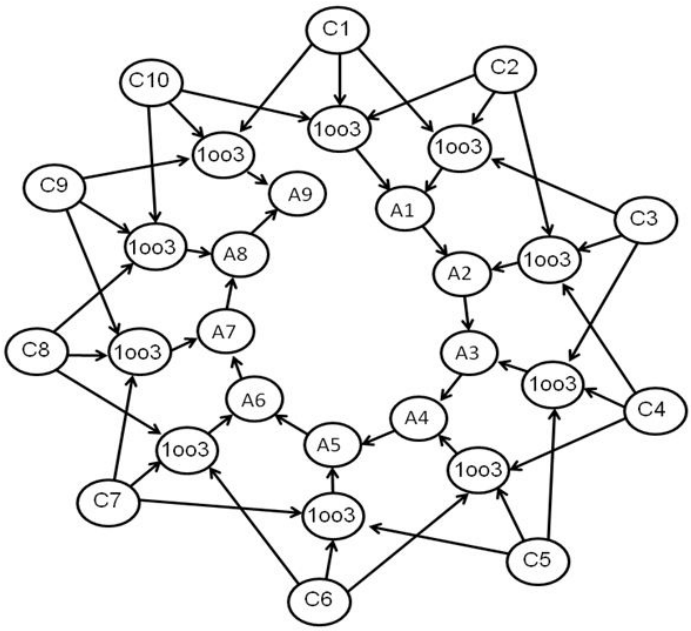

Figure 2.17: A Directed Acyclic Graph for a Circular Consecutive-1-out-of-3 System

2.6 Conclusion

In this chapter, we have defined some concepts related to the system reliability. Then, we have sketched some traditional techniques used to conduct the system reliability analysis and presented their major drawbacks. Next, we have focused on the problem of uncertainty in the reliability field which has made the evidential graphical models a tool of a great interest and has led to their application in this field. In the second part of this chapter, we have described some well-known systems in the reliability field and presented the directed graphical models modeling them.

3

Comparing Evidential Models for Imprecise Reliability

3.1 Introduction

Evidential graphical models relying on both the evidence theory and the graph theory are of a major interest in the reliability field. They allow modeling complex systems and representing both stochastic and epistemic uncertainties in the knowledge related to them.

In this chapter, we apply two different evidential networks to the study of the reliability of several well-known systems that we have presented in chapter 2 in order to show the capacity of each one to model uncertainty in knowledge and to manage it. Then we present a comparison of these two evidential models applied to the study of systems reliability under uncertainty. The first one which is proposed by Simon and Weber (Simon et al., 2008) is based on the junction tree inference algorithm, using an extension of the Bayes' theorem to the representation of DS's belief functions theory. The second one which is proposed by Ben Yaghlane (Ben Yaghlane and Mellouli, 2008) is based on the modified binary join tree (MBJT) algorithm, using the disjunctive rule of combination (DRC) and the generalized Bayesian theorem (GBT), both proposed by Smets.

The chapter is structured as follows. In section 4.2, we first present some implementation aspects and choices in order to apply the two evidential networks to the reliability analysis. In section 4.3, we describe different aspects allowing to compare the evidential network and the directed evidential network with conditional beliefs functions on the basis of different criterion specified progressively.

3.2 Implementation Aspects

In this section, we present the choices relative to the implementation of the EN and the DEVN and their application in the reliability field.

3.2.1 Implementation Aspects in the EN

The evidential network (Simon et al., 2008) combines the Bayesian network with the evidence theory. The inference process is done in a junction tree using an extension of the Bayes' theorem to the representation of DS's belief functions theory.
To implement the evidential network and to perform the inference process, we have used the Bayes Net Toolbox (BNT) for matlab.

After applying the EN to the different systems described in chapter 3, we obtain the corresponding junctions trees. Figure 3.1 shows a junction tree for the two-components parallel system, and figure 3.2 shows a junction tree for the two-components series system.

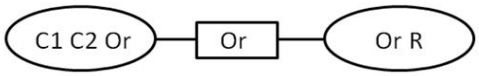

Figure 3.1: Junction Tree for the two-components Parallel System

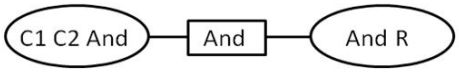

Figure 3.2: Junction Tree for the two-components Series System

The junction tree for the 2oo3 system and the 1oo4 system are depicted in figures 3.3 and 3.4.

Figure 3.5 shows the obtained junction tree for the parallel-series system and figure 3.6 shows the obtained junction tree for the bridge system.

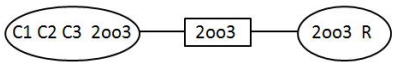

Figure 3.3: Junction Tree for the 2oo3 System

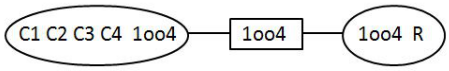

Figure 3.4: Junction Tree for the 1oo4 System

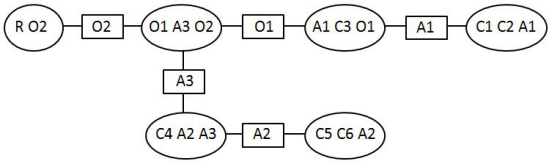

Figure 3.5: Junction Tree for the Parallel-Series System

Figure 3.7 gives the junction tree corresponding to a circular consecutive-1-out-of-3 system. Each node Ki correponds the the $i-th$ gate '1oo3'.

3.2.2 Implementation Aspects in the DEVN

The computational effort required for the inference process in the DEVN depends on the structure of the obtained modified binary join tree. It increases exponentially with the maximum node size. For the same problem, different elimination sequences lead to several graphical structures having different maximum node sizes. Thus, finding the best elimination sequence is of a great interest for constructing a modified binary join tree having the minimum node size which reduces the amount of computations and makes fast the inference process.

When finding the best deletion sequence, we should pay a real attention to another aspect. In fact, two different elimination sequences can lead to two different MBJTs which yield differ-

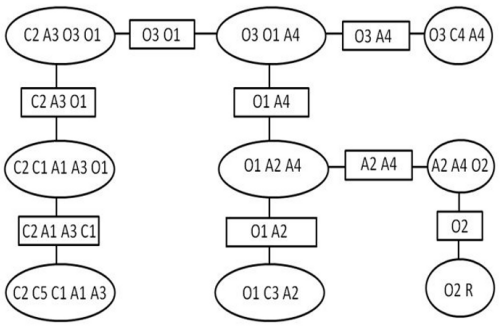

Figure 3.6: Junction Tree for the Bridge System

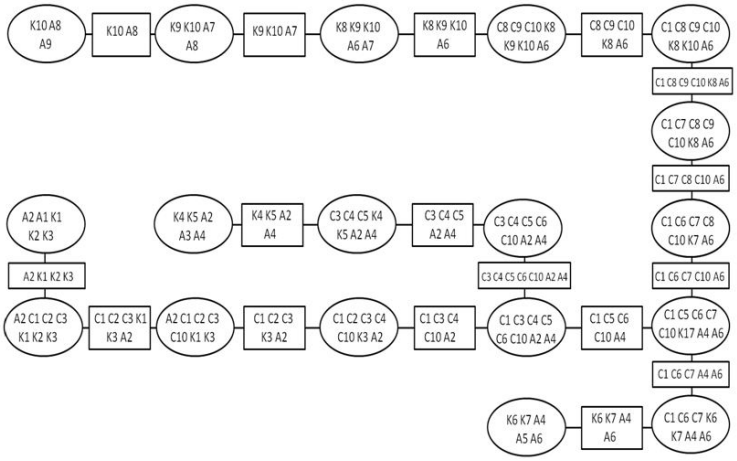

Figure 3.7: A Junction Tree for a Circular Consecutive-1-out-of-3 System

ent marginals for the same variable after performing the inference process. Since our objective is to propagate beliefs through the obtained MBJT in order to evaluate the system reliability, the assessment of this reliability obtained by the graphical structure must be compared with the exact value of the system reliability obtained by the mathematical formulation corresponding to the reliability block diagram of the system. This comparison is very important to ensure that the obtained results are exact or close to the exact value.

