Coopération et évaluation cognitive d'agents artificiels pour la supervision

Gilles Dumont d'Ayot

N° ORDRE : 772

THÈSE

présentée
devant l'INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE TOULOUSE

en vue de l'obtention du

DOCTORAT

Spécialité : Systèmes Industriels

par M.
Gilles DUMONT D'AYOT

COOPÉRATION ET ÉVALUATION COGNITIVE D'AGENTS ARTIFICIELS POUR LA SUPERVISION

Soutenue le 23 mai 2005

Rapporteurs

R. LOPEZ DE MANTARAS : directeur de recherche au IIIA-CSIC à Barcelone (Espagne)
C. KOLSKI : professeur de l'université de Valenciennes et du Hainaut-Cambrésis
M. POLIT : professeur de l'université de Perpignan

Directeurs de thèse

L. TRAVE-MASSUYES : directeur de recherche au LAAS-CNRS à Toulouse
J. AGUILAR-MARTIN : directeur de recherche émérite au LAAS-CNRS à Toulouse

Président du jury

M.V. LE LANN : professeur de l'INSA de Toulouse

Invité

X. OLIVE : ingénieur-docteur au sein d'Alcatel Space Industries à Cannes

REMERCIEMENTS

Je tiens tout d'abord à remercier particulièrement mes directeurs de thèse :

- Joseph AGUILAR-MARTIN, directeur de recherche au Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS-CNRS) à Toulouse, pour m'avoir accueilli au sein du groupe DISCO du LAAS-CNRS et m'avoir permis de réaliser cette thèse.

- Louise TRAVE-MASSUYES, directeur de recherche au LAAS-CNRS à Toulouse, pour m'avoir donné la possibilité de travailler sur un sujet vraiment orienté vers mes centres d'intérêt profonds.

Tous deux m'ont en outre donné tout au long de ma thèse de précieux conseils et ont été d'une grande disponibilité.

Je tiens également à remercier les membres du jury pour avoir pris le temps de juger mes travaux :

- Monique POLIT, professeur à l'université de Perpignan, et directeur du Laboratoire de Physique Appliquée et d'Automatique (LP2A),

- Marie-Véronique LE LANN, professeur à l'INSA de Toulouse, et membre du LAAS-CNRS,

- Xavier OLIVE, docteur de l'université Paul Sabatier de Toulouse, et membre du groupe ALCATEL SPACE INDUSTRIES à Cannes,

- Ramon Lopez DE MANTARAS, directeur de recherche au Artificial Intelligence Research Institute (IIIA) - Spanish Scientific Research Council (CSIC) en Espagne,

- Christophe KOLSKI, professeur à l'université de Valenciennes et du Hainaut-Cambrésis, membre du Laboratoire d'Automatique, de Mécanique, et d'Informatique industrielles et Humaines (LAMIH UMR-CNRS), et qui a en outre été un de mes enseignants lorsque j'ai effectué un DESS en Informatique à l'université de Valenciennes.

Résumé :

L'idée générale de cette thèse est d'évaluer la plausibilité cognitive de modèles de raisonnement couramment utilisés en Intelligence Artificielle en les mettant en correspondance avec le raisonnement humain, ceci dans des situations de supervision de systèmes dynamiques. Pour cela, un micro-monde lié au domaine de la physique hydraulique est utilisé. Il s'agit donc en premier lieu de développer des opérateurs artificiels capables de superviser ce processus, chacun dans un "style cognitif" spécifique. Trois types d'agents ont ainsi été mis au point, basés sur le raisonnement qualitatif, la classification à partir d'exemples, et le raisonnement à base de cas. La seconde phase consiste à comparer les séquences d'actions correspondant aux raisonnements d'un opérateur humain et d'un agent artificiel durant la tâche de supervision, notamment par l'intermédiaire de la découverte des intentions associées aux actions, puis à définir une coopération entre ces trois modes fondamentaux de raisonnement.

Mots clés :

Supervision dynamique de procédés, micro-monde, processus cognitifs, raisonnement qualitatif, classification à partir d'exemples, raisonnement à base de cas, comparaison entre les raisonnements d'opérateurs humain et artificiel, intention associée à une action, logique floue.

Abstract :

The general idea of this work is the evaluation of the cognitive plausibility of classic reasoning models used in Artificial Intelligence, by comparing them with human reasoning in supervision situations. For that purpose, a micro-world has been developed in the area of physical hydraulic. The first part is to implement artificial agents, each modelling a specific cognitive style, based on qualitative reasoning, categorization, and case-based reasoning. The second part is to compare the sequences of actions made by a human operator and an artificial agent during the supervision task, by the discovery of the intentions hidden behind the actions, and to define cooperation between these three fundamentals types of reasoning.

Key words :

Dynamic process supervision, micro-world, cognitive process, qualitative reasoning, categorization, case-based reasoning, comparison between human and artificial operator, action's intention, fuzzy logic.

SOMMAIRE

Chapitre 1

INTRODUCTION

L'idée générale de ces travaux est d'étudier la coopération et d'évaluer la plausibilité cognitive de modèles issus de l'Intelligence Artificielle (IA) et utilisés par le groupe Diagnostic Supervision COnduite (DISCO) du LAAS-CNRS, dans le cadre de la supervision de processus, un des thèmes prépondérant du groupe DISCO.

La supervision de procédés est mise en œuvre dans plusieurs domaines, tels que les centrales nucléaires, la médecine, la navigation aérienne,… Des erreurs de la part de l'opérateur de supervision peuvent avoir des conséquences graves. Pour limiter ces erreurs, on peut soit automatiser le processus de façon poussée (voire ne plus utiliser d'opérateur humain), ce qui est possible pour des tâches simples et routinières, soit fournir à l'opérateur humain une aide intelligente ou expertise (tout en laissant la décision finale à l'opérateur humain), ce qui implique une compréhension poussée de son raisonnement, aussi bien lorsqu'il y a réussite de l'opération de supervision que lorsqu'il y a des erreurs. En effet, le raisonnement mis en jeu dans la supervision fait appel à divers processus cognitifs complexes : prise de décision (éventuellement en temps contraint), diagnostic, prédiction, génération de stratégies de résolution de problèmes, planification d'un but, retour arrière en cas d'erreur de raisonnement, adaptation à des situations nouvelles, ... Cette aide peut s'appuyer naturellement sur des solutions issues de l'IA. Le but étant de s'approcher au mieux des processus cognitifs du cerveau humain, il serait utile de comparer le raisonnement d'un opérateur humain en situation de supervision avec le raisonnement simulé grâce à une technologie d'IA afin d'évaluer la plausibilité cognitive de cette aide [Ponsa et Catala 99 (b)]. Sachant également que chaque technologie d'IA modélise en général une seule caractéristique du raisonnement humain (par exemple la logique floue modélise le raisonnement imprécis et incertain, le raisonnement à base de cas modélise le raisonnement à partir d'expériences passées, les systèmes de maintien de vérité TMS et ATMS [Haton 91] modélisent le raisonnement hypothétique, la logique formelle et les systèmes à base de règles modélisent le raisonnement déductif, certaines logiques non classiques modélisent le raisonnement temporel ou non monotone), il paraît intéressant de vouloir faire coopérer ces diverses technologies toujours dans le but de s'approcher du raisonnement humain dans ses diverses composantes. D'autre part, le but de cette aide intelligente est également la conception d'interfaces homme-machine (IHM). Globalement, une IHM doit permettre à l'opérateur humain de tirer avantage de ses capacités de perception et d'action, tout en fournissant le soutien nécessaire pour les activités de résolution de problème afin de gérer efficacement les évènements non anticipés [Thuriot *et al.* 01]. L'IHM peut être considérée comme un médiateur entre l'opérateur humain et le processus qui doit être contrôlé. Dès lors, il paraît naturel de vouloir évaluer la compatibilité entre d'une part la représentation du processus par l'interface, liée aux sciences de l'ingénieur (intelligence artificielle, …), et d'autre part la représentation du même processus par l'humain, liée aux sciences cognitives (ergonomie cognitive, psychologie cognitive, …).

Notre projet est d'étudier la supervision de processus par un opérateur humain d'une part, et par une machine (opérateur artificiel) d'autre part, afin de comparer leurs raisonnements et évaluer leur compatibilité.

Pour cela, un environnement artificiel a été mis au point. Ce simulateur possède les avantages d'être contrôlable, réaliste, reproductible, simple et permet donc d'isoler les composantes du raisonnement d'un opérateur humain. La tâche de supervision consiste à gérer un mini réseau hydraulique, c'est-à-dire un ensemble de réservoirs reliés par des conduites comportant des vannes et éventuellement des pompes. Les vannes sont les moyens d'action sur le système. La tâche est un problème d'optimisation qui consiste à vidanger le réseau en un minimum de temps tout en minimisant les débordements des réservoirs. Le raisonnement mis en œuvre n'est pas forcément simple car des stratégies complexes peuvent être développées, mais l'intérêt est que les concepts physiques sont relativement simples, de sens commun, ce qui permet d'accéder plus facilement aux différents types de raisonnement, sans l'aide d'un expert du domaine. Il est donc à noter que notre étude porte sur un problème de commande. En effet, le nombre d'états possibles est infini, un état représentant, pour une configuration donnée du réseau, les positions des vannes (ouvertes ou fermées) sur les conduites et les hauteurs d'eau dans les réservoirs. Même si on verra que l'on peut abstraire le problème, ce nombre reste très grand. Il ne s'agit donc pas d'un problème de diagnostic, où le nombre d'états (fautes) est beaucoup plus restreint. Ceci a des répercussions importantes sur le type de méthodes adéquates pour modéliser le raisonnement de l'opérateur de supervision.

Nous avons choisi de développer trois types d'opérateurs artificiels, qui chacun possède une caractéristique essentielle du raisonnement humain, et en outre sont complémentaires :

● un humain a très souvent tendance à raisonner de manière qualitative, par opposition à un raisonnement numérique. Il est capable d'approximer et simplifier une réalité physique complexe sans utiliser les concepts mathématiques et numériques lourds, voire inaccessibles. Bien souvent d'ailleurs, dans la vie courante, il n'a d'ailleurs pas accès à des valeurs numériques et il est alors amené à modéliser de façon imprécise et incertaine, mais dans la grande majorité des situations, la justesse du raisonnement n'en est pas affectée, car il utilise un grand nombre de connaissances liées au contexte, et il modélise les concepts physiques qualitatifs de sens commun, généralement sous la forme de relations causales entre des variables ou des composants d'un système. Ainsi, un opérateur artificiel basé sur un modèle causal de raisonnement qualitatif a été mis au point. Il utilise un graphe causal représentant le système (les réservoirs sont les nœuds, et les conduites les arcs). Il définit la notion d'état d'alarme d'un réservoir, ainsi que deux classes de situations auquel il peut se trouver confronté (situation d'alarme ou non), et pour chacune d'elles un ensemble de buts à satisfaire. L'évaluation numérique de ces buts repose pour certains d'entre eux sur des prédictions qualitatives. L'action choisie est celle qui maximise un score numérique global, parmi un ensemble d'actions admissibles.

● un humain utilise très souvent ses expériences passées, qu'il s'agisse de réussites ou bien d'échecs, afin de résoudre un problème courant, puis apprend à partir de ce problème résolu. Cela consiste généralement à rechercher en mémoire une expérience similaire survenue dans le passé, à adapter cette expérience au contexte présent, à la valider, la ré-utiliser, puis enfin à la stocker en vue d'une ré-utilisation future. Dès lors il paraît intéressant de développer un opérateur artificiel basé sur le raisonnement à partir de cas : l'idée est d'utiliser une librairie de cas stockés pour résoudre un problème nouveau, en se basant sur les notions de similarité, d'indexation et de mise en correspondance. Un cas représente les caractéristiques du problème

(état initial, contraintes), la solution (état final), et éventuellement les pas intermédiaires menant de l'état initial à l'état final. A chaque cas est associé un index comportant les indices descriptifs importants et discriminants. Dans le cadre de notre étude, cet opérateur possède une bibliothèque (une base) de cas représentant des situations type modélisées par un petit nombre de réservoirs et de conduites. Chaque cas est associé à une action qui permet d'atteindre un but. L'opérateur cherche, dans chaque état du système, à reconnaître l'application d'au moins un cas en mémoire. Il effectue alors une mise en correspondance entre le graphe complet modélisant la totalité de la structure et de l'état du réseau auquel il se trouve confronté, et le sous graphe modélisant une partie de réseau décrite de manière abstraite et correspondant au cas en mémoire à appliquer le cas échéant. Chaque différence donne lieu à une pénalisation numérique, et le cas choisi est celui minimisant la pénalisation globale.

● un humain a une très forte propension à catégoriser, qu'il s'agisse de reconnaître un objet (une forme visuelle, un type de problème, un concept, un événement) ou d'attribuer une propriété à un objet. Ainsi, un opérateur artificiel basé sur la classification a été mis au point : il s'agit de catégoriser des situations représentées par un vecteur de couples attributs-valeurs en les associant à une classe ou un concept lors d'une première phase d'apprentissage, puis de reconnaître la classe correspondant à une situation réelle lors de la phase de résolution. Dans le cadre de notre étude, l'entrée correspond à un état du système modélisant le réseau hydraulique à un instant donné, et la sortie est une action sur le système. On peut également utiliser le raisonnement par classification pour catégoriser les actions (ou des séquences d'actions) par l'intention, le but associé à l'action. On peut même imaginer de définir les classes représentant les intentions de manière plus poussée en associant des caractéristiques comportementales, par exemple telle action a été réalisée dans le but de réduire un état d'alarme avec un comportement prudent, ou bien risqué. Cette approche serait utile pour comparer les raisonnements. En effet, comparer deux séquences d'actions peut être vu comme la reconnaissance puis la comparaison des intentions associées aux actions.

Ayant décliné différents types d'agents artificiels, le but est alors d'étudier d'une part l'adéquation entre un opérateur humain et un agent artificiel, et d'autre part dans quelles situations tel opérateur artificiel est plus adéquat qu'un autre, en se basant sur une comparaison avec le raisonnement d'un opérateur humain. L'objectif ultime est de mettre en évidence une complémentarité entre ces trois modes fondamentaux de raisonnement, afin de définir un quatrième agent artificiel qui fera coopérer les trois premiers.

Pour parvenir à ce but, il est nécessaire de définir des critères absolus permettant de mesurer la qualité de la supervision menée par un opérateur (humain ou artificiel), et des critères relatifs permettant de comparer les raisonnements menés par un opérateur humain et un opérateur artificiel. En outre, la comparaison des raisonnements peut se faire en mode Hors Ligne, c'est-à-dire que l'opérateur artificiel résout le problème de supervision de manière autonome et indépendante de l'opérateur humain (d'où une comparaison globale des deux séquences d'actions), ou bien en mode En Ligne, où les deux opérateurs sont face au même problème simultanément et à chaque fois que l'opérateur humain effectue une action, l'agent artificiel indique quelle action il ferait (d'où une comparaison locale action par action).

Nous avons développé plusieurs critères de comparaison.
Tout d'abord une méthode locale permettant une comparaison En Ligne action par action a été développée.

