Approche à base de logique floue pour le test et le diagnostic des circuits analogiques

F. Mohamed

THÈSE

présentée par

Firas MOHAMED

pour obtenir le titre de **DOCTEUR**

de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

(arrêté ministériel du 30 mars 1992)

Spécialité : **Microélectronique**

Approche à base de logique floue pour le test et le diagnostic des circuits analogiques

Date de Soutenance : 03 JUILLET 1997

Composition du jury :

Pierre GENTIL	*Président*
Christian LANDRAULT	*Rapporteur*
Adam OSSEIRAN	*Rapporteur*
Bernard COURTOIS	*Examinateur*
Anne DERIEUX	*Examinateur*
Meryem MARZOUKI	*Examinateur*

Thèse préparée au sein du Laboratoire TIMA/INPG

إِلَى بَلَدِي
وَ إِلَى رِيم

à mon Pays
et à Rim

1

Remerciements

Je remercie monsieur Bernard Courtois, Directeur de recherches au CNRS et Directeur du Laboratoire TIMA, pour m'avoir accueilli et accepté de co−diriger mes travaux avec madame Meryem Marzouki, Chargée de recherches au CNRS et Responsable du groupe DCS, dont les conseils et les encouragements m'ont été très utiles dans l'orientation et l'avancement de mes travaux.

Je tiens à remercier monsieur Pierre GENTIL, Directeur du Collège Doctoral à l'INPG qui me fait l'honneur de présider mon jury. Ensuite, je tiens à remercier monsieur Christian LANDRAULT, Directeur de recherche CNRS au Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) pour m'avoir fait l'honneur d'être rapporteur de cette thèse, ainsi que monsieur Adam OSSEIRAN, Professeur à l'École d'ingénieurs de Genève, qui lui aussi m'a fait l'honneur d'être rapporteur de cette thèse.

Mes remerciements s'adressent aussi à madame Anne Derieux, maître de conférence, chercheur au laboratoire LIP6 d'avoir accepté de participer à ce jury et aussi pour les discussions très utiles qu'on a eu durant mon séjour au LIP6.

Je profite de cette occasion pour adresser mes remerciements à tous mes collègues de bureau 101 : M. Hédi Touati pour sa collaboration à ce travail et Walid Maroufi pour ses remarques intéressantes, ainsi que Hong Ding, Maher Rahmouni et Sebastien Colin.

Je tiens aussi à remercier les slovènnes de l'Institut Jožef Stefan de Ljubljana , Professeur Franc Novak, Toni Biasizzo, Alenka Zuzek, Marina Santo, et les autres pour leur accueil et leur collaboration à ce travail.

J'adresse un message particulier à mes deux familles à Jableh et à tous mes amis en Syrie et en France : Bachar, Zein, Yasser, Iyad, Hassan, Muhammad, Walid, Dureid, Nidal, Hasan et les autres.

Finalement, j'adresse mes remerciements au reste du Laboratoire TIMA, et notamment à sa partie administrative.

Résumé

Les circuits et systèmes analogiques sont de plus en plus utilisés dans le cadre d'applications nouvelles. Ils deviennent également plus complexes, ce qui crée la nécessité de disposer de méthodes automatiques pour leurs tests et leurs diagnostics qui à eux deux représentent un problème crucial dans ce domaine. Bien que le test et le diagnostic des circuits digitaux ait été développé avec succès au point qu'il a pu être automatisé, son développement pour les CA en est encore à ses premiers pas. Il y a deux raisons principales à cette situation, il n'y avait pas un besoin pressant dans l'industrie jusqu'à récemment, d'une part, et d'autre part, la nature electrique des CA est très complexe.

Cette thèse, qui se place dans ce contexte, a pour objectif de développer une nouvelle approche pour le test et le diagnostic des CA et mixtes. Une voie relativement peu explorée, mais prometteuse eu égard aux résultats obtenus pour le diagnostic d'autres dispositifs et systèmes dynamiques, consiste en l'étude d'approches de l'IA pour la résolution de problèmes. Nous avons étudié les différentes approches possibles et notamment les approches suivantes : l'approche à base de modèles profonds, l'approche qualitative et l'approche à base de logique floue.

Le résultat de cette étude a donné lieu à une nouvelle approche développée utilisant la logique floue et ses techniques. Cette nouvelle approche a été implémentée dans un système nommé FLAMES. FLAMES, qui est conçu pour faire le diagnostic des CA, a apporté plusieurs améliorations de l'état de l'art notamment en définissant les tolérances comme des intervalles flous. Il est aussi capable de réaliser la simulation des circuits et de choisir les meilleurs points à tester lorsque le diagnostic reste ambigu. Cette dernière possibilité est réalisée par une approche semi-qualitative à base du raisonnement qualitatif et de logique floue. Il est possible de la développer pour le test et le diagnostic des systèmes mixtes, puisqu'elle s'applique aux circuits digitaux également. Finalement, les différents résultats obtenus confirment la validité de l'approche développée et implémentée.

Abstract

Analog circuits and systems are more and more used in the frame of novel applications. Also, they become more complex, which creates the necessity to dispose of automatic methods for their test and diagnosis which represents a crucial problem in this domain.

Although the test and the diagnosis of digital circuits has been successfully developed to the point that it can be automated, its development for analog circuits (AC) is still in its first steps. There are two principle reasons for this situation. One is that there has been no pressing need from the industry. The other reason is the very complex electric nature of AC.

This thesis, which is situated in this context, has as objective the development of a novel approach for the test and the diagnosis of analog and mixed circuits.

The issues of artificial intelligence have not been widely explored in this domain. But, their application to the test and the diagnosis of other analog devices and systems have given a promising results.

We have studied the different possible approaches, specially the following ones : model-based reasoning, qualitative reasoning and fuzzy logic based approach.

The result of this study is is a development of a novel approach using fuzzy logic and its techniques. Other approaches are not excluded. This novel approach has been implemented in a system named FLAMES. FLAMES, which is built to achieve the diagnosis of AC, has brought many enhancements to the state of the art specially in defining the tolerances as fuzzy intervals. It is also able to realize the simulation and to choose the best test point when the diagnosis is ambiguous. This last possibility is realized by a semi-qualitative approach which considers a purely qualitative step and a quantitative step based on fuzzy logic. It is also possible to develop this approach for the diagnosis of mixed systems since it is equally applicable to digital circuits.

Finally, obtained experimental results validate the approach developed and implemented.

Table des Matières

Liste des Figures

Liste des Tableaux

Chapitre 0

Introduction

Chapitre 0

Introduction

Le test et le diagnostic des circuits analogiques (CA) est un thème important et constitue un besoin vital pour l'industrie électronique. La recherche dans ce domaine a commencé depuis plus de deux décennies. Bien que nombreux, les résultats obtenus sont peu satisfaisants en pratique [NMSZB93].

Le test et le diagnostic des circuits digitaux a été développé avec succès au point qu'il a pu être automatisé. Malheureusement, son développement pour les CA en est encore au point où l'intuition de l'ingénieur reste l'outil le plus puissant utilisé dans l'industrie ! En fait, il n'y avait pas jusqu'ici de besoin pressant pour l'industrie : les circuits analogiques étaient souvent de petite taille.

Une autre raison est la nature électrique complexe des CA, une application directe des modèles digitaux de fautes se révèle inadéquate pour capturer leurs comportements incorrects [NCA93]. Les difficultés sont nombreuses, citons :

1- La nature des signaux analogiques est continue, l'ensemble de valeurs à considérer est alors infini. Les mesures sont imprécises et on est obligé de traiter les valeurs avec des tolérances. Les paramètres électriques, tels que l'intensité du courant, sont mal connus, et leur évaluation précise est difficile, voire impossible, surtout dans un circuit fautif. Enfin, c'est un univers très complexe, marqué par l'imprécision et l'incertitude et cela demande un traitement particulier et

délicat.

2- Il y a trop de modes d'échecs et d'effets conséquents, ce qui rend les méthodes utilisant des dictionnaires de fautes non applicables [MW89a].

3- Ces circuits tendent à contenir des boucles fermées des chemins de retour et un grand nombre de composants qui n'ont pas d'entrée ni de sortie, ce qui rend la méthode des graphes orientés non applicable.

4- Il n'y a pas un ensemble unique de valeurs possibles que les mesures doivent considérer pour un CA en fonctionnement. Les résultats de test acceptables ont des marges dépendantes de la tolérance dans les circuits et de la précision des mesures [MW89a].

5- L'ajout à un circuit analogique de composants de test peut créer beaucoup de problèmes d'interactions et même changer le comportement nominal du circuit, cela favorise le test et le diagnostic externes tels que le système de diagnostic développé dans le cadre de nos travaux de recherche.

La tendance actuelle en conception de circuits VLSI étant le développement de circuits mixtes analogiques/digitaux, le circuit total sous test devient complexe et l'intuition de l'ingénieur ne suffit plus à sa validation ni à sa maintenance. Il y a donc un besoin urgent d'une approche plus systématique pour le test et le diagnostic des circuits analogiques et mixtes.
Une autre tendance vers une approche plus systématique de résolution de ce problème est *le développement récent des circuits VLSI analogiques.* En réseaux de neurones, qui sont l'espoir du développement des machines intelligentes, on a trouvé que les puces VLSI analogiques ont beaucoup d'avantages sur les digitales, surtout en temps de calcul et d'adaptation.

En fait, l'histoire du diagnostic n'était pas en suspens dans toutes les directions. Le diagnostic des *circuits linéaires* a déjà été développé et bien compris. Mais l'application directe de ces techniques ou leur adaptation aux circuits modernes ont été très limitées

à cause de la non linéarité de ces circuits, notamment dans des conditions de défaillance. Récemment, quelques méthodes ont été développées pour les circuits non linéaires [Liu87], ainsi que des systèmes basés sur des techniques d'intelligence artificielle.
Cette dernière voie, celle de l'Intelligence Artificielle (IA), est relativement peu explorée mais prometteuse eu égard aux résultats obtenus pour le test et le diagnostic des circuits analogiques. Elle consiste en l'étude d'approches développées en intelligence artificielle pour la résolution de nombreux problèmes dans ce domaine.

Notre travail se situe dans ce cadre. Nous avons surtout étudié des approches de l'IA déjà utilisées et des nouvelles approches non explorées auparavant.

La contribution de l'IA dans le domaine du test des circuits électroniques n'est pas récente. Les dictionnaires de fautes, les méthodes à base de règles, le raisonnement à base de modèles profonds, le raisonnement qualitatif ont été utilisés avec plus ou moins de succès.
Le nombre de fautes possibles est infini dans le cas des CA ce qui rend les dictionnaires de fautes d'une efficacité très limitée, sauf pour les fautes très fréquentes. Quant aux méthodes à base des règles, celle ci se sont avérées insuffisantes.
Le raisonnement à base de modèles profonds et le raisonnement qualitatif apparaissent plus appropriés pour résoudre ce genre de problèmes, c'est pourquoi nous avons choisi de les étudier dans les chapitres suivants.
D'autre part, la nature floue des circuits analogiques nous a conduit à étudier une nouvelle approche pour leur test et leur diagnostic, celle basée sur la logique floue.

Dans ce travail, nous introduisons pour la première fois l'utilisation de la logique floue pour le test et le diagnostic de CA : les *ensembles flous* sont utilisés pour exprimer l'imprécision et l'incertitude sur les paramètres. Nous montrerons leur importance, leur nécessité et avant tout, leur capacité à simuler l'environnement imprécis des CA.

Le premier chapitre de ce document présente une étude comparative des possibilités offertes par chacune des trois approches, l'approche à base de modèles profonds, l'approche qualitative et celle de la logique floue, pour le test et le diagnostic

des circuits analogiques et mixtes.

Dans le chapitre 2, l'état de l'art des systèmes déjà développés à base des approches de l'IA pour le test et le diagnostic des CA est présenté. Une évaluation et une discussion des points forts et points faibles de ces systèmes sont présentées, ainsi qu'une première étude de notre approche qui nous a conduit vers l'utilisation de la logique floue.

Le chapitre 3 présente le système FLAMES (A Fuzzy Logic ATMS and Model-based Expert System) que nous avons conçu et implémenté à partir des idées déduites des deux premiers chapitres. Les différentes unités de FLAMES sont présentées et surtout l'ATMS-flou, qui est le noyau du système, est bien détaillé. Ensuite les chapitres 4 et 5 sont consacrés à la présentation de deux autres unités, respectivement en charge de la recherche du meilleur nœud à tester, qui a pour but d'améliorer le diagnostic et de la prise de décision floue pour la vérification fonctionnelle et le choix d'une méthode de conception en vue du test.

Le chapitre 6 est consacré à l'implémentation de FLAMES et aux différents résultats obtenus par son application à différents circuits analogiques et digitaux. Finalement, nous concluons sur le bilan du travail effectué et les perspectives pour les futurs travaux de recherche dans le domaine des circuits analogiques et mixtes.

Chapitre 1

Approches de l'IA pour le test et le diagnostic des circuits analogiques

Chapitre 1

Approches de l'IA pour le test et le diagnostic des CA

Ce chapitre est consacré à l'étude d'approches d'Intelligence Artificielle (IA) pour le test et le diagnostic des circuits analogiques (CA). Nous développerons notamment les trois approches suivantes :

- Techniques de raisonnement à base de modèles profonds ("model-based reasoning") ;
- Techniques de raisonnement qualitatif ("qualitative reasoning") ;
- Techniques de raisonnement basées sur la logique floue ("fuzzy logic") ;

Une étude des possibilités offertes par chacune de ces approches est présentée, ainsi que les idées intéressantes de leurs développements, afin de construire une méthodologie basée sur une combinaison adéquate de ces approches.

1.1 Le raisonnement

Notons que le *raisonnement* rencontré dans le domaine de l'IA est multiforme. Certaines formes correspondent à la nature du raisonnement (raisonnement hypothétique, analogique, etc.), d'autres à la nature des connaissances sur lesquelles s'appuie le raisonnement (raisonnements approximatif, qualitatif, etc.).
Dans notre travail, différentes formes du raisonnement sont utilisées afin de résoudre

au mieux notre problème. Du fait de son importance, le raisonnement approximatif occupe une grande part de ce travail, on va rapidement donner sa définition afin que le lecteur voit clairement l'environnement dans lequel nous évoluons.
L'imperfection des connaissances et des données, qui est une caractéristique principale du monde réel, est la raison pour laquelle ce genre de raisonnement a été développé. Un grand nombre de termes sont d'ailleurs utilisés (parfois, avec négligence) pour exprimer les divers aspects de cette imperfection : *imprécision, incertitude, etc.* La question est de développer des mécanismes de *raisonnement approximatif* efficaces, capables de prendre en compte ces imperfections. Concevoir de tels mécanismes nous oblige à sortir du cadre strict de la logique mathématique classique. D'ailleurs, on peut dire qu'autant la logique considérée se rapproche de la réalité autant on aura du succès. Il s'agit donc de :

* *définir une représentation de l'incertitude et de l'imprécision ;*
* *étendre les schémas de raisonnement pour prendre en compte ces nouveaux aspects ;*
* *propager l'approximation au cours des étapes du raisonnement.*

L'approche probabiliste a été largement utilisée pour exprimer l'*incertitude*. Quant à l'*imprécision*, elle a été traitée via les intervalles numériques ou bien par des valeurs qualitatives comme nous allons le voir dans les chapitres suivants.

1.2 Le raisonnement à base de modèles profonds

La construction d'un dispositif, ou de n'importe quel système, suit généralement la démarche suivante : (a) le but doit être bien connu et bien déterminé. C'est ce qui peut être appelé "le fonctionnement" (b) un modèle intuitif imaginaire correspondant aux composants principaux et à leur fonctionnement est identifié (c) ce modèle est raffiné petit à petit jusqu'à arriver à un modèle acceptable comme premier pas de la construction, etc. Le modèle est un concept très important dans le déroulement de la construction.
Pour déterminer la raison de l'arrêt d'un système, il est utile de savoir comment il devrait fonctionner. C'est cette idée simple qui peut expliquer en partie l'importance

considérable générée récemment par ce mode de raisonnement à base de modèles ; en particulier, dans ses applications en diagnostic et mise au point [DH88].

Le mot *modèle* a été défini de différentes manières suivant le domaine de son utilisation. Pour nous, le modèle est une formulation mathématique qui décrit d'une façon fidèle le comportement du système réel d'un certain point de vue, il devrait permettre la prédiction de certains résultats et il faut qu'il soit d'un coût acceptable dans sa formulation, la plus simple possible. Dans le domaine du diagnostic, le modèle est construit à partir des composants et de leur comportement et si le modèle du comportement prévu est visé on tend vers ce qu'on appelle "des modèles profonds".

La tâche principale de ce mode de raisonnement est de suivre les différences (divergences) entre les observations qui indiquent le comportement réel du dispositif, et les prédictions qui indiquent son comportement théorique.

Si le modèle est correct, alors la divergence, si elle existe, vient des défauts de la machine. Mais, cette hypothèse est nécessairement fausse dans tous les cas (le modèle n'est pas le dispositif lui même [DH88]). Quelquefois, si le modèle est proche de la réalité, cette approche reste *acceptable*. Mais que se passe-t-il quand les approximations ne sont pas suffisamment bonnes ?

*** Pourquoi des modèles profonds ?**

La définition précédente du modèle n'est pas suffisamment précise car il y a plusieurs méthodes pour la construction du modèle d'un même système.

Les modèles profonds sont ceux construits à partir de la structure du système : les composants, leurs interconnexions et les comportements prévus des composants de ce système.

*** Pourquoi, et quand, utiliser cette approche ?**

a- *Comparée aux méthodes de vérifications* : ces méthodes ne diagnostiquent pas, elles font des vérifications (par test de tous les cas possibles). Par contre, le diagnostic à base de modèles profonds est un diagnostic dirigé par les symptômes (on commence par un symptôme spécifié et on "remonte" vers la faute).

b- *Comparée aux dictionnaires de fautes* : dans une telle approche, les fautes sont bien spécifiées. Cette méthode peut être utile pour des fautes très fréquentes. Par

contre, dans l'approche à base de modèles profonds, la faute est définie par exclusion comme *un comportement différent du comportement prévu*, une conséquence de cette vue étant la capacité de couvrir une classe plus large de fautes. La figure 1.1 [CB90] peut expliquer cette idée.

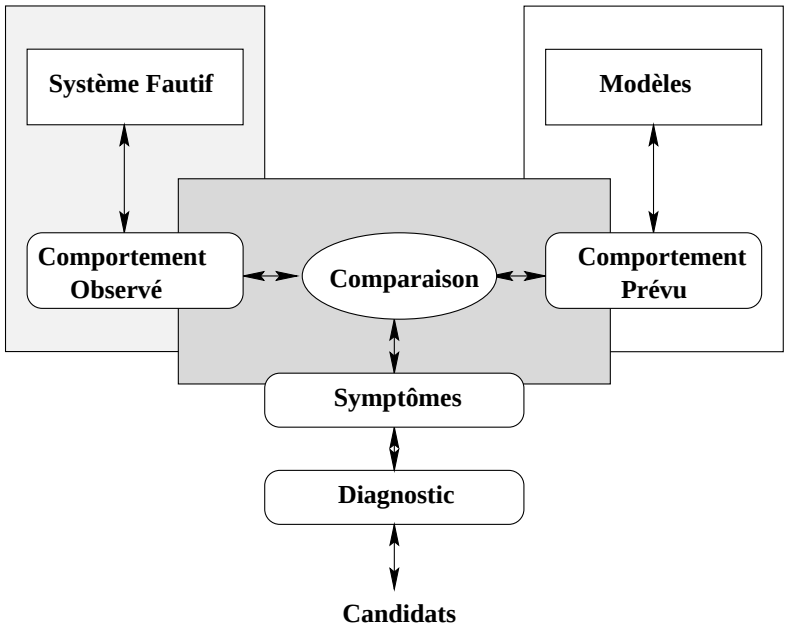


Figure 1.1: Diagnostic à base de modèles profonds

c-*Comparée aux systèmes à base de règles* : Cette méthode est fortement dépendante du dispositif sous test. L'expert qui devient familier avec le système à diagnostiquer décrit les fautes sous forme d'implications "si/alors". Un nouvel ensemble de règles est nécessaire pour chaque nouveau dispositif et le temps nécessaire pour accumuler l'expérience afin d'arriver à ces règles est important. Enfin, le raisonnement est lié à ces règles. Par contre, l'approche à base de modèles profonds est fortement indépendante du dispositif et peut-être moins coûteuse au niveau utilisation. On s'appuie sur la description du dispositif (structure et comportement) pour construire le modèle. Si on change le dispositif, on pourra utiliser le même raisonnement déjà utilisé. Cette approche est applicable sur une grande variété de dispositifs.

d-*Comparée aux arbres de décision* : Les arbres de décision fournissent un moyen simple et efficace pour écrire les séquences de test et les conclusions nécessaires pour diriger le diagnostic. Mais, ils ont un défaut important : ils constituent des méthodes

pour arriver à la réponse, mais ils n'offrent aucune indication sur la connaissance utilisée pour créer cette réponse. La mise à jour est difficile et enfin, l'arbre doit être recréé pour chaque nouveau dispositif.

Les comparaisons effectuées nous conduisent aux conditions de choix de chaque méthode. Pour l'approche à base de modèles profonds, on trouve que son utilisation est liée à deux conditions : la première est que la structure et le comportement de la machine doivent être raisonnablement connus et simples à modéliser, mais suffisamment complexe pour que la simulation exhaustive soit irréalisable, la deuxième est que l'ensemble possible de fautes est trop complexe à connaître auparavant [DH88].

Avant de présenter les différents travaux réalisés dans ce domaine, il faut rappeler de quel type de modèles on parle et pour quelle tâche ; en diagnostic et mise au point, le modèle est celui qui est basé sur la structure et le comportement ; il doit permettre la simulation (prédiction des sorties à partir des entrées) et l'inférence (inférer les entrées à partir des observations).

1.2.1 Les travaux de Davis et al.

Ces travaux ont été développés essentiellement en vue du diagnostic de systèmes électroniques. Le diagnostic, d'après ces travaux, passe par trois étapes : la génération d'hypothèses, la vérification, et la discrimination d'hypothèses. Ces trois tâches seront discutées plus loin. La figure 1.2 montre un exemple très fréquent, il a été utilisé par les équipes de Davis [Dav84] et de deKleer [Kle86] pour illustrer leurs travaux. On l'utilisera surtout pour présenter l'ATMS, "Assumption Truth Maintenance System", de deKleer (Annexe A) et ensuite notre ATMS flou. Le dispositif à étudier est composé de deux additionneurs A1 et A2, et de trois multiplieurs M1,M2, et M3.

Pour faciliter les discussions, on va donner les termes de base :
système : c'est le dispositif tout entier ;
composant : n'importe quel module (M1,M2,M3,A1, ou A2) du système ;
comportement : le comportement prédit du composant (F=12, est la valeur prédite à

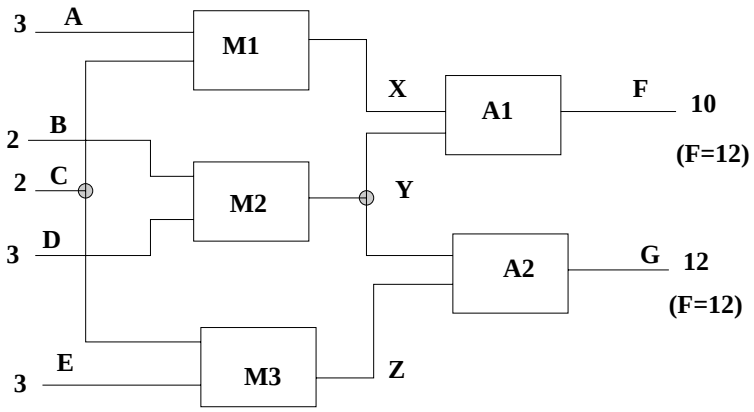


Figure 1.2: Un circuit familier

la sortie F) ;

symptôme: n'importe quelle différence entre une prédiction et une observation. Par exemple "F est égale à 10 par observation et à 12 par prédiction" est un symptôme ;

suspect : un composant que l'on suppose être responsable de la divergence ;

candidat : un suspect qui demeure après la vérification des hypothèses.

1.2.1.1 *Description de la structure et du comportement*

En général, la représentation de la structure doit considérer les points suivants :

* elle doit être hiérarchique ce qui permet un diagnostic hiérarchique ;

* elle doit être centrée objet et isomorphique à l'organisation du dispositif. Par centrée objet, on veut dire qu'à chaque composant est attachée une description de son comportement. La représentation doit être isomorphique dans le sens que les interconnexions entre les objets devraient être harmonisées avec les interconnexions de la structure ;

* le comportement peut être exprimé par des expressions. Par exemple, le comportement de l'additionneur à deux entrées A et B et une sortie C peut être exprimé par : C =A+B, B= C-A, et A= C-B [DH88].

1.2.1.2 *Concepts principaux*

Les concepts principaux de cette approche s'adressent aux trois grands problèmes du diagnostic.

* *la génération d'hypothèses* : étant donné un symptôme de dysfonctionnement, quels sont les suspects ? Il est préférable qu'un générateur soit complet (génère toutes les hypothèses), non redondant, et informé (génère peu d'hypothèses inutiles).

* *la vérification* : étant donnés tous les conflits trouvés dans la première étape, quels sont les candidats ? Pour effectuer cette tâche, une méthode simple dite "simulation à modèles de fautes" est utilisée ; il s'agit d'énumérer toutes les manières selon lesquelles les composants peuvent mal fonctionner. Une méthode plus avancée dite "suspension de contraintes" a été développée par Davis [Dav84] et utilisé par McKeon et al. [MW91] pour le diagnostic des CA. Cette étape sert à former l'ensemble de candidats à partir de l'ensemble des suspects.

* *la discrimination d'hypothèses* : quelles informations supplémentaires peuvent permettre d'aboutir effectivement au diagnostic (des mesures supplémentaires, etc.) ?

1.2.1.3 *Évaluation de la méthode "suspension de contraintes"*

En principe, cette méthode consiste à suspendre les contraintes d'un module (en pratique, remplacer le module par une *boite noire* et vérifier la consistance entre les valeurs propagées dans le nouveau circuit et les valeurs du circuit primaire). La consistance entre les différentes valeurs permet de dire que le module est fautif.
La *"suspension de contraintes"* constitue un outil important pour la mise au point, elle est capable de déterminer quels composants peuvent être responsables des symptômes observés [Dav93]. Dans [SBRM90], les auteurs critiquent cette méthode ; ils proposent une méthode complémentaire appelée *satisfaction de contraintes symboliques.*
a- La propagation de valeurs dans le réseau :
La propagation de valeurs exige l'existence d'au moins un chemin entre le candidat et la valeur observée sur lequel tous les composants sont inversibles pour pouvoir calculer

par retour en arrière. Ceci veut dire que la méthode n'est pas applicable dans une structure comme celle de la figure 1.3 [SBRM90], car quand la contrainte de C1 est suspendue, les entrées de C3 ne peuvent plus être calculées. Le problème est que cette méthode n'utilise pas toutes les informations offertes, ce qui peut être caractérisé par :

* sortance multiple reconvergente : le candidat affecte un autre composant (qui réside sur le chemin au point où la valeur est connue) par plusieurs chemins. C'est le cas de la figure 1.3.

* il y a des composants dans le système dont le comportement n'est pas inversible, alors soit l'information est divisée (figure 1.3), soit elle est perdue.

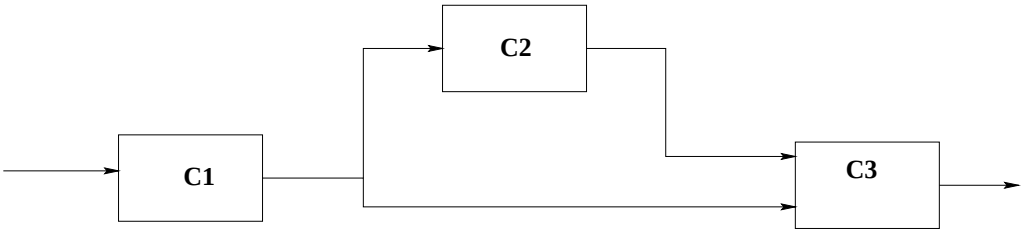


Figure 1.3: Système à sortance multiple reconvergente

La méthode de *suspension de contraintes* n'est pas générale et l'utilisation de la propagation de valeurs la restreint beaucoup plus.

b- Introduction des contraintes symboliques :

La méthode des contraintes symboliques n'a pas les limitations de la première méthode. Elle manipule les équations qui décrivent le système : les équations qui décrivent le comportement de composants, les équations qui décrivent les connexions entre ces composants, et les équations qui décrivent les valeurs d'entrée et les valeurs de sortie observées. Plutôt que de suspendre la contrainte du composant candidat, on élimine l'équation décrivant son comportement et on résout le nouveau système d'équations.

La nouvelle méthode est plus efficace, beaucoup plus applicable, et bien adaptée à l'utilisation de *modèles de fautes*, mais elle a besoin de l'utilisation d'une interface algébrique pour résoudre les systèmes d'équations qui peuvent être très coûteux, ce qui nous conduit à l'utiliser dans les cas où la *suspension de contraintes* échoue [SBRM90].

1.2.2 Le système GDE et l'ATMS

1.2.2.1 *ATMS*

Les ATMS (Annexe A) sont des systèmes de maintien de la cohérence adaptés pour le raisonnement hypothétique. Les ATMS n'effectuent pas d'inférence. Un résolveur de problèmes leur transmet les inférences effectuées (les *justifications*) et les *contradictions* détectées. En retour, l'ATMS communique au résolveur les *labels* des faits et l'ensemble des *nogoods*. Un ensemble d'hypothèses s'appelle un *environnement*. L'ensemble des environnements dont dépend le fait est son *label*. L'ensemble des environnements minimaux incohérents constitue la base des *nogoods*. Le label vérifie les quatre propriétés suivantes: *cohérence, fondement, complétude et minimalité* [Kle86].

1.2.2.2 *GDE*

GDE (un système conçu par dekleer) [DH88] fournit un mécanisme unique de génération d'hypothèses pour les *fautes multiples* aussi bien que pour les fautes simples, et il présente une stratégie bien construite pour la sélection de mesures [DH88]. Les étapes de génération et de vérification d'hypothèses sont combinées en utilisant des techniques ATMS (Assumption-based Truth Maintenance System) (Annexe A). GDE propage non seulement les valeurs, mais également les hypothèse formulées. Dans le même exemple (Figure 1.2), la valeur X=6 peut être prédite d'après l'hypothèse : "M1 marche bien" ; alors on propage (X=6; M1). Les valeurs mesurées ont comme hypothèses l'ensemble vide.

Quand deux hypothèses se contredisent au même point, le raisonnement doit commencer. La valeur observée F=10, la prédiction en Y (Y=6), et l'hypothèse "A1 marche bien", donnent X=4, X=6 (M1), X=4 (M2, A1). On fait alors démarrer le processus de la construction de conflits qui découvre qu'un des trois composants est fautif {M1,M2,A1}.

1.2.2.3 *Discrimination d'hypothèses :*

Plusieurs méthodes sont proposées, on peut effectuer de nouvelles observations ou bien changer les entrées et répéter les processus décrits ci-dessus. Dans tous les cas, le but est de rassembler la plupart des informations possibles avec un coût minimal. Certaines méthodes considèrent des informations sur les modes d'échecs des composants ; d'autres considèrent leurs probabilités d'échecs.
Étant donnée une étape particulière dans le processus de diagnostic, on analyse les conséquences de chaque mesure pour déterminer laquelle effectuer à l'étape suivante. Une bonne fonction qui évalue le coût est *l'entropie des probabilités des candidats* ; la meilleure mesure est celle qui minimise l'entropie prévue des probabilités de candidats résultant des mesures. Cette méthode est coûteuse ; plusieurs algorithmes ont été développés pour l'améliorer afin de l'utiliser avec efficacité, par exemple le système FIS [PDS87]. Une nouvelle méthode basée sur la logique floue est proposée dans ce document, elle est beaucoup moins coûteuse, plus efficace et plus simple (section 4.1).

1.2.3 Questions et lignes de recherche

Beaucoup de questions se posent : Que se passe-t-il en cas de traitement de comportements plus complexes ? Pour chaque dispositif plusieurs modèles sont possibles, lequel d'entre eux faut-il choisir ? Quand la taille du dispositif est très grande ou quand il devient très complexe, est-ce que les idées précédentes restent valables ? Comment peut-on traiter les fautes qui changent la structure ? Comment peut-on décrire le comportement de composants très complexes : une ALU, un microprocesseur, ou un contrôleur de disque ? Est-ce que les méthodes considérées sont valables pour des systèmes à comportement dynamique?

En fait, ces questions constituent de vrais problèmes, plus précisément elles constituent des lignes de recherche dans le domaine du raisonnement à base de modèles profonds. En particulier, dans le domaine du diagnostic de circuits analogiques, les méthodes précédentes sont menacées par la non-directionnalité des connexions, par

la nécessité de propager des intervalles plutôt que des entiers, etc. Par exemple, dans [Kle91], J. de Kleer propose de se concentrer sur les parties les plus probablement fautives. En utilisant une telle technique, on peut améliorer l'efficacité et diagnostiquer des dispositifs de 3000 composants qui peuvent faire exploser le GDE sans cette technique.

1.2.3.1 *Traitement des systèmes de grande taille*

Le problème est résolu en utilisant un mécanisme appelé **concentration** [Kle91] : plutôt que de générer tous les candidats en même temps, les candidats sont générés en ordre décroissant de probabilité. Mais ceci cause un problème : on a besoin de connaître les probabilités postérieures. En fait, ces probabilités sont calculées à partir des prédictions et conflits qui résultent de chaque candidat. Le calcul des prédictions et des conflits pour tous les candidats peut faire échouer le but d'utilisation de ce mécanisme. Pour résoudre ce problème, on génère un candidat à la fois en utilisant les probabilités antérieures comme estimation pour leurs probabilités postérieures ; les autres étapes sont exécutées pour chaque candidat seul, et enfin les probabilités postérieures sont estimées et l'ordre est donné. Nous avons résolu ce problème en développant une méthode à base de logique floue qui ne nécessite pas le calcul de probabilités postérieures ni de connaître les probabilités antérieures, comme le montrent les chapitres suivants.
L'approche de deKleer utilise la génération de candidats concentrée dans laquelle les probabilités postérieures sont calculées en utilisant le théorème de Bayes. La propagation de contraintes concentrée est réalisée en utilisant un TMS Hybride (HTMS) [Kle91]. On verra que l'approche que nous proposons peut répondre à la plupart des problèmes décrits ci-dessus.

1.2.3.2 *Intégration des modèles de fautes dans le GDE*

Struss et al. [SD89], proposent l'intégration des modèles de fautes dans le GDE, ce qui peut améliorer son efficacité puisqu'il n'exploite pas de connaissances sur le

comportement fautif des composants. Nous sommes convaincus que cette idée est valable mais il faut bien tenir compte du type de l'application.

1.2.3.3 *L'apprentissage de l'expérience*

Un système expert peut souffrir de plusieurs inconvénients : les règles employées sont propres à l'application, donc inexploitables par ailleurs, la masse de connaissances nécessaires est importante et souvent redondante, la mise à jour et le test du système s'avèrent délicats [DTF91]. Y. Koseki décrit, dans [Kos89], un système expert de diagnostic à base de modèles profonds. Son système est capable d'apprendre de l'expérience. Il associe l'approche à base de règles avec celle à base de modèles profonds. Rappelons qu'une règle est une association symptôme-échec.
Le système est constitué de plusieurs modules ; deux d'entre eux nous intéressent : diagnostic à base de règles et diagnostic à base de modèles profonds. A chaque symptôme déduit, le système cherche dans le premier module pour trouver une règle correspondante (et qui correspond à une expérience ancienne). Si une règle est trouvée alors le résultat sera donné ; sinon il faut exploiter le deuxième module.
Dans ce module, le raisonnement est produit comme dans les autres systèmes expliqués précédemment. Le résultat de cette expérience est formulée dans une règle qui sera ajoutée au premier module et ainsi de suite.

L'architecture de ce système offre une solution aux problèmes de systèmes à base de règles, ainsi qu'aux problèmes d'efficacité de systèmes à base de modèles profonds [Kos89].

1.3 Le raisonnement qualitatif

Le raisonnement naturel de l'homme devant un problème à résoudre se réfère à un modèle mental de nature qualitative, qu'il s'est forgé par expérience plutôt que de résoudre un système mathématique d'équations différentielles. Ce type de raisonnement est concerné par la représentation et le raisonnement sur l'univers physique.

En outre, dans ce monde, on a quelques fois besoin de résoudre des problèmes en l'absence d'informations quantitatives détaillées dans les modèles numériques de simulation. La simple utilisation de connaissances qualitatives permet de prédire en gros ce qui peut se passer devant un phénomène donné [HBC+91].
En ce qui concerne l'IA, un programme, comme l'homme, doit pouvoir résoudre un problème en l'absence d'informations quantitatives détaillées [HBC+91], c'est pourquoi le raisonnement qualitatif sur les systèmes physiques est devenu un des domaines les plus actifs et productifs ces dernières années [Kui93].

Le traitement d'un problème physique commence toujours par la compréhension du problème et le raisonnement qualitatif sur sa solution dans tous les cas. Un tel raisonnement se passe avant l'utilisation, même avant l'écriture des équations différentielles. Enfin, il est surtout nécessaire pour l'interprétation des résultats.
Le but premier de la recherche en IA concernant les physiques qualitatives considère la tâche de la modélisation elle-même, la couche conceptuelle du raisonnement sur le système physique, l'explication intuitive du comportement du système, etc. Le raisonnement qualitatif (Qualitative Reasoning : QR) cherche à compléter les systèmes traditionnels qui simulent et contrôlent les systèmes physiques pour permettre une représentation et un contrôle cognitifs adéquats de ces systèmes, ce qui explique sa relation avec l'IA [Str92].

Mais, un problème majeur qui peut heurter ce type de raisonnement est que le manque de données peut quelques fois causer une grande perte des résultats, ce qui peut être critique dans certains cas comme dans celui des circuits analogiques ainsi qu'on le verra plus loin (Voir [Kui86] pour un exemple détaillé).

1.3.1 Les approches qualitatives fondamentales

Un numéro spécial de la revue "Artificial Intelligence" a été entièrement consacré, en 1984, à cette discipline, dont deux systèmes ont été décrits :

ENVISION [KB84] et QPT [For84] ; le système QSIM [Kui86] avec les deux premiers constituaient les trois approches essentielles de ce domaine. Outre les techniques utilisées, la différence principale entre ces trois approches sont les primitives ontologiques (les concepts de base) qu'elles utilisent pour décrire un système physique. L'approche QSIM (Qualitative simulation), appelée aussi *centrée contrainte*, est la plus simple ; elle considère un ensemble de contraintes sur les paramètres du système. L'approche ENVISION, appelée *centrée composant*, s'appuie sur la structure de l'installation, ses composants et leurs interconnexions. Enfin, l'approche QPT *centrée processus*, considère le processus physique comme notion clé. Bien qu'il existe trois approches différentes, il y a des points communs entre elles pour décrire et prédire le comportement des systèmes, des composants et de leurs comportements, les interactions entre ces composants, les états de systèmes caractérisés par des valeurs qualitatives, et le comportement par des séquences de ces états.