To select the optimal elimination sequence which takes into account the two aspects presented previously, it is necessary to generate all the possible deletion sequences and to test them. The number of the possible deletion sequences depends on the number of variables in the problem. If a problem is described by n random variables, then we would obtain $n!$ elimination sequences. Therefore, this approach is possible when dealing with a problem with a relatively small number of variables. But if we have a large or a complex problem like the bridge system which has 13 random variables, it will be difficult or nearly impossible to generate and to perform experimentations with 13! deletion sequences.

Since DAGs corresponding to the two-components parallel system shown in figure 2.7, the two-components series system shown in figure 2.5, the $2oo3$ system shown in figure 2.14(a), and the $1oo4$ system in figure 2.15 have a relatively small number of variables, we generated easily for each problem all the possible deletion sequences, then we built the MBJT corresponding to each deletion sequence, and next we performed the inference process through each MBJT in order to select the best elimination sequence that allows at once to reduce the computational complexity and to yield a result close to the exact value of reliability given by the mathematical formulation.

After testing and experimenting all the generated elimination sequences, we selected for each system the best elimination sequence that gives the modified binary join tree respecting both constraints.

Figure 3.8 depicts the modified binary join trees for the two-components parallel system (a) and the two-components series system (b). Each MBJT is constructed using the best elimination sequence selected from all the generated elimination sequences.

Figure 3.9 gives the modified binary join trees for the $2oo3$ system (a) and the $1oo4$ system (b).

The best deletion sequences selected for constructing the MBJTs corresponding to these four systems shown in figures 3.8 and 3.9 will be very useful and helpful when constructing a MBJT for a system with a large number of variables.

Since it is nearly impossible to perform experimentations with all the possible deletion sequences generated for problems involving a large number of variables, a logical process is

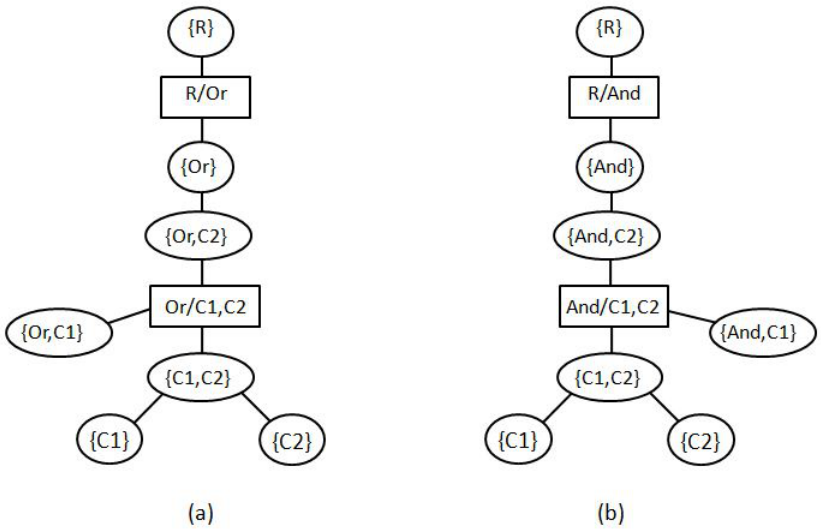

Figure 3.8: Modified Binary Join Trees for the parallel system and the series system

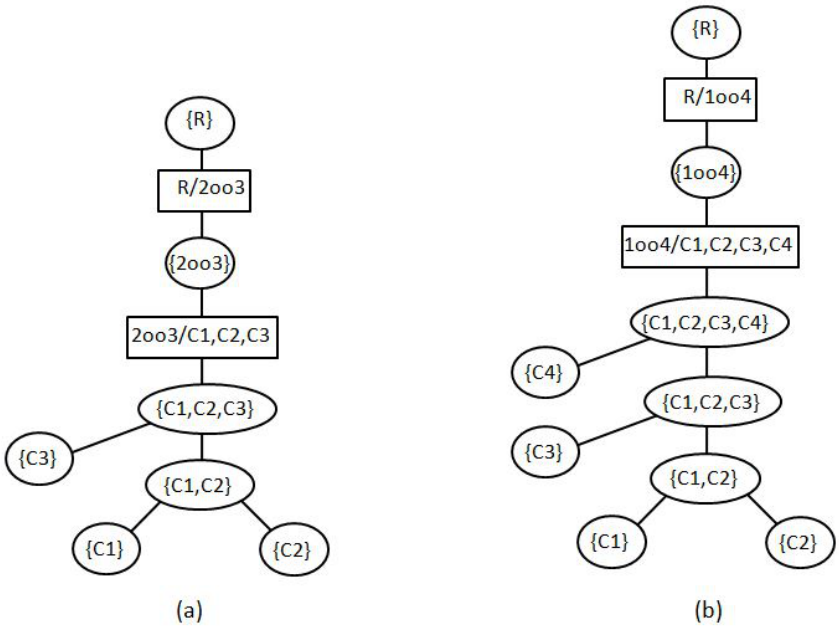

Figure 3.9: A modified Binary join tree for the '2*oo*3' system and the '1*oo*4' system

performed in order to obtain the modified binary join trees for those systems based on the use of these MBJTs.

The main idea of this process is to divide a complex problem in smaller problems and to deal separately with each individual problem in order to solve the whole problem. In other words, the process used to construct a modified binary join tree for systems with large number of variables is based on the assumption that a system consists of many subsystems. To obtain a MBJT for the whole system which is divided in several subsystems, we successively construct a partial MBJT for each subsystem and we arrange them in the final MBJT representing the system. For example, the DAG of the bridge system given in figure 2.11 shows that this system can be divided in 7 subsystems which are either parallel or series. Each gate in the DAG forms with its inputs a subsystem in the whole system. A MBJT corresponding to each subsystem will have the same structure as one of the four MBJTs shown in figures 3.8 and 3.9. If for example a subsystem consists of the logical '*And*' gate and its inputs, then the MBJT built for this subsystem will have the same structure as the MBJT depicted in the right part of figure 3.8. Therefore, the number of logical gates in the DAG determines the number of subsystems to be considered, and then the number of partial MBJTs to build.

The MBJT for the whole system is constructed step-by-step by building for each current gate in the DAG, denoted by G_c, its corresponding partial MBJT, denoted by M_{G_c}, according to this process that we have implemented and tested:

- if the inputs of G_c are two components Ci (represented by two root nodes in the DAG), then M_{G_c} representing the subsystem including the two components and G_c, can be constructed using the corresponding best elimination sequence. For example the inputs of the gate $'A1'$ in the DAG of the bridge system are the two components $C1$ and $C5$. The MBJT built for this subsystem is obtained using the same order of variables in the deletion sequence used to build the MBJT of the series system shown in figure 3.8 (b).
- if one of the two inputs of G_c is a gate, denoted by $G1$, then M_{G_c} can be built if and only if the MBJT corresponding to $G1$, denoted by M_{G1} has been built. If M_{G1} has been built, then the same node representing $G1$ in this MBJT will be used to construct the partial MBJT M_{G_c}. In other words, it is necessary to construct M_{G1} before constructing M_{G_c} if the gate $G1$ is an input of the gate G_c.
- if both inputs of G_c are gates, denoted by $G1$ and $G2$, then M_{G_c} can be built if and only if the MBJTs corresponding to $G1$ and $G2$, denoted by M_{G1} and M_{G2}, have been built. The two nodes corresponding to $G1$ and $G2$ which have been created when building M_{G1} and M_{G2} will be used to construct M_{G_c}.