Ensuite, afin d'effectuer une comparaison globale Hors Ligne, il paraît utile d'avoir un critère numérique simple permettant de juger globalement de la qualité de la supervision menée par un opérateur, humain ou artificiel. Mais il faut aller plus loin et proposer une méthode permettant de juger de la compatibilité globale entre le raisonnement mené par un opérateur artificiel et celui mené par un opérateur humain, grâce à la reconnaissance de l'intention associée à chaque action de l'opérateur humain. Il s'agit alors dans un premier temps d'étudier le comportement de l'agent artificiel afin d'associer des situations génériques (situation d'alarme, situation d'anticipation de débordement, situation d'accélération du processus, …) à des intentions. La comparaison consiste à voir dans quelle mesure une action de l'opérateur humain s'inscrit dans ce comportement, ce qui nécessite d'une part une mise en adéquation floue entre la situation concrète qui a amené l'opérateur humain à agir de telle façon et une situation générique, et d'autre part de vérifier si certaines conditions sont remplies, notamment l'appartenance de l'action de l'opérateur humain à l'ensemble des actions admissibles correspondant à l'intention associée à la situation générique précédemment mise en correspondance.

Des simulations informatiques ont été réalisées, d'une part sur les agents artificiels seuls afin de tester et améliorer leurs comportements, caler les paramètres de l'agent qualitatif et de certains cas, et enfin montrer qu'une coopération consiste à d'abord laisser agir l'agent à base de cas, qui possède des stratégies globales heuristiques à appliquer en priorité, puis si ce dernier n'a pas de proposition d'action, alors à donner la main à l'agent qualitatif, qui possède des stratégies locales optimisées afin de réduire une alarme ou d'ouvrir un chemin. D'autre part des tests ont été menés sur des sujets humains afin de comparer leurs raisonnements avec ceux d'agents artificiels. Ces tests ont montré que l'agent artificiel basé sur une coopération a des performances globales équivalentes à celles d'un sujet humain (sur les types de problèmes testés), et que les sujets humains, après un apprentissage de base, ont des comportements globalement équivalents à celui de cet agent.

Il est à noter que nos travaux reprennent ceux déjà effectués par [Trave-Massuyes *et al.* 99], [Ponsa et Catala 99 (a)] et [Ponsa *et al.* 02] : l'environnement de supervision (le micro-monde lié à la physique hydraulique) a donc été développé auparavant et nous l'avons simplement utilisé, et d'autre part l'agent artificiel basé sur le raisonnement qualitatif a lui aussi déjà été développé, et nous l'avons implémenté et testé. Nous avons donc choisi d'axer les travaux sur le développement de l'agent artificiel basé sur des cas.

Signalons pour finir que tout au long de ces travaux nous avons cherché à modéliser certains processus cognitifs humains, mais il s'agit dans tous les cas de simplifications car nous sommes loin de la finesse et la variété du raisonnement humain.

Le second chapitre de ce document détaille le point de vue des opérateurs (humain et artificiel), ainsi que l'environnement de supervision utilisé.
Les chapitres 3 à 6 précisent d'une part la base du raisonnement et d'autre part l'architecture informatique de chaque agent artificiel, c'est-à-dire l'agent à base de modèle, à base de cas, et à base de classification à partir d'exemples, respectivement.
Le chapitre 7 analyse la comparaison des raisonnements, ici sous la forme de séquences d'actions.
Le chapitre 8 est consacré aux simulations informatiques, et le chapitre 9 conclut sur les travaux futurs et les autres approches possibles en terme d'agents artificiels.

Chapitre 2

CONTEXTE DE L'ETUDE

Ce chapitre détaille l'environnement de supervision, ainsi que les problèmes posés par la définition du pas de temps de calcul de l'agent artificiel, puis présente quelques modèles de l'activité cognitive d'un opérateur humain en situation de supervision, justifie les trois modèles d'IA utilisés pour notre étude, et enfin présente les diverses méthodes de comparaison des raisonnements, ainsi qu'une définition de la coopération.

2.1. ENVIRONNEMENT DE SUPERVISION

2.1.1. Généralités

La base de notre étude repose sur un environnement de supervision. Nous allons donc commencer par donner les raisons qui ont motivé notre choix : plutôt que de placer l'opérateur dans un environnement industriel réel, par essence très complexe et donc non contrôlable, ce qui peut rendre très délicate la comparaison entre les raisonnements des opérateurs humain et artificiel, il semble préférable d'effectuer les expérimentations dans un environnement contrôlable. Dans ce dernier cas, il est possible d'analyser le raisonnement de l'opérateur par des observations directes (interview, analyse de tâche) et également par des critères numériques (temps de réaction, analyse statistique). Ces environnements contrôlables sont appelés simulateurs ou micro-mondes. Leur mise en œuvre est soumise à des contraintes : il est important de modéliser un environnement qui soit simple, reproductible, présente suffisamment de réalisme du point de vue industriel, fasse référence à des phénomènes communs à tout humain, et permette de mettre en avant différentes composantes du raisonnement humain dans des situations de supervision d'un système. Il est en effet important d'exhiber les comportements d'un humain lorsqu'il raisonne face à un système :

● dynamique (il doit s'adapter à une situation qui évolue),

● décomposable (suivant les cas, il faut concentrer son attention sur telle ou telle partie du système, sur tel ou tel ensemble de composants),

● et non linéaire.

En outre cet environnement expérimental doit posséder des caractéristiques de flexibilité, afin de modéliser divers types de situations et de processus, de maîtriser la complexité (quantité d'informations à prendre en compte, risque, incertitudes, temps contraint), et d'observer l'impact sur l'opérateur de modifications des paramètres de l'IHM (quantité et types d'informations affichées).

Ainsi, un micro-monde lié à la physique de l'hydraulique [Ponsa et Catala 99 (a)] a été développé avant le début de nos travaux et a déjà donné lieu à une étude de la part de P. Ponsa. Il est composé principalement de réservoirs, de conduites reliant les réservoirs et éventuellement de vannes sur ces conduites (voir 2.1.2.). Cet outil de simulation présente des caractéristiques intéressantes : d'une part, il s'agit bien de la supervision d'un système dynamique, décomposable, non-linéaire, d'autre part, cet outil présente des avantages de reproductibilité, flexibilité et simplicité. En effet, les concepts utilisés pour résoudre les problèmes associées à la supervision de ce procédé font partie de la culture générale commune : l'eau s'écoule sous l'effet de la gravité (sauf dans le cas d'une pompe) d'un réservoir haut vers un réservoir bas à travers la conduite les reliant. La propagation est contrainte uniquement par une éventuelle vanne binaire (ouverte ou bien fermée). Le micro-monde est donc contrôlé par l'opérateur qui agit sur les vannes.

2.1.2. Le simulateur de notre étude

Les détails de cet environnement sont présentés dans [Ponsa et Catala 99 (a)]. Les paramètres de supervision sont les suivants :

● Pour chaque réservoir $\boldsymbol{T_i}$, on donne ses dimensions (hauteur $\boldsymbol{H_i}$, largeur $\boldsymbol{L_i}$, profondeur $\boldsymbol{P_i}$), les cotes en x (axe horizontal) et z (axe vertical), et $\boldsymbol{V_i^{max}}$ sa capacité ($V_i^{max} = H_i.L_i.P_i$). En outre, à chaque instant t, on connaît sa hauteur d'eau $\boldsymbol{h_i(t)}$, et son volume d'eau $\boldsymbol{V_i(t)} = h_i(t).L_i.P_i$.
Une des consignes données à l'opérateur en début de session de test est d'éviter tout débordement (voir ci-dessous). Ainsi, la notion d'**alarme d'un réservoir** est présentée implicitement. Celle-ci correspond à une situation où le niveau d'eau est proche de la hauteur du réservoir (donc à une situation potentielle de débordement), qui peut être mise en œuvre de plusieurs façons : en terme de volume ou de hauteur d'eau, de manière fixe ou variable. En tout état de cause, elle est variable chez un humain (fonction du réservoir, de la situation) et il s'agit donc d'une zone plus ou moins floue. Pour les agents artificiels, nous avons utlisé un **seuil d'alarme fixe** $\boldsymbol{\alpha_i}$ définie arbitrairement par une valeur numérique (si le niveau d'eau d'un réservoir est supérieur à ce seuil α_i et croît au cours du temps, alors le réservoir est en alarme – voir 3.2.1.) :
$\boldsymbol{V_i^{max} - \alpha_i.L_i.P_i = \Delta V^{alarme}}$ avec $\boldsymbol{\Delta V^{alarme}}$ un paramètre du modèle représentant une marge en volume avant débordement. Il peut être réglé en fonction de l'opérateur humain auquel on compare l'agent qualitatif.
Une difficulté est donc d'établir une correspondance entre le seuil flou de l'humain et le seuil de l'agent artificiel.
En outre, les configurations de réseau envisagées présentent toute un réservoir "source" (dont la cote en z est la plus élevée) et un réservoir "puit" (dont la cote en z est la plus faible).

● Pour chaque conduite, on donne son diamètre, sa longueur. Eventuellement, on peut placer une vanne $\boldsymbol{V_{ij}}$ sur une conduite reliant les deux réservoirs T_i et T_j. Les vannes ne prennent que deux positions : ouvertes (***ON***) ou bien fermées (***OFF***). Les vannes sont donc les moyens d'action de l'opérateur sur le système.

● Les hypothèses sont :

- deux réservoirs peuvent être reliés par au plus une conduite.
- le réservoir "source" et le réservoir "puit" ont la même capacité.
- en début de test, tous les réservoirs sont vides sauf le réservoir "source".

● Il est également possible d'ajouter divers éléments hydrauliques tels que des pompes introduisant un flux de retour, des apports aléatoires de fluide dans les réservoirs, et des fuites aléatoires au niveau des réservoirs ou des conduites, ce qui permet ainsi de construire des micro-mondes suivant des degrés de difficultés divers.

● Un opérateur passe un test de supervision sur ce simulateur, et une ***consigne*** est donnée en début de session. Celle-ci reflète les buts à optimiser et les contraintes. Par exemple, une instruction peut être : *transférer l'eau du réservoir source vers le réservoir puit en un minimum de temps et en évitant les débordements des réservoirs intermédiaires, en agissant sur les vannes binaires*.
Il est à noter que dans ce cas, les deux buts (accélérer le processus et éviter les débordements) sont en conflit (ouvrir une vanne pour accélérer le processus va accroître le risque de débordement) et nécessite un compromis au niveau des décisions.

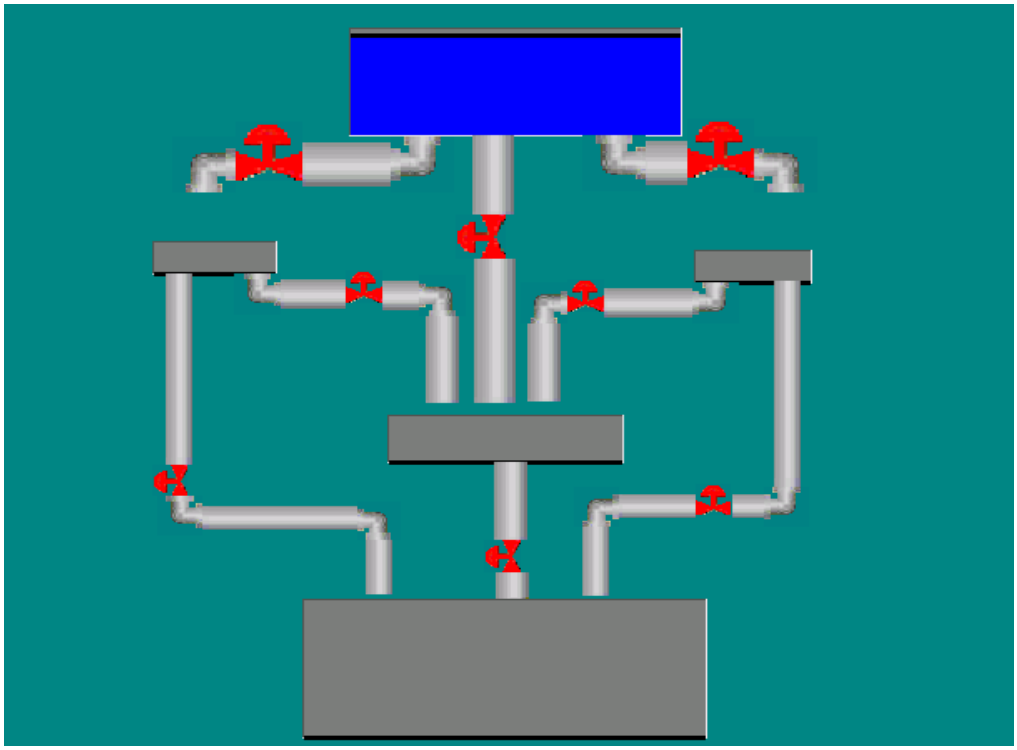
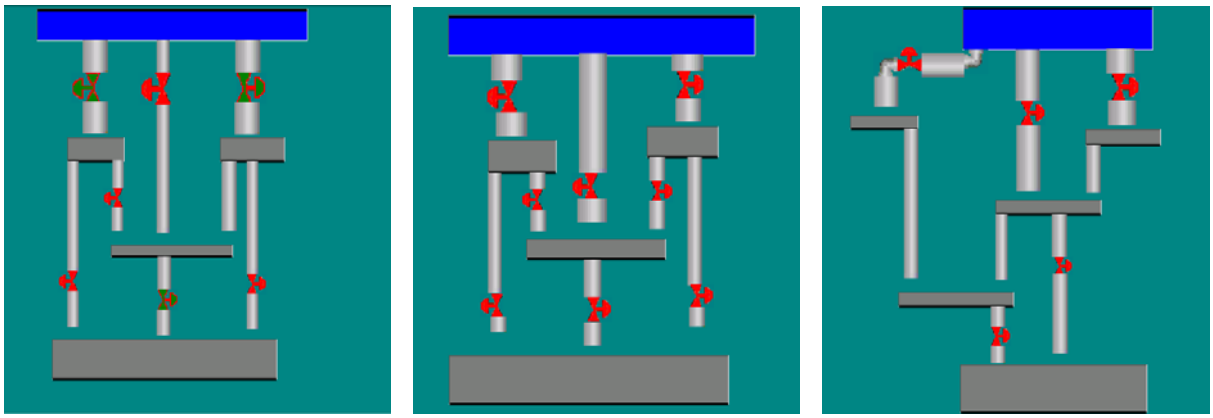


Figure 1 : Un modèle de l'environnement de supervision

● Précision du modèle : le modèle mathématique sous-jacent est approximatif : une approximation légitime est de négliger la turbulence et de supposer le fluide parfait. Une autre approximation est de négliger les pertes de charge singulières (au niveau des jonctions réservoir → conduite, au niveau des coudes et au niveau des vannes) et linéaires (le long des conduites). En effet, d'une part les distances mises en jeu dans le cadre de notre étude sont faibles (quelques mètres), et d'autre part la prise en compte des pertes de charge complique nettement la résolution et la précision des calculs n'est pas le but recherché.

● Les tests informatiques concernant tous les types d'agents artificiels (qualitatifs, à base de cas, …) ont été réalisés sur les configurations de réseau de la figure 2. Certaines configurations demandent la gestion d'une (ou plusieurs) conduite sans vanne (configurations 1, 2 et 6), certaines configurations sont identiques excepté dans les dimensions des réservoirs

et conduites (configurations 1 et 2 d'une part, 3, 4 et 5 d'autre part) afin de faire varier la vitesse de vidange et donc la gestion des situations d'alarme.



