IA ET SYSTÈME À BASE DE CONNAISSANCES

1.1 Introduction

Dans ce chapitre, nous allons en premier lieu décrire quelques notions sur l'intelligence artificielle. l'IA est née dans les années cinquante et a connu depuis une vingtaine d'années un essor considérable. Les chercheurs dans ce domaine se sont fixés pour objectif d'analyser les comportements intelligents de l'homme.
Au début l'objectif des chercheurs de L'IA était de doter les ordinateurs de mécanisme logiques leur permettant de résoudre des problèmes très généraux comme la démonstration des théorèmes. Plusieurs systèmes experts ont été alors réalisés intégrant chacun une base de connaissances d'un domaine particulier et qui tentent de résoudre les problèmes posés par raisonnement logique.
La base de connaissances est modifiée en cours de temps soit par enrichissement automatique soit par ajout ou suppression de connaissance par l'expert du domaine. Ces opérations peuvent provoquer certaines anomalies. Dans ce chapitre, nous présentons des méthodes de vérification et de validation de bases de connaissances afin de maintenir la cohérence de la base de connaissances et du raisonnement.
En deuxième lieu on essaye d'expliquer le processus de maintien de cohérence SMV qui va vérifier toute nouvelle insertion ou suppression dans la base de connaissances.

1.2 L'intelligence artificielle

Pour bien comprendre les systèmes experts, il convient de les replacer dans un contexte plus large qui est celui de l'intelligence artificielle. En effet, les chercheurs tigués jusqu'alors, à savoir les différentes facettes mystérieuses de l'intelligence humaine. C'est ainsi qu'ils se sont attelés à modéliser des fonctions telles que la parole, la vision, la compréhension, les mouvements..., tant de perceptions si banales pour tout être humain mais tellement complexe à appréhender et à modéliser. La médecine a apporté de nombreuses contributions récentes quant au fonctionnement de notre cerveau mais de nombreux aspects demeurent encore flous voire inconnus. De plus, les fonctions implémentées au niveau de la machine offrent des résultats encore timides en comparaison de l'efficacité démon-

trée par notre intelligence. Néanmoins, ces résultats sont encourageants et promulguent l'émergence d'une nouvelle discipline informatique baptisée intelligence artificielle. [10]

Définitions :

Il s'avère difficile de cerner par quelques mots toute la richesse couverte par ce domaine néanmoins, nous pouvons reprendre ici quelques définitions d'auteurs et de chercheurs en intelligence artificielle qui nous semblent significatives :

E.Feigenbaum et P.McCorduck font la réflexion suivante : «...le terme latin "Intelligence" signifie choisir parmi, comprendre, percevoir et savoir. Si nous pouvons imaginer un objet capable de réunir, de rassembler, de choisir, de comprendre, de percevoir et de savoir, nous nous trouvons en face d'une intelligence artificielle». [10]

Clyde W.Holsapple et Andrew B.Whinston :«l'intelligence artificielle s'efforce de créer des machines telles que les ordinateurs capables d'un comportement intelligent, c'est à dire un comportement qui pourrait raisonnablement passer pour intelligence si on l'observait chez des êtres humains. Aptitude à comprendre le langage naturel et aptitude à raisonner sont deux clés de voûtes de l'intelligence. Elles représentent chacune un domaine de recherche principal pour l'IA.» [10]

Elaine Rich :«l'intelligence artificielle est le domaine qui étudie comment faire exécuter par l'ordinateur des tâches pour lesquelles l'homme est, aujourd'hui encore, le meilleur». [10]

A.VLW définit l'intelligence artificielle comme une nouvelle classe d'outils informatiques qui veulent représenter de manière explicite et manipuler une importante quantité d'informations pour résoudre des problèmes à caractère symbolique très spécifiques. [10]

Les différents domaines :

Les domaines d'application aujourd'hui de l'I.A., sont de plus en plus variés : le diagnostic médical, la conduite processus industriel, l'interrogation de banques de données, le conseil de placement financier, le pilotage de véhicules autonomes, le design des puces, la chimie interprétative, le design de structure, la vision, la robotique, la reconnaissance de langage parlé et écrit, l'enseignement, etc. [18]

1.3 Systèmes à base de connaissances

La conception de systèmes à base de connaissances et, notamment, de systèmes experts, constitue un domaine majeur en IA. Les systèmes experts sont conçus pour atteindre les performances d'experts humains dans des domaines limités en exploitant un ensemble de connaissances acquises pour l'essentiel auprès de ces experts. Apparus vers 1975 (cf. le système de diagnostic médical MYCIN4), ils ont eu un impact certain sur l'IA, et aussi un retentissement médiatique parfois exagéré. Le terme de système expert disparaît, au profit du concept plus général de système à base de connaissances (SBC). Ce concept est fondé sur une séparation entre les connaissances nécessaires pour résoudre un problème et les mécanismes de raisonnement exploitant ces connaissances [7].

