REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU
FACULTE DE GENIE ELECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'ELECTRONIQUE

MEMOIRE DE MAGISTER EN ELECTRONIQUE

OPTION : ***Télédétection***

Présenté par :

Mr. BELAL Smail

THEME :

Modélisation et analyse de sûreté des systèmes par arbre de défaillance

Devant le jury composé de :

Président :	AMEUR Soltane	Professeur, UMMTO
Rapporteur :	ZIANI Rezki	Maître de conférences A, UMMTO
Examinateurs :	HADDAB Salah	Maître de conférences A, UMMTO
	LAGHROUCHE Mourad	Maître de conférences A, UMMTO
	LAHDIR Mourad	Maître de conférences B, UMMTO

Soutenu le : 11 / 12 / 2011

REMERCIEMENTS

Je tiens à remercier vivement Monsieur R.ZIANI, maître de conférence à l'université UMMTO, de m'avoir confié ce travail en lui exprimant ma gratitude pour le soutien qu'il m'a apporté, et ma profonde reconnaissance pour le temps précieux qu'il ma consacré ainsi que pour ses encouragements, sa contribution et ses orientations.

Je tiens également à remercier chacun des membres de jury d'avoir accepter de participer au Jury qui examine ce travail.

Que tous ceux qui ont de prés ou de loin, contribué à ce travail par des renseignements, des conseils ou par un apport documentaire soient assurés de ma vive reconnaissance.

SOMMAIRE

CHAPITRE I

Généralités sur la sûreté de fonctionnement des systèmes

CHAPITRE II

Méthodes d'analyse de la sûreté de fonctionnement

CHAPITRE III

Les arbres de défaillance

CHAPITRE IV

Algorithme Kumamoto-Henley

Résumé :

L'arbre de défaillance (AdD) est une technique d'ingénierie bien connue et largement utilisée dans les études de sûreté de fonctionnement des systèmes. Elle consiste à modéliser le système par ses ensembles minimaux. Depuis 1961 qui voit naître ce concept, un grand nombre d'algorithmes pour la recherche des ensembles minimaux ont été développés. Deux algorithmes sont cités comme référence, ce sont l'algorithme MOCUS de Fussell et Vesely pour les AdD cohérents et l'algorithme de Kumamoto et Henley pour les AdD non cohérents.

L'inconvénient majeur induit par cette technique (AdD) réside dans l'espace mémoire requis au stockage de tous les ensembles minimaux obtenus et au temps d'exécution qui peut s'avérer alors prohibitif. Ceci se produit notamment, lors du traitement d'arbres de grande taille.

Pour améliorer la performance de ces algorithmes, l'effort des chercheurs a surtout porté sur l'étape de réduction. En effet, cette dernière reste celle qui consomme le plus de temps à cause du nombre de comparaisons nécessaires pour l'obtention des ensembles minimaux.

Nous nous sommes intéressés dans notre travail à l'algorithme de Kumamoto et Henley auquel nous avons apporté une amélioration. L'atout majeur de notre approche, est que lors de la recherche des ensembles (implicants), un certain nombre d'entre eux ne sont pas calculés. Ainsi, la réduction portera sur un nombre de termes plus restreint. A travers des exemples d'application nous montrons que cette nouvelle approche permet un gain important en espace mémoire et en temps d'exécution.

Mots-clés : sûreté de fonctionnement des systèmes, arbre de défaillance, systèmes non cohérents, l'algorithme de Kumamoto et Henley, ensembles minimaux, coupes minimales, implicants premiers.

INTRODUCTION

Introduction générale

La sûreté de fonctionnement est apparue récemment dans l'histoire, et s'est développée au cours du XX^{ème} siècle pour être un domaine incontournable actuellement pour les industries à risque telles que l'industrie nucléaire et l'industrie chimique, mais aussi de plus en plus pour toute l'industrie en raison de sa corrélation avec la notion de qualité, et l'impact sur l'environnement.

L'intérêt porté à la fiabilité remonte au début des années 1900 avec l'invention des tubes à vide. L'accroissement de la durée de vie des tubes fut une préoccupation constante. Lors du développement des locomotives à vapeur, la durée de vie des roulements à billes faisait également l'objet d'études systématiques. A cette époque la notion associée à la fiabilité d'un équipement était celle de durée de vie.

La fiabilité est entrée dans une ère nouvelle avec l'apparition de l'électronique. Suite à des études sur la fiabilité des équipements militaires durant la deuxième guerre mondiale, des travaux relatifs aux pannes affectant des équipements aéronautiques et électroniques ont été publiés pour la première fois. A cette époque un système fiable était considéré comme un système sur.

Vers la fin des années cinquante, avec la complexité croissante des systèmes et le haut niveau de performance recherché, la fiabilité à elle seule ne suffit plus pour caractériser la performance d'un système. D'autres notions, la disponibilité, la maintenabilité et la sécurité apparaissent. Et c'est ainsi qu'une nouvelle discipline, la "sûreté" regroupant ces quatre composantes naît.

A partir de 1961 qui voit naître le concept des arbres de défaillance, dans les laboratoires de Bell Telephone, la fiabilité entre dans une ère nouvelle. L'utilisation des arbres de défaillance pour l'évaluation de la sûreté du système de commande de lancement du missile MINUTEMAN[(1)], est la première étude importante de sûreté. Des analystes de la société BOEING font ensuite progresser cette technique et des programmes sur ordinateur sont développés. Haasl [HAS 65] et d'autres contribuent à cet important développement.

Plus tard cette technique est étendue aux centrales nucléaires et à d'autres domaines. Elle révèle son efficacité grâce au rapport WASH 1400 de Rasmussen[(2)], sur l'évaluation des

---

(1) Lunch Control Safety Study', Section V11, vol. 1, Murry Hill, NJ USA, 1961

(2) *Reactor Safety Study , An Assessment of Accident Risk in U.S.Commercial Nuclear Power Plants, WASH 1400, U.S.A.E.C, Washington, D.D 1974.*

risques d'accident dans les centrales nucléaires aux USA.

A partir de 1970, les techniques de la sûreté atteignent un haut niveau de sophistication et la théorie de la fiabilité ne cessera de connaître un progrès considérable. Les résultats obtenus pour les systèmes binaires cohérents seront généralisés aux systèmes non cohérents et aux systèmes multiperformants, c'est à dire présentant plus de deux niveaux de performance. Après la fiabilité des composants et la fiabilité des systèmes, vont également se développer la fiabilité du logiciel et la fiabilité humaine.

Depuis 1980, nous assistons à la consolidation de la théorie de la fiabilité par des disciplines adjacentes, telles que l'intelligence artificielle et la théorie des ensembles flous. La décennie 90 est notable pour l'introduction des méthodes de la sûreté de fonctionnement dans toutes les industries (automobile, production d'hydrocarbures, pétrochimie...) et aussi dans le génie civil et le bâtiment. Aujourd'hui nous pouvons compter sur une expérience très importante acquise dans le domaine des études de sûreté.

Le travail présenté dans ce mémoire porte sur la modélisation et l'analyse de sûreté des systèmes par arbre de défaillance

Le manuscrit est composé de quatre chapitres. Nous aborderons dans le premier chapitre les concepts relatifs à la sûreté de fonctionnement, les définitions et les notions de base.

Dans le deuxième chapitre, nous développerons les différentes méthodes de modélisation et d'analyse en montrant les avantages et les limites de chaque méthode.

Le troisième chapitre traitera le problème de la modélisation des systèmes par la méthode des arbres de défaillance qui est une technique majeure en sûreté des systèmes**.** Nous donnerons d'abord quelques définitions et notions de base, ensuite nous énumérerons les différentes étapes de modélisation par cette technique.

Le quatrième chapitre sera consacré aux systèmes non cohérents. Nous donnerons une description de l'algorithme de Kumamoto et Henley utilisé dans le traitement qualitatif des arbres de défaillance non cohérents et nous présenterons les améliorations que nous avons apportées à cet algorithme.

Nous terminerons notre travail par une conclusion et perspectives.

CHAPITRE I

Généralités sur la sûreté de fonctionnement

I.1 Préambule

L'évaluation de la sûreté de fonctionnement d'un système consiste à analyser les défaillances des composants pour déterminer leurs causes et estimer leurs conséquences sur le service rendu par le système. L'analyse de sûreté de fonctionnement peut être qualitative et/ou quantitative. Les méthodes utilisées suivent deux types de raisonnement logique (Figure I.1) :

- *Approche inductive* : raisonnement du particulier au général. Dans ce cas, on recherche les effets d'une défaillance sur le système ou son environnement.
- *Approche déductive* : raisonnement du général au particulier. Dans ce cas, on sait que le système est défaillant et on recherche les causes possibles de la défaillance

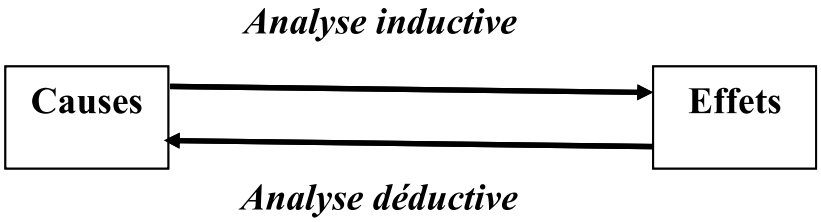


Figure I.1 Différents raisonnements logiques

I.2 Définitions et notations

La sûreté de fonctionnement (notée Sdf) peut être définie, au sens large, comme la science des défaillances [VIL 88]. Elle inclut leur connaissance, leur évaluation, leur prévision, leur mesure et leur maîtrise. Selon [LAP 96], la Sdf est la propriété d'un système permettant à ses utilisateurs de placer une confiance justifiée dans le service délivré. Au sens strict, la sûreté de fonctionnement est l'aptitude d'une entité à satisfaire une ou plusieurs fonctions requises dans des conditions données. Elle est caractérisée par quatre attributs : la fiabilité, la disponibilité, la maintenabilité et la sécurité.

I.2.1 Fiabilité

On appelle fiabilité la fonction R(t), qui correspond à la probabilité de fonctionnement sans défaillance pendant une durée t. Le dispositif étudié est supposé en état de marche à l'instant t = 0. R(t) est une fonction décroissante dans le temps, dont la valeur varie de $R(0) = 1$ à $R(\infty) = 0$.

L'équation liant la fiabilité d'un système à son taux de défaillance s'écrit :

$$\lambda(t)R(t) + \frac{dR(t)}{dt} = 0$$

Où t est le temps de mission en heure.

Et λ représente le taux de défaillance du système (ou le nombre de pannes par unité de temps). Il s'exprime en h^{-1}.

La résolution de cette équation donne :

$$R(t) = \exp - \int_0^t \lambda(u)du)$$

qui devient dans le cas de la loi exponentielle (λ(t) constant) :

$$R(t) = e^{-\lambda t}$$

Des études statistiques ont montré que l'évolution du taux de défaillance suit une courbe dite « en baignoire » représentée sur la figure I.2.

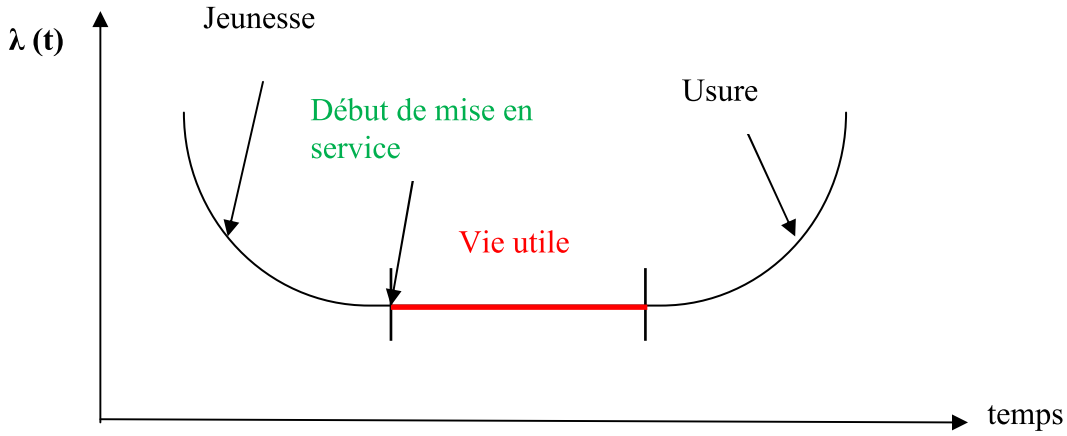


Figure I.2 Courbe en baignoire du taux de défaillance

Cette courbe montre 3 phases distinctes :

Phase 1 : période dite de jeunesse, ou de mortalité infantile. Elle se caractérise par un taux de défaillance important mais décroissant. L'occurrence de défaillances durant cette période n'est pas aléatoire au cours du temps mais plutôt le résultat de défauts de conception. En général, on s'affranchit de l'étude dans cette zone par des tests de déverminage ou rodage.

Phase 2 : période de vie utile caractérisée par un taux de défaillance faible et relativement constant. Les composants sont affectés par l'apparition de défauts aléatoires.

Phase 3 : période dite de vieillesse ou d'usure, caractérisée par un taux de défaillance croissant. L'occurrence de défaillances durant cette période est due à l'usure critique des composants. Là aussi, on peut s'affranchir de l'étude dans cette zone, car on estime en général que le composant devient obsolète avant d'y arriver (soit il tombe en panne avant d'atteindre cette zone, soit il est remplacé avant).

Ceci justifie l'utilisation de la loi exponentielle (λ constant) dans les études de fiabilité.

I.2.2 Maintenabilité

La maintenabilité d'un élément notée M(t*)* correspond à l'aptitude d'être maintenu ou rétabli dans un état dans lequel il peut accomplir une fonction requise. Elle est définie par la probabilité de maintenir ou de rétablir en un instant t le système en un état de fonctionnement. Tout comme la fiabilité est liée à un taux de défaillance $\lambda(t)$, la maintenabilité est liée à un taux de réparation noté $\mu(t)$. Le taux de réparation peut s'assimiler à la proportion d'entités réparées sur l'intervalle [t, t + d t] rapportées aux entités non réparées à l'instant t.

En prenant un raisonnement similaire à celui utilisé pour le calcul de la fiabilité et à l'aide du théorème des probabilités conditionnelles, on déduit, après des calculs simples :

$$\mathbf{M(t) = 1 - \exp\left(-\int_0^t u(\upsilon)\,\delta\upsilon\right)}$$

qui devient dans le cas de la loi exponentielle ($\mu(t)$ constant) :

$$\mathbf{M(t) = 1 - e^{-\mu t}}$$

La maintenabilité dépend de deux types de facteurs :

- ✓ *Facteurs intrinsèques* **:** ils sont propres à l'équipement. L'accessibilité, l'interchangeabilité des pièces et les moyens mis à disposition pour diagnostiquer l'origine précise de la panne, constituent les principaux facteurs.
- ✓ *Facteurs extrinsèques* : ces facteurs concernent plus particulièrement l'organisation de la maintenance. Ils touchent à la disponibilité du personnel de maintenance, à la qualification, à

l'outillage utilisé, au stock de pièces de rechange et à la politique de maintenance (préventive, curative, conditionnelle, systématique…).

I.2.3 Disponibilité

La disponibilité notée A(t), est définie comme étant la probabilité qu'un équipement soit en mesure d'accomplir sa mission à un instant donné t.

La disponibilité d'un système est tributaire de sa fiabilité et de sa maintenabilité. En effet, pour qu'un système soit en état de marche à un instant donné, il faut, soit qu'il n'ait pas arrêté de fonctionner (*fiabilité*), soit qu'il ait pu être remis en état de marche en cas de défaillance (*maintenabilité*).

Elle s'exprime comme suit :

$$A(t) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\mu+\lambda} e^{-(\mu+\lambda)t}$$

I.2.4 Sécurité

La sécurité notée S(t) indique l'aptitude d'une entité à ne pas conduire à des accidents catastrophiques. Elle est mesurée par la probabilité qu'une entité évite de faire apparaître, dans des conditions données, des événements critiques ou catastrophiques.

La figure I.3 présente les relations entre les indicateurs de sûreté.