This process, used when building a MBJT for a large system, ensures that the obtained structure will be connex since each partial MBJT built at step i includes at least one node

belonging to the partial MBJT created at step $j < i$. The obtained MBJT for the whole system will give after performing the inference process a result which is close to the exact value of reliability given by the mathematical formulation. This is because the deletion sequences used to construct successively the partial MBJTs were tested and selected before. For example, if we restrict the propagation of beliefs only in the MBJT corresponding to the first subsystem, we are certain that we will find the good value of reliability of this subsystem. If we continue the inference process, this result will be propagated and we will obtain also a result close to the exact value of the system reliability.

After performing the process described previously for constructing a MBJT for large systems, we noticed that a problem arises for some examples. In fact, for some systems, a component may be the input of more than one gate at the same time. For instance, component $C5$ in the DAG of the bridge system shown in figure 2.11 is at once the input of the gate $'A1'$ and the gate $'A3'$. If we perform the process for bulding the MBJT for those systems, we will have a multiply connected structure. For example for the bridge system, the obtained structure is a multiply connected structure because of using the same node representing the variable $C5$ in the two partial MBJTs corresponding to the gate $'A1'$ and $'A3'$.

Figure 3.10 shows the partial multiply connected structure obtained when performing the process for building the MBJT for the bridge system. To solve this problem, we propose the

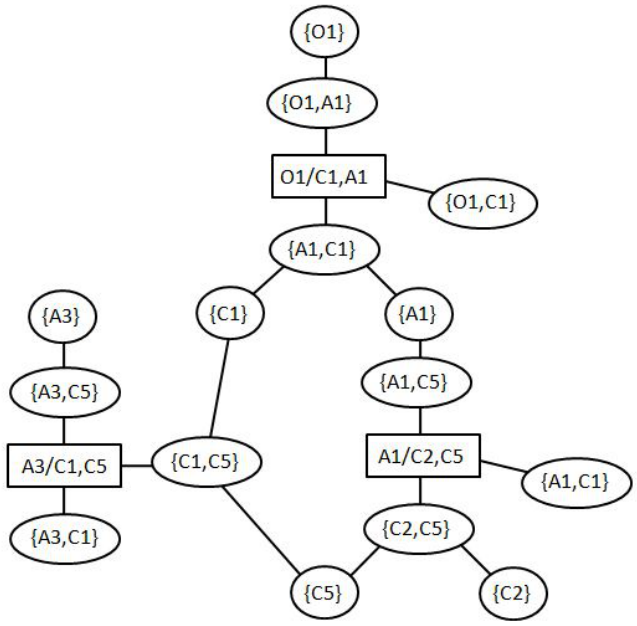

Figure 3.10: A Partial Multiply Connected Structure

following heuristic: *if a component is an input for more than one gate, then this component will be represented by more than one node in the obtained MBJT.*

For example, since component $C5$ in the bridge system is the input of the gates $'A1'$ and $'A3'$, then two nodes representing $C5$ will be inserted in the MBJT, once when building the partial MBJT for the susbsystem $(C2, C5, A1)$ and once when building the partial MBJT for the susbsystem $(C1, C5, A3)$. That is why, we see in the MBJT for the bridge system given in figure 3.11, duplicated nodes of $C1$, $C2$ and $C5$, respectively.

When initializing the obtained MBJT for the whole system, the different nodes representing the same component will be quantified by the same initial potential corresponding to this component.

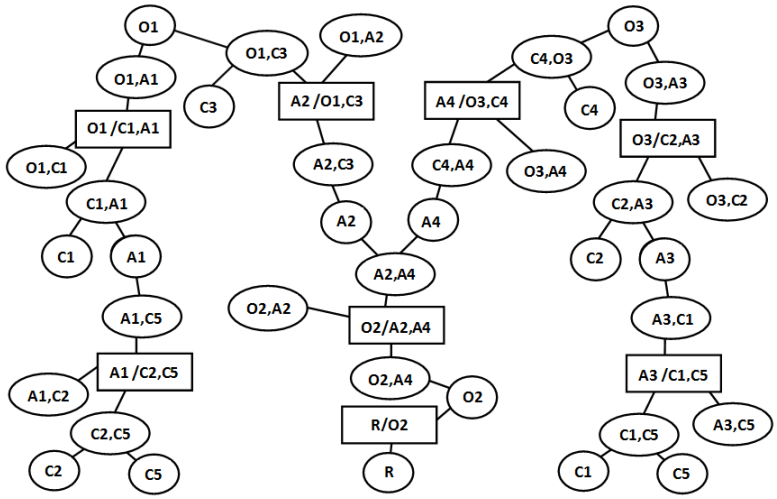

Figure 3.11: The Modified Binary Join Tree for the Bridge System

3.3 Comparing Aspects

In this section, we present a comparative analysis between the EN and the DEVN. This comparison is based on different aspects. In particular, the original structure, the graphical structures for the inference, the message passing schemes, the storage efficiencies, the computational efficiencies and the exactness of the results are studied.

3.3.1 Original Structure

Evidential network (EN) and directed evidential network (DEVN) are graphically two directed acyclic graphs representing the uncertainty in the knowledge using the framework of the belief

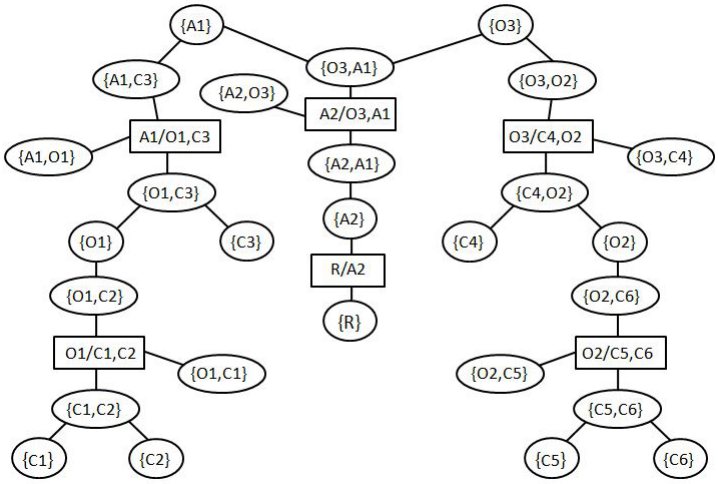

Figure 3.12: The Modified Binary Join Tree for the Parallel-Series System

functions theory.

In the two networks modeling the system reliability, root nodes represent the system components and child nodes represent the logical gates which can be '*And*' gates, '*Or*' gates, '2*oo*3' gates or '1*oo*4' gates. The edges describe the dependencies relations between the logical gates and their inputs. Thus, for a system reliability modelling by means of an evidential network or a directed evidential network, the graphical representation (*i.e.* the DAG) is the same.

In the first network, each child node X represents a conditional relation between the variable X and its parents $Pa(X)$. This conditional relation is defined by its own conditional belief mass table $m[Pa(X)](X)$. That means that a conditional belief mass table defines the relation between the belief masses on the frame of discernment of the child node and the belief masses on the frame of discernment of the variables in the parent nodes.

In the second network, each edge represents a conditional relation between the two nodes it connects. Thus, conditional beliefs are defined in the two networks in different manners.

The number of conditional relations in the evidential network depends on the number of child nodes. But, it depends on the number of edges in the DEVN. Nevertheless, the conditional belief function assigned to a child node given all its parents in the directed evidential network, can be obtained by the means of the formula proposed by Xu and Smets (Xu and Smets, 1994), using the initial conditional belief functions (*i.e.* conditional belief functions between only two nodes or variables).