1.4 Système expert

Edward Feigenbaum a donné la définition suivante pour les systèmes experts :«les systèmes experts sont des programmes conçus pour raisonner habilement à propos des tâches

dont on pense qu'elles requièrent une expertise humaine considérable » [10]
Elaine Rich et Kiven Kuright définissant les systèmes experts comme des programmes résolvant des problèmes qui sont habituellement résolus par des experts humains. Pour cela, ils requièrent un accès à une base de connaissances conséquente qui doit être construite de façon efficiente. Ils doivent être à même de fournir divers modes de raisonnement et de justifier les conclusions auxquels ils aboutissent. [10]
D'après les deux définitions, on trouve qu'un système expert est un programme qui simule le raisonnement d'un spécialiste dans un domaine précis. Un système expert est un programme interactif qui permet à l'utilisateur de prendre des décisions suite à des raisonnements par des questions et des réponses. [10]

1.5 conditions d'application d'un système expert :

Pour utiliser des systèmes experts de façon à cerner les applications où une telle approche apporte réellement une puissance et une souplesse plus grande que la programmation classique [10] :
1- La nature de problème :
celui-ci doit faire appel à la notion d'expérience où on traite avant tout une information du type symbolique plutôt que numérique la résolution des problèmes s'appuiera sur des heuristiques et utilisera un minimum d'algorithmes.
2- La disponibilité de l'expertise c'est à dire des experts humains dans le domaine ou bien des documents tel que des livres etc...
3- La nécessité de résoudre des problèmes comportant une certaine incertitude.
4- La taille et la complexité de problème :
comme la représentation des connaissances se fait bien souvent de façon explicite et exhaustive, le domaine d'expertise doit être relativement étroit.
5- La stabilité des connaissances afin de minimiser les coûts associés à la maintenance du système.

1.6 Les principaux composants d'un système expert

Les deux composants majeurs d'un système expert sont la base de connaissances et le moteur d'inférence (Figure1.1).

1.6.1 Base de connaissances

La base de connaissances peut être imaginée comme étant au système expert ce que constitue la base de données au système de gestion de bases de données(SGBSD). [10]
Une grande différence peut-être remarquée : la base de données ne contient que des données passives, par contre la base de connaissances contient à la fois des données passives(les faits), et des données actives (les règles). [10]

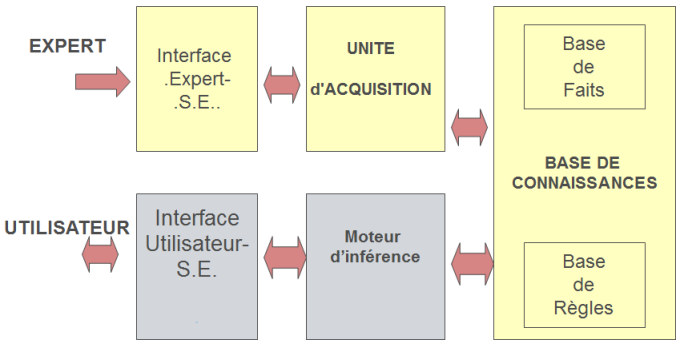

FIGURE 1.1 – Schéma d'un système expert

1.6.1.1 La représentation des connaissances

Une question fondammentale à se poser concerne le formalisme selon lequel sera exprimé l'ensemble des connaissances dont on dispose au sujet du domaine abordé.
Plusieurs formalismes existent. Nous allons présenter dans la suit.

a) Les réseaux sémantiques :

Un réseau sémantique est constitué de différents noeuds représentant des concepts reliés par des arcs exprimant les relations qui les lient entre eux. Il s'agit d'un graphe orienté sans cycle où les noeuds représentent les concepts. Les réseaux sémantiques sont à la base de l'interprétation du langage naturel (Figure1.2). [10]

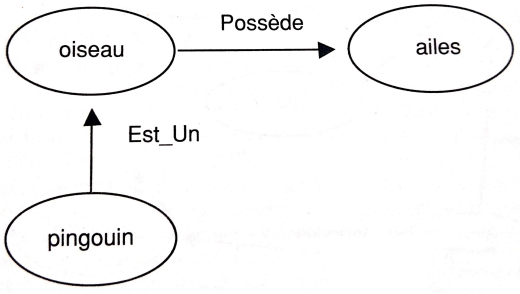

FIGURE 1.2 – réseau sémantique simple

b) Les réseaux neuronaux :