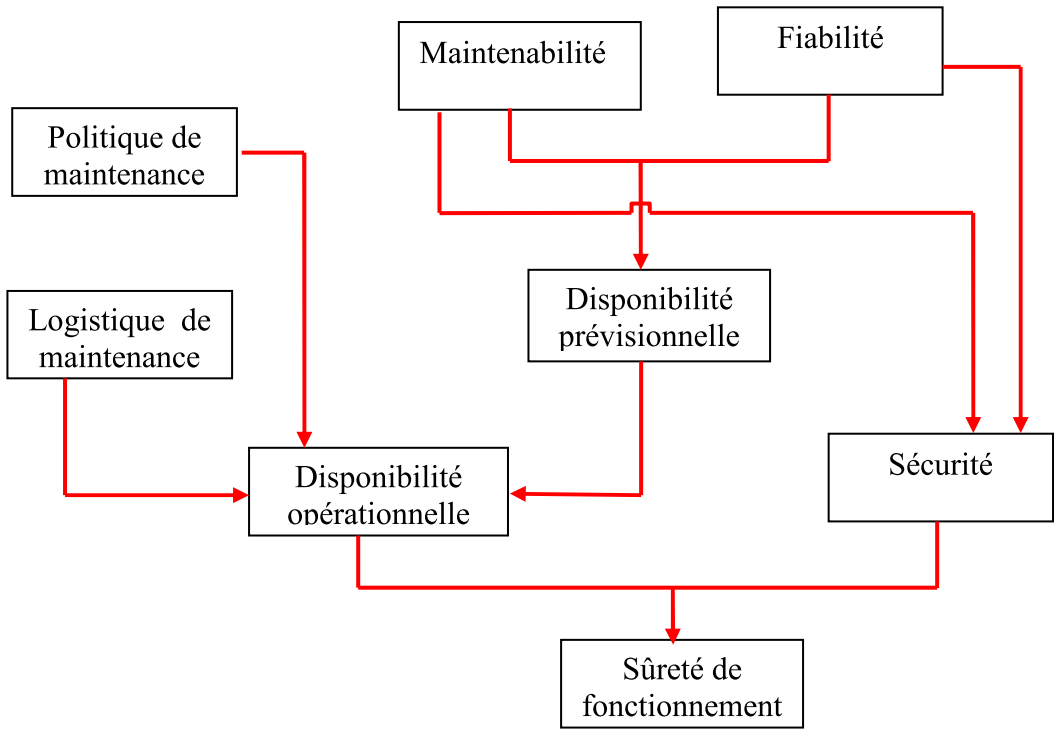


Figure I. 3 Relations entre les indicateurs de sûreté

D'autres grandeurs ou indicateurs peuvent être utilisés pour caractériser un système de sûreté. Ce sont :

- **MTTF** (Mean Time To Failure) : temps moyen de bon fonctionnement avant la première défaillance.
- **MTBF** (Mean Time Between Failure) : temps moyen entre deux défaillances d'un système réparable.
- **MDT** (Mean Down Time) : durée moyenne de défaillance comprenant la détection de la panne, la durée d'intervention, le temps de la réparation et le temps de remise en service.
- **MTTR** (Mean Time To Repair) : temps moyen de réparation.
- **MUT** (Mean Up Time) : durée moyenne de bon fonctionnement après réparation.

Ces temps moyens sont représentés dans la figure I.4.

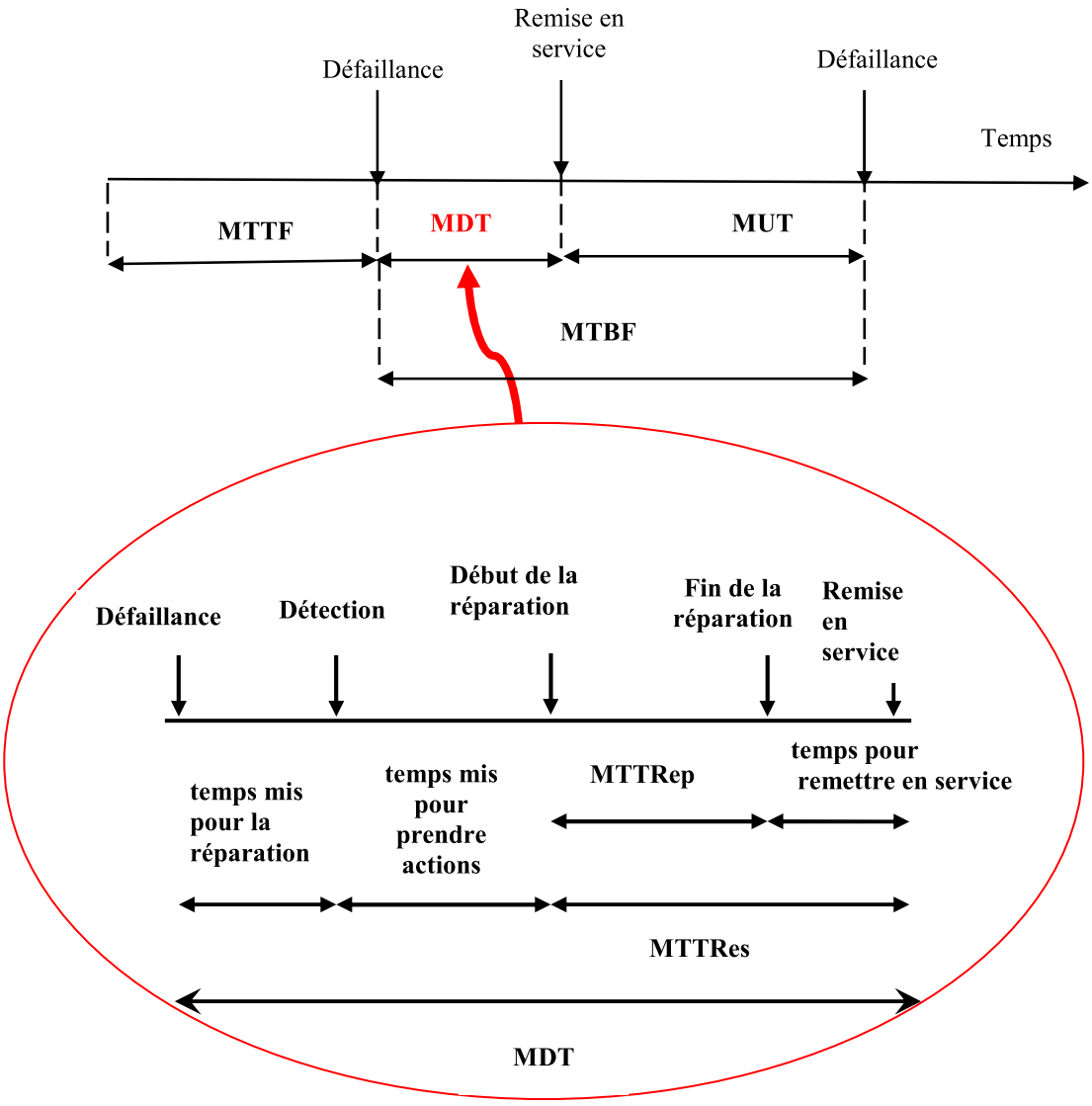


Figure I.4 Chronologie des temps moyens

On a: $\text{MTTF} = \int_0^{\infty} R(t)\,\mathrm{dt}$

$\text{MTTR} = \int_0^{\infty} [1 - M(t)]\,\mathrm{dt}$

I.3 Méthodologie générale d'une étude de sûreté

Une analyse de sûreté passe par trois étapes à savoir, l'étape préliminaire de définition et d'analyse des risques qui a pour but de fixer les objectifs de sûreté du système, et d'identifier les modes de défaillance avec les risques qu'ils représentent par la gravité de leurs conséquences, l'évaluation qualitative qui consiste en la modélisation du système par un modèle logique, l'évaluation quantitative ou stochastique qui consiste à décrire les caractéristiques du système à partir de celles de ses éléments dans un espace de probabilité.

Nous définirons tout système comme un ensemble d'éléments : matériels, logiciels, humains, assemblés en vue d'accomplir une mission ou une fonction donnée.

Soit C = {i Є N / i =1, 2, 3,…, n} un système d'ordre n formé de n éléments distincts. A l'instant t, tout élément i de C est décrit à l'aide d'une variable aléatoire $X_i(t)$ pouvant prendre N_i valeurs correspondant aux divers états possibles de i**.**

Soit $x_i(t)$ toute valeur observée de $X_i(t)$ et $E_i = \{0, 1, 2, \ldots, N_i\}$ l'espace des états possibles de i**.**
Le vecteur aléatoire $X(t) = [X_1(t), X_2(t),\ldots,X_n(t)]$ décrit conjointement les états des n éléments du système et l'état du système est décrit à l'aide de la variable aléatoire $\psi(X(t))$
La fonction **ψ** est appelée fonction de structure du système.

Nous admettons de l'expérience de chacun, que l'état d'un système est une fonction des états de ses composants. Si nous ne considérons que deux états possibles (état de fonctionnement et état de panne) pour le système et ses éléments, le système est dit binaire. Nous avons alors $|E_i|=2$, et nous adopterons la convention suivante :

$$X_i(t) = \begin{cases} 1 \text{ si l'élément i est en état de fonctionnement à l'instant t} \\ 0 \text{ si l'élément i est en état de panne à l'instant t} \end{cases}$$

et $\psi(X(t)) = \begin{cases} 1 \text{ si le système est en état de fonctionnement à l'instant t} \\ 0 \text{ si le système est en état de panne à l'instant t} \end{cases}$

I.3.1 Etapes fondamentales d'une étude de sûreté

I.3.1.1 Etude préliminaire

Elle consiste en :

- La définition du système, ses missions, ses fonctions et ses interfaces avec l'environnement.
- La décomposition physique du système pour l'indentification des parties constituantes et la mise en évidence des interrelations entre les différentes parties.
- La définition des modes de défaillances de chaque composant, c'est-à-dire, les différentes manières dont se manifeste la défaillance.
- L'identification des dangers présents, en tant que facteurs potentiels d'accident, et la détermination des risques qu'ils représentent spécialement par la gravité de leurs conséquences.
- La fixation des objectifs de sûreté, à partir de la définition du système, de sa mission, de son environnement et des critères d'évaluation des risques.

On utilise pour cela une APR (Analyse préliminaire des risques) ou une AMDEC (Méthode d'analyse des modes de défaillance, de leurs effets et de leur criticité). Ces deux méthodes ont pour rôle d'analyser les conséquences des défaillances et d'identifier les pannes ayant des conséquences importantes. Elles sont notamment nécessaires dans l'étude de sûreté des systèmes qui font appel à des technologies mal connues.

Pour réaliser une AMDEC, on utilise un tableau qui comporte les colonnes suivantes :

- composant ou sous-ensemble.
- modes potentiels de défaillance.
- causes possibles de chaque mode de défaillance.
- effets de chaque mode de défaillance sur le système.
- indice de fréquence F ou probabilité d'occurrence.

- indice de gravité G.
- les mesures mises en place pour détecter la défaillance.
- actions recommandées et/ou remarques (suggestions éventuelles...).

Composant	Fonction	Mode de défaillance	Cause de défaillance	Effet de la défaillance	Probabilité d'occurrence F	Gravité G	Criticité C	Mode de détection	Remarque

La criticité est la caractéristique du couple (probabilité d'occurrence, gravité). On l'évalue par le produit : C = F x G (plus C est grand, plus le mode de défaillance est critique). Cette notion permet de hiérarchiser les situations entre elles et d'orienter les actions correctives.

I.3.1.2 Analyse qualitative

L'analyse qualitative consiste à modéliser le système par un modèle logique. Il existe diverses représentations logiques. Les trois les plus importantes sont le diagramme bloc de fiabilité (DBF), l'arbre de défaillance (AdD), la méthode de l'espace des états (MEE). Nous reviendrons plus en détail sur ces méthodes dans le chapitre suivant.

I.3.1.3 Evaluation quantitative

L'analyse quantitative consiste à décrire les caractéristiques du système, à partir de celles de ses éléments dans un espace de probabilité :

- D'un point de vue déterministe, par la fonction de structure. Cette fonction exprime la variable d'état du système à partir de celles de ses éléments.
- D'un point de vue stochastique, par la fonction de performances. Cette fonction exprime la fiabilité ou la disponibilité du système en fonction de la fiabilité ou la disponibilité de ses éléments.

I.3.1.4 Analyse complémentaire

Des fois, il est utile lors d'une analyse de sûreté, de faire appel à une quatrième étape dite analyse complémentaire ou analyse d'incertitude. Cette étape basée sur des calculs d'importance a pour but d'évaluer l'influence relative sur le fonctionnement du système des divers éléments constitutifs afin d'identifier les points faibles du système.

Cette influence est déterminée par deux facteurs :

- La place de l'élément dans le système
- Les données de fiabilité de l'élément.

La méthode utilisée pour mesurer cette influence consiste à attribuer à chaque élément une fonction appelée facteur d'importance, et à ordonner les valeurs de cette fonction. Un grand nombre de facteurs d'importance ont été développés .Parmi les plus utilisés, nous citerons :

- Le facteur d'importance de BIRNBAUM
- Le facteur d'importance de LAMBERT
- Le facteur d'importance de VESELY-FUSSELL

I.3.1.4.1 Le facteur d'importance de BIRNBAUM [BIR 69]

Le facteur d'importance de BIRNBAUM de l'élément i indique la probabilité que le système se trouve dans un état de marche ayant l'élément i comme critique et donc si i tombe en panne, le système tombe en panne. Il est aussi appelé facteur d'importance marginal. Mathématiquement, il est défini comme étant la dérivée partielle de l'indisponibilité du système par rapport à l'indisponibilité de l'élément.

$$B_i(t) = \frac{\delta Q(t)}{\delta q_i(t)} = g[1_i, q(t)] - g[(0_i, q(t)]$$

I.3.1.4.2 Le facteur d'importance de LAMBERT [LAM 75]

Le facteur d'importance de LAMBERT évalue l'influence conjointe de l'indisponibilité d'un élément et de sa variation, sur l'indisponibilité du système. Il représente la probabilité que l'élément i ait provoqué la défaillance du système (il n'est pas forcément le seul en panne, mais dans ce cas il est le dernier à être tombé en panne) sachant que le système est défaillant. Il est de ce fait appelé facteur d'importance critique.

Il est donné par la relation :

$$C_i(t) = \frac{q_i(t)}{Q(t)} \cdot \frac{\delta Q(t)}{\delta q_i(t)}$$

$$C_i(t) = \frac{qi(t)}{Q(t)} B\,i(t)$$

Ce facteur d'importance est bien adapté pour répondre aux questions ci-dessous lorsque l'on s'intéresse à la disponibilité du système :

- Quel élément faut-il améliorer en priorité pour augmenter la fiabilité du système ?
- Quels sont les paramètres qui ont plus d'influence sur le résultat que les autres ?

I.3.1.4.2 Le facteur d'importance de VESELY-FUSSELL [FUS 75]

Ce facteur représente la probabilité que l'élément i soit en panne sachant que le système est en panne, ou encore la probabilité que le composant i contribue à la défaillance du système.

Il est donné par la relation :

$$VF_i(t) = \frac{q_i(t)}{Q(t)}$$

Ce facteur d'importance est très utilisé pour le diagnostic des causes de défaillance du système; d'où son appellation de facteur d'importance diagnostic. Il permet de répondre à la question : Quel élément faut-il réparer en priorité ? Sachant que le système est en panne.

L'information fournie par chaque facteur d'importance étant différente, le choix du facteur à calculer est fonction du but recherché. Ces calculs d'importance sont très utiles pour le développement des systèmes. Ils peuvent servir d'aide à la conception (identification des points faibles dans le système) ou d'aide au diagnostic des pannes (génération d'une liste de contrôle que pourra suivre un dépanneur).

CHAPITRE II

Méthodes d'analyse de la sûreté de fonctionnement

II.1 Préambule

L'élaboration d'un modèle de sûreté repose sur deux approches :

La première approche dite déterministe consiste à rechercher une fonction implicite ou explicite exprimant l'état du système en fonction de l'état de ses composants. La recherche de cette fonction dite fonction de structure n'est pas toujours facile surtout lorsqu'on s'éloigne des configurations classiques. En principe, une telle recherche est précédée par une représentation graphique du système basée sur une méthodologie inductive ou déductive ou mixte.

La seconde dite probabiliste consiste à définir un modèle de probabilité et à évaluer une probabilité d'occurrence d'un ou plusieurs événements définis au niveau du système à partir des grandeurs probabilistes des évènements de base. Le calcul peut être exact ou approché par bornes, par simulation ou direct, dynamique ou statique, avec ou sans maintenance.

Les méthodes inductives correspondent à une approche ascendante où l'on identifie toutes les combinaisons d'événements élémentaires possibles qui entraînent la réalisation d'un événement unique indésirable : la défaillance. Pour les méthodes déductives, la démarche est inversée puisque l'on part de l'événement indésirable, la défaillance, et l'on recherche ensuite par une approche descendante toutes les causes possibles.

Une méthode dynamique permet de prendre en compte l'évolution de la configuration des composants du système au cours du temps, contrairement à une méthode statique.

Selon la complexité du système, le fait qu'il soit réparable ou non réparable, on utilisera des méthodes de modélisation différentes. On peut regrouper les techniques utilisées en quatre classes :

- La modélisation analytique ou directe permettant des calculs manuels
- Les méthodes basées sur les ensembles minimaux (coupes minimales / chemins minimaux)
- Les méthodes basées sur les processus stochastiques (modèles markoviens)
- Les méthodes de simulation (méthode de Monté Carlo)

II.2 Les différentes méthodes d'analyse de sûreté

II.2.1 Diagramme bloc de fiabilité (DBF)

Un diagramme bloc de fiabilité est une représentation des dépendances de fiabilité entre les différents composants d'un système. C'est une méthode naturelle et simple car elle est proche de la

structure physique du système. Elle consiste à construire un diagramme composé de blocs, chacun d'eux représentant une entité (composant ou sous-système) reliés par des lignes orientées. Un bloc qui entraîne la défaillance du système est monté en série, un bloc dont la défaillance ne provoque la défaillance du système qu'en combinaison avec d'autres blocs est monté en parallèle avec ces derniers. C'est l'une des premières représentations logiques d'un système de sûreté.