For instance, let us consider the two spaces Ω_{C2} and Ω_{C5} associated to the two nodes C2 and C5 in the DAG of the bridge system shown in figure 2.11. We use $m_{And1}[c2](and1)$ to represent the conditional mass function induced on the space Ω_{And1} given $c2 \subseteq \Omega_{C2}$, and $m_{And1}[c5](and1)$ to represent the conditional mass function induced on the space Ω_{And1} given $c5 \subseteq \Omega_{C5}$. The following formula of Xu and Smets allows to obtain $m[C2, C5](And1)$:

$$\forall c_{25} \subseteq \Omega_{C_{25}}, m[c_{25}](a) = \sum_{a1 \cap a2 = a} m_{And1}[C2](a1) m_{And1}[C5](a2) \tag{3.1}$$

where $\Omega_{C_{25}} = \Omega_{C2 \cup C5}$

Thus, unlike the evidential network in which a conditional belief mass table should be specified for each child node given all its parents, in the directed evidential network, we can either define this table for each child node given all its parents or simply specify one table for each child node given separately each one of its parents, and then obtain via the formula the conditional belief function of the node given all the parents. Therefore, the directed evidential network offers more flexibility than the evidential network when defining the parameters of the DAG.

In each system, the state of an '*And* ' gate (or an '*Or* ' gate) is easily defined given the states of its two inputs. For this reason, the truth tables of the logical gates, represented by the child nodes in the DAG, are modeled in the two networks by conditional belief mass tables representing the relations between the gate and its two inputs. The conditional belief mass tables for the child nodes, representing '2*oo*3' gates or '1*oo*4' gates, define also the relation between the gate and all its inputs since it is simple to determine the gate's state given the states of all the inputs.

3.3.2 Graphical Structures for Message Propagation

For the evidential network, propagation of beliefs is done in a junction tree. For the DEVN, propagation of beliefs is done in a modified binary join tree.

The nodes of the junction tree are the cliques of a triangulated moral graph of the original evidential network. The corresponding modified binary join tree includes the subsets forming the hypergraph as well as singleton variables which are not already included in the hypergraph.

Let us suppose that we have a DAG with n random variables. To construct a junction tree for this DAG, each singleton variable forms a clique in the worst case. The number of cliques in the resulting junction tree is n.

To obtain the corresponding MBJT, we start with an hypergraph containing n subsets to which we add p singleton variables not occuring in the hypergraph. Since the MBJT construc-

tion process is motivated by the idea of the fusion algorithm (Shenoy, 1997), k new subsets resulting from the combinations done during the construction process will be included in the MBJT. The number of nodes in the resulting MBJT is $n + p + k$.

Therefore, a MBJT modeling a system reliability has more nodes than a corresponding junction tree. For example, the junction tree for the bridge system shown in figure 3.6 has 9 nodes whereas the corresponding modified binary join tree shown in figure 3.11 has 45 nodes. For the two-components series system, the junction tree depicted in figure 3.2 consists of two nodes, whereas the corresponding modified binary join tree in figure 3.8 (b) has 9 nodes. The junction tree for the $1oo4$ system shown in figure 3.4 has two nodes, while the corresponding MBJT in figure 3.9 (b) has 11 nodes.

As we said before in section 1.4.3, the two obtained data structures, used for inferring the beliefs satisfy the running intersection property (Lauritzen and Spiegelhalter, 1988). This means that if a variable appears in two different nodes, it also appears on all nodes on the path linking them.

The link between two adjacent nodes in the junction tree is materialized by a separator which is labeled by the intersection of the two nodes, while in the modified binary join tree, the link between two adjacent nodes is simply represented by the edge.

The construction of the first structure takes as parameters the cliques of the triangulated moral graph. The obtained model is an undirected graph showing only undirected links between nodes. It doesn't show any (in)dependence relations between the nodes composing the junction tree.

In contrast, the construction of the second structure needs as parameters a directed evidential network quantified by conditional belief functions added to an hypergraph. The resulting structure is an undirected graph showing at once directed and undirected links between nodes. It emphasizes explicitly the independence relations shown in the initial directed evidential network. These (in)dependence relations between nodes in the MBJT are represented by the created conditional nodes which are going to be very useful when inferring beliefs through the MBJT (Ben Yaghlane and Mellouli, 2008).

For instance, the node $2oo3, C1, C2, C3$ in the junction tree for the $2oo3$ system shown in figure 3.3 does not allow to deduce explicitly that this node represents in fact the conditional relationship over the gate $2oo3$ given its inputs (*i.e.* its parents in the DAG) $C1, C2$ and $C3$, denoted by $2oo3|C1, C2, C3$. In the corresponding MBJT given in figure 3.9 (a), the conditional node $2oo3|C1, C2, C3$ emphasizes explicitly the dependency relationship between the variables. It is easy to deduce that the node $2oo3$ has three parent nodes in the DAG which are $C1$, $C2$

and $C3$.

3.3.3 Message-Passing Schemes

Once the graphical structure for message propagation is constructed, it can be initialized by assigning potentials to its nodes.

In the junction tree, the clique potential is set to the joint density of the distributions assigned to it. The potentials for the separator nodes are set to unity. Thus each clique and each separator in the junction tree stores a potentiel.

In the modified binary join tree, the initialization is done by means of joint belief functions for not conditional nodes (*i.e.* joint nodes), and conditional belief functions for conditional nodes (Ben Yaghlane and Mellouli, 2008).

To begin propagating messages in the junction tree, a clique is selected arbitrarily as the root node. Each link in the junction tree is used twice during the message passing, once in the inward phase, in which each clique receives a message from each of its outward neighbors, and once in the outward phase, where each clique receives a message from its inward neighbor.

The inward phase is done by collecting messages from the leaves to the root node. In reverse, distribution of messages is done in the outward phase away from the root towards each leaf in the junction tree.

The inference in the junction tree involves all the created nodes:

In the inward pass, every clique Cl_i (except the root) is allowed to send a message to its inward neighbor Cl_j over the separator S_{ij} between it and its neighbor only when it has received messages from all its outward neighbors. Since the leaves in the junction tree have no outward neighbors, they are ready to send a message right away.

The clique Cl_i computes a message by marginalizing its current potential to the domain of the separator S_{ij}, and then sends the message to it. When receiving the message, the separator S_{ij} has first to divide it by its local potentiel, to send the obtained message to the clique Cl_j, and then to save the received message. After receiving the message from S_{ij}, the clique Cl_j multiplies it into its own potential, and then stores the resulting message as its new potential.

On the outward pass, every clique Cl_j (except the leaves) is allowed to send messages to all its outward neighbors only when it has received a message from its inward neighbor. Since the root has no inward neighbor, it is ready to send messages right away.

The clique Cl_j computes a message for each clique Cl_i of its outward neighbors by marginalizing its current potential to the domain of the separator S_{ij} between it and Cl_i, and then sends the message to S_{ij}. As in the inward pass, before passing the received message to the clique Cl_i, the separator S_{ij} has first to divide it by its local potential, to send the resulting message to Cl_i, and then to save the received one. Thus, computations in the junction tree are done by each clique and by each separator.

After completing the bidirectional message-passing scheme described above, we can compute the marginals for desired singleton variables. The potential at each clique and at each separator corresponds to the marginal of the joint for the variables belonging to that node. To compute the marginal distribution for a desired variable, we need to marginalize the smallest potentiel containing it over the remaining variables.

As in the junction tree, the propagation of messages in the MBJT is done in two stages: the propagation-up and the propagation-down. The root is determined according to the propagation-up scheme, where messages are passing from the leaves over the path to the center of the tree.

Unlike the inference in the junction tree, the inference through the MBJT involves only the joint nodes which are asked to send and to receive messages. Neither conditional nodes, nor separators (edges) are able to do computation or to exchange messages. These conditional nodes are considered as 'bridges' when inferring beliefs between joint nodes (Ben Yaghlane, 2008). They hold an important role by allowing to determine whether the message sent from one joint node to another joint node is a 'parent' message or a 'child' message.