1.3.2 L'approche centrée composant, ENVISION

Cette approche considère les composants comme concepts essentiels de traitement. ENVISION traite les systèmes composés de matériaux (fluide, électricité, etc.) de composants (vanne, transistor, etc.) et de conduits (tuyau, fil, etc.), les deux derniers ensembles sont connectés via des terminaux. Sa tâche principale est de dériver le fonctionnement à partir de la structure d'un dispositif arbitraire.
La structure est considérée comme "fixe", ce qui constitue une restriction importante lors du traitement des systèmes qui changent leur structure dynamiquement ; la violation de la structure conçue peut causer beaucoup de problèmes en diagnostic de systèmes électroniques.

L'espace de quantité d'ENVISION est Q={ -, 0, + }, où les variables prennent les signes des valeurs réelles comme valeurs qualitatives.

1.3.3 L'approche centrée contrainte, QSIM

Cette approche est fondée sur la simulation qualitative pour prédire le comportement d'un système. L'idée principale est que les comportements possibles d'un système peuvent être prédits par la simulation qualitative basée sur des équations de contraintes et d'un état initial [Kui86]. Le système QSIM commence par la description qualitative de la structure qui est elle-même déduite de la description mathématique donnée sous forme des équations différentielles et de l'état initial du système.

J.B. Kuipers considère [Kui93] que tous les modèles sont une abstraction de l'univers. Les modèles qualitatifs sont relatifs aux équations différentielles ordinaires (Ordinary Differential Equation : ODE), mais ils sont plus expressifs des connaissances incomplètes (figure 1.4). En ce qui concerne la simulation qualitative, il faut bien

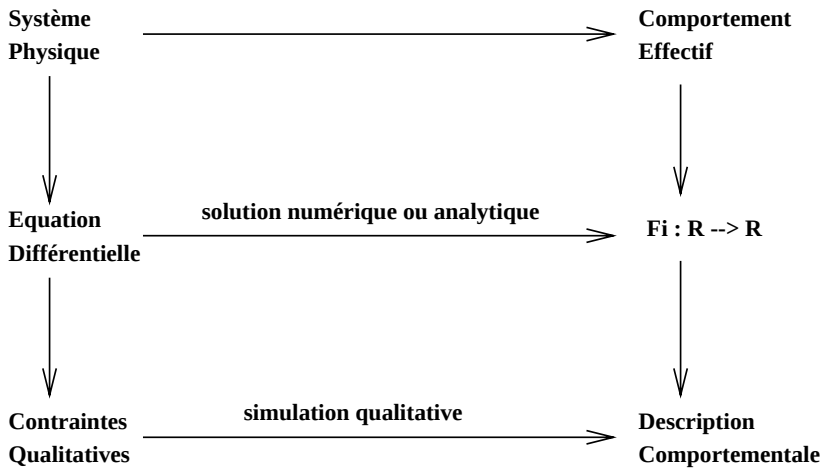


Figure 1.4: Tous les modèles sont une abstraction de l'univers

distinguer entre :

structure $\longrightarrow$ **comportement** $\longrightarrow$ **fonctionnement**,

et leurs relations de dépendance. Le comportement et le fonctionnement sont confondus en général. Le comportement décrit les comportements potentiels du système comme un réseau des états qualitatifs, distincts et possibles. Par contre, Le fonctionnement est le but (en pratique) de la structure pour produire le comportement du système [Kui84].

Un des théorèmes essentiels de QSIM est celui de l'incomplétude : quelques comportements qualitatifs peuvent être faux, ils ne fournissent pas de solutions réelles à une ODE correspondant à la QDE (qualitative differential equation).

1.3.4 L'approche centrée processus, QPT

Dans cette approche, le processus qui caractérise les changements d'états dans le temps et leurs effets dans l'univers physique constitue le concept principal. K.D. Forbus a développé le système QPT (Qualitative Process Theory) qui a pour objectif d'offrir les mécanismes nécessaires pour obtenir la représentation explicite et structurée des processus [For84]. Pour donner une approche plus complète, le QPT rassemble les outils les plus élaborés des approches qualitatives. Il y a la notion d'espace des quantités ainsi que la description qualitative des équations différentielles et des fonctions. Il a également repris la notion d'histoire de P.J.Hayes [HBC+91], et d'*envisionment* de J. De Kleer [HBC+91]. Notons avant tout que cette approche est convenable pour les systèmes dynamiques.

1.3.5 Comparaison des trois approches et limites

Dans les approches précédentes, la possibilité de prédire et d'expliquer est cruciale ; elles adoptent une même vue de formalisation, malgré leur diversité méthodologique. L'espace de quantité et l'algorithme de simulation sont des concepts essentiels dans toutes les approches qualitatives. Mais *la perte d'informations quantitatives* par la description qualitative (espace de quantité très qualitatif), d'une part, et le caractère local de la simulation d'autre part, causent la prédiction d'un ensemble de comportements futurs pouvant contenir des comportements non valides ou l'oubli des comportements valides.

Quant aux *espaces de quantité*, qui est un concept très important dans le raisonnement qualitatif, les différents types considérés dans les trois approches précédentes *laissent beaucoup d'ambiguïté* [Lei92]. La résolution de ce problème nous conduit à considérer des espaces plus fins. Des extensions ont été proposées ; une des plus récentes est

celle que nous proposons, et qui considère la logique floue (comme on le verra dans la troisième partie).

Un regard rapide sur les trois approches qualitatives précédentes permet de montrer qu'elles sont très convenables pour traiter des systèmes dynamiques. En ce qui concerne les CA, elles peuvent être une bonne base pour des systèmes d'apprentissage parce qu'elles sont capables d'explication et de prévision : l'explication des comportements actuels et la prévision des futurs comportements.
Par conséquent, une approche moins qualitative, mais capable de traiter l'imprécision et l'incertitude de l'information pouvait être la réponse à ce problème et la logique floue comme paradigme de développement d'une telle approche a été adoptée.
En effet, un des systèmes de diagnostic des CA qui ont considéré une approche qualitative est le système DEDALE que l'on présentera dans le chapitre suivant. DEDALE est basé sur les ordres de grandeur qui sont exposés dans la section suivante.

1.3.6 Les ordres de grandeur

Le raisonnement basé sur les ordres de grandeur peut être considéré comme un raisonnement qualitatif. Il traite des informations quantitatives incomplètes et réduit le poids du calcul. Dans le domaine analogique, ce genre de raisonnement est couramment utilisé pour expliquer le comportement du circuit, même s'il n'est pas toujours nommé ainsi.
Intuitivement, un tel raisonnement peut être comparé à l'utilisation d'une balance grossière (figure 1.5), qui pèse des quantités avec un niveau variable de précision, au lieu d'une balance précise [Rai91]. Ainsi, des équations d'ordres de grandeur sont définies avec des précisions qui dépendent de celles des équilibres grossiers fixés. Une

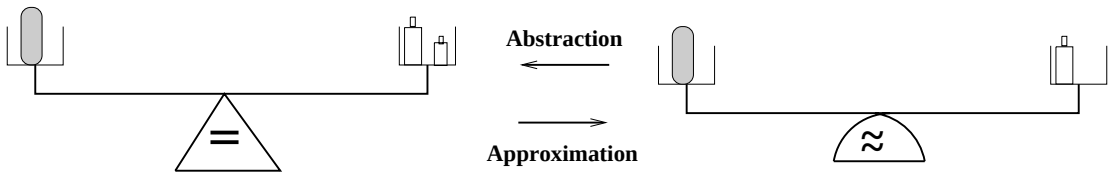


Figure 1.5: La balance des ordres de grandeur

quantité q est définie par sa valeur grossière V(q). Cette dernière est définie comme l'ensemble des valeurs qui contiennent la valeur exacte de q. Dans le domaine des réels, par exemple, l'intervalle [0,0.1] peut être choisi pour définir une petite quantité. La figure 1.6 montre l'opérateur d'égalité des ordres de grandeur. Deux quantités p et q auront le même ordre de grandeur quand les deux ensembles V(p) et V(q) se chevauchent. Cet opérateur nous intéresse en particulier parce qu'il montre toute l'idée de ce mode de raisonnement d'une part, d'autre part il nous servira au chapitre 2 pour discuter le système DEDALE [DRDM87] qui implémente les ordres de grandeur pour le diagnostic des CA.

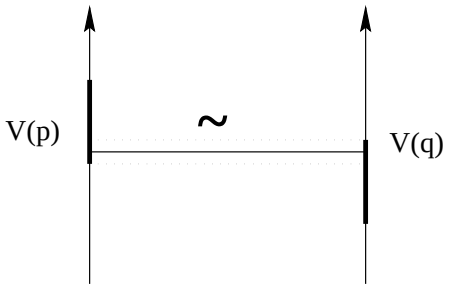


Figure 1.6: Égalité entre ordres de grandeur

1.4 Le raisonnement basé sur la logique floue

Dans le cadre de la logique classique, une proposition est soit vraie, soit fausse, soit inconnue ou indéterminée par rapport à une théorie. Mais le raisonnement humain s'appuie fréquemment sur des connaissances et des données inexactes, incertaines ou bien imprécises. D'ailleurs, le progrès continuel dans tous les domaines de notre vie est accompagné par une complexité croissante : des applications de plus en plus complexes et des systèmes de plus en plus sophistiqués nous obligent à chercher de théories capables de traiter ces complexités.

Cet environnement a précipité le besoin de traiter de propositions plus générales et de recueillir des données toujours plus loin d'être précises et définitives ; les sources d'information ne sont pas totalement fiables (et donc introduisent de l'incertitude) et, par leur nature, sont génératrices d'imprécision. La recherche de nouvelles théories

reste un besoin.

1.4.1 L'imprécis et l'incertain

L'imprécis et l'incertain, qui sont généralement utilisés sans distinction, décrivent des connaissances bien différentes. Le qualitatif "incertain" s'applique à des éléments de connaissance dont la valeur de vérité n'est pas exactement connue, elle est connue avec plus ou moins de précision comme dans l'exemple suivant : *il va probablement pleuvoir ce soir*. Par contre, le terme "imprécis" s'applique à des éléments de connaissance dont le contenu est imprécis comme montre l'exemple suivant : *Jean mesure entre 170 et 175 cm*. Où se situe le flou, alors ? En fait, quand on dit que Jean mesure environ 175 cm, la mesure se situe autour de 175 cm de façon floue. D'un point de vue pratique, un élément d'information peut être caractérisé par sa valeur et la confiance donnée à cette valeur. Dans ce contexte, on peut différencier clairement les concepts d'imprécis et d'incertain : l'imprécis concerne le contenu de l'information (la "valeur") tandis que l'incertain est relatif à sa vérité (la "confiance"). Par exemple :

- *il est probable que Jean mesure 170 cm ; incertain.*
- *Jean mesure environ 170 cm ; imprécis.*

Différentes approches ont été développées pour prendre en compte cette idée ; l'approche probabiliste (et de raisonnement Baysien) est une des premières. Les grands types de problèmes concernés sont essentiellement le diagnostic et la prise de décision.
Le modèle probabiliste est adapté à la saisie d'informations précises, mais dispersées [DP87]. Dès que la précision fait défaut, on tend à sortir du domaine de validité du modèle. Cette approche semble être un cadre trop normatif pour rendre compte de tous les aspects du jugement incertain et elle a des limites. Entre autres :

* elle exige de pouvoir estimer toutes les probabilités *a priori*, ce qui est souvent difficile.

* il est impossible de représenter l'ignorance : en probabilités, on définit la mesure

de probabilité caractérisée par la relation "P(s) + P(¬s) =1", qui signifie que la connaissance de l'événement implique la connaissance totale de son contraire. En fait donner la valeur 0.5 à la probabilité conditionnelle P(H/E) peut être interprété, dans le domaine de systèmes experts, par accorder une confiance 0.5 à une règle de la forme "si E alors H". Mais cela ne signifie pas pour autant que l'expert accorde la même valeur de confiance à la règle "si E alors ¬H" [HBC+91].

Le problème est qu'on accepte de donner des probabilités aux événements liés et inconnus, par exemple, supposons que Jean a trois amis qui lui téléphonent de façon "équilibrée" ; alors le fait que le prochain appel soit de l'un des trois a la même chance de survenir que les deux autres. Soient p, q, et r les propositions mutuellement incompatibles correspondantes. Il est raisonnable de poser P(p)=P(q)=P(r)= 1/3, ce qui entraînerait notamment P(¬p) = 2/3, alors que p et ¬p sont tous les deux également inconnus. Ce résultat est qualifié de *paradoxe de l'ignorance totale.* Les limites du raisonnement probabiliste ont conduit à la recherche d'autres mesures d'incertain dans [0,1] et qui permettent, en particulier, de placer la confiance en des éléments de connaissances à l'intérieur de sous-intervalles de [0,1].

Pour g une application définie sur l'ensemble de propositions dans l'intervalle [0,1], p et q sont des propositions, V est la proposition toujours vraie et F est la proposition toujours fausse, les axiomes suivants définissent des familles de mesures satisfaisantes :

* g(F) = 0 ;
* g(V) = 1 ;
* si q est une conséquence logique de p, alors g(q)≥g(p).

La définition de g a les conséquences logiques suivantes :

* g(p∧q) ≤ min(g(p),g(q)) ;
* g(p∨q) ≥ max(g(p),g(q)) ;

par contre dans les probabilités on a : P(p∨q) = P(p) +P(q) - P(p∧q).

Alors, une théorie plus adaptée est née. La théorie des possibilités formulée par **L. Zadeh** en 1977, offre un modèle de quantification du jugement qui permet aussi une généralisation canonique du calcul d'erreurs [DP87]. En fait, l'origine de cette théorie date de la publication par L. Zadeh d'un article sur un nouveau type d'ensembles :

les ensembles flous [Zad65], et depuis cette théorie fait l'objet de beaucoup de discussions. Récemment, elle a pris beaucoup d'importance, et a trouvé plusieurs champs d'application. On citera notamment les commandes de processus à base de signaux analogiques ainsi que le contrôle de processus à large spectre de paramètres (par exemple le freinage anti-patinage plus connu sous le label ABS). Mais avant tout, on insiste sur le fait que *la logique floue est la logique qui traite le flou, ce n'est pas la logique qui est elle-même floue.*

1.4.2 Le flou

Dans la langue naturelle, il y a beaucoup de termes qui renvoient à l'imprécis, tel que "vague", "flou", "général", ou "ambigu". L'ambigu est une forme d'imprécision liée au langage ; une information est ambiguë dans la mesure où elle renvoie à plusieurs contextes ou référentiels possibles. Ici, on ne va pas considérer ce type d'imprécision : on suppose connu le référentiel associé à l'élément d'information. Le caractère vague, ou flou, d'une information réside dans l'absence de contour bien délimité de l'ensemble des valeurs affectées aux objets qu'elle décrit. Alors, il y a des connaissances imprécises et floues et des connaissances imprécises mais non floues comme le montrent les exemples suivants [DP87] :

- *($x = y$ à ϵ près) ; imprécise non floue.*
- *x est approximativement égal à y ; imprécise floue.*

Le terme vague "approximativement" désigne plusieurs valeurs plus ou moins adéquates de ϵ. Enfin, une information peut être à la fois incertaine et floue :*Il est probable qu'il pleuve beaucoup demain.*

Un élément x, en logique classique, soit appartient à un ensemble X soit n'y appartient pas ; il n'y a pas de troisième solution. En logique floue x peut appartenir à un ensemble flou avec un degré d'appartenance égal à 0.8 par exemple, comme on va le voir. Cette dernière idée est vraiment importante parce qu'on définit un intervalle classique L de réels par ses deux extrémités, disons L= [a,b], mais le réel qui est exactement supérieur à b n'appartient pas à L ; on peut alors se demander si cette stricte détermination des extrémités n'entraîne pas le risque de perdre des informations ?

1.4.3 Le degré d'appartenance

Quand on dit :
1- Jean va venir ce soir avec une probabilité de 0.5, ou
2- il va pleuvoir ce soir avec une probabilité de 0.4 ;
on parle d'événements du futur. Quand le moment arrive, si Jean vient, alors la probabilité sera 1 sinon elle sera 0, mais dans le deuxième cas l'affirmation sera plus difficile : est ce qu'une seule goutte sera suffisante pour dire qu'il pleut ? est-ce que quelques gouttes seront suffisantes? ou bien quel nombre de gouttes sont nécessaires pour la pluie ? On peut dire que "une seule goutte d'eau signifie qu'il pleut avec 0.000...01 de degré d'appartenance, et 10000...000 gouttes signifie qu'il pleut avec 0.89 de degré d'appartenance [She93]. Le degré d'appartenance est un concept crucial dans la logique floue ; toutes les structures de cette logique seront définies à partir de ce concept.

L'exemple suivant (figure 1.7) montre ce qu'on veut dire : si on définit l'intervalle "grand = [80,100]", alors comment peut-on évaluer le nombre 79 ? On sait seulement qu'il n'est pas grand ! Maintenant si on ajoute un nouvel intervalle "petit = [60,79]", alors 79 sera petit mais pas grand, bien qu'il y ait une petite différence (figure 1.7.b) ! En logique floue (figure 1.7.a : une représentation par des intervalles flous est montrée), 79 sera considéré comme petit avec un degré d'appartenance égal à 1 et grand avec un degré d'appartenance égale à 0.9 (figure 1.7.c). Les intervalles flous sont plus riches en informations que les intervalles classiques. Pour l'instant, un ensemble flou A sera défini sur un domaine T par la donnée d'une fonction μ_A à valeurs dans [0,1]. μ_A(t) est le degré d'appartenance de t $\in$ T à A. Prenons l'exemple du domaine T des tailles possibles pour un homme. La proposition " *Jean est de grande taille* " contient le prédicat imprécis grand.
Sur la figure 1.7.c, chaque point représente le degré d'appartenance à l'ensemble flou grand ou à l'ensemble flou petit. Notons que cette définition des ensembles correspond à ce qu'on appelle "distribution des possibilités". Ce concept, qui est dérivé de la théorie des possibilités qui constitue la base théorique de la logique floue, est introduit

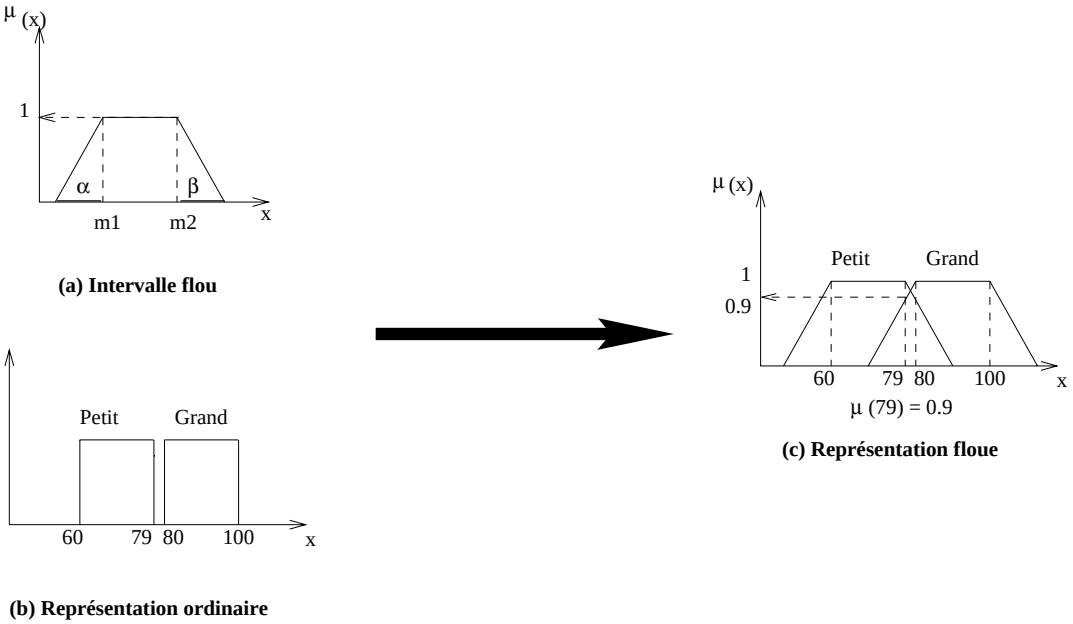


Figure 1.7: Le degré d'appartenance et les intervalles flous

dans la section suivante.

1.4.4 Possibilités et raisonnement possibiliste

On cherche à traiter l'imperfection de l'information.

L'information détenue par un individu (ou d'autres sources), sera exprimable par des propositions logiques comportant des prédicats, et éventuellement des quantificateurs, comme on va le voir plus tard. Notre système de connaissances sera un ensemble d'informations relatives à un même problème, et les prédicats peuvent alors s'interpréter comme sous-ensembles d'un même référentiel Ω dit "événement (toujours) certain". Les propositions peuvent être vues comme des affirmations relatives en l'occurrence des événements.

L'ensemble vide est identifié à l'événement (toujours) impossible. A chaque événement $A \subseteq \Omega$, on suppose que l'on peut associer un nombre réel g(A) qui évalue la confiance que l'on peut avoir relativement à l'occurrence de cet événement, par convention. De plus, si A est un événement impossible, on pose g(A)=0. En particulier $g(\emptyset)=0$ et $g(\Omega)=1$.

Dans ce cadre, l'incertitude d'un événement est décrite à la fois par le degré de possibilité de cet événement et par le degré de possibilité de l'événement contraire, ces deux degrés n'étant pas que faiblement liés. Le complément à 1 du degré de possibilité de l'événement contraire peut s'interpréter comme un degré de nécessité (certitude). Alors, dans cette approche on définit deux mesures d'incertain : une mesure de possibilité Π et une mesure de nécessité N, définies par :

$$\Pi(p \cup q) = \max(\Pi(p),\Pi(q)) \qquad (F.1)$$

$$N(p \cap q) = \min(N(p),N(q)) \qquad (F.2)$$

Cette théorie conduit à la théorie de la logique floue dans laquelle la représentation de l'imprécision est faite à l'aide de fonctions d'appartenance à un ensemble et à la façon d'associer à cette représentation un facteur de confiance traduisant l'incertain.

1.4.5 Le certain, l'incertain et l'ignorance

En probabilités, $P(p) = 1$ signifie que p est certain. Ici, $\Pi(p) = 1$ ne signifie pas que p est certain car $\Pi(\neg p)$ peut être aussi égale à 1 si $N(\neg p)$ est nulle. En revanche, l'égalité $N(p) = 1$ qui implique $N(\neg p)=0$, qualifie un événement certain.
En fait, affirmer que A et $\neg$A sont également possibles correspond à la situation d'ignorance totale.

1.4.6 Les distributions de possibilité

Quand l'ensemble Ω est fini, toute mesure de possibilité peut être définie à partir de ses valeurs sur les singletons de Ω :

$\forall$ A, $\Pi(A) = \sup\{\pi(w) \ / \ w \in A\}$ où $\pi(w) = \Pi(\{w\})$;

π, appelée *distribution de possibilité*, est une application de Ω dans [0,1].
Quand l'ensemble Ω est infini, l'existence d'une distribution de possibilité n'est pas assurée. Elle ne le devient que si l'on étend l'axiome de base, celui de l'union, à des unions infinies d'événements [DP87].
illustration [HBC+91] :
Une proposition floue prend la forme X est A, où X est une variable prenant ses

valeurs dans un domaine T et A un ensemble flou : *Jean est de grande taille.* On appelle proposition élémentaire une proposition de la forme X prend la valeur t $\in$ T. Une telle proposition est dite précise par rapport au référentiel T : *la taille de Jean est 182cm.* La proposition toujours fausse F correspond à "X prend sa valeur sur $\emptyset$" et la proposition toujours vraie V correspond à "X prend sa valeur sur T".

Le contenu de la proposition imprécise "X est A" peut être représenté par l'ensemble flou des valeurs plus ou moins possibles que peut prendre la variable X.
Quel que soit t $\in$ T, la possibilité pour que X prenne la valeur t vaut μ_A(t), ce qui s'écrit : $\pi_X(t) = \mu_A(t)$,
π_X est la distribution de possibilité attachée à X. Cela signifie que la possibilité que X soit égale à t est mesurée par le degré d'appartenance de t à A. Il faut noter que cette égalité ne traduit pas une identité, les deux fonctions ne possédant pas les mêmes propriétés. Mais, c'est cette relation qui permet de lier les deux concepts.
La relation précédente se lit : *la possibilité que la taille de Jean vaille t sachant que Jean est grand est égale au degré de vérité de la proposition Jean est grand sachant qu'il mesure t.*
A partir de maintenant, on ne va plus parler des distributions de possibilités. On va plutôt parler des ensembles flous (ou des intervalles flous qu'on utilise en pratique) qui sont plus clairs et permettent notre travail.

Définitions et remarques

On appelle *support* d'un ensemble flou A le sous-ensemble des valeurs de t telles que $\mu_A(t) > 0$ et *noyau* de A le sous-ensemble non flou des valeurs de t telles que μ_A(t) = 1. Le support inclut toujours le noyau. Ces deux ensembles peuvent ne pas être finis.
La figure 1.7.a montre un intervalle flou avec son noyau = [m1,m2] et son support = [m1-α, m2+β]. Ce qui nous conduit aux remarques suivantes:
- un fait est précis lorsque μ_A ne vaut 1 que pour une valeur unique de T et 0 partout ailleurs, par exemple, *la taille de Jean est exactement 160 cm* (figure 1.8.a) ;
- un fait est flou s'il existe des valeurs t telles que μ_A(t) est compris strictement entre 0 et 1 ;

- un fait est imprécis mais non flou lorsque μ_A vaut 1 pour plusieurs valeurs de t mais 0 partout ailleurs, par exemple, *Jean mesure entre 160 et 170 cm* (figure 1.8.a) ;
- un fait est imprécis mais flou comme sur la même figure :*la taille de Jean est de l'ordre de 190 cm* ;

Notons aussi que les ensembles flous (ou les distributions de possibilité) peuvent être utilisés à plusieurs fins (figure 1.8) [HBC+91] :
*pour présenter des données précises ou imprécises (figure 1.8.a). Notons que la courbe est souvent approchée par un trapèze, comme l'indique la figure 1.8.b ; Les

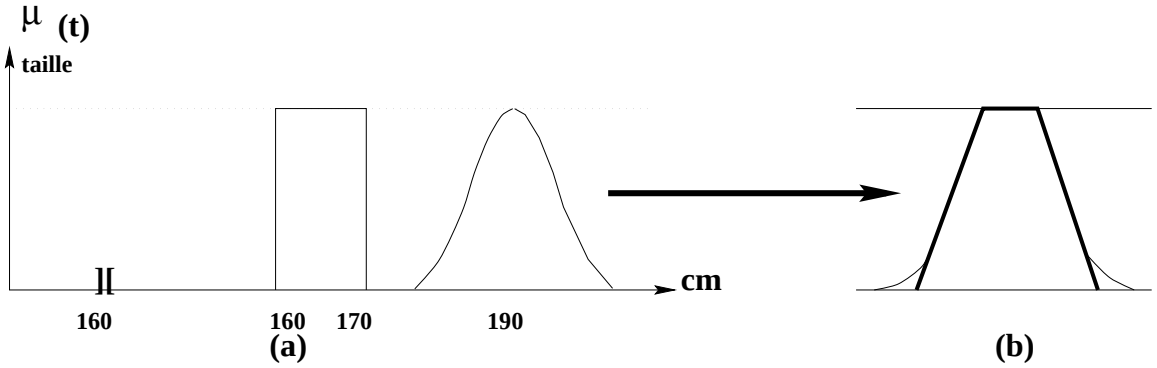


Figure 1.8: Les ensembles flous

distributions de possibilité trapézoïdales sont complètement définies par la donnée du noyau et du support : un trapèze se définit par un quadruplet (a,b,α,β) où α et β désignent la largeur des intervalles du support autour du segment noyau [a,b], comme on a vu auparavant.
* pour définir des prédicats flous, comme *Grand ou Jeune* (figure 1.9). Ici, on montre aussi comment la forme continue peut, dans la pratique, être approchée par une ligne brisée ;
* pour définir des ensembles flous de proportions traduisant des modificateurs comme *pas* ou *très* ou des quantificateurs flous comme *assez* ou *généralement*. Les fonctions d'appartenance associées sont définies sur [0,1] et prennent leur valeur sur [0,1].

La composition de fonctions est aussi possible, elle permet de définir des ensembles flous comme *assez_jeune*. L'effet de modificateurs sur l'appartenance à un ensemble

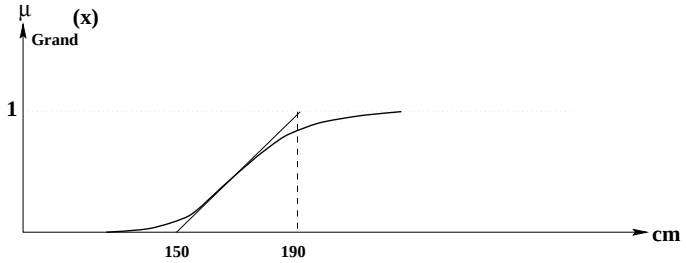


Figure 1.9: Prédicats flous par des fonctions d'appartenance

flou peut être illustré par l'exemple suivant :
Très_jeune peut être défini par : si t∈T, alors $\mu_{Tres\_jeune}(t) = [\mu_{Jeune}(t)]^2$;
Assez_jeune résulte de la composition de la fonction traduisant *assez* avec μ_{Jeune}.
Toutes ces définitions sont très intéressantes, nous les utiliserons pour le développement de nos méthodes aussi bien dans le chapitre 3 que dans le chapitre 4. L'influence des modificateurs sera aussi utilisée pour calculer les poids des objectifs pour la prise de décision floue (chapitre 5).

1.4.7 Opérations élémentaires sur les ensembles flous

Les définitions les plus couramment adoptées sont les suivantes (si s∈S et t∈T) :

* l'inclusion : $\mu_A(s) \leq \mu_B(s)$ si B⊇A ;
* l'égalité : $\mu_A(s) = \mu_B(s)$ si B=A ;
* l'union : $\mu_{A\cup B}(s) = \max(\mu_A(s),\mu_B(s))$;
  qui équivaut à $N(A\cup B) = \max(\Pi(A), \Pi(B))$;
* l'intersection : $\mu_{A\cap B}(s) = \min(\mu_A(s),\mu_B(s))$;
  qui équivaut à $N(A\cap B) = \min(N((A),N((B))$;
* la complémentation : $\mu_{\tilde{A}}(s) = 1 - \mu_A(s)$; qui équivaut à $\forall$ A, $\Pi(A) = 1 - N(\tilde{A})$ et qui postule qu'un événement est nécessaire lorsque son contraire est impossible.
* le produit cartésien : $\mu_{A\times B}(s,t) = \min(\mu_A(s),\mu_B(t))$;

* la cardinalité : card(A) = $\Sigma_{sj \in A}$ (μ_A(sj)) pour tous les éléments *sj* de A si A est dénombrable ou passage au continu (intégrale) dans le cas contraire.

Notons que ces définitions ne sont pas uniques ; d'autres choix pourraient être effectués liés aux applications. C'est ainsi que l'intersection peut s'exprimer également par le produit μ_A * μ_B ou la fonction max(0,$\mu_A + \mu_B$-1).

1.4.8 Quantités, intervalles et nombres flous

Une quantité floue Q est un ensemble flou sur les réels, c'est-à-dire, une application μ_Q de Q dans [0,1]. On supposera sauf précision contraire que μ_Q est normalisée. On appelle *valeur modale* de Q, tout nombre réel **m** du noyau de Q. Un intervalle flou est une quantité floue convexe, c'est-à-dire, dont la fonction d'appartenance est quasi-concave [DP87] :
$\forall$ u,v, $\forall$ w $\in$ [u,v], μ_Q(w) $\geq$ min(μQ(u),μ_Q(v)) qui peut être écrite sous la forme connue de la convexité :
$\forall$ u,v, $\lambda \in [0,1]$, $\mu_Q(\lambda u + (1-\lambda)$v) $\geq$min(μ_Q(u),μ_Q(v)).
D'une manière rigoureuse, on peut dire que les intervalles fermés sont généralisés par des intervalles flous dont la fonction d'appartenance est semi-continue supérieurement (s.c.s).
Les sous-ensembles compacts de $\Re$ (fermés et bornés) sont généralisés par les quantités floues s.c.s, à support compact. On appellera *nombre flou* un intervalle s.c.s. à support compact et de valeur modale unique. Si **M** est un nombre flou de valeur modale **m**, **M** est une représentation possible de *environ m* [DP87]. Dans le cas d'un intervalle flou, l'ensemble des valeurs modales est un intervalle.
Un intervalle flou est une représentation commode de quantités imprécises, plus riche en informations qu'un intervalle précis.
L'intervalle flou permet d'avoir une bonne représentation : on choisira le support de façon à être sûr que la quantité cernée n'en sortira pas, et le noyau contiendra les valeurs les plus plausibles. Dans ce cas la détermination d'une fonction d'appartenance peut être faite par un individu.

1.4.9 Calcul pratique d'intervalles

1.4.9.1 *Calcul d'intervalles ordinaires*

L'utilisation de l'arithmétique des intervalles a été largement étudiée [Moo66] [Ste74]. Ce genre de calcul n'est pas facile à gérer parce que toutes les opérations arithmétiques ont lieu sur des intervalles. Des problèmes d'explosion dans les valeurs sont fréquents et des erreurs de calcul peuvent avoir lieu. L'exemple suivant montre bien ces problèmes [Ste74] :
Supposons qu'une quantité x entre dans le calcul en plusieurs endroits. Le calcul d'intervalles ne reconnaît pas le fait que x doit avoir la même valeur en chaque point dans le calcul, et ceci crée le problème d'avoir comme résultat un intervalle plus large. Afin d'illustrer cela supposons qu'on veut calculer :

$$w = \frac{x+y}{x+z}$$

où x,y, et z sont représentés par des intervalles ordinaires [x1,x2], [y1,y2], et [z1,z2], respectivement, et pour lesquelles on prend les valeurs suivantes : [1,2], [.01,.02], et [.001,.002].
Le calcul donne :

$$[w1, w2] = \frac{[1.01, 2.02]}{[1.001, 2.002]} = [.50449550, 2.0179821]$$

Mais si on écrit la même relation sous la forme

$$w = \frac{1 + y/x}{1 + z/x}$$

Le calcul final donnera :

$$[w1, w2] = [1.0029940, 1.0194903]$$

qui est beaucoup plus précis que la première réponse.

Malheureusement, ce n'est pas si simple de trouver la meilleure formule à utiliser. Supposons que l'on désire résoudre le même problème, mais pour des données différentes :

[.001, .002], [1.001, 1.002], et [1.001,1.002] pour x, y, et z, respectivement.
Les calculs donneront [w1,w2] = [.99800796, 1.0019961], par la première formule et [.5, .2] par la deuxième !
Ainsi, on ne peut pas rejeter la première formule et utiliser toujours la deuxième. La formule à utiliser dépend des données, et si on choisit la mauvaise on pourrait produire un intervalle beaucoup plus large que nécessaire.

En fait, le calcul fonctionnel est le plus strict, mais très difficile à utiliser. Étudier

$$f(x, y, z) = \frac{x + y}{x + z}$$

ceci donne, pour les première données, [w1,w2] = [1.0039960, 1.0189811], ce qui montre que même la deuxième formule a donné un intervalle plus large que nécessaire !

Dans ce contexte, l'arithmétique de ce calcul doit être soigneusement implémentée et utilisée. Mais, le calcul d'intervalles reste une méthode puissante et pratique.

Dans ce qui suit, nous allons considérer le calcul d'intervalles flous dont le calcul d'intervalles ordinaires est un cas particulier.

1.4.9.2 *Calcul d'intervalles flous*

Une bonne représentation paramétrée d'un intervalle flou est celle donnée dans la figure 1.10 [SL92b][DP87] où [m1,m2] est le noyau de M, m1 (resp. m2) est appelé valeur modale inférieure (resp. supérieure) de M. [m1-$\alpha, m2 + \beta$] est le support de M si M est à support borné. α *et* β sont appelés étalements à gauche et à droite respectivement. Cette représentation est la plus générale, elle permet de représenter uniformément :
un nombre réel m par M = [m,m,0,0], un nombre flou par M = [m,m,α, β], et un intervalle ordinaire [m1,m2] par M=[m1,m2,0,0]. Notons que m1 ou m2 peuvent prendre des valeurs infinies.
Les opérations arithmétiques
Pour deux intervalles flous M=[m1,m2,α, β] et N=[n1,n2,γ, δ], on peut accepter les

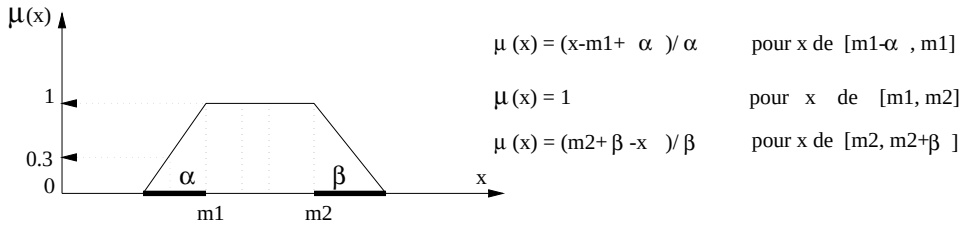


Figure 1.10: Un intervalle flou

opérations arithmétiques suivantes :

- $M \oplus N = [m1 + n1, m2 + n2, \alpha + \gamma, \beta + \delta]$;
- $M \ominus N = [m1 - n2, m2 - n1, \alpha + \delta, \beta + \gamma]$;

La multiplication est définie d'une manière fonctionnelle comme dans [DP87] par :
$\mu_{Q1} \otimes \mu_{Q2}(\mathrm{w}) = \sup\{\min(\mu_{Q1}(u), \mu_{Q2}(\mathrm{w/u})) \ / \ \mathrm{u} \in \Re\text{-}0 \ \}$ if w $\neq 0$
$= \max(\mu_{Q1}(0), \mu_{Q2}(0))$ if w=0
sera approchée par [MM96] :

- $M \otimes N = [min(S), max(S), abs(min(S') - min(S)), abs(max(S') - max(S))]$,
  pour S = [m1*n1,m1*n2,m2*n1,m2*n2] et $S' = [(m1 - \alpha) * (n1 - \gamma), (m1 - \alpha) * (n2 + \delta), (m2 + \beta) * (n1 - \gamma), (m2 + \beta) * (n2 + \delta)]$,

La division sera approchée de la même manière par [MM96] :

- $M \oslash N = [min(S), max(S), abs(min(S') - min(S)), abs(max(S') - max(S))]$,
  pour S = [m1/n1,m1/n2,m2/n1,m2/n2] et $S' = [(m1 - \alpha)/(n1 - \gamma), (m1 - \alpha)/(n2 + \delta), (m2 + \beta)/(n1 - \gamma), (m2 + \beta)/(n2 + \delta)]$, dans le cas où n1 et n2 sont non nuls et du même signe.

Dans les autres cas où $n1 < 0$ *et* $n2 > 0$, on aura deux intervalles comme résultat : $]\text{-}\infty, x, 0, \beta[$ et $]\mathrm{y}, +\infty, \alpha, 0[$, pour lesquels on distingue entre plusieurs cas :

1- $m1 < 0, m2 < 0$:

$]\text{-}\infty, m2/n2, 0, abs((m2 + \beta 2)/(\text{n2}+\beta 2)\text{-}(\text{m2}/\text{n2}))]$

$]\text{m2}/\text{n1}, +\infty, abs((m2 + \beta 1)/(\text{n1}\text{-}\alpha 2)\text{-}(\text{m2}/\text{n1})), 0]$

2- $m1 > 0, m2 > 0$:

$]\text{-}\infty, m1/n1, 0, abs((m1 - \alpha 1)/(\text{n1}\text{-}\alpha 2)\text{-}(\text{m1}/\text{n1}))]$

$]\text{m1}/\text{n2}, +\infty, abs((m1 - \alpha 1)/(\text{n2}+\beta 2)\text{-}(\text{m1}/\text{n2})), 0]$

3- $m1 < 0, m2 > 0$:

dans ce cas, les deux intervalles augmentent jusqu'à former $\Re$.