Configurations 1 et 2 *Configurations 3, 4 et 5* *Configuration 6*

Figure 2 : Configurations testées

2.1.3. Pas de temps de calcul des agents artificiels

Il est à noter qu'un problème se pose pour tous les agents artificiels, celui de la définition du pas de temps de calcul Δt qui peut être défini comme la somme de la durée d'observation et de réflexion mêlées, et de la durée d'exécution d'une action. Ce Δt est très variable chez un humain d'une part, et d'un humain à un autre d'autre part. Dans le cadre de la supervision, on peut imaginer que l'observation d'une situation d'urgence constitue un point de focus et il faut agir vite (réduction du pas de temps), quitte à abandonner (momentanément ou définitivement) une stratégie qui était en cours d'élaboration. S'il n'y a pas d'urgence, alors l'opérateur a le temps d'élaborer une ou plusieurs stratégies de façon poussée (augmentation du pas de temps), ce qui peut ensuite aboutir à un plan et donc à une série d'actions enchaînées rapidement (réduction du pas de temps). Ces aspects dynamiques ont bien été mis en évidence par les modèles cognitifs de l'opérateur humain de supervision (voir notamment [Amalberti et Cacciabue 90] et [Hoc 98]).

Exposons tout d'abord le travail réalisé par Allen Newell [Newell 90] sur ce thème.

Echelle (sec)	Unité de temps	Système	Théorie
10^{7}	Mois		Bande sociale
10^{6}	Semaine		
10^{5}	Jour		
10^{4}	Heure	Tâche	Bande rationnelle
10^{3}	10 mn	Tâche	
10^{2}	Minute	Tâche	
10	10 sec	Tâche	Bande cognitive
1	1 sec	Opération	
10^{-1}	100 ms	Opération délibérée	
10^{-2}	10 ms	Circuit neuronal	Bande biologique
10^{-3}	1 ms	Neurone	
10^{-4}	100 µs	Organelle	

Tableau 1 : Echelle de temps d'une action humaine selon Newell [Newell 90]

Selon A. Newell, une action humaine fait appel à différents niveaux de traitement imbriqués les uns dans les autres. Le tableau 1 résume ces différents niveaux. Nous allons nous intéresser ici à la bande cognitive :

● la délibération est liée au fait d'accéder à des connaissances, de les assembler et de choisir une opération élémentaire, c'est-à-dire que le système est engagé dans un processus de recherche de connaissance en mémoire, mais pas dans un processus de recherche d'une solution à un problème. Ce niveau est qualifié d'automatique, non conscient.

● le niveau des opérations simples est lié au processus de recherche de solution à un problème. Cette réponse est composée d'une collection d'opérations délibérées.

● le niveau des opérations composées est lié à la recherche d'une solution dans l'espace d'un problème général. La réponse est composée d'une collection d'opérations simples. La durée moyenne est de 10 secondes, mais cela peut aller de 2 à 30 secondes.

La bande rationnelle correspond à la satisfaction des buts, mais sans regarder comment les processus internes accomplissent le lien entre les actions et le but.

Revenons maintenant à l'application qui nous intéresse. Deux possibilités s'offrent à nous : définir soit un pas de temps variable en fonction de la situation, soit un pas de temps fixe.

● Un pas de temps variable pourrait être mis en œuvre de la façon suivante : le pas de temps peut être court (Δt_1 de l'ordre de 1 seconde) dans un cas sans alarme car la réflexion pour mettre en place une stratégie dans ce cas est relativement simple. En cas d'alarme, deux sous-cas se présentent : si le niveau d'alarme est faible (on est assez loin du débordement) et si la vitesse de montée est plutôt lente, le pas de temps peut être plus long (Δt_2 de l'ordre de 3 secondes) car l'opérateur se donne un délai de réflexion pour mettre en place une stratégie, puis éventuellement enchaînement des actions correspondantes selon un pas de temps réduit Δt_1. Si le niveau d'alarme est élevé (on est proche du débordement) ou si la vitesse de montée est rapide, alors le pas de temps est court (Δt_1) car il faut trouver rapidement une action afin d'éviter le débordement.
Ainsi, la mise en place d'un pas de temps variable nécessite de prendre en compte la vitesse de montée de l'eau dans chaque réservoir, et éventuellement plusieurs niveaux d'alarme, afin d'une part de donner à l'opérateur des délais de réflexion si possible, et d'autre part de l'obliger à agir vite si la situation l'exige.
On pourrait même imaginer que les valeurs Δt_i soient fonction de la configuration du micro-monde, c'est-à-dire fonction de la surface horizontale et de la hauteur maximale de chaque réservoir, ainsi que des diamètres des conduites.

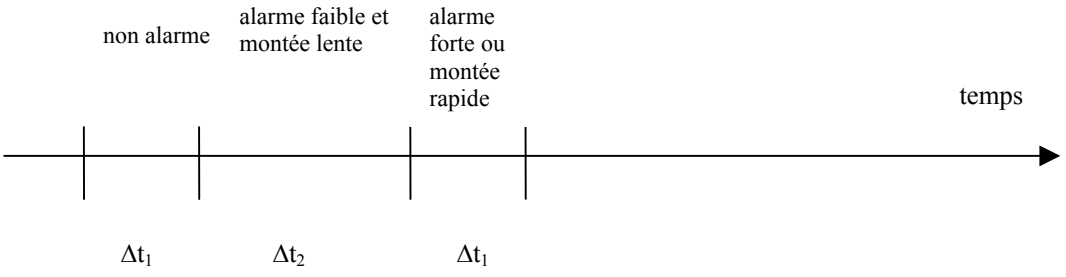


Figure 3 : Pas de temps de calcul variable

Les différents instants représentés (traits verticaux) correspondent à des actions de l'opérateur.

● Mais de manière plus simple, on peut imaginer un pas de calcul fixe et moyen (Δt de l'ordre de 2 secondes). Mais ce pas de temps doit tout de même être adapté pour chaque configuration du réseau, car il est évident que si la vidange des réservoirs est relativement lente, un pas de temps de 3 secondes est suffisant car la situation évolue lentement, alors que lors d'une vidange rapide des réservoirs, il vaut mieux réduire ce pas de temps à 1 seconde car la situation évolue rapidement. Ceci correspond d'ailleurs bien à une caractéristique d'un opérateur humain qui adapte sa durée de réflexion en fonction de l'évolution de la situation. Nous avons choisi d'appliquer cette stratégie, où le pas de temps des agents artificiels est fixé en début de simulation. Les tests informatiques effectués avec divers pas de temps justifient l'adaptation du pas de temps selon la configuration du réseau.
Par ailleurs [Ponsa *et al.* 00], lors de tests effectués sur une configuration donnée d'un réseau, utilise un pas de temps de 0.1 seconde, alors que l'opérateur humain effectuant ce test a un pas de temps variant entre 1.1 et 1.7 secondes. Ce pas de temps de 0.1 seconde ne paraît pas réaliste.
Un travail futur pourrait consister à étudier la mise en place d'un pas de temps variable.

2.2. POINT DE VUE DE L'OPERATEUR HUMAIN

2.2.1. Généralités

Face à la supervision d'un système dynamique par un opérateur humain, divers processus cognitifs sont mis en avant. Ces différentes facettes incluent :

● la perception. Tout d'abord, concernant la détection d'un signal, Moray indique une théorie basée sur un modèle quantitatif à deux paramètres [Moray 97] : le rapport signal / bruit (le bruit dans le canal de communication peut masquer la présence d'un signal et donc limiter la capacité à le détecter) et le critère de décision qui représente un certain niveau d'excitation à partir duquel la présence d'un signal est détectée (mais le bruit peut entraîner un dépassement de ce seuil et donc une fausse détection). Ensuite, l'attention visuelle est contrainte par les mouvements des yeux. Le champ périphérique visuel détecte les changements ou mouvements, ce qui engendre par réflexe une tentative de suivi ou de fixation de la source de changement. Les mouvements des yeux semblent être limités à deux par seconde, sans jamais excéder quatre par seconde. Mais la perception précise nécessite une fixation de la source d'information. En outre, l'opérateur suit visuellement certaines variables au détriment d'autres, en fonction de ses points de focus (lié à l'attention sélective). D'autre part il a une certaine habileté visuelle pour juger par exemple la qualité d'un matériau à partir de sa texture, sa couleur, sa surface, … Enfin, il mesure les variables du processus non pas avec des valeurs d'échelle numérique de manière absolue, mais en terme de valeurs relatives [Bainbridge 78].

● la gestion du temps intervient à plusieurs niveaux. L'opérateur humain possède des connaissances temporelles sur le fonctionnement du processus et sur son propre fonctionnement (méta-connaissances). Il détermine une fenêtre spatio-temporelle adéquate (le champ de supervision) dont l'étendue lui permet d'agir non seulement sur les conséquences mais aussi sur les causes des phénomènes [Hoc 98]. En outre, aux actions de l'opérateur sont liés des délais de réponse. Tous ces éléments impliquent le développement d'activités

d'anticipation et de planification. Par ailleurs, dans la coopération homme-machine, deux dynamiques doivent être synchronisées : celle du déroulement du processus avec les dates-limites des actions, et celle du déroulement des processus cognitifs. L'opérateur gère donc un compromis entre un niveau de compréhension suffisant pour agir le mieux possible, et ses contraintes temporelles de décision. Aussi, avec l'expérience, il développe des automatismes et des procédures pour diminuer la charge mentale [Rasmussen 83].

● l'anticipation : une caractéristique de l'opérateur humain est en effet de toujours chercher à prédire ce qui va se passer, trouver les effets de ses actions possibles.

● l'explication : en cas d'échec de prédiction [Schank 99], l'opérateur humain cherche à diagnostiquer le problème, trouver les causes possibles de l'événement anormal. En outre, dans une situation dynamique, qui continue à évoluer pendant le déroulement du diagnostic, et qui comporte des délais de réponse, il s'agit de diagnostiquer un état futur sur lequel l'opérateur peut agir maintenant, sans qu'il ne dispose encore de toutes les informations. Il faut donc gérer un compromis entre une action inconsidérée trop rapide et une action bien justifiée mais lancée trop tardivement. Par ailleurs les stratégies de diagnostic et de pronostic amènent en permanence à tester des hypothèses, élaborées dans le cadre d'une mise à jour constante de la compréhension de l'évolution de la situation, ce qui permet de gérer un compromis entre le niveau de risque et le niveau de performance [Hoc 98].

● les buts : une caractéristique importante du raisonnement est d'être constamment orienté but (toute action, réflexion est liée à un but à satisfaire) (voir 6.1.1.2.1. Intentions et buts).

● les plans : de quelle façon, c'est-à-dire au moyen de quelles actions l'opérateur compte-t-il atteindre un but ou un sous-but en fonction des contraintes existantes et de l'état du système ? La tâche à réaliser peut être vue comme une séquence linéaire, voire un arbre de buts et sous-buts, afin de tenir compte de l'aspect conditionnel du raisonnement et de la composition des connaissances chez l'humain.

● la prise de décision. Cela concerne un choix parmi plusieurs buts, plans, actions possibles, notamment lors de situations nouvelles (adaptation), lors de situations de danger (point de focus, décision en temps contraint), et en cas d'erreur (réparation, retour-arrière).

● la non-linéarité du raisonnement : l'opérateur réfléchit sur un point, puis sur un autre, puis revient au premier point, il déduit des informations à partir de l'état courant (chaînage avant) et à partir des buts (chaînage arrière). La gestion d'un compromis entre compréhension et action et la non-séquentialisation des tâches de contrôle sont deux sources de partage de temps entre tâches. Ainsi, à une étape de la réalisation d'une tâche qui nécessite un contrôle attentionnel, une autre tâche peut devoir être interrompue, tout en conservant en mémoire son point d'interruption [Hoc 98].

● les méta-connaissances. Elles sont liées à l'expertise de l'opérateur et jouent un rôle crucial dans le choix délibéré d'une stratégie, dans l'adaptation à des situations nouvelles, et dans le contrôle des actions. Il s'agit également des connaissances sur son propre fonctionnement (capacités d'exécution, rapidité d'exécution, limitations, risques de commettre une erreur, d'être en surcharge mentale, de ne plus comprendre, …). Pitrat fait une étude détaillée des méta-connaissances pour les systèmes experts dans [Pitrat 93]. Des méta-connaissances très utilisées lors du contrôle de processus sont les préférences [Amalberti et Cacciabue 90], qui servent à réduire les risques pris et l'effort physique et mental, tout en fournissant une réponse

satisfaisante en terme de performance, et respectant les diverses contraintes. Amalberti et Cacciabue proposent un modèle des méta-connaissances en trois niveaux pour le contrôle de processus [Amalberti et Cacciabue 90] : le premier niveau concerne les heuristiques générales (indépendantes du domaine) de management des ressources, qui sont relatives aux limitations mentales et cherchent donc à conserver les processus mentaux dans des limites acceptables de façon à pouvoir exécuter des actions et gagner du temps pour des activités supplémentaires. Des exemples sont : éviter les situations dans lesquelles on n'est pas expérimenté, encourager les situations dans lesquelles les réponses sont efficaces, réduire l'ambition du but avant d'être surchargé, préserver le maximum de temps libre, continuer la procédure normale aussi longtemps que possible. Les auteurs ont défini trois types d'heuristiques générales, liées au principe d'économie (préférer les procédures bien maîtrisées même si elles ne sont pas les meilleures, maintenir au mieux la sérialité des procédures), à la gestion du temps (lors d'une situation bien comprise donner la priorité à la convergence vers une position stable à court terme afin de gagner du temps pour le raisonnement à long terme, gérer la situation immédiate de façon à avoir au moins une réponse satisfaisante à fournir aux événements les plus probables à venir, lorsque la pression liée au temps augmente préférer l'action au raisonnement) et au principe du contrôle (vérifier les résultats attendus d'une action, vérifier périodiquement les paramètres qui ne sont pas liés directement au processus en cours). Le second niveau reflète l'application de ces heuristiques générales à un domaine spécifique. Il s'agit des préférences. Le troisième niveau concerne les procédures spécifiques (scripts d'exécution) du domaine, contrôlées par les préférences (second niveau).

2.2.2. Modèles classiques de l'activité cognitive d'un opérateur de supervision

Nous allons présenter succinctement quelques approches classiques permettant de modéliser l'activité cognitive d'un opérateur de supervision. Les trois premiers modèles sont généraux et simplifiés. Le quatrième détaille l'activité de contrôle dynamique de processus, et le cinquième tente de mieux formaliser le savoir-faire de l'opérateur. Les modèles suivants sont davantage basés sur l'activité cognitive : l'approche KADS tente de modéliser l'expertise quel que soit le domaine, et l'approche de Rasmussen, spécifique à l'activité de supervision, modélise de façon précise le processus de prise de décision et insiste sur une représentation des connaissances orientées but, est donc particulièrement utile dans le cadre de nos travaux. Divers travaux ultérieurs ont tenté d'améliorer ce modèle. Nous en présentons trois.

2.2.2.1. Classification de Hayes-Roth

Il s'agit d'une classification simple des activités cognitives de base, ainsi que d'activités cognitives plus complexes définies de manière composite à partir d'activités de base [Hayes-Roth *et al.* 83] [Salazar-Ferrer 95].