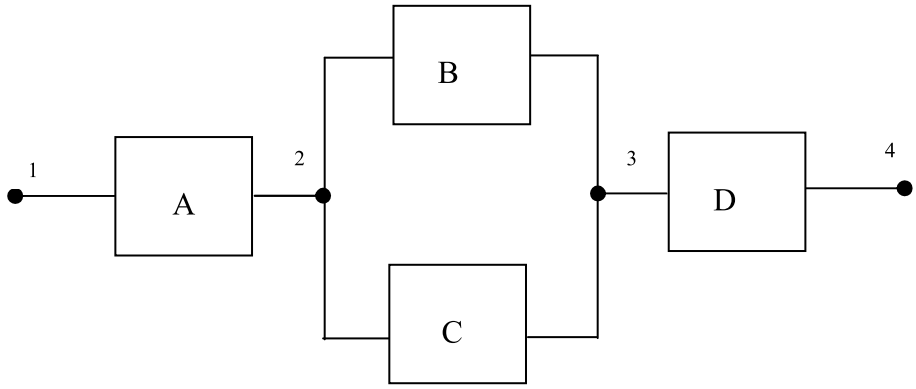


Figure II.1 Exemple de diagramme bloc de fiabilité

Cette méthode de décomposition aborde la fiabilité selon une approche structurelle. Elle permet le calcul de la fiabilité d'un système de manière relativement simple puisque l'identification des causes d'échec d'une mission est visuelle dans la plupart des cas.

Le DBF ne permet pas toujours une analyse en profondeur des défaillances. Pour les systèmes complexes et l'étude des combinaisons de défaillance, on utilisera d'autres méthodes.
Historiquement, l'arbre de défaillance a été développé pour pallier ces insuffisances [KHA 02].

II.2.2 Arbre de défaillance (AdD)

L'arbre de défaillance, appelé également arbre des fautes ou arbre des causes est une méthode déductive qui consiste à se fixer un événement sommet associé à la défaillance du système puis à rechercher les causes et toutes les combinaisons d'évènements qui conduisent à la réalisation de cet événement sommet. Il est défini comme un graphe orienté formé de niveaux successifs tel que chaque événement est généré par des évènements inférieurs agissant à travers des portes logiques. Le développement de cette méthode remonte au début des années 60 où elle fut utilisée dans les études de sûreté des systèmes aéronautiques.

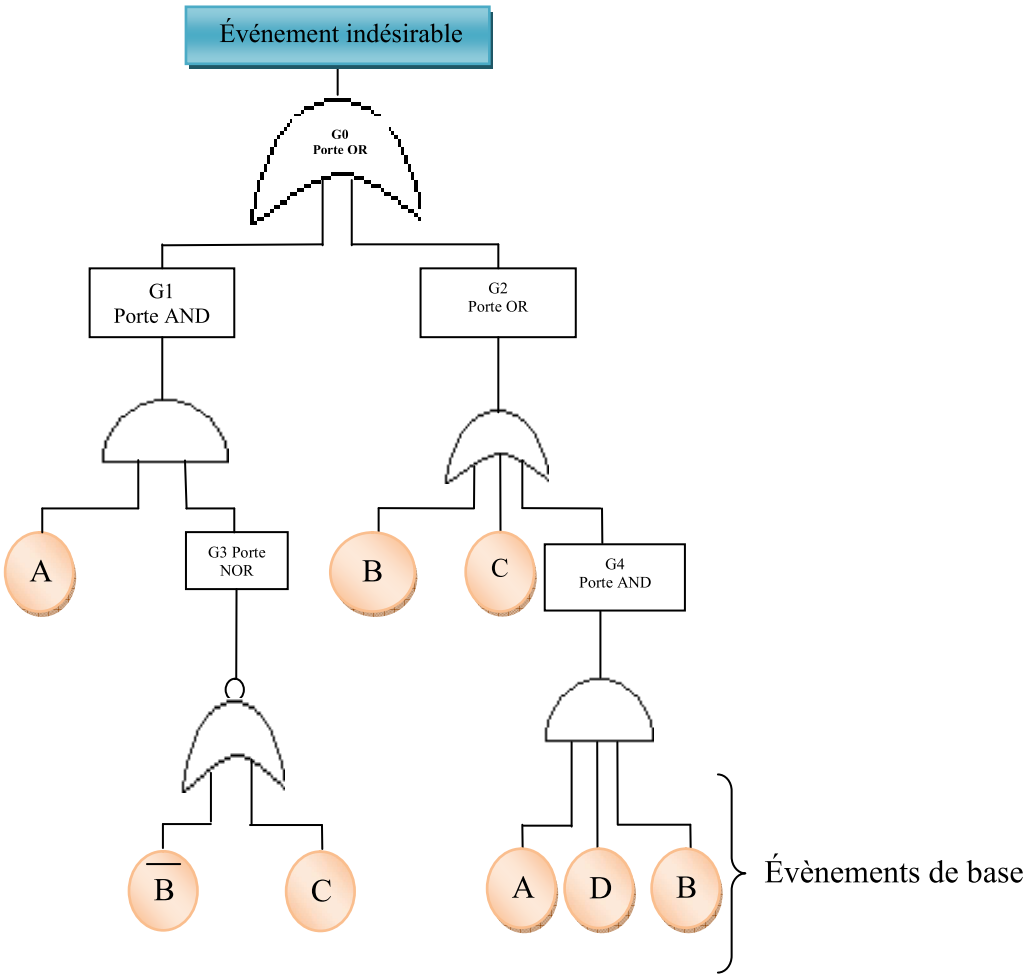


Figure II.2 Exemple d'arbre de défaillance

Cette représentation est la plus utilisée de nos jours et spécialement en sécurité des systèmes. Nous y reviendrons plus en détail dans le chapitre III.

II.2.3 Arbre d'Evénements (AdE)

L'arbre d'évènements, appelé également arbre des conséquences est issu de l'industrie nucléaire. Il trouve ses applications dans les études de sûreté des systèmes à fonctionnement binaire (état de marche ou état de panne) et à évolution chronologique (mécanisme non bouclé). La séquence des événements de l'arbre se déroule de façon inductive, à partir de l'événement initiateur jusqu'aux événements finaux, chaque nœud faisant l'objet d'une recherche de probabilité d'occurrence [VIL 88].

Les arbres d'évènements sont conventionnellement construits horizontalement, à partir de la gauche, c'est-à-dire à partir de l'événement initiateur. Le développement de l'arbre se fait alors chronologiquement, en étudiant le comportement de chaque élément : succès ou échec. Cette

méthode exige de recenser tous les scénarios pour avoir une description complète des comportements potentiels du système étudié.

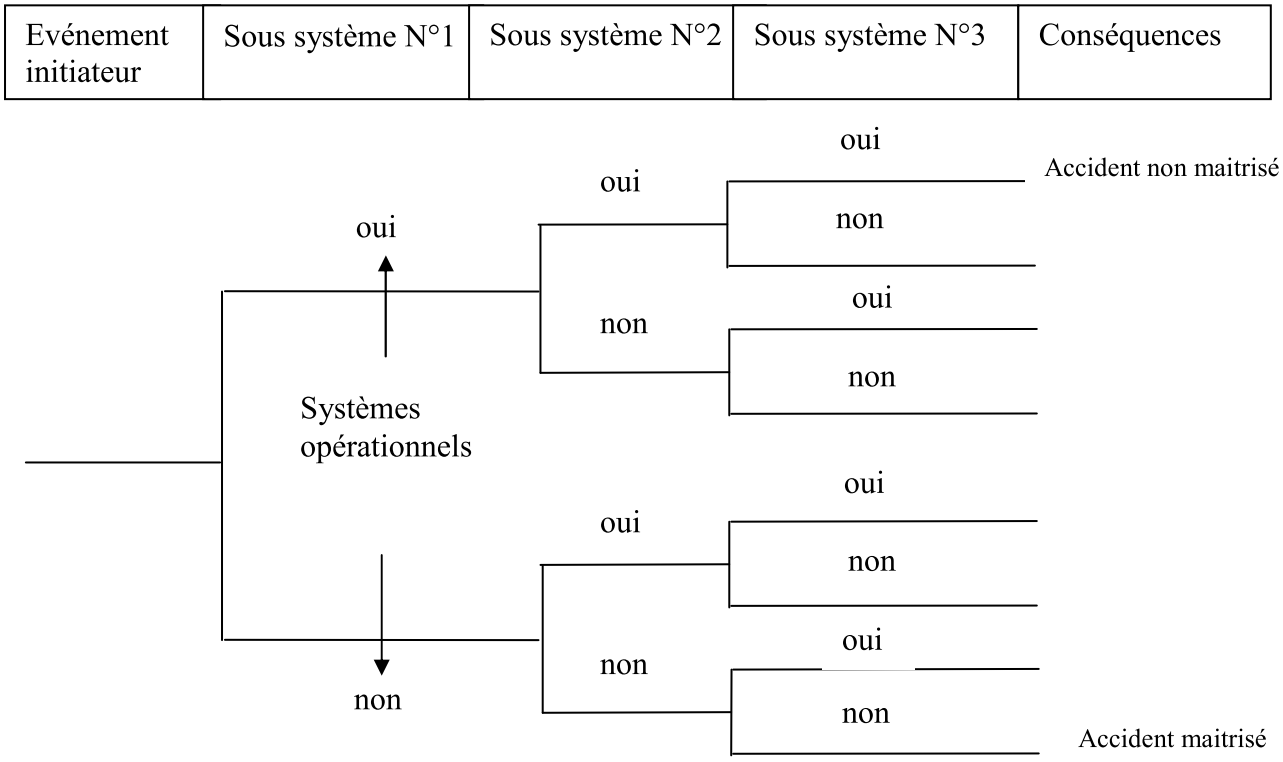


Figure II.3 Structure d'un arbre d'événements

II.2.4 Méthode de l'espace des états (MEE)

La MEE [COR 75], [MAZ 81] consiste à considérer le système comme un ensemble de composants pouvant se trouver dans un nombre fini d'états de fonctionnement ou de panne. Il faut alors construire un graphe formé de sommets et d'arcs. Les sommets représentent les différents états du système. Les arcs représentent les transitions entre les états et sont caractérisés par deux paramètres, le taux de défaillance λ et le taux de réparation μ. Les chaînes de Markov à paramètre continu (CdM) sont particulièrement utilisées.

Son principal intérêt réside dans la possibilité de pouvoir faire les calculs en fonction du temps et de modéliser certaines formes de dépendances. Elle tient compte également de la dynamique du système. Son inconvénient est l'explosion combinatoire due à l'énumération de tous les états possibles du système (pour un système binaire à n éléments, le graphe de Markov contient 2^n états).

Un système dynamique passe d'état en état au bout de durées aléatoires régies par les divers phénomènes (défaillances de composants, réparations…) auxquels il est soumis. La figure II.4 donne un exemple de ce phénomène.

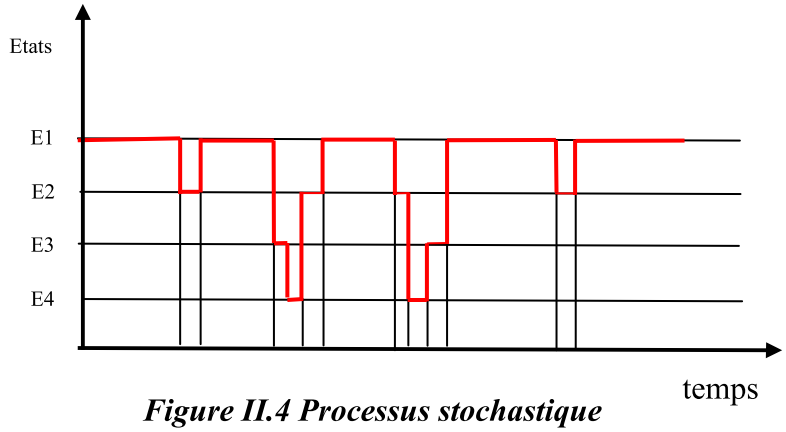


Figure II.4 Processus stochastique

Ce comportement est dit stochastique et sa modélisation relève du ressort des processus stochastiques.

Plusieurs solutions s'offrent à l'analyste qui désire mettre en œuvre les processus stochastiques :

- développer un modèle analytique spécifique à son problème.
- utiliser les processus markoviens qui sont les plus simples des processus stochastiques.
- réaliser une modélisation par réseau de Pétri stochastique .
- simuler directement le comportement du système à l'aide de la méthode de Monte-Carlo.

Soit le modèle de Markov d'un système constitué de deux machines identiques travaillant en parallèle (Figure II.5).

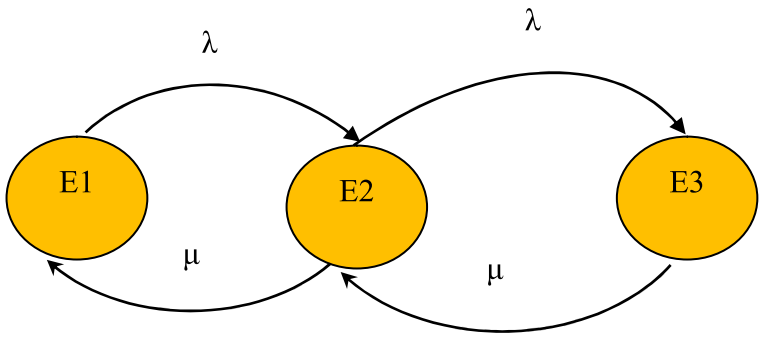


Figure II.5 Modèle de Markov avec deux machines identiques

E1, E2 et E3 représentent les trois états possibles du système

- E1 : les deux machines fonctionnent,
- E2 : une machine est défaillante,
- E3 : les deux machines sont en panne,

λ et μ sont le taux de défaillance et le taux de réparation d'une machine,

Le graphe est traduit en un système d'équations différentielles du premier ordre à coefficients constants, dit de Chapman-Kolmogorov.

$$P_1 = -2\lambda P_1(t) + \mu P_2(t)$$
$$P_2 = -(\lambda+\mu) P_2(t) + 2\lambda P_1(t) + \mu P_3(t)$$
$$P_3 = -\mu P_3(t) + \lambda P_1(t).$$

avec :

$$Pi = \frac{Pi(t+dt) - Pi(t)}{dt}$$

On peut ainsi calculer les trois grandeurs de sûreté : la fiabilité, la disponibilité, la maintenabilité, ainsi que les grandeurs moyennes MTBF, MTTR,..etc.

II.2.5 Les réseaux de Pétri stochastiques [BEN 07], [MAR 86]

Les réseaux de Petri ont été inventés en Allemagne en 1962 par le Dr Petri, à l'origine pour décrire le comportement d'automates. Un réseau de Pétri se représente par un graphe biparti orienté constitué de places (représentées par des cercles Pi) figurant les différents états susceptibles d'être occupés par le système, de transitions (traits horizontaux Tj) représentées par des tirets transversaux divisant les arcs précédents en arcs amont et arcs aval. Pour chaque transition, on attribue un délai déterministe ou probabiliste, un ensemble fini d'arcs qui font le lien entre les transitions et les places, et un ensemble fini de messages représentés par des lettres et qui sont des expressions booléennes permettant de gérer les processus interactifs évoluant en parallèle.
Deux places ne peuvent pas être reliées entre elles, ni deux transitions. Les places peuvent contenir des jetons, représentant généralement des ressources disponibles. A chaque transition, on attribue un délai déterministe ou probabiliste. La dynamique du système est représentée par le déplacement de « jetons » de place en place, suite au « tir » de transitions.

Un réseau de pétri ressemble au schéma suivant :

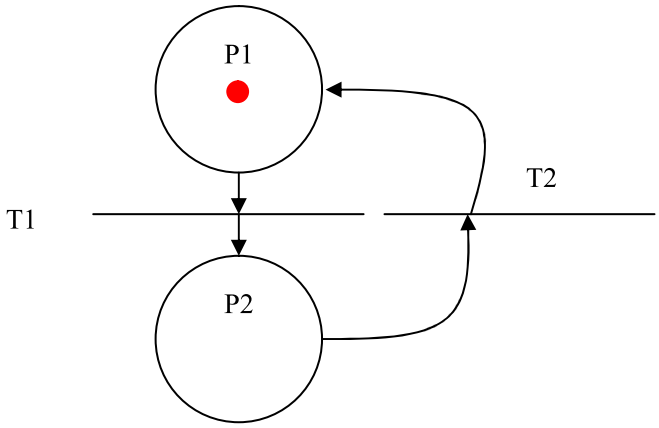


Figure II.6 Eléments du réseau de Pétri

II.2.6 Simulation de Monté Carlo

La simulation de Monte Carlo est une technique utilisée pour estimer la probabilité de résultats en répétant un grand nombre de fois une expérience à l'aide de la simulation et en utilisant des variables aléatoires [KEH 04]. On entend par simulation toute méthode qui a pour but d'imiter un système réel. On utilise généralement la simulation de Monte Carlo lorsque d'autres analyses sont mathématiquement trop complexes.

Le principe de cette méthode est basé sur la simulation du modèle global d'un système, en fonctionnement ou en dysfonctionnement pendant sa durée de vie. Si un événement redouté apparaît, on arrête la simulation et on comptabilise le nombre d'événements redoutés et on recommence une nouvelle histoire. Si aucun événement redouté ne se produit, on continue la simulation jusqu'à une date limite de fin de simulation et on recommence une nouvelle.