Il faut noter que ces opérations imposent un sens unique pour la résolution des équations floues. Par exemple, l'équation X$\oplus$A=B, n'a pas, en général, X = B$\ominus$A comme solution. Le problème ici, et partout avec les quantités floues, est l'imprécision qui change suivant le sens de l'opération.
Pour résoudre ce problème dans notre cas, où on utilise les intervalles flous, on a étudié les changements de l'imprécision d'après les relations fonctionnelles des opérations floues (voir [DP87] pour ces relations) et on a trouvé qu'on peut accepter les solutions suivantes :
- pour l'addition et la soustraction, l'imprécision de X dans X$\oplus$A=B ou dans X$\ominus$A=B reste l'addition des imprécisions de A et de B, c'est-à-dire que X=B$\ominus$A est une solution de la première équation.
- par contre, pour la multiplication et la division l'imprécision de X sera le maximum des deux imprécisions.

1.4.10 Conclusion

Dans ce chapitre nous avons présenté une étude comparative des différentes approches de l'IA, pour le test et le diagnostic des circuits analogiques. Cette étude de l'état de l'art nous a montré qu'il n'y a pas une approche unique qui peut seule résoudre le problème. L'approche à base de modèles profonds est nécessaire, l'approche qualitative n'est pas convenable à cause des particularités des CA (pourtant elle peut être

utile dans certains cas comme va le montrer cette thèse) et finalement, l'approche de la logique floue parait assez attirante et prometteuse.

Le chapitre suivant continue à présenter l'état de l'art en examinant les systèmes de test et diagnostic des CA. Une évaluation et une discussion de ces systèmes sont données, suivies par une étude primaire des capacités de l'approche floue, les idées essentielles ainsi que les premiers résultats qui ont motivé le travail.

Chapitre 2

Systèmes de diagnostic des CA : état de l'art

Chapitre 2

Systèmes de diagnostic des CA : état de l'art

Ce chapitre présente l'état de l'art des systèmes développés spécialement pour le test et le diagnostic des circuits analogiques. On va voir que la théorie du diagnostic à base de modèles profonds de deKleer (GDE) constituait la base de la plupart de travaux effectués dans le domaine des circuits électroniques.
Une discussion et une évaluation de chaque idée sont données, ainsi que des propositions de développement de ces systèmes pour les rendre plus efficaces en tenant compte des particularités des CA.

2.1 Le système expert DEDALE

Ce système [DRDM87] considère un mixage de deux approches, le raisonnement à base de modèles profonds et le raisonnement qualitatif.
DEDALE s'appuie sur le GDE (General Diagnostic Engine) de deKleer et Williams [KW87]. La solution proposée est d'introduire les ordres de grandeur [Rai86] [Rai91], à partir lesquels on effectue un raisonnement qualitatif dans la partie défaillante. Mais ce genre de raisonnement a nécessité l'utilisation d'une hypothèse assez forte : *un défaut cause des changements significatifs dans le comportement du dispositif* que l'on

peut exploiter en utilisant les ordres de grandeur.

On va parler un peu plus de DEDALE afin de pouvoir montrer, ensuite, la puissance de DEDALE-Flou dont les ordres de grandeur sont définis via des ensembles flous. DEDALE-Flou est une extension du système DEDALE que nous proposons dans ce document.

2.1.1 La modélisation qualitative dans DEDALE

Afin de prendre en compte les changements significatifs, la valeur de la quantité à considérer est constituée de son signe et de son ordre de grandeur relatif. Les ordres de grandeur apparaissent dans un espace de quantités fixe. Les expressions et les variables qualitatives sont reliées par les opérateurs de comparaison qualitatifs $\cong, \ll, \approx$ mutuellement exclusifs, définis comme suit [Rai91] :

$A \ll B$, signifie que A est négligeable devant B ;

$A \cong B$, signifie que A est très voisin de B ;

$A \approx B$, signifie que A a le même ordre de grandeur que B.

Les deux opérateurs $\cong$ *et* $\approx$ représentent des relations d'équivalence, et $\ll$ est un ordre partiel entre les classes d'équivalence de $\approx$: si $A \approx B$, alors $B \ll C$ implique que $A \ll C$. L'opérateur $\cong$ est plus précis que $\approx$: si $A \cong B$, alors $A \approx B$ et $(A-B) \ll B$.
Les relations définies entre des quantités en utilisant ces opérateurs, appelées *contraintes qualitatives*, ne considèrent pas les valeurs réelles car ces valeurs peuvent ne pas être connues ; elles considèrent des quantités exprimées en termes symboliques [DRD87]. Le système FOG (Formalisation du Raisonnement sur les Ordres de grandeur) a été intégré dans DEDALE. Il interprète symboliquement, à l'aide d'un ensemble de règles, la sémantique qualitative de ces opérateurs. Il faut noter que FOG enlève beaucoup de l'ambiguïté que l'utilisation de l'espace {-,0,+} introduit [Rai86]. Certaines limites de FOG ont été soulevées et analysées [HBC+91].
Enfin, notons que des travaux concernant les modèles d'ordres de grandeur montrent qu'on peut définir des structures d'algèbre qualitative [TMP89].

2.1.2 DEDALE-Flou

La section 1.3.6 présentait les ordres de grandeurs et discutait des problèmes tels que le problème de DEDALE. DEDALE souffre des composants qui travaillent aux limites de leurs comportements nominaux.
On peut démontrer que ce problème peut être résolu si on définit les opérateurs des ordres de grandeurs via des ensemble flous [MMNB95]. Cette définition est moins qualitative, ainsi elle permet un traitement plus précis des règles de FOG puisqu'on peut déterminer le degré d'appartenance du résultat d'une règle ; par exemple, si $(A \cong B)_{0.98}$ et $(B \cong C)_{0.93}$, alors $(A \cong C)_{0.93}$ qui est le minimum des deux degrés.
Ici, on n'a pas l'intention de développer un système de diagnostic semblable à DEDALE (pourtant, c'est une branche importante à développer). On veut juste montrer les capacités des ensembles flous pour pouvoir conclure à la fin que les ensembles flous constituent l'approche la plus générale.

La figure 2.1 présente une définition générale des opérateurs principaux. Les ensembles flous constituent une forme des ordres de grandeurs, mais une forme plus précise [MMT96] [MMNB95].

*** Les règles de FOG avec les ensembles flous**

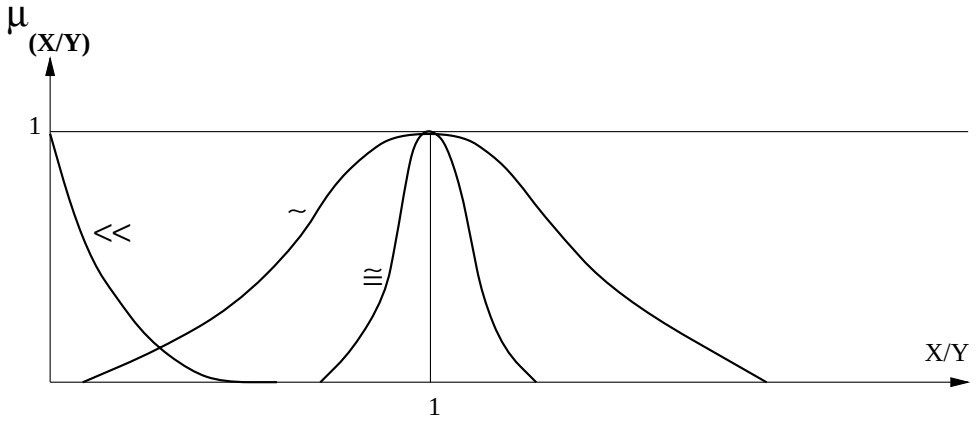


Figure 2.1: Les opérateurs des ordres de grandeurs via des ensembles flous

En pratique, les intervalles flous peuvent tout à fait servir à définir les opérateurs des ordres de grandeurs :

$A \ll B \Leftrightarrow$ négligeable(A,B) $\in$ [0.98,1.02,0.03,0.03]

$A \cong B \Leftrightarrow$ très-voisin(A,B) $\in$ [1,1,0.02,0.02]

$A \approx B \Leftrightarrow$ même-grandeur $\in$ [0.8,1.2,0.3,0.3]

On peut montrer que le raisonnement dans DEDALE-Flou est plus précis que dans DEDALE puisque la notion de degré d'appartenance est utilisée. Les comparaisons suivantes vont clarifier ce point :

* la relation (X≈Y),(Y≈Z)⇒(X≈Z)

qui est très ambiguë aura la forme suivante en DEDALE-Flou :

même-grandeur$(X,Y)_\alpha$, même-grandeur$(Y,Z)_\beta \Rightarrow$ même-grandeur$(X,Z)_\gamma$

où $\gamma = \alpha * \beta$

* la relation (X≅Y)⇔ (Y≅X)

aura la même forme :

très-voisin$(X,Y)_\alpha$ = très-voisin$(Y,X)_\alpha$

On peut voir que γ est plus petit que α *et* β ce que l'on démontre facilement par : X/Z = (X/Y) * (Y/Z).

Les autres relations du système FOG auront des versions semblables avec des degrés d'appartenance ce qui donne à DEDALE-Flou cette précision par rapport à DEDALE.

2.2 CATS et la propagation des intervalles

L'équipe de Ph. Dague [DDLT90b] [DDLT90a] [DJT90], a réalisé un programme (CATS) pour le diagnostic des systèmes analogiques. CATS introduit une séparation entre le module de description du système à diagnostiquer et le module de diagnostic afin de l'utiliser pour tout type de système physique analogique. La première application était le programme DIANA dans le domaine des CA.

*** CATS : le chercheur de conflits**

Afin de pouvoir représenter et gérer des grandeurs évoluant dans le temps, CATS [DJT90] utilise des tableaux d'*intervalles numériques* comme valeurs manipulées. Les tableaux sont utilisés pour conserver les résultats de l'échantillonnage des valeurs continues, et les intervalles pour exprimer l'incertitude inhérente aux mesures. CATS est

un programme de recherche de conflits, "ATMS-like" et indépendant du domaine. L'imprécision est exprimée par des intervalles numériques gérés avec soin afin de ne jamais produire de conflits à tort, en particulier : prendre en compte la précision de l'appareil de mesure et la dynamique de la valeur, et les procédures de prédiction effectuent les opérations arithmétiques dans les plus mauvais cas. Ces intervalles sont propagés suivant la propagation classique de Brown, Burton et deKleer [BBdK82]. Tandis que pour le choix de la mesure suivante, les auteurs ont choisi d'utiliser des heuristiques (liées à l'application) plutôt que les méthodes probabilistes proposées par deKleer.

Par rapport à DEDALE les avantages sont les suivants : le comportement nominal n'a plus à être décrit, le recours à une modélisation numérique améliore la finesse des pannes détectables et élimine l'hypothèse qu'une panne doit entraîner une variation "significative" du comportement du circuit, et le programme n'est plus limité aux pannes observables en continu [DDLT90b].

CATS introduit plusieurs idées intéressantes, mais on va montrer que les intervalles flous sont plus convenables pour la détection de fautes d'un coté, d'un autre coté, ils peuvent exprimer des valeurs qualitatives et quantitatives en même temps ce qui leur confère leur force.

2.3 FIS : Système d'Isolement de Fautes

Les constructeurs de FIS [PDS87] (Fault Isolation System) ont adopté un modèle simplifié de comportement qualitatif des modules à remplacer et de la structure du système sous test qui sont faciles à acquérir et peuvent être utilisés efficacement pour le diagnostic. Ces modèles portent le nom de "modèles causaux qualitatifs".

Un modèle causal qualitatif de l'UST (Unité Sous Test) est constitué de (1) Une description causale de l'ensemble de modules à remplacer (2) Une description de la connexion (structure) de ces modules, et (3) Un ensemble de tests possibles à partir duquel des séquences de diagnostic peuvent être construites. De plus, le modèle peut

contenir des estimations des taux d'échecs *a priori* et les coûts de tests à effectuer et des modules à remplacer.
Le problème avec ce genre de modélisation c'est qu'il est orienté systèmes, c'est à dire, qu'il n'est pas convenable ni suffisant pour le cas des circuits.

FIS utilise un algorithme probabiliste efficace pour trouver le meilleur point à tester. Mais l'approche probabiliste pose beaucoup de problèmes (Voir chapitre 4). De plus, l'approche qualitative considérée ici paraît trop qualitative pour pouvoir résoudre les problèmes des CA.

2.4 Quelques autres systèmes

2.4.1 Diagnostic des CA par l'utilisation des réseaux de neurones

Les réseaux de neurones ont été utilisés surtout pour leur capacité d'apprentissage et de classification. Wu et Meador [WM94] ont présenté une étude préliminaire suggérant l'utilisation de ces réseaux pour le diagnostic des CI. Pour leur part Spina et Updhyaya [SU92] ont utilisé cette technique pour réaliser des analyseurs de signatures. Finalement, Sutton [Sut92] a proposé de les utiliser combinés avec des règles d'inférence floues pour le diagnostic des circuits mixtes. Dans le cas où le premier diagnostic, effectué par les réseaux de neurones donne plusieurs composants supposés fautifs, des règles d'expertise floues sont utilisées afin de les distinguer entre eux. Par exemple, si le premier diagnostic indique que Cx et Cy sont fautifs. Les règles suivantes sont définies pour décider de changer Cx avant Cy. Il doit y avoir plusieurs raisons justifiant ce choix.

Règle 1 :

SI Cx = HI

ALORS OUTPUT = CxFAUTIF

Règle 2 :

SI Cx = LO ET Cy = HI

ALORS OUTPUT = CyFAUTIF

où HI, LO, CxFAUTIF et CyFAUTIF sont définis comme des ensembles flous.

L'étape d'apprentissage représente le goulot d'étranglement lors de l'utilisation de réseaux de neurones. Les données fournies doivent être suffisantes et bien choisies pour bien *apprendre* le réseaux. De ce fait, l'utilisation de ces réseaux était limitée à la classification. Les exemples utilisés étaient de petites tailles.

2.4.2 Diagnostic automatique des fautes dans les CA et Mixtes

McKeon et al. [MW91] utilisent la méthode de "suspension de contraintes" pour le diagnostic des CA.
La première étape de leur approche est d'appeler une heuristique de décomposition [SVCC77], pour essayer de réduire les régions contenant les composants suspects. La deuxième utilise la suspension de contraintes pour localiser les composants défectueux. Mais cette dernière méthode souffre des problèmes présentés dans la section 1.2 et de plus elle crée la nécessité d'effectuer beaucoup de mesures ce qui n'est pas très conseillé.

2.4.3 Conception en vue du test (DFT) par CLP($\Re$)

CLP($\Re$) "Constraint Logic Programming" est un langage de programmation logique, capable de résoudre des systèmes d'équations linéaires et d'inégalités. L'algorithme de diagnostic [NMSZB93] utilise différents modes de test et de DFT et des résultats de mesure de voltage pour des fréquences différentes, et calcule l'ensemble des composants suspects.
Le test interne ("in-circuit") et le test de fonctionnement sont établis pour le test des CA et digitaux. La première méthode essaie d'isoler les composants et de les tester individuellement. Pour arriver aux conditions désirées d'entrée pour un composant, souvent, il faut effectuer un "backdriving" sur les composants qui pouvaient y conduire. Mais le fonctionnement du circuit complet n'est pas testé, et comme les densités d'intégration augmentent, l'accès nécessaire aux conditions des composants devient de plus en plus difficile. Par contre, le test de fonctionnement est souvent préféré. Le test de fonctionnement d'un circuit est effectué en exerçant les fonctions

de composants de façon aussi précises que possible. Cette méthode est beaucoup plus difficile à implanter et la localisation des fautes est plus compliquée. La complexité de test des CA encourageait l'étude des principes de DFT pour les CA et mixtes. La plupart des circuits sont conçus par étapes afin de localiser les effets des composants. La motivation de ce travail était l'idée de l'utilisation de CLP($\Re$) pour la simulation et le diagnostic des CA [Som90].

L'approche considère *l'hypothèse de faute simple* à cause de l'utilisation de CLP($\Re$) aux systèmes linéaires ; le cas de fautes multiples rend les systèmes d'équations non linéaires, systèmes qui ne sont pas résolus par CLP($\Re$). Un problème plus pratique est celui de la sélection des fréquences de test qui doivent être déterminées par le concepteur et finalement les valeurs avec tolérances ne sont pas encore traitées.

2.4.4 DRAFTS : Un Simulateur Discret de Fautes pour les CA

DRAFTS [NCA93], est un simulateur de fautes efficace pour les CA linéaires. Une simulation de fautes est réalisée par abstraction du comportement du CA et sa transformation du domaine continu de Laplace au Z-domaine discret.
Mais cette transformation ne se fait pas sans problèmes :
Un des problèmes qui se pose à ce niveau, est qu'il n'y a pas une transformation ("mapping") univoque des fautes, car en transformant le circuit dans le Z-domaine, on peut avoir des fautes multiples correspondant à une faute simple du domaine continu.
DRAFTS a donné de bons résultats en comparaison de ceux obtenus par le simulateur PSPICE. Il y avait moins de 5% de différence d'erreur. De plus, il est beaucoup plus rapide. Un avantage de DRAFTS est qu'il est utilisable pour choisir les fréquences pour lesquelles il est utile de faire le test du circuit. Alors un système global utilisant DRAFTS d'abord, puis ensuite un simulateur, avec tolérances si possible, peut être très intéressant.

2.4.5 Prise de décision dans un environnement flou

Dans le domaine de conception électronique, par exemple, et avec l'explosion dans le nombre d'alternatives de conception, les concepteurs rencontrent le problème de prise de décision pour choisir la meilleure méthode DFT (Design-For-Testability). Les concepteurs ont besoin d'outils de CAO pour organiser, évaluer et choisir les techniques de test qui satisfont les objectifs de la conception. Mais, quand les objectifs ne sont pas définis d'une manière précise, le processus de prise de décision devient plus difficile à gérer. En effet, l'environnement devient flou et le processus lui même devient flou, i.e, il doit considérer des stratégies floues.
La prise de décision floue a été déjà abordée dans ce domaine [FK93], où les auteurs ont montré son utilité pour la vérification fonctionnelle de circuits et pour le choix de DFT approprié.
L'approche importante de Fares et al. est une des premières qui ait utilisé la logique floue dans ce domaine. Nous la développons dans le chapitre 5 dans sa forme généralisée où nous montrons son utilité pour le test et le diagnostic des circuits analogiques.

2.5 Discussion générale : vers la logique floue

Le test et le diagnostic des CA est une tâche très sensible. N'importe quelle approximation peut cacher des fautes ou créer des ambiguïtés.
DEDALE souffre des composants qui travaillent aux limites de leurs valeurs nominales. En fait, le concept des ordres de grandeurs n'est pas très convenable pour les CA, il constitue une approximation imprécise. L'opérateur $\approx$ peut être la source de problèmes (section 2.1.2).

L'utilisation des intervalles ordinaires pour exprimer l'imprécision et pour tenir compte des tolérances est une bonne méthode, mais non suffisante, car la détection de faute peut tomber dans le même piège où des composants travaillent aux limites de leurs comportements nominaux. On va montrer que les intervalles flous sont les plus généraux, ils offrent la possibilité d'exprimer l'imprécision dans ses différents niveaux,

de réduire l'explosion qui peut avoir lieu avec les intervalles ordinaires, et d'exprimer l'incertitude. Ces deux concepts, l'incertitude et l'imprécision, qui sont deux concepts différents, sont généralement confondus chez les électroniciens seront bien distingués dans les chapitres qui suivent.

Les intervalles ordinaires contiennent tous les genres d'imprécisions sans distinction, ce qui peut provoquer l'explosion dans les valeurs propagées dans le circuit. L'exemple suivant (figure 2.2), où le circuit a été emprunté à [MW89b], montre la différence entre les valeurs exactes et celles avec tolérances. Le circuit est composé de trois amplificateurs de valeurs d'amplification 1,2 et 3 comme le montre le schéma de la même figure. Ainsi, pour des tolérances prises de 5% pour les composants et de 5% pour les mesures, la sortie **C**, par exemple, aura comme valeur avec tolérances [5.46,6.56], pourtant il y a juste un passage par des multiplieurs. Notre idée est qu'une valeur

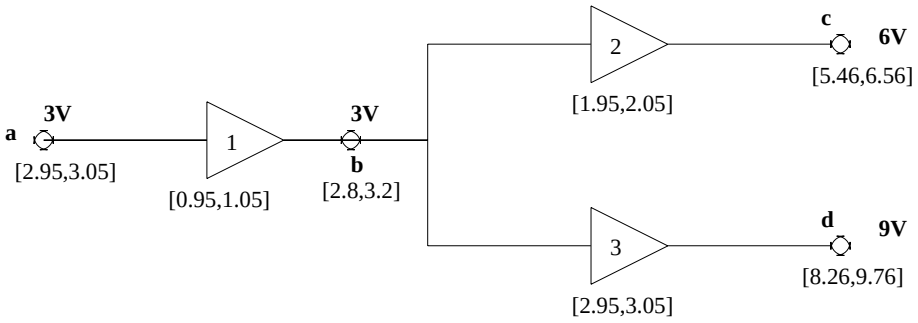


Figure 2.2: La propagation des intervalles ordinaires

qui dépasse les bornes d'un intervalle ordinaire sera considérée comme fautive, mais probablement juste avec les ordres de grandeurs. Avec les intervalles flous, elle sera considérée comme fautive avec un certain degré d'appartenance (entre 0 et 1). Par exemple, prenons le circuit simple de la figure 2.3 (une résistance de 2KΩ), qui travaille à la limite de son comportement nominal.

Pour une tolérance de 0.1kΩ, **I** aura comme valeur acceptable [2.38,2.63] mA. Maintenant, si **R** est fautif avec une résistance de 2.12 KΩ, alors **I** sera 2.36 mA. Cette valeur est fautive avec les intervalles ordinaires, bonne avec les ordres de grandeurs, mais fautive (ou bonne) avec un degré d'appartenance dépendant des valeurs d'entrées, avec les intervalles flous.

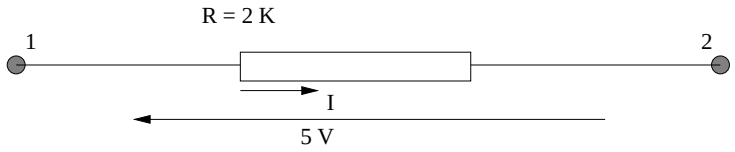


Figure 2.3: Le comportement aux limites de valeurs nominales

On propose l'utilisation de la logique floue où les intervalles flous remplacent les intervalles ordinaires pour représenter l'imprécision des CA. Cette représentation est la plus générale puisqu'elle qu'elle représente la connaissance incorporée aux cotés souples des intervalles et qu'il n'y a pas d'exclusion de valeurs, ce qui constitue un aspect important des ensembles flous et important pour capturer l'intuition de bon sens. Elle permet de distinguer entre différents types d'imprécision, celle de la fabrication, celle des experts et celle des instruments. On peut, par exemple, considérer la valeur 3 ± 0.05 comme un nombre flou [3,3,0.05,0.05], ainsi 3.05 et 2.95 seront les valeurs les plus mauvaises. Prenons l'exemple de la figure 2.2 pour montrer cela clairement [MMNB95] :
(a) Va peut être pris comme un nombre flou [3,3,0.05,0.05] ou comme un intervalle flou avec des valeurs différentes [2.98,3.02,0.03,0.03]
(b) On peut représenter les amplificateurs comme des nombres flous, comme des intervalles flous, ou simplement comme des intervalles ordinaires [0.95,1.05,0,0]
Ce qu'il faut voir ici, c'est que la même représentation permet d'exprimer tous les autres types. D'ailleurs, les ensembles flous permettent de définir des probabilités floues au lieu des probabilités *a priori* des composants fautifs. Cette dernière idée, nous l'avons utilisée pour développer une méthodologie à base de logique floue pour trouver le meilleur point à tester comme montrera le chapitre 4.

Comme exemple de démarrage, on va étudier l'exemple suivant (figure 2.4) : On va comparer les deux cas, intervalles ordinaires et intervalles flous.
1- le premier cas avec des intervalles ordinaires : r1[99,100], et r2[0.99,1.01] (tolérance 1%).
2- le deuxième cas où les intervalles flous sont considérés : r1[99.5,100.5,0.6,0.6], et r2[0.995,1.005,0.01,0.01]. Notons que, avec les intervalles flous, on peut prendre

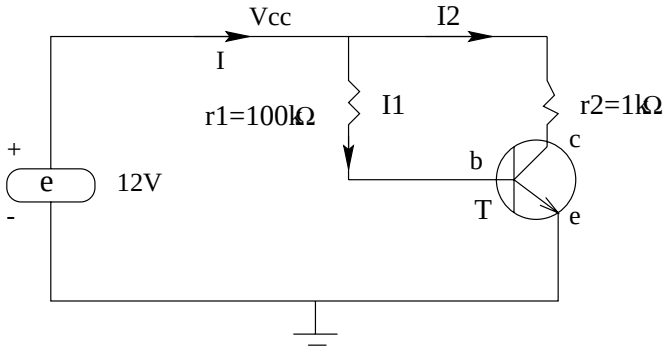


Figure 2.4: Un circuit simple

toutes les valeurs possibles. On n'est pas obligé de traiter seulement les tolérances ou de couper l'ensemble de valeurs d'une façon abrupte.

Les valeurs nominales seront (on a pris V, mA,et kΩ comme unités) :

Ic[11.04, 11.57]	Ic[11.14,11.46,0.2,0.2]
Vc[0.20 , 1.17]	Vc[0.44,0.96,0.32,0.034]
(1)	(2)

Chaque valeur de l'intervalle flou a un degré d'appartenance qui, pour nous, donne son degré d'acceptation.
Si les mesures donnent Vc=1.1V, les propagations inverses donneront : Ic[10.69,11.11] avec les intervalles ordinaires et Ic[10.8,11.01,0.16,0.16] avec les intervalles flous.
Ces résultats, donnés sur la figure 2.5, montrent que la prise de décision dans le cas des intervalles flous est plus simple. Un *nogood* (l'ensemble de composants qui ont probablement produit la faute) avec un degré entre 0 et 1 apparaît. Ce degré donne la gravité de la faute. Dans le cas de notre exemple, une faute de degré 0.35 sera annoncée (bon au degré 0.65) ce qui peut signifier qu'une déviation existe mais la valeur calculée reste très proche de la valeur nominale.

Les intervalles flous peuvent former une base mathématique très puissante pour la représentation des différentes informations, qualitatives ou quantitatives. D'ailleurs, ils permettent de traiter les différents types de fautes, paramétriques ou catastrophiques sans distinction. Balivada et al. [BA96] ont classifié les fautes des circuits analogiques

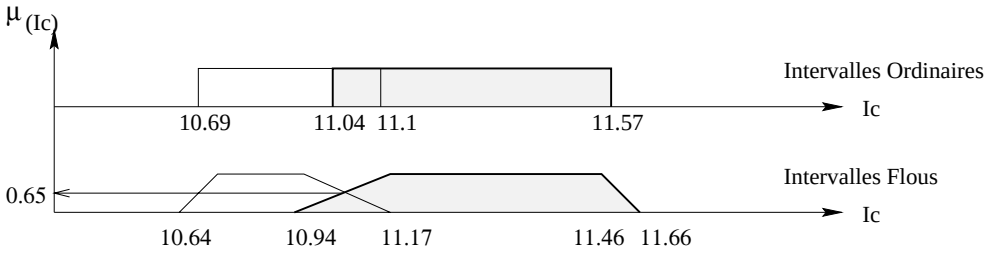


Figure 2.5: Des intervalle ordinaires ou des intervalle flous

et mixtes (figure 2.6), leur classification se croise avec la classification naturelle des intervalles flous comme le montre la même figure.

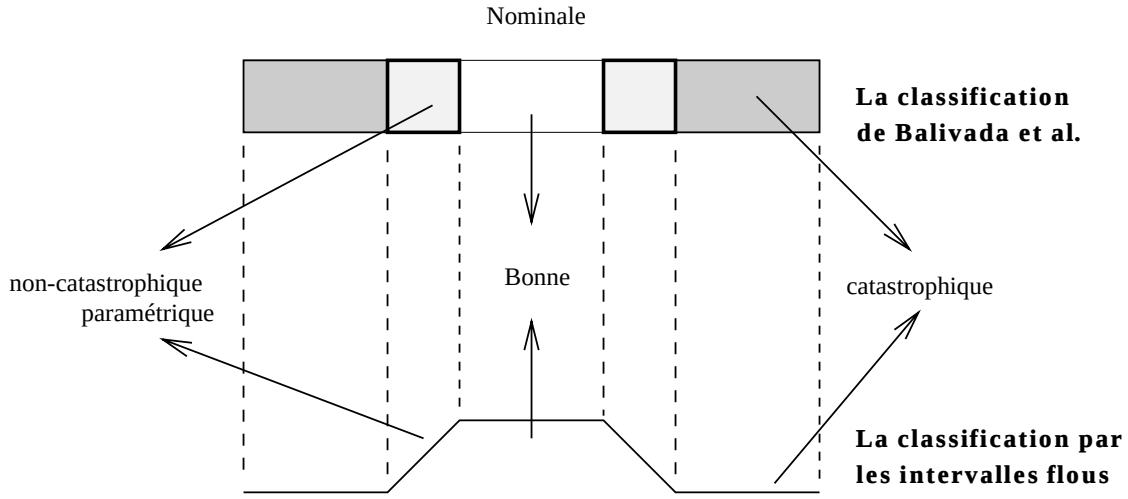


Figure 2.6: Taxinomie de fautes des CA et mixtes

2.6 Conclusion

Les approches qualitatives sont trop qualitatives pour être capables de résoudre les problèmes des circuits analogiques et mixtes. L'approche à base de modèles profonds est nécessaire, mais pas suffisante. L'approche de la logique floue peut améliorer le diagnostic dans certaines applications.

L'application, qui est la clé principale, détermine l'approche à utiliser.

Il y a deux points importants à signaler :

* ces approches ne sont pas mutuellement exclusives ;

* le système développé doit être orienté application.

Bien sur, un système général pourrait être construit, mais son efficacité en serait diminuée.

D'après toute cette étude détaillée de l'état de l'art du test et du diagnostic des circuits analogiques, et l'étude comparative des approches déjà utilisées ou des nouvelles approches, nous avons décidé de construire un nouveau système expert à base de logique floue et de l'approche modèles profonds, qui n'exclut pas pour autant l'approche qualitative quand celle-ci permet d'améliorer le diagnostic, d'une part, et quand il est difficile de fournir au système des données quantitativement précises d'autre part.
Dans le prochain chapitre, nous allons présenter notre système, nommé FLAMES : "A Fuzzy Logic ATMS and Model-based Expert System".

Chapitre 3

FLAMES : A Fuzzy Logic ATMS and Model-based Expert System

Chapitre 3

Le système FLAMES : "A Fuzzy Logic ATMS and Model-based Expert System"

D'après l'étude détaillée du chapitre précédent nous avons décidé de concevoir un système expert, mais à base de modèles profonds pour le test et le diagnostic des CA et mixtes. Toutes les unités de ce systèmes vont être construites dans l'environnement des CA et mixtes, mais ceci n'empêche pas de l'utiliser pour d'autres types de systèmes dynamiques.
Pour ce faire, il suffirait d'ajouter les modèles du nouveau système et peut-être quelques règles spécifiques pour aider le diagnostic.

3.1 Présentation générale

FLAMES est un système expert pour le test et le diagnostic des circuits analogiques et mixtes à base de logique floue, de modèles profonds, et d'un ATMS-flou. Ses composants principaux sont (figure 3.1) :

- un ATMS-flou (le noyau de FLAMES) qui propage des intervalles flous et des hypothèses floues (définies comme des ensembles flous, comme on le verra plus loin)

- une base de données de modèles qui sera utilisée pour un diagnostic à base de la structure et des modèles corrects des composants d'une part, et des hypothèses exprimant l'absence de faute dans les composants et l'exactitude des observations d'autre part

- une base de connaissances contenant des règles qualitatives et des modèles de fautes des composants pour aider le processus de diagnostic. C'est là où l'expertise humaine doit être principalement résumée. Cette expertise est très utile pour que le diagnostic soit efficace.

- une unité de stratégies de recherche qui aident à trouver le meilleur (au moindre coût) nœud à tester dans le cas où le processus de diagnostic nécessite plus d'information. Si le premier diagnostic donne un grand nombre de candidats, il sera nécessaire de faire plus de tests afin de discriminer ces candidats et pouvoir localiser le(s) composant(s) fautif(s).

- un module d'apprentissage dans lequel le système pourrait utiliser son diagnostic précédent pour apprendre par expérience et éviter de répéter le même travail plusieurs fois. Ainsi, FLAMES devient plus efficace.

- une unité de prise de décisions floue à utiliser pour le test des circuits mixtes. Elle utilise l'unité de stratégies de recherche, qui a été utilisée avec succès pour le diagnostic de circuits analogiques et digitaux et des circuits mixtes.

- une interface graphique qui facilite l'interaction entre l'utilisateur et le système. Elle est responsable de la lecture et de la traduction du circuit introduit dans un format de structures de données, de l'acquisition des commandes à exécuter, et de l'affichage sur l'écran des résultats obtenus.

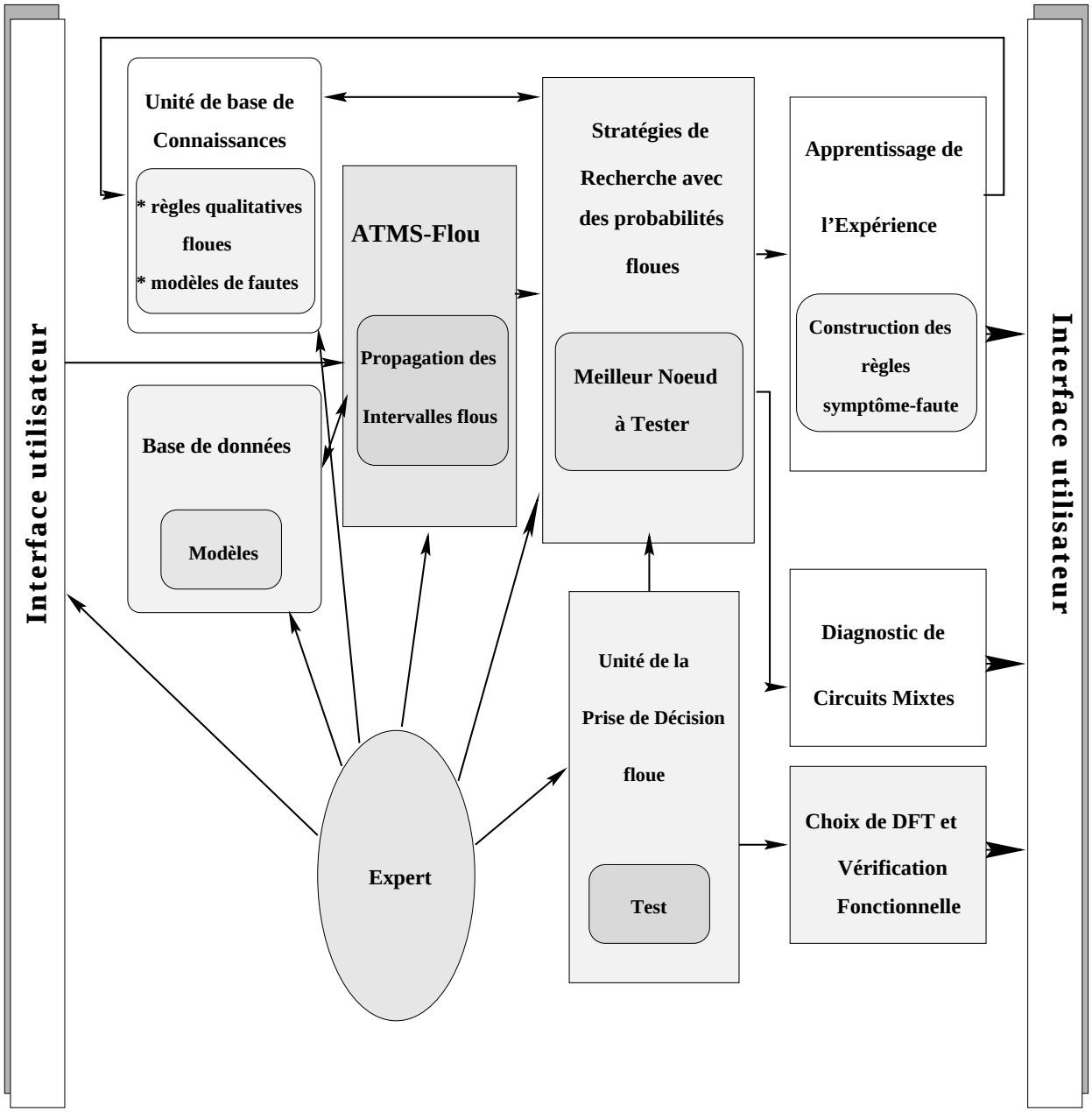


Figure 3.1: Le système FLAMES

Finalement, puisque l'expert humain est irremplaçable, on a décidé de construire FLAMES comme un système ouvert, ainsi un expert peut interagir avec chacune de ses unités.

Il est clair que FLAMES est principalement basé sur les principes de la logique floue, et ceci a plusieurs avantages (autres que ceux indiqués précédemment) : ceci permet une représentation précise du CA sous test, et une représentation simple mais précise (appelée semi-qualitative) de l'expertise humaine, soit des estimations à priori de fautes, soit de la méthode à considérer pour la mise à jour de ces estimations après les tests. Ce problème de représentation de connaissances, qui est très lié au problème d'acquisition de connaissances, est crucial dans les applications d'IA. Le choix d'une représentation qualitative seule n'est pas très conseillé pour les circuits analogiques (section 1.3), et celui d'une représentation quantitative seule n'est pas conseillé non plus. Ainsi, la logique floue, qui est capable de représenter les deux types, est choisie dans FLAMES.

FLAMES procède comme suit :

l'ATMS-flou prend les modèles et les valeurs mesurées comme entrée, il propage ces valeurs afin de trouver des divergences avec les valeurs prévues si elles existent. Son résultat est formé des ensembles de candidats avec des degrés de cohérence. A cette étape, une procédure de raffinement commence : des comparaisons entre les ensembles de candidats (des heuristiques) pour essayer de réduire leur nombre ; ensuite, l'unité de choix de la meilleure stratégie pour le test démarre avec les candidats restants et la base de connaissances pour essayer de raffiner cet ensemble en proposant à l'expert une liste triée (à l'aide d'une fonction de coût) de points de tests afin de se rapprocher de(s) composant(s) fautif(s). Une fois la (les) faute(s) trouvée(s), une règle *symptôme-faute* avec un degré de certitude est construite et ajoutée à la base de connaissances pour aider au futur diagnostic.