Les réseaux neuronaux est un formalisme qui s'inspire de modèle biologique de neurone. [10]

Le neurone informatique : le neurone informatique (formel) est modélisé comme suit (Figure1.3) :

ou wi est le poids de l'entreé i(ei)
E=h(e1,e2,...,en) ou h est la fonction d'entrée totale.
f= fonction d'activation.
s= seuil d'activation.
S=g(f(E))=réponse du neurone.
Le neurone calcule la somme pondérée (par les poids) de ses entrées. Il fournira une réponse positive si et seulement si cette somme dépasse un certain seuil (seuil d'activation). [10]

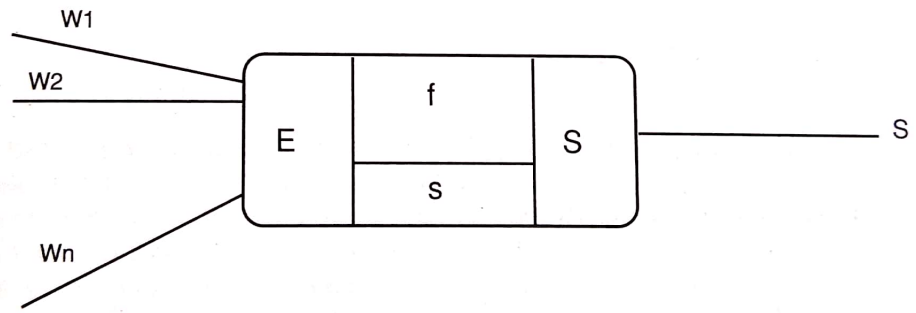

FIGURE 1.3 – Le neurone formel

Perceptron multi couches :
- Le perceptron multi couches est un ensemble des neurones organisés en couches successives au sein desquelles une information circule de la couche d'entrée vers la couche de sortie uniquement il s'agit donc d'un réseau à propagation directe (Figure1.4). [12]
- Chaque couche est constituée d'un nombre variable de neurones, les neurones de la dernière couche dite «de sortie» étant les sorties du système global (Figure1.4). [12]
- La couche d'entrée reçoit les signaux (ou variables) d'entrée et la couche de sortie fournit les résultats. Enfin, les neurones des autres couches (couches cachées) n'ont aucun lien avec l'extérieur et sont appelés neurones cachés (Figure1.4). [12]

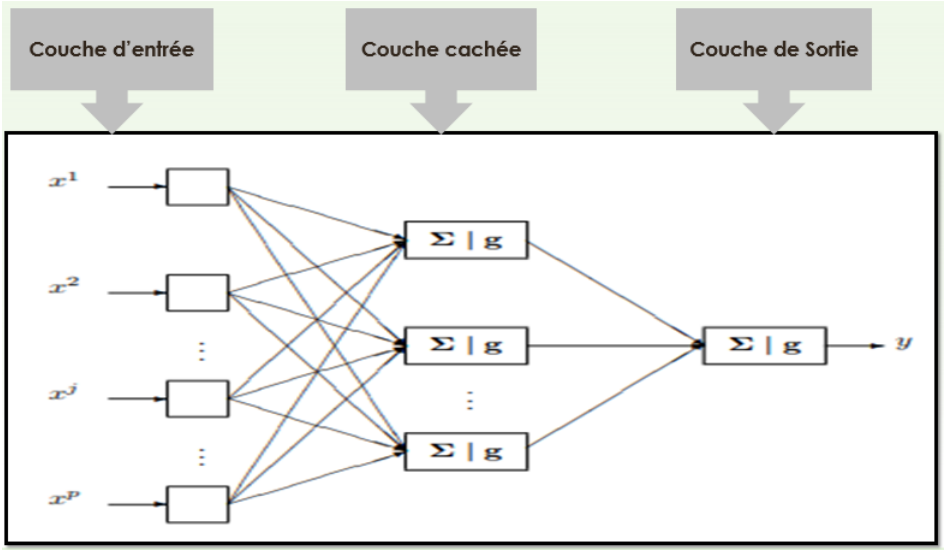

FIGURE 1.4 – Perceptron multi couches

c) Les règles de production :
La majorité des systèmes experts ont été développés en utilisant le formalisme règle de production pour la représentation de la connaissance. En général, une règle de production est exprimée sous forme : Si conditions Alors conclusion (Action). [10]
Ce formalisme présente de nombreux avantages : [10]
- Facilité d'expression.