La probabilité de l'événement redouté étudié est estimée par le quotient du nombre de simulations ayant abouti à l'occurrence de cet événement par le nombre global de simulations. La probabilité recherchée est d'autant plus précise que le nombre de simulations est plus grand ; il est de l'ordre de 100.000 simulations.

La simulation de Monte-Carlo constitue une méthode très intéressante car elle donne accès à de nombreux paramètres inaccessibles par les autres méthodes et conduit à des analyses extrêmement détaillées des systèmes étudiés :

- Elle n'est pas limitée par le nombre d'états du système étudié car, même s'il y en a des centaines de milliers, seuls les états prépondérants se manifestent au cours de la simulation ;
- Elle permet la prise en compte de n'importe quelle loi de probabilité ;
- Elle permet l'association dans le même modèle de phénomènes déterministes et de phénomènes aléatoires ;
- Son implémentation informatique est facile.

En raison de l'augmentation continue de la puissance des moyens informatiques cette méthode est de plus en plus utilisée.

II.2.7 Les diagrammes de décision binaires (BDD)

Les diagrammes de décision binaires [COU 94] sont apparus plus récemment. Ils se basent sur une sorte de réduction de l'arbre de factorisation de Shannon mise au point dans le traitement de fonctions booléennes par Akers [AKE 78] et plus tard par Bryant [BRY 92]. La figure II.7 illustre la transformation d'un arbre de Shannon en BDD [THO 02].

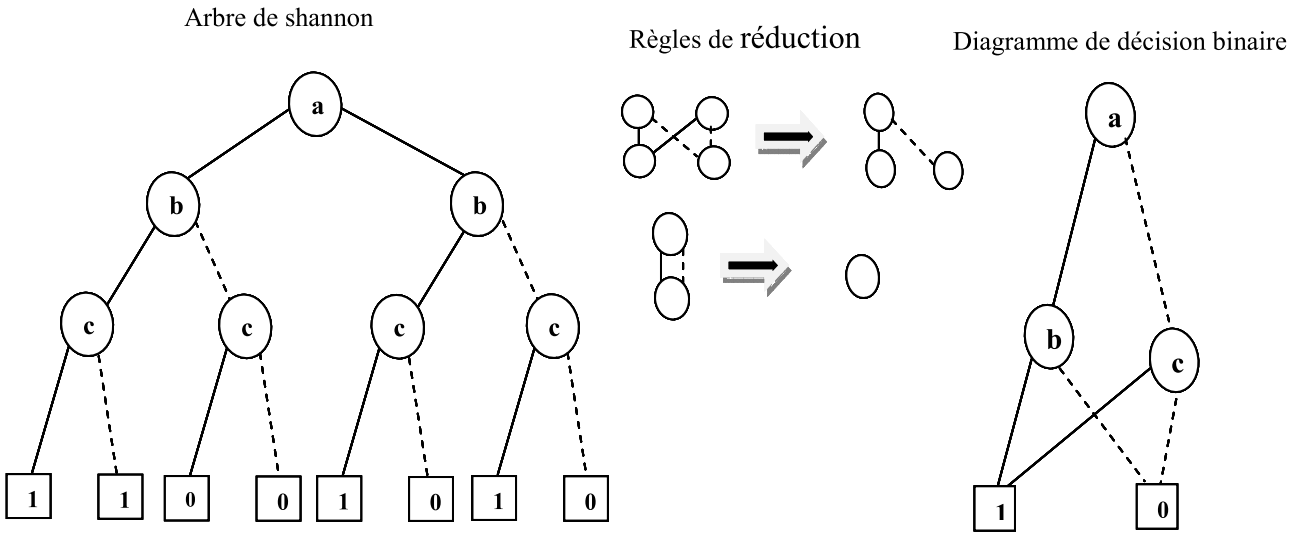


Figure II.7 Transformation d'un arbre de Shannon en BDD

A l'aide de cette nouvelle technique, une algorithmique complète (analyse logique, évaluation probabiliste, calcul d'importance) est proposée dans la littérature [RAU 93], [ODE 95]. Ces algorithmes basés sur les DDB sont plus rapides que les algorithmes classiques. Il faut cependant noter que ces algorithmes n'ont pas changé la nature du problème qui reste toujours

NP-complet. Des cas difficiles à résoudre même par ces algorithmes peuvent se présenter. Ceci justifie l'effort de recherche d'algorithmes toujours plus performants [CAR 99], [LIM 05].

II.2.8 Méthodes hybrides

L'arbre de défaillance connaît des limitations quant à sa représentation graphique statique qui ne tient compte ni de l'ordre d'apparition des événements ni des dépendances fonctionnelles. Ces deux facteurs sont pourtant très déterminants dans un système physique [BOU 03], [JEN 98], [VIL 88].

Afin de combler en partie ces limites, d'autres modélisations plus complexes ont été développées. Parmi les approches intégrant l'aspect dynamique des systèmes dans les modèles d'analyse de la sûreté de fonctionnement, nous citerons la méthode des graphes de flux dynamique (DFM), proposée dans [KHA 01], [KHA 02], [KHA 03], [LAP 96] et [PAG 80]. Une telle technique de modélisation et d'analyse a permis d'étudier les risques des systèmes embarqués. Elle a été aussi utilisée dans les études de risque des systèmes aérospatiaux et nucléaires [FUL 88] et [KHA 01]. Nous citerons également deux méthodes hybrides basées sur les arbres de défaillance : les BDMP (Boolean Logic Driven Markov Processes) et les FTPN (Fault Tree Petri Net)

II.2.8.1 La méthode BDMP

Les BDMP (Boolean logic Driven Markov Processes) [BOU 03] ont pour but de faciliter la construction et la résolution de modèles markoviens de grande taille en combinant dans le même paradigme des concepts issus des arbres de défaillance et des graphes de Markov.

Le principe simplifié du formalisme des BDMP est de remplacer :

- Les modèles simples de feuilles d'un arbre de défaillance par des processus de Markov quelconques. Les états de ces processus sont classés en deux catégories. Suivant la catégorie à laquelle appartient l'état d'une feuille à un instant donné, l'événement correspondant à cette feuille est considéré comme vrai ou faux.
- L'indépendance totale des feuilles d'un arbre de défaillance par des dépendances simples. Chaque feuille a deux modes "sollicité" et "non sollicité", correspondant à deux processus de Markov différents. Le choix du mode dans lequel une feuille se trouve à un instant donné, est déterminé par la valeur, vraie ou fausse, d'un ensemble de feuilles. Les transitions entre ces deux modes définissent éventuellement des états instantanés dans lesquels on peut déclencher

des transitions instantanées probabilisées, pour modéliser par exemple des refus de démarrage...

La structure globale d'un BDMP est donnée par une fonction logique de type arbre de défaillance (Figure II.8).

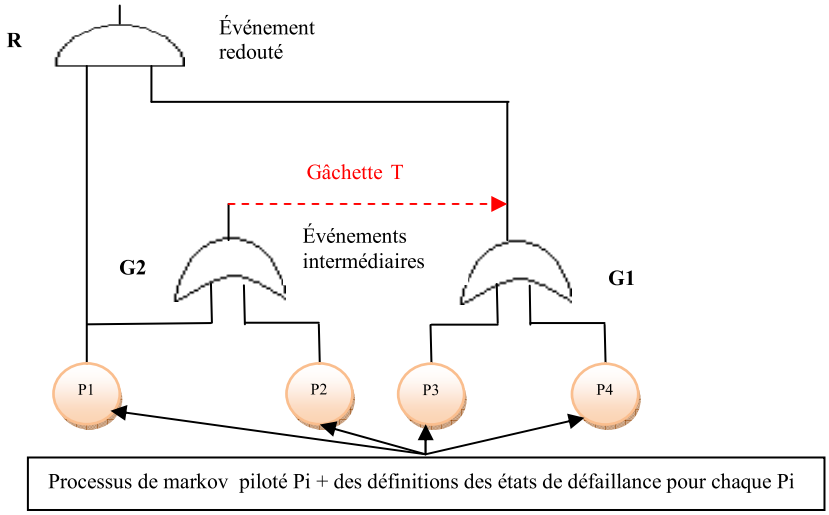


Figure II.8 Structure d'un BDMP de quatre processus

Un BDMP est constitué dans sa forme la plus simple, par les éléments suivants :

- un arbre de défaillances.
- un événement redouté R.
- deux événements intermédiaires G1 et G2.
- un ensemble de gâchettes T.
- un ensemble de processus de Markov pilotés Pi associés aux événements de base de l'arbre de défaillance.
- une définition de deux catégories d'états (marche et panne) pour les processus Pi.

Le principal événement redouté (R) du BDMP est sensé représenter l'ensemble des événements de base ou intermédiaires causant la panne du processus markovien global.

Le BDMP de la figure précédente a la structure logique d'un arbre de défaillance. Il représente, en plus, une gâchette T, provenant de la porte G1 et pour cible la porte G2, et des

définitions pour les processus Pi. La gâchette entre les deux portes G1 et G2 joue un rôle d'activation ou d'inhibition des modes de défaillances des processus P3 et P4.

Bien que cette méthode hybride ait atténué le problème majeur de l'arbre de défaillance, elle reste toujours limitée au niveau des lacunes liées au modèle de Markov, car elle ne tient pas compte des dépendances entre les événements [BOU 03].

II.2.8.2) La méthode FTPN

Les FTPN [BEN 07] constituent une approche hybride basée sur la combinaison conjointe et simultanée des arbres de défaillance et des réseaux de Pétri stochastiques. L'intérêt principal de cette méthode, consiste en l'aspect dynamique introduit dans les arbres de défaillance par l'exploitation spécifique des propriétés des réseaux de Pétri stochastiques. Il s'agit d'accorder à chaque événement du système étudié, un modèle de réseau de Pétri qui décrit son état : marche, panne ou réparation, tout en respectant ses interactions avec les autres ressources (Annexe 4). Le réseau de Pétri utilisé doit satisfaire aux propriétés d'être borné, vivant et réinitialisable.

II.3) Avantages et limites des méthodes de la sûreté de fonctionnement

Le tableau ci-dessous résume les avantages et les limites des principales méthodes d'analyse de sûreté.

Méthodes	Type de démarche	Type d'évaluation	Avantages	Limites et inconvénients
Diagramme bloc de fiabilité	/	Qualitative et quantitative	Simple et général	Ne permet pas l'analyse en profondeur de la défaillance.
Arbre de défaillance	Déductive	Qualitative et quantitative	Facilement compréhensible par des personnes autres que le créateur même.	-Représentation statique, ne tient pas compte ni de l'ordre d'apparition d'un événement, ni des dépendances fonctionnelles entre les différents composants du système.

Modèle de Markov	Déductive	Quantitative	-Facilite la recherche des grandeurs instantanées fondamentales de la SdF. -permet de comparer des architectures et des modes opératoires.	-Explosion combinatoire du nombre d'états pour les grands systèmes, ce qui rend le mode d'utilisation un peu difficile pour les non spécialistes. -limités aux phénomènes exponentiels.
Réseau de Pétri	Déductive	Quantitative	-Dynamique en fonctionnement (conditions, durée) -Structures de contrôle: Parallélisme, synchronisation -Répond aux besoins de comptage (ressources)	-Modélisation typique pour chaque système. Il n'est pas adapté pour la manipulation des données. -Absence de hiérarchie
BDMP	Déductive	Qualitative et Quantitative	- permet d'entrer progressivement dans les niveaux de détail d'un tel modèle -facilite les calculs probabilistes	-Perte de temps à examiner des coupes non minimales. -elle ne tient pas compte des dépendances entre les événements

Synthèse des principales méthodes d'analyse de la Sdf

CHAPITRE III

Les arbres de défaillance

III.1 Préambule

Les arbres de défaillance constituent une technique majeure en sûreté de fonctionnement. Ils sont utilisés en tant qu'outil principal dans la plupart des études de sécurité des systèmes, mais aussi dans les études de fiabilité et de disponibilité. Ils ont été conçus en 1961 par Watson dans les laboratoires de Bell Telephone, où ils furent utilisés pour la première fois dans le programme de développement du missile Minuteman comme nouvelle méthode d'analyse de la sécurité d'un système. Son efficacité a été notamment révélée grâce au rapport «WASH 1400» de Rasmussen sur l'évaluation des risques d'accident dans les centrales nucléaire.

Actuellement beaucoup de sociétés industrielles utilisent les arbres de défaillance pour étudier la fiabilité ou/et la sécurité de leurs produits, systèmes de production ou services dans des domaines très variés : nucléaire, espace, armement, chimie, automobile, traitement des eaux…etc

III.2 Définitions et notions de base

L'arbre de défaillance noté AdD est défini comme un graphe orienté formé de niveaux successifs, tel que chaque évènement est généré par des évènements d'ordre inférieur, agissant à travers des portes logiques. Son principe consiste à partir d'un événement « sommet » associé à la défaillance du système et à rechercher les causes puis toutes les combinaisons d'évènements qui conduisent à la réalisation de cet événement sommet.

Selon la nature du système, nous distinguons deux types d'arbres de défaillance : les arbres de défaillance cohérents (AdD-C) destinés à modéliser les systèmes cohérents et les arbres de défaillance non cohérents (AdD-NC) utilisés pour modéliser les systèmes non cohérents [ZIA 86]. Alors que les AdD-C sont caractérisés par les opérateurs logiques ET, OU et par des variables de base monoformes représentant la défaillance des éléments du système (évènements élémentaires de même catégorie x_i), les AdD-NC font appel en plus à des opérateurs logiques tels que NAND, NOR, EOR, NON et peuvent contenir des variables de base biformes représentant la défaillance et la non défaillance des éléments (évènements élémentaires de catégorie différente x_i et $\overline{x_i}$).

III.3 Modélisation et analyse par la méthode des arbres de défaillance

La modélisation par arbre de défaillance passe par trois étapes :

- Construction.
- Analyse qualitative.
- Analyse quantitative.

Parfois une quatrième étape dite « analyse de sensibilité » peut être utile.

III.3.1 Construction de l'arbre

La construction est un travail très vaste qui nécessite une très bonne connaissance du système et de son utilisation. Une analyse des modes de défaillance de leurs effets et criticité (AMDEC) des dispositifs fournit une aide à la construction.

La construction peut se résumer à trois étapes :

- Analyse préliminaire de définition et d'analyse des risques.
- Spécification.
- Construction.

III.3.1.1 Analyse préliminaire

Cette étape consiste en :

- Une décomposition du système.
- Une identification des composants du système.
- Une définition des modes de défaillance des composants.
- Une reconstitution du système par ses composants.

III.3.1.2 Spécification

La spécification consiste en :

- La détermination des phases (différents modes de fonctionnement du système). Par exemple, pour un avion, on a au moins trois modes : le décollage, le vol en altitude et l'atterrissage.

- La détermination des conditions aux limites (elles concernent les interactions du système avec son environnement).
- Hypothèses spécifiques (concernant les conventions faites sur le système).
- Conditions initiales (hypothèses concernant le début de la phase étudiée).

III.3.1.3 Construction

La construction commence d'abord par la définition de l'événement sommet de l'AdD, encore appelé événement indésirable ou évènement redouté, qui doit être défini sans ambiguïté, puis sa décomposition en événements intermédiaires. Ces événements intermédiaires sont développés à leur tour et la construction s'achève lorsque tous les évènements causes sont décomposés. On obtient alors des évènements élémentaires correspondant aux modes de défaillance des composants.

Le graphisme se fait à l'aide de trois types de symboles (Annexe 1) :

- Les opérateurs ou portes logiques : ET, OU, NAND, NOR, EOR, NON
- Les événements intermédiaires ou élémentaires.
- Les triangles de transfert : pour montrer les transferts dans l'arbre.

La difficulté et le coût rencontrés lors de la construction ont conduit à concevoir des techniques systématiques de construction afin de pouvoir automatiser cette étape. Deux familles de techniques ont été développées :

- le code DRAFT conçu pour des systèmes électriques, l'idée consistant à modéliser chaque composant par une fonction de transfert de défaillance
- le code CAT qui se sert des tables de décision pour modéliser les états des composants du système.

Ces codes sont décrits de façon détaillée dans [HAS 80].

III.3.2 Evaluation qualitative

L'évaluation qualitative ou l'analyse logique consiste en la recherche des ensembles minimaux (ensemble d'éléments dont la défaillance conduit à la défaillance du système). Ces ensembles minimaux sont appelés coupes minimales / chemins minimaux dans le cas des AdD-C et implicants premiers / impliqués premiers dans le cas des AdD-NC. Les chemins et les impliqués

sont un concept dual des coupes ou implicants, ce sont un ensemble d'éléments dont les états de bon fonctionnement simultanés sont nécessaires pour assurer le fonctionnement du système.

Les coupes minimales sont obtenues en formant l'expression booléenne de l'événement sommet en fonction des évènements de base (fonction de structure). Cette expression est ensuite réduite à sa forme minimale. On dira que la fonction de structure est réduite si on ne peut plus lui appliquer les lois de l'absorption de l'algèbre de Boole :

$$x.y + x = x \quad \text{et} \quad (x + y).x = x.$$

Un grand nombre d'algorithmes pour la recherche des ensembles minimaux ont été développés [LEE 85], [LIM 05]. Ce fut d'abord des algorithmes de simulation et des algorithmes de manipulations algébriques, puis des algorithmes de décomposition dits matriciels. Les deux algorithmes classiques parmi les algorithmes matriciels sont :

- l'algorithme MOCUS de Fussell [FUS 72] pour les systèmes cohérents
- L'algorithme de Kumamoto et Henley [KUM 78] pour les systèmes non cohérents.