Interprétation	Inférer une situation à partir de données (capteurs, …)
Prédiction	Inférer les conséquences probables d'une situation
Diagnostic	Inférer les pannes d'un système à partir de données d'observation
Conception	Configurer des objets sous contraintes
Planification	Concevoir, configurer des actions
Monitoring	Comparer des observations à des vulnérabilités attendues
Déboggage	Prescrire des remèdes et réponses adaptées à des défauts et dysfonctionnements d'un système
Réparation	Exécuter un plan pour appliquer un remède prescrit à la suite d'un dysfonctionnement d'un système
Contrôle	Interpréter, prédire, réparer et assurer le monitoring du comportement d'un système

Tableau 2 : Classification des activités cognitives de supervision

2.2.2.2. L'activité cognitive comme classification heuristique

Ce modèle distingue deux grands types d'activités, conduisant soit à la synthèse, soit à l'analyse d'un système [Clancey 85] [Salazar-Ferrer 95]. Le schéma ci-dessous montre les opérations génériques conduisant à la synthèse ou à l'analyse d'un système. On notera également des points communs avec la classification précédente, mais celle-ci est plus détaillée et indique clairement les liens entre les activités cognitives.
En outre, toutes les activités cognitives sont une instanciation d'une activité générique de classification heuristique (voir aussi 6.2), consistant-en :

● une abstraction (qualitative ou par généralisation) des données,

● une mise en correspondance heuristique vers une solution abstraite (par exemple une classe de causes de défaillances),

● un raffinement, une spécialisation de la solution.

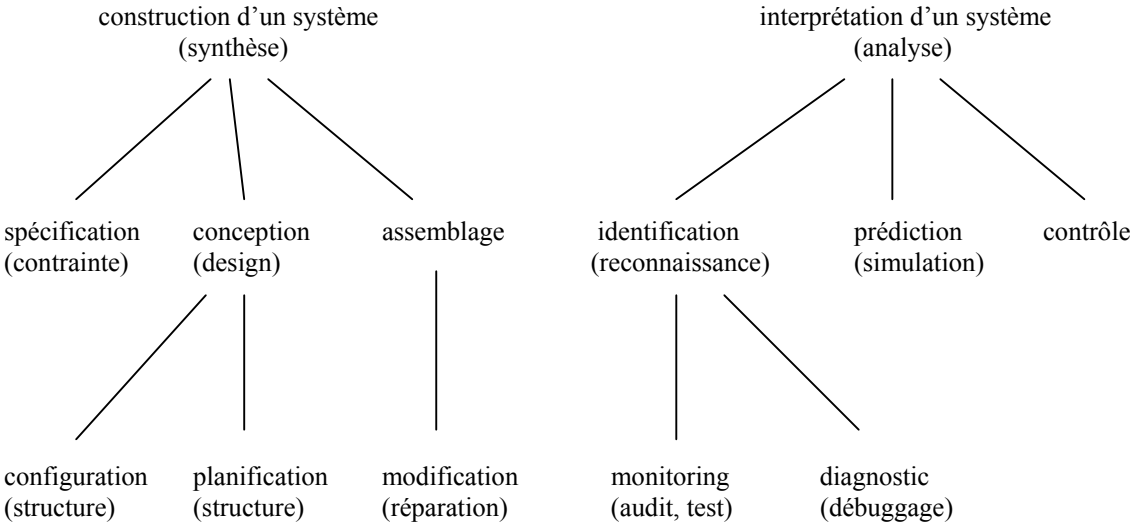


Figure 4 : Classification des activités cognitives de supervision

2.2.2.3. L'approche des tâches génériques

L'approche des tâches génériques [Chandrasekaran *et al.* 89] [Salazar-Ferrer 95] consiste à décomposer les activités cognitives en un petit nombre de tâches génériques simples :

● la classification hiérarchique qui intervient lorsque le domaine de connaissance peut être représenté par une hiérarchie (par exemple, dans une tâche de diagnostic les nœuds représentent les causes, et il s'agit d'apparier des symptômes avec un nœud de l'arbre)

● l'évaluation d'hypothèse intervient lors de la mise en correspondance au sein de la hiérarchie. Il s'agit d'évaluer l'adéquation entre un nœud et des données d'observation, par exemple à l'aide d'une hiérarchie structurant la connaissance sur les données portées par un nœud hypothèse, la racine de la hiérarchie définissant un degré de vraisemblance de l'hypothèse.

● l'assemblage d'hypothèses par abduction. Le but est de générer des hypothèses rendant compte le mieux possible des données, puis de les assembler afin de construire une nouvelle hypothèse composite. Il s'agit donc d'une modélisation du raisonnement abductif, notamment pour le diagnostic de causes apparemment multiples qu'il convient d'intégrer dans une hypothèse composite.

● l'abstraction d'état qui a pour but de prédire les conséquences d'une action ou d'une perturbation sur un système. Il s'agit d'une simulation causale consistant à propager des effets au sein d'un modèle hiérarchique causal de fonctionnement comportant plusieurs niveaux de description/généralité.

● la propagation dirigée d'information, pour rendre compte d'activités non liées à une classification. Par exemple dans le cas du diagnostic, il s'agit d'accéder à une hypothèse indépendamment d'une hiérarchie de causes, au moyen de liens spécifiques.

● la synthèse d'objets par sélection de plan. Lors d'une activité de conception ou de planification, si la structure à concevoir est connue à un certain niveau d'abstraction, l'utilisation d'une connaissance sous la forme d'une hiérarchie permet de préciser les caractéristiques de l'objet à concevoir ou des actions à planifier.

2.2.2.4. Le contrôle dynamique

L'opérateur doit identifier l'état actuel du processus, juger s'il est acceptable, et sinon choisir et introduire un changement dans les commandes de conduite. L'approche développée par Bainbridge est la suivante [Bainbridge 78] :

● La description de la tâche est réalisée grâce à la modélisation des variables impliquées dans la qualité de la production et leurs interactions, sous la forme d'un graphe de fluence.

● Par ses connaissances de la dynamique et du comportement du processus, l'opérateur est capable de prévoir l'effet sur la sortie d'un réglage d'une valeur donnée d'une commande pour un temps donné. Autrement dit, il a une connaissance de l'amplitude et la chronologie des effets dans le processus. En outre, la connaissance de la dynamique du processus n'est pas continue mais divisée en un nombre limité de catégories, si bien que l'opérateur réagit de la même manière à un ensemble de comportements. La représentation de l'effet des actions est en général basée également sur des catégories. Les informations concernant l'état présent, le comportement prévu et les actions sont incorporées au sein de l'image mentale de l'opérateur. Cette image ne contient pas les informations brutes, mais des informations traitées sous une forme utile en terme de but ou d'action impliquée : telle valeur de la variable est-elle bonne, mauvaise, nécessite-t-elle une action urgente ? Des groupements de variables (par exemple celles qui sont dans le même état) peuvent également favoriser la prise de décision.

● Un point important souligné par l'auteur est l'échantillonnage des signaux correspondant aux variables suivies. La connaissance et l'expérience concernant la chronologie, l'amplitude des variations, les délais de réponse, permettent à l'opérateur d'échantillonner moins souvent certains signaux, ce qui lui permet notamment de gagner du temps pour faire d'autres tâches, puisqu'il n'a pas à surveiller ces signaux. Cependant il doit augmenter la fréquence d'échantillonnage si la sortie est proche des limites de tolérance, ou si sa prévision atteint un certain degré d'incertitude, ou encore s'il anticipe un événement critique. Dans un processus à

variables multiples, il doit répartir son attention entre les variables. Le taux d'échantillonnage d'une variable peut varier avec son taux de perturbation, son importance, son évolution récente et l'état des variables avec lesquelles elle est en relation. Les attentes de l'opérateur lui permettent de savoir si la valeur signalée représente vraiment la valeur du processus. Les coûts d'échantillonnage des informations (et de l'action impliquée) ont une influence sur le choix d'une stratégie, à cause de la tendance humaine à minimiser l'effort mental (et physique). Dans des conditions de surcharge, l'opérateur se concentre sur le plus important et n'échantillonne pas, ce qui peut d'ailleurs encore augmenter encore son stress. Les sous-tâches demandant le plus d'effort seraient les premières à être négligées, à moins qu'elles ne soient très importantes.

● Pour finir, l'auteur distingue le contrôle dynamique (qui concerne le choix entre des réglages quantitativement différents d'une même variable de commande), et la décision de planification relative à la manière d'utiliser les moyens disponibles (qui concerne des alternatives de natures différentes). Dans ce cas, l'opérateur partage son temps entre le contrôle et la vérification de l'état présent du processus et l'élaboration de la meilleure séquence d'activités. La liste d'activités à venir comporte les moments de contrôle et de vérification de l'état, la séquence d'actions choisies, et la mise à jour de cette liste. Il s'agit donc d'un comportement récursif car la mise à jour de la liste est une des activités de la liste, et la décision à un instant *t* dépend de la liste établie à *t – 1*, cette liste pouvant être mise à jour à l'instant *t* en fonction de nouvelles informations.

2.2.2.5. Modélisation du savoir-faire de l'opérateur

L'approche de Amalberti tente de définir les relations des connaissances à l'action, précisant l'idée selon laquelle les connaissances pratiques se développent sur la base d'un acquis préalable [Amalberti 91].

● Trois domaines de connaissances théoriques sur *les outils à manipuler*, *la tâche à réaliser*, et le système global (par exemple *les règles concernant l'organisation d'une profession*) se complètent mutuellement pour participer à la formation de connaissances fonctionnelles (nommés *schémas ou modèles mentaux* dans l'article) qui permettent d'agir en situation réelle et constituent le savoir-faire de l'opérateur.

● Chacun de ces domaines de connaissances théoriques est à son tour scindé en trois catégories : *les connaissances descriptives*, *les connaissances d'usage pour l'action* (ou micro-procédures locales), et *les connaissances de consigne pour l'action* qui fournissent des instructions pour utiliser les *connaissances d'usage* précédentes, et qui sont sensibles à l'expérience, à des heuristiques générales de savoir-faire, et à des croyances.

● En outre, les raisons qui justifient ces *connaissances de consigne* sont liées à l'existence de *connaissances d'Univers*, sous la forme de connaissances générales (profondes) et de méta-connaissances. Ces *connaissances d'Univers* interviennent dans les situations nouvelles, inattendues, mais également dans l'apprentissage et la création de nouveaux *schémas* procéduraux en triant les connaissances effectivement utilisées sur l'outil, la tâche et l'organisation. Autrement dit, l'opérateur fait appel à des connaissances générales sur lesquelles il greffe des connaissances spécifiques sur le système, et par essai-erreur, avec l'entraînement, sélectionne les connaissances qui servent à automatiser de plus en plus la conduite.

L'entraînement est lié à la fréquence d'utilisation. Plus la fréquence est grande, plus la procédure est disponible en mémoire, en lien avec l'heuristique générale : première hypothèse évoquée, hypothèse préférée.
Un aspect du savoir-faire et de l'apprentissage est l'adaptation aux contraintes spécifiques du travail, notamment les contraintes temporelles sur la réalisation de la tâche. Une heuristique générale est : dans une situation où la pression du temps est importante, gérer le court terme avant le long terme.
Un autre aspect lié au temps est le choix d'une procédure (*schéma*) en fonction du temps disponible. Une heuristique générale est d'éliminer les procédures trop longues par rapport au temps disponible même si elles sont meilleures.
Une heuristique encore plus générale est : plutôt que de chercher la solution optimale, commencer par éliminer les solutions impossibles, du fait des éléments présents et de contraintes externes (temps, consignes, …).
Concernant l'optimalité, une stratégie générale est : si on a le choix, alors préférer agir vite par reconnaissance de *schémas* plutôt que différer son action pour mieux la calculer et l'optimiser.
Lors de situations conflictuelles, où l'opérateur a des difficultés à associer un *schéma* aux indices externes, du fait de la présence de discordances, celui-ci utiliserait une heuristique générale du type : si on ne comprend pas, alors il s'agit d'un cas rare.
Ainsi, les heuristiques générales de savoir-faire réguleraient la mise en place et l'emploi des connaissances théoriques et des heuristiques locales sur un domaine, et prendraient le relais des heuristiques locales dans une situation conflictuelle ou inhabituelle. Ces heuristiques générales seraient acquises à travers les expériences de la vie quotidienne, par analogie inter-domaines.

● Les *schémas* permettent d'agir concrètement. Ils sont issus d'heuristiques générales appliquées au domaine et d'heuristiques locales au domaine (issues de la sélection des connaissances théoriques du domaine et de la pratique). Ils sont très puissants pour traiter, même sous forte contrainte temporelle, toutes les variantes des situations connues. Ils sont peu accessibles à l'introspection, incomplets, non optimaux (mais permettent en contrepartie d'économiser des efforts mentaux), très personnels, instables (oubli rapide si non utilisés) et mal délimités. Pour réaliser une action, il n'y aurait pas un *schéma* mais plusieurs *schémas* correspondant à différents objectifs reliés hiérarchiquement du plus ambitieux (optimalité) au moins risqué (assurer le minimum de résultat en préservant soi-même et le matériel).

2.2.2.6. La modélisation de l'expertise selon la méthode KADS

La modélisation de la connaissance et de l'activité d'expertise de la méthode *KADS* (*Knowledge Acquisition Design Support*) [Wielinga *et al.* 92] [Salazar-Ferrer 95] est fondée sur quatre niveaux :

● la connaissance du domaine, modélisée par des concepts contenant des propriétés auxquelles correspondent des valeurs, des relations entre concepts (est un, partie de, …), des relations entre expressions (relation "=" entre une propriété et une valeur, relation causale, …). Il est à noter que la relation "partie de" est très importante car elle permet de représenter le système à différents niveaux d'abstraction, en fonction du but, en regroupant plusieurs composants en un seul objet ayant des fonctions et propriétés [Moray 97].

● l'inférence élémentaire est une tâche élémentaire prédéfinie (par exemple instancier, classifier, sélectionner, décomposer, comparer, généraliser, …) qui s'applique à une classe d'objets du domaine qui partagent le même rôle vis-à-vis de l'inférence.

● les tâches et sous-tâches représentent des activités de résolution de problèmes et liées à l'exécution d'un but. Elles décrivent le contrôle sur des inférences par le biais de conditions, boucles, …

● éventuellement les stratégies globales de résolution de problème.

2.2.2.7. La modélisation cognitive de l'opérateur en situation de contrôle de procédé

L'approche de Rasmussen est une modélisation spécifique de l'opérateur en situation de contrôle de procédé, qui peut être résumée de la manière suivante [Hoc 98] : l'opérateur réagit aux alarmes, recherche des informations complémentaires, émet une hypothèse, interprète et évalue la situation en fonction des objectifs du système, définit une tâche, adopte une procédure et l'exécute, jusqu'à ce qu'une nouvelle alarme l'engage à nouveau dans ce cycle. Mais de nombreux courts-circuits de ce cycle permettent d'aboutir à une décision sans analyse approfondie de la situation quand cette dernière est familière.

De façon plus détaillée, cette approche comporte quatre points principaux [Rasmussen 83] [Salazar-Ferrer 95] :

● les activités de l'opérateur sont classifiées selon trois niveaux :

- le niveau des activités effectuées de façon automatisée, inconsciente, en situation très familière, routinière.

- le niveau des activités basées sur des procédures apprises, effectuées de façon consciente, lors de situations anormales connues.

- et le niveau des activités basées sur les connaissances générales, effectuées de façon consciente, lors de situations anormales inconnues.

● le modèle de prise de décision. L'activité globale est décomposée en une séquence type d'activités plus simples se répétant dans le temps :

- détection d'un besoin d'intervention (alerte)

- observation des données

- identification de l'état courant du système

- évaluation des conséquences attendues

- définition d'un état à atteindre

- planification d'une tâche

- spécification d'une procédure d'action

- exécution de la procédure d'action

Des raccourcis sont possibles au sein de cette séquence. La linéarité peut être rompue (aller-retour, coopération entre deux activités)

● le modèle de l'activité de diagnostic sous la forme de deux types de stratégies de recherche de cause :

- la recherche topographique qui fait intervenir un modèle de fonctionnement normal de chaque composant du système,

- la recherche par symptômes fondée sur un modèle du système défaillant, et agissant selon trois niveaux : la recherche d'hypothèse par reconnaissance de forme dans une situation de routine ou familière, la recherche d'hypothèse par table de décision (règles heuristiques, appariement entre l'état du système et des états de panne), et la recherche par génération d'hypothèse et test (génération d'une hypothèse de panne, construction d'un modèle de fonctionnement défaillant, prédiction de l'effet de la panne, comparaison avec les données d'observation).