- Large diffusion des logiciels utilisant cette technique.
- Clarté de la connaissance grâce à la transparence des règles permettent une implication plus profonde des utilisateurs dans le développement.
- Mise à jour aisée.
- Vérification et validation de la connaissance possible.

1.6.1.2 Base de faits

- Un fait :est un atome qui doit être vérifié ou qui apporte la vérité, un fait est toujours vrai selon un critère donné. [1]
- Base de faits : espace de travail contient les faits utilisés et les nouveaux faits inférés, cet espace joue le rôle d'une mémoire RAM (Random access memory) dans un ordinateur, c'est le savoir du système. [11]
Les faits peuvent être de deux types :
- Les faits d'ordre zéro (on parle alors de système d'ordre 0), exemple : "Il fait beau". [9]
- Les faits d'ordre un (on parle alors de système d'ordre 1) représentent non pas un fait précis, mais une classe de faits grâce à des variables qui peuvent prendre une infinité des valeurs, exemple :"le prix du produit P chez le concurrent C et X". [9]

1.6.1.3 Base de régles

- Une règle : Une règle est fournie par l'expert humain qui s'appuie sur son expérience dans le domaine de son expertise, une règle permet de déduire d'autres faits. [1]
- Une règle est fondamentalement représentée sous forme d'un couple (<condition>,<action>) en respectant la syntaxe Si <condition> Alors <action>. [11]
- La base de règles : Est une base de données qui contient un ensemble des règles de production, c'est le savoir faire du système. [11]

1.6.1.4 Métas connaissances

Ce sont des connaissances sur les connaissances. Par exemple nous savons la limite et l'origine de notre connaissance sur un sujet particulier, ou sur la fiabilité de certaines informations, ou l'importance relative d'un fait particulier du monde qui nous entoure, ou notre expertise dans un domaine particulier [18].
Exemple :
lorsque vous donnez une importance à une connaissance, ceci est considéré comme une méta connaissance.

1.6.2 Interfaces

Utilisateur : Interroge le système à travers une interface dédiée. Il n'est pas forcément un expert dans le domaine traité par l'application. L'interaction entre l'utilisateur et le système doit se faire au travers d'une interface de dialogue dont l'ergonomie pas de connaissances informatiques. [11]

1.6.2.1 Interface utilisateur

Pour la communication entre le système expert et l'utilisateur.
Il est décomposé de deux modules :
1- module de dialogue : pour permettre aux systèmes experts de poser des questions à l'utilisateur au besoin et lui fournir des réponses. [11]
2- module d'explication : Pour permettre au moteur d'inférence d'expliquer comment il utilise le contenu de la base de connaissances pour répondre aux questions de l'utilisateur. [11]
L'Expert : Enrichis la base de connaissances du système. La phase d'acquisition de la connaissance est une première tâche à effectuer pour constituer un système expert. Le système possède en général une interface pour l'introduction et la mise à jour des connaissances. L'expert est assisté par un cogniticien. Ce dernier possède une double compétence. Il a des connaissances approfondies dans le domaine de l'expertise. De plus, il possède les connaissances du domaine de l'IA afin de proposer des représentations adéquates pour formaliser et pour structurer les données de l'expertise. [11]

1.6.2.2 Interface expert

L'interface experte comporte trois choses [11] :
1- un éditeur pour insérer ou modifier la base de faits et la base de règles.
2- un éditeur de dialogue : permettant aux experts de prévoir les questions que le système devra poser.
3- un analyseur pour traduire les connaissances sous forme des données reconnues par les moteurs d'inférence.

1.6.3 Moteur d'inférence

C'est un programme qui simule le raisonnement humain, il forme la partie du système qui fournit des réponses aux questions des utilisateurs.
Le moteur d'inférence est l'équivalent de l'esprit logique de l'expert humain qui raisonne sur des faits connus pour en sortir de nouvelles vérités. [1]