III.3.3 Evaluation quantitative

L'évaluation quantitative ou l'analyse probabiliste consiste à calculer la probabilité d'occurrence de l'évènement sommet, que l'on désignera par événement TOP, ce qui revient à déterminer la probabilité de défaillance du système à partir de celles de ses éléments. Elle utilise les coupes minimales et nécessite la connaissance des données de fiabilité de chacun des composants.

Le calcul peut se faire de manière directe si l'AdD ne contient pas d'évènements répétés ou de façon plus générale à l'aide de la méthode d'inclusion-exclusion ou formule de Sylvester Poincaré. Il est également possible de faire un calcul approché. L'indisponibilité du système est alors encadrée par des bornes [LIM 05].

III.3.3.1 Calcul direct

Cette méthode consiste à partir des évènements de base et à remonter dans l'arbre de défaillance jusqu'à l'évènement sommet. Les probabilités d'occurrence en sortie des portes sont obtenues comme suit (Annexe 3):

- Pour une porte ET ayant comme entrées les évènements E_1 et E_2 de probabilité d'occurrence p_1 et p_2, nous avons :
  $P(E_1. E_2) = P(E_1)$ X $P(E_2) = p_1. p_2$
- Pour une porte OU ayant comme entrées les évènements E_1 et E_2 de probabilité d'occurrence p_1 et p_2, nous avons :
  $P(E_1+ E_2) = P(E_1) + P(E_2) - P(E_1)$ X $P(E_2) = p_1 + p_2 - p_1. p_2$

Sachant que l'évènement E_i représente la défaillance de l'élément i, p_i correspond à l'indisponibilité de cet élément. Connaissant le taux de défaillance λi et le taux de réparation μi de cet élément, on a :

$$p_i = [\lambda i / (\lambda i + \mu i)] \, e^{-(\lambda i + \mu i)t}$$

Généralement, les calculs sont faits à l'état stationnaire ou état limite ($t \rightarrow \infty$). On a alors

$$p_i = \lambda i / (\lambda i + \mu i) = \lambda i \, \tau i / (1+\lambda i \, \tau i)$$

avec τi désignant le temps de réparation de l'élément i ($\tau i = 1/ \mu i$)

Dans le cas d'éléments présentant une bonne fiabilité, donc un taux de défaillance faible, et avec une politique de maintenance efficace consistant à minimiser le temps de réparation, l'indisponibilité est alors approximée par la relation :

$$p_i = \lambda i \, \tau i$$

III.3.3.2 Formule de Sylvester Poincaré ou Méthode d'inclusion-exclusion

Soient $K = \{K_1, K_2, \ldots, K_n\}$ l'ensemble des n coupes minimales, et $C = \{C_1, C_2, \ldots, Cm\}$ l'ensemble des m chemins minimaux de l'AdD. Le système peut être représenté par ses coupes minimales ou ses chemins minimaux (Figures III.1 et III.2)

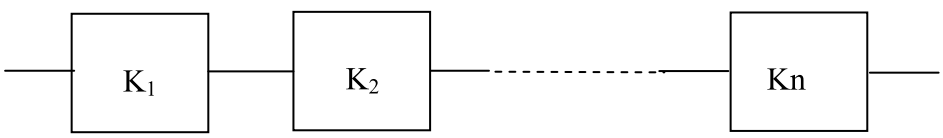


Figure III.1 Modélisation par les coupes minimales

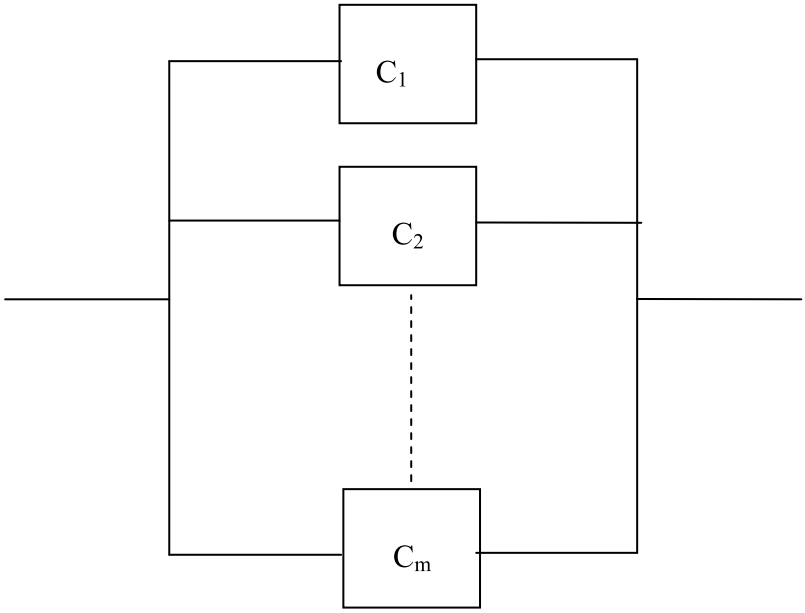


Figure III.2 Modélisation par les chemins minimaux

1- Par les coupes minimales

$$Q = P_{Top} = \text{Prob}\ \{ K_1 \cup K_2 \cup \text{----} \cup K_n)$$

$$= S_1 - S_2 + \text{---} + (-1)^{j-1} S_j \text{---} + (-1)^{n-1} S_n$$

$$= \sum (-1)^{r-1} S_r$$

avec $S_1 = \sum P(K_i) = P(K_1) + P(K_2) + \text{------} + P(K_n)$

$i=1,n$

$S_2 = \sum P(K_i \cap K_j)$

$i<j$

$S_r = \sum P(K_i \cap K_j \text{------} \cap K_r)$

$1 \leq i < j < \text{----} < r \leq m$

Et la probabilité d'occurrence de la coupe K_j donnée par :

$$P(K_j) = \prod_{i \in K_j} p_i$$

2- Par les chemins minimaux

$$P_{Top} = 1 - \text{Prob} \{ C_1 \cup C_2 \cup \text{----} \cup C_m)$$

3-Bornes de l'inclusion – exclusion

$$S_1 - S_2 \leq P_{Top} \leq S_1$$

Si la disponibilité q_i du composant i est proche de 1, ce qui est souvent le cas en pratique, le terme S_1 donne une valeur assez proche de la valeur exacte.

Les données de fiabilité à savoir le taux de défaillance λ et le taux de réparation μ des circuits sont à la base de toute étude de sûreté et malheureusement ces données ne sont pas toujours disponibles. Des sources de données de fiabilité existent certes. Elles sont fournies par des normes ou recueils, dont les principales sources sont :

- Le Military Handbook (MIL-HDBK-217F) : norme américaine créée en 1962, conçue pour les systèmes électroniques du secteur militaire.
- Le Recueil de Fiabilité (RDF 70 puis RDF 2000 ou CNET 2000) : recueil de fiabilité construit à partir d'un retour d'expérience de France Télécom pour les composants électroniques et mécaniques.
- Le Guide FIDES 2008 : guide de fiabilité prévisionnelle construit sur la base des recueils précédemment cités à partir du retour d'expérience d'un consortium d'industriels français. Aujourd'hui, ce recueil a été transformé en une norme dénommée UTE C 80-811.

Mais malgré la richesse de leur contenu, les données fournies s'avèrent insuffisantes dans le cadre d'applications spécifiques. C'est pour cela que l'on assiste au développement de bases de données internes.

Pour définir le taux de défaillance, il convient d'avoir un retour d'expérience construit sur :

- Une durée suffisante d'exploration du composant.
- Des conditions d'environnement variées (climatisation, vibration, température, CEM).
- Des sollicitations suffisantes du composant.

III.4 Mise en œuvre de la méthode de l'arbre de défaillance

Si la maîtrise conceptuelle des arbres de défaillance est désormais acquise, nous nous trouvons encore confrontés à des problèmes lors de leur exploitation informatique. Ces difficultés apparaissent lors du traitement qualitatif de l'AdD pour la recherche des ensembles minimaux. L'inconvénient majeur induit par ce genre de technique réside dans l'espace mémoire requis au stockage de tous ces ensembles et au temps d'exécution qui peut s'avérer prohibitif. Ceci se produit notamment, lors du traitement d'arbres de grande taille.

III.4 .1 Cas des systèmes cohérents

III.4 .1 .1 Algorithme de MOCUS [FUS 72]

Il a été développé en 1972 par Fussell et Vesely pour le traitement qualitatif des arbres de défaillance cohérents. Il procède en deux étapes. La première étape consiste à construire la fonction de structure de l'AdD (Annexe 3). Le développement de cette fonction fournit l'ensemble des coupes. La seconde étape est une étape de réduction dont le but est d'éliminer les coupes redondantes, pour ne garder que les coupes minimales. Deux règles sont utilisées, il s'agit des règles de l'algèbre de Boole :

$$nX = X \qquad \text{et} \qquad X + XY = X$$

Cette étape de réduction est celle qui consomme le plus de temps à cause du nombre de comparaisons nécessaires. Pour réduire n coupes, il faut réaliser n (n-1) / 2 comparaisons.

Des versions améliorées de l'algorithme MOCUS ont été proposées, nous citerons : l'algorithme de Benjiamin et Al.[BEN 76], l'algorithme FATRAM de Rasmuson et Marshall [RAS 78] et celui de Limnios et Ziani [LIM 86].

III.4 .1 .2 Algorithme de Bengiamin et Al.[BEN 76]

La philosophie et la technique de cet algorithme sont de réduire le temps consacré aux calculs par élimination des coupes minimales fictives générées en cours de traitement aussi tôt que possible.

L'algorithme est basé sur trois étapes principales. D'abord l'AdD est réduit par l'élimination des événements répétés qui sont des entrées pour les opérateurs logiques OU. Ensuite, avec l'application de l'algorithme MOCUS à l'AdD réduit, on obtient un ensemble qui constitue un premier groupe. Enfin, ce groupe est traité pour obtenir un autre ensemble qui va constituer un deuxième groupe. L'ensemble des coupes de premier et de deuxième groupe obtenu constitue l'ensemble des coupes minimales de l'AdD initial.

III.4 .1 .3 Algorithme FATRAM [RAS 78]

L'algorithme FATRAM est basé sur quatre étapes principales. D'abord la suppression de tous les operateurs OU primaires en les remplaçant par les événements de base. Ensuite, on cherche les coupes minimales (avec l'application de l'algorithme MOCUS par exemple). Puis on traite les événements répétés qui sont des entrées des operateurs OU. Enfin, dans la dernière étape, on passe au traitement des opérateurs OU restants. Tous les ensembles obtenus sont des coupes minimales.

III.4 .1 .4 Algorithme de Limnios et Ziani [LIM 86]

Dans le but de réduire le nombre de comparaisons à faire dans l'étape de réduction et ainsi diminuer le temps d'exécution, l'idée de cet algorithme consiste à diviser l'ensemble des coupes obtenues en deux classes, l'une K1 constituée des coupes contenant des évènements répétitifs et l'autre K2 constituée des coupes n'en contenant pas. La réduction ne portera alors que sur les coupes de la première classe. La classe K2 est irréductible.

L'algorithme FATRAM, aussi bien que l'algorithme de Bengiamin et Al, apportent une amélioration par rapport à l'algorithme MOCUS que dans le cas où l'AdD possède des opérateurs OU primaires [LIM 05].

III.4 .2 Applications et comparaison

III.4 .2 .1 Applications de l'algorithme de Limnios et Ziani

Afin d'illustrer l'algorithme de Limnios et Ziani, nous considérons l'arbre de défaillance de la figure III.7. Il contient un événement répété 6. Toutes les étapes de l'algorithme sont illustrées comme suit :

- L'application de l'algorithme MOCUS génère les 9 coupes suivantes :

{1}, {2}, {3}, {6}, {8}, {4,6}, {4,7}, {5,7}, {5,6}.

- Pour réduire ces coupes, il faut faire 36 comparaisons. En se limitant uniquement aux coupes contenant l'événement répété 6, c'est-à-dire les {6}, {4,6}, {5,6}

(Les autres coupes sont déjà minimales), nous avons 3 comparaisons à effectuer.

- Les coupes minimales sont :

{1}, {2}, {3}, {6}, {8}, {4,7}, {5,7}.

L'application de l'algorithme MOCUS pour l'arbre de défaillance de la figure III.8 nous donne 36 coupes minimales, 20 appartiennent à K1 et 16 à K2, le nombre minimal de comparaisons nécessitant une réduction est de 190.

III.4 .2 .2 Comparaison de l'algorithme de Limnios et Ziani avec FATRAM

L'application de l'algorithme FATRAM pour l'arbre de défaillance de la figure (III.7) nous donne :

Groupe1 :{ 1}, {2}, {G4 G5}, {3}, {G7}

La réduction nécessite un nombre maximal de comparaisons égal à 10

groupe2 : {1}, {2}, {G4,G5}, {3}, {G7}, {6,G6}, {6}

La réduction nécessite un nombre maximal de comparaisons égal à 21 (6 si on procède groupe par groupe)

groupe3 : {1}, {2}, {3}, {6}, {8}, {4,7}, {5,7} sont les coupes minimales

Pour l'arbre de défaillance de la figure III.8, FATRAM procède exactement comme MOCUS. La réduction est effectuée avec un nombre maximal de comparaison égale à 630.

III.4 .2 .3 Comparaison de l'algorithme de Limnios et Ziani avec Bengiamin et Al.

L'application de l'algorithme Bengiamin et Al. pour l'arbre de défaillance de la figure III.7 impose que la réduction soit effectuée avec un nombre maximal de comparaisons égal à 3. Comme l'algorithme FATRAM, cet algorithme procède comme MOCUS pour traiter l'arbre de défaillance de la figure III.8.

A partir de l'AdD de la figure III.8, nous concluons que l'application de ces deux algorithmes dépend des arbres de défaillance utilisés.

FATRAM s'applique si l'arbre de défaillance a au moins une porte OU qui a comme entrées des événements seulement.

Le second algorithme ne s'applique pas si les événements répétés de l'arbre de défaillances sont tous des entrées des portes ET.