● la représentation du domaine selon les buts de l'opérateur. Il s'agit de relier la décomposition du système au niveau de sa structure (relation est une partie de) à la décomposition fonctionnelle du système (processus physiques et chimiques, boucles de régulation, flux d'énergie, de matière, d'information), et d'interpréter cette décomposition fonctionnelle en terme d'une hiérarchie ou d'un réseau de buts/contraintes et sous-buts.

Limitations du modèle de Rasmussen

Ce modèle comporte aussi des inconvénients [Hoc 98] : l'opérateur est essentiellement réactif, dépourvu de capacités d'anticipation. La compréhension de l'état ou de l'évolution du système permettrait en effet d'anticiper des alarmes. Le processus d'attention sélective n'est pas mis en évidence : cela lui permettrait de se focaliser sur certaines informations et le rendrait plus enclin à formuler certaines hypothèses au détriment d'autres. En outre, l'évolution du processus pendant le déroulement du diagnostic et de la prise de décision n'est pas pris en compte.
Pour pallier ces inconvénients, Hoc et Amalberti [Hoc 98] ont développé un modèle parallèle de gestion de situation dynamique nommé *GSD*, par opposition au modèle séquentiel de Rasmussen, Cacciabue a développé le modèle de prise de décision à deux niveaux nommé *COSIMO* [Cacciabue *et al.* 90], et Hollnagel a développé un modèle de contrôle contextuel nommé *COCOM* [Hollnagel 93].

2.2.2.8. Architecture de Gestion de Situation Dynamique (GSD)

La psychologie a coutume d'opposer les processus contrôlés, c'est-à-dire qui relèvent du contrôle attentionnel séquentiel, des processus automatiques, qui autorisent un large parallélisme grâce au recours à des ressources différentes.

● L'architecture proposée par Hoc et Amalberti [Hoc 98] étend le modèle de Rasmussen. Ils introduisent la notion de *représentation occurrente de la situation*, qui est à la base des processus de diagnostic et de prise de décision, est liée au contrôle attentionnel, et est basée sur des connaissances spécifiques à la situation. Cette représentation trouve sa place entre les

automatismes et les connaissances générales définies dans le modèle de Rasmussen. Elle porte sur l'état et l'évolution du processus, sur les plans d'action, la représentation des risques et les données nécessaires à la gestion des ressources. Elle soutient donc les prévisions. Sa mise à jour correspond aux activités de diagnostic, et plus généralement de compréhension.

● Par ailleurs, la rétroaction adaptative est au centre de l'architecture, par l'intermédiaire de trois boucles. Une boucle à court terme permet au contrôle automatique de s'exercer. Il s'agit du contrôle rapproché du processus, qui a toujours la priorité, et qui est mis en place dans le cadre d'une représentation occurrente. Il peut en outre envoyer au contrôle attentionnel des indices de désalignement entre la compréhension et l'action, déclenchant un processus au niveau attentionnel. Une boucle à moyen terme permet d'ajuster la représentation occurrente à la situation, par exemple en corrigeant le plan. Cet ajustement pouvant prendre du temps, cette replanification peut s'effectuer en parallèle à la boucle réactive (à court terme), avant de réorienter cette dernière vers une nouvelle enveloppe plus cohérente. Enfin, certains désalignements entre la situation et la représentation occurrente peuvent nécessiter une remise en cause de cette dernière, afin de produire un nouveau plan ou pour récupérer des rétroactions appropriées. Cette boucle à long terme et la boucle précédente à moyen terme peuvent se développer ensemble, non pas en parallèle mais plutôt sous la forme d'une gestion de tâches en temps partagé, les tentatives d'ajustement du plan actuel n'étant pas abandonnées pendant la construction d'un nouveau plan.

2.2.2.9. Modèle COSIMO

Le modèle *COSIMO* (*COgnitive SImulation MOdel of human decision making and behaviour*), mis au point par Cacciabue [Cacciabue *et al.* 90] est basée sur le fait qu'un opérateur humain utilise ses expériences passées et effectue une mise en correspondance entre ses connaissances en mémoire et les éléments issus de l'environnement. Le modèle comporte deux niveaux cognitifs :

● le haut niveau de décision (HD) est associé aux activités de diagnostic et de planification. Ce modèle comporte deux éléments : *la mémoire de travail*, zone de stockage temporaire, dans laquelle arrivent des indices externes, mais aussi internes (générés par un processus cognitif). Le nombre maximum est un paramètre du modèle. Ces indices sont traités par un *filtre cognitif* qui opère une sélection grâce à divers principes : dominance visuelle (classement selon des dimensions physiques), détection d'un changement, lien au contexte. Un paramètre du modèle permet de donner plus d'importance aux indices physiques (par exemple dans une situation de stress) ou au contraire aux indices cognitifs (pertinents pour résoudre le problème).
Les indices sélectionnés sont ensuite exploités par la *base de connaissance*, le second élément du modèle. Celle-ci est composée de *frames* ou schémas (voir 4.2.1.4 pour plus de précisions sur les concepts de base), qui représentent les connaissances de l'opérateur d'un processus (structures géométriques, relations causales, comportements, tâches, procédures, actions et leurs effets). Un *frame* contient une étiquette qui le relie à un événement spécifique, des attributs, ainsi que sa fréquence d'utilisation (ou la récence). Un *frame* ou les parties d'un *frame* (récursivité de la représentation) sont adressables par le contenu, grâce à des mises en adéquation entre les attributs et les indices filtrés de la mémoire de travail. Les attributs étant des termes linguistiques vagues, la logique floue est utilisée pour les modéliser. Ainsi, l'opérateur, confronté à une nouvelle situation perçue grâce à des indices dans la mémoire de travail, effectue un processus de reconnaissance des *frames* pertinents dans sa base de connaissance, grâce une mise en adéquation floue. Il s'agit donc d'un processus dirigé par les

indices (données). La sélection finale d'un *frame* est basée sur la fréquence (ou la récence). Lorsqu'un *frame* est instancié, l'opérateur recherche l'information la plus pertinente au sein de ce *frame*, par exemple une hypothèse de diagnostic ou d'évaluation de la situation. Il s'agit donc d'un processus dirigé par les *frames*. Cette hypothèse est ensuite évaluée par l'opérateur, en utilisant un seuil de confiance (paramètre du modèle). Il est à noter que le processus de diagnostic est dynamique et peut être interrompu et redémarré plus tard. Selon Moray, un *frame* est une généralisation d'expériences similaires, il contient des informations par défaut et est assez pauvre en détails (il est une simplification de la réalité). Ainsi, l'information perçue se combine aux *frames* généraux pour représenter la situation courante de façon détaillée [Moray 97].
Une fois confirmé, le *frame* active une structure de but à atteindre. La planification, si l'opérateur est expérimenté, consiste à instancier des *frames*, soit en utilisant une procédure spécifique destinée à être immédiatement exécutée, soit en adaptant un plan à la situation actuelle par analogie avec une situation similaire, soit en construisant un plan grâce à des parties de plans existants (les parties sont adressables par leur contenu) qui sont adaptés et combinés pour former le nouveau plan. Si l'opérateur n'est pas expérimenté, un réel processus de planification a lieu, où les liens logiques entre les composants du système ainsi que le raisonnement inférentiel sont exploités. Cela se produit effectivement si aucun *frame* en mémoire représentant un plan ne peut être mis en adéquation avec les indices présents dans la mémoire de travail, et si le temps n'exerce pas de contrainte sur la prise de décision.

● le bas niveau de décision (BD) correspond à l'exécution d'une réponse ou d'une stratégie planifiée. La structure utilisée à ce niveau est un arbre "et-ou" : le sommet représente le but principal, décomposée récursivement en sous-buts, qu'il faut tous atteindre (liens hiérarchiques "et"), ou dont l'un au choix doit être atteint (liens hiérarchiques "ou"). Chaque but est caractérisé par des paramètres, dont certains sont calculés durant l'exécution du plan et reflètent le degré de satisfaction de l'opérateur (corrélation entre le résultat d'un but et l'attente de l'opérateur) et le degré de certitude (mesure de l'atteinte d'un but). La logique floue et la logique des possibilités sont utilisées à cet effet. Ces calculs conditionnent la continuation du plan, ou son arrêt, ou encore la recherche d'informations pour se décider en cas de conflit (par exemple l'opérateur est certain de ses précédentes actions mais n'est pas satisfait par le but courant).

● la rétroaction a lieu en modifiant la séquence d'actions en cours, au niveau BD, ou bien en recherchant un nouveau plan, au niveau HD.

● le modèle de méta-connaissances à trois couches [Amalberti et Cacciabue 90] (voir ci-dessus) est incorporé. Les heuristiques générales (niveau 1) permettent de prendre en compte la capacité à exécuter une stratégie, le temps requis, l'importance d'un événement la connaissance sur la fiabilité des composants utilisés. Les préférences (niveau 2) utilisées sont les suivantes : l'interprétation de la dynamique du processus et l'évaluation du temps disponible peuvent modifier la priorité des stratégies (repousser à plus tard une stratégie pour traiter une situation d'urgence, modifier l'ordre d'exécution, abandonner une stratégie de faible priorité) ; la recherche de nouvelles informations (éventuellement redondantes) peut permettre de confirmer un diagnostic ; lors de la communication, les messages sans importance pour le groupe sont filtrés par l'opérateur, c'est-à-dire mis de coté et non communiqués ; la catégorisation des messages peut diminuer la charge de la mémoire de travail. La sélection de procédures spécifiques (niveau 3) peut s'appuyer sur l'évaluation de concepts liés aux préférences (risque, pression liée au temps, …). Par exemple, afin d'évaluer la pression liée au temps, les caractéristiques suivantes sont prises en compte : nombre

d'alarmes, dynamique des événements, temps disponible, connaissances de l'opérateur sur la stratégie courante (familiarité, prédictibilité, …).

2.2.2.10. Modèle COCOM

Le modèle de Rasmussen est défini par une séquence d'activités, de la détection d'une alerte à l'exécution d'une action. Il s'inscrit donc dans le cadre des modèles de cognition basés sur les prototypes procéduraux, qui présument qu'il existe en mémoire des séquences prédéfinis d'actions. Le contrôle est donc implicite puisque la prochaine action attendue est donnée par l'ordre naturel des actions du prototype. Par opposition, le modèle de contrôle contextuel *COCOM* (*COntextual COntrol Model*) [Hollnagel 93] présume que les actions sont déterminées par le contexte, défini comme l'interprétation (ou compréhension) de la situation par l'opérateur, qui est influencée par ses connaissances concernant le but à atteindre, les liens entre les actions, les ressources et contraintes. D'autre part, il est impossible *a priori* de postuler des relations entre les actions, bien qu'il puisse y avoir des séquences fréquemment rencontrées, et donc réutilisées dans des circonstances similaires, ce qui correspond à une caractéristique de la cognition humaine : la tendance à minimiser l'effort mental.

L'idée de base du modèle *COCOM* est de séparer le contrôle des actions et les connaissances sur les actions.

● *Le modèle des compétences* concerne les connaissances sur les actions. En fonction de l'expérience de l'opérateur, une action peut être vue comme un processus complexe se décomposant en une série de plusieurs actions élémentaires, donc nécessitant de nombreux pas, ou au contraire comme un processus simple, élémentaire, ne nécessitant qu'un seul pas (en fait dans ce cas la séquence complexe est regroupée au sein d'une procédure d'exécution acquise au cours de l'expérience). Il existe en fait plusieurs types d'association entre actions : les plans, les procédures, les heuristiques vues comme des guides, les liens forts. Chaque pas d'une association peut être une action (plus petite unité du modèle) ou bien une autre association, en fonction du but de l'analyse et du contexte.

● Par ailleurs, le *modèle de contrôle* détermine le choix de la prochaine action, dans le cadre d'une planification à court terme. Le choix d'une action dépend de l'expérience de l'opérateur, des indices externes, du niveau de détail (granularité) de l'analyse. Il est même possible pour les besoins d'un problème que les actions soient représentées à différents niveaux de détails, c'est-à-dire que la séquence corresponde à un assemblage d'actions simples et composites. Quatre modes de contrôle sont définis : le contrôle "au hasard" (le choix de la prochaine action ne peut être prévu, il est réalisé par essai-erreur, sans réflexion, dans des situations de panique), le contrôle opportuniste (le choix est basé sur le contexte courant, soit sur les indices externes qui attirent l'attention dans une situation inhabituelle, soit sur ceux perçus comme importants d'après l'expérience, et donc fait peu appel à la planification et à l'anticipation), le contrôle tactique (le choix est fait en suivant plus ou moins une procédure connue, à partir d'une planification à horizon très limité), et enfin le contrôle stratégique (le choix est basé sur une planification à horizon plus large, et les dépendances fonctionnelles entre les actions (par exemple les pré-conditions d'une action, la séquentialité entre identification, comparaison et sélection des alternatives) jouent un rôle important). Ainsi, les retours-arrière qui peuvent être fréquents en mode de contrôle opportuniste le sont beaucoup moins en mode de contrôle stratégique.

2.3. POINT DE VUE DE L'OPERATEUR ARTIFICIEL

Un système d'aide à la décision doit fournir une expertise à l'opérateur humain, mais laisser la décision finale à ce dernier. Le but est d'améliorer la qualité de la décision humaine et de réduire la charge mentale, car il a été largement prouvé qu'une charge excessive induit des erreurs de la part de l'opérateur humain [Moray 97]. Il s'agit donc pour nous d'étudier de nouvelles façons d'utiliser les concepts existants en Intelligence Artificielle (IA) afin d'améliorer l'aide à un opérateur humain, et la conception d'IHM plus pertinentes.

L'idée est de développer des opérateurs artificiels issus de l'IA et capables de résoudre des problèmes de supervision dans un "style cognitif" spécifique. Le but est de s'approcher au mieux du raisonnement humain mis en œuvre lors d'une opération de supervision de procédé afin d'améliorer l'aide et l'interaction avec l'opérateur, ainsi que l'intelligibilité et la capacité explicative du système. Ainsi, l'agent artificiel peut être vu comme une référence. Trois approches seront testées et détaillées au cours des chapitres 3 à 6. Les raisons qui ont motivé leur choix sont les suivantes :

● ***la modélisation qualitative causale.*** Etant donné qu'un humain a très souvent tendance à raisonner de manière qualitative, par opposition à un raisonnement numérique, c'est-à-dire à simplifier une réalité physique complexe en appliquant des règles de sens commun, il paraît utile de modéliser ce type de raisonnement. Il s'agit de formalismes permettant de représenter des systèmes physiques et de raisonner sur ces derniers. L'idée de base est que plus les concepts sont intuitifs, naturels et qualitatifs, et plus ils révèlent la pertinence de l'information et plus ils permettent d'approcher la causalité sous-jacente du système à modéliser. Ainsi, plutôt que de résoudre les équations du système physique représenté, ce qui peut être extrêmement complexe, mieux vaut parfois ne considérer que les influences entre variables. Par exemple, un humain face au problème du micro-monde lié à l'hydraulique (voir 2.4. et [Ponsa et Catala 99 (a)]) ne résout pas les équations de la mécanique des fluides, mais raisonne en utilisant des concepts qualitatifs, de sens commun qui s'y rattachent. Ainsi, un agent qualitatif a été mis au point dans le cadre de notre étude (chapitre 3).