When inferring beliefs in the MBJT, two types of possible messages are exchanged between nodes in the two stages.

In the propagation-up, after receiving the messages from all its outward neighbors, each joint node N_i is asked to send a message to its inward neighbor:

- If the inward neighbor is a joint node, denoted by N_j, the node N_i computes a message by first combining all the received messages with its own potential and then marginalizing the result to the intersection of its domain with its neighbor's domain. This message is similar to the one exchanged between nodes in the binary join tree (Shenoy, 1997). After receiving the message from N_i, N_j stores it without updating its own potential which remain unchanged during the propagation process.

- In contrast, if the inward neighbor of N_i is a conditional node, denoted by N_c, N_i is asked to send a message to each node N_j of the inward joint neighbors of this conditional node N_c. The sent message is computed with conditional belief functions and depends on the

sender and the receiver. The conditional node N_c between the two joint nodes allows to determine whether the sender is the parent or the child of the receiver node: if N_i is the parent and N_j is the child, then the message to be sent is computed by the disjunctive rule of combination (DRC) proposed by Smets (Smets, 1993). However, if N_i is the child and the receiver is the parent, the message sent is computed by the generalized theorem of Bayes (GBT) proposed also by Smets (Smets, 1993).

In the propagation-down, after receiving the messages from its inward neighbor, each joint node N_j is asked to send a message to each node of its outward neighbors:

- If the outward neighbor is a joint node, denoted by N_i, N_j computes a message by combining its own potential with all the received messages except that one it has received from N_i and then by marginalizing the result of the combination to the intersection of its domain with its neighbor's domain.

- If the outward neighbor is a conditional node, denoted by N_c, N_j is asked to send a message to each of the outward joint neighbors of this conditional node N_c. The message sent is computed with conditional belief functions by applying either the DRC or the GBT. It it important to mention that, if the DRC rule is used in the propagation-up for computing a message to be sent from a joint node N_i to an other joint node N_j, then the GBT rule is used in the propagation-down for computing a message to be sent from the node N_j to the node N_i. But if the GBT rule is used in the propagation-up for computing a message to be sent from N_i to a N_j, then the DRC rule is used in the propagation-down for computing a message to be sent from N_j to N_i.

The two phases of inference in the MBJT previously described are different, unlike in the junction tree where the inward phase is similar to the outward phase.
Figure 3.13 shows a simple DEVN (a) with two nodes N_j and N_j and the propagation process performed using the DRC (b) and the GBT (c) in the corresponding MBJT.
Formally, the message to be sent from node N_i to node N_j is computed by the following disjunctive rule of combination (DRC) proposed by Smets (Smets, 1993):

$$pl_1^{N_j}(n_j) = \sum_{n_i \subseteq N_i} m_0^{N_i}(n_i)(1 - \prod_{n_i \in N_i} (1 - pl^{N_j}[n_i](n_j))) \tag{3.2}$$

where $m^{N_j}[n_i] : n_i \in N_i$ is the conditional bba of the node $N_j \mid N_i$, $m_0^{N_i}$ is the bba of the node N_i that must be propagated through node $N_j \mid N_i$ in order to compute $m_1^{N_j}$

The message to be sent from node N_j to node N_i is computed by the following generalized theorem of Bayes (GBT) proposed also by Smets (Smets, 1993):

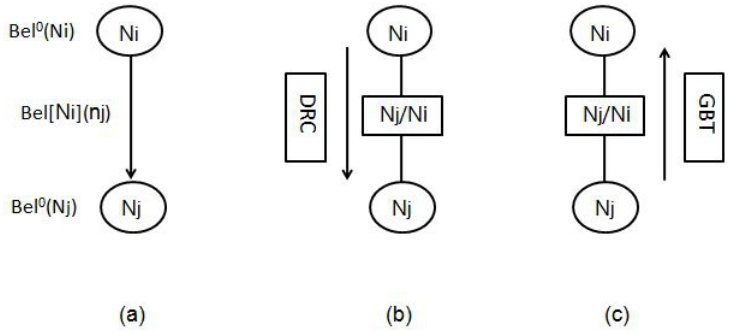

Figure 3.13: The propagation process in the MBJT using the GBT and the DRC

$$pl_1^{N_i}(n_i) = \sum_{n_j \subseteq N_j} m_0^{N_j}(n_j)(1 - \prod_{n_i \in N_i} (1 - pl^{N_j}[n_i](n_j))) \tag{3.3}$$

where $m^{N_j}[n_i] : n_i \in N_i$ is the conditional bba of the node $N_j \mid N_i$, $m_0^{N_j}$ is the bba of the node N_j that must be propagated through node $N_j \mid N_i$ in order to compute $m_1^{N_i}$.

After carrying out message passing in the MBJT, we can compute the marginal of a desired variable in the corresponding singleton node, by combining its own potential with the messages received from all the inward and the outward neighbors.

3.3.4 Storage Efficiencies

The potential storage memory space in the two structures depends on the number of potentials, on the domain on which each potential is specified and on the form in which each potential is stored.

In a junction tree, all the cliques and all the separators have local joint potentials. Since at the end of the message-passing scheme the junction tree yields only marginals for cliques and separators and not marginals for desired singleton variables, there is further storage memory space required to store marginals for those variables (Lepar and Shenoy, 1998). Moreover, the storage requirements for storing the input potentials are added.

In a MBJT, joint potentials are stored in the joint nodes, while conditional potentials are stored in the conditional nodes. Each conditional node stores exactly one potential, but not all the joint nodes in a MBJT store potentials.

If two adjacent joint nodes in a MBJT are separated by an edge, then either one or two potentials are stored in this edge. It depends on the number of messages exchanged between the two nodes in the two phases of propagation. However, if two joint nodes are separated by a conditional node denoted by C, then the edge that connects the first joint node and C and the edge that connects C and the second joint node are regarded as one edge because the conditional node C will neither send nor receive messages. In this case, two messages are at most exchanged between the two joint nodes through the two edges, and so two potentials are at most stored.

As in a junction tree, a storage memory space is included for storing the input potentials in a MBJT. The marginal for each singleton variable is computed by combining the local potential of the corresponding node in the MBJT with all the messages received by it in the two phases. Thus, as in the junction tree, a storage memory space is also required for storing output potentials.

As mentioned in (Ben Yaghlane, 2008), storing a joint potential allocates more memory space than storing a conditional potential. A junction tree is storing only potentials in joint form, whereas a MBJT is storing potentials in joint and conditional forms since it consists of both conditional and joint nodes. This offers an important memory space gain as seen for example in the bridge system, in the $1oo4$ system and in the parallel-series system:

Example 3.1 The bridge system storage.
In the junction tree of the bridge system shown in figure 3.6, the memory space requirement described in units of floating point numbers (fpn) is:

Input storage (5 root nodes, 7 child nodes with two parent nodes and one child node with one parent node): $5 * 4 + 7 * 64 + 1 * 16 = 484\ fpn$
Output storage (13 singleton variables): $13 * 4 = 52\ fpn$
Clique storage (2 with 5 variables, 1 with 4 variables, 5 with 3 variables and 1 with 2 variables): $2 * 1024 + 1 * 256 + 5 * 64 + 1 * 16 = 2640\ fpn$
Separator Storage (1 with 4 variables, 1 with 3 variables, 5 with 2 variables and 1 with 1 variable): $1 * 256 + 1 * 64 + 5 * 16 + 1 * 4 = 404\ fpn$

In the MBJT of the bridge system given in figure 3.11, the memory space requirement described in units of floating point numbers (fpn) is:

Input storage (5 joint singleton nodes with 1 joint potential, 7 conditional nodes with 3 variables and one conditional node with 2 variables): $5 * 4 + 7 * 16 + 1 * 8 = 140\ fpn$
Output storage (13 singleton variables): $13 * 4 = 52\ fpn$
Separator Storage (22 separators with 2 potentials,7 separators with 2 potentials(1 with 1 variable,1 with 2 variables): $(22 * (2 * 4) + 7 * (4 + 16)) = 316\ fpn$

Thus the total storage space needed for the bridge system is of 3580 fpn in the junction tree compared to 508 fpn in the MBJT. The storage memory space needed in the junction tree is larger than the one required in the MBJT. The order of magnitude of the two structures is near to 1/7 for the example of the bridge system. We notice that we have computed the 1oo4 system storage requirements and the parallel-series system storage requirements, and we have obtained the relatively similar conclusion. ■

By using the conditional form for storing some potentials, the MBJT requires less memory space than the junction tree for the brithe system, the $1oo4$ system and the parallel-series system. For instance, all input potentials are included in the two structures but the memory space needed for storing them in the junction tree corresponding to each system is more larger than the memory space required in the corresponding MBJT. This statement regarding the input storage efficiency can be generalized because for the same problem, the joint form is used in a junction tree for storing all the input potentials, while some of them are stored in the conditional form in the corresponding MBJT.

The memory space gain offered when using the conditional form for storing some input potentials in a MBJT is as important as the number of parents of nodes in the DAG. For example as seen in the DAG of the $1oo4$ system given in figure 2.15, the node representing the $1oo4$ gate has 4 parents nodes. To store the corresponding potential, the junction tree requires 1024 fpn comparing to only 64 fpn required in the MBJT. The large difference in the memory space used in the $1oo4$ system when storing this potential is very important since the number of parent nodes of the node $1oo4$ is relatively large.

The storage requirements for storing the marginals of the singleton variables (output potentials) is always the same in the two structures.

Furthermore, the junction tree requires a memory space for the clique storage and the separator storage, while the MBJT requires only a memory space for separator storage. That is because the potentials stored in the MBJT's nodes correspond to the input potentials which remain unchanged during the inference process and the exchanged messages are stored in the MBJT's separators (edge). Therefore we need to compare the aggregate of the clique storage and the separator storage in a junction tree to the separator storage in a MBJT. For each of the bridge system, the $1oo4$ system and the parallel-series system, the MBJT has less separator storage than the aggregate of separator storage and clique storage in the corresponding junction tree. Although the MBJTs of these three systems consist of more nodes and then of more separators than the corresponding junction trees, the domains on which are defined the potentials associated with separators in MBJTs are smaller than the domains on which are defined the potentials assigned to cliques and separators in the corresponding junction trees. Indeed, in a MBJT, the domain of the potential of the conditional node containing the maximum number

of variables corresponds always to the largest domain. In other words the number of variables on which is defined the largest potential corresponds to the number of variables of the largest initial conditional potential.

In contrast, in the junction tree, the domains of the clusters and of the separators may be very large, and so the corresponding potentials. During the junction tree construction, it may happen that the size of clusters can be very large, in particular for systems with large number of variables. The largest clique size in the junction tree increases exponentially with the number of variables in the initial DAG. It can consist of more variables than the largest initial conditional potential.

The use of the conditional form in the MBJT adds to its storage efficiency as seen in the three examples. The generalization of this statement regarding the storage efficiency in the two structures requires more experimentations.

3.3.5 Computational Efficiencies

Both of the junction tree and the MBJT are structures for local computation of marginals avoiding to compute explicitly the marginal of the joint for the variables. Thus, in the first structure, the complexity is exponential in the maximum clique size, while in the second structure, the complexity is exponential in the maximum node size. It is important to construct the junction tree that has minimal clique size and the corresponding MBJT with the minimal node size. But for a general graph, this task is computationally difficult.

We will examine the computational efficiencies by analysing the difference in the number of operations performed when computing the bridge system reliability, the parallel-series system reliability, the 1*oo*4 system reliability and the circular consecutive-1oo3 system reliability by means of the two structures.

Let us assume that binary multiplications are done for computing in the junction tree (*i.e.* potentials are multiplied two at time):
For the bridge system, propagation in the junction tree shown in 3.6 requires 5256 additions, 6608 multiplications and 404 divisions, compared to only 168 additions, 1344 multiplications and 0 divisions in the corresponding MBJT in 3.11. For the parallel-series system, 552 additions, 976 multiplications and 20 divisions are down in the junction tree shown in figure 3.5 compared to only 120 additions, 960 multiplications and 0 divisions in the corresponding MBJT in 3.12. For the 1*oo*4 system, 5124 additions, 5136 multiplications and 4 divisions are down in the junction tree shown in figure 3.4 compared to only 268 additions, 2598 multiplications and 0 divisions in the corresponding MBJT in 3.9. For the circular consecutive-1oo3 system system, 434016 additions, 592512 multiplications and 34640 divisions are down in the junction tree shown in figure 3.7 compared to only 46160 additions, 55556 multiplications and 0 divisions

in the corresponding MBJT.

For the four systems, it is clear that the MBJT is computational more efficient than the junction tree. We notice that the MBJT does fewer additions than the junction tree. In the MBJT, additions are done for marginalizing messages to the domain of the receiver, whereas in the junction tree, additions are done for the same reason as well as for computing the marginals of singleton variables.

Since computations of the marginals for those variables are done in the separators (or in the worst case in the cliques), there are further additions required to achieve these computations in the junction tree. The domains on which the potentials are defined in the junction tree are larger than the domains on which the potentials are defined in the MBJT. Moreover, all the potentials are defined in the junction tree with joint form, while some of them are defined in the MBJT in the conditional form. A penalty in the computational efficiencies arises when using the joint form which requires more operations when computing. Thus the junction tree does more additions than the MBJT.

Figure 3.14 shows the evolution of operations of addition in the two structures. We notice that the increase of the number of variables involved in the problem has more impact in the amount of performed operations of addition in the junction tree than in the MBJT.

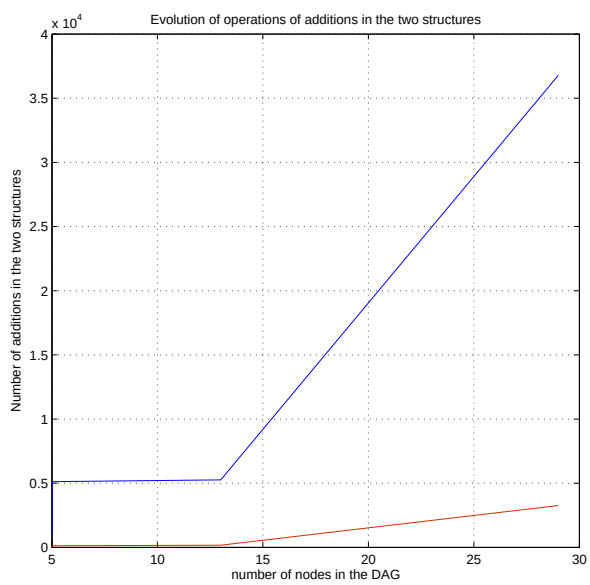

Figure 3.14: Evolution of operations of addition in the two structures

Figure 3.15 shows the evolution of operations of multiplication in the two structures. We notice that as for amount of performed operations of addition, the increase of the number of variables involved in the problem has more impact in the amount of performed operations of multiplication in the junction tree than in the MBJT.