1.6.3.1 Les caractéristiques d'un moteur d'inférence

Les caractéristiques d'un moteur d'inférence sont [10] :
1- Un cycle de base

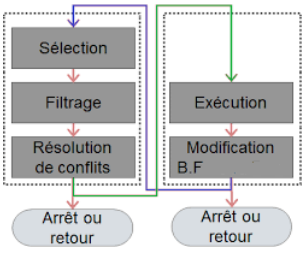

FIGURE 1.5 – Cycle de base d'un moteur d'inférence

- Phase de sélection(restriction) :
L'objectif de cette tâche est de trier et de rassembler un sous-ensemble des faits et des règles de BC, cette phase permet une économie de temps au profit de la phase suivante (Figure1.5).
- Phase de filtrage :
Déterminer l'ensemble des règles applicables (on l'appel aussi ensemble conflit) (Figure1.5).
- Phase de résolution de conflit :
Le choix de la règle à appliquer (Figure1.5).
- Phase d'exécution :
Dans cette phase on applique la règle choisie à la phase précédente, cette étape permet d'ajouter un ou plusieurs faits dans BF(Base de faits) (Figure1.5). [10]
2- Les stratégies de recherche
- La recherche en largeur d'abord :
Dans ce mode de recherche, l'on parcourt l'arbre à partir de la racine de gauche à droite. Ainsi, dans l'arbre qui suit, les noeuds buts c'est-à-dire ceux qui appartiennent au domaine des solutions recherchées, sont étiquetés par la lettre b. Les chiffres correspondent à l'ordre de prise en considération (Figure1.6). [10]

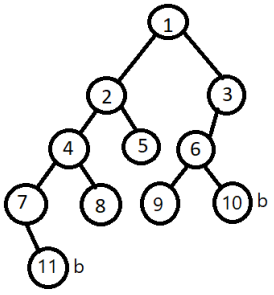

FIGURE 1.6 – recherche en largeur d'abord

- La recherche en profondeur d'abord :
Dans ce mode, on parcourra l'arbre de résolution à partir de sa racine en explorant une branche jusqu'à son terme avant de passer à la branche adjacente (Figure1.7). [10]

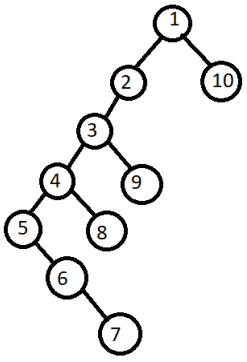

FIGURE 1.7 – recherche en profondeur d'abord

- La recherche heuristique :
Plutôt de procéder à des recherches aveugles, on exploite la connaissance spécifique sur le problème traité. Pour cela on injecte dans le processus de résolution du problème une certaine connaissance sous la forme agrégée d'une fonction d'évaluation. Celle-ci permet de mesurer la «promesse» de se trouver sur le chemin le plus intéressant menant au but recherché. Il est important de noter qu'une heuristique garantit une bonne solution mais pas forcément la solution optimale. [10]

3- Les méthodes de chaînage

- Chaînage avant : Dans la phase filtrage on déduit les règles dont les prémisses sont vérifiées (voire l'algorithme 1). [10]

Algorithme 1: ChaînageAvant

```
1  Entree : BF,BR,F(est le fait but cherché);
2  début
3  |  tant que (F ∉ BF et ∃ R ∈ BR) faire
4  |  |  (R applicable);
5  |  |  - choisir une règle applicable R;
6  |  |  BR = BR - R;
7  |  |  BF = BF + conclusion de R;
8  |  fin
9  |  si (F ∈ BF) alors
10 |  |  F est établi;
11 |  sinon
12 |  |  F n'est pas établi;
13 |  fin
14 fin
```

- Chaînage arrière : Le moteur d'inférence prend le but final comme hypothèse, les conditions inconnues de la prémisse des règles qui concluent sur ce but deviennent des sous-buts, le processus est répété de manière récursive jusqu'à ce que tous les sous-buts sont à leur tour démontrés. [10]

Remarque :
S'il n'est pas possible de sélectionner des règles, alors le système peut demander à l'utilisateur de résoudre un ou plusieurs sous-but (en posant des questions à l'utilisateur) et le processus recommencera (voir l'algorithme 2). [10]

Algorithme 2: ChaînageArrière(P); `/* P : le but cherché */`

```
15 début
16   si (p ∉ BF) alors
17     S← Ensemble des règles concluant sur P;
18     si (S ≠ ∅) alors
19       A← choix d'une règle de S;
20       Marquer la règle A;
21       si (A est applicable) alors
22         Appliquer A puis ajouter P à BF;
23       sinon
24         pour (toutes hypothèse h de A, h∉BF) faire
25           ChaînageArrière(h);
26         fin
27         si ( A ) alors
28           appliquer A puis ajouter h à BF
29         sinon
30           si toutes les règles de S sont marquées alors
31             le but P n'est pas atteint
32           sinon
33             A ← choix d'une règle de S non marquée;
34           fin
35         fin
36       fin
37     fin
38   fin
39 fin
```

- Chaînage mixte : Combine les deux chaînages (voire l'algorithme 3). [10]

Algorithme 3: ChaînageMixte

```
40 Entrée : F (à déduire);
41 début
42   tant que (F n'est pas déduit mais peut encore l'être faire) faire
43     - Saturer la base de faits par chaînage avant;
44     - Chercher quels sont les faits encore déductibles;
45     - Déterminer une question pertinente à poser à l'utilisateur et ajouter sa
       réponse à la base de faits;
46   fin
47 fin
```

4- Stratégies de résolution de conflit

Deux types de stratégie :

Stratégie fixe :

- La dernière règle utilisée.
- La règle la plus utilisée.
- la règle se rapprochant le plus de l'objectif.