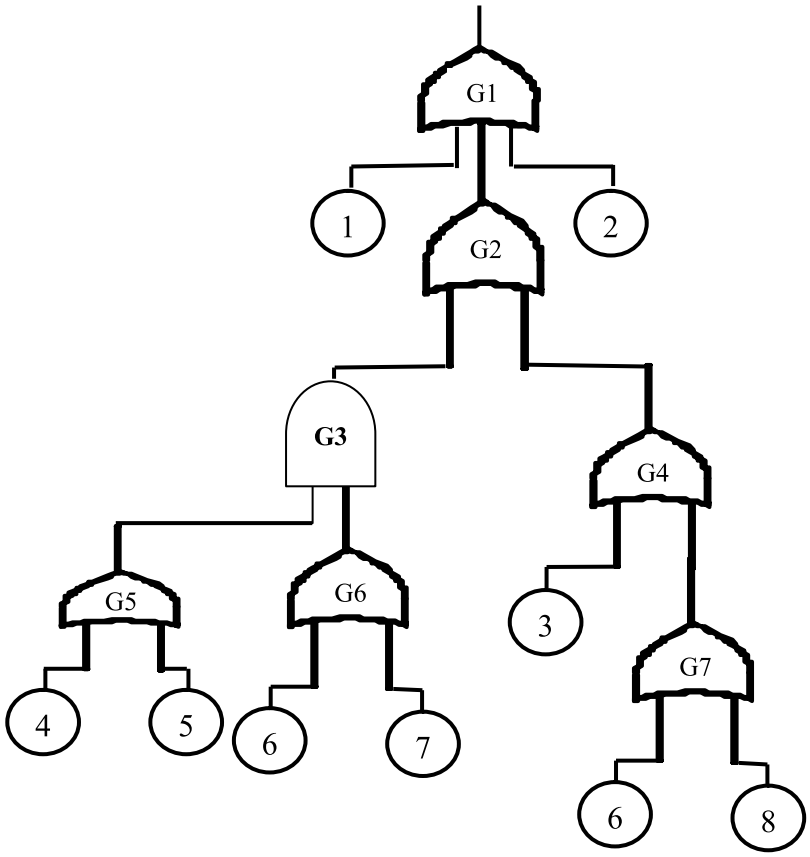


Figure III.7 Arbre de défaillance 1

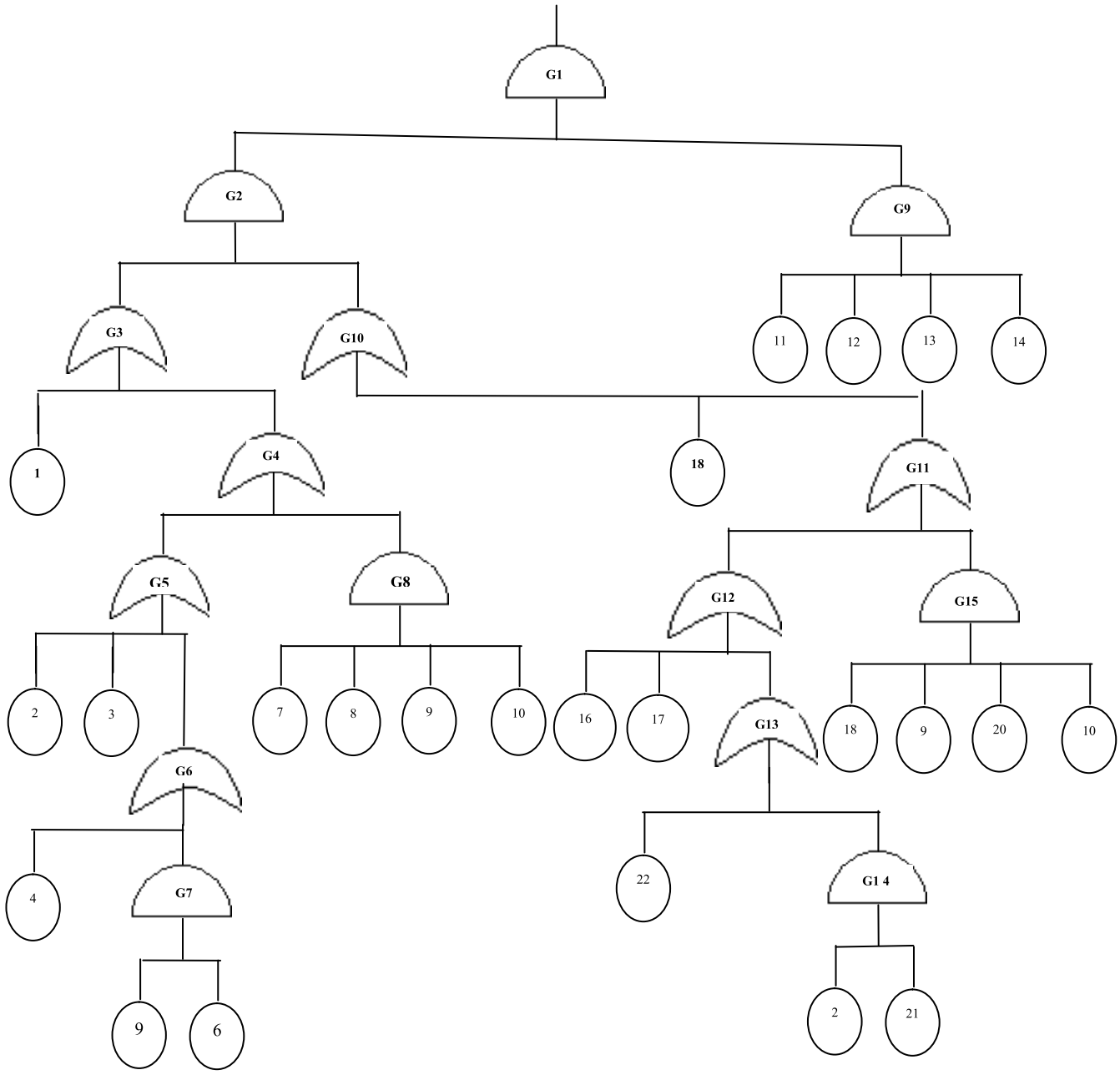


Figure III.8 Arbre de défaillance 2

L'algorithme de Limnios et Ziani peut être combiné avec l'algorithme FATRAM ou/et Bengiamin et Al. pour donner de meilleurs résultats.

CHAPITRE IV

Algorithme Kumamoto- Henley

IV.1 Préambule

Les systèmes non cohérents sont en pratique rencontrés dans les systèmes comprenant des boucles de régulation. Leur traitement à l'aide des arbres de défaillance présente beaucoup plus de difficulté que celui des systèmes cohérents, à cause des implicants de consensus.

Un algorithme connu dans le domaine des arbres de défaillance non cohérents est celui de Kumamoto et Henley [KUM 78]. Dans ce chapitre, nous donnons une description de cet algorithme et nous présentons quelques améliorations que nous pouvons lui apporter, afin de réduire la place mémoire et augmenter sa vitesse d'exécution.

IV.2 Principe de l'algorithme de Kumamoto et Henley

L'algorithme est basé sur le principe suivant : une condition nécessaire et suffisante pour qu'une fonction de structure Ψ admette un produit de littéraux $P = L_1\ L_2\ \ldots\ldots\ L_n$ comme implicant est que tous les produits de son dual Ψ^* s'annulent quand tous les littéraux du produit P sont portés à zéro. Sa mise en oeuvre comprend trois parties comme illustré par l'organigramme donné en figure IV.1.

1ére partie : Cette étape consiste à construire l'arbre dual et à déterminer les expressions locales des différentes portes ainsi que leur domaine. Pour ce faire, nous procédons comme suit :

- Remplacer dans l'arbre de défaillance les portes NAND, NOR, EOR par leur équivalent en portes AND, OR, NON, donné en annexe 2-a.
- Ramener l'opérateur NON au dernier niveau de l'AdD où il sera pris en compte de manière implicite en remplaçant la variable par sa variable complémentaire comme indiqué en annexe 2-b.
- Construire l'arbre dual. Celui-ci est obtenu en remplaçant les portes AND par des portes OR et les portes OR par des portes AND. L'arbre de défaillance de la figure IV.6 est obtenu.
- Obtenir les expressions locales des portes de l'arbre dual en décomposant chaque porte en ses entrées (Annexe 3).
- Obtenir le domaine P[Gi] de chaque porte Gi en déterminant tous les littéraux qui s'y rapportent.

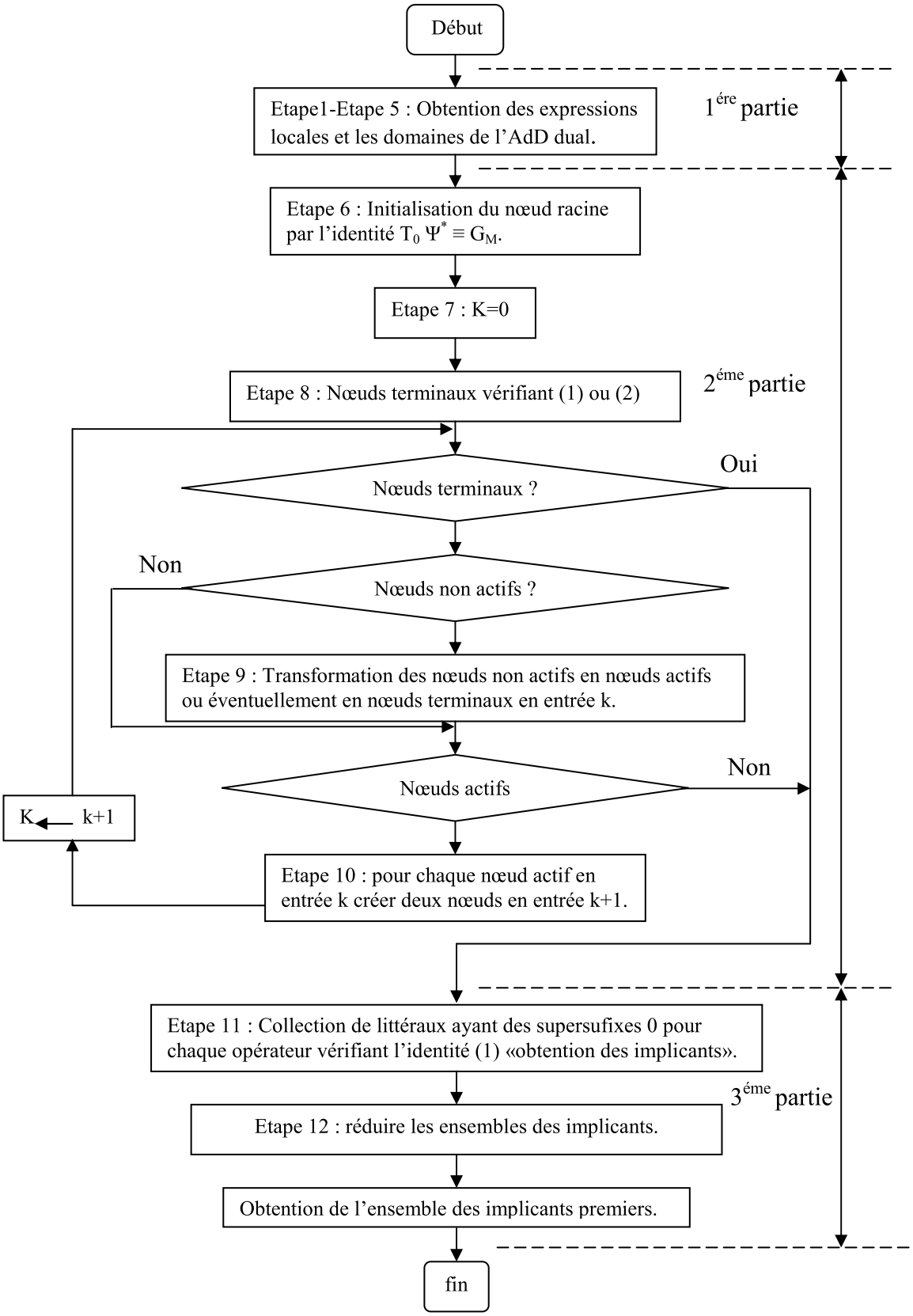


Figure IV.1 Organigramme de l'algorithme de Kumamoto-Henley.

2éme partie : Cette étape consiste à rechercher dans l'arbre dual les opérateurs L_k^{ik} ……. L_1^{i1} qui satisfont l'une des identités suivantes :

$$\mathbf{T_k\Psi^* = L_k^{ik} \ldots\ldots\ldots\ldots L_1^{i1}\ \Psi^* = 0} \qquad (1)$$

ou $$\mathbf{T_k\Psi^* = L_k^{ik} \ldots\ldots\ldots\ldots L_1^{i1}\ \Psi^* = 1} \qquad (2)$$

Avec

Ψ^* : la fonction booléenne de l'événement sommet de l'AdD dual.

i_1, ……. i_k : des entiers prenant la valeur 0 ou 1.

L_i : un littéral désignant un événement de base x_i ou son événement complémentaire $\overline{x_i}$.

Lorsqu'un littéral L apparaît dans l'expression booléenne de Ψ^*, nous procédons à une classification dichotomique engendrant ainsi deux classes d'implicants L^1 et $L^{0.}$ La classe L^1 qui représente la classe d'implicants ne contenant pas L est obtenue en remplaçant dans Ψ^* le littéral L par un « 1 » (L=1). La classe L^0 qui représente la classe d'implicants contenant L est obtenue en remplaçant dans Ψ^* le littéral L par un « 0 » (L=0) et le littéral $\overline{L}$ par un « 1 » ($\overline{L}$ =1).

Chaque nouvelle sous-classe subit à son tour une classification dichotomique et le traitement se poursuit jusqu'à arriver à un nœud terminal vérifiant l'identité (1) ou (2). On dit alors que la classe est irréductible. Cette procédure est décrite par l'arbre binaire de la figure IV.3 dans laquelle le nœud terminal est représenté par un cercle noir. Les noeuds non terminaux sont classés en 2 catégories, à savoir les nœuds actifs contenant des littéraux et les nœuds non actifs ne contenant que des portes logiques.

La décomposition pour la recherche des implicants utilise un parcours en largeur (Figure IV.2)

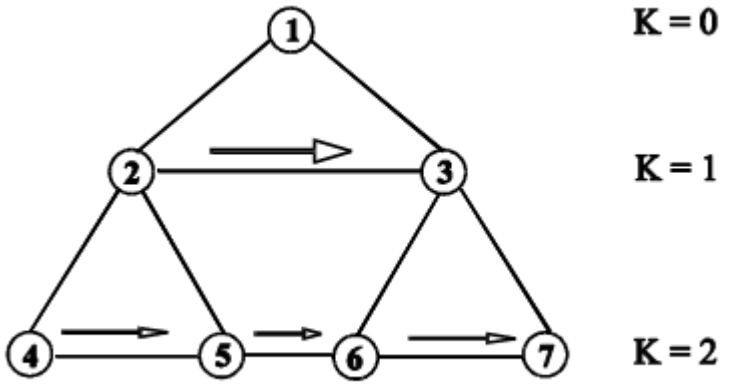
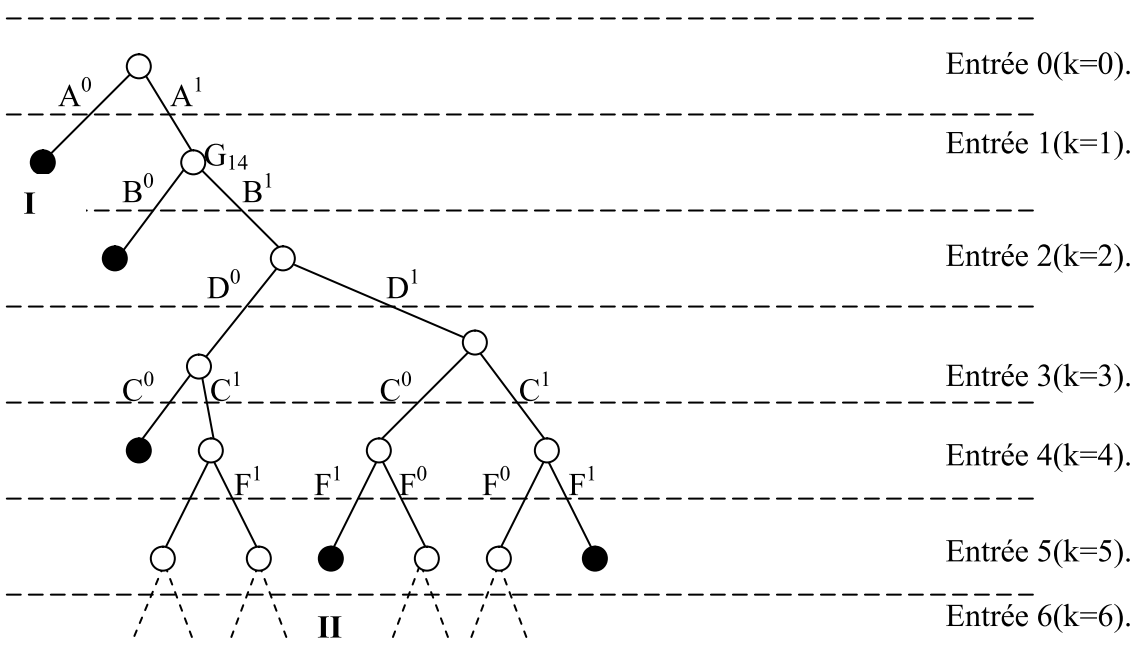


Figure* IV.*2 Parcours en largeur

La classe vérifiant l'identité (1) signifie que l'ensemble des littéraux de cette classe constitue un implicant pour la fonction Ψ et celui ci est un implicant premier dans sa classe. La classe vérifiant l'identité (2) signifie que l'ensemble des littéraux de cette classe ne constitue pas un implicant pour la fonction Ψ. Ainsi par exemple sur la figure IV.3, les nœuds terminaux I et II vérifient respectivement :

$A^0\Psi^* = 0$ et $F^1C^0D^1B^1A^1\Psi^* = 1$

Ce qui signifie que le littéral A est un implicant premier.



● : Nœud terminal.
○ : Nœud non terminal.

$A^i, B^i, \ldots$: Evénements de base.

Figure IV.3 Arbre de traitement binaire

3éme partie : Détermination des implicants premiers

Pour ce faire, nous regroupons tous les littéraux L_j ayant l'exposant $i_j=0$ pour chaque opérateur qui satisfait l'identité (1). Chaque groupe de littéraux obtenu de cette manière est un implicant. Nous procédons ensuite à une réduction afin d'éliminer les implicants redondants. Un implicant P_i est éliminé s'il existe un implicant $P_j \neq P_i$ tel que $P_j \subset P_i$. La liste des implicants restants constitue l'ensemble des implicants premiers de l'arbre de défaillance étudié.

IV.3 Améliorations

Le type de parcours en largeur utilisé à l'étape 2 impose de traiter 2^k nœuds à l'étape k. De plus il est nécessaire d'avoir comme information les 2^{k-1} nœuds précédents. Pour des AdD dans lesquels les nœuds terminaux n'apparaissent qu'à une certaine profondeur de l'arbre, le nombre de nœuds à examiner et à conserver peut devenir élevé et peut ainsi induire des problèmes de capacité mémoire.

Afin de réduire ce problème de capacité mémoire, nous proposons un autre type de parcours que nous appellerons parcours droite gauche (Figure IV.4). Ainsi, nous traitons le nœud racine puis tous les nœuds se trouvant dans la branche de droite jusqu'à aboutir à un nœud terminal. Nous remontons ensuite jusqu'au premier nœud ayant un nœud adjacent, et en partant de ce dernier nous réalisons la même opération. Nous recommençons cette procédure jusqu'à traitement de tous les nœuds.