● ***la catégorisation et la reconnaissance des formes***. Etant donné que l'humain a une forte propension à catégoriser, qu'il s'agisse de reconnaître un objet (une forme visuelle, un type de problème, un concept, un événement) ou d'attribuer une propriété à un objet, lui permettant de raisonner efficacement et rapidement, il paraît intéressant d'utiliser une approche basée sur la classification (numérique ou symbolique) et la reconnaissance des situations typiques mises en jeu lors d'une opération de supervision de processus. Ainsi, nous avons tenté dans le cadre de notre étude de mettre au point un agent basé sur la classification à partir d'exemples (chapitre 6).

● ***le raisonnement à base de cas***. Cette approche se base sur le fait qu'un humain utilise très souvent ses expériences passées, qu'il s'agisse de réussites ou bien d'échecs, afin de résoudre un problème courant. La méthode consiste généralement à rechercher en mémoire une expérience jugée similaire, à adapter cette expérience au contexte présent, à la valider, la réutiliser, puis enfin à la stocker. Ainsi, un agent à base de cas a été mis au point dans le cadre de notre étude (chapitres 4 et 5).

Certaines des caractéristiques des modèles cognitifs de l'opérateur humain de supervision précédents (chapitre 2.2) sont présentes au sein des agents artificiels : ces derniers utilisent un modèle de décision similaire à celui proposé par Rasmussen (observation de la situation

courante, détection d'une alarme, prédiction de l'évolution, définition d'un but, choix d'une action, exécution de l'action). En outre, les agents artificiels effectuent une action en se basant uniquement sur l'état courant du système, et introduisent donc une similarité avec le modèle *COCOM* qui préconise que le choix de la prochaine action est basé sur le contexte courant. Enfin l'agent à base de cas utilise le même principe de base que le modèle *COSIMO* (utilisation d'expériences passées et mise en correspondance).

Chacune des approches possède une *caractéristique essentielle du raisonnement humain*. Les qualités et défauts mis en évidence lors des tests informatiques des agents artificiels permettront d'analyser dans quelles situations une approche est plus appropriée qu'une autre, afin de développer une coopération entre les agents. Cette coopération a pour but d'approcher le raisonnement d'un opérateur humain en situation de supervision de procédé, et donc d'approcher la *complémentarité* qui existe entre ces modes de raisonnements.

2.4. COMPATIBILITE ENTRE LES DEUX POINTS DE VUE

2.4.1 Comparaison des raisonnements d'un opérateur humain et d'un agent artificiel

Des tests seront effectués sur des sujets humains normaux afin de comparer leurs raisonnements avec ceux des agents artificiels. Les objectifs sont :

● de trouver pour chaque sujet humain l'opérateur artificiel le plus proche,

● d'exhiber les situations dans lesquelles tel opérateur artificiel est plus adéquat qu'un autre, et de faire ressortir les éléments permettant de définir une architecture pour la coopération entre les trois agents, de façon à être le plus proche possible du raisonnement humain.

Il s'agit alors de construire une distance issue de la comparaison entre le raisonnement mené par un agent artificiel et celui d'un opérateur humain. Cette mesure peut être quantitative ou bien qualitative.

Par exemple, pour le problème des Tours de Hanoï, un agent artificiel pourrait effectuer une recherche optimale dans un arbre d'états, et le critère de comparaison serait le nombre de pas utilisés par le sujet pour trouver la solution diminué de la longueur du plus court chemin dans l'arbre d'états élaboré par l'agent artificiel.
Le problème de ce type de critère numérique est qu'il ne tient pas compte des stratégies et des buts. Il semble important que le critère de comparaison accède aux stratégies des opérateurs.

Dans le cadre de notre étude, nous avons développé plusieurs critères de comparaison. Tout d'abord un critère numérique simple permet de juger globalement de la qualité de la supervision menée par un opérateur grâce à une fonction linéaire pondérée des deux buts à optimiser, le temps et le volume total qui a débordé des réservoirs.
Nous avons également mis au point une méthode permettant de juger de la compatibilité globale entre un opérateur artificiel et un opérateur humain par la reconnaissance de l'intention associée à chaque action de l'opérateur humain. Il s'agit dans un premier temps d'étudier le comportement de l'agent artificiel afin de dégager des associations situation générique → intention. Une situation générique correspond à un petit ensemble de réservoirs reliés et associés à des conditions générales portant par exemple sur les volumes ou les hauteurs d'eau des réservoirs. Chacune de ces situations est associée à une intention, par

exemple éliminer une alarme, ou accélérer le processus, … Il est à noter qu'une situation est définie volontairement de manière imprécise afin d'une part d'être applicable dans des conditions variées et d'autre part de pouvoir s'adapter aux notions floues chez un humain que sont la perception d'une hauteur d'eau, ou l'état d'alarme d'un réservoir, atteint lorsque le niveau d'eau est supérieur à un seuil et croît au cours du temps (voir 2.1.2. et 3.2.1.). La comparaison consiste alors à voir dans quelle mesure une action d'un opérateur humain s'inscrit dans le comportement de l'agent artificiel, ce qui nécessite d'une part une mise en adéquation floue entre la situation concrète qui a amené l'opérateur humain à agir de telle façon et des situations génériques définies lors de l'étude du comportement de l'agent artificiel, et d'autre part de vérifier si certaines conditions sont remplies, notamment l'appartenance de l'action de l'opérateur humain à l'ensemble des actions admissibles correspondant à l'intention associée à la situation générique. La modélisation floue permet alors le calcul des "degrés d'intention" associés à l'action de l'opérateur humain, et ceci pour chacune des intentions possibles de l'agent artificiel.

2.4.2. Coopération entre opérateurs

Il a été largement prouvé que la communication et la coopération jouent un rôle central entre les membres d'une équipe de travail [Moray 97]. Il paraît donc naturel d'étudier ces concepts afin d'améliorer la coopération d'une part entre un opérateur humain et un agent artificiel [Vanderhaegen 99] et d'autre part entre plusieurs agents artificiels.
Selon Vanderhaegen, Millot et Chalmé [Vanderhaegen *et al.* 04], la **coopération** a pour but d'optimiser la performance des systèmes ou l'intérêt collectif en interagissant avec les autres et de diminuer les erreurs de contrôle. Il s'agit donc d'un comportement orienté vers le succès. Deux opérateurs humains coopèrent si chacun tente d'atteindre des buts qui interfèrent avec ceux de l'autre opérateur et si chacun peut contrôler ces interférences de sorte à faciliter les activités de l'autre opérateur. Par opposition, la **compétition** a pour but d'optimiser la performance individuel ou l'intérêt personnel en interagissant avec les autres, ou de faire faire des erreurs aux autres opérateurs ou de les tromper. Deux opérateurs humains sont en compétition si leurs buts interfèrent et chacun tente de contrôler ces interférences de sorte à faciliter sa propre activité ou induire l'autre opérateur en erreur.

L'approche des systèmes multi-agents est très intéressante à ce sujet. Détaillons le point de vue de Ferber [Ferber 95]. Tout d'abord, la coopération a lieu dans une situation d'interactions. Une interaction est une mise en relation dynamique de deux ou plusieurs agents par le biais d'un ensemble d'actions réciproques. Les interactions s'expriment à partir d'une série d'actions dont les conséquences exercent en retour une influence sur le comportement futur des agents. La coopération dépend de trois paramètres :

- les buts des agents. Des agents sont dans une situation de coopération si leurs buts sont compatibles. Deux buts caractérisés par l'atteinte des deux états p et q sont incompatibles si $p \Rightarrow \neg q$

- les ressources des agents. Il s'agit de tous les éléments environnementaux et matériels utiles à la réalisation d'une action : outil, énergie, temps, espace, …

- le rapport entre les capacités des agents et les tâches à accomplir. Un agent peut-il réaliser seul une tâche, ou a-t-il besoin des autres ?

Buts	Ressources	Compétences	Types de situation	Catégorie
Compatibles	Suffisantes	Suffisantes	Indépendance	Indifférence
Compatibles	Suffisantes	Insuffisantes	Collaboration simple	Coopération
Compatibles	Insuffisantes	Suffisantes	Encombrement	
Compatibles	Insuffisantes	Insuffisantes	Collaboration coordonnée	
Incompatibles	Suffisantes	Suffisantes	Compétition individuelle pure	Antagonisme
Incompatibles	Suffisantes	Insuffisantes	Compétition collective pure	
Incompatibles	Insuffisantes	Suffisantes	Conflits individuels pour des ressources	
Incompatibles	Insuffisantes	Insuffisantes	Conflits collectifs pour des ressources	

Tableau 3 : Classification des situations d'interactions [Ferber 95]

Le tableau 3 est interprété comme suit :

- Indépendance : il s'agit d'une simple juxtaposition des actions des agents pris indépendamment sans qu'il y ait effectivement d'interaction.

- Collaboration simple : il s'agit d'une simple addition des compétences. Les agents ne font que partager des connaissances pour résoudre un problème.

- Encombrement : les agents se gênent mutuellement dans l'accomplissement de leurs tâches alors qu'ils n'ont pas besoin les uns des autres.

- Collaboration coordonnée : les agents doivent coordonner leurs actions pour pouvoir disposer de la synergie de l'ensemble de leurs compétences. Ainsi s'ajoutent aux problèmes d'allocation de tâches des aspects de coordination dus aux ressources limitées.

- Compétition individuelle pure : si les buts sont incompatibles, les agents doivent lutter ou négocier. L'accès aux ressources ne constitue pas l'enjeu du conflit. Il n'y a pas de problèmes spécifiques d'interaction. Simplement, que le meilleur gagne.

- Compétition collective pure : lorsque les agents n'ont pas la compétence suffisante, ils doivent se regrouper au sein de coalitions ou d'associations. Il y a donc un double mouvement : le premier tend à liguer les individus au sein de groupes unis par des liens de collaboration coordonnée et le second à opposer les groupes entre eux.

- Conflit individuel pour des ressources : les ressources ne peuvent être partagées, et il s'agit donc d'un conflit dont les ressources sont l'enjeu.

- Conflits collectifs pour des ressources : les coalitions doivent lutter les unes contre les autres pour obtenir des ressources. Cette situation combine la compétition collective aux conflits individuels pour des ressources.

Il est à noter qu'une situation de coopération (au niveau macroscopique) peut produire au niveau local (microscopique) des situations de compétition. La coopération peut être vue comme une attitude intentionnelle de la part des agents engagés envers un but collectif, ou bien sous l'angle du bénéfice positif obtenu grâce aux interactions des agents. Un autre point de vue est de voir la coopération comme une qualification de l'activité d'un ensemble d'agents par un observateur extérieur qui n'aurait pas accès aux états mentaux des agents.

Les méthodes de coopération sont :

- Le regroupement et la multiplication : il s'agit pour les agents, de se rapprocher physiquement, soit par le biais d'un bloc plus ou moins homogène dans l'espace, soit par le biais d'un réseau de communication, un groupe agissant comme un organisme distribué, avec divers types de spécialistes. En outre, l'augmentation du nombre d'agents peut accroître les performances du groupe.

- La communication : le système de communication agrandit les capacités perceptives des agents en leur permettant de bénéficier des informations et du savoir-faire des autres agents.

- La spécialisation : il s'agit du processus par lequel les agents deviennent progressivement de plus en plus adaptés à leurs tâches. Cela n'est pas nécessairement le fruit d'un choix *a priori*.

- La collaboration par partage de tâches et de ressources : cela consiste à travailler à plusieurs sur un projet commun. Il s'agit pour cela de (se) répartir des tâches, des informations et des ressources. L'allocation de tâches peut passer par des mécanismes d'offre et de demande. On distingue la répartition centralisée grâce à un agent coordinateur, de la répartition distribuée, où tout agent peut être à la fois offrant ou demandeur, qui nécessite soit les représentations mutuelles des capacités de chacun (réseaux d'accointances), soit la mise au point de la notion de marché par des techniques d'appels d'offre. Vanderhaegen distingue en outre un mode de répartition selon une organisation hiérarchisée, dans laquelle un agent A est contrôlé par un autre agent B de niveau supérieur. L'auteur indique également la possibilité d'un contrôle "mixé" (*mixed allocator control*) lorsque l'agent A est en outre contrôlé par lui-même. Par exemple, l'allocation d'une tâche à A peut être anticipée par B, puis être modifiée "on-line" par A [Vanderhaegen 99].

- La coordination d'actions : la réalisation de tâches productives entraîne tout un cortège de tâches de coordination (non directement productives) sans lesquelles les premières ne peuvent être accomplies. Ceci est lié à la définition de l'ordre des actions à effectuer.

- La résolution de conflit par arbitrage et négociation : l'arbitrage conduit à la définition de règles de comportement qui agissent comme des contraintes sur l'ensemble des agents, mais dont le résultat global a pour effet de limiter les conflits et de préserver à la fois les individus mais surtout la société d'agents. Les agents doivent alors posséder une sorte de "libre arbitre" afin d'être capables de peser le pour et le contre des conséquences de leurs actions. Mais lorsque les agents sont en conflit d'objectif ou de ressource, on préfère souvent ne pas recourir à un arbitre, mais laisser les agents résoudre eux-mêmes le conflit par la recherche d'un accord bilatéral au cours d'un processus de négociation.

Il existe d'autres modèles que les systèmes multi-agents pour gérer les situations de coopération et de compétition. Citons le modèle Bénéfice-Coût-Déficit (BCD) [Vanderhaegen *et al.* 04] : il s'agit d'évaluer l'impact attendu d'une action A, grâce au calcul du coût acceptable basé sur un critère i et généré par A et son succès $C_i(A)$, du bénéfice attendu basé sur un critère j et généré par A et son succès $B_j(A)$, du possible déficit inacceptable basé sur un critère k et généré par A et son échec $D_k(A)$, et de l'erreur ε sur B, C et D. Ce modèle peut ainsi être appliqué afin de prendre en compte une alternative entre les activités de compétition et de coopération, en évaluant les impacts de ces activités. Il est alors possible de planifier une augmentation ou une diminution des impacts positifs ou négatifs de ces activités.

Chapitre 3

RAISONNEMENT A BASE DE MODELE ET RAISONNEMENT QUALITATIF

Nous présentons brièvement le raisonnement à base de modèles qualitatifs, puis détaillons l'agent artificiel basé sur ces principes : à partir d'informations qualitatives et d'un graphe d'influences, le système cherche à atteindre les buts sous-jacents aux modes de fonctionnement, ce qui le conduit à exécuter une action grâce notamment à une prédiction qualitative des effets des actions possibles. Nous analysons enfin les résultats des tests informatiques de cet agent.

3.1. INTRODUCTION

Fournir des modèles qui représentent les systèmes physiques est soumis à des contraintes (disponibilité des connaissances, adaptation à une tâche précise), ce qui limite l'application des modèles numériques traditionnels et ouvre la voie aux **modèles qualitatifs** [Trave-Massuyes et Dague 03] [Weld et De Kleer 90] dont les caractéristiques sont les suivantes :

● gestion des connaissances imprécises et incertaines

● les résultats fournis équivalent à une représentation compacte d'une infinité de résultats numériques

● les prédictions appréhendent les distinctions qualitatives significatives du comportement du système

● les primitives de modélisation permettent une interprétation intuitive.