- la règle la plus spécifique on dit que R1 est plus spécifique que R2 si R1 contient toutes les propositions de R2. [10]

Méta règle :

Celles-ci sont des règles qui concernent l'état de processus de déduction et qui modifient dynamiquement l'ordre de priorité de règle.

En fonction de l'état des faits déjà établis, l'ordre de déclenchement des règles peut être changé. [10]

1.7 Les systèmes de maintien de vérité

Un SMV est un programme qui fonctionne en tandem avec un moteur d'inférence ou avec n'importe quel autre agent à qui il offre certains services en enregistrant les raisonnements qu'ils ont effectués. [6]

L'origine de ces systèmes remonte au milieu des années soixante-dix avec les travaux de Stallman et Sussman qui s'intéressent aux problèmes des retours en arrière guidés par dépendance (Dependency directed backtracking) [5]. Au cours de cette période, plusieurs systèmes ont été développés entre autres par London (1978), McAllester (1978), Doyle (1979), Reiter et De Kleer (1987), De Kleer et William (1986), etc [5]. L'expression "Système de maintien de vérité" a été cependant popularisée par Doyle (1979) qui la remplace par celle de "Système de Maintien des Raisons" afin d'éviter certaines ambiguïtés [5].

Le fait de recourir aux SMV présente l'avantage de laisser le moteur d'inférence s'occuper de la résolution du problème alors que la tâche de maintien de vérité est déléguée à un système indépendant. Pour fonctionner, les deux systèmes doivent utiliser un même protocole de communication et chacun doit gérer ses propres bases de données (Figure1.8). Ces bases doivent être synchronisées. Par conséquent, à chaque assertion de la base de faits du moteur d'inférence, un élément correspondant doit exister dans celle du SMV [6].

Tout SMV doit être capable d'accomplir les fonctions suivantes [6] :

- Mettre en cache un raisonnement et le fournir sur demande ;
- Permettre d'effectuer des révisions ;
- Détecter des contradictions quand elles surviennent ;
- Fournir des explications ;
- Servir de guide lors des retours en arrière guidés par dépendances ;
- Permettre d'effectuer des raisonnements par défaut.

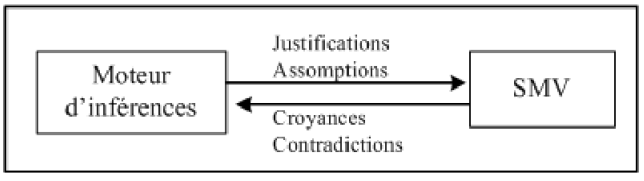

FIGURE 1.8 – Architecture d'un solveur de problème

1.7.1 Les SMV à base de justifications

Un SMV est un programme qui sert à déterminer un ensemble de croyances à partir d'un ensemble de raisons ou justifications [4]. Ce système permet de mettre à jour cet ensemble de croyances en tenant compte de toute nouvelle "raison" qui y est ajoutée ou rétractée au cours d'un processus de raisonnement. Une croyance qui est mise hors de l'ensemble des croyances est dite OUT. Elle est dite IN quand elle est introduite [4]. IN et OUT peuvent être interprétées respectivement par "il est vrai" et "il est faux" qu'un fait existe dans la base de croyances. [6]
Les croyances initiales qui peuvent être rétractées sont appelées "assomptions" ; donc, un SMV doit déterminer les conséquences de tout changement dans l'état de ces assomptions comme il doit pouvoir retracer les raisons qui ont conduit à ces changements. Le même processus est entrepris si une croyance doit être reconsidérée [4]. L'ensemble des changements dans la base du SMV simule donc l'état dans lequel la base de faits du moteur d'inférence devrait être si celle-ci a été modifiée réellement.