3.2 L'ATMS-flou : la propagation des intervalles flous

Les applications classiques d'IA comme la planification, le diagnostic, etc. impliquent un raisonnement dans un univers incomplet ou incertain. Pour faire face à

l'incertitude et à l'incomplétude, les systèmes d'IA doivent être capables de choisir entre plusieurs solutions, décisions, voire croyances. Plusieurs critères peuvent être considérés : degrés d'incertitudes (probabilités, mesures de nécessité, etc.), degrés d'utilité, les préférences de l'utilisateur (ou l'expert), etc. [CRS92].

Dans FLAMES, un ATMS un peu différent de celui de deKleer [Kle86] a été développé. Nommé ATMS-flou, il incorpore dans ses mécanismes un traitement de l'imprécision par la propagation des intervalles flous et un traitement de l'incertitude en calculant un degré de cohérence entre les valeurs nominales et celles mesurées. Ce degré de cohérence (Dc), qui sera calculé comme l'intersection entre des intervalles flous, peut signifier la certitude que la valeur mesurée est bonne (comme montreront les sections suivantes). En outre, cet ATMS-flou n'ignore pas l'avis de l'expert, en tenant compte de ses préférences dans le processus de prise de décision.
Ainsi, cet ATMS-flou apparaît comme l'ATMS possibiliste de Dubois et al. [DLP90], où les clauses sont incertaines (Voir annexe A). Notons que l'idée de l'utilisation de l'ATMS possibiliste pour le diagnostic des circuits analogiques a été reprise dans [BP95] où il a montré qu'il peut apporter une amélioration au diagnostic par rapport à l'utilisation d'un ATMS classique.
L'ATMS est un système de maintien de vérité basé sur la manipulation d'ensembles d'hypothèses. Il est capable de travailler efficacement avec des informations incohérentes, il évite la restriction usuelle qu'un seul point de l'espace de recherche peut être examiné à la fois et finalement la plupart des retours en arrière de TMS [Doy79] est ainsi évitée aussi.
Le raisonnement sur les hypothèses est très général. En électronique, par exemple, une hypothèse pourrait être le fonctionnement correct de chaque composant [Kle86]. A noter que l'ATMS est nécessaire parce qu'on envisage la possibilité de fautes multiples où l'espace de candidats potentiels augmente exponentiellement avec le nombre de fautes considérées (on trouvera plus de détails sur les ATMS en annexe de ce document).

Dans notre approche, les résultats du diagnostic, constitués d'ensembles de candidats, seront qualifiés par degrés indiquant leur gravité. Un expert peut utiliser cette information supplémentaire, en plus de son expérience, pour choisir entre les candidats.
Quand cela est possible, le diagnostic peut être aidé par quelques règles qui décrivent l'unité sous test, son comportement qualitatif correct et celui fautif.

3.2.1 Un moteur de reconnaissance de conflits

La tâche centrale du diagnostic est de détecter les divergences entre les valeurs prévues et les valeurs mesurées et de construire les ensembles de candidats qui justifient ces divergences ; l'étape importante de la détection de fautes est mieux réalisée par l'utilisation des intervalles flous (voir section 2.5). Les divergences sont détectées et les "nogoods" minimaux correspondants (un nogood est l'ensemble d'hypothèses qui justifient la faute) et les candidats minimaux sont construits.

Maintenant, nous allons présenter un moteur de reconnaissance de conflits flou pour diagnostiquer les circuits analogiques (il peut être très utile pour d'autres systèmes dynamiques), dont les valeurs sont représentées par des intervalles flous. Mais les variations des quantités physiques avec le temps ne sont pas traitées pour le moment.

3.2.2 La propagation des intervalles flous

Une étude détaillée sur les langages d'étiquettes (label languages) peut être trouvée dans [Dav87]. Ici, on se concentre sur l'étiquetage d'intervalles flous (une étiquette pour une quantité réfère à l'ensemble de valeurs possibles que cette quantité peut prendre et qu'il ne faut pas confondre avec l'étiquette d'hypothèses qui peut être attachée à chaque valeur particulière).

Les valeurs de quantités prennent deux formes : valeurs prévues (des modèles), et valeurs mesurées. Une quantité floue sera propagée chaque fois qu'une valeur est entrée. La propagation se déroule à travers les contraintes qui constituent le modèle

du circuit (des opérations floues sont considérées). La découverte d'une valeur pour un nœud pour lequel on connaît déjà une valeur propagée (prévue) est appelée une *coïncidence*. La figure 3.2 résume les cas possibles de coïncidence dans les deux cas des intervalles flous ou ordinaires. Pour rendre ce problème plus compréhensible, nous

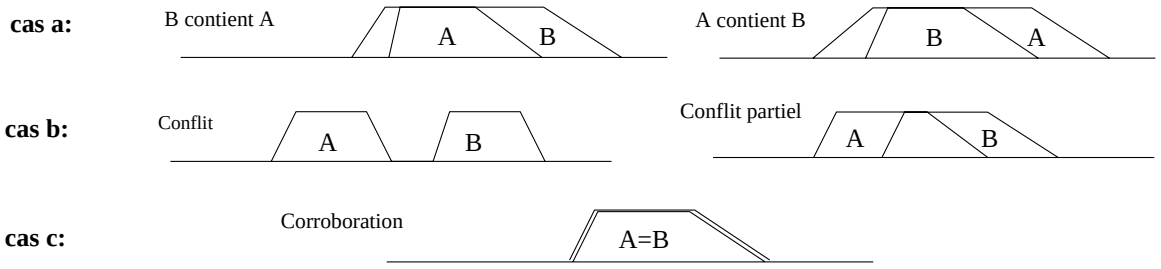


Figure 3.2: Les cas possibles de coïncidences

allons donner une description résumée d'un système de contraintes et de quantités. Un système général sophistiqué peut traiter des valeurs d'entrée sans distinction ; on peut le voir comme un ensemble de quantités (V1, V2, I1, I2, ...), des règles (contraintes entre quantités), et un noyau qui utilise les contraintes pour calculer des valeurs pour ces quantités chaque fois qu'il y a une nouvelle valeur mesurée ou qu'une valeur est entrée.
Un ATMS est aussi nécessaire pour manipuler les hypothèses. Un tel système est lourd, il peut exploser à cause du grand nombre de candidats et il est très probable de produire des conflits entre les valeurs, ce qui peut être évité par un traitement long et spécial [DJT90] (mais ce qui est naturellement traité dans notre ATMS-flou).

Une autre représentation plus simple de ce système peut être utilisée, lorsqu'on distingue la phase du calcul de toutes les valeurs possibles (des modèles corrects) et la phase de diagnostic qui utilise les mesures. Dans cette version de notre système, nous considérons la dernière approche : nous ne considérons qu'un modèle correct avec ses valeurs prévues.

Le cas des coïncidences (figure 3.2) entre deux valeurs propagées est le plus compliqué, car il doit prendre en considération les hypothèses des deux propagations. La figure 3.3 montre un tel circuit où deux valeurs sont propagées sur des chemins

différents avec des ensembles d'hypothèses différentes.
En général [BBdK82], dans le cas d'un conflit, la liste des candidats peut être réduite à l'union des deux ensembles d'hypothèses. Mais le cas de corroboration est plus compliqué, on doit raisonner sur l'intersection de leurs hypothèses non vérifiées. Si cet ensemble est vide, une corroboration indique la correction des valeurs propagées ; dans l'autre cas, la corroboration devient très difficile parce qu'une faute dans l'intersection pourrait signifier que les deux chemins sont fautifs.

En tout cas, raisonner sur l'intersection est dangereux, cela peut causer une perte d'informations, et raisonner sur l'union peut être très lourd. Notre processus de *résolution de conflits* consiste à considérer qu'une coïncidence entre deux valeurs propagées n'est qu'une coïncidence entre chacune d'entre elles et la valeur nominale. Par conséquent, il raisonne sur un degré de cohérence Dc (comme on va le voir dans la section suivante) et une attention particulière devrait être donnée au chemin du degré le plus petit (figure 3.3). Le raisonnement sur Dc élimine la plupart des problèmes indiqués, d'autant plus que nous pouvons tirer profit de la capacité de l'ATMS à gérer les hypothèses en parallèle.

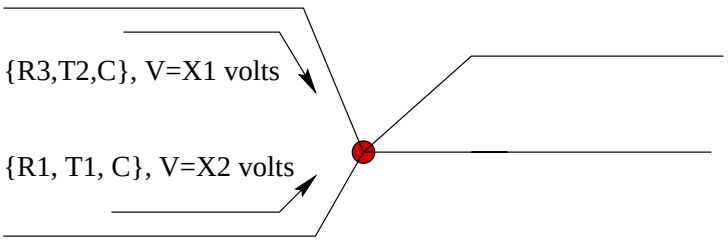


Figure 3.3: Coïncidence entre deux valeurs propagées

3.2.3 ATMS-flou

Les différents cas de la figure 3.2 seront résumés par un degré de cohérence entre la valeur nominale et la valeur mesurée.
L'idée est que si on décide que Vn est la valeur nominale de la quantité X et qu'une mesure a donné Vm pour X, alors afin de clarifier la situation il faut évaluer la proposition $X \in Vn$ (rappel : on raisonne sur des intervalles flous).

En général on peut dire que,

si Vm $\subseteq$ Vn alors la proposition est nécessairement vraie ;
sinon
si Vn $\cap$ Vm $\neq$ 0 alors elle est seulement possiblement vraie.

Dc, qui $\in$ [0,1], est défini comme le degré de cohérence entre Vm et Vn de façon topologique par la formule suivante :

$$\mathbf{Dc = surface(Vm \sqcap Vn) \ / \ surface(Vm)}$$

Mais, cette définition pose encore un petit problème à régler avant de continuer le travail. C'est que la multiplication floue, par exemple, de deux intervalles flous donne un ensemble flou c'est-à-dire un intervalle dont les côtés sont des courbes (c'est valable pour les autres opérations non linéaires). En effet, il y a une perte d'informations quand on fait l'approximation, une perte qu'il faut quantifier par un calcul intégral qui peut rendre notre calcul total un peu long. Mais on peut remarquer que la

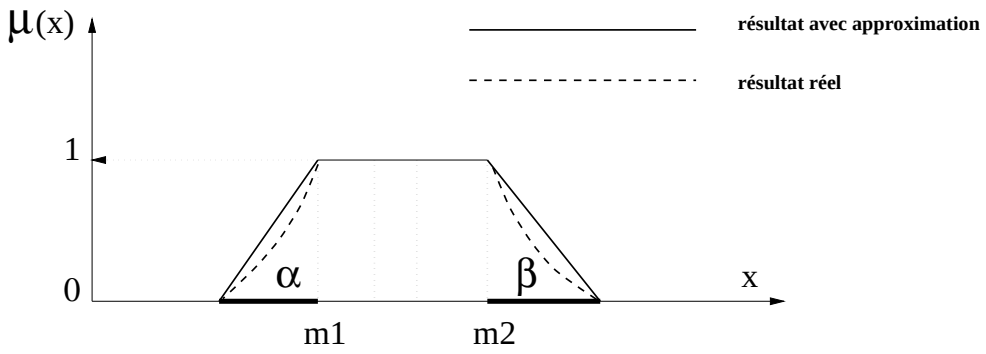


Figure 3.4: Le résultat d'une multiplication de deux intervalles flous

représentation paramétrique qu'on a choisi peut aider à résoudre ce problème puisque d'une part seulement les quatre paramètres considérés sont ceux de la base, et d'autre part la hauteur de l'intervalle flou est 1 ce qui rend le calcul de la surface équivalent

au calcul entre ces paramètres, comme va le montrer l'étude suivante.

$$Dc = \begin{cases} 1 & \text{si } Vm \subseteq Vn \\ 0 & \text{si } Vm \bigcap Vn = \emptyset \\ < 1 & \text{si } Vm \bigcap Vn \neq \emptyset \end{cases}$$

Comme Vm et Vn sont définis en termes d'intervalles flous (comme des distributions de possibilités en logique possibiliste), la justification de ce choix est que le degré de certitude que la quantité X prenne une valeur $\in$ Vn est cette intersection entre sa valeur propagée (la valeur possible actuelle) et sa valeur nominale (la valeur possible nominale).

Il faut dire que le calcul pratique de Dc n'est pas lourd car toutes nos valeurs sont des intervalles flous normalisés ($\Leftrightarrow$ il y a au moins un élément qui a 1 comme degré d'appartenance), ainsi la surface d'un intervalle flou [x,y,α, β] se calcule par :

$$\text{(y-x)} * 1 + (\beta * 1 + \alpha * 1)/2$$

qui équivaut à

$$\text{(y-x)} + (\beta + \alpha)/2$$

comme le montre la figure 3.5. En pratique on distingue entre +Dc et -Dc où le signe

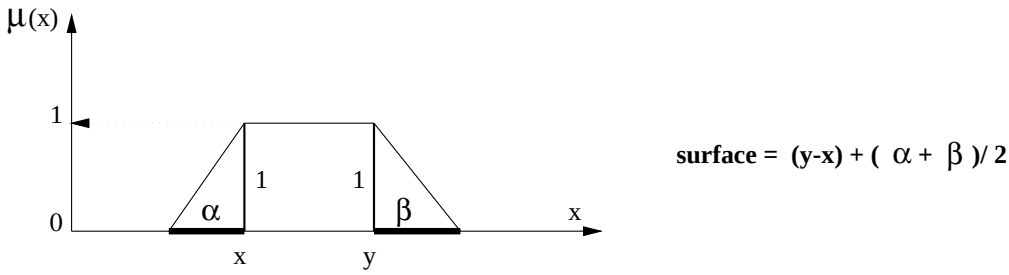


Figure 3.5: La surface d'un intervalle flou

indique si Vm est située à droite ou à gauche de Vn, respectivement. Ce signe aide à

indiquer le mode de défaillance de composant quand c'est possible (*high* ou *low*, par exemple).

Le traitement des circuits analogiques est très délicat : un composant, par exemple, peut être fautif sans manifester de symptômes. Les corroborations (cas c : figure 3.2) n'impliquent pas toujours que les composants impliqués sont non fautifs. Un conflit (cas b) indique un nogood avec un degré 1, et un conflit partiel indique un nogood de degré < 1.

La résolution se déroule en deux étapes : la propagation qui donne comme résultat une conjonction d'hypothèses avec leurs degrés de cohérence, et une seconde étape qui cherche les environnements contradictoires. Dans cet ATMS-flou, les clauses ne sont pas réduites à celles de Hörn (comme dans [DLP90]) et toutes les informations sont représentées dans le cadre de la logique possibiliste. Ainsi, il permet à l'expert d'ajouter des estimations de fautes ou de construire des modèles de fautes de composants avec des degrés de certitude.

Un autre avantage de cette méthode, qu'il faut noter ici, est la possibilité de donner à l'utilisateur une liste de *nogoods* triés suivant leurs Dc ce qui permet de réduire, voire limiter, l'effet d'explosion.
Mais, une question importante se pose ici : comment et quand ce processus s'arrête, puisqu'on travaille dans un environnent imprécis (le niveau imprécis de l'ordinateur). Pour être plus clair, les approximations que l'ordinateur effectue avec les problèmes de calcul d'intervalles peuvent impliquer que deux résultats, qui devraient être égaux, diffèrent ! Ceci crée le besoin d'une condition d'arrêt du processus.
L'ATMS-flou tourne indéfiniment tant que la propagation dans le circuit est possible. Dans chaque branche et à chaque nœud, de nouveaux ensembles de candidats sont calculés avec leurs degrés Dc attachés. Le problème est que même pour un seul nouvel ensemble trouvé, la propagation doit reprendre. Ainsi, la propagation s'arrête quand il n'y a plus de nouveaux candidats.
Un autre problème plus délicat se pose, comment déterminer si l'ensemble de candidats

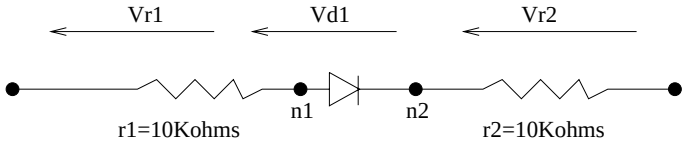
calculé n'est pas déjà trouvé ? Pour différencier deux ensembles, deux conditions doivent être vérifiées : la première est que les deux ensembles doivent contenir les mêmes éléments (candidats) et la deuxième est qu'ils doivent avoir le même degré Dc. C'est la deuxième condition qui pose problème et nous étions obligés de considérer une condition définie d'une manière qualitative floue parce que si deux ensembles ont respectivement 0.64 et 0.63 comme degrés de cohérence, on peut attribuer la différence aux problèmes d'approximations mentionnés auparavant et accepter qu'ils sont les mêmes.
Mais jusqu'à quel degré peut-on être tolérant ? Car accepter des approximations revient au même que de définir des opérateurs d'ordre de grandeurs ! Nous avons donc décidé de définir une égalité floue comme on le verra plus loin.

Finalement, notons que notre degré de cohérence est compatible avec la définition des ensembles flous. Il est probable que gérer en plus, à coté de Dc, un autre degré qui indique la distance entre la valeur prévue et la valeur nominale peut être utile, puisque les modèles des composants contiennent des inégalités : e.g., si la tension base-émetteur propagée est au-dessous d'un seuil, le fait que le courant de collecteur est indépendamment déterminé comme égal à zéro n'est pas suffisant pour corroborer les hypothèses, puisque la tension de la base peut être différente (mais au dessous du seuil).

3.2.4 Exemple explicatif

Le système DIANA [DJD+91] utilise aussi une extension d'un ATMS qui considère la propagation des intervalles ordinaires afin de détecter l'incohérence entre les valeurs et de construire les ensembles de candidats. L'exemple suivant (figure 3.6) [DDLT90b] peut expliquer notre vue de ce problème en compatibilité avec la logique floue.
Pour être capable de capturer le plus faible changement des valeurs, nous allons définir les opérateurs arithmétiques et logiques de manière floue. Ainsi, la condition ≤ 100 (figure 3.6) sera définie par [-∞,100,0,10] (figure 3.7), par exemple, et dans ce cas si les mesures donnent : Vr1 = 1.05 V alors Ir1 = 105 mA et Nogood{r1,d1} aura comme

degré d'appartenance 1-0.5 = 0.5 (il est bon à 0.5 degré). D'une manière semblable, pour Vr2 = 2 V, on aura Nogood{r2,d1} avec 1 comme degré d'appartenance (bon pour 0 degré). Les candidats dans ce cas sont $[r1, d1]_{0.5}, [r2, d1]_1$ ce qui indique un certain ordre entre eux. L'expert pourrait utiliser cet ordre pour se concentrer plus



Mesures	**Modèle**	**Prédiction**	**Hypothèse**
Vd1=0.2 V	Le modèle de Diode	Id1 =< 100 microA	{d1}
	Lois de Kirchhoff en n1	Ir1 =< 100 microA	{d1, r1}
	Lois de Kirchhoff en n2	Ir2 =< 100 microA	{d1, r2}
Vr1=1.05 V	Lois d'Ohm	Ir1 = 105 microA	{r1}
	conflit sur Ir1 -----> Nogood{r1,d1}		
Vr2=2 V	Lois d'Ohm	Ir2 =200 microA	{r2}
	conflit sur Ir2 -----> Nogood{r2,d1}		
Le calcul de candidats donnera finalement →			**CANDIDATS : [d1] or [r1,r2]**

Figure 3.6: Candidats avec des intervalles ordinaires

sur [r2,d1], ou bien il pourrait utiliser les estimations à priori pour décider si la diode ou une des deux résistances est fautive. En tout cas, considérer les modes de fautes de la diode (coupure ou court-circuit), nous entraîne à fortement suspecter la résistance r2.

Dans le cas où les intervalles et les opérations ordinaires sont utilisés, on pourrait seulement suspecter les trois composants avec le même poids.

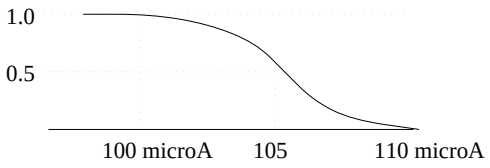


Figure 3.7: L'opération floue :$\leq$ 100

3.2.5 Algorithme général

3.2.5.1 *Sémantique de données*

Une quantité est une paire (Q,F) où Q est son nom et F est l'intervalle flou qui contient sa valeur exacte. Dans notre approche, la valeur de Q appartient à F avec un degré d'appartenance.

Les règles suivantes décrivent la sémantique de données :

* la règle d'intersection qui est la plus utilisée et qui constitue la base du diagnostic :

(Q,Fm),(Q,Fn)→ $(Q, Fm)_{Dc(Fm,Fn)}$

explique le résultat d'intersection entre une valeur nominale et une valeur propagée (ou mesurée) ; on calcule le degré Dc entre les deux valeurs qu'on propage dans le circuit.

* la règle de "Nogood", qui résulte d'une intersection vide et découvre un "Nogood" :

$(Q, Fm)_{Dc=0} \rightarrow \perp$

quand l'intersection est vide la valeur propagée est fautive (son Dc = 0)

* la règle d'inclusion qui signifie une implication logique :

(Q,Fm),(Q,Fn), $Fm \subset Fn \rightarrow (Q, Fm)_{Dc=1}, (Q, Fm) \Rightarrow (Q, Fn))$

c'est quand pour la même quantité, une valeur propagée est incluse dans une autre valeur déjà trouvée ; dans ce cas Dc = 1 et il vaut mieux éliminer Fn, qui est redondante, et propager Fm (principe de *minimalité*).

On peut facilement remarquer que la deuxième et la troisième règles sont des cas particuliers de la première.

3.2.5.2 *Couplage avec l'ATMS*

Une quantité, dans l'ATMS, est liée à un environnement, c'est-à-dire qu'elle est définie par ((Q,F),E) où F est l'intervalle flou et E est l'environnement à partir duquel F a été obtenu. Q, à ce niveau, est caractérisée par les deux concepts : l'intervalle flou et l'environnement. Un environnement E, dans notre approche, est un ensemble

d'hypothèses floues (définies comme des ensembles flous), par exemple :
E = $\{(Ii)_{Di}, (Vj)_{Dj}, (Rk)_{Dk}, \ldots.\}$ où chaque hypothèse est caractérisée par un degré indiquant à quel point elle est applicable (juste).
Chaque environnement est caractérisé par un degré de cohérence global qui indique la cohérence entre l'intervalle flou F de la quantité Q et la valeur nominale du modèle. Ainsi, E devient :
E = $\{(Ii)_{Di}, (Vj)_{Dj}, (Rk)_{Dk}, \ldots.\}_{Dglobal}$

On doit alors mettre à jour les règles données ci-dessus pour prendre en compte les environnements.
En effet, il faut introduire les environnements dans les règles sémantiques car si pour la même quantité Q, la propagation a donné deux environnement différents ou deux valeurs différentes (c'est-à-dire deux chemins différents), l'information obtenue n'est pas la même. Ceci constitue une information plus riche et plus utile pour le diagnostic. Deux résultats (F1,E1) et (F2,E2) pour la même quantité Q sont identiques si et seulement s'ils ont les mêmes valeurs (F1=F2) et les mêmes environnements (E1=E2). La première règle devient :

(Q,Fm,Em), (Q,Fn,En) $\rightarrow (Q, (Fm \cap Fn)_{Dc(Fm,Fn)}, (En \cup Em))$

et l'implication logique, qui aide à éliminer beaucoup de candidats et ainsi à éviter l'explosion possible, peut être écrite par :

((Q,F1,E1) $\leq$ (Q,F2,E2)) $\Leftrightarrow$ (F1 $\subseteq$ F2) $\wedge$ (E1 $\subseteq$ E2)

Cet ordre entre les deux paires (intervalle, environnement) est une implication logique, c'est-à-dire (Q,F2,E2) est une conséquence logique de (Q,F1,E1). Donc, seules les paires minimales ont besoin d'être gardées et (Q,F1,E1) seulement sera propagé dans le circuit.
Il est important de rappeler qu'en cas d'intersection et à l'inverse des systèmes de diagnostic qui utilisent l'ATMS (comme DIANA [DJD+91] ou SOPHIE [BBdK82]) qui gardaient seulement l'intersection entre les valeurs propagées, notre ATMS-flou propage la valeur dont le Dc est le plus petit et garde dans sa mémoire les autres valeurs afin de les utiliser quand le diagnostic a besoin de plus d'informations (Figure 3.3).

Maintenant on arrive (ou on revient) au problème délicat de pouvoir décider si deux quantités ne sont pas identiques. Le problème se pose comme suit :
Pour une quantité Q, on a deux paires (F1,E1) et (F2,E2) données par la propagation comment tester si elles sont identiques ? En fait, comparer les environnements est simple puisqu'ils sont formés d'hypothèses. Mais, comparer deux intervalles flous reste le problème.
Pour répondre à cette question, nous avons adopté une technique basée sur la théorie des possibilités [DP87], qui consiste à calculer le degré pour lequel un intervalle flou **P** est plus grand qu'un intervalle flou **Q**.
Une telle technique, qui a été développée et utilisée dans FLAMES, consiste à calculer quatre degrés de possibilités afin de trouver le degré demandé (figure 3.8). En fait,

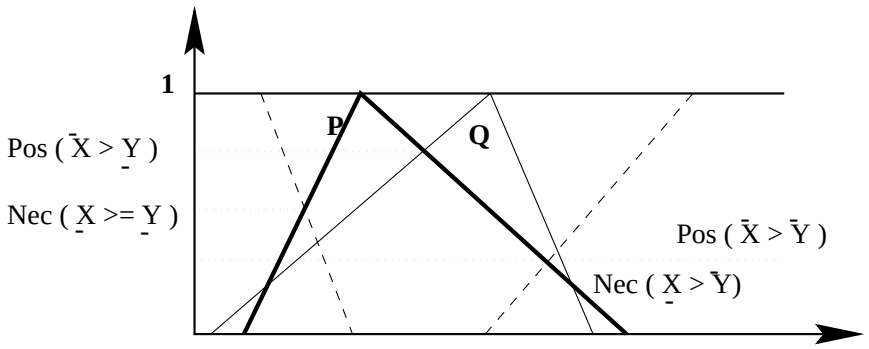


Figure 3.8: Comparaison entre deux intervalles flous

le calcul de ce degré revient à comparer deux intervalles flous. En pratique, les quatre indices, vus sur la figure 3.8 où Pos(X>Y) signifie la possibilité que X soit supérieur à Y et Nec(X>Y) la Nécessité que X soit supérieure à Y, suffisent pour discuter des positions relatives de deux intervalles flous [DP87].
L'opérateur > (I1, I2), où I1 et I2 sont des intervalles flous, et qui calcule à quel degré I1 est supérieur à I2, est défini comme la moyenne des quatre degrés précédents. Ainsi, < sera son complément à 1. Ces notions sont illustrées dans la figure 3.9.

```
Si E1=E2 alors
 Si Dc(F1,Fn)=Dc(F2,Fn) alors  //où Fn est la valeur nominale
   égale <- Comparer(F1,F2)
   Si égale = VRAI alors Même quantité, éliminer une des deux
   Sinon des quantités différentes, propager les deux
 Sinon des quantités différentes, propager les deux
```

```
/* où la procédure Comparer(F1,F2) teste d'abord, si un des deux
intervalles F1 ou F2 est inclu dans l'autre, sinon elle applique
une méthode possibiliste. */
```

```
//Comparer deux intervalles flous : à quel degré F1 est > F2
   tester si F1 est inclu dans F2 ou F2 est inclu dans F1 :
   Si oui, éliminer le plus grand
   Sinon, appeler la me'thode suivante :
```

```
  Comparer(F1,F2)
   calculer PSE(F1,F2)     // la possibilité que F1 est >= F2
   calculer NCE(F1,F2)     // la nécessité que F1 est >= F2
   calculer PS(F1,F2)      // la possibilité que F1 est > F2
   calculer NC(F1,F2)      // la nécessité que F1 est > F2

//Le degré demandé sera alors :
   Comparer(F1,F2) = (PSE+NCE+PS+NC)/4;
//Ainsi, deux intervalles flous sont égaux si et seulement si
   Comparer(F1,F2) = Comparer(F2,F1)
```

Figure 3.9: L'égalité entre deux quantités

FLAMES offre un ensemble d'opérateurs intelligents, comme +,-,*,<, >,etc. qui sont des opérateurs très généraux, puisque chacun d'entre eux est capable d'être appliqué aux différents types d'opérandes. Par exemple, on peut utiliser <(x,y), où x et y sont des nombres flous, des intervalles flous ou bien des environnements. C'est le compilateur qui choisit la bonne opération, comme on va le voir dans le chapitre 6.

3.3 La base de données : les modèles

En général, n'importe quel circuit peut être considéré comme un ensemble de modules ou de composants liés d'une manière spécifique. Les relations entre les paramètres, comme la tension et le courant, aux nœuds d'interconnexions des composants sont utilisés pour représenter leur comportement.

L'approche à base de modèles profonds est très convenable pour le traitement de ce genre de problèmes.
Une description de la structure du circuit, des modèles de comportements corrects des composants, des hypothèses contrôlant la validité des modèles, et des mesures (qui sont les observations) constituent les éléments principaux de cette approche.
Les lois de Kirchhoff et celles d'Ohm sont toujours vérifiées et les contraintes qui gouvernent les comportements des composants sont utilisées. Une résistance est gouvernée par Ir = Vr/r et Vr = Ir*r, par exemple, mais la correction de la résistance est l'hypothèse qui permet d'appeler ces règles, ce qui est exprimé par :

$$\textbf{Correct(R)} \rightarrow \textbf{Ir = Vr/r et Vr = Ir*r.}$$

Des modèles qualitatifs sont aussi utilisés chaque fois qu'il est nécessaire. Un transistor npn, par exemple, en état "actif" a une tension de base entre 0.5 et 0.8 Volts où l'état actif est défini sous forme d'un ensemble flou, comme on le verra.
Certains composants peuvent avoir plusieurs modes de fonctionnement. Chaque mode admet plusieurs règles qui le décrivent. Par exemple, pour un transistor DC npn, on peut avoir comme modes de fonctionnement, "actif, saturé ou bloqué: Les hypothèses

de bon fonctionnement sont combinées et propagées dans le modèle du circuit d'une manière semblable à celle de la propagation des valeurs.
Une contrainte générale peut avoir comme antécédent une conjonction de contraintes représentant des préconditions utilisées pour définir un mode particulier de fonctionnement dont la contrainte conclusion est valable, comme le montre l'exemple suivant :

$$\textbf{Correct(transistor\_T), V\_T\_be} \geq \textbf{0.6, V\_T\_ce} \geq \textbf{0.3}$$
$$\rightarrow \textbf{I\_T\_c = Beta * I\_T\_b};$$

qui peut être traduit par :
Si le transistor est bon et si sa tension de base émetteur est ≥ 0.6 et sa tension de collecteur émetteur est ≥ 0.3 Alors le courant du collecteur peut être calculé à partir de celui de l'émetteur par Ic = Beta * Ib.

La conclusion peut aussi être une hypothèse si, par exemple, le mode de fonctionnement en question est explicitement représenté :

$$\textbf{Correct(transistor\_T), V\_T\_be} \leq \textbf{0.4}$$
$$\rightarrow \textbf{Off(transistor\_T)}$$

Voici quelques exemples de modèles utilisés dans la version actuelle de FLAMES :

- **Une résistance** admet un seul mode de fonctionnement. La tolérance d'une résistance est utilisée pour former l'intervalle des valeurs permises. De nouveaux intervalles, des courants et des tensions, sont calculés en appliquant les lois d'Ohm mentionnées ci-dessus et une hypothèse de la correction de la résistance est diffusée également dans l'ATMS-flou.
- **Un amplificateur opérationnel** est modélisé en DC par le circuit de la figure 3.10.

- **Un transistor npn** est modélisé en DC par des modèles très simples, et on distingue trois modes de fonctionnement :

  mode actif défini par : Vb=Ve+Vbe, Vc≥Vb, Ib≥0, Ic=Beta*Ib, Ie=Ic+Ib

  mode saturé défini par : Vb=Ve+Vbe, Vc=Ve+Vce_sat, Ib≥0, Ic≥0, Ie=Ic+Ib

  mode bloqué défini par : Vb<Ve+Vbe, Ib=0, Ic=0, Ie=0

Notons que des modèles plus sophistiqués peuvent être utilisés quand c'est nécessaire. Dans ce cas, une ou plusieurs hypothèses gouvernent la validité des modèles. Pour

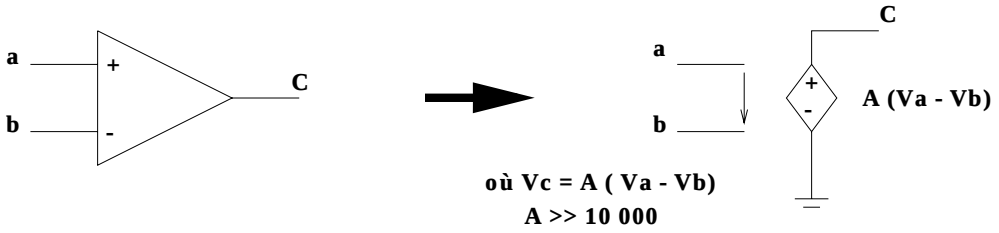


Figure 3.10: Modélisation de l'amplificateur opérationnel

nous, une hypothèse appartient à la logique floue, et non à la logique binaire ; cela veut dire qu'elle a un degré d'appartenance qui est son degré de validité. Prenons un exemple un peu détaillé pour clarifier ceci :
Si le transistor T est Correct et Vbe(T) ≤ 0.4 Alors T doit être dans l'état bloqué. Puisque l'opérateur $\leq$ est flou (comme tous les opérateurs arithmétiques et logiques dans FLAMES) et Correct(T) constitue un ensemble flou, alors Off(T) constituera un ensemble flou aussi. Définir les hypothèses, les quantités, et les modèles de fautes (comme on va le voir) comme des ensembles flous, donne à FLAMES sa souplesse ; ainsi, le changement le plus petit sera découvert. Par exemple, supposons qu'un transistor en mode normal soumis à : $Ib \ll Ic$ *et* $Ic \cong Ie$, alors définir les opérateurs $\ll, \cong$ comme dans la figure 2.1 peut illustrer cette idée. Si les résultats donnent $\mu_{\ll}$(Ib/Ic)=0.9, et $\mu_{\cong}$(Ic,Ie)=0.87, le transistor sera bon à un degré égal à min(0.87,0.9)=0.87.

3.4 Apprentissage par expérience et construction de la base de connaissances

En fait, le diagnostic à base de modèles profonds n'est pas aussi efficace que celui à base d'heuristiques (ou de règles), puisqu'il demande un calcul beaucoup plus compliqué [Kos89], mais il est plus sûr.

Néanmoins, les candidats obtenus restent plus nombreux avec la méthode à base de modèles. Cela est dû au fait que la méthode "système expert", par exemple, utilise des règles de justification faisant des hypothèses sur les pannes possibles, et présente le risque d'une erreur de diagnostic si la panne effectivement présente n'a pas été considérée lors de la réalisation du système expert.

Ainsi, pour améliorer son efficacité, on a décidé d'ajouter cette unité à FLAMES. Il faut bien savoir qu'en ce qui concerne les modèles de fautes, nous n'avons pas l'intention de définir un dictionnaire de fautes, mais pour améliorer l'efficacité du système, il est préférable d'introduire des modèles de faute dans FLAMES. Cela ne remet pas en cause le principe de diagnostic à base de modèles de bon fonctionnement ; en effet, comme l'ont montré [KW89] [Kos89] et [SD89], il est possible de coupler modèles de faute et modèles de bon fonctionnement. Les modèles de faute ne sont dans notre cas qu'un moyen supplémentaire mis à la disposition des utilisateurs (ou experts).

D'ailleurs, il faut être prudent car une utilisation exagérée de cette méthode peut rendre le système *très lourd*. L'exemple suivant [DTF91] de la figure 3.11 montre le travail supplémentaire à ajouter pour une partie très simple d'un circuit :

Si on ne possède pas de mesure sur B,C,D et que la mesure de A est 4, le programme de diagnostic détectera une contradiction au niveau du nœud **n** et fournira le "no-good" {A,B,C,D}, prouvant que l'un des quatre composants est en panne.

L'ajout de modèles de fautes les plus probables pour affiner ce résultat peut être fait en considérant, par exemple, la panne du circuit ouvert (donc i=0) et relancer le diagnostic avec les mêmes mesures.

Deux cas peuvent se produire :

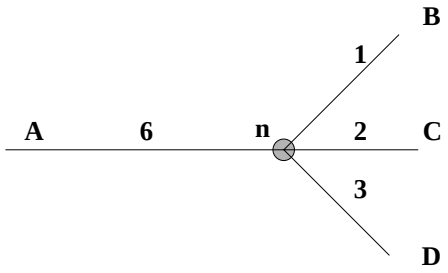


Figure 3.11: Modèles de faute pour un nœud de 4 branches

- si la contradiction apparaît encore, le composant dont on a introduit le modèle de faute n'en est pas responsable. Sous réserve que la faute effectivement présente fasse partie des fautes reconnues et que la faute soit unique, c'est l'un des problèmes des systèmes ne reposant que sur des modèles de fautes.
- si au contraire, la contradiction disparaît, le composant est susceptible d'être celui qui la provoque, on le conserve donc comme candidat.

Dans l'exemple présenté ici, on verra que seul C peut provoquer la contradiction détectée sur le nœud **n** car :

si A est en panne (circuit ouvert) : $0 \neq 1+2+3$

si B est en panne (circuit ouvert) : $4 \neq 0+2+3$

si C est en panne (circuit ouvert) : $4 = 1+0+3$

si D est en panne (circuit ouvert) : $4 \neq 1+2+0$

Quant à l'apprentissage de l'expérience, il a été introduit pour améliorer la performance du raisonnement à base de modèles d'une manière progressive. Cette unité doit être appelée seulement comme la dernière étape afin de raffiner les ensembles de candidats. Des autres systèmes, comme SOPHIE [BBdK82], utilisent les conflits pour éliminer des candidats. Supposons, par exemple, qu'un conflit indique que la résistance est fautive, il doit être *high* ou *open* tandis qu'un autre indique que le résistor doit être *low* ou *shorted*. La combinaison de ces deux résultats peut nous permettre d'éliminer ce résistor.

En fait, ce raisonnement est théoriquement faux dans le cas de fautes multiples ; pour cette raison, on ne l'utilise pas. Peut-être, l'utiliser comme heuristique pour aider le diagnostic serait mieux.

Les modes de fautes communes (comme *open, short, high,* ou *low* pour les résistances) dans notre approche sont définis comme des ensembles flous (figure 3.12). Ceci pourra nous éviter d'utiliser des heuristiques pour détecter les petites déviations.
Quand le système réussit à localiser une faute, une règle de type *symptôme-faute* qui

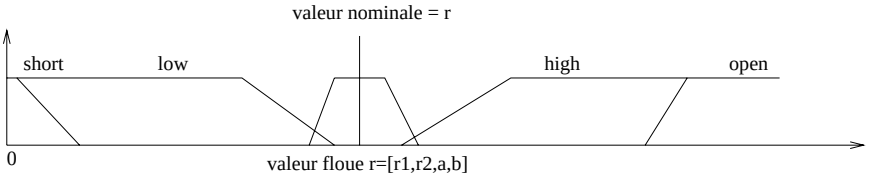


Figure 3.12: Les modes de fautes communes pour un résistor

résume le travail pourrait être construite et une estimation sera donnée au composant correspondant. Cette règle est donnée avec un degré de certitude compatible avec la logique floue d'un coté, et avec la nature complexe des circuits analogiques de l'autre coté. Cette information apprise de l'expérience est ajoutée à la base de connaissances pour des utilisations ultérieures.
Notons que l'expert peut ajouter à la base de connaissance des règles qui résument son expérience ou son expertise (ou l'expertise générale), soit localement à-propos des composants, soit globalement à-propos l'unité sous test. Les règles locales doivent définir les estimations de défectuosité de composants suivant des degrés de certitude, et les règles globales doivent décrire la causalité entre les composants suivant des degrés de certitude.