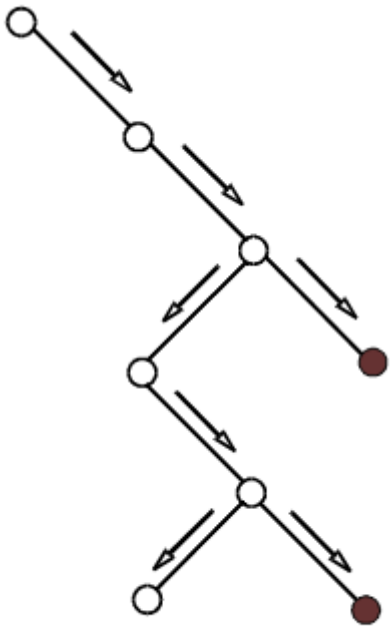


Figure IV.4 Parcours droite gauche

L'intérêt principal de ce type de parcours est l'élimination de certains implicants redondants avant l'étape de réduction. Ainsi par exemple, dans le cas de figure IV.5 dans laquelle la présence de littéraux est notée P et l'absence de littéraux A, pour tous les chemins du type AAPAPAAA ayant abouti à un nœud terminal M tel que $T_k\Psi^* = 0$, nous pouvons remonter dans l'arbre jusqu'au premier P rencontré : le nœud S. Tous les nœuds situés à la gauche de S sont : soit des nœuds aboutissant à la relation $T_k\Psi^* = 1$, donc sans intérêt, soit des nœuds aboutissant à la relation $T_k\Psi^* = 0$, ce qui donne

des implicants redondants à ceux obtenus au nœud M. Il est donc inutile de traiter ces nœuds. Dans la figure IV.5 ces nœuds se trouvent dans la partie hachurée.

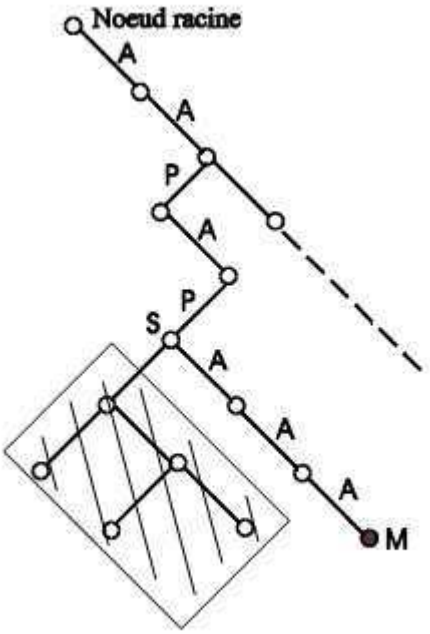


Figure IV.5 Le parcours droite gauche

IV.3 Applications

Afin d'évaluer l'efficacité de l'amélioration, nous traitons deux exemples d'arbre de défaillance avec les deux versions.

1er Exemple : AdD 1 de la figure IV.6

La logique de cet AdD contient 11 portes et 9 évènements élémentaires. Son traitement par notre algorithme a conduit aux résultats suivants :

- ✓ Nombre de nœuds étudiés = 77
- ✓ Nombre d'implicants obtenus = 8
- ✓ Nombre d'implicants premiers obtenus = 8
- ✓ Liste des implicants premiers:

  $\{1\}, \{2\}, \{3,4\}, \{4,5,\overline{6}\}, \{4,\overline{6},7\}, \{4,\overline{6},8\}, \{4,\overline{6},9\}, \{3,\overline{5},6,\overline{7},\overline{8},\overline{9}\}$

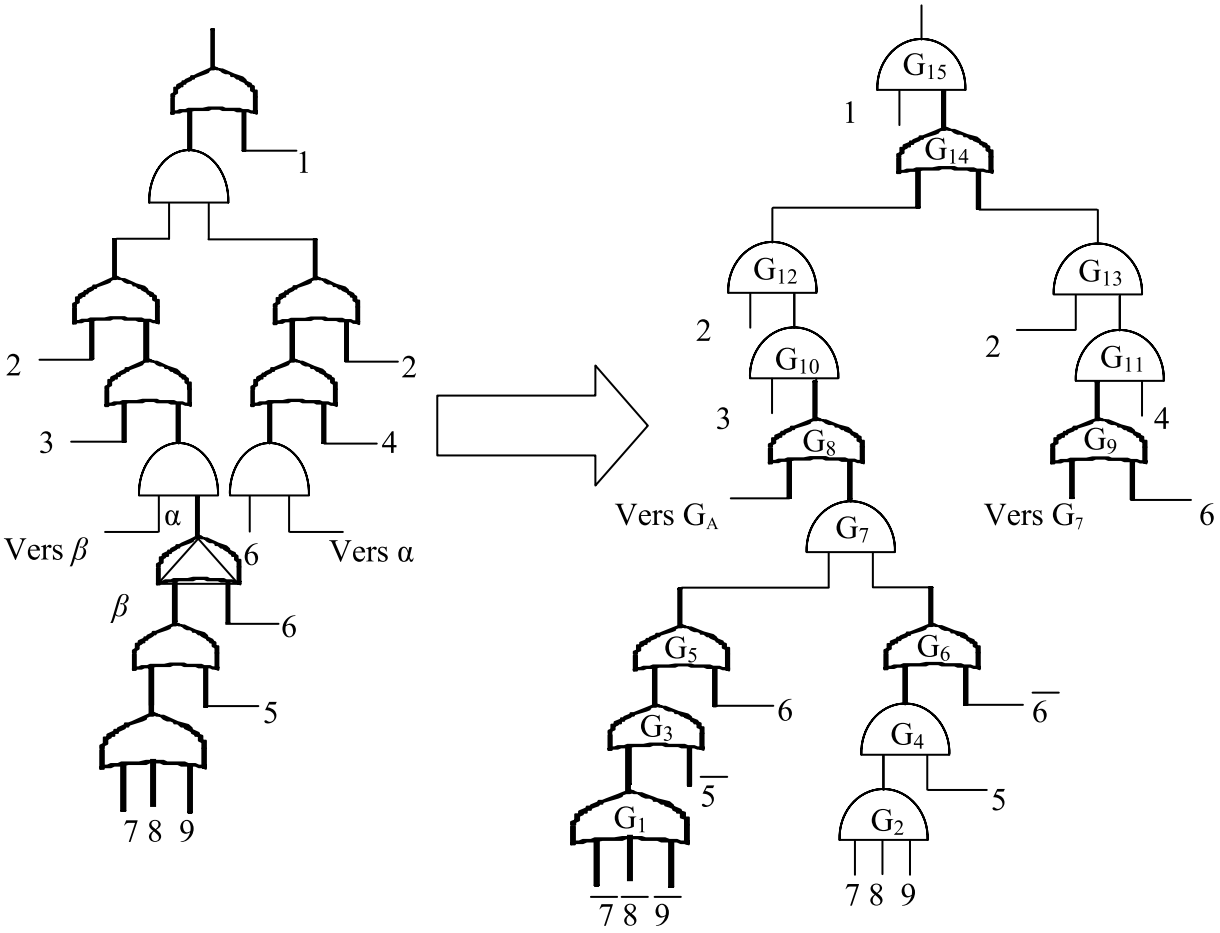


Figure IV.6 Arbre de défaillance 1 et son arbre dual

Le temps de calcul en centième de seconde est égal : 5

Notons que pour cet exemple tous les implicants obtenus sont premiers. L'étape de réduction n'est donc pas nécessaire.

Le traitement à l'aide de l'algorithme de Kumamoto donne les mêmes résultats

2éme Exemple : AdD 2 de la figure IV.7

La logique de l'AdD contient 16 portes et 19 évènements élémentaires. Son traitement par notre algorithme a conduit aux résultats suivants :

- ✓ Nombre de nœuds étudiés = 2447
- ✓ Nombre d'implicants obtenus = 165
- ✓ Nombre d'implicants premiers obtenus = 69

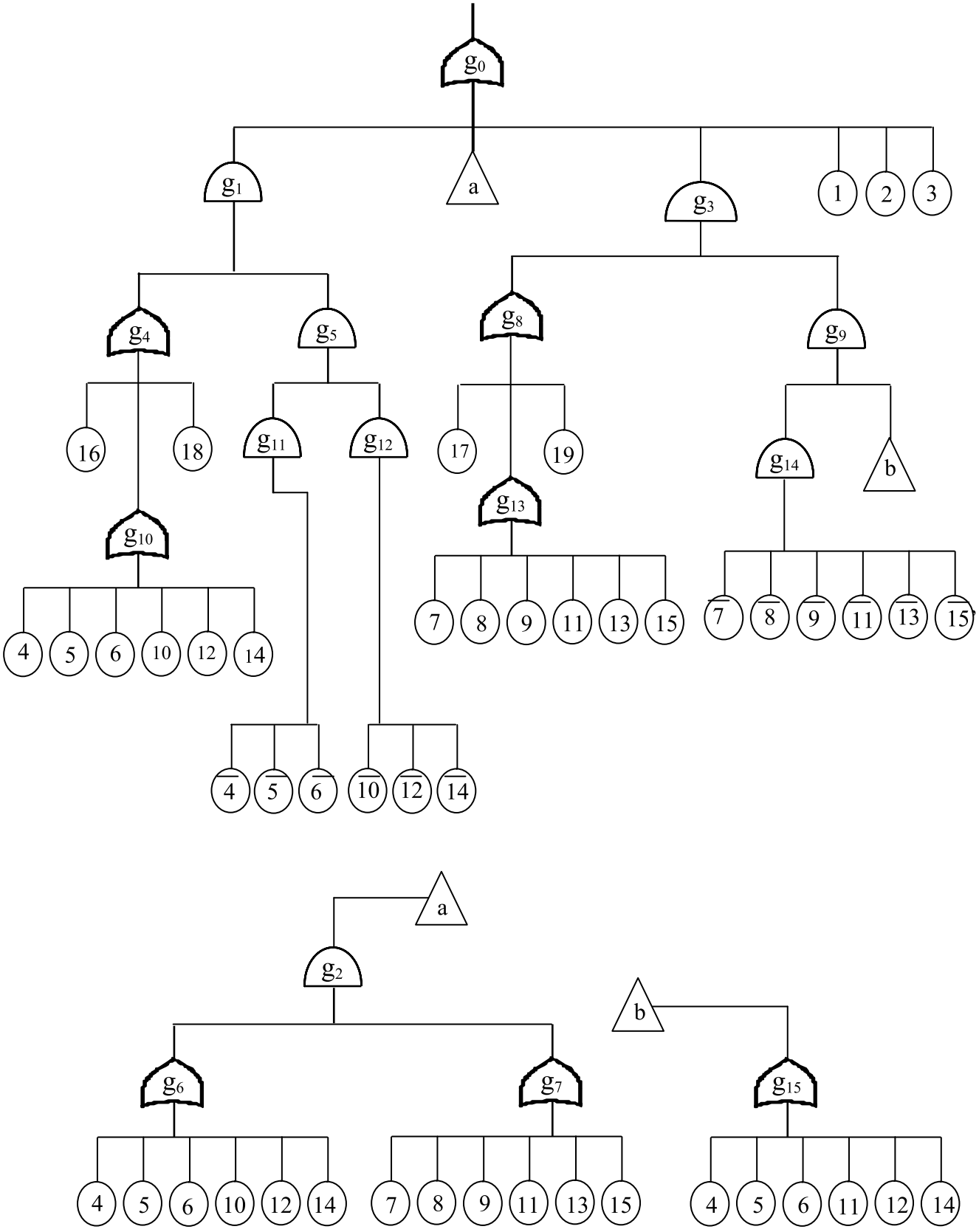


Figure* IV.*7* *Arbre de défaillance 2

Pour éliminer les implicants redondants, nous passons par une phase de réduction qui nécessite (165 *164) / 2 = 13530 comparaisons.

Après réduction nous obtenons 69 implicants premiers :

- 3 implicants premiers d'ordre 1 :
  {1}, {2}, {3}
- 64 implicants premiers d'ordre 2 :
  {14,17}, {14,19}, {14,15}, {14,13}, {14,11}, {14,9}, {14,8}, {14,7}, {12,17}, {12,19}, {12,15}, {12,13}, {12,11}, {12,9}, {12,8}, {12,7}, {10,17}, {10,19}, {10,15}, {10,13}, {10,11}, {10,9}, {10,8}, {10,7}, {6,17}, {6,19}, {6,15}, { 6,13}, {6,11}, {6,9}, {6,8}, {6,7}, {5,17}, {5,19}, {5,15}, {5,13}, {5,11}, {5,9}, {5,8}, {5,7}, {4,17}, {4,19}, {4,15}, {4,13}, {4,11}, {4,9}, {4,8}, {4,7}, {16,17}, {16,19}, {16,15}, {16,13}, {16,11}, {16,9}, {16,8}, {16,7}, {18,17}, {18,19}, {18,15}, {18,13}, {18,11}, {18,9}, {18,8}, {18,7}
- 2 implicants premiers d'ordre 7 :
  $\{16,\bar{4},\bar{5},\bar{6},\overline{14},\overline{12},\overline{10}\}$, $\{18,\bar{4},\bar{5},\bar{6},\overline{14},\overline{12},\overline{10}\}$

Le temps de calcul en centième de seconde est égal : 19

Son traitement à l'aide de l'algorithme de Kumamoto donne les résultats suivants :

- ✓ Nombre de nœuds étudiés = 5003
- ✓ Nombre d'implicants obtenus = 1557
- ✓ Nombre d'implicants premiers obtenus = 69

La phase de réduction nécessite dans ce cas (1557*1556) / 2 = 1211346 comparaisons.

IV.4 Discussion des résultats

L'algorithme que nous avons décrit précédemment ainsi que celui de Kumamoto ont été implémentés en langage C sur un micro-ordinateur de type « Core (TM) 2duo CPU 2x2,10Ghz », La comparaison des résultats obtenus par ces deux algorithmes montre que dans l'exemple de l'AdD 1 de la figure IV.6, notre algorithme donne les mêmes résultats que celui de Kumamoto car il n'y a pas de phase de réduction. Néanmoins cet exemple montre que l'algorithme fonctionne correctement.

Dans l'exemple de l'AdD 2 de la figure IV.7 nous constatons que notre algorithme donne de meilleurs résultats. En effet d'une part, le nombre de nœuds étudiés est réduit de moitié et le nombre d'implicants obtenus avant réduction est réduit d'un facteur 10. Ce qui permet un gain appréciable en espace mémoire. D'autre part, le temps d'exécution a ainsi pu être réduit de moitié.

CONCLUSION

Conclusion et perspectives

Dans ce mémoire, nous avons dans un premier temps présenté la méthodologie générale d'une étude de sûreté de fonctionnement, et nous avons décrit en détail la modélisation par arbre de défaillance qui est une technique majeure en sûreté des systèmes. Ensuite nous nous sommes intéressés à l'algorithme de Kumamoto et Henley qui constitue un standard dans le domaine des arbres de défaillance non cohérents. Nous l'avons implémenté dans sa version originale et nous lui avons apporté des améliorations. Le traitement de deux exemples d'arbre de défaillance avec les deux versions nous a permis de montrer que notre algorithme apporte un gain appréciable en espace mémoire et en temps d'exécution.

Tous les algorithmes que nous avons présentés dans ce travail sont basés sur la technique des arbres de défaillance. Ces dernières années, une nouvelle algorithmique concernant ces derniers a été proposée, elle est désignée sous le vocable de diagramme de décision binaire (DDB) et se base sur une sorte de réduction de l'arbre de factorisation de Shannon mise au point dans le traitement des fonctions booléennes. A l'aide de cette nouvelle technique, une algorithmique complète (analyse logique, évaluation probabiliste, calcul d'importance) est proposée dans la littérature. Il semblerait que ces algorithmes soient plus performants. Ils ont permis le traitement de cas tests difficiles, en des temps extrêmement courts (quelques secondes) en comparaison des algorithmes classiques qui nécessitent plusieurs heures.

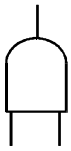
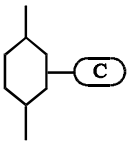
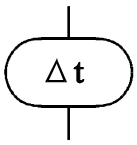
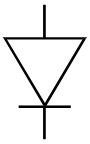
Il faut cependant noter que ces algorithmes n'ont pas changé la nature du problème qui reste toujours NP-complet. Des cas difficiles à résoudre même par ces algorithmes peuvent se présenter. Le problème concerne la taille du DDB. Différents ordres de variables donnent des tailles différentes et dans certains cas cette différence peut être très grande, ce qui implique un traitement inégalé. Dans des applications de grande taille, nous avons intérêt à bien choisir l'ordre des variables. Le problème de choix de l'ordre des variables est un problème qui reste ouvert pour l'instant. Ceci justifie l'effort de recherche d'algorithmes toujours plus performants.