1.7.1.1 Justification

Formellement une justification est une connaissance qui permet de déterminer l'état d'un noeud c.a.d justifier son inclusion dans le contexte. Une justification comporte deux listes [11] :
- Liste IN : ensemble de noeuds devant avoir l'état IN pour que la justification puisse être valide.
- Liste OUT : ensemble de noeuds devant avoir l'état OUT pour que la justification puisse être valide.
- On note un noeud justifier par :

{Proposition /
Etat /
IN (l'ensemble des nœud IN)
OUT (l'ensemble des nœud OUT) }

L'état : L'état d'un noeud reflète la croyance attachée à la proposition qu'il représente
Etat=IN : noeud valide(noeud est dans le contexte courant). [11]
Etat=OUT : noeud invalide (noeud $\notin$ au contexte courant)
OUT est l'état d'un noeud exclu du contexte parcequ'il est incompatible avec l'un des noeuds du contexte courant ou bien aucune justification ne permet de croire en sa validité.
Proposition : C'est une connaissance manipuler par le MR (Module de Raisonnement), généralement elle est factuelle (fait) mais elle peut être aussi une règle de production et même une contrainte. [11]

1.7.1.2 Traitement de la négation et des contradictions

La représentation explicite de la négation dans un SMV nécessite la création d'un nœud distinct. Une contradiction est détectée automatiquement lorsque :
un nœud ainsi que le nœud représentant sa négation existent (A et $\neg$A) et sont à IN simultanément. [6]
Une contradiction peut résulter aussi de la combinaison de n'importe quels autres nœuds qui sont mutuellement exclusifs quand cela est spécifié expressément. [6]

Dans ces deux cas, un nœud est automatiquement créé par le SMV pour signaler la contradiction.
La figure1.9 est un exemple qui illustre ces deux situations ainsi que les choix possibles qui doivent être pris pour résoudre les contradictions qui en résultent. [11]

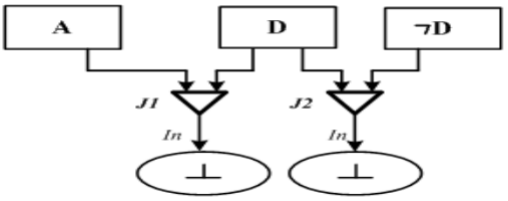

FIGURE 1.9 – Gestion des contradictions

1.7.1.3 Propagation et non monotonie

Un simple SMV ne fait que propager les mêmes états alors qu'un SMV non monotone permet de déduire l'état d'activité d'un nœud en l'absence de celle d'un autre [17]. Dans ce cas, le changement de l'état d'un nœud peut se traduire aussi bien par la propagation de cet état que par celle de l'état contraire. Le support de la non monotonie est essentiel car il permet de représenter l'exclusion mutuelle et la logique des défauts [6].
Pour supporter la non monotonie dans le système de Doyle, chaque nœud est muni d'une paire de listes, une liste des nœuds à mettre à IN et une liste des nœuds à mettre à OUT. Quand l'état d'un nœud change, les nœuds des deux listes sont mis à jour en conséquence.

1.7.1.4 Propagation et circularité

Lorsqu'un nœud doit être activé ou désactivé, le SMV doit propager tout changement dans le reste du réseau, détecter en même temps toute contradiction et composer avec les chemins circulaires. Un changement peut survenir si l'état d'activité d'un nœud change, si un nœud est ajouté au réseau ou si une quelconque relation entre nœuds est modifiée [4] [6].
Le processus de propagation doit tenir compte d'un certain ensemble de contraintes. Premièrement, un et un seul processus de propagation est effectué à la fois. Deuxièmement, si une contradiction est détectée, le processus doit être arrêté. Troisièmement, si un état est propagé vers un nœud qui est déjà dans cet état, alors aucun changement ne doit être effectué et la propagation doit être arrêtée. Elle peut se poursuivre éventuellement sur d'autres chemins tant que ceux-ci contiennent des nœuds dont l'état peut être modifié. Finalement, le SMV doit être en mesure d'éviter de rentrer dans des boucles infinies. Pour éviter les circuits, le SMV marque simplement tout nœud visité et le maintient dans cet état tant que le processus de propagation n'a pas été complété. [6]

1.7.1.5 Circularité et supports bien fondés

Un nœud peut posséder plus qu'un précédent et il ne peut être désactivé que si tous ses précédents ne sont pas dans un état qui lui permet de rester actif. On appelle "support

bien fondé" toute justification qui, sans être parmi les conséquents d'un nœud, permet à ce dernier de rester actif [4]. En effet, quand un nœud a été désactivé, l'état de l'ensemble de ses précédents doit être vérifié pour trouver un support lui permettant de rester actif. Cependant, pour éviter qu'un support ne soit trouvé parmi les conséquents, sa recherche ne doit être entreprise qu'à la fin d'un même processus de propagation [6]. Par exemple dans la Figure1.10, si le nœud C est désactivé, B le sera aussi. Si le SMV cherche immédiatement un support pour B, il va trouver A par J1 alors que celui-ci sera mis à OUT à travers J2. Dans ce cas, A ne peut pas servir de support pour B et le SMV doit continuer la propagation jusqu'à la fin du circuit pour éviter de le considérer.