3.5 Stratégies de recherche du meilleur nœud à tester

Un système de diagnostic devrait être capable de recommander à n'importe quel point du processus, le meilleur test à effectuer sur l'unité sous test. A partir d'un ensemble valable de tests prédefinis et, en considérant les résultats de quelques tests, il doit pouvoir estimer la probabilité qu'un composant remplaçable soit fautif. Ceci résume l'idée de cette unité qui peut être expliquée par le fait qu'une fois une déviation détectée, le processus de diagnostic est lancé : l'ATMS-flou analyse les données

et crée les ensembles de candidats. Si une décision est difficile à prendre, (beaucoup de candidats, ambiguïté du diagnostic), le diagnostic aura besoin d'informations supplémentaires, ce qui actionne le processus de recherche du meilleur nœud à tester. Ce processus a été implémenté à base d'une approche floue décrite dans cette section. Mais, pourquoi une approche à base de la logique floue et non à base des probabilités ?

Les probabilités et la logique floue

Un grand débat avait (et a toujours) lieu sur les différences entre les probabilités et la logique floue et ce que peut apporter l'un mais pas l'autre. Les partisans des probabilités disent que tout ce que la logique floue peut faire, peut être fait par les probabilités. Les partisans de la logique floue se demandent si les probabilités existent ! Kosko [Kos91] a écrit : si on pouvait inverser la ligne de la vie, les probabilités auraient encore existé ; elles perdraient tout sens.

Mais, nous ne sommes pas extrémistes. Chacun des deux domaines a ses applications, les probabilités ont servi avec succès pendant longtemps et dans un nombre illimité de problèmes et elles continueront à y servir. La logique floue, qui est par rapport à l'autre une théorie nouvelle, est plus convenable pour exprimer l'imperfection humaine et pour résoudre les problèmes qui se créent à cause de ça.

En fait, nous croyons que les ensembles flous constituent une base mathématique puissante pour résoudre les problèmes de l'imprécision plus efficacement que les autres logiques. Ils sont capables de définir des approches moins qualitatives [SL92c] [SL92a] que les approches connues puisque les termes qualitatifs peuvent être définis via des ensembles flous soumis des opérations quantitatives. Ainsi, par exemple, QSIM a été redéfini [SL92b] et utilisé pour le diagnostic des systèmes dynamiques.

En tout cas, la logique floue a su remplacer les probabilités dans plusieurs domaines et de meilleurs résultats ont été obtenus.

En tenant compte des difficultés et des exigences de l'utilisation des probabilités (revoir section 1.4.1). Nous avons développé une méthodologie à base de la logique floue pour mieux exprimer et traiter l'imprécision de l'expert. On va montrer qu'elle est très efficace, plus facile et plus rapide que les méthodologies développées à base des probabilités.

Nous l'avons testée sur des circuits analogiques aussi bien que sur des circuits digitaux où elle a porté ses fruits. A cause de son importance et des résultats obtenus, on va consacrer le chapitre suivant tout entier à la bien présenter.

3.6 Unité de prise de décision floue

Cette unité de FLAMES, pour laquelle le chapitre 5 sera consacré, traite le problème connu par la prise de décision floue.
C'est quand l'environnement du travail n'est pas précis (il peut être imprécis et vague), la prise de décision floue devient une nécessité.

Dans le domaine de la conception électronique, par exemple, et avec l'explosion dans le nombre de possibilités de conception, les concepteurs rencontrent des problèmes de prise de décision pour choisir la meilleure méthode de DFT (Design-For-Testability). Les concepteurs ont besoin d'outils de CAO pour organiser, évaluer et choisir les techniques de test qui satisfont les objectifs de la conception. Mais, quand les objectifs ne sont pas définis d'une manière précise, le processus de prise de décision devient plus difficile à gérer. En effet, l'environnement devient flou et le processus lui même devient flou, i.e, il doit considérer des stratégies floues.

Cette unité a été développée et ajoutée à FLAMES pour aider le test, le diagnostic des CA, ainsi que pour proposer des méthodes efficaces pour le choix de DFT.
L'approche développée présente un processus de prise de décision flou, mais à plusieurs niveaux. Cela veut dire que la notion de flou est la plus générale, ce qui est utile et valable dans n'importe quel système où des tolérances des différents types sont utilisées.

3.7 Conclusion

Après avoir passé en revue les différentes approches utilisées pour le test et le diagnostic des circuits analogiques et mixtes, nous avons développé le système FLAMES

détaillé dans ce chapitre. FLAMES, construit principalement à base de logique floue, a déjà montré sa différence.

Les chapitres suivants vont continuer à détailler ses unités et présenter les résultats obtenus.

Chapitre 4

Recherche du meilleur nœud à tester

Chapitre 4

Stratégies de recherche du meilleur nœud à tester

L'approche qualitative floue, développée dans FLAMES, a été expérimentée, comme on l'a dit, sur des circuits digitaux [TMM96] et des circuits analogiques [MTM96].
Ce chapitre détaille cette approche et son application au cas des circuits digitaux, puis au cas des circuits analogiques.
Les résultats obtenus dans le cas des circuits digitaux sont présentés, ceux obtenus dans le cas ds circuits analogiques sont présentés et comparés à des résultats obtenus à l'aide d'une autre approche.

4.1 Approche développée en vue du diagnostic

Une solution efficace pour le test et le diagnostic des circuits digitaux doit répondre aux critères suivants :

– Mise en œuvre simple,

– Applicable à tout type de systèmes (séquentiels et combinatoires pour les systèmes digitaux).

– Applicable efficacement aux systèmes analogiques et être compatible avec la partie digitale (les chapitres suivants détailleront cette idée qui sera la base d'une approche pour les systèmes mixtes).

– Indépendante de tout modèle de fautes.

– Rapide à l'exécution et nécessitant peu d'espace mémoire.

Parmi les solutions possibles, nous avons opté pour une méthode basée sur une approche semi-qualitative. C'est une méthode qui opère en trois phases successives :

- Une première phase responsable de la génération des candidats. Elle repose sur les informations topologiques extraites de la description du système.
- Une deuxième phase basée sur un raisonnement qualitatif, et dont la tâche consiste à mettre à jour les estimations relatives au degré d'implication de chaque module dans le dysfonctionnement observé.
- Une dernière phase responsable de la localisation du bloc défectueux. Elle utilise un raisonnement semi−qualitatif associant des notions de logique floue.

4.1.1 Génération des candidats

Nous nous plaçons dans le cadre des hypothèses de travail suivantes :

- Le système à diagnostiquer est décrit au niveau des blocs. Ces derniers sont considérés comme des boites noires avec des entrées/sorties. Aucune indication sur la fonctionnalité du bloc n'est requise.
- Nous disposons d'un ensemble de vecteurs de test à appliquer sur les entrées primaires du système, et les réponses correctes relatives à chaque vecteur.

Le processus de diagnostic est invoqué lorsqu'un mauvais fonctionnement du système est constaté. Lors du diagnostic, on peut être confronté à deux types de systèmes :

soit des systèmes combinatoires ou des systèmes séquentiels. Dans la méthode que nous proposons, ces deux types de systèmes sont traités globalement de la même façon. Néanmoins, les systèmes séquentiels requièrent un traitement supplémentaire spécifique qu'on indiquera au moment opportun.

4.1.1.1 *Cas des systèmes combinatoires*

La procédure de génération des candidats utilise un algorithme basé sur la notion de causalité. Cet algorithme se déroule en deux phases distinctes :

- Une première phase dite de propagation arrière ("backtracing phase") durant laquelle l'algorithme cherche à mettre à jour les blocs qui ont pu contribuer à engendrer une sortie erronée. Pour cela, et pour chaque sortie affectée, l'algorithme parcourt le circuit en répertoriant tous les blocs convergeant vers cette sortie. Ces blocs appartiennent tous à ce que l'on désignera par "cône de fanin" ou "coverage cone" [Mac84]. Ces blocs seront déclarés comme "Présumé Fautif", et stockés dans une liste.

- Une deuxième phase dite de propagation avant ("forward propagation") dans laquelle on fait la mise à jour des estimations obtenues lors de la première phase, et en même temps on essaye d'identifier les blocs complètement observables (c'est à dire dont les entrées/sorties sont primaires). Ces blocs seront définitivement déclarés comme "Correct" ou "Fautif" en fonction de leur état, et retirés de la liste des blocs "Présumé Fautif". Les blocs non affectés et débouchant sur des sorties correctes seront déclarés dans un premier temps comme "Présumé Correct". La mise à jour des estimations se fait de la manière présentée dans le paragraphe 4.1.2.

A ce stade de l'algorithme, un choix sur l'hypothèse de faute doit être fait. Si on choisit de se positionner dans l'hypothèse de la faute unique, alors l'étape suivante consiste à chercher l'intersection des ensembles générés (si plusieurs sorties sont erronées). Le

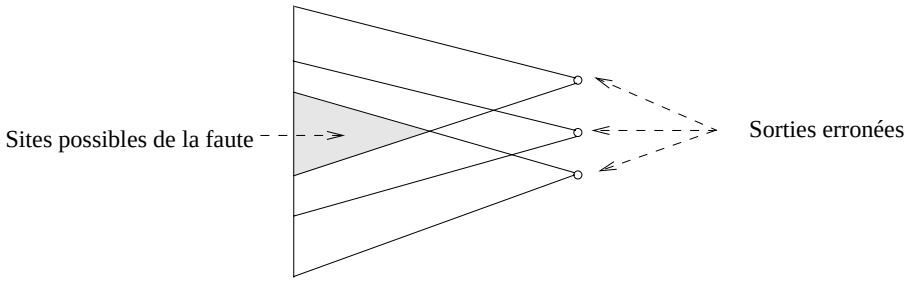


Figure 4.1: Intersection en cas de faute unique

bloc fautif appartenant nécessairement à cet ensemble résultant de candidats (Figure 4.1). Si la séquence de test appliquée assure une couverture de faute de 100%, alors les blocs restants seront considérés comme "Vraisemblablement Correct" . Sinon, ils seront déclarés comme "Présumé Correct".

Par contre, dans le cas où on opte pour l'hypothèse de faute multiple, aucun traitement supplémentaire n'est effectué avant le commencement de la phase suivante.

4.1.1.2 *Cas des systèmes séquentiels*

Le même algorithme est étendu aux systèmes séquentiels. L'extension est basée sur le fait que l'on doit seulement tenir compte des contraintes supplémentaires relatives aux parties séquentielles du système. A l'état actuel de notre étude, la méthode ne prend en compte que les systèmes séquentiels synchrones. Ces systèmes sont composés de modules combinatoires et de bascules (figure 4.2). Pour ce type de systèmes, le temps est un paramètre très important. En effet, pour propager une valeur à partir d'un nœud donné vers une sortie primaire, on est amené à traverser un certain nombre de bascules. La traversée de chaque bascule nécessite un coup d'horloge. Il est donc clair que l'instant de mesure, en termes de coups d'horloge exécutés depuis le début de la séquence de test, sera une information déterminante pour le processus de diagnostic. En pratique, cette information va servir à éliminer de la liste des candidats, tous ceux dont l'état ne peut être observé sur une sortie primaire au moment de la mesure. Pour cela, nous allons utiliser la notion de profondeur de séquentialité, définie comme le nombre maximal de bascules à traverser pour aller d'un élément donné à une sortie

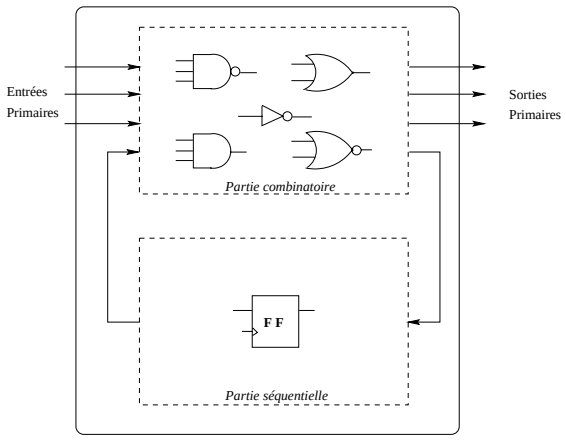


Figure 4.2: Modélisation d'un système séquentiel

primaire. Néanmoins, dans notre cas la profondeur de séquentialité sera calculée par rapport aux sorties erronées.

4.1.2 Mise à jour des estimations

A la fin de l'étape de génération des candidats, on se retrouve donc avec pour chaque vecteur de test, une liste de candidats avec les estimations qui leur sont associées. La question qu'on se pose naturellement est comment combiner ces diverses sessions de diagnostic (et par conséquent listes de candidats) afin d'obtenir une session unique ?

Généralement, les estimations sont obtenues après un calcul probabiliste. Plusieurs approches sont possibles pour réactualiser leurs valeurs lorsque de nouvelles informations sont disponibles. Cependant, ces approches nécessitent souvent des calculs très lourds et par conséquent très longs. C'est pour cette raison que nous avons choisi d'utiliser plutôt une méthode qualitative pour réaliser cette opération.

Dans le contexte du raisonnement qualitatif, nous avons défini des estimations qualitatives pour caractériser le degré de défectuosité de chaque élément. Ces estimations couvrent l'intervalle compris entre les états "Correct" et "Fautif". Comme le processus est qualitatif, nous avons découpé cet intervalle selon l'ensemble des estimations suivant [TMM96]:

- **C**orrect (C)
- **V**raisemblablement_**C**orrect (V_C)
- **P**lutôt_**C**orrect (P_C)
- **S**upposé_**C**orrect (S_C)
- **A**mbigu (Amb)
- **S**upposé_**F**autif (S_F)
- **P**lutôt_**F**autif (P_F)
- **V**raisemblablement_**F**autif (V_F)
- **F**autif (F)

Il est clair que cette décomposition purement arbitraire, peut être modifiée en fonction des exigences de l'application. Elle dépendra surtout du degré de granularité désiré pour le diagnostic. Pour faciliter la procédure de modification de cette décomposition, nous avons opté pour une représentation matricielle (figure 4.3). Avec ce type de représentation, changer la décomposition revient à remplacer la matrice par une autre. Le fusionnement des résultats des différentes sessions de diagnostic se déroule donc selon les règles qualitatives définies dans la matrice de la figure 4.3. Cette matrice doit être interprétée de la manière suivante : Si un premier vecteur de test donne à un composant X une estimation i, et un deuxième lui donne une estimation j, alors la nouvelle estimation pour ce composant sera $\mathrm{M}(i,j)$, avec M la matrice et i, j respectivement la ligne et la colonne (ou l'inverse étant donné que la matrice est symétrique). Par exemple, si X est estimé S_C (i=*3*) par un vecteur et S_F (j=*7*) par un autre, le résultat est $\mathrm{M}(S\_C,S\_F) = Amb$. On remarquera que cette matrice comprend deux lignes et colonnes particulières ; il s'agit des lignes et colonnes "Correct" et "Fautif". Elles expriment le fait que "Correct" et "Fautif" sont plus proches de l'affirmation que de l'estimation. Ceci découle du fait qu'elles sont obtenues à partir de mesures (observation directes des entrées/sorties primaires du bloc concerné) et

que, hormis le cas où elles auraient été effectuées lors d'un "test escape case" [Ben94], ces mesures sont considérées comme fiables. Il est à noter que dans ce processus,

	Correct	**V_C**	**S_C**	**P_C**	**Amb**	**P_F**	**S_F**	**V_F**	**Fautif**
Correct	Correct	Correct	Correct	Correct	Correct	Correct	Correct	Correct	
V_C	Correct	Correct	V_C	S_C	S_C	S_C	P_C	Amb	Fautif
S_C	Correct	V_C	V_C	S_C	P_C	P_C	Amb	P_F	Fautif
P_C	Correct	S_C	S_C	S_C	P_C	Amb	P_F	S_F	Fautif
Amb	Correct	S_C	P_C	P_C	Amb	P_F	P_F	S_F	Fautif
P_F	Correct	S_C	P_C	Amb	P_F	S_F	S_F	S_F	Fautif
S_F	Correct	P_C	Amb	P_F	P_F	S_F	V_F	V_F	Fautif
V_F	Correct	Amb	P_F	S_F	S_F	S_F	V_F	Fautif	Fautif
Fautif		Fautif	Fautif	Fautif	Fautif	Fautif	Fautif	Fautif	Fautif

Figure 4.3: Règles de combinaison des estimations

aucune hypothèse n'a été avancée sur la capacité de diagnostic ("diagnosis power") de chaque vecteur. Cette notion exprime le fait qu'un vecteur permet de diagnostiquer plus ou moins facilement une faute. En fait, cette notion prend toute son importance dans l'hypothèse de la faute unique. Un vecteur de test qui détecte un nombre réduit de fautes (donc pas très efficace pour le taux de couverture) sera efficace lors du diagnostic. Or cette information ne peut être obtenue que par une simulation de fautes, ce qui présume de la disponibilité simultanée de la description au niveau portes ("netlist") du système et de l'outil de simulation. Or ces deux contraintes ne font pas partie de nos hypothèses de travail. De ce fait, le même poids (en terme de capacité de diagnostic) est appliqué à tous les vecteurs. Cependant, si ces informations sont disponibles, leur intégration au processus de combinaison sera toujours possible.

4.1.3 Exemple d'application

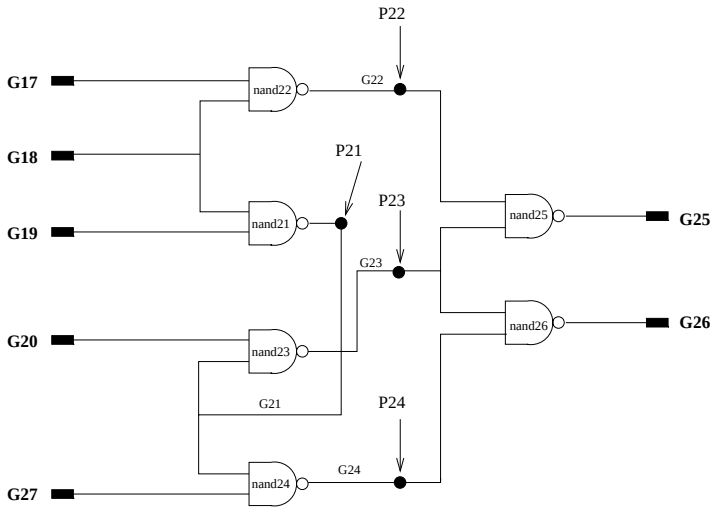


Figure 4.4: ISCAS85 c17 (6 blocs dans le module)

Le circuit de la figure 4.4 représente le c17, le plus petit des benchmarks de la série ISCAS'85 [BF85]. Dans ce circuit, nous avons injecté un collage logique à 1 (s-a-1) sur le nœud G23. On a généré à l'aide de HITEST [Aut94] deux vecteurs de test capables de détecter cette faute. Les conditions de test sont résumées dans le tableau 4.1. Le

Session	G25	G26
Session 1	erreur	erreur
Session 2	erreur	ok

Table 4.1: Conditions des différentes sessions pour le c17

tableau 4.2 montre les résultats de la phase de génération des candidats et la mise à jour des estimations dans l'hypothèse de la faute unique. Le tableau 4.3 donne les résultats obtenus dans l'hypothèse de fautes multiples. On remarquera une perte de précision par rapport au résultat précédent. Ceci s'explique par le fait que l'hypothèse de faute simple est plus restrictive et ne correspond pas souvent à la réalité.

Candidat	nand21	nand22	nand23	nand 24	nand25	nand26
Session1	S_F	V_C	S_F	V_C	V_C	V_C
Session2	S_F	S_F	S_F	V_C	S_F	V_C
Résultat	S_F	P_C	S_F	C	P_C	C

Table 4.2: Génération des candidats : Hypothèse de faute simple

Candidat	nand21	nand22	nand23	nand 24	nand25	nand26
Session1	S_F	S_F	S_F	S_F	S_F	S_F
Session2	S_F	S_F	S_F	V_C	S_F	V_C
Résultat	S_F	S_F	S_F	S_C	S_F	S_ C

Table 4.3: Génération des candidats : Hypothèse de faute multiple

4.2 L'approche qualitative floue

Pour échapper aux restrictions de l'approche probabiliste, nous avons vu qu'une méthode basée sur un raisonnement qualitatif pour la génération et la classification des candidats peut être très utile pour ce but. Cette classification utilise des estimations qualitatives basées sur des quantifications linguistiques, permettant ainsi d'éviter de manipuler des calculs lourds et fastidieux. La fusion des différentes sessions de test est effectuée à l'aide d'une matrice. Ceci permet une simplification du processus et une grande malléabilité quant à la définition de la granité du diagnostic. En effet, pour la changer, il suffit de remplacer cette matrice par une nouvelle matrice.

La technique proposée repose uniquement sur l'observation des fautes sur les sorties primaires des systèmes à diagnostiquer. Elle n'est pas liée à un modèle de faute particulier et ne suppose aucune information sur la fonctionnalité des blocs internes composant le système. Néanmoins, si on se limite à la phase de génération des candidats, ceci peut s'avérer suffisant pour des petits systèmes. Mais ce n'est généralement pas suffisant pour effectuer un bon diagnostic lorsqu'il s'agit de systèmes un peu plus

complexes. La qualité du diagnostic obtenu à ce moment va dépendre de la qualité des vecteurs de test et le profil du système à tester. Afin de remédier à ces insuffisances, nous proposons dans le prochain chapitre une nouvelle stratégie complémentaire susceptible d'améliorer la précision du diagnostic réalisé à l'aide de cette approche.

Quelle que soit la méthode utilisée, les mesures effectuées sur les nœuds primaires du système permettent rarement de remonter directement au site de la faute. Les techniques de diagnostic reposent généralement sur l'établissement d'un chemin de causalité pour la propagation des fautes à l'intérieur du système sous test. Par conséquent, des mesures supplémentaires au niveau des nœuds internes du système sont nécessaires pour raffiner le diagnostic, en diminuant les divergences à l'intérieur du chemin de causalité.

Il est donc nécessaire de résoudre le problème du choix des nœuds à tester. Ce choix doit être effectué de manière pragmatique, car une procédure de choix aléatoire peut se révéler coûteuse et totalement inefficace pour l'amélioration du diagnostic. Pour cette raison, nous proposons dans ce chapitre une méthode basée sur la logique floue qui permet d'indiquer les meilleurs nœuds à sonder pour une convergence rapide vers le (ou les) site(s) de la (ou les) faute(s).

4.3 Raisonnement qualitatif à base de logique floue

Notre choix de l'utilisation de la logique floue est motivé par le souci d'éviter toute approche purement probabiliste. Ces approches ont déjà été utilisées dans d'autres systèmes de diagnostic. Elles ont dû être délaissées parce qu'elles s'accompagnent souvent de lourds calculs, d'hypothèses fastidieuses sur les probabilités *a priori* concernant les différents modules du système sous test, ainsi que sur les dépendances statistiques qui les unissent (section 1.4.1).

Une approche à base de logique floue peut aborder le problème du diagnostic à la manière d'un expert, par la réduction des calculs nécessaires, en remplaçant les quantifications numériques par des quantifications linguistiques.

4.3.1 Probabilités linguistiques floues

Un expert n'aime pas donner des nombres précis, des probabilités *a priori* comme estimation de défaillance des modules d'une part, et d'autre part ces nombres seront imprécis. Il est plutôt plus raisonnable de lui demander de donner des estimations en termes linguistiques décrivant les défaillances de ces modules.
Quelques études ont montré que traiter de telles probabilités offre un autre avantage en comparaison avec les probabilités numériques : les résultats sont bien meilleurs quand on demande aux experts de donner des probabilités linguistiques [BD86].
L'idée est de décomposer l'intervalle [0,1] entre plusieurs intervalles flous définissant des termes linguistiques. Le degré de granularité de cette décomposition dépend de l'application et de ce que l'expert assume le plus convenable à son application. Dans [BD86], on peut trouver une étude détaillée de cette idée de laquelle nous avons adapté notre espace de quantité qui constitue l'ensemble des termes linguistiques considérés (QS). Notre Qs sera le suivant (figure 4.5) :
{ Correct [0,0,0,0],
Vraisemblablement_Correct[0.01,0.2,0.01,0.05],
Plutôt_correct[0.1,0.18,0.06,0.05],
Supposé_Correct [0.22,0.36,0.05,0.06],
Ambigu[0.40,0.59,0.09,0.07],
Supposé_Fautif[0.63,0.80,0.05,0.06],
Plutôt_Fautif[0.78,0.92,0.06,0.05],
Vraisemblablement_Fautif[0.98,0.99,0.05,0.01],
Fautif[1,1,0,0] }.
A noter que cet ensemble n'est pas symétrique (sur la même figure : Ent(Plutôt_Fautif) < Ent(Plutôt_Correct), etc.). Les entropies sont des entropies floues calculées suivant la formule donnée en section 4.3.3.

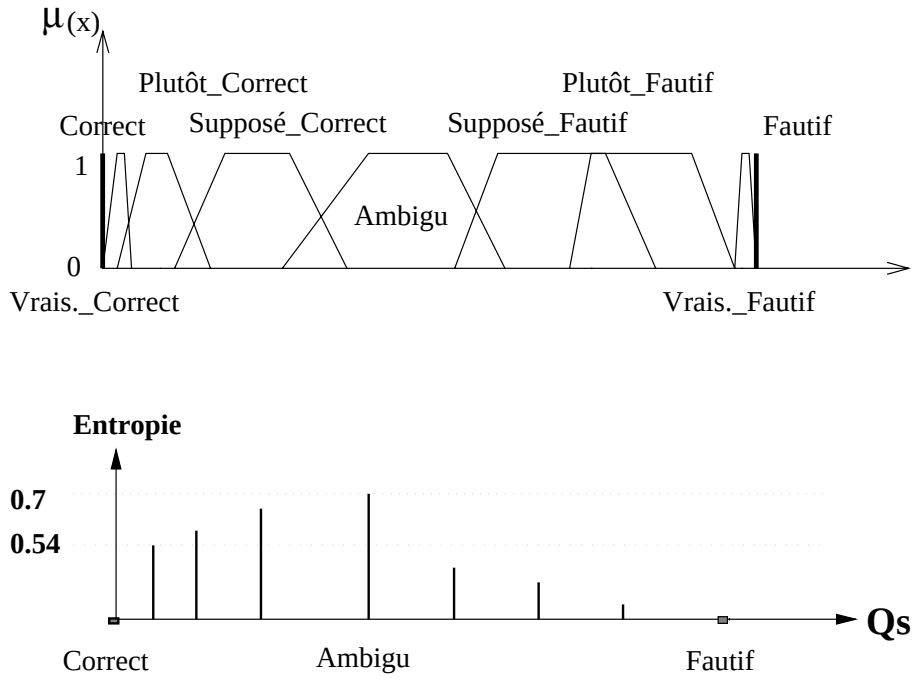


Figure 4.5: Probabilités linguistiques et leurs entropies correspondantes

4.3.2 Meilleur point à tester : une approche qualitative floue

Jusqu'à présent, la méthode que nous proposons permet de synthétiser les différentes sessions de diagnostic en une session unique. Par conséquent, partant de plusieurs listes de candidats avec leurs estimations, nous obtenons à la fin du processus une liste unique. A ce stade de la méthode, et mis à part le cas des petits systèmes ou bien le cas où l'ingénieur de maintenance juge suffisantes les informations mises à sa disposition, une importante ambiguïté persiste concernant l'origine de la ou des fautes qui induisent le dysfonctionnement observé du système sous test.

Afin de remédier à ce problème et d'améliorer l'efficacité de la méthode développée, nous proposons une solution consistant en une étape supplémentaire responsable de la détermination de l'ensemble des nœuds à tester du système pour localiser le ou les sites potentiels de fautes. Le choix de ces nœuds doit se faire selon des critères bien déterminés de façon à diminuer le nombre de nœuds et par suite le coût de cette opération en termes de temps et d'équipements nécessaires.

Au cours de cette étape supplémentaire, une fonction de coût est calculée pour chaque nœud afin d'estimer l'importance de l'information qui serait recueillie sur ce dernier. Cette fonction utilise à cet effet trois paramètres :

- Le cône de Fanout : C_{fout} (figure 4.6). On désigne par ce terme l'ensemble des composants qui sont susceptibles d'être influencés par le nœud considéré.
- Le cône de Fanin : C_{fin} (figure 4.6). C'est l'ensemble des composants qui peuvent exercer une influence sur le nœud considéré.
- L'entropie floue : il s'agit de la mesure de l'entropie des probabilités des candidats.

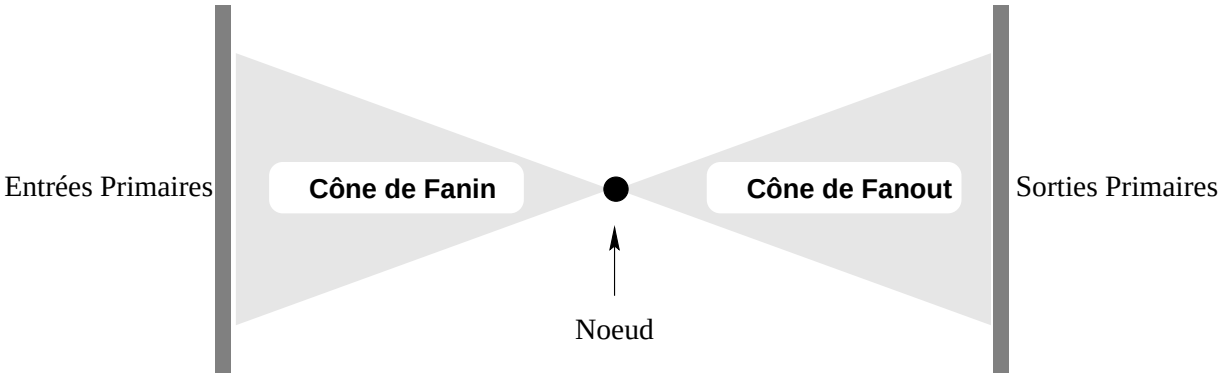


Figure 4.6: Cône de fanin et Cône de fanout

Le choix de ces paramètres est motivé par les raisons suivantes :

– En ce qui concerne l'entropie, étant donnée une étape particulière dans le processus de diagnostic, on est amené à évaluer l'apport de chaque mesure effectuée sur le système pour déterminer la mesure à effectuer au cours de l'étape suivante. Dans ce cas, avec les nombreuses propriétés importantes [BD86] qu'elle possède, la mesure de l'entropie des probabilités constitue une bonne indication comme elle l'a déjà démontré dans d'autres applications. Pour les composants estimés "*Vraisemblablement_Correct*" ou "*Vraisemblablement_Fautif*", l'entropie sera minimale ; de tels candidats ne coûtent pas chers à identifier. Par contre, pour les éléments dont l'estimation est "*Ambigu*", l'entropie sera maximale et par conséquent la quantité d'information apportée par la

mesure de ce nœud n'aura pas beaucoup d'effet sur le degré d'ambiguïté rémanant ; ces éléments sont chers à identifier. Comme les données manipulées sont floues, et dans le but d'éviter l'influence des cardinaux des ensembles considérés, on calcule *l'entropie moyenne floue* comme expliqué dans le paragraphe 4.3.4. L'ensemble présentant l'entropie minimale sera considéré comme le meilleur. Il présentera une dégénérescence réduite et par conséquent une ambiguïté moindre.

– Le choix du *cône de fanin* comme paramètre structurel s'explique par le fait que, si la mesure d'un nœud appartenant au chemin de propagation de la faute montre qu'il est correct, alors on peut en déduire que la faute se trouve quelque part dans l'ensemble de ses successeurs (cône de fanout), ce qui permet de réduire l'ambiguïté sur son cône de fanin.

– Le choix du *cône de fanout* s'explique de manière analogue à celle du cône de fanin. Si une mesure effectuée sur un nœud donne une valeur erronée, ceci indique que la faute se situe quelque part dans l'ensemble de ses prédécesseurs (cône de fanin), et par suite l'ambiguïté est atténuée sur le cône de fanout.

En plus, le fait de considérer simultanément ces deux cônes (fanin et fanout) permet d'équilibrer leurs influences respectives. En effet, considérer uniquement le cône de fanout revient à favoriser les composants voisins des entrées primaires. A l'inverse, considérer uniquement le cône de fanin revient à favoriser les éléments voisins des sorties primaires. Or les nœuds situés au milieu des chemins sont souvent les plus intéressants à sonder (quelles que soient les conditions, on éclaire 50% des éléments). Pour cette raison, nous définissons le paramètre *Degré d'influence* ($D_{in}f$), une combinaison linéaire de C_{fin} et de C_{fout} de la manière suivante :

$$\boxed{D_{inf} = \alpha * C_{fin} + \beta * C_{fout}}$$

α, β sont les coefficients de pondération. Ils permettent de donner un poids plus ou moins important aux éléments qui leurs sont associés. Le choix de ces coefficients sera étudié dans le paragraphe 4.3.4

4.3.3 L'entropie floue

L'aléatoire exprime l'incertitude de l'occurrence d'un événement décrit précisément. Le flou traite des cas où l'objet lui-même est intrinsèquement imprécis. Cependant, il peut arriver que le flou et l'aléatoire soient ensemble présents. Un domaine de probabilités floues, un événement flou et une entropie d'un sous ensemble flou peuvent être définis [NR75].
Kosko [Kos91], traite le même problème mais d'un point de vue géométrique et il donne une discussion détaillée de l'entropie floue ; son entropie floue mesure à quel degré le système flou est flou !

Ici, on a besoin d'étudier l'entropie d'un système de probabilités floues qui mesure à quel degré ce système flou est aléatoire. Le module sous test est considéré comme un système de composants pour lesquels on a estimé leurs états en termes de probabilités floues. Alors, on a adapté la définition de l'entropie de Shannon pour calculer cette entropie floue.

Pour notre application, le module sous test est considéré comme un ensemble de composants dont chacun est muni d'une estimation de son état en terme de probabilité floue. Pour calculer cette entropie, nous avons adopté la définition intrinsèque de *Shannon* qu'on a adaptée au calcul flou.

Soit S un ensemble de n composants caractérisés par leurs estimations floues. L'entropie floue de cet ensemble se calcule de la manière suivante :

$$\boxed{Ent(S) = \oplus_{i=1}^{n} F_i \otimes Log_2(1 \oslash F_i)}$$

avec F_i l'estimation de défectuosité du composant i, et $\oplus$, $\otimes$ et $\oslash$ représentant respectivement l'addition, la multiplication et la division floues (paragraphe 1.4.9). Les fonctions Logarithme et calcul de l'inverse sont les suivantes [BD86] :

Pour $m = [m1, m2, \alpha, \beta]$

* $log_2 m = [log_2(m1), log_2(m2), log_2(m1/(m1-\alpha)), log_2(m2+\beta)/m2]$ avec $\alpha > 0$
* $1/m = [1/m2, 1/m1, \beta/(m2*(m2+\beta)), \alpha/(m1*(m1-\alpha))]$ avec $m1 > 0$ or $m2 < 0$

4.3.4 Choix du point à tester

Afin de déterminer le meilleur point à tester, nous devons évaluer l'importance topologique de chaque nœud dans le processus de diagnostic. Pour cela, nous disposons de deux informations concernant chaque nœud : l'entropie et le degré d'influence.

Le problème du choix du meilleur point à tester peut être formulé de la manière suivante : étant donné n points (chaque point représente un ensemble de candidats de cardinal k), on calcule son entropie floue moyenne et son degré d'influence. Donc, pour chaque point, on obtient deux vecteurs d'information suivants :

$$\boxed{Ent_{moy}(i) = \frac{1}{k} * Ent_{i\ (i=1,k)} \ \text{ et } \ Df_{j\ (j=1,k)}}$$

Supposons maintenant que $E = min\ \{Ent_{moy}(i)_{(i=1,n)}\}$ et $D = max\ \{Df(j)_{(j=1,n)}\}$. Il est alors clair que le point $P_{opt}\ (E, D)$ représente le meilleur point à tester car il possède en même temps, un fort degré d'influence et une faible entropie.

Mais en pratique, il est rare de tomber sur le point idéal vérifiant simultanément ces deux conditions. Dans ce cas, le meilleur point serait celui qui s'en rapproche le plus. Reste à trouver un moyen pour chercher ce point.

4.3.4.1 *La distance de Hamming*

Une solution possible pour la recherche du point le plus proche du point optimal est *la distance de Hamming.*

Soit $P_{opt}\,(E, D)$ et les deux vecteurs $Ent_{i\ (i=1,n)}$ et $Df_{j\ (j=1,n)}$. Pour ces deux vecteurs, on calcule la distance suivante :

$$D_{ham}\,(Pk, P) \;=\; \sqrt{(Ent(k) - E)^2 + (1 - \tfrac{Df(k)}{D})^2} \quad \text{pour chaque } k \in [1, n].$$

Le point donnant la distance minimale correspondra au point recherché. Si pour le même i, on a $Ent(i) = E$ et $Df(i) = D$, alors la distance obtenue sera nulle, et il s'agira à ce moment du point optimal. On notera au passage que les termes de cette fonction sont tous normalisés ($Ent_{moy} < 1$ par définition et $\frac{Df_i}{D} \leq 1$). Cette normalisation permet d'obtenir un résultat borné ce qui facilite la détection des problèmes de calculs.

4.3.4.2 *La fonction de coût*

Le calcul d'une fonction de coût à minimiser est aussi une solution possible pour la recherche du meilleur nœud à tester. Nous savons déjà, grâce à la théorie de l'information et celle des décisions que l'entropie est un bon paramètre d'estimation. Si à ce dernier, on ajoute des informations d'ordre structurel (par conséquent spécifiques au système sous test), on devrait arriver à construire une fonction de coût efficace. Comme on désire minimiser l'entropie et maximiser le degré d'influence, nous avons choisi une fonction qui a la forme suivante :

$$Cost\,(Ent, Df) = \alpha * Ent \;+\; \beta * \tfrac{1}{Df}$$

Avec α et β des coefficients de pondération.
Ces coefficients de pondération peuvent avoir des importances diverses. Différents poids peuvent leur être affectés en fonction de l'importance à donner à chaque paramètre de la fonction. A ce niveau, la logique floue est particulièrement adaptée, car on n'est pas tenu de donner des valeurs numériques précises à chaque coefficient. Il suffit simplement de définir des classes de poids qu'on adapte à la granularité désirée.

4.3.4.3 *Choix des coefficients*

Qu'il s'agisse du degré d'influence (Df) ou de la fonction de coût ($Cost$ ( )), le choix des coefficients de pondération aura un grand impact sur le résultat du calcul final.

Pour notre application, nous avons choisi de décomposer l'intervalle [0,1] en trois poids (intervalles) floues : Faible, Moyen, et Fort (figure 4.7). Ce choix est totalement arbitraire, et peut être rapidement modifié si cela s'avère nécessaire.