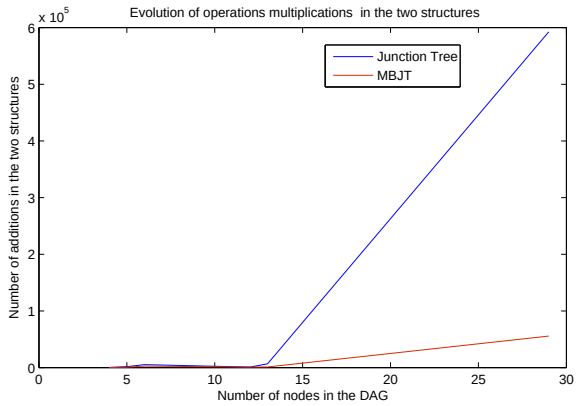

Figure 3.15: Evolution of operations of multiplication in the two structures

We notice also that the MBJT does fewer multiplications than the junction tree. In the junction tree, multiplications are done for the initialization of the structure and for updating the local potentials of the cliques during the propagation of messages, while in the MBJT, multiplications are done when combining potentials in the propagation process and when computing the marginal of the singleton variables. The junction trees corresponding to the 4 systems are composed of large cliques and separators. For instance, the junction tree corresponding to the circular consecutive 1oo3 system in figure 3.7 has 9 cliques with 8 variables and 2 cliques with 7 variables. Thus an important number of multiplications is done to assign potentials to nodes and to update their potentials during the propagation process in the junction tree.

In the MBJT, the computations of the marginals for singleton variables are done in the corresponding nodes, thus the number of multiplications done for those computations is not as important as in the junction tree. For instance to obtain the marginal for variable $C7$ in the MBJT for the circular consecutive 1oo3 system, 16 multiplications are required comparing to 1020 additions in the corresponding junction tree.

This gain in computation comes from the use of the GBT and DRC rules in the MBJT for propagating messages in order to increase the computational efficiencies by avoiding many multiplications required for computing the messages to be sent. Thus the junction tree for the circular consecutive 1oo3 does more multiplications than the MBJT. Making general statements regarding relative computational efficiencies in the junction tree and in the modified binary join

tree requires more experimentations.

3.3.6 Comparison of Models Results

Evidential network and directed evidential network are two evidential network-based approaches allowing to model complex system reliability in order to evaluate its total reliability. The evaluation of the system reliability by means of graphical models is based on the DAG modeling it and the reliability of each component. The system reliability can be also obtained by a mathematical formulation of reliability allowing to calculate the total reliability of the system based on its structure and the reliability of each component.

To show the capacity of the two evidential networks to compute the system reliability, it is very important to compare the assessment of the system reliability obtained by the evidential models-based reasoning with the value of the system reliability obtained by the mathematical formulation corresponding to the reliability block diagram of the system. The comparisons in this section are illustrated by the bridge system. Three cases are distinguished according to whether the evidences are Bayesian or not and the world is closed or not.

Precise knowledge: Bayesian frame

Let us suppose that the probability for each component $Ci_{i=1,2,5}$ to be in the state {Up} is given by $P(Ci_{i=1,2,5} = \{Up\}) = 0.7$ and the probability for each component $Ci_{i=3,4}$ to be in the state {Up} is given by $P(Ci_{i=3,4} = \{Up\}) = 0.8$. The a priori belief mass distribution of node Ci is as follows:

$$M_{i=1,2,5} = [m(\emptyset) = 0 \;\; m(\{Up\}) = 0.7 \;\; m(\{Down\}) = 0.3 \;\; m(\{Up, Down\}) = 0]$$
$$M_{i=3,4} = [m(\emptyset) = 0 \;\; m(\{Up\}) = 0.8 \;\; m(\{Down\}) = 0.2 \;\; m(\{Up, Down\}) = 0]$$

The belief mass m({Up,Down}) = 0 reveals that evidences are Bayesian and $Bel(Ci = \{Up\}) = P(Ci = \{Up\}) = Pls(Ci = \{Up\})$. That means that the problem is Bayesian. The system reliability obtained by both EN and DEVN is $R_S = 0.8534$. $Bel(S = \{Up\}) = 0.8534$ and $Pl(S = \{Up\}) = 0.8534$. The system reliability obtained by (3.3.6) is $R_S = 0.8534$.
This simple example shows that both of the evidential network and the directed evidential network compute the exact value of the system reliability when coding a priori belief masses with Bayesian evidences.

Uncertainty Knowledge: Closed-world assumption

Let us now suppose that a priori belief mass distribution defining each component state as

follows:

$$M_{i=1,2,5} = [m(\emptyset) = 0 \;\; m(\{Up\}) = 0.6 \;\; m(\{Down\}) = 0.3 \;\; m(\{Up, Down\}) = 0.1]$$
$$M_{i=3,4} = [m(\emptyset) = 0 \;\; m(\{Up\}) = 0.8 \;\; m(\{Down\}) = 0.2 \;\; m(\{Up, Down\}) = 0]$$

The uncertainty in the knowledge induces a belief mass $m(\{Up, Down\}) > 0$. It describes a non Bayesian frame in which $Bel(Ci = \{Up\}) < Pr(Ci = \{Up\}) < Pls(Ci = \{Up\})$. The system reliability obtained by both EN and DEVN is $R_S = 0.7757$. $Bel(S = \{Up\}) = 0.7757$ and $Pl(S = \{Up\}) = 0.8534$. The bridge system reliability obtained by (3.3.6) is $R_S = 0.7757$. The bridge system reliability is between [0.7757,0.8534]. We notice that the bounding property is verified.

This simple example shows that both of the evidential network and the directed evidential network compute the exact value of the system reliability when coding a priori belief masses with a closed-world assumption.

Uncertainty Knowledge: Open-world assumption
Let's now suppose that a priori belief mass distribution defining each component state is as follows:

$$M_{i=1,2,5} = [m(\emptyset) = 0.1 \;\; m(\{Up\}) = 0.6 \;\; m(\{Down\}) = 0.2 \;\; m(\{Up, Down\}) = 0.1]$$
$$M_{i=3,4} = [m(\emptyset) = 0 \;\; m(\{Up\}) = 0.8 \;\; m(\{Down\}) = 0.2 \;\; m(\{Up, Down\}) = 0]$$

The non-exhaustivity of all hypotheses induces a belief mass $m(\emptyset) > 0$. It translates an open-world in which $Bel(Ci = \{Up\}) < Pr(Ci = \{Up\}) < Pls(Ci = \{Up\})$. The system reliability obtained by the two evidential networks is $R_S = 0.7642$. $Bel(S = \{Up\}) = 0.7642$ and $Pl(S = \{Up\}) = 0.9165$. The system reliability obtained by () is $R_S = 0.7757$. It is between [0.7642,0.9165].

We can note that the two models can estimate correctly the system reliability. From this perspective, there are no essential differences between the two evidential networks, since they provide the same reliability evaluation.

3.4 Conclusion

In this chapter, we have compared two evidential networks applied to the complex system reliability, the EN proposed by (Simon et al., 2008) and the DEVN proposed by Ben Yaghlane(Ben Yaghlane and Mellouli, 2008).
This comparison was based on different criterion. Comparing the two structures applied to

evaluate many systems, the directed evidential network is computationally more efficient and more storage efficient than the evidential network. This statement regarding relative computational efficiencies and storage efficiencies in the two networks may be generalized, but this requires more experimentations by applying the two evidential networks to different complex systems in order to find a mathematical proof.

Conclusion

Complex industrial systems and equipments require a very high reliability or a very low probability of failure. Therefore, reliability analysis of these complex systems becomes today very important in many fields. The reliability of a complex system is based on its structure and on the reliability of each of its components. The information on the behavior of the system components reliability usually incorporates epistemic uncertainties. In this context, the theory of belief functions (Shafer, 1976) seems to be a great solution to efficiently handle these uncertainties.

The complexity of industrial systems makes the computation of their reliability very hard and difficult. The use of graphical models to estimate the reliability of systems is of a major interest since they are an important asset which can be used in this context to represent efficiently the relations among the systems parts and to make the computational tasks manageable (Donat et al., 2009).