Pour détecter la présence de circuits, le SMV marque tout nœud visité. À la fin de chaque processus, tout nœud marqué est remis dans son état initial et un nouveau chemin peut être exploré alors [6].
La Figure1.11 illustre un cas non monotone qui peut aboutir à deux résultats différents

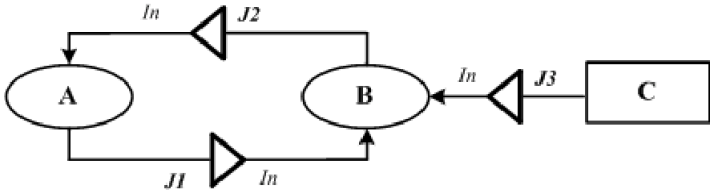

FIGURE 1.10 – Supports bien fondés

mais tous valables. Dans cet exemple, si le nœud A est actif alors B et C vont l'être automatiquement. S'il est désactivé alors deux cas sont possibles dépendamment du chemin que la propagation empruntera en premier, par J3 ou par J4.
Si la propagation débute par J3 alors B sera mis à OUT puis marqué. Puisque B est

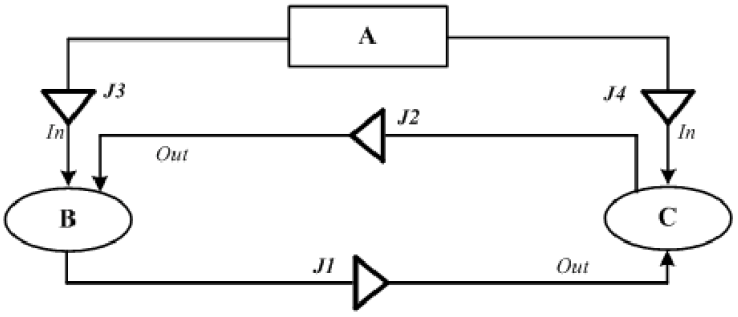

FIGURE 1.11 – Circularité et propagation

OUT, le SMV essaiera de mettre C à IN mais celui-ci l'est déjà. B sera ensuite démarqué et un support sera cherché mais aucun ne sera trouvé. En effet, A étant à OUT, il ne reste que C mais celui-ci est à IN. Si C est à IN, B doit être à OUT et il l'est déjà. Le processus de propagation par J3 prend fin et continue par J4. C sera mis à OUT puis marqué, B sera ensuite mis à IN puis marqué. La propagation prend alors fin puisqu'elle ne peut pas continuer par J1 car C est déjà marqué. Un support sera ensuite cherché pour C mais ni A ni B ne peut servir à cette fin. C sera alors démarqué ainsi que A et la propagation

s'arrête définitivement. A et C finiront donc à OUT et B à IN.

Si la propagation avait débuté par J4, alors A et B auraient fini à OUT et IN respectivement puisque le réseau est symétrique.

L'ordre dans lequel le SMV enregistre ses nœuds ainsi que celui qu'il choisit pour la propagation déterminent donc l'état final du réseau. Cet ordre doit être considéré dans la stratégie de la résolution d'un problème de l'agent qui recourt au SMV. La compréhension de la propagation dans un contexte non monotone et circulaire est essentielle et les multiples résultats possibles ne doivent pas forcément être pris pour des dysfonctions du système. [6]

1.8 Conclusion

Dans ce chapitre, nous avons dans un premier temps donné une introduction à l'intelligence artificielle et des généralités sur les connaissances, ensuite nous avons présenté la notion de Système à base de connaissances système expert (domaines d'application, architecture principale et le cycle de fonctionnement du moteur d'inférence).
Dans notre travail nous avons choisi de concevoir et réaliser un système expert utilisant le formalisme règles de production de la logique des prédicats pour la représentation de connaissance et un raisonnement en chaînage avant pour la résolution des exercices de géométrie de classes moyennes.
Pour le problème de vérification et validation d'une base de connaissances nous avons présenté des outils de vérification classique, puis l'approche SMV.
Dans le chapitre suivant, nous introduisons l'évolution de l'IA vers l'IAD et particulièrement, les systèmes à tableaux noirs et les systèmes multi-agents.