Pour ce qui concerne le degré d'influence (Df), nous avons fait le choix de donner le poids "Moyen" pour le cône de fanout et le poids "Faible" pour le cône de fanin.

$$Df = Moyen \otimes C_{fin} + Faible \otimes C_{fout}$$

Le choix de donner un poids plus important au cône de fanout est motivé par le souci d'accorder plus d'importance à l'observabilité (le cône de fanout débouche sur les sorties primaires, par conséquent observables) qu'à la contrôlabilité (le cône de fanin prend naissance aux entrées primaires, par conséquent totalement contrôlables).
Le chapitre suivant montrera une méthode uniforme (plus puissante) pour le choix de poids de paramètres. L'expert ne sera pas limité à un petit nombre de poids, des modificateurs seront offerts pour lui permettre de mieux choisir et d'une manière plus précise les poids convenables. En ce qui concerne la fonction de coût, il existe deux

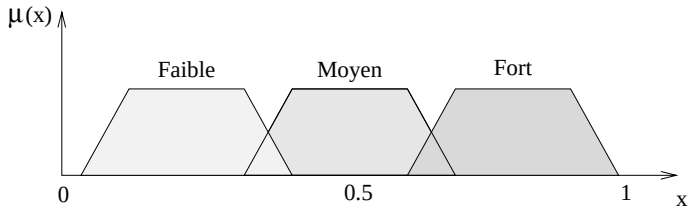


Figure 4.7: Poids flous des paramètres

façons possibles pour choisir les poids à affecter aux coefficients des paramètres de cette fonction :

– une manière "statique" qui consiste à faire un choix arbitraire comme dans le cas du degré d'influence. Mais si dans ce dernier cas, le choix était guidé par des critères de contrôlabilité et d'observabilité, ce n'est plus le cas ici.
– une manière "dynamique" dans laquelle les poids sont automatiquement déterminés en fonction de la quantité d'information apportés par chaque paramètre. On calcule pour cela l'écart absolu moyen (ε) des entropies moyenne des points de test. Cet écart est donné par la relation suivante :

$$\varepsilon = \frac{1}{n} * \sum_{i=1}^{n} \mid E_i - \bar{E} \mid$$

où n désigne le nombre de points, E_i l'entropie moyenne du point i, et $\bar{E}$ la valeur moyenne des entropies moyennes du système.

Si cet écart est jugé significatif, ceci implique que l'entropie apporte suffisamment d'informations pour la discrimination des points, et elle aura par conséquent un poids "Fort" tandis que le degré d'influence aura le poids "Moyen". Dans le cas inverse, l'entropie aura le poids "Moyen" et le degré d'influence aura le poids "Fort".

4.3.5 Entropie floue prévue

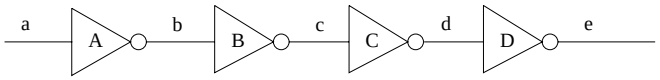
A ce stade de travail, il reste possible d'avoir un ensemble de points avec les mêmes valeurs. Dans un cas pareil, une autre procédure est déclenchée pour régler ce problème. Elle consiste à injecter les résultats prévus pour chaque nœud de cet ensemble et essaie de trouver celui qui peut donner la meilleure information supplémentaire. Pour ce but, l'entropie prévue peut être utile.

Adaptée de [KW87], cette entropie, notée $Ent_{ex}(X_i)$, calcule l'entropie d'un ensemble donné après avoir mesuré la quantité X_i ; elle est donnée par :

$$Ent_{ex}(X_i) = \bigoplus_{k=1}^{n} F_i(X_i = V_{ik}) \otimes Ent(X_i = V_{ik})$$

avec $V_{i1}, \ldots, V_{in}$ sont des valeurs possibles de X_i, F_i et Ent comme définie au-dessus.

DeKleer [KW87], présente une approche probabiliste qu'il propose d'utiliser pour calculer cette entropie et trouve le meilleur point à tester. Ici, nous allons comparer notre approche (qui utilise des estimations floues) avec la sienne en considérant un exemple (figure 4.8) détaillé dans son papier. Cette comparaison montre que notre approche, qui n'a besoin d'aucune information *a priori* contrairement à celle de deKleer, est aussi efficace, mais plus adaptée à notre travail, et avec un moindre coût.



a : Coûts prévus après a=1

	Approche de deKleer	Approche floue
b	0.98	0.43
c	0.96	0.34
d	0.94	0.24
e	0.93	0.14

b : Coûts prévus après a=1 et e=0

	Approche de deKleer	Approche floue
b	0.45	0.259
c	0.31	0.185
d	0.44	0.259

Figure 4.8: Inverseurs en cascade

4.3.5.1 *Inverseurs en cascade : a=1*

Dans ce premier cas (figure 4.8.a), supposons que la probabilité initiale de la défectuosité du composant soit 0.01 et que l'entrée a=1 est donnée. Dans l'approche floue, Cette probabilité peut être exprimée par l'estimation "Vraisemblablement_Correct". Le tableau de résultats montre le même ordre dans les deux approches.

4.3.5.2 *Inverseurs en cascade : a=1, e=0*

Dans ce cas (figure 4.8.b), supposons que l'entrée soit connue (a=1) et que la sortie mesurée e égale à 0, ainsi au moins, un des composants est fautif. Le meilleur point à tester est celui équidistant des deux extrémités déjà mesurée, comme les deux approches l'indiquent.

4.4 Exemples d'application

Pour illustrer et en même temps évaluer l'efficacité de la démarche que nous avons développée dans ce chapitre, nous allons l'appliquer sur des exemples.

4.4.1 Application à un système combinatoire

Pour ce cas, nous allons traiter la configuration Scan Complet du s27 de la série des benchmarks séquentiels ISCAS'89 [BBK89]. Cette configuration permet d'obtenir un module combinatoire composé de 10 blocs avec 7 entrées et 4 sorties primaires (figure 4.9).
Nous avons injecté une faute du type collage logique à 1 (Stuck-at-1) sur le nœud G15

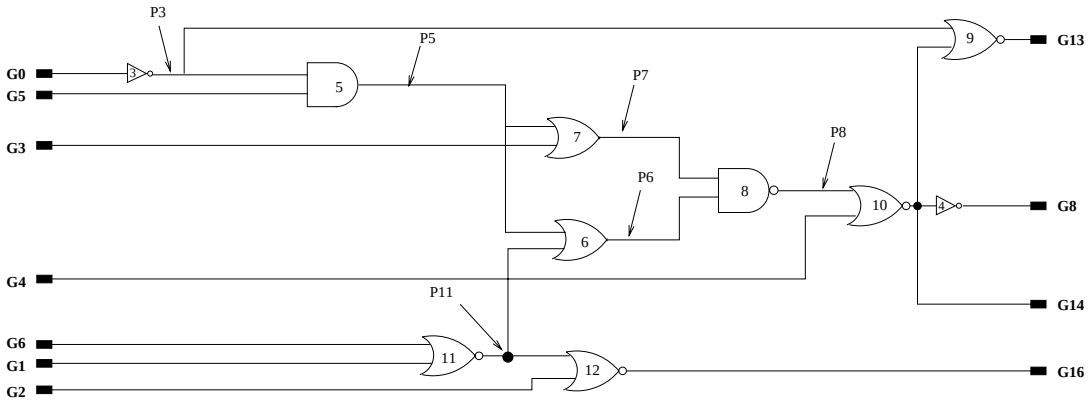


Figure 4.9: s27 scan complet (10 blocs dans ce module)

(point P11 sur la figure 4.9). Nous avons ensuite généré un ensemble de vecteurs de test permettant une couverture de 100% des fautes de collage. Deux de ces vecteurs permettent de détecter la faute injectée sur le nœud G15. Le premier vecteur exhibe cette faute sur la sortie G16, tandis que le deuxième le fait sur les sorties G16, G14 et G8. Deux sessions de diagnostic sont donc exécutées. Le tableau 4.4 montre les résultats de l'étape de la génération et de la mise à jour des candidats. Au cours de la deuxième étape, nous avons calculé l'entropie et le degré d'influence pour chaque nœud du circuit. Les résultats obtenus sont résumés dans le tableau 4.5. Les résultats obtenus sont satisfaisants. Pour trouver la faute, il fallait mesurer le point P11 pour découvrir que le bloc nor11 était fautif, vu que sa sortie aurait été erronée alors que

Candidat	not3	not4	and5	or6	or7	nand8	nor9	nor10	nor11	nor12
Session 1	S_C	C	S_C	S_C	S_C	S_C	S_C	V_C	S_C	S_F
Session 2	S_F	C	S_F	S_F	S_F	S_F	S_C	V_F	S_F	S_F
Résultat	Amb	C	Amb	Amb	Amb	V_C	Amb	Amb	V_F	V_F

Table 4.4: Génération des candidats pour le s27 Scan

Nœud à tester	**Entropie moyenne**	**Degré D'influence**	**Fonction de Coût**	**Classement Obtenu**
P3	0.627	2.247	0.635	sixième
P5	0.627	2.505	0.621	cinquième
P6	0.481	3.046	0.483	deuxième
P7	0.627	2.528	0.620	quatrième
P8	0.530	3.844	0.502	troisième
P11	0.046	1.998	0.187	premier

Table 4.5: Recherche du meilleur point à tester pour le s27 Scan

ses entrées étaient correctes. Notre stratégie de recherche du meilleur point à tester a précisément indiqué que le point P11 devait être testé en premier, ce qui conduisait à tomber directement sur le site de la faute.

Le deuxième essai porte sur le c432nr (figure 4.10). Ce circuit est composé de 157 portes, 36 entrées et 7 sorties primaires. Nous avons injecté une faute du type collage logique à 1 (Stuck-at-1) sur le nœud G107 (point P107 sur la figure 4.10). Nous avons ensuite généré un ensemble de vecteurs de test permettant une couverture de 100% des fautes de collage. Trois de ces vecteurs permettent de détecter la faute injectée sur le nœud G107.

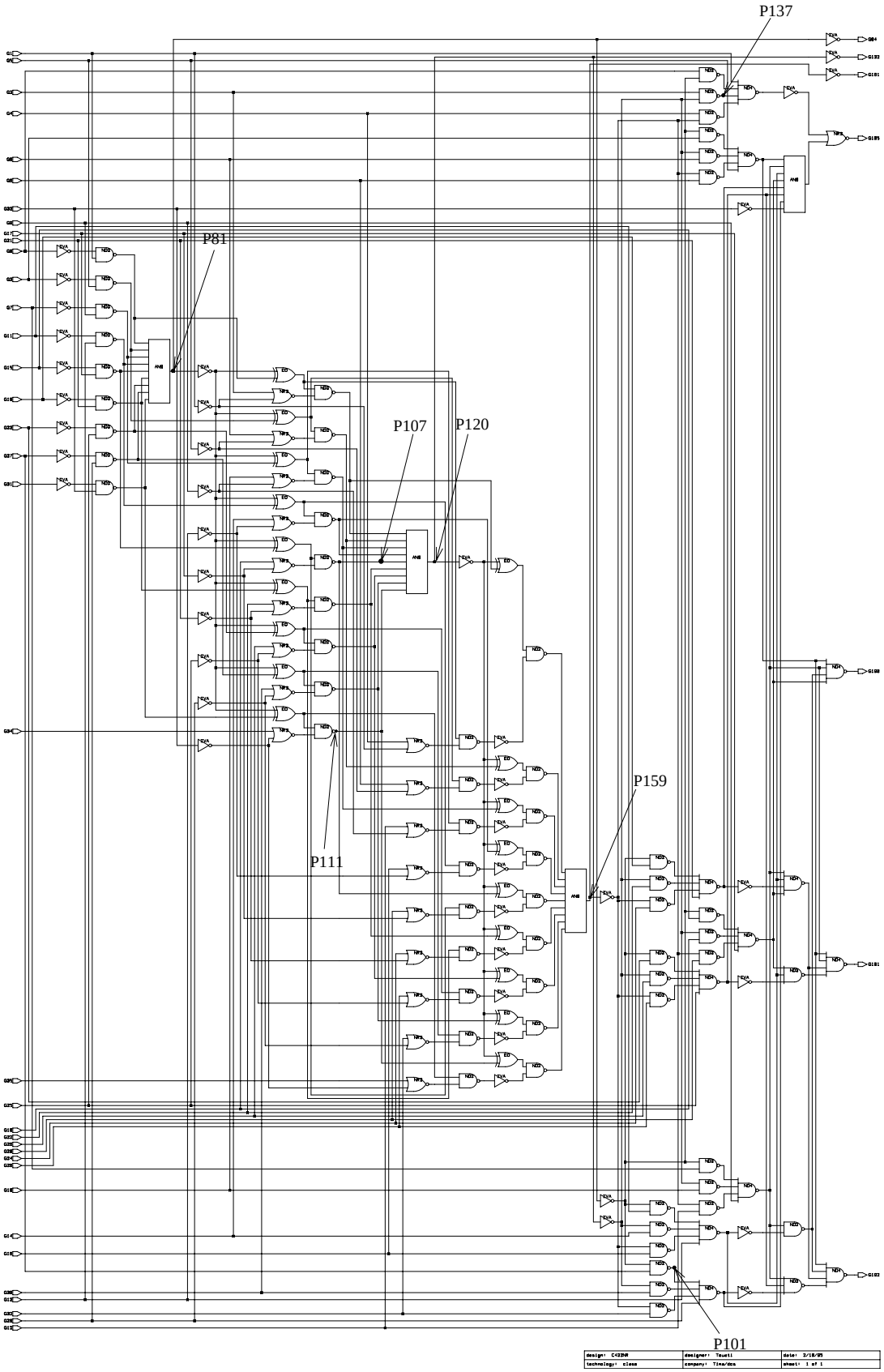


Figure 4.10: Module c432nr (157 blocs)

Les résultats obtenus avec les procédures de diagnostic et de localisation sont résumés dans le tableau 4.6. Nous remarquons que, avec seulement des informations limitées aux entrées/sorties primaires, et malgré l'importance relative de la taille du système (157 blocs), la stratégie de localisation a parfaitement fonctionné (convergence rapide vers le site de la faute). On notera également l'intérêt de l'utilisation de la distance de *Hamming* qui permet dans ce cas une meilleure discrimination que la fonction de coût.

Nœud à tester	Entropie moyenne	Degré D'influence	Fonction de Coût	Distance de Hamming	Classement Obtenu
P81	0.25	36.97	0.240	0.18	troisième
P101	0.29	12.38	0.300	0.36	sixième
P107	0.25	27.85	0.244	0.25	quatrième
P111	0.25	27.85	0.244	0.25	quatrième
P120	0.25	44.63	0.241	0.12	deuxième
P137	0.27	31.50	0.260	0.22	cinquième
P159	0.25	61.86	0.239	0	premier

Table 4.6: Recherche du meilleur point à tester pour le c432nr

4.4.2 Application à un système séquentiel

Pour illustrer la méthode sur un système séquentiel, nous allons traiter le module s27 auquel nous avons apporté une légère modification pour la clarté de l'explication. En effet, dans sa version originale (figure 4.11), ce circuit ne comprend qu'une seule sortie primaire, ce qui n'est pas vraiment le cas idéal pour le diagnostic.

La figure 4.12 montre le circuit modifié, sur lequel l'expérience a eu lieu. Nous avons rajouté une sortie supplémentaire et supprimé une boucle de rétroaction.
Nous avons injecté un collage logique à 1 sur le nœud G13 (P9). Dans l'ensemble de vecteurs de test que nous avons généré pour ce module, un seul vecteur permet de

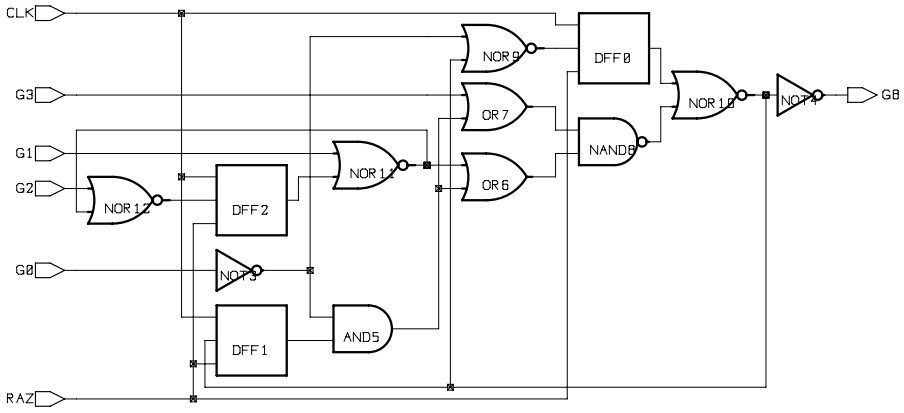


Figure 4.11: Module s27 original

détecter cette faute. La détection se fait sur la sortie G8. Une seule session de diag-

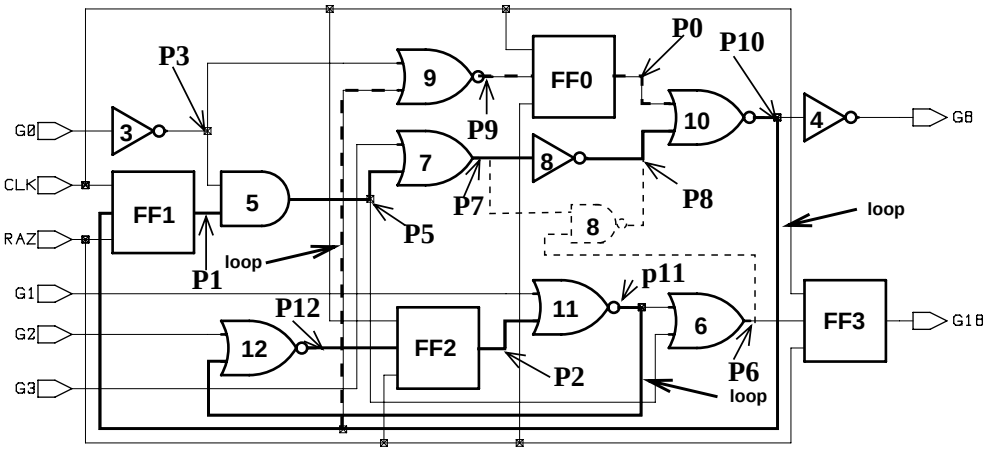


Figure 4.12: Module s27 modifié : 14 blocs dans le module

nostic est exécutée. Les résultats de la phase de génération des candidats sont donnés dans le tableau 4.7. Dans le tableau 4.8, nous présentons les résultats de la procédure de recherche du meilleur nœud à tester. Pour les modules séquentiels, nous avons à

Candidat	FF0	FF1	FF2	FF3	not3	not4	and5	or6	or7	not8	nor9	nor10	nor11	nor12
Résultat	S_F	S_F	S_C	S_C	S_F	S_F	S_F	S_C	S_F	S_F	S_C	V_F	S_C	S_C

Table 4.7: Génération des candidats pour le s27 modifié

résoudre le problème des boucles séquentielles. Comme on peut le constater dans le

tableau 4.8, tous les nœuds appartenant à une boucle ont des caractéristiques identiques (cône de fanin, cône de fanout et les estimations de fautes). Nous avons donc besoin d'au moins une information supplémentaire pour les différencier. A cet effet, nous utilisons la notion de profondeur de séquentialité comme paramètre de distinction entre ces candidats. Nous considérons que les éléments dont la profondeur de séquentialité correspond au nombre de coups d'horloges effectués au cours du test sont plus à même de contenir la ou les fautes, et que par conséquent ils doivent être sondés en premiers.

Nœud à tester	Entropie moyenne	Degré D'influence	Fonction de Coût	Prof_Seq		Classement Obtenu
				G8	G18	
P0	0.269	6.579	0.260	0	0	deuxième
P1	0.269	6.579	0.260	0	1	deuxième
P2	0.573	2.773	0.566	-	1	cinquième
P3	0.269	2.995	0.315	0	0	troisième
P5	0.269	6.579	0.260	0	1	deuxième
P6	0.370	6.499	0.342	-	1	quatrième
P7	0.269	6.579	0.260	0	0	deuxième
P8	0.269	6.579	0.260	0	0	deuxième
P9	0.269	6.579	0.260	1	1	premier
P10	0.269	6.579	0.260	0	0	deuxième
P11	0.573	2.773	0.566	-	1	cinquième
P12	0.573	2.773	0.566	-	2	cinquième

Table 4.8: Recherche du meilleur point à tester pour le s27 modifié

4.4.3 Application à un circuit analogique

Le circuit suivant (figure 4.13.a), qui a été proposé dans [DRDM87] et traité par le système DEDALE, montre la différence de volume de travail demandé pour son diagnostic par FLAMES par rapport à celui de DEDALE. Le circuit de la figure 4.13.b

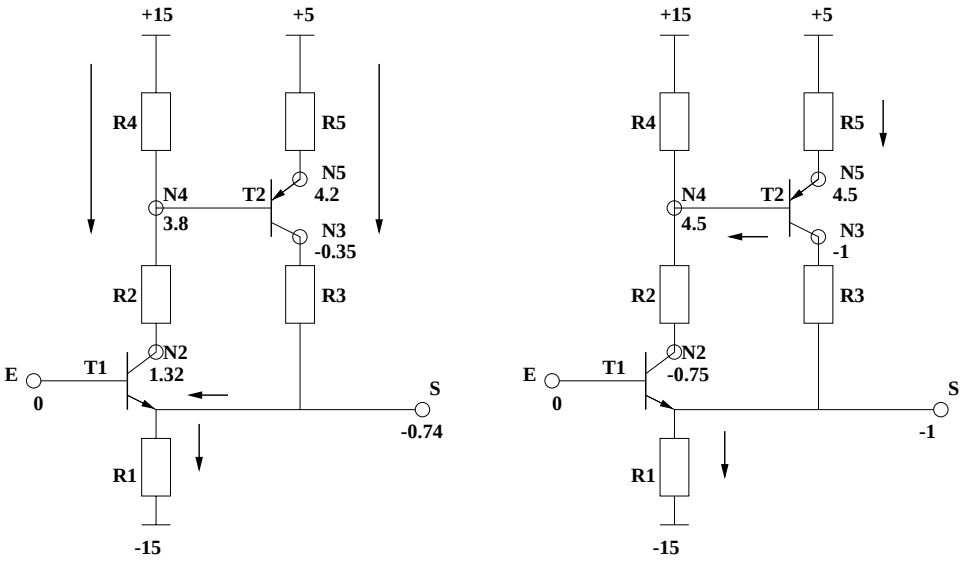


Figure 4.13: Circuit simple étudié par DEDALE

est le circuit défectueux. La tension observée à l'entrée est égale à celle du circuit nominal, par contre celui de la sortie est sensiblement inférieure.

DEDALE réagit comme suit [DRDM87] : il trouve que les ensembles des hypothèses qui sont cohérentes avec les mesures prises sont :

B2 = [N2,T1,R2] A(B2) = {{N2,T1(on),R2},{N2,T1(s),R2}}

B3 = [N3,T2,R3] A(B3) = {{N3,T2(op),R3}}

B4 = [N4,T2,R2,R4] A(B4) = {{N4,T2(op),R2,R4}}

B5 = [N5,T2,R5] A(B5) = {{N5,T2(op),R5}}

où Ri et Ni correspondent aux modèles uniques de bon fonctionnement des résistances et des nœuds.

DEDALE continue à examiner ces ensembles de candidats. Il appelle le système FOG pour les analyser et essayer de déduire de nouvelles relations possibles entre les composants. Ainsi, après un calcul long et délicat dans lequel le système FOG (section 2.1.1) est invoqué pour trouver des candidats potentiels en considérant ses règles d'inférence qualitatives à base des opérateurs $\cong, \ll, et \approx$ afin de tester la cohérence de chaque ensemble des quatre ensembles donnés ci-dessus (le processus complet peut être trouvé dans [DRDM87]. DEDALE finit par déduire deux *conflits*, qui correspondent aux ensembles minimaux de candidats, qui sont les suivants : <N4,T2,R2,R4>

et <N5,T2,R5>.
Si les nœuds sont considérés comme corrects, alors les trois candidats minimaux seront [T2], [R2,R5] et [R4,R5]. Finalement, en considérant l'hypothèse de la faute simple, DEDALE pourra conclure que T2 est le composant fautif ; c'est le seul composant qui peut expliquer les deux conflits.

FLAMES, par contre, bénéficie du degré Dc pour arriver à la même conclusion beaucoup plus rapidement, comme on va le montrer [Moh97].
Les quatre mêmes ensembles de candidats sont formés et attachés aux nœuds du circuit. Le processus du meilleur nœud à tester démarre en assemblant de nouvelles informations afin de raffiner le diagnostic.
Deux paramètres sont considérées pour le choix du nœud à tester : le degré d'influence et l'entropie floue.
En ce qui concerne l'entropie floue, on peut supposer que les estimations de défectuosité des composants sont : *Plutôt_Fautif* pour les résistances et *Vraisemblablement_Fautif* pour les transistors.
Donc, en prenant les mêmes ensembles de candidats considérés plus haut, on aura le tableau 4.9. Quant aux degrés d'influence, on trouve que les ensembles des composants constituant les *fanins* et le *fanouts* sont :

- N2 : {R2,T2,R5,R4} comme fanin et {T1,R1} comme fanout;
- N3 : {R5,T2} comme fanin et {R3} comme fanout;
- N4 : {R5} comme fanin et {T2,R3,R2,T1,R1} comme fanout;
- N5 : {R4,T2,R5} comme fanin et {R2,T1,R1} comme fanout;

Ainsi, ces informations, qui sont résumées dans le tableau 4.9, montrent qu'en mesurant deux points seulement on peut tomber sur l'élément fautif, qui est T2 dans ce cas [MMT97].
En effet, mesurant N4 puis N5, N4 puis N3, N5 puis N4 ou N3 puis N4 peut fortement suspecter le transistor T2 puisque le signe de Dc peut jouer un rôle important, comme le montre ce qui suit :

- Mesurer la sortie indique que la déviation est vers la gauche : la valeur mesurée est située à gauche de la valeur nominale $\Rightarrow$ Dc= -1;

Nœud	Entropie	Df	Ordre
N2	0.22	3.305	troisième
N3	0.17	2.047	quatrième
N4	0.23	4.546	premier
N5	0.20	3.672	deuxième

Table 4.9: Recherche du meilleur nœud à tester pour le circuit analogique étudié

- Mesurer N4 indique que la déviation est vers la droite ⇒ Dc=+1;
- Mesurer N3 ⇒ Dc=-1
- Mesurer N5 ⇒ Dc=+0.7

Alors en regardant de près le flux de ces valeurs, on verra que pour N3 (ou N5) et N4 le signe de Dc a été inversé.

4.4.4 Application à un système analogique ou mixte

Cette méthode du meilleur nœud à tester est directement applicable dans le cas des systèmes analogiques et mixtes puisque les composants du système peuvent être vus comme étant des blocs. La modélisation d'un système est plus simple que celle des circuits analogiques.

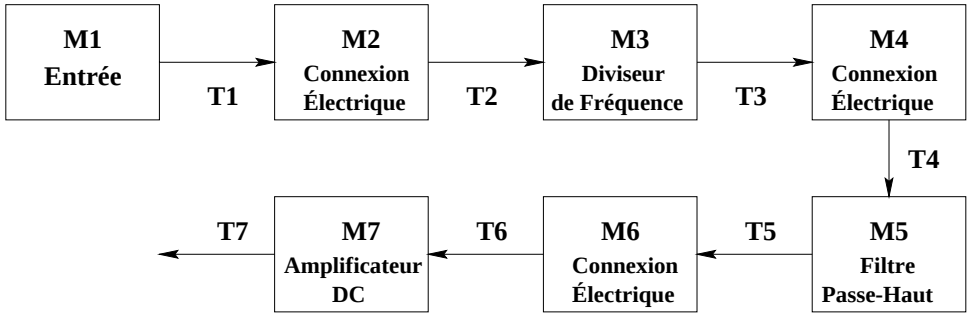
Le cas des circuits analogiques est le plus dur à traiter et ceci en se référant à tout ce qui a été dit dans cette thèse. Par contre, un système peut être modélisé par des méthodes qualitatives simples, comme celle exposée dans FIS [Pip86], qui se révèlent suffisantes pour ce genre de traitement.

Un système est modélisé dans FIS en décrivant son comportement correct et son comportement fautif par des règles qualitatives enchaînées afin de modéliser les interconnexions. L'exemple de la figure 4.14 constitue un exemple simple qui justifie ceci. Des règles comme les suivantes modélisent les systèmes sous test dans FIS :

* *"Given that T4 frequency is high can cause frequency high at T5"*
* *"Given that Dc voltage is high at T3 can cause frequency low at T4"*

* *"Given that M3 is not dividing can cause frequency high at T3"*

Noter que la troisième règle est une "règle d'anomalie". La même figure (4.14)



Résultats de Test	Ensembles d'Ambuïté
T6 freq High	{ M1, M3}
T6 dc Low	{M5, M6, M7}
T6 dc High	{M5}

Figure 4.14: Un exemple simple traité par FIS

montre le système étudié et un tableau qui contient les résultats de tests et leurs ensembles de candidats (nommés ensembles d'ambiguïté dans FIS) qui justifient les fautes.

La deuxième étape du diagnostic faite par FIS est un processus "meilleur nœud à tester" basé sur une approche probabiliste avec ses problèmes (nous avons discuté cette approche dans 2.3).

Ainsi, notre approche se montre directement applicable sur l'exemple de la figure 4.14. Si nous reconsidérons l'entropie floue prévue étudiée dans 4.3.5 et son exemple de démonstration on pourra facilement constater qu'une correspondance entre les deux exemples est faisable.

4.5 Conclusion

Étant donné le niveau de complexité atteint par les systèmes VLSI actuels d'une part, et la difficulté de test et de diagnostic des circuits analogiques et mixtes d'autre

part, les méthodes de diagnostic reposant uniquement sur des mesures effectuées sur les entrées/sorties primaires ne peuvent plus permettre une localisation précise la (ou les) faute(s). Souvent des mesures supplémentaires sont nécessaires, que se soit par l'intermédiaire d'équipements spéciaux (microscope électronique, sondes mécaniques.. etc) ou par l'intermédiaire des dispositifs de conception en vue d'une meilleur testabilité (notamment le Scan Path pour les circuits, et le Boundary Scan pour les cartes ou les MCMs).

Dans ce chapitre, nous avons présenté une méthode de recherche automatique des meilleurs nœuds à tester. Elle vient compléter l'approche de diagnostic présentée dans le chapitre précédent dans le but d'améliorer le processus de localisation des fautes. Basée sur un raisonnement qualitatif pour la génération et la classification des candidats en ce qui concerne les circuits digitaux et sur un raisonnement semi-qualitatif en ce qui concerne les circuits analogiques, cette méthode utilise des estimations qualitatives basées sur des quantifications linguistiques, permettant ainsi d'éviter de manipuler des calculs lourds et fastidieux. La fusion des différentes sessions de test est effectuée à l'aide d'une matrice ce qui permet une simplification du processus et une grande malléabilité quant à la définition de la granularité du diagnostic.

La technique proposée n'est pas liée à un modèle de faute particulier et ne suppose aucune information sur la fonctionnalité des blocs internes composant le système. Néanmoins, si on se limite à la phase de génération des candidats, ceci peut s'avérer suffisant pour des petits systèmes. Mais ce n'est généralement pas suffisant pour effectuer un bon diagnostic lorsqu'il s'agit de systèmes un peu plus complexes. Afin de remédier à ces insuffisances, nous avons complété cette méthode par une stratégie qui repose sur l'analyse de la structure du système sous test ainsi que sur les informations issues de la première étape de diagnostic.

La fonction de coût définie pour estimer l'importance de chaque nœud est facile à mettre en œuvre, et ne demande pas de calculs fastidieux. Les premiers résultats montrés dans ce chapitre sont plutôt prometteurs. D'autres résultats seront présentés

dans le dernier chapitre, qui montrent une bonne efficacité de l'ensemble de l'approche de diagnostic développée.

Chapitre 5

Unité de prise de décision floue

Chapitre 5

Unité Prise de Décision Floue

Ce chapitre présente l'unité de prise de décision floue de FLAMES. Cette unité qui traite le problème connu où l'expert (ou l'utilisateur) doit choisir (prendre une décision) entre plusieurs solutions (alternatives) d'un problème à plusieurs contraintes (attributs) où chaque alternative est caractérisée par les valeurs de ses attributs. Les attributs n'ont pas, en général, les mêmes poids. D'ailleurs, leurs valeurs peuvent être numériques (précises), imprécises ou même des termes linguistiques.
Plus généralement, l'environnement du travail n'est pas toujours précis : il peut être imprécis et vague. L'imprécision peut être introduite à plusieurs niveaux :

- les opérateurs,
- les valeurs données,
- les informations données par l'expert, etc.

La **logique floue** est très convenable pour le traitement de ce genre de problèmes comme le montrent les sections suivantes.

Dans le domaine de la conception électronique, par exemple, et avec l'explosion dans le nombre de possibilités de conception, les concepteurs rencontrent des problèmes de prise de décision pour choisir la meilleure méthode de DFT (Design-For-Testability). Les concepteurs ont besoin d'outils de CAO pour organiser, évaluer et choisir les techniques de test qui satisfont les objectifs de la conception. Mais, quand les objectifs ne sont pas définis d'une manière précise, le processus de prise de décision devient plus

difficile à gérer. En effet, l'environnement devient flou et le processus lui même devient flou, i.e, il doit considérer des stratégies floues.

Cette unité a été développée et ajoutée à FLAMES pour aider le test, le diagnostic des CA, ainsi que pour proposer des méthodes efficaces pour le choix de DFT.
L'approche développée présente un processus de prise de décision flou, mais à plusieurs niveaux. Cela veut dire que la notion de flou est la plus générale, ce qui est utile et valable dans n'importe quel système où des tolérances des différents types sont utilisées.

La prise de décision floue a été déjà abordée dans ce domaine [FK93], où les auteurs ont montré son utilité pour la vérification fonctionnelle de circuits et pour le choix de DFT approprié. Nous allons tout d'abord présenter leur approche puis notre approche multi-niveaux.

5.1 Prise de décision dans un environnement flou

L'environnement du travail est un espace de test d'exploration où un grand choix de MDFT (méthodologie de DFT) se présente au concepteur.
Du point de vue de la prise de décision, le problème de sélection peut être formulé en utilisant un seul modèle de décision. Ce modèle assume que le concepteur peut choisir une parmi **m** possibilités (alternatives) Alt_j (j = 1, ... , m). Chacune de ces alternatives peut être décrite par un nombre de paramètres ou d'attributs a_i (i = 1, ... , n). Les attributs peuvent refléter les différents coûts et gains associés à une MDFT spécifiée.
Ce modèle peut être résumé par la figure 5.1.a [MTM96], où une matrice de **n** lignes (attributs) et de **m** colonnes (alternatives) le décrit. Un ensemble d'objectifs est donné sur le plan des conditions que le choix doit vérifier. Le problème de décision est difficile parce que le nombre d'alternatives est grand, les attributs ne peuvent être comparés directement et leur nombre peut aussi être grand, les contraintes et les objectifs des attributs ne sont pas de même importance, et enfin, dans la plupart des

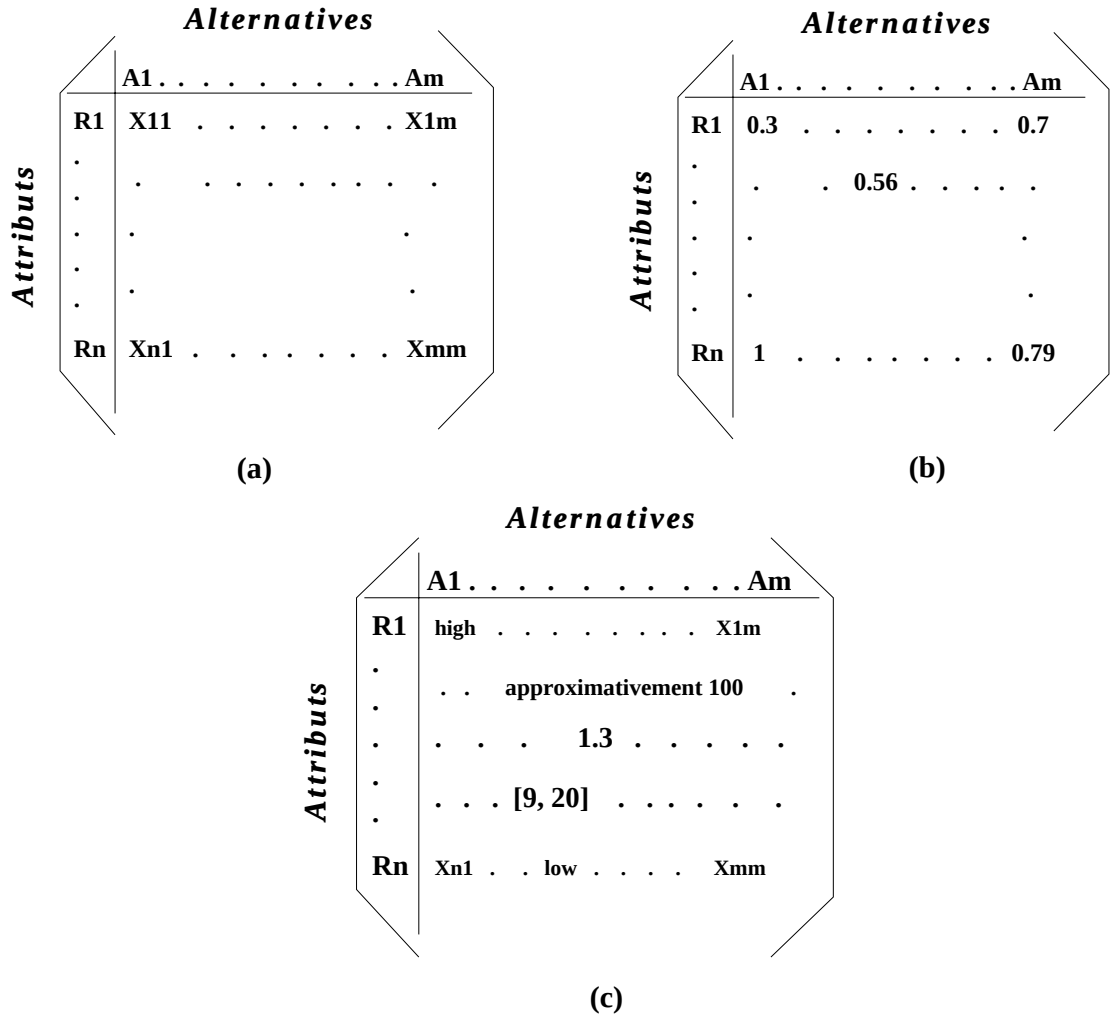


Figure 5.1: Les matrices d'évaluation

cas il n'existe pas une alternative unique qui satisfait toutes les conditions.

*** Exemple explicatif**

Supposons qu'on a un espace **S** de MDFTs [FK93] :

S = $MDFT_1, ..., MDFT_n$ et un ensemble d'attributs : taux de couverture (TC), surface additionnelle (SA), et délai supplémentaire (DS), par exemple. Les objectifs du concepteur peuvent être donnés comme suit :

R_1 : La MDFT doit avoir un TC approximativement égale à 100

R_2 : La MDFT doit avoir une SA acceptable ;

R_3 : La MDFT doit avoir un DS très faible.