Our work is situated in this context because we have applied in this master thesis two evidential models, the evidential network and the directed evidential network with conditional belief functions, to the study of system reliability in order to compare between them and to show the powerful representation and reasoning under uncertainty of each one. The comparison is led by the application of the two models on different examples of well-known systems in the reliability field and is based on different criterion.

Nevertheless, the proposed work is still subject to improvement. As future work, we will tend to investigate different horizons in order to improve this work:

- We will deal with multi-state systems. Thus, not only systems with two states can be modeled by evidential graphical models.
- We will develop a toolbox dealing with belief functions theory which will be of great help in the reliability studies of complex systems under epistemic uncertainty. This toolbox

will be considered as another alternative for Bayesialab [1] which is a tool relying on the Bayesian networks used today for applying the reliability analysis.

- We will extend the directed evidential network into the dynamic directed evidential network.

[1] http://www.bayesia.com

Bibliography

Ben Yaghlane, B. (1992). *Uncertainty representation and Reasoning in Directed Evidential Networks*. PhD thesis, Institut Superieur de Gestion de Tunis Tunisia.

Ben Yaghlane, B. (2008). A comparison of architectures for exact inference in evidential networks. In *12th Int. Conf. on Inf. Processing and Management of Uncertainty in KBS (IPMU'08)*, Malaga, Spain.

Ben Yaghlane, B. and Mellouli, K. (2008). Inference in directed evidential networks based on the transferable belief model. *International Journal of Approximate Reasoning*, 48:399–418.

Dempster, A. P. (1967). Upper and Lower probabilities induced by a multivalued mapping. *Annals of Mathematical Statistics*, 38:325–339.

Donat, R., Bouillaut, L., Neji, A., and Aknin, P. (2009). Comparison of two graphical models approaches for the modelling of multi-components system's reliability. *CIE' 09 39 th IEEE International Conference on Computers and Industrial engineering troyes*, pages 1261–1266.

Dubois, D. and Prade, H. (1988). Representation and combination of uncertainty with belief functions and possibility measures. *Computational Intelligence*, 4:244–264.

Dugan, J., Bavuso, S., and Boyd, M. (1992). Dynamic fault tree models for fault tolerant computer systems. *IEEE Transaction on reliability*, 40:363–377.

IEC60870 (1990). Telecontrol equipment and systems. *International Electrotechnical Commission (IEC)*.

IEC61025 (1990). IEC 61025: fault tree analysis. *International Electrotechnical Commission (IEC)*.

IEC61511 (2000). Functional safety: Safety instrumented systems for the process industry sector. *International Electrotechnical Commission (IEC)*.

Jensen, F. (2001). *Bayesian Networks and Decision Graphs*. Springer Verlag.

Jensen, F., Lauritzen, S., and Olesen, K. (1990). Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, 4:269–282.

Kleinerman, M. M. and Weiss, G. H. (1954). On the reliability of networks. In *Proceedings of the National Electronics Conference*, volume 10, pages 128–136.

Kozine, I. and Utkin, L. (2002). Interval valued finite markov chaines. *Reliable computing*, 8:97–113.

Kuo, W. and Zuo, M. J. (2003). *Optimal reliability modeling: Principles and Applications*. John Wiley & Sons, Inc.

Langseth, H. and Portinale, L. (2007). Bayesian networks in reliability. *Reliability Engineering and System Safety*, 92:92–108.

Lauritzen, S. and Shenoy, P. (1996). Computing marginals using local computation. Technical report, Kansas University, School of Business.

Lauritzen, S. and Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of Royal Statistical Society*, Series B, 2(50):157–224.

Lauritzen, S. L. (1996). *Graphical Models*. Clarendon Press.

Lepar, V. and Shenoy, P. (1998). A comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer architectures for computing marginals of probability distributions. *Cooper, G. and Moral, S. (Eds).UAI'98*, pages 328–337.

Lindqvist, B. H. and Doksum, K. A. (2003). *Mathematical and statistical methods in reliability*. World Scientific Publishing.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Pub.

Rae, A. and Lindsay, P. (2004). A behaviour-based method for fault tree generation. In *22nd International System Safety Conference*, pages 289–298.

Satoh, N., Sasaki, M., Yuge, T., and Yanagi, S. (1993). Reliability of 3-state device systems with simultaneous failures. *IEEE Transactions on Reliability*, 42:470–477.

Shafer, G. (1976). *A mathematical theory of evidence*. Princeton University Press.

Shenoy, P. (1991). Valuation-based systems for discrete optimization. In *Uncertainty in Artificial Intelligence (UAI)*, pages 385–400, North Holland, Amsteram.

Shenoy, P. (1997). Binary join trees for computing marginals in the shenoy-shafer architecture. *Int. J. of App. Reasoning*, 17:239–263.

Shenoy, P. and Shafer, G. (1990). Axioms for probability and belief functions propagation. *Uncertainty in Artificial Intelligence*, 4:159–198.

Shenoy, P. P. (1994). Using dempster-shafers belief-function theory in expert systems. In Ronald R. Yager, M. F. and janusz Kacprzyk, editors, *Advances in the Dempster-Shafer Theory of Evidence*, pages 395–414. John Wiley & Sons.

Simon, C. and Weber, P. (2009). Evidential networks for reliability analysis and performance evaluation of systems with imprecise knowledge. *IEEE Transactions on Reliability*, 58:69–87.

Simon, C., Weber, P., and E.Levrat (2007). Bayesian networks and evidence theory to model complex systems reliability. *Journal of Computers*, 2:33–43.

Simon, C., Weber, P., and Evsukoff, A. (2008). Bayesian networks inference algorithm to implement dempster shafer theory in reliability analysis. *Reliability Engineering and System Safety*, 93:950–963.

Smets, P. (1978). *Un modèle mathématico-statistique simulant le processus du diagnostic médical*. PhD thesis, Université Libre de Bruxelles, Belgium.

Smets, P. (1993). Belief Functions: the Disjunctive Rule of Combination and the Generalized Bayesian Theorem. *International Journal of Approximate Reasoning*, 9:1–35.

Smets, P. and Kennes, R. (1994). The Transferable Belief Model. *Artificial Intelligent*, 66:191–234.

Tavakkoli-Moghaddam, R., Safari, J., and Sassani, F. (2008). Reliability optimization of series-parallel systems with a choice of redundancy strategies using a genetic algorithm. *Reliability Engineering and System Safety*, 93:550–556.

Torres-Toledano, J. and Sucar, L. (1998). Bayesian networks for reliability analysis of complex systems. In *Artificial Intelligence-IBERAMIA 98, H. Coelho (Ed.), Springer-Verlag*, pages 195–206, Berlin.

Villemeur, A. (1987). *Evaluation de la fiabilité, disponibilité et maintenabilité des systèmes réparables : la méthode de lÉspace des Etats*. Eyrolles.

Villemeur, A. (1992). *Reliability, availability, maintainability and safety assessment: methods and techniques*. John Wiley and Sons Inc.

Villemeur, A. (1998). *Sureté de fonctionnement des systèmes industriels*. Eyrolles.

Watson, H. A. (1961). *Launch control safety study*. Bell labs.

Weber, P. and Jouffe, L. (2006). Complex system reliability modelling with dynamic object oriented bayesian networks (doobn). *Reliability Engineering and System Safety*, 91:149–162.

Weber, P. and Simon, C. (2008). Dynamic evidential networks in system reliability analysis: A dempster shafer approach. In *16th Mediterranean Conference on Control and Automation*, volume 93, pages 262–267, France.

Xu, H. and Smets, P. (1994). Evidential reasoning with conditional belief functions. In *Heckerman, D., Poole, D., and Lopez De Manatars, R., editors, Uncertainty in Artificial Intelligence*, pages 598–606, Morgan Kaufman, San Mateo, Ca.

Zadeh, L. A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1(3):3–28.