Le point clé est la discrétisation du domaine de valeurs des variables en un nombre fini de symboles ordonnés, c'est-à-dire *l'abstraction de l'espace d'état*, complété par *l'abstraction fonctionnelle* qui permet de représenter des relations fonctionnelles entre quantités connues de manière qualitative. Il est à noter que le raisonnement qualitatif est fondé sur des résultats théoriques qui garantissent que les modèles qualitatifs sont des abstractions correctes de modèles réels.

Dans ce cadre, le **raisonnement causal** peut être utile aussi bien pour prédire l'état d'un système, que pour expliquer le fonctionnement normal ou la défaillance d'un système. Ainsi,

un domaine privilégié d'application est le diagnostic, et plus généralement la supervision de processus. Il est issu de la problématique suivante : comment passe-t-on d'une théorie scientifique sous forme de lois non causales à des explications causales des phénomènes de la nature ? Les caractéristiques importantes de cette approche sont :

● ***la création d'un modèle qualitatif***, généralement sous la forme d'un graphe orienté dans lequel les nœuds représentent les variables du système et les arcs représentent les relations entre les variables (liens causaux ou influences de divers types). Ce graphe est construit à partir des équations statiques et dynamiques décrivant le système. En outre, les arcs peuvent être étiquetés par le type de la relation (additive, multiplicative, différentielle, …) ainsi que par les paramètres de la relation.

Dans des cas plus complexes, il peut être nécessaire de prendre en compte les modes opératoires du système [Trave-Massuyes et Pons 97], par exemple dans le cas où sont présents des switchs ou des vannes tout ou rien, ou bien lorsque des processus ont un comportement différent suivant des conditions opératoires. Dans ce cas, chaque équation est associée à une condition sous la forme d'une formule logique associée à un mode opératoire. Le modèle est donc un graphe orienté comme précédemment, mais les arcs sont étiquetés par une condition. Il est également possible, toujours dans des cas complexes, d'utiliser un modèle d'équations orienté composant. Le comportement d'un composant est représenté par un ensemble d'équations (et donc de variables). Ainsi, les arcs et/ou les nœuds du graphe orienté peuvent être étiquetés par le composant associé.

Il existe d'autres façons de modéliser le système physique : il peut s'agir d'un ensemble de formules logiques du premier ordre décrivant le fonctionnement normal (et éventuellement anormal) du système [Reiter 87] [Cordier *et al.* 00], ou bien d'un ensemble de contraintes numériques (linéaires ou non) issues des lois physiques et permettant de prendre en compte des variables dont les domaines de valeurs sont des intervalles réels. Il est également possible de prendre en compte des imprécisions et incertitudes sous la forme d'intervalles réels concernant les paramètres du système [Trave-Massuyes *et al.* 01] [Jimenez Molina 00]. Un modèle de contraintes dont la résolution utilise l'algèbre des intervalles et la propagation, permet alors de filtrer (réduire) les domaines des variables et paramètres.

L'idée sous-jacente à la notion de modèle est que la connaissance sur le système (le modèle) est découplée de la connaissance de raisonnement (l'utilisation du modèle pour résoudre un problème) qui elle est générique.

● ***l'utilisation des modèles*** : il peut s'agir d'une prédiction du comportement du système, grâce à un modèle (modèle de détection) du fonctionnement normal du système, que l'on compare aux observations ou mesures. Ceci est utile pour détecter un comportement anormal du système. Dès lors, il est judicieux d'ajouter un module de diagnostic afin de rechercher la (ou les) cause du comportement anormal, et qui utilise lui aussi un modèle (modèle de localisation), dans lequel les hypothèses de bon fonctionnement sous-tendent de manière explicite les relations de comportement. Une identification plus précise peut ensuite être faite grâce à des modèles de fautes.

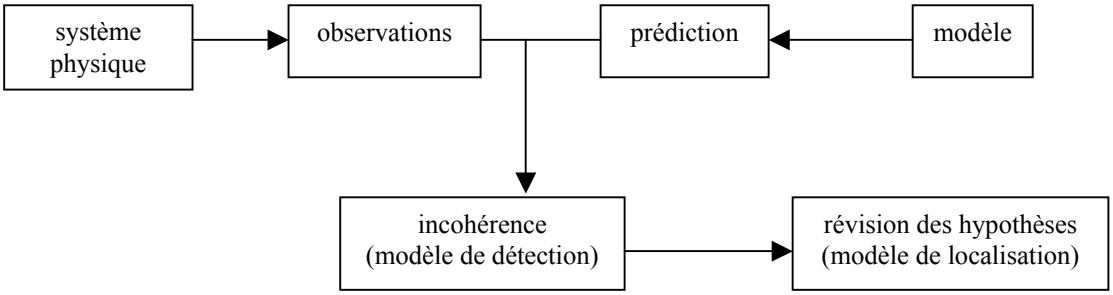


Figure 5 : Utilisation d'un modèle pour le diagnostic

D'un point de vue historique, il est utile de présenter (succinctement) les deux approches qui sont à l'origine du raisonnement qualitatif :

● ***l'ordonnancement causal***. La méthode développée par Iwasaki et Simon comporte trois phases [Trave-Massuyes et Dague 03] [Iwasaki et Simon 86] [Weld et De Kleer 90] :

- la modélisation : à partir d'un système d'équations linéaires décrivant le système physique à l'équilibre, il s'agit de déterminer les variables exogènes (variables d'entrée du système) en utilisant trois heuristiques (le temps : "si il y a un ordre temporel strict entre deux variables $x_i \rightarrow x_j$, alors x_i est probablement plus exogène que x_j" ; l'ordre de grandeur des influences des variables : "si les effets de x_j sur x_i peuvent être négligés, alors x_i est probablement plus exogène que x_j" ; l'expérimentateur : "si l'expérimentateur peut contrôler directement une variable x_i, alors x_i est probablement exogène"), ainsi que les variables endogènes qui sont elles influencées par des variables du système.

- la partition des variables : on génère successivement des sous-ensembles de variables du système : S_0, S_1, …, S_n. Ainsi, en partant des variables exogènes, on va fixer progressivement les valeurs de toutes les variables.

- les liens causaux entre variables : il s'agit d'établir des liens causaux (orientés) entre les variables d'un sous-ensemble S_i et celles d'un sous-ensemble S_{i+1}.

● ***la physique qualitative***. Trois approches ont été développées : l'approche centrée contraintes [Kuipers 84], l'approche centrée composant [De Kleer et Brown 84], et l'approche centrée processus [Forbus 86]. On peut également citer les travaux de Hayes sur la "physique naïve" qui avaient pour but de formaliser le raisonnement de sens commun utilisé en physique.
Nous allons présenter *l'approche centrée composant* [De Kleer et Brown 84]. Le système physique (par exemple un circuit hydraulique) est modélisé par trois types d'objets : *les matériaux* (un fluide) auxquels on associe des variables (pression, débit) qualitatives, c'est-à-dire qui ne prennent qu'un ensemble fini de valeurs discrètes, *les conduits* qui transportent les matériaux d'un composant à un autre, et *les composants* qui agissent sur les matériaux, et définis par des équations qualitatives intégrant des variables qualitatives et leurs dérivées par rapport au temps. On définit alors les différents modes de fonctionnement (ou états qualitatifs) de chaque composant, ainsi que le diagramme d'état du système qui permet de représenter la notion de causalité (un changement d'état est lié à un changement de valeur d'une variable ; les états sont ordonnés dans le temps, la cause précédant l'effet). Ensuite, on simule le système en résolvant les équations qualitatives grâce à un algorithme qui utilise la gestion d'hypothèses et la propagation de contraintes.

3.2. AGENTS QUALITATIFS DANS LE CADRE DE NOTRE ETUDE

L'approche mise au point dans [Trave-Massuyes *et al.* 99] est la suivante : à partir d'informations numériques concernant les hauteurs d'eau, le système les traduit en informations uniquement qualitatives (tendance évolutive, niveau de danger/alarme). Cela correspond à la phase de perception. Deux situations peuvent alors se présenter qui correspondent aux modes de fonctionnement du système : cas avec alarme et cas sans alarme. Chacune de ces deux situations est liée à des buts à atteindre. Pour atteindre ces buts, le système détermine un ensemble d'actions potentielles grâce à un modèle qualitatif représentant les relations causales entre les composants, puis il prédit qualitativement les effets des différentes actions envisageables.

Présentons tout d'abord l'agent qualitatif que l'on qualifie de naïf (Naive Qualitative Agent **NQA**), développé dans [Trave-Massuyes *et al.* 99], et que nous avons implémenté :

3.2.1. Modèle perceptif

Nous reprenons ici les notations introduites au paragraphe 2.1.2.
Le système est perçu à travers les quantités suivantes :

● des hauteurs d'eau qualitatives dans chaque réservoir. Nous utilisons les quatre valeurs :

EMPTY ($h_i(t) = 0$), **LOW** ($h_i(t) < \alpha_i$), **HIGH** ($\alpha_i \leq h_i(t) \leq H_i$) et **FULL** ($h_i(t) = H_i$)

● des tendances qualitatives dans chaque réservoir. Les trois valeurs suivantes sont utilisées :

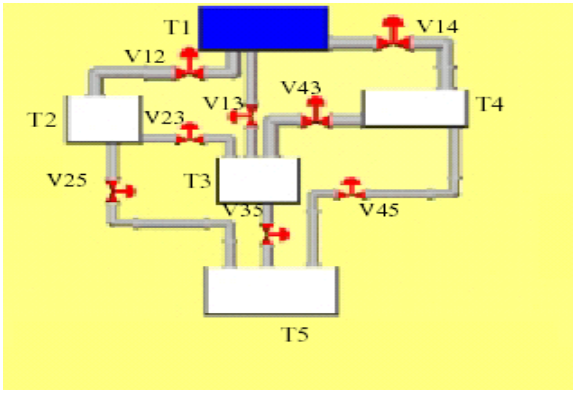
INC (increase : $h_i(t) - h_i(t\text{-}1) > 0$), **DEC** (decrease : $h_i(t) - h_i(t\text{-}1) < 0$), **STD** (steady : $h_i(t) - h_i(t\text{-}1) = 0$).

● des alarmes. Un réservoir est en alarme si sa hauteur est *HIGH* et sa tendance *INC* ou si sa hauteur est *FULL* et sa tendance *STD*.

● des positions *ON* ou *OFF* des vannes.

3.2.2. Stratégie de contrôle

Tout d'abord, le système est représenté par un graphe orienté dont les nœuds sont les réservoirs et les arcs sont les conduites (voir figure 5). Un poids est affecté aux arcs (0 → vanne fermée ; 1 → vanne ouverte ; 2 → pas de vanne). On définit la notion de chemin ouvert, qui est un chemin orienté du graphe ne comportant pas de vanne fermée. Le graphe représente donc les influences entre les réservoirs.



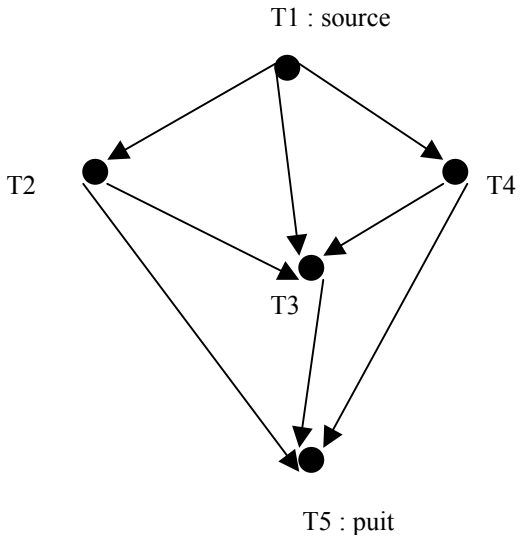


Figure 6 : Graphe orienté associé à une configuration de réseau

Deux situations peuvent se présenter, le cas sans alarme et le cas avec alarme, qui correspondent à des buts différents. Il est donc à noter qu'ici, le point de focus est l'alarme.

Cas sans alarme

L'objectif est d'accélérer le processus, c'est-à-dire de transporter le maximum d'eau du réservoir "source" vers le réservoir "puit", que l'on peut traduire par les deux buts :

1. avoir une tendance *INC* pour le réservoir "puit".
2. augmenter le nombre de chemins ouverts (qui ne comportent que des vannes en position *ON*) entre le réservoir "source" et le réservoir "puit".

Un algorithme détermine les ***actions admissibles en cas de non alarme*** (qui ouvrent une vanne) et choisit la meilleure action selon divers critères : le principe est d'ouvrir une vanne située le plus bas possible et sur un chemin comportant le moins de vannes fermées :

Étape 1 : déterminer tous les chemins entre le réservoir "source" et le réservoir "puit". Si le réservoir "source" est vide, alors éliminer les sous-chemins à partir du haut correspondant à des réservoirs vides.

Étape 2 : étiqueter chaque chemin *i* par le nombre de vannes fermées : $label_i$.

Étape 3 : choix d'une action.
Si tous les chemins sont ouverts, alors on ne fait rien.
Sinon, on considère les chemins d'étiquette minimale. S'il y en a un seul, alors on ferme la vanne la plus basse sur ce chemin. S'il y en a plusieurs, alors on considère pour chacun de ces chemins le diamètre minimum de conduite, et on choisit le chemin qui a la valeur maximale. S'il y en a plusieurs, on tire au hasard. Sur le chemin choisi, on ouvre la vanne la plus basse.

Autrement dit, de manière plus formelle, le choix de l'action *ao* est tel que :
Déterminer les chemins i entre le réservoir "source" et le réservoir "puit"
$\forall$ *le chemin i,* $label_i$ *= nombre de vannes fermées sur i*
$I = arg\ (min\{label_i\})$ *= ensemble des chemins d'étiquette minimale*
$io = arg\ (max_{\ i \in I}\ (min\_\Phi(i)))$
ao = ouvrir vanne la plus basse sur le chemin io
$min\_\Phi(i)$ *= diamètre minimum sur le chemin i*

Cas avec alarme

Si au moins un réservoir est en alarme, l'objectif principal est de revenir à une situation sans alarme, tout en essayant d'accélérer le processus par ailleurs, ce que l'on peut traduire par les quatre buts ordonnés selon un critère d'importance :

1. ne pas accroître le phénomène d'alarme (par exemple ne pas ouvrir (resp. fermer) une vanne en amont (resp. aval)).
2. réduire le nombre de réservoirs en alarme.
3. avoir une tendance *INC* pour le réservoir puit.
4. augmenter le nombre de chemins ouverts (qui ne comportent que des vannes en position *ON*) entre le réservoir source et le réservoir puit.

La méthode consiste, à chaque pas de temps, à exécuter les trois étapes suivantes :

Étape 1 : trouver les ***actions admissibles en cas d'alarme*** (qui sont susceptibles de réduire les alarmes).
Il s'agit d'actions directes : fermer une vanne juste en amont du réservoir en alarme, ou bien ouvrir une vanne juste en aval du réservoir en alarme.
S'il n'y en a pas, alors on recherche des actions indirectes : le principe est de remonter ou redescendre d'un cran par rapport au cas des actions directes, si le réservoir en alarme est relié par des conduites sans vannes (voir figure 7), et ceci de manière récursive tant qu'il y a des conduites sans vanne.