Une quantification précise est très difficile et le sens flou est clair dans ces objectifs ce qui conduit à utiliser les nombres flous (figure 5.2) pour quantifier et pouvoir manipuler ce problème ou un problème analogue.
Supposons, maintenant, que S a un ensemble de 5 MDFTs :

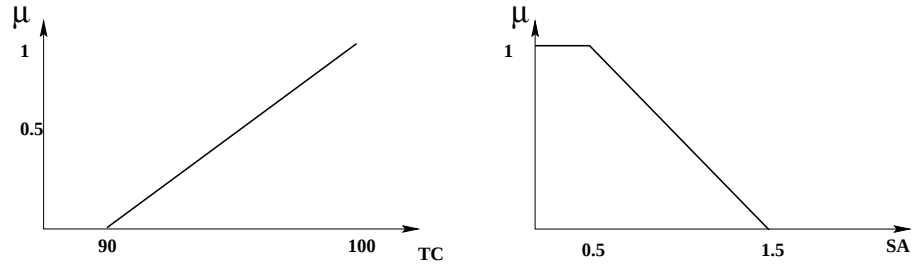


Figure 5.2: Ensembles flous pour TC et SA

$$S = MDFT_i \; i = 1, \ldots, 5$$

La classe A des MDFTs qui ont des surfaces supplémentaires *faibles*, est un ensemble flou. Cet ensemble peut être caractérisé par la fonction d'appartenance présentée en figure 5.3. Supposons qu'on a les valeurs suivantes de SA pour les MDFTs du tableau 5.1. Pour tous les MDFTs, on obtient l'ensemble flou suivant :

MDFTs	MDFT1	MDFT2	MDFT3	MDFT4	MDFT5
SA(%)	0	22.5	8	11.5	5

Table 5.1: Un exemple simple

$\mathbf{A} = \{ MDFT_1/1, MDFT_2/0.25, MDFT_3/0.73, MDFT_4/0.62, MDFT_5/0.83 \}$. Les

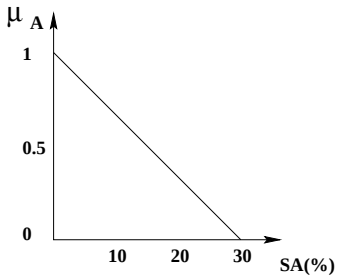


Figure 5.3: Fonction d'appartenance pour la classe A

concepts dans cette approche seront définis comme des ensembles flous. Alors, chaque

condition R_i sur un attribut a_i sera caractérisée par une fonction d'appartenance μ_{Ri} :

$$\mu_{Ri} : \mathbf{S} \rightarrow \mathbf{[0,1]}$$
$$Alt_j \rightarrow \mu_{Ri}(Alt_j)$$

Une fois les fonctions d'appartenance définies, le degré d'appartenance de chaque Alt_j à l'ensemble flou R_i est évalué.

Pour les **n** conditions $(R_1, R_2, \ldots\ldots, R_n)$, on obtient la matrice de la figure 5.1.b, où les valeurs $\in$ [0,1], ce qui crée ce que nous appelons un premier niveau de flou.

5.2 La prise de décision floue

L'alternative choisie doit satisfaire :

$$R_1 \ et \ R_2 \ldots\ldots \ et \ R_n$$

Puisque *et* correspond à l'intersection, ceci donne :

$$R_1 \cap R_2 \ldots\ldots \cap R_n$$

Ainsi, la décision prise D sera définie par :

$$\mathbf{D} = R_1 \cap R_2 \ldots\ldots \cap R_n$$

Et comme l'intersection se traduit par **min** dans le cadre de la logique floue. La fonction d'appartenance de D devient :

$$\mu_D(Alt_j) = min(\mu_{R1}(Alt_j), \ldots\ldots, \mu_{Rn}(Alt_j))$$

La décision optimale est une alternative Alt_j de **S** qui maximise $\mu_D(Alt_j)$:

$$\mu_D(Alt_{op}) = max(\mu_D(Alt_j)) \text{ j} = 1, \ldots\ldots, \text{m}$$

Enfin, notons qu'on peut introduire les degrés d'appartenance des objectifs pour mieux choisir l'alternative optimale.

Cette technique peut être appliquée aux différents problèmes, ceux rencontrés dans le domaine de conception des circuits digitaux et analogiques. La vérification fonctionnelle d'un circuit analogique est un bon exemple illustratif. Cependant, les spécifications de la conception sont exprimées d'une manière floue, et une quantification précise peut limiter l'espace des bons circuits ce qui cause le rejet fonctionnel des bons circuits.

Donc, en définissant un "bon circuit" par des spécifications convenables et les fonctions d'appartenance correspondantes, on peut calculer la fonction générale μ_D qui évalue un circuit. Quand le degré d'appartenance est égal à 1, on a un circuit qui satisfait complètement toutes les spécifications. Une valeur entre 0 et 1 indique le degré d'appartenance à l'ensemble de bons circuits. Nous allons considérer un exemple de ce genre pour illustrer cette approche et la nôtre.

L'approche développée dans [FK93] est limitée au premier niveau de flou où les opérateurs sont flous, ce qui n'est pas suffisant pour satisfaire les besoins des circuits analogiques, puisque la recherche dans ce domaine cherche à introduire les tolérances aussi bien en simulation, qu'en test et diagnostic. Alors, on doit s'attendre aux valeurs avec tolérances (comme dans FLAMES) ce qui nous conduit vers le développement nécessaire d'une nouvelle approche générale pour la prise de décision floue où plusieurs niveaux de flou peuvent être introduits : pour les valeurs, les poids sur les attributs, et les attributs eux-mêmes. Ainsi, cette unité de FLAMES, qui implémente cette approche [MTM96], est appelée pour la recherche du meilleur nœud à tester et pour la vérification fonctionnelle et le choix du meilleur DFT.

La figure 5.1.c résume le problème le plus général. Dans un environnement imprécis général, les valeurs X_{ij} peuvent être des nombres ou des intervalles ordinaires, des nombres ou des intervalles flous, ou même des termes qualitatifs linguistiques (comme *low, high, etc.*).

Afin de bien présenter l'approche, nous allons reprendre le problème dès le premier niveau.

5.2.1 Premier niveau de flou : attributs flous

Dans ce cas, les valeurs et les poids sont ordinaires, mais les attributs sont définis via des ensembles flous. La meilleure alternative peut être exprimée par [FK93] :

$$\mu_D(Alt_{best}) = \mathbf{max_{j=1}^{m}}(\sum_{i=1}^{n} \alpha_i \;\; \mu_{Ri}(Alt_j))$$

où αi (poids) ≥ 0 for i= 1, ... , n

et $\frac{1}{n}\sum_{i=1}^{n} \alpha i = 1$

5.2.2 Second niveau de flou : valeurs et attributs flous

Dans ce cas, les valeurs données par un simulateur ou par un expert sont définies comme des intervalles flous. Puisque les attributs sont définis par des ensembles flous, alors on a besoin d'une technique qui peut comparer un intervalle flou avec un ensemble flou, ou bien deux intervalles flous. Une telle technique, qui a été développée et utilisée dans FLAMES (voir section 3.2.5, la figure 5.4.a) rappelle la méthode expliquée dans la section 3 où on calculait à quel degré un intervalle flou satisfait une contrainte (figure 5.4.b). Dans la figure 5.4.c, par exemple, l'objectif est satisfait avec un degré égal à 0.0833, qui correspond exactement à la surface de la zone remplie. En fait, le

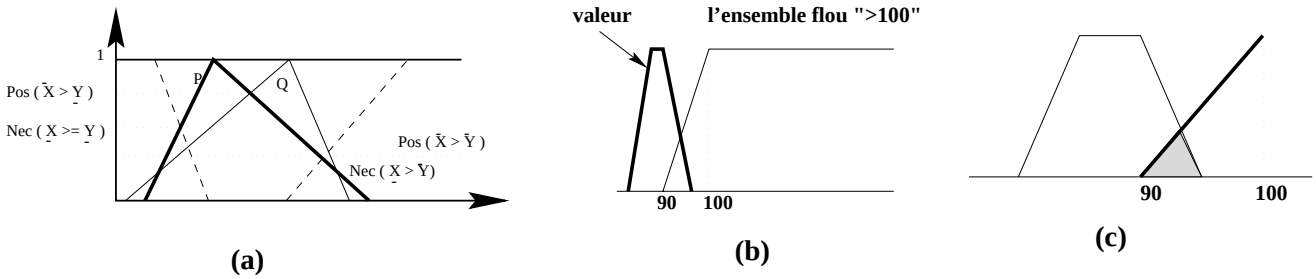


Figure 5.4: Comparaison entre deux intervalles flous

calcul de ce degré revient à comparer deux intervalles flous. En pratique, les quatre indices, vus sur la figure 5.4 où Pos(X>Y) signifient la possibilité que X soit supérieure à Y et Nec(X>Y) la nécessité que X soit supérieure à Y, suffisent pour discuter les positions relatives de deux intervalles flous [DP87].

L'opérateur > (I1, I2), où I1 et I2 sont des intervalles flous, calcule à quel degré I1 est supérieur à I2, il est défini comme la moyenne des quatre degrés précédents. Ainsi, < sera son complément à 1.

5.2.3 Cas général : valeurs, poids et attributs flous

Il nous reste à aborder la question importante de choix de poids. Une approche, qui a été largement appliquée dans ce domaine, est celle de Saaty *Analytic Hierarchy Process* [Saa77]. Elle est basée sur une comparaison par paires entre les éléments, ce qui est très lourd et il est possible d'obtenir des résultats qui n'expriment pas honnêtement les importances des objectifs.
Supposons que l'expert a défini N objectifs, il sera alors demandé de les comparer par des comparaisons binaires. S'il compare l'élément i avec l'élément j deux valeurs m_{ij} et m_{ji} seront déclarées comme suit :

1. $m_{ii} = 1$
2. $m_{ij} = 1/m_{ji}$
3. Si i est plus important que j, un nombre réel reflétant l'intensité de l'importance est assigné à m_{ij}.

Ensuite, une matrice NxN est construite. Les poids α_i sont obtenus à partir de cette matrice et en utilisant la méthode de Saaty nommée *Saaty's eigenvector method* [Saa77] [FK94].

L'exemple suivant illustre notre point de vue à propos de cette méthode. Considérons que nous avons trois objectifs spécifiés par les comparaisons suivantes :

- R_2 *est légèrement plus important que* R_1
- R_3 *est quelque part entre égal et légèrement plus important que* R_1
- R_2 *est légèrement plus important que* R_3

Pour ces comparaisons entre R_1, R_2 *et* R_3, la méthode de Saaty donne comme résultat final (il y a aussi une étape transitoire de calcul matriciel !) : 0.48, 1.77 et 0.75 pour R_1, R_2 *et* R_3, respectivement, qui ne correspondent pas vraiment aux spécifications des poids (figure 5.5).

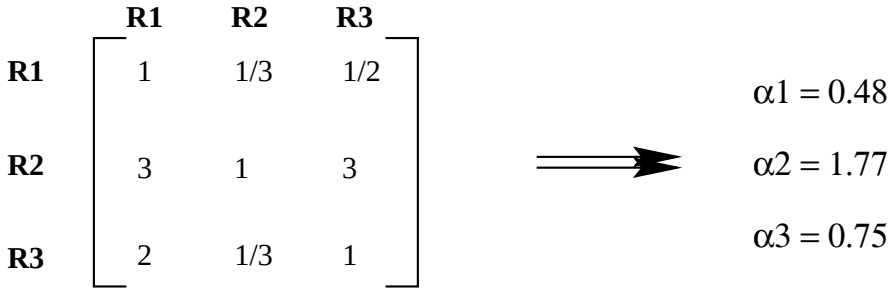


Figure 5.5: Matrice et poids par la méthode de Saaty

5.2.3.1 *Détermination des poids*

La méthode de Saaty a été développée pour le cas des nombres flous [Zim87], ce qui la rend plus lourde et rend les calculs plus complexes que dans les cas de calcul matriciel ordinaire.

Nous avons développé une approche qualitative floue et interactive pour la détermination des poids. Elle est plus simple et plus compatible avec l'esprit de notre système et notre approche de prise de décision. L'idée est de décomposer l'intervalle [0,1] en plusieurs intervalles flous qui correspondent aux poids que l'expert peut utiliser (figure 5.6). Noter que cette décomposition n'est pas unique, sa granularité peut être changée d'un exemple à l'autre. L'expert doit choisir l'échelle la plus convenable à son problème.

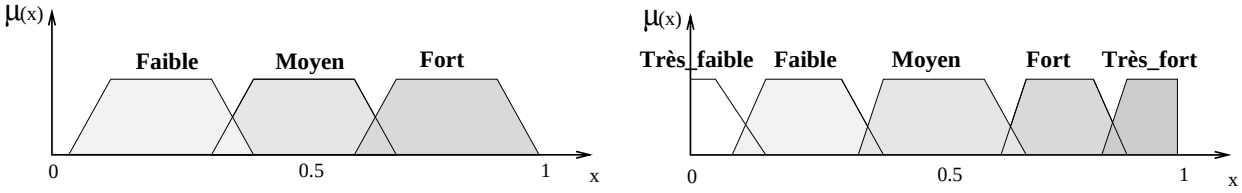


Figure 5.6: Des granularités pour des poids flous

5.2.3.2 *Modificateurs linguistiques*

Mais, comme ces poids peuvent être insuffisants pour exprimer (et distinguer) l'importance réelle, FLAMES offre un ensemble de modificateurs (comme *very, approximately, etc.*) qui aident l'expert à mieux décrire et avec une meilleure précision

les poids. Par exemple, un objectif R_i peut avoir *strong* comme poids, tandis qu'un autre R_j peut avoir *very-strong*.

Les modificateurs, qui sont implémentés par des opérateurs flous, offrent un grand choix et par conséquent une bonne précision pour le choix des poids. Pour l'instant, *very, more-or-less, approximately, rather, about, strongly,* et *not* sont définis, mais cet ensemble peut être facilement étendu. Les définitions de ces modificateurs sont les suivantes [BM91] [BZ87] :

very : $\forall$ x $\in$ U, μ_{veryA}(x)=$(\mu_A(x))^2$

more-or-less : $\forall x \in U, \mu_{more-or-lessA}(x) = \sqrt{\mu_A(x)}$

not : $\forall$ x $\in$ U, μ_{notA}(x)=1-$\mu_A(x)$

approximately : $\forall$ x $\in$ U, $\mu_{approximatelyA}$(x)=min(1,$\lambda\ \mu_A(x)$) avec $\lambda \in$ [1,2]

rather : $\forall$ x $\in$ U, $\mu_{ratherA}$(x)=max(0,$\nu\mu_A(x)$+1-ν) avec $\nu \in$]1/2,1[

about : $\forall$ x $\in$ U, μ_{aboutA}(x)=min(1,max(0,μ_A(x)+μ)) avec $\mu \in$]0,1/2[

strongly : $\forall$ x $\in$ U, $\mu_{stronglyA}$(x)=μ_A(x-γ), avec $|\gamma| \leq$ min(m2-m1,$(\alpha+\beta)/2$)

pour M=[m1,m2,α, β]

La figure précédente (fig 5.7) montre bien l'influence de chaque modificateur sur le

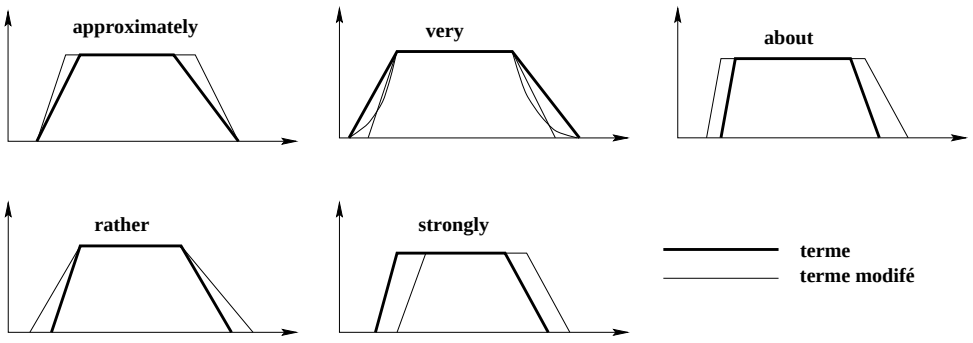


Figure 5.7: Modificateurs définis pour le calcul des poids

noyeau, les étalements ou sur tous les élements de l'ensemble.

Notons que l'influence du modificateur *very* a lieu sur les étalements seulement (pas sur le noyau de l'ensemble flou) à cause du choix de l'opération carrée pour ce modificateur et le résultat ne sera pas un intervalle flou (figure 5.7 : la courbe sur le modificateur *very*) qui est très difficile à gérer, alors nous avons choisi de prendre comme approximation la relation suivante :

very([m1,m2,α, β]) = [m1,m2,$\alpha/2, \beta/2$] comme le montre la même figure.
L'influence des modificateurs peut être mieux expliquée par l'exemple principal (figure 5.8) pris par Zadeh lui même dans beaucoup de ses articles [Zad75]. Cet exemple montre l'influence des modificateurs sur la fonction d'appartenance *young*.

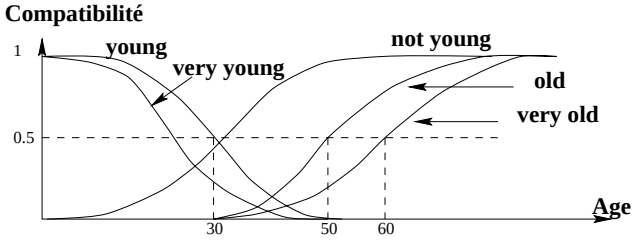


Figure 5.8: Fonctions de cohérence entre *young* et ses modificateurs

5.2.3.3 *Détermination des poids*

Dans des cas simples, la détermination des poids peut être faite en assignant des poids linguistiques (ou des poids avec modificateurs) aux objectifs. Par exemple, on peut décrire les objectifs suivants par :

- R_1 a une *forte* influence.
- R_2 a une *très forte* influence.
- R_3 a une *faible* influence.

Il est très important de noter qu'on peut utiliser une combinaison de modificateurs.

Finalement, afin d'obtenir une bonne approximation de poids, l'expert doit décrire l'influence et offrir des comparaisons possibles entre les objectifs.

5.2.3.4 *La meilleure alternative*

Les valeurs et les poids sont des intervalles flous dans ce cas, la meilleure alternative peut être calculée comme suit :
* La satisfaction de l'alternative j à tous les objectifs est :

$$Sat(Alt_j) = \bigoplus_{i=1}^{n} \alpha_i \otimes Val_{ij}$$

où α_i est l'intervalle flou qui représente le poids de l'objectif et Val_{ij} est la valeur de l'alternative j correspondant à l'objectif i.
Finalement, $\oplus$ et $\otimes$ sont, respectivement, l'addition et la multiplication floues entre intervalles.
* La meilleure alternative est celle qui maximise la satisfaction, alors :

$$Alt_{best} = max(Sat(Alt_j)) \textbf{ for j = 1, ...,m}$$

5.2.4 Exemple : vérification fonctionnelle

Le tableau de la figure 5.9 montre les résultats de cette approche appliquée à un circuit analogique (un amplificateur opérationnel CMOS à deux étages) extrait de [FK93] sur lequel les auteurs ont appliqué leur approche. La première colonne contient les attributs, la deuxième les tolérances, tandis que la troisième montre les valeurs simulées par HSPICE et la 4ème les résultats. Les auteurs assument qu'un bon circuit est défini par les spécifications suivantes que nous considérons aussi :

- $gain > G_{min}$, $gain\text{-}bandwidth > GB_{min}$,
- $settling\ time < ST_{max}$, etc.

Notons que dans [FK93], la ressource de flou est constituée des opérateurs $< et >$. Mais pour nous, les opérations, les poids et les valeurs sont flous. Ainsi, les deux dernières colonnes donnent les résultats quand les tolérances sont appliquées au même circuit : les valeurs obtenues sont des nombres flous donnés sous la forme $[M, \alpha, \beta]$ et les degrés d'appartenance ont été calculés par la méthode de la section 5.2.2.
Le degré d'appartenance global qui peut être interprété comme la mesure de fonctionnalité du circuit est de 0.8 pour les valeurs sans tolérance [FK93] et de 0.74 pour les valeurs avec tolérance. Ce qui montre la validité de notre approche.

5.2.5 Conclusion

Ce chapitre a présenté une nouvelle technique à base de logique floue pour résoudre le problème de la prise de décision dans un environnement flou. La technique développée

Paramètre	Spécification	Tolérance	HSPICE	μ	Valeurs avec tolérances	μ
Gain (dB)	>70	10	84.3	1.0	[84.3, 2, 3]	1.0
Gain-bandwidth (MHz)	> 1	0.2	0.9	0.5	[0.9, 0.1, 0.1]	0.3
Power dissipation (mW)	< 10	2	1.7	1.0	[1.7, 0.5, 0.6]	1.0
Phase margin (deg)	> 60	5	58	0.6	[58, 4, 5]	0.46
Slew rate (V/μs)	> 2.0	0.2	1.9	0.5	[1.9, 0.1, 0.15]	0.4
Settling time (μ s)	< 1	0.1	0.5	1.0	[0.5, 0.1, 0.09]	1.0
PSRR (dB)	> 60	5	89	1.0	[89, 5, 4]	1.0

Figure 5.9: Prise de décision quand les valeurs ont des tolérances

est capable de traiter des valeurs données avec des tolérances sous formes d'intervalles ordinaires, d'intervalles flous ou même sous forme des termes qualitatifs.
Cette technique peut être utile dans le domaine de la conception électronique, par exemple, pour le choix du meilleur DFT à appliquer ou pour choisir les techniques de test qui satisfont les objectifs de la conception surtout quand les objectifs ne sont pas définis d'une manière précise. Le processus de prise de décision devient plus difficile à gérer. En effet, l'environnement devient flou et le processus lui même devient flou, i.e, il doit considérer des stratégies floues.

Cette unité a été développée et ajoutée à FLAMES pour aider le test, le diagnostic des CA, ainsi que pour proposer des méthodes efficaces pour le choix de DFT.

Chapitre 6

Implémentation, expérimentation et résultats

Chapitre 6

Implémentation, Expérimentations et Résultats

6.1 Implémentation

FLAMES a été implémenté en C++ sur Sun SparcStation, puisque les modèles de l'approche de raisonnement à base de modèles profonds, ainsi que les composants des circuits électroniques, peuvent être facilement représentés en utilisant un langage orienté-objet.

Les classes représentent les différents types de composants pour lesquels les méthodes correspondantes et les modèles corrects sont attachées. Ainsi, l'instanciation représente un circuit spécifique à diagnostiquer.

Toutes les opérations, arithmétiques ou logiques, dans FLAMES sont des opérations floues (celles de la section 1.4.9). Une classe nommée *fuzzy number* a été définie avec toutes les opérations possibles attachées comme *méthodes*, ainsi l'utilisateur peut utiliser les signes conventionnels $+, -, *, /, <, >$, log, power, etc., qui sont en réalité des opérations floues c'est-à-dire sur des intervalles flous.

La classe *fuzzy-number* est celle de la figure 6.1. Un des points forts des langages ori-

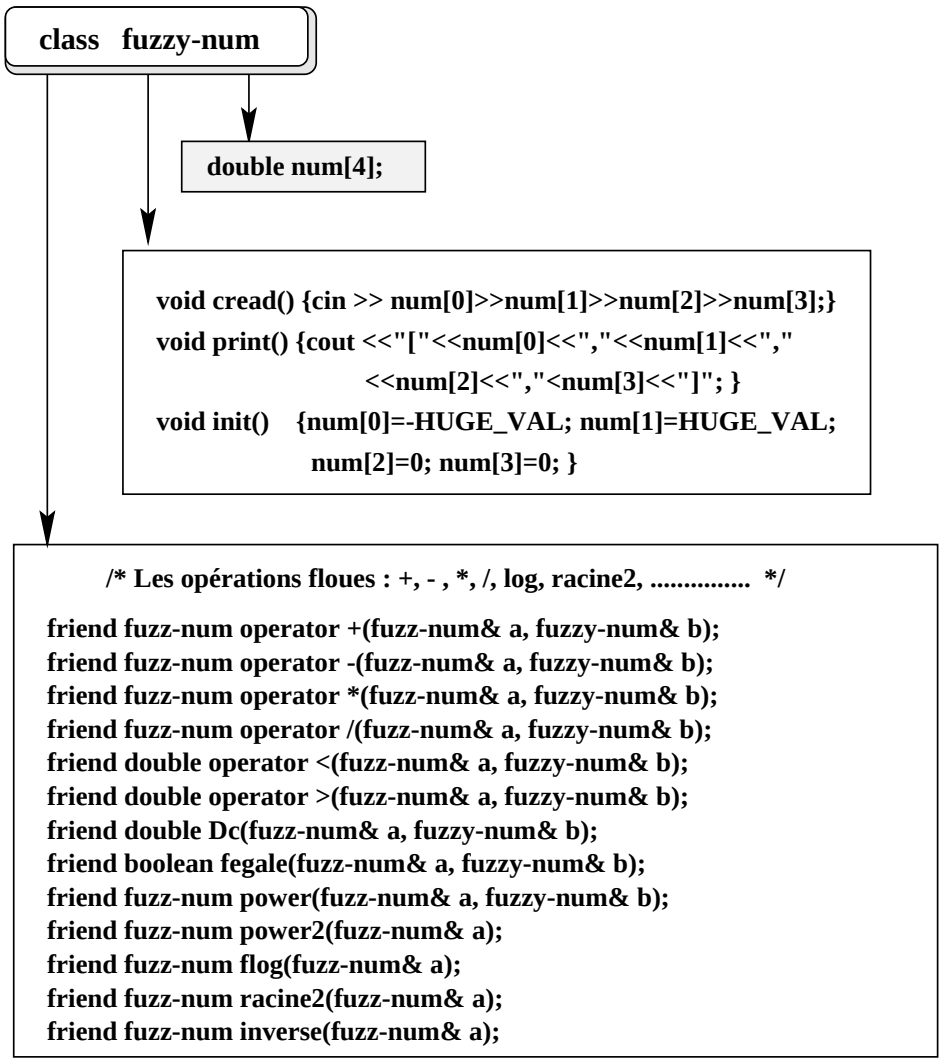


Figure 6.1: La classe *fuzzy-number*

enté-objet est la possibilité de définir des opérateurs par "operator" [Poh89], comme +, -, *, etc. dans la classe fuzzy-number. Ceci permet au compilateur de distinguer entre a + b où a et b sont des nombres et a + b où a et b sont des intervalles flous, les deux écritures sont bonnes et c'est au compilateur de choisir l'opérateur convenable. Ainsi, la programmation et le développement du système deviennent des tâches simples.

Toutes les opérations possibles arithmétiques ou logiques ont été définies et programmées sur les intervalles flous. Un fichier indépendant, nommé *operators.cc*, contient toutes ces opérations.

Dans la conception et la programmation de FLAMES, nous avons pris en compte la modularité pour lui donner une souplesse acceptable et faciliter son amélioration ultérieure. L'unité des modèles est programmé dans un module indépendant, ce qui permet à FLAMES d'être appliqué sur un autre problème que les CA. Pour cela, on a besoin de définir les nouveaux modèles. Les modèles sont aussi définis comme des classes, par exemple, il y a les classes suivantes : *class resistor*, *class transistor* et *class capacitor*, et il est important de noter que cette unité peut être facilement mise à jour.

L'unité ATMS-flou a été complètement programmée avec toutes ses opérations sur les hypothèses aussi bien que sur la propagation des valeurs. La figure 6.2 montre un exemple d'une représentation interne d'un circuit dans FLAMES :
- les composants sont des classes : R1 et R2 sont des instances de la *classe resistor*, T est une instance de la *classe npn-Transistor*, etc.
- la connexion est réalisée par des nœuds { N0, N1, N2, N3} qui sont de deux types : des nœuds simples comme N1, N2, et N3, et des nœuds multiples comme N0.

Pour le moment, un circuit à tester doit être spécifié, dans un fichier nommé *data*, par ses composants et ses nœuds, par exemple la résistance R1 de la figure 6.2 est définie dans *data* par (son nom, sa valeur, ses nœuds), comme suit :

Resistor R1
99500 100500 500 500
0 1

Le circuit, comme cette figure le montre, devient une netlist composée des nœuds

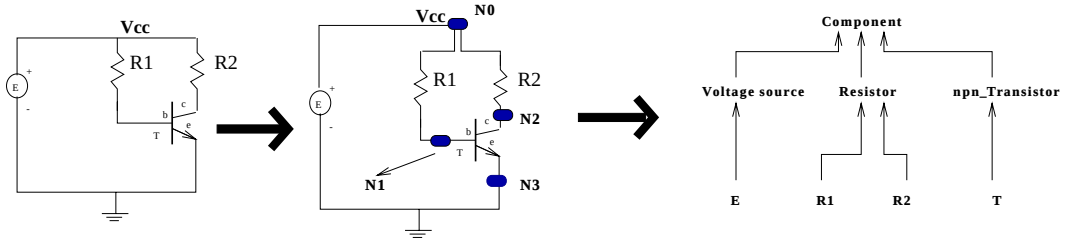


Figure 6.2: Représentation Interne d'un circuit dans FLAMES

et des composants. Chaque fois qu'il y a une nouvelle valeur entrée, la netlist est

stimulée et la propagation a lieu, de nouvelles valeurs et de nouveaux environnements sont calculés. Une fois la propagation finie, chaque quantité est attachée par ses environnements, ses valeurs possibles et les degrés de Dc correspondants. Le processus du diagnostic démarre et ainsi de suite.

6.2 Expérimentation et diagnostic

Tout au début, nous avons utilisé le filtre de la figure 6.3, qu'on va plus tard utiliser pour la simulation aussi, pour essayer l'effet du degré Dc sur la détection de fautes [MMNB95]. Les concepteurs ont donné les tolérances suivantes :
Pour les capacités, [CN+0.1%, CN+0.5%, 0, 0.5%], où CN est la valeur nominale de la capacité, et [RN-0.5%, RN+0.5%, 0.5%, 0.5%] pour les résistances, où RN est la valeur nominale.
Les deux cas suivants ont été testés :

- pour R3 = 99.200 kΩ, qui est fautive à 0.4 degré (par rapport à la valeur nominale qui égale à 100 kΩ), $Dc(Vn, Vm)_{output} = +0.45$, qui indique l'existence d'une faute, mais le(s) composant(s) fautif a (ont) une valeur proche de la valeur nominale.

- pour C2 =400 pF, qui est fautive (la nominale est 200 pF), $Dc(Vn, Vm)_{output} =$ -0, indique l'existence d'une faute. La signe '-' indique que la valeur mesurée est située à gauche de la valeur nominale.

Mais, afin de bien expliquer le processus de diagnostic, on va reprendre l'exemple simple de la figure 6.2, qui a été considéré dans la section 2.5.
On considère 12V comme valeur d'entrée au nœud N0 et 0V au nœud N3 qui sont représentés dans l'ATMS-flou par leurs valeurs et leurs environnements correspondants (considérons les tolérances comme des nombres flous) :

$[12,12,0.05,0.05]\{(Node0)_1\}_1$;

$[0, 0, 0, 0\ ]\{(Node3)_1\}_1$;

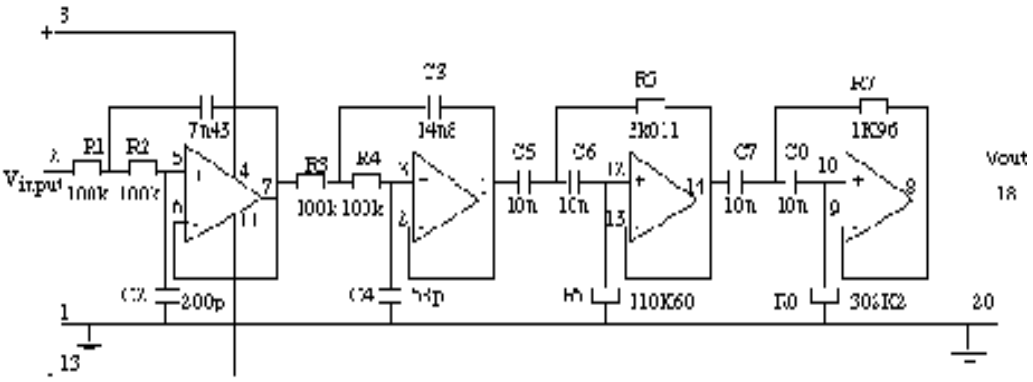


Figure 6.3: Broad Band filter 0.7/1.7

La propagation dans le circuit donne pour Nœud1 et Nœud2, par exemple :

Node1 : [0.7,0.7,0.05,0.05] $\{(Node3)_1, (T1)_1\}_1$;

Node2 : [0.59,0.8,0.9,0.9]$\{(Node0)_1, (Node3)_1, (T1)_1, (R1)_1, (R2)_1\}_1$;

et ces sont les valeurs courantes calculées pour chaque nœud.

Maintenant, quand on mesure la valeur du Nœud2 comme étant 0.4V, la propagation donnera :

Node2 : [0.4,0.4,0.04,0.04] $\{(Node2)_1\}_{-0.60}$;

Node1 : [0.28,0.52,0.9,0.85] $\{(Node0)_1, (Node2)_1, (R2)_1, (T1)_1, (R1)_1\}_{-0.62}$;

Les environnements contiennent les composants et les nœuds de propagation, ils indiquent les justifications des valeurs calculées et les degrés de cohérence entre les valeurs propagées et les valeurs nominales.

Un Dc est attaché à chaque composant (comme 1 dans $(Node)_1$) pour indiquer à quel degré son modèle est valable ; par exemple, comme le modèle de transistor contient des inégalités (comme $Vc \leq Vb$) et les valeurs sont des intervalles flous, donc on a besoin de calculer à quel degré une telle condition ou contrainte est vérifiée et ce degré est celui attaché au composant.

Le Dc global est celui qui contrôle tous les environnements comme -0.62 qui est calculé pour le Node1 dans la seconde étape.

Noter que pour calculer ce degré on applique la technique utilisée dans la section 5.2.2.

Le processus de diagnostic considère tous les résultats de tous les nœuds, il trie les ensembles de candidats suivant leurs degrés Dc, et élimine quelques candidats avec des Dc égaux à 1 quand il n'y a pas de doute quant à leur correction. Finalement, l'unité de recherche du meilleur nœud à tester est lancée pour aider et améliorer le diagnostic. Ici, le Nœud1 est proposé et quand il est mesuré, le processus de diagnostic sera répété et finalement R1 sera déclaré comme le suspect principal. Ce processus est soutenu par des règles qualitatives et les autres unités de FLAMES comme nous l'avons vu auparavant.

Le circuit suivant (figure 6.4) comprend les premiers éléments qui ont été modélisés dans FLAMES, c'est un de premiers exemples traités et qui a bien encouragé le travail : Le tableau de résultats (figure 6.5) montre comment le degré Dc joue un rôle important

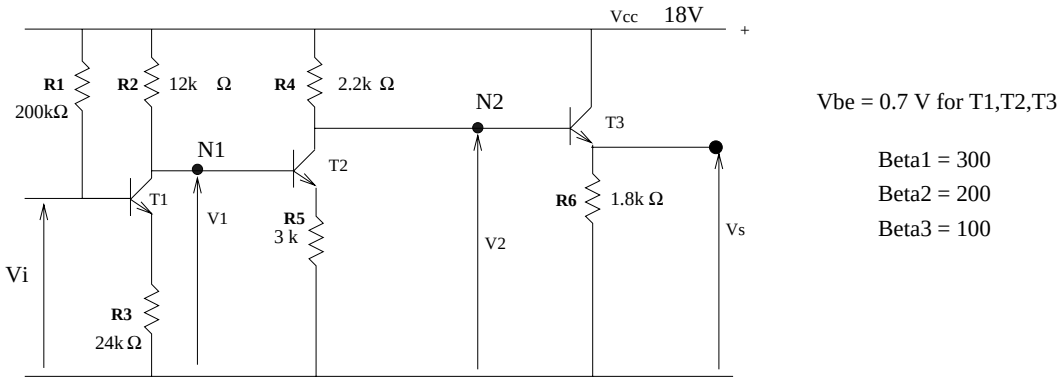


Figure 6.4: un circuit d'amplification à 3 étages

dans l'élimination et le choix de candidats, et ainsi permet de réduire leur nombre et de la possibilité d'explosion. Pour Beta2 = 194, qui est légèrement petite, le degré Dc trouvé est égale à 0.04. Cette valeur correspond à la légèreté de la faute. Une telle faute peut être négligée ou bien recherché suivant le choix de l'expert. Dans le cas du circuit ouvert en R3, Dc prend un valeur égale à 1 ou -1 suivant les valeurs mesurées. Le signe de Dc peut être formulée comme le montre le tableau de la figure 6.4. Il permet aussi de supprimer des candidats si les propagations donnent, par exemple, des signes différents pour des chemins différents. Dans notre cas, on peut conclure qu'un court circuit existe en R3 ou bien que le circuit est ouvert en R2.

Défaut	**Diagnostic**	**Commentaires**
Court circuit en R2	{R1,R2,R3,T1} ==> {R1} {R2} {R3} 1 1 1 1	La propagation de valeurs de V1 et V2 réduit les candidats à R2
R2 est légèrement "HIGH", R2 = 12.18k	{R1,R2,R3,T1} ==> {R2} {T1} **0.11** 0.11 0.11	Grâce à Dc, car Dc(Vsm,Vsn)=0.89, Dc(V2m,V2n)=0.89 et Dc(V1m,V1n)=0.89
Beta2 est légèrement "LOW", Beta2 = 194	{R2,R4,R5,T2} ==> {T2} {R4} 0.04 0.04 0.04	Dc(Vsm,Vsn)=0.96, Dc(V2m,V2n)=0.96, et Dc(V1m,V1n)= 1
Circuit ouvert en R3	{R1,R2,R3,T1} ==> {R2} {R3} 1 1	Grâce au signe de Dc ; Dc(Vsm,Vsn)=1,Dc(V2m,V2n)=1 et Dc(V1m,V1n)=-1 => R2 est très "LOW" ou R3 est très "HIGH"
Court circuit en N1	{T2} {R4} 1 1	Grâce au modèle du transistor, la mesure de V1 est décisive

Figure 6.5: Dc degré pour le diagnostic par FLAMES

6.3 Simulation par FLAMES

Le point fort de FLAMES est sa capacité à propager différents types de tolérances et à simuler des circuits. Le filtre de la figure 6.3 a été utilisé pour montrer comment FLAMES peut être utilisé pour accomplir une simulation analogique [MMBN96a], quand des valeurs ordinaires, des intervalles ordinaires, des nombres flous, ou des intervalles flous, sont utilisés.

Le premier tableau (figure 6.6) montre les résultats de comparaisons entre CLP(R) [NMSZB93], SPICE, et FLAMES où chaque colonne donne la partie réelle et la partie imaginaire des valeurs de Vout (les colonnes de FLAMES donnent des intervalles au lieu de nombres). Ces résultats sont obtenus par la propagation des nombres ordinaires représentés dans FLAMES par [n,n,0,0] [MMBN96b].