ANNEXES

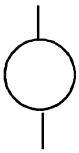
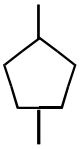
Annexe 1

Symboles utilisés dans les arbres de défaillance

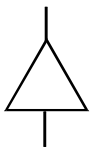
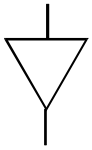
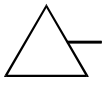
Opérateurs

Symbole	Nom	Signification
	OU	Sortie générée si au moins une des entrées existe
	ET	Sortie générée si toutes les entrées existent
	OU EXCLUSIF	Sortie générée si une entrée et une seule existe
	SI	Sortie générée si l'entrée existe et la condition C est vérifiée
	DELAI	Sortie générée avec un retard Δt sur l'entrée qui doit être présente pendant Δt
	NON	Sortie générée lorsque l'entrée ne se produit pas
	Porte COMBINAISON m/n (ici 2/4)	Sortie générée si m des n évènements d'entrée sont présents (ici 2/4)

Evénements

Symbole	Nom	Signification
	Rectangle	Evènement sommet ou intermédiaire
	Cercle	Evènement de base élémentaire
	Losange	Evènement de base non élémentaire mais dont les causes ne sont pas développées
	Maison	Evènement de base considéré comme normal

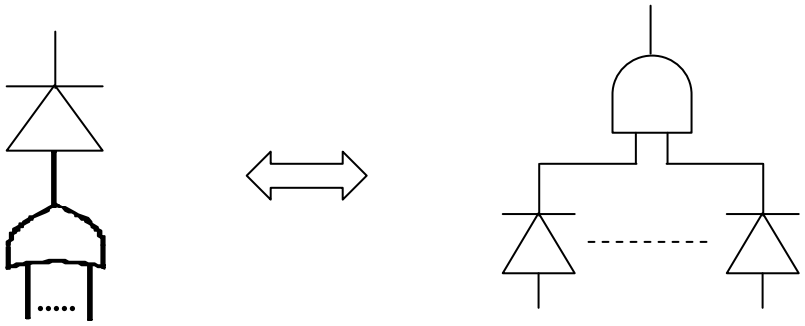
Triangles de transfert

Symbole	Nom	Signification
	Transfert identique	La partie de l'AdD qui suit n'est pas indiquée car identique à la partie repérée par △
	Transfert semblable	La partie de l'AdD qui suit n'est pas indiquée car semblable à la partie repérée par △
	Identification du transfert	Indique un sous arbre identique ou semblable

Annexe 2

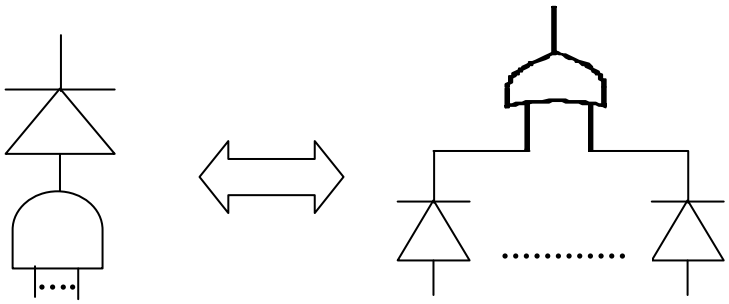
a. Transformations équivalentes

Opérateur NOR (a) et transformation équivalente (b)



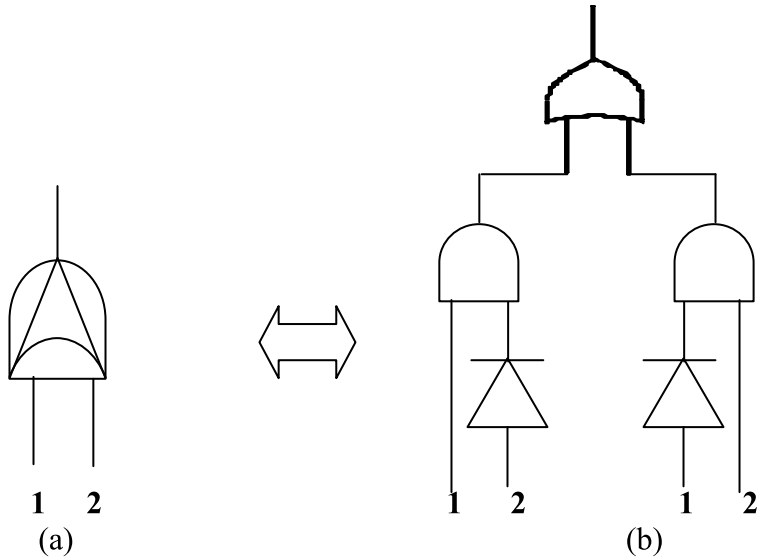
(a) (b)

Opérateur NAND (a) et transformation équivalente (b)

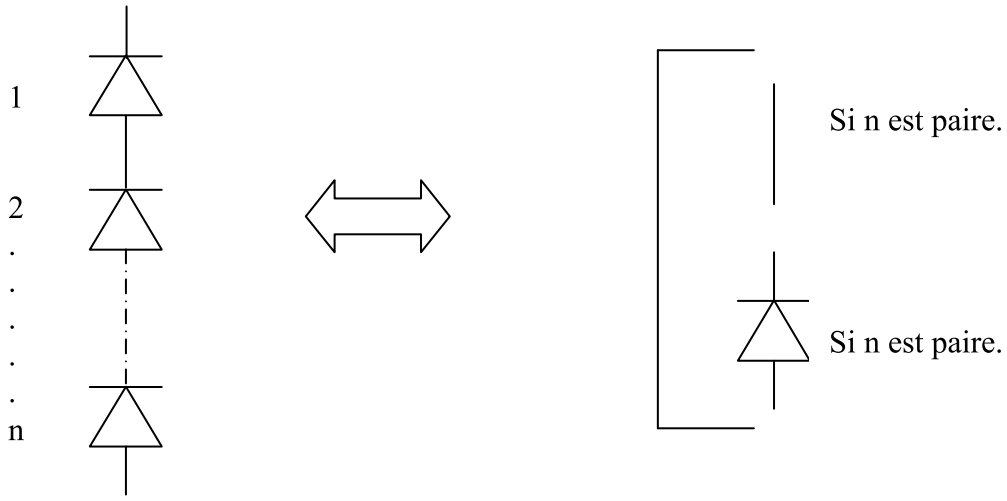


(a) (b)

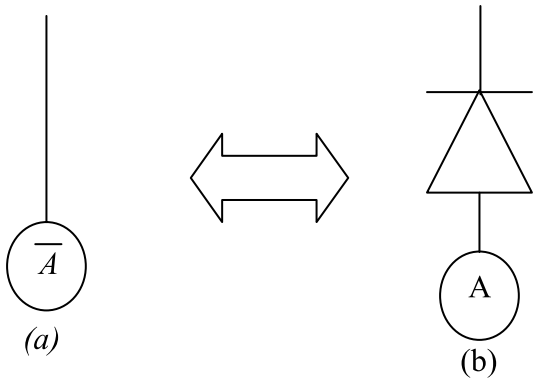
Opérateur XOR «OU exclusif» (a) et transformation équivalente (b)



b. Opérateurs NOT réduits



Evénement complémenté (a) et transformation équivalente (b)



Annexe 3

Equivalence des événements avec les variables indicatrices :

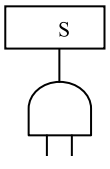
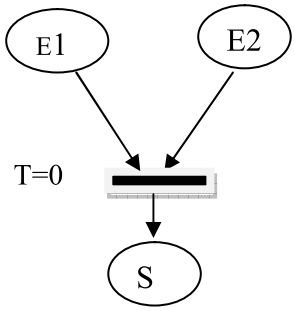
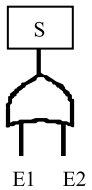
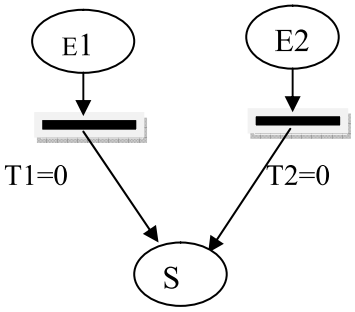
Portes logiques	Ensembles et/ou d'événements.	Variables indicatrices
OU	U	+
ET	∩	•

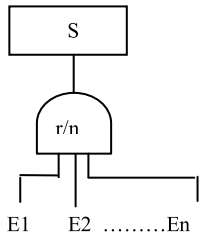
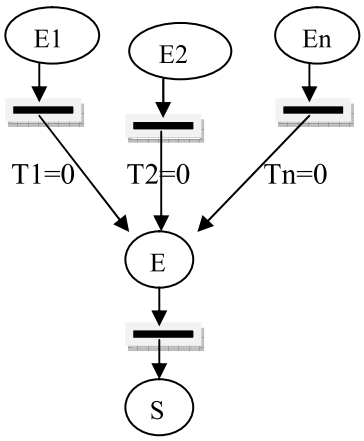
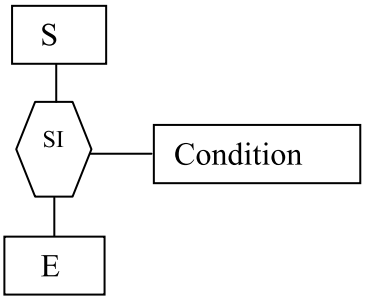
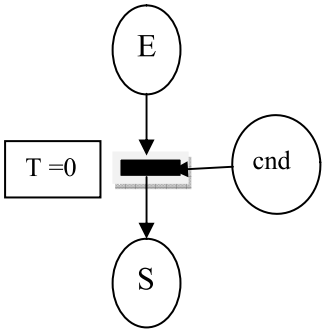
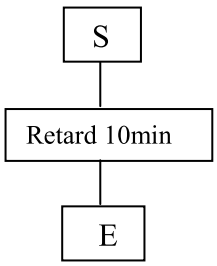
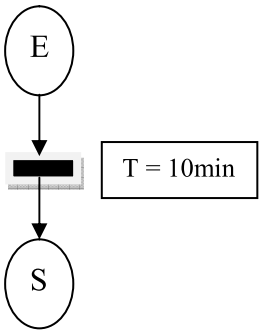
Pour les événements A et B, notons X_A et X_B respectivement leurs fonctions indicatrice (variables booléennes). Nous avons les équivalences suivantes :

$E = A \cap B \Leftrightarrow X_E = X_A X_B$.
$E = A \cup B \Leftrightarrow X_E = X_A + X_B$.

Annexe 4

Modélisation des portes logiques de l'AdD par leurs RdP associés :

Composant AdD	Assignation en RdP	Causalité
Porte ET : 		L'événement de sortie se produit si chacune des places en entrée (E1, E2) contienne au moins un jeton, condition nécessaire.
Porte OU : 		L'événement de sortie se produit si l'une au moins des places en entrée contienne un jeton, condition écessaire.

Porte combinaison r/n : 		L'événement de sortie se produit si "r" places contiennent chacune au moins un jeton, condition nécessaire. * r représente le poids de l'arc dirigé vers S. * Ei est une place intermédiaire.
Porte SI : 		L'événement de sortie est réalisé si la condition est vraie et si l'événement E est produit. * Cnd : est une place indiquant l'état vrai ou faux de la condition.
Porte délai : 		L'événement de sortie est réalisé avec un retard donné (10min) après l'apparition de l'événement d'entrée ; (Rdp temporisé).

BIBLIOGRAPHIE

Bibliographie

[AKE 78] S.B. Akers, Binary decision diagrams, IEEE Trans. On Computers, vol. C-27, N° 6, 1978.

[BEN 76] N.N. Benjiamin, B. A. Bowen, K.F. Schen, An efficient algorithm for reducing the complexity of computation in fault tree analysis, IEEE Trans. Nucl. Science, Vol. NS-23, N° 5, Octobre 1976.

[BEN 07] G. Benattar, La vérification de propriétés tctl sur les réseaux de Pétri temporels à chronomètres en temps discret, Mémoire de Master, Université de Nantes, septembre 2007.

[BEN 07] A. Ben Amor, M. Trabelsi Contribution à l'Analyse Hybride en Sûreté de Fonctionnement (SdF), Fault Tree Petri Net (FTPN), CPI' 2007, Rabat, Maroc, 2007**.**

[BIR 69] Z.W. Birnbaum, On the importance of different components in a multi component system, Multivariate Analysis II, Edition Pr. Krishnaiaf, New York, Academic Press, 1969.

[BOU 03] M. Bouisson et J. L. Bon, A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes, Reliability Engineering and System Safety, pp. 149-163, novembre 2003.

[BRY 92] R.E. Bryant, Symbolic Boolean manipulation with ordered binary decision diagrams, ACM Computing Surveys, vol. 24, N° 3, September 1992.

[CAR 99] J.A. Carrasco, V. Sune , An algorithm to find minimal cuts of coherent fault tree with even-classes, using a decision tree, IEEE Trans. On Reliability, vol. 48, march 1999.

[COR 75] M. Corazza, Techniques mathématiques de la fiabilité prévisionnelle, Editions Cepadues, 1975.

[COU 94] O. Coudert, J.C. Madre, MetaPrime: An Interactive Fault-Tree Analyser, IEEE Trans. On Reliability, vol. 43, march 1994.

[FUL 88] R.R. Fullwood et R.E. Hall, Probabilistic Risk Assessment in the Nuclear Power, Industry Fundamentals and Applications, Editions Pergamon Press, 1988.

[FUS 72] J.B. Fussell, W.E. Vesely, A new methodology for obtaining cut sets for fault trees, Trans. Amer. Nucl. Soc., Vol. 15, N° 1, 1972.

[FUS 75] J.B. FusselL, How to hand calculate system reliability characteristics, IEEE Trans. On Reliability, vol. R-2, N° 3, august 1975.

[HAS 80] M. Hassan, Méthodes automatiques de construction et de calcul de l'arbre des défauts d'un système complexe, Thèse de Dr Ingénieur, Université de Compiègne, avril 1980.

[JEN 98] W. Jeng, G.R. Liang, Reliable automated manufacturing system design based on SMT framework, Elsevier Computers in Industry, pp. 121-147, 1998.

[KHA 01] S. Khalfaoui, E. Guilhem, H. Demmou et R. Valette, Modeling critical mechatronic systems with Petri Nets and feared scenarios derivation, 5th Workshop on Electronics, Control, Modeling, Measurement and Signals, Toulouse, France, mai 2001.

[KHA 02] S. Khalfaoui, H. Demmou, N. Rivière, E. Guilhem, A method for deriving critical scenarios from mechatronic systems, Journal Européen des Systèmes Automatisés, vol 36, N° 7, 2002.

[KHA 03] S**.** Khalfaoui, Méthode de recherche des scénarios redoutés pour l'évaluation de la sûreté de fonctionnement des systèmes mécatroniques du monde automobile. Thèse de Doctorat, Institut National Polytechnique, Toulouse, 2003.

[KEH 04] C. Kehren, C. Seguin, P. Bieber, C. Castel, Analyse des exigences de sûreté d'un système électrique par model-checking, Actes du Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement, Bourges, France, Octobre 2004.

[KUM 78] H**.** Kumamoto, E.J. Henley, Top Down algorithm for obtaining prime implicant sets of non-coherent fault trees. IEEE Trans. On Reliability, vol R 27, N° 4, 1978.

[LAM 75] H.E. Lambert, Fault trees for decision making in system safety and availability, Thèse de Doctorat de l'Université de la Californie, Berkeley 1975.

[LAP 96] J.C. Laprie, Guide de la Sûreté de Fonctionnement, 2ème édition Editions Cépaduès, 1996.

[LEE 85] W.S. Lee, D.L. Grosh, F.A. Tillman, and C.H. Lie, Fault tree analysis, methods and applications a review, IEEE Trans. On Reliability, vol. R-34, august 1985.

[LIM 86] N. Limnios, R. Ziani, An Algorithm for Reducing Cut Sets in Fault-Tree Analysis, IEEE Trans. On Reliability, vol R 27, décembre 1986.

[LIM 86] N. Limnios, R. Ziani, J.F.Guyonnet, Traitement automatique des arbres de défaillance non cohérents : Le logiciel ARBRE-NC, 4éme Séminaire européen de la 3SF, Deauville, 4-6 juin 1986.

[LIM 05] N. Limnios, Arbre de défaillance, Edition Hermès, 2005.

[MAR 86] M.A. Marsan et G. Chiola, On Petri Nets with Deterministic and Exponentially Distributed Firing Times, European Workshop on Applications and Theory of Petri Nets, pp. 132-145, juin 1986.

[MAZ 81] N. Mazars, De l'indépendance et des dépendances stochastiques en théorie de la fiabilité, Thèse de Dr 3éme cycle, Université Paul Sabatier de Toulouse, 1981.

[ODE 95] K. Odeh, Nouveaux algorithmes pour le traitement probabiliste et logique des arbres de défaillance, Thèse de Doctorat de l'Université de Technologie de Compiègne, décembre 1995.

[PAG 80] A. Pagès et M. Gondran, Terminologie relative à la Fiabilité – Maintenabilité - Disponibilité (FMD), Collection de la Direction des Etudes et Recherches d'Electricité de France, Edition Eyrolles, Paris, Octobre 1980.

[RAS 78] D.M. Rasmuson, N.H. Marshall, FATRAM – A core efficient cut set algorithme, IEEE Trans. On Reliability, vol. 27, N° 4, October1978.

[RAU 93] A. Rauzy**,** New algorithms for fault trees analysis, Reliability Engineering and System Safety, vol. 40, 1993.

[THO 02] P. Thomas**,** Contribution à l'approche booléenne de la sûreté de fonctionnement : L'atelier logiciel Aralia WorkShop, Thèse de doctorat de l'Université Bordeaux I, école doctorale de sciences physiques et de l'ingénieur, février 2002.

[VIL 88] A. Villemeur, Sûreté de Fonctionnement des systèmes industriels, Collection de la Direction des Etudes et Recherches d'Electricité de France, Edition Eyrolles 1988.

[ZIA 86] R. Ziani, Vérification des objectifs de disponibilité et de maintenabilité des systèmes complexes modélisés par leurs ensembles minimaux, Thèse de Doctorat Ingénieur,Université de Compiègne , Décembre1986.