On remarque que FLAMES, dans ce cas, fait aussi bien que les autres simulateurs qui sont construits pour ce genre de calcul. Ainsi, à part le but principal de la construction de FLAMES, qui est le diagnostic, il peut aussi être utilisé comme un simulateur flou. Au delà de la comparaison de FLAMES avec d'autres simulateurs, il est intéressant d'examiner comment il peut prendre en considération les tolérances sur les valeurs de paramètres et sur leurs degrés d'acceptation.

f [Hz]	*CLP(R)* Vout [V]	*SPICE* Vout [V]	*FLAMES* Vout [V]
300	(3.5090,-1.0885) 10 $^{-2}$	(3.509335,-1.088039) 10^{-2}	[3.50898,3.50898,0,0] 10 $^{-2}$ [-1.08853,-1.08853,0,0] 10 $^{-2}$
500	(5.8971,-4.7044) 10 $^{-1}$	(5.899580,-4.702675) 10^{-1}	[5.89707,5.89707,0,0] 10 $^{-1}$ [-4.70442,-4.70442,0,0] 10 $^{-1}$
700	(-15.4220,0.7680)	(-15.43548,0.7751102)	[-15.422,-15.422,0,0] [0.768012,0.768012,0,0]
1000	(13.5416,5.3335)	(13.54514,5.334026)	[13.5416,13.5416,0,0] [5.33352,5.33352,0,0]
1200	(11.9127,-9.4294)	(11.91672,-9.430864)	[11.9127,11.9127,0,0] [-9.42939,-9.42939,0,0]
1400	(-3.9358,-14.5351)	(-3.935100,-14.54053)	[-3.93582,-3.93582,0,0] [-14.5351,-14.5351,0,0]
1500	(-9.7954,-10.3015)	(-9.796561,-10.30809)	[-9.79535,-9.79535,0,0] [-10.3015,-10.3015,0,0]
1600	(-14.3621,-4.1904)	(-14.37026,-4.199249)	[-14.3621,-14.3621,0,0] [-4.19036,-4.19036,0,0]
1700	(-13.1528,8.2180)	(-13.17026,8.228991)	[-13.1528,-13.1528,0,0] [8.21805,8.21805,0,0]
1800	(-0.8592,9.0945)	(-0.8528107,9.103439)	[-0.859229,-0.859229,0,0] [9.09449,9.09449,0,0]
2000	(1.4097,2.0849)	(1.411279,2.084567)	[1.40967,1.40967,0,0] [2.08488,2.08488,0,0]
2600	(0.26516,0.15341)	(0.265221,0.1533463)	[0.265156,0.265156,0,0] [0.153411,0.153411,0,0]

Figure 6.6: Une comparaison entre CLP(R), SPICE et FLAMES

Le second tableau (figure 6.7) montre les résultats donnés par FLAMES quand différents types de valeurs sont considérées (les tolérances prises sont de 0.5% pour les résistances et de 1% pour les capacités). Dans la première colonne, des nombres flous ont été propagés (valeurs comme [100,100,0.5,0.5] pour R1), la deuxième colonne montre les résultats de la propagation des intervalles ordinaires (valeurs comme [99.5,100.5,0,0] pour R1), et dans la troisième colonne des intervalles flous sont utilisés (comme [99.75,100.25,0.25,0.25] pour R1) qui correspondent à [R1-0.25%, R1+0.25%, 0.25%, 0.25%]. Noter que le concepteur peut utiliser quelque chose comme [C-0.3%, C+0.2%, 0.2%, 0.3%] par exemple quand c'est plus convenable) et ceci est naturellement possible avec les intervalles flous.

Quand on regarde le tableau de la figure 6.7 (spécialement la ligne qui correspond à la fréquence de 700Hz), on remarque une grande différence entre les intervalles ordinaires et les intervalles flous. Cela est dû au calcul d'erreurs de virgule flottante qui est réduit en faveur du cas des intervalles flous. La figure 6.8 montre les amplitudes

f [Hz]	***Nombres flous*** Vout [V]	***Intervalles class.*** Vout [V]	***Intervalles flous*** Vout [V]
300	**[0.035,0.035,0.007,0.009] [-0.01,-0.01,0.003,0.002]**	**[0.02,0.10,0,0] [-0.03,-0.006,0,0]**	**[0.025,0.06,0.003,0.007] [-0.018,-0.008,0.002,0.008]**
500	**[0.59,0.59,0.21,0.29] [-0.47,-0.47,0.19,0.14]**	**[0.09,3.72,0,0] [-2.67,-0.19,0,0]**	**[0.29,1.46,0.07,0.33] [-1.09,-0.31,0.21,0.05]**
700	**[-15.42,-15.42,13.64,7.29] [0.768,0.768,7.93,9.12]**	**[-99.98,-1.87,0,0] [-24.53,75.59,0,0]**	**[-26.28,-4.91,15.03,1.22] [-8.15,24.34,6.18,10.81]**
1000	**[13.5416,13.5416,0,0] [5.33,5.33,3.72,5.36]**	**[4.02,32.93,0,0] [-4.19,13.75,0,0]**	**[7.35,21.83,1.64,4.95] [0.98,2.09,2.09,3.32]**
1200	**[11.91,11.91,4.60,6.74] [-9.42939,-9.42939,0,0]**	**[4.72,22.25,0,0] [-21.26,-2.39,0,0]**	**[7.38,16.46,1.61,4.06] [-14.92,-5.18,3.75,2.05]**
1400	**[-3.94,-3.94,6.65,5.21] [-14.54,-14.54,13.63,8.24]**	**[-13.09,1.49,0,0] [-25.74,-3.71,0,0]**	**[-8.14,-0.88,3.76,2.28] [-19.57,-9.91,4.64,2.16]**
1500	**[-9.80,-9.80,9.49,6.75] [-10.30,-10.30,8.90,5.53]**	**[-22.05,-2.25,0,0] [-21.19,-3.71,0,0]**	**[-15.25,-5.43,5.48,2.83] [-15.00,-6.30,5.25,2.14]**
1600	**[-14.36,-14.36,13.63,8.24] [-4.19,-4.19,9.31,7.43]**	**[-30.93,-5.02,0,0] [-14.45,4.06,0,0]**	**[-21.34,-8.58,5.68,3.04] [-8.40,-0.42,5.60,4.08]**
1700	**[-13.15,-13.15,9.74,7.08] [8.22,8.22,7.83,8.75]**	**[-24.73,-5.11,0,0] [0.15,18.78,0,0]**	**[-18.18,-8.58,8.19,3.42] [3.83,13.05,3.58,4.87]**
1800	**[-0.86,-0.86,4.73,5.79] [9.09,9.09,4.70,7.70]**	**[-5.85,5.52,0,0] [3.82,18.06,0,0]**	**[-3.07,1.91,2.66,3.33] [6.04,12.99,1.96,4.47]**
2000	**[1.41,1.41,1.01,1.47] [2.08,2.08,0.77,1.24]**	**[0.35,3.10,0,0] [1.17,3.50,0,0]**	**[0.81,2.16,0.44,0.84] [1.55,2.71,0.32,0.71]**
2600	**[0.26,0.26,0.08,0.11] [0.15,0.15,0.04,0.05]**	**[0.17,0.38,0,0] [0.11,0.21,0,0]**	**[0.21,0.32,0.04,0.06] [0.13,0.18,0.02,0.03]**

Figure 6.7: Les capacités de FLAMES avec de différents types de tolérances

des sorties simulées dans le cas des intervalles ordinaires (partie a), et celui des intervalles flous (partie b). Une concentration peut être remarquée dans le second cas, qui une fois de plus justifie l'approche adoptée et développée dans cette thèse.
Finalement, en ce qui concerne le temps CPU, il a été évalué dans le dernier exemple. Nous avons trouvé qu'il est de l'ordre de 20 ms, ce qui indique qu'une explosion du temps n'a pas eu lieu. D'ailleurs, un examen des opérations floues elles-mêmes pourrait donner une idée concernant la complexité de ce type de calcul dans FLAMES.

6.4 Diagnostic des circuits digitaux

Dans le chapitre 4 nous avons présenté les stratégies de recherche du meilleur nœud à tester et leur application aux circuits digitaux. Cette section donne quelques résultats pratiques obtenus dans les cas des circuits combinatoires et des circuits séquentiels.

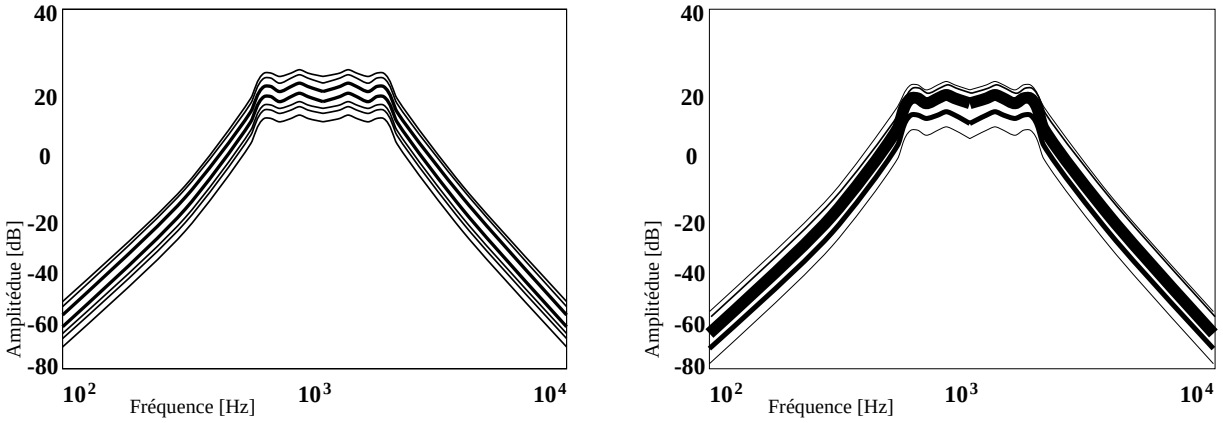


Figure 6.8: Les amplitudes simulés dans les deux cas : ordinaires et flous

Pour illustrer et en même temps évaluer l'efficacité de la démarche que nous avons développée dans ce chapitre, nous allons l'appliquer sur deux exemples.
Nous avons testé notre approche sur les ISCAS'85 pour les systèmes combinatoires et les ISCAS'89 pour les séquentiels. Pour chaque circuit, nous avons injecté une faute, ensuite nous avons démarré le processus de diagnostic à partir des vecteurs de test qui mettent en évidence cette faute. Pour évaluer les performances de l'outil, nous avons défini le paramètre *efficacité* que nous calculons de la manière suivante :

$$Eff = 100 \times \frac{N-(k-1)}{N}$$

Circuit	# total de blocs	Bloc Fautif	Rang du Bloc Fautif	Efficacité en (%)
c17	6	NAND23	1	100
c432nr	171	NAND71	34	80.7
c880	383	NOR73	11	97.3
c2670nr	961	AND243	21	97.9

Table 6.1: Résultats pour quelques ISCAS'85

Avec N le nombre total de composants du système et k le rang du site de la faute (sortie du bloc fautif) dans la liste des points de test proposés. Le tableau 6.1 présente les résultats obtenus sur des modules combinatoires. Ces résultats sont dans l'ensemble assez satisfaisants. Le tableau 6.2 présente les résultats obtenus sur des modules

Circuit	**# total de blocs**	**# de Bascules**	**Bloc Fautif**	**Prof. de Séquentialité**	**Long. du Vecteur**	**Rang du Bloc Fautif**	**Efficacité en (%)**
s298	133	14	NOT22	1	2	36	73.6
s344	175	15	DFF11	2	3	29	84
s510	217	6	OR99	2	27	100	54.3
s1196	547	18	AND159	1	2	158	73.6

Table 6.2: Résultats pour quelques ISCAS'89

séquentiels. Bien que moins bons que ceux obtenus pour les modules combinatoires, ces résultats sont dans l'ensemble assez encourageants. On notera cependant le cas du s510 pour lequel l'efficacité se situe légèrement au dessus de 50%. Ce résultat s'explique en partie par la qualité des vecteurs de test disponibles. Les séquences sont longues et par conséquent la profondeur de séquentialité ne joue pas son rôle de discrimination des candidats.

De toutes manières, en ce qui concerne les circuits séquentiels, une amélioration demeure possible. Elle devra surtout porter sur le traitement des boucles séquentielles.

Chapitre 7

Conclusion et Perspectives

Chapitre 7

Conclusion et Perspectives

Dans ce travail, nous avons proposé, développé et implémenté une nouvelle approche, à base de logique floue, pour le test et le diagnostic des circuits analogiques et mixtes. Un système, nommé FLAMES, résumant tout le travail fait a été programmé et essayé avec succès sur plusieurs types de circuits.

La logique floue est connue pour sa capacité à traiter l'imprécision et l'incertitude des systèmes dynamiques et à représenter l'expertise humaine. En plus, les intervalles flous sont, comme cette thèse a pris le soin de le montrer, les plus puissants pour la détection des fautes et représentent d'une manière la plus générale les autres concepts. Les nombres ordinaires, les nombres flous et les intervalles ordinaires peuvent être uniformement représentés et n'oublions pas la possibilité de représenter les termes qualitatifs pour un traitement semi-qualitatif plus puissant d'une part, d'autre part les tolérances de différents types peuvent être représentées ainsi qu'avec les différentes types de distribution comme dans le cas des capacités.

La représentation des tolérances via des intervalles flous rend le calcul des tolérances de toutes les quantités du circuit sous test beaucoup plus simples que les autres méthodes comme le calcul de la sensibilité de Hamida et al. [HK93], car une seule propagation suffit pour calculer les tolérances. Ainsi, l'idée d'un simulateur flou devient très interessante.

En outre, le raisonnement qualitatif a été utilisé pour la conception des circuits analogiques [MT95] [TM95], dans cette direction aussi nous trouvons un domaine où la logique floue peut jouer un rôle important.
Nous avons conçu et implémenté le système FLAMES avec l'idée de bénéficier de

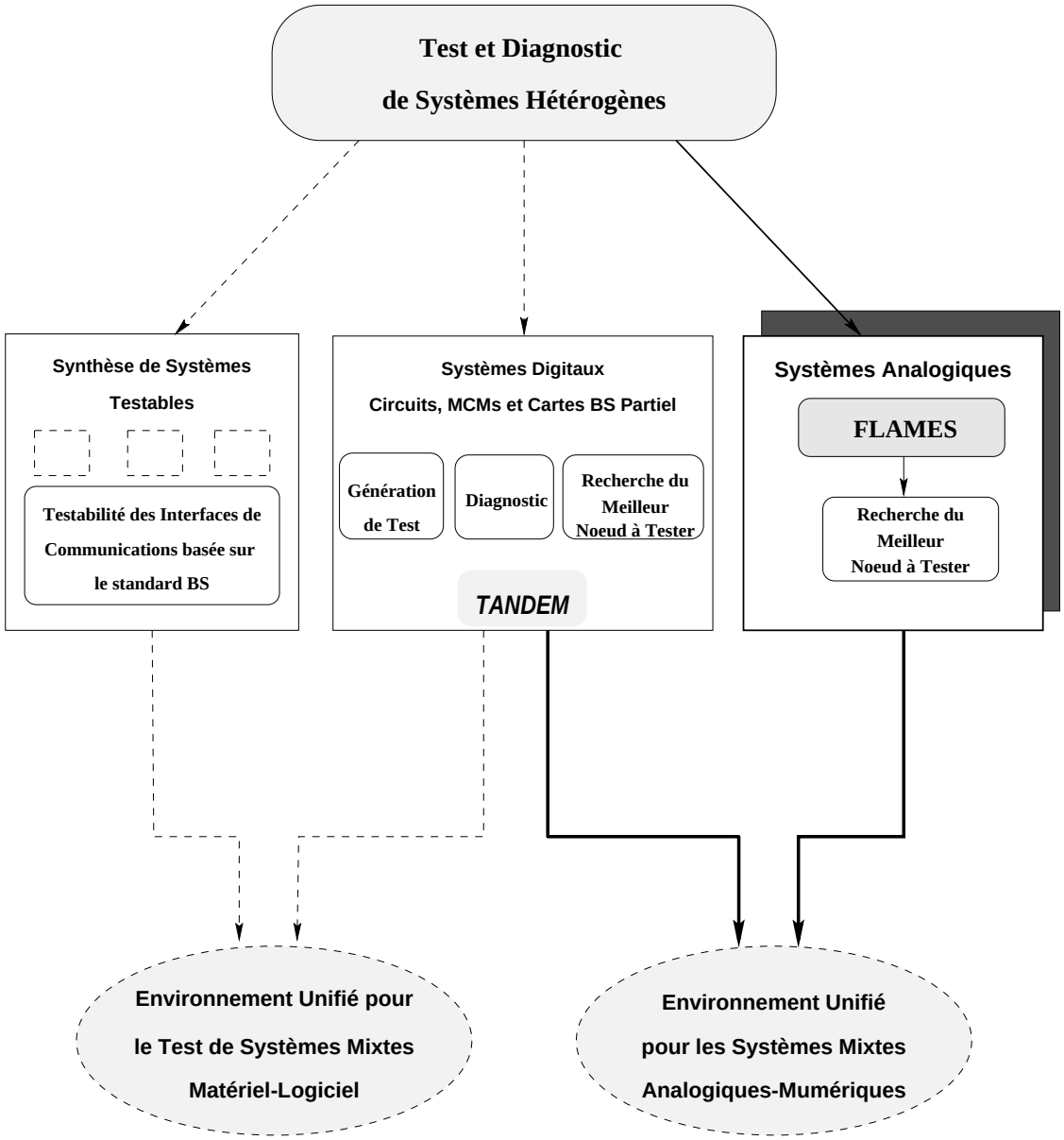


Figure 7.1: Réalisation et perspectives

toutes les approches utiles pour le diagnostic des CA. Bien qu'il y ait encore des unités qui n'ont pas été implémentées (mais initiées) comme l'unité d'apprentissage de l'expérience et l'unité de la base de connaissances, les autres ont été implémentées

et testées.
Pour l'apprentissage, les réseaux de neurones nous paraissent une approche convenable surtout qu'elle peut être utilisé dans d'autres parties du système comme pour la classification de fautes. En effet, les réseaux de neurones ont prouvé leur importance dans plusieurs domaines dont les applications sont nombreuses [Kar96]. C'est une approche à étudier soigneusement ; sa contribution dans FLAMES pourrait être utile. Ce système peut être très utile pour son but initial, qui est le test et le diagnostic des circuits analogiques et mixtes, et pour tout autre système dynamique car la seule unité dépendante du système est l'unité de modèles. Nous encourageons, sans hésitation, à continuer le travail pour le développement de ce système surtout que son développement pour jouer le rôle d'un simulateur s'annonce bien.

FLAMES, développé au sein de l'équipe de diagnostic des systèmes complexes du laboratoire TIMA, participe au cadre général de la recherche dans le domaine du test et du diagnostic des systèmes hétérogènes. La figure 7.1 montre les trois principales tâches couvertes, et la façon dont les résultats de ce travail y participent. En ce qui concerne les systèmes mixtes analogiques-digitaux, le but envisagé, le développement d'un environnement unifié pour le test et le diagnostic de ces systèmes, a été réalisé puisqu'on a vu qu'il y a une certaine compatibilité entre les deux systèmes : FLAMES et TANDEM [Tou96] utilisent une méthodologie à base de logique floue, ils utilisent aussi la même méthode pour la recherche du meilleur nœud à tester et le plus important est que leurs résultats de diagnostic sont compatibles. Ainsi, les rassembler pour former une méthodologie générale pour le diagnostic des circuits mixtes parait très possible. D'ailleurs, nous croyons qu'une telle méthodologie peut être efficace et très convenable pour ce problème.
Mais, malheureusement, nous n'avons pas eu le temps nécessaire pour achever le travail consistant à diagnostiquer les circuits mixtes, bien que les outils nécessaires, FLAMES d'un coté pour la partie analogique et TANDEM [Tou96] pour la partie digitale d'un autre coté, soient disponibles. D'ailleurs, nous croyons qu'une telle méthodologie peut être efficace et convenable pour ce problème.

Bibliographie

Bibliographie

[Aut94] Veda Design Automation. *System Hilo 4, user manual*, 1994.

[BA96] A. Balivada and J. Abraham. Fault modeling and fault simulation of analog and mixed-signal circuits : A tutorial. In *International Mixed-Signal Workshop*, Quebec City, Canada, May 1996.

[BBdK82] J. S. Brown, R. R. Burton, and J. de Kleer. Pedagogical, natural language and knwledge engineering techniques in SOPHIE I, II, and III. In D. Sleeman and J.S. Brown, editors, *Intelligent Tutoring Systems*, chapter 11, pages 227–282. Academic Press, 1982.

[BBK89] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *IEEE ISCAS*, pages 1929–1934, 1989.

[BD86] P. P. Bonisson and K. S. Decker. Selecting uncertainty calculi and granularity : An experiment in trading-off precision and complexity. In L. N. Kanal and J. F. Lemmer, editors, *Uncertainty in AI*. Elsevier Science Publishers B.V, North-Holland, 1986.

[Ben94] B. Bennets. Progress in design for test : A personal view. *IEEE Design and Test of Computers*, pages 53–59, 1994.

[BF85] F. Brglez and H. Fujiwara. A neural netlist of 10 combinational benchmark circuits and a target translator in fortran. In *IEEE International Test Conference*, pages 785–794, 1985.

[BM91] B. Bouchon-Meunier. How to replace computations by simple rules in the framework of a fuzzy logic. In T. Kohonen and F. Fogelman-Soulié, editors, *COGNITIVA 90*, pages 239–245. Elsevier Science Publishers B.V, North-Holland, 1991.

[BP95] Corinne Bos-Plachez. Apport d'un atms possibiliste pour le diagnostic des circuits analogiques. In *Rencontres Francophones sur la Logique Floue et ses Applications*, pages 111–117. CEPADUES EDITIONS, Paris, France, November 1995.

[BZ87] R. E. Bellman and L. A. Zadeh. Decision-Making in Fuzzy Environment. In R.R. Yager, S. Ovchinnikov, R.M. Tong, and H.T. Nguyen, editors, *Fuzzy Sets and Applications : Selected Papers by Zadeh*, pages 53–79. John Willey and Sons, New York, 1987.

[CB90] P. Charbonnaud and R. Bertin. A technical diagnosis expert system using qualitative physics. In T. Kohonen and F. Fogelman-Soulie', editors, *COGNITIVA '90*, pages 295–303, North-Holland, 1990. Elsevier Science Publishers B.V, 1991.

[CRS92] C. Cayrol, V. Royer, and C. Saurel. Management of Preference in Assumption-based Reasoning. In *4th Int. Conf. on Information Processing of uncertainty in Knowledge-Based Systems*, Palma de Mallorca, Spain, July 1992.

[Dav84] R. Davis. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence*, 24(1):347–410, 1984.

[Dav87] R. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32:281–331, 1987.

[Dav93] R. Davis. Retrospective on 'Diagnostic resoning based on structure and behavior'. *Artificial Intelligence*, 59, 1993.

[DDLT90a] P. Dague, P. Deves, P. Luciani, and P. Taillibert. Analog systems diagnosis. In *9th European Conf. on AI*, Stockholm, August 1990.

[DDLT90b] P. Dague, P. Deves, P. Luciani, and P. Taillibert. Diagnostic de systèmes analogiques. In *Journées internationales sur les systèmes experts et leur applications*, Avignon, France, 1990.

[DH88] R. Davis and W. Hamsher. Model-based Reasoning : Troubleshooting. In H.E. Shorbe and the American Association of AI, editors, *Exploring Artificial Intelligence*, chapter 8. Morgan Kaufmann, 1988.

[DJD+91] P. Dague, O. Jehl, P. Deves, P. Luciani, and P. Taillibert. When occillators stop oscillating. In *Qualitative Reasoning*, pages 1109–1115. IJCAI-91, 1991.

[DJT90] P. Dague, O. Jhel, and P. Taillibert. An Interval Propagation and Conflict Recognition Engine for Diagnosing Continuous Dynamic Systems. *Lecture Notes in AI*, 462, September 1990.

[DLP90] D. Dubois, J. Lang, and H. Prade. Gestion d'hypothèse en logique possibiliste : un exemple d'application au diagnostic. In *10th Conf. on Expert Systems and their applications*, Avignon, France, 1990.

[Doy79] J. Doyle. A truth maintenace system. *Artificial Intelligence*, 12:231–272, 1979.

[DP87] D. Dubois and H. Prade. *Théorie des Possibilités*. Masson, Paris, France, 2 edition, 1987.

[DRD87] P. Dague, O. Raiman, and P. Deves. Troubleshooting : When modeling is the trouble. In *Proceed. AAAI-87*, pages 600–605, Seattle, 1987.

[DRDM87] P. Dague, O. Raiman, P. Deves, and J-P Marx. DEDALE : An expert system for troubleshooting analogue circuits. In *IEEE International Test Conference*, pages 586–594, 1987.

[DTF91] P. Devès, P. Taillibert, and C. Fischer. Diagnostic à base de modèles : Une alternative aux systèmes experts. In *Int. Workshop on Expert Systems and their Applications*, pages 53–63, Avignon, France, 1991.

[Euz90] J. Euzenat. *Un système de maintenance de la vérité à propagation de contextes*. PhD thesis, Joseph Fourier - Grenoble 1, Grenoble, France, 1990.

[FK93] M. Fares and B. Kaminska. A Fuzzy Decision-making for Test Space Exploration. In *European Test Conference*, pages 37–46, Rotterdam, The Netherlands, April 1993.

[FK94] M. Fares and B. Kaminska. Exploring test space with fuzzy decision making. *IEEE Design and Test of Computers*, pages 17–27, 1994.

[For84] K. D. Forbus. Qualitative Process Theory. *Artificial Intelligence*, 24, 1984.

[HBC+91] J. P. Haton, N. Bouzid, F. Charpillet, M. C. Haton, B. Laasri, H. Laasri, P. Marquis, T. Mondot, and A. Napoli. *Le raisonnement en Intelligence Artificielle*. Interedition, France, 1991.

[HK93] N. B. Hamida and B. Kaminska. Multiple fault for analog circuit testing by sensitivity analysis. In *Analog Integrated Circuits and Signal Processing 4*, pages 231–243, Boston, 1993. Kluwer Academic Publishers.

[Kar96] Stamatios V. Kartalopoulos. *Understanding Neural Networks and Fuzzy Logic*. IEEE Press, New York, 1996.

[KB84] J. De Kleer and J. S. Brown. A Qualitative Physics Based on Confluences. *Artificial Intelligence*, 24, 1984.

[Kle86] J. De Kleer. An Assumption-based TMS. *Artificial Intelligence*, 28:127–162, 1986.

[Kle91] J. De Kleer. Focusing on probable diagnosis. In *AAAI-91*, pages 842–848, Anaheim, 1991.

[Kos89] Y. Koseki. Experience learning in Model-based diagnostic systems. In *IJCAI-89*, pages 1356–1361, 1989.

[Kos91] B. Kosko. *Neural Networks and fuzzy Systems*. Prentice-Hall, 1991.

[Kui84] B. Kuipers. Commonsense reasoning about causality : Deriving behavior from structure. *Artificial Intelligence*, 24, 1984.

[Kui86] B. Kuipers. Qualitative Simulation. *Artificial Intelligence*, 29:289–338, 1986.

[Kui93] B. Kuipers. Reasoning with Qualitative Models. *Artificial Intelligence*, 59, 1993.

[KW87] J. De Kleer and C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32:97–129, 1987.

[KW89] J. De Kleer and B. Williams. Diagnosis with behavioral modes. In *the Eleventh IJCAI*, Detroit, August 1989.

[Lei92] R. Leitch. Recent Progress in the Development of Qualitative Reasoning. In *IMACS Congress on Math. of the Analysis and Design of process Control*, North-Holland, 1992.

[Liu87] R-W Liu. A circuit theoretic approach to analog fault diagnosis. In R-W Liu, editor, *Testing and Diagnosis of Analog Circuits and Systems*, chapter 1. Van Nostrand Reinhold, New York, 1987.

[Mac84] E. J. MacClustry. Verification testing-a pseudo-exhaustive test technique. *IEEE Transactions on Computers*, 33:541–546, June 1984.

[McA91] D. McAllester. Truth maintenance. In *AAAI-91*, pages 1109–1116, 1991.

[MM96] F. Mohamed and M. Marzouki. Test and diagnosis of analog devices : When fuzziness can lead to accuracy. *Journal of Electronic Testing : Theory and Applications*, 9:203–216, 1996. Special Issue on Mixed-Signal Testing, Kluwer Academic Publishers.

[MMBN96a] F. Mohamed, M. Marzouki, A. Biasizzo, and F. Novak. Analog Circuit Simulation and Troubleshooting with FLAMES. In *14th IEEE VLSI Test Symposium*, pages 495–502, New Jersey, 1996.

[MMBN96b] F. Mohamed, M. Marzouki, A. Biassizo, and F. Novak. Testing and Diagnosis of Analog Devices by FLAMES. In *KoREMA'96 - 41st Annual Conference*, Optijia, Croatia, September 1996.

[MMNB95] F. Mohamed, M. Marzouki, F. Novak, and A. Biasizzo. A Fuzzy Logic Approach for Analog Circuit Diagnosis. In *Int. Mixed Signal Testing Workshop*, pages 101–106, Grenoble, France, June 1995.

[MMT96] F. Mohamed, M. Marzouki, and M. H. Touati. FLAMES : A Fuzzy Logic ATMS and Model-based Expert System for Analog Diagnosis. In *European Design and Test Conference*, pages 159–163, Paris, France, March 1996.

[MMT97] F. Mohamed, M. Marzouki, and M.H. Touati. A cost-effective approach for analog, digital and mixed-signal test and diagnosis. In *IEEE European Test Workshop*, Italie, 1997. To appear.

[Moh97] Firas Mohamed. Test et diagnostic des circuits analogiques et mixtes. In *Colloque CAO de circuits intégrés et systèmes*, page 88, Grenoble, France, January 1997.

[Moo66] R. Moor. *Interval Analysis Series in Automatic Computation.* Prentice-Hall, 1966.

[MT95] C. A. Makris and C. Toumazou. Analog ic design automation: Part ii-automated circuit correction by qualitative reasoning. *IEEE Transactions on Computer-Aided Design*, 14(2):239–254, February 1995.

[MTM96] F. Mohamed, M.H. Touati, and M. Marzouki. Multi-level fuzzy decesion-making for DFT, test and diagnosis of analog circuits. In *2nd IEEE Mixed-Signal Testing Workshop*, Quebec, Canada, May 1996.

[MW89a] A. McKeon and A. Wakeling. Fault diagnosis in analog circuits using ai techniques. In *IEEE International Test Conference*, pages 118–123, 1989.

[MW89b] A. McKeon and A. Wakeling. Fault Diagnosis in Analogue Circuits Using AI Techniques. In *IEEE International Test Conference*, pages 118–123, 1989.

[MW91] A. McKeon and A. Wakeling. The automatic diagnosis of faults in analogue and mixed-signal circuits. In *European Design Automation Conference*, pages 89–93, 1991.

[NCA93] N. Nagi, A. Chatterjee, and J. Abraham. DRAFTS : Discretized Analog Circuit Fault Simulator. In *30th ACM/IEEE Design Automation Conference*, 1993.

[NMSZB93] F. Novak, I. Mozetic, M. Santo-Zarnik, and A. Biasizzo. Enhancing Design-for-Test for Active Analog Filters by Using CLP(R). *Journal of Electronic Testing : Theory and Applications*, 4:315–329, 1993.

[NR75] C. V. Negoit and D. A. Ralsec. *Applications of Fuzzy Sets to Systems Analysis*. Birkhuser Verlag, Stuttgart, 1975.

[PDS87] F. Pipitone, K. Dejong, and W. Spears. An AI Approach to Analog System Diagnosis. In R-W Liu, editor, *Testing and Diagnosis of Analog Circuits and Systems*, chapter 7. Van Nostrand Reihold, New York, 1987.

[Pip86] F. Pipitone. The FIS electronics troubleshooting system. *IEEE Computer*, pages 69–76, July 1986.

[Poh89] Ira Pohl. *C++ for C Programmers*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, California, 1989.

[Rai86] O. Raiman. Order of magnitude reasoning. In *AAAI-86*, pages 100–104, 1986.

[Rai91] O. Raiman. Order of Magnitude Reasoning. *Artificial Intelligence*, 51:11–38, 1991.

[Saa77] T. L. Saaty. Exploring the interface between Hierarchies, Multiple Objectives and Fuzzy sets. *Fuzzy Sets and Systems*, 1:57–68, 1977.

[SBRM90] D. C. Van Soest, R. R. Bakker, F. Van Raalte, and N. J. I Mars. Improving Effectivness of Model-based Diagnosis. In *International Workshop on Expert Systems and their Applications*, pages 105–114, 1990.

[SD89] P. Struss and O. Dessler. Physical Negation : Integrating Fault Models into the GDE. In *IJCAI-89*, pages 1318–1324, Detroit, 1989.

[She93] Z. Shen. Book Review "Fuzzy Sets and Applications : Selected Papers by Zadeh. *Artificial Intelligence*, 61:351–358, 1993.

[SL92a] Q. Shen and R. Leitch. Application studies of fuzzy qualitative simulation. In *IMACS congress on Mathematics of the Analysis and Design of Process Control*, North-Holland, 1992.

[SL92b] Q. Shen and R. Leitch. Combining qualitative simulation and fuzzy sets. In B. Falting and P. Struss, editors, *Recent Advances in Qualitative Physics*, chapter 6. MIT Press, 1992.

[SL92c] Q. Shen and R. Leitch. Diagnosing Continuous Dynamic Systems Using Qualitative Simulation. In *5th National Conf. on Control*, 1992.

[Som90] M. Soma. A design-for-test methodology for active analog filters. In *IEEE International Test Conference*, pages 183–191, 1990.

[Ste74] Pat H. Sterbenz. *Floating-Point Computation.* in Automatic Computation. Prentice-Hall, New York, 1974.

[Str92] P. Struss. Qualitative modeling of physical systems in ai research. *Artificial Intelligence*, 24, 1992.

[SU92] R. Spina and S. Updhyaya. Fault diagnosis of analog integrated circuits using artificial neural networks as signature nalyzers. In *Fifth Annual IEEE Int. ASIC Conf. and Exhibit*, pages 355–358, Rochester, NY, 1992.

[Sut92] J.C. Sutton. Identification of electronic component faults using neural networks and fuzzy systems. In *IEEE Int. Conf. on Industrial Electronics, Control, Instrumentation, and Automation*, pages 1466–1471, San Diego, CA, 1992.

[SVCC77] A. Sangiovanni-Vincentelli, L. Chen, and O. Chua. An efficient heuristic cluster algorithme for tearing large-scale networks. *IEEE Transactions on Circuits and Systems*, 24(12), 1977.

[TM95] C. Toumazou, , and C. A. Makris. Analog ic design automation: Part i-automated circuit generation: New concepts and methods. *IEEE Transactions on Computer-Aided Design*, 14(2):218–238, February 1995.

[TMM96] M.H. Touati, F. Mohamed, and M. Marzouki. System fault diagnosis based on a fuzzy qualitative approach. In *European Design and Test Conference*, Paris, France, March 1996.

[TMP89] L. Trave-Massuyes and N. Piera. The order of magnitude model as qualitative algebras. In *IJCAI-89, Workshop on model-based reasoning*, Detroit, 1989.

[Tou96] M.H. Touati. *Test et Diagnostic des Cartes et des MCMs Partiellement Boundary SCAN.* PhD thesis, TIMA Lab./INPG, Grenoble, FRANCE, January 1996.

[WM94] Angus WU and Jack Meador. Measurement selection for ic fault diagnosis. *Journal of Electronic Testing : Theory and Applications*, 5:9–18, 1994.

[Zad65] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[Zad75] L. A. Zadeh. Calculus of fuzzy restrictions. In L. A. Zadeh, K-S Fu, K. Tanaka, and M. Shimura, editors, *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, chapter 1. Academic Press, Inc., New York, 1975.

[Zim87] H. J. Zimmermann. *Fuzzy sets, Decision Making, and Expert Systems.* Management Science/Operations Research. Kluwer Academic Publishers, Boston, 1987.

Annexes

Annexe A

L'ATMS

A.1 TMS : Truth Maintenance Systems

En 1978, Jon Doyle a décrit dans un module indépendant de sa thèse un système de maintien de vérité. C'était le début des travaux sur les TMSs [McA91].
Tous les TMSs manipulent des propositions symboliques et des relations entre eux. Les TMS monotones manipulent des propositions symboliques et des contraintes booléennes. Une contrainte booléenne est une formule booléenne construite à partir des symboles et des connecteurs standards $\rightarrow, \Lambda, \neg$. Il existe aussi des TMS non monotones qui permettent des relations heuristiques ou non monotones. Comme la plupart des développements qui ont été faits sur les algorithmes de TMS et en particulier l'algorithme de ATMS de deKleer concernaient les TMS monotones, et que la plupart des applications de TMS utilisent cette approche (simulation qualitative, diagnostic de fautes,etc), nous allons nous concentrer sur ce type de TMS.
Un TMS monotone constitue une facilité générale pour manipuler des contraintes booléennes ou des propositions symboliques. Étant données des propositions sur les automobiles, un ensemble de contraintes sur ses propositions et un ensemble d'observations sur une automobile particulière, un TMS peut être utilisé pour poser des questions sur les conséquences des observations. Il conserve un ensemble de contraintes implicites appelées *contraintes internes* et utilise aussi des interfaces génériques comme *add-constraint*, pour ajouter une

contrainte à l'ensemble interne, etc. Dans un système de diagnostic d'automobiles, par exemple, l'ensemble des contraintes internes constitue des faits vrais sur toutes les automobiles et les ensembles de prémisses constituent des observations sur des automobiles particulières.

A.2 ATMS : Assumption-based Truth Maintenance Systems

J. deKleer [Kle86] propose une autre approche du maintien de vérité, c'est le système ATMS (Assumption-based TMS) qui est plus efficace pour beaucoup de tâches et qui présente plus d'interfaces cohérentes entre le TMS et le résolveur de problèmes. D'ailleurs, l'ATMS évite la restriction usuelle qu'un seul point de l'espace de recherche peut être examiné à la fois. L'objectif essentiel de l'ATMS est d'éviter les imperfections connues des TMS.

Considérons un TMS à base de justifications. Le raisonnement se base sur un résolveur de problèmes qui produit les inférences (appelées des justifications) et un TMS qui les conserve. Le TMS fonctionne comme un cache pour toutes les inférences.
En cas de contradiction, le TMS utilise une procédure de retour en arrière dirigé par les justifications (dependency-directed backtracking) pour identifier et ajouter des justifications afin d'éliminer les contradictions [Kle86].
L'ATMS peut être considéré comme un cache intelligent, il étend l'idée du cache, simplifie le maintien de la vérité et évite la plupart des retours en arrières. Il utilise une méthode de propagation de contraintes.

Différents travaux ont utilisé l'ATMS de deKleer ou des ATMS-like développés à base de cet ATMS, citons :

- l'ATMS possibiliste de Dubois et al. [DLP90], c'est une extension de l'ATMS classique qui incorpore dans ses mécanismes un traitement de l'incertitude basé sur la théorie des possibilités. Dans un tel ATMS, les hypothèses, les faits et les justifications peuvent être incertains.

- un ATMS à propagation de contextes [Euz90], qui permet de raisonner simultanément

sur plusieurs contextes à l'aide d'inférences non monotones.

- l'ATMS de Dague et al. [DJT90], qui propage des intervalles ordinaires (comme valeurs avec tolérances) et des hypothèses de bon fonctionnement des composants.

- Notre ATMS flou qui propage des intervalles flous pour traiter l'imprécision et l'incertitude d'information et des hypothèses définies comme des ensembles flous.

Enfin, cet algorithme calcule les ensembles minimaux des hypothèses nécessaires pour dériver une formule donnée, ce qui est utile en diagnostic quand on veut trouver le nombre minimal de fautes qui peut expliquer un comportement observé. Notons que si on est intéressé par l'approche de faute simple, l'ATMS général n'est pas nécessaire [McA91].