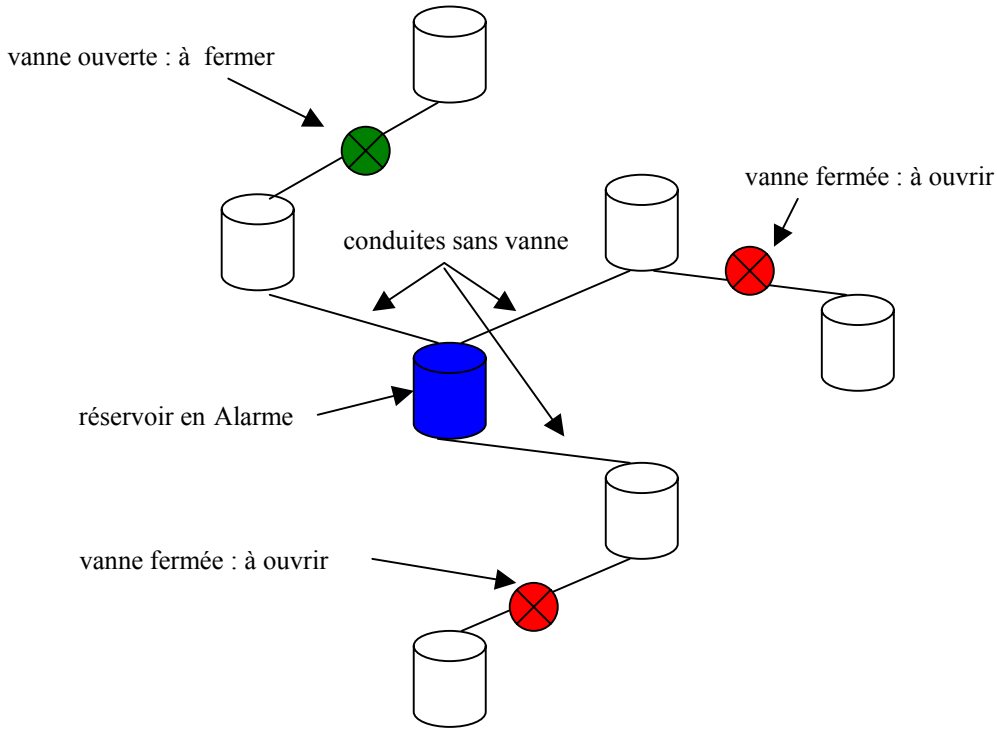


Figure 7 : Actions indirectes en cas d'alarme

S'il n'y a toujours pas d'action, alors on ne fait rien.

Étape 2 : prédiction qualitative à un pas.
Il s'agit, pour chaque action admissible, de trouver les réservoirs influencés par cette action : supposons que l'action concerne une vanne V_{ij} reliant le réservoir amont T_i (en alarme) et le réservoir aval T_j. L'ensemble des réservoirs influencés est :
$G\_V_{ij} = T_i \cup G\_V_{ij}(T_i) \cup T_j \cup G\_V_{ij}(T_j)$

Avec $G\_V_{ij}(T_i)$ = ensemble des réservoirs en aval de T_i sur des chemins ouverts qui n'incluent pas V_{ij}.
Et $G\_V_{ij}(T_j)$ = ensemble des réservoirs en aval de T_j sur des chemins ouverts.

Ensuite, on étiquette chaque réservoir influencé par l'action suivant le sens de cette influence : une influence positive (négative) sur un réservoir indique la croissance (décroissance) du niveau d'eau de ce réservoir (voir tableau 4). On distingue les influences directes sur les réservoirs T_i et T_j (valeurs $\pm \infty$) et les influences indirectes sur les réservoirs T_k appartenant à $G\_V_{ij}(T_i) \cup G\_V_{ij}(T_j)$.

	T_i	$\forall T_k \in G\_V_{ij}(T_i)$	T_i	$\forall T_k \in G\_V_{ij}(T_i)$
Ouverture V_{ij}	$-\infty$	-1	$+\infty$	+1
Fermeture V_{ij}	$+\infty$	+1	$-\infty$	-1

Tableau 4 : Influences sur les réservoirs liées à une action sur une vanne V_{ij}.

Ceci permet de combiner ces influences. En effet, on traduit les influences en terme de tendance en effectuant la somme *S_infl* des influences sur chacun de ces réservoirs, car il se peut que $G\_V_{ij}(T_i) \cap G\_V_{ij}(T_j) \neq \varnothing$ et que $T_j \in G\_V_{ij}(T_i)$.

S_infl > 0	S_infl < 0
INC → INC+	INC → INC-
DEC → DEC-	DEC → DEC+
STD → INC	STD → DEC
Al → Al+	Al → Al-

Tableau 5 : Prédiction de tendance des réservoirs selon l'opérateur NQA

On obtient donc finalement des prédictions de tendances pour chacun des réservoirs influencés par chacune des actions admissibles. Ceci correspond à l'idée intuitive suivante : si une vanne en amont d'un réservoir dont la tendance est *INC* est ouverte, alors le niveau d'eau augmente davantage, d'où la tendance prédite *INC+*. Si une des vannes amont est fermée (les autres restant ouvertes), le niveau d'eau augmente moins, d'où la tendance *INC-*. Le principe est similaire pour les autres symboles.

Étape 3 : choix de l'action.
Pour chaque action admissible, on calcule quatre critères (G_i pour i = 1 à 4) correspondant respectivement à la satisfaction de chacun des quatre buts, en utilisant la prédiction :

- si l'action génère n alarmes *Al+*, $G_1 = -n$.
- si le nombre d'alarmes *Al-* est n et le nombre de nouvelles alarmes est m, $G_2 = n-m$.
- si le réservoir le plus bas est dans l'état *INC+*, alors $G_3 = 2$.
  Si l'état est *INC* ou *INC-* alors $G_3 = 1$. Si l'état est *STD*, alors $G_3 = 0$.
- G_4 = variation du nombre de chemins ouverts (peut être positif ou négatif).

Le score affecté à l'action est une somme pondérée de ces quatre critères : $G = \sum_i p_i.G_i$. (avec $\sum_i p_i = 1$ **et** $p_1 > p_2 > p_3 > p_4$). Les p_i représentent donc des poids reflétant l'importance associée au but i. L'action choisie est celle qui maximise le score G.

Autrement dit, de façon plus formelle, le choix de l'action *ao* est tel que :
A = ensemble des actions admissibles (directes, sinon indirectes)
1- prédiction à un pas :
$\forall a \in A$, T_a = *{ensemble des réservoirs influencés par a}*
$\forall T \in T_a$, *P(T, a) = prédiction de la tendance de T*
2- choix d'une action ao :
$ao = arg\ (max_{a \in A} (\Sigma_{i = 1 à 4}\ p_i.G_i(a)))$
$G_i(a)$ = *fonction du sous-but i et des prédictions P(T,a) avec* $T \in T_a$
p_i = *poids (importance du sous-but i)*

Remarque : on pourrait mettre en place une stratégie semblable à celle de la méthode d'optimisation par recherche *Tabou* [Charon *et al.* 96] [Hanafi 00] [Taillard *et al.* 96] [Aarts et Lenstra 97] : les conduites correspondant aux actions effectuées sont placées dans une *liste Tabou* et y restent pendant un certain nombre de pas de temps (3 ou 4.Δt par exemple). Toute vanne sur une conduite qui se trouve dans la liste Tabou ne peut pas être actionnée. Cela permet d'ajouter de la cohérence dans les actions par effet de mémoire.

En effet, il s'agit d'empêcher d'agir pendant un certain temps sur la vanne qui a permis de réduire une alarme car cette action peut entraîner une tendance décroissante et donc une élimination de l'alarme au pas de temps suivant. Il est alors possible que la même vanne soit ré-ouverte (afin d'ouvrir un chemin par exemple), ce qui entraîne une alarme au pas suivant. On observe alors une succession rapide d'ouverture et de fermeture de la même vanne. L'idée sous jacente à la mise en place de la liste Tabou est un effet "cohérence dans les actions par effet mémoire": on empêche de revenir sur une action effectuée pendant un certain temps (inhiber cette action).
Il semble préférable que cette stratégie Tabou soit active uniquement dans une situation sans alarme. Il ne semble pas judicieux en effet d'empêcher une action permettant de réduire une alarme car cela peut entraîner un débordement de réservoir.

3.2.3. Raffinements dans le niveau d'abstraction

Un agent qualitatif plus raffiné (Refined Qualitative Agent **RQA**) fut également proposé par [Trave-Massuyes *et al.* 99], afin notamment de tenir compte des diamètres des conduites. Nous avons implémenté cet agent.

Tout d'abord, les tendances qualitatives sont plus précises : elles possèdent cinq valeurs : **INCL** (comme *increase a lot*) , **INCS** (comme *increase slightly*) **DECL** (comme *decrease a lot*), **DECS** (comme *decrease slightly*), **STD** (comme *steady*).

Pour cela nous définissons pour chaque réservoir le rapport :
$Q_i(t)$ = somme des diamètres Φ_e des conduites (avec écoulement) arrivant au réservoir i / somme des diamètres Φ_s des conduites (avec écoulement) sortant du réservoir i.

$Q_i(t) = \Sigma\Phi_e/\Sigma\Phi_s$	Tendance
$Q_i(t) \geq q_1$	$\partial\, h_i(t)$ = INCL
$q_2 < Q_i(t) < q_1$	$\partial\, h_i(t)$ = INCS
$Q_i(t) \leq q_3$	$\partial\, h_i(t)$ = DECL
$q_3 < Q_i(t) < q_2$	$\partial\, h_i(t)$ = DECS
$Q_i(t) = q_2$	$\partial\, h_i(t)$ = STD

Tableau 7 : Tendances qualitatives selon l'opérateur RQA

Les constantes utilisées ici (q_1 ; q_2 ; q_3) peuvent être considérées comme des paramètres du modèle et peuvent être réglées en fonction de l'opérateur humain auquel on compare l'agent qualitatif, mais les valeurs (***$q_1 = 3$; $q_2 = 1$; $q_3 = 1/3$***) par défaut furent proposées par [Trave-Massuyes *et al.* 99] comme semblant correspondre à celles d'un humain pour discriminer les situations.

Ainsi, la notion d'alarme est modifiée : on parle d'alarme forte **AlL** (hauteur *HIGH* et tendance *INCL* ou hauteur *FULL* et tendance *STD*) ou d'alarme faible **AlS** (hauteur *HIGH* et tendance *INCS*).

Dans le cas avec alarme, la prédiction qualitative à un pas (étape 2) est légèrement différente, ainsi que le choix final (étape 3) :

Étape 2* : prédiction qualitative à un pas.
on va distinguer, pour chaque action admissible, les deux réservoirs en amont et en aval de la vanne actionnée, pour lesquels on effectue une prédiction de la tendance qualitative $Q_i(t+1)$ qui résulterait à l'instant $t + 1$ si on réalisait l'action, et les autres réservoirs influencés par cette action, pour lesquels on effectue la somme des influences indirectes *S_infl* (voir tableau 6), de manière similaire au calcul effectué à l'étape 2 du paragraphe 3.2.2..

S_infl > 0	S_infl < 0
INCS → INCS+	INCS → INCS-
AlS → AlS+	AlS → AlS-
INCL → INCL+	INCL → INCL-
AlL → AlL+	AlL → AlL-

Tableau 6 : Prédiction de tendance pour l'opérateur RQA

Étape 3* : choix de l'action.
Le calcul des quatre critères diffère légèrement :

- si l'action génère n_1 nouvelles *AlL* et n_1' *AlL+*, $G_1 = -(n_1 + n_1')$

- si n_2 = nombre de *AlS* éliminées, n_3 = nombre de *AlL* éliminées,
  n_4 = nombre de *AlS-*, n_5 = nombre de *AlL-*,
  n_6 = nombre de nouvelles *AlS*, n_7 = nombre de nouvelles *AlL*,
  n_8 = nombre de *AlS+* et n_9 = nombre de *AlL+*,
  alors $G_2 = n_2 + n_3 + n_4 + n_5 - n_6 - n_7 - n_8 - n_9$

- si le réservoir "puit" est dans l'état *INCL*, *INCL+* ou *INCL-*, alors $G_3 = 2$
  si il est dans l'état *INCS*, *INCS+* ou *INCS-*, alors *G3 = 1*
  si il est dans l'état *STD*, alors $G_3 = 0$

- G_4 = variation du nombre de chemins ouverts (peut être positif ou négatif).

3.2.4. Apprentissage

A la suite de ces travaux menés par [Trave-Massuyes *et al.* 99], nous avons souhaité améliorer les capacités de l'agent qualitatif. Or il se trouve que les paramètres du modèle sont fixés de manière arbitraire alors qu'ils conditionnent le comportement de l'agent de façon importante. Nous avons donc étudié la mise en place d'un apprentissage des paramètres numériques du modèle [Rich 87], qui sont le seuil en volume ΔV^{alarme}, les poids p_i (i = 1 à 4) de pondération des gains G_i (i = 1 à 4) des actions admissibles en cas d'alarme, et les constantes q_i (i = 1 à 3) permettant le calcul des tendances. Nous avons choisi de nous intéresser aux poids de pondération p_i (i = 1 à 4), donc à un apprentissage uniquement en cas d'alarme.

Deux types d'apprentissage peuvent alors être envisagés.

Apprentissage supervisé

A la fin d'une simulation, un "professeur" (un utilisateur humain) intervient pour indiquer au système qu'à l'instant t ce dernier aurait du effectuer telle action sur telle vanne.

Si le système a préalablement enregistré pour chaque instant et pour chaque action admissible les gains intermédiaires G_i, alors le vœu du professeur se traduit par une contrainte :

$$\sum_i p_i.G_i^M \leq \sum_i p_i.G_i^B$$

G_i^M sont les gains intermédiaires concernant la **M**auvaise action, celle réalisée par l'agent

G_i^B sont les gains intermédiaires concernant la **B**onne action, celle indiquée par le professeur

Il est alors utile de conserver toutes les contraintes de ce type correspondant chacune à un apprentissage, afin d'éviter des conflits entre les différentes phases d'apprentissage, mais un problème peut se poser au niveau de la taille lorsque le nombre de contraintes augmente.
Le problème est ici la résolution d'un système de contraintes (CSP) [Ginsberg *et al.* 96] [Bartak 98] :

$$\sum_i p_i = 1$$

$$1 > p_1 > p_2 > p_3 > p_4 > 0$$

$$\sum_i p_i.G_i^{Mj} \leq \sum_i p_i.G_i^{Bj} \quad \text{pour j = 1 à nombre d'apprentissages}$$

Domaine (p_i) =] 0 ; 1 [

Ce type d'apprentissage n'a cependant pas été implémenté.

Apprentissage non supervisé

Le système doit pouvoir apprendre de lui-même sans aucune intervention extérieure. Le système doit donc trouver un ensemble d'actions concurrentes à celle qu'il a réalisé à un instant *t*, et va ensuite juger par le biais d'une simulation laquelle est la meilleure grâce à un critère. Nous avons implémenté la méthode suivante :

● L'ensemble des actions concurrentes est :

- l'action qui maximise le Gain global G ($= \sum_i p_i.G_i$) et qui est donc l'action que réaliserait le système sans autre information

- l'action qui maximise G_1 (puis G en second critère)

- l'action qui maximise G_2 (puis G en second critère)

- l'action qui maximise G_3 (puis G en second critère)

- l'action qui maximise G_4 (puis G en second critère)

● On simule chacune de ces actions, puis on laisse le réseau se vidanger dans cet état, c'est-à-dire que l'on n'agit plus sur aucune vanne. On note en fin de simulation le volume total débordé $V_{déb}$ et la durée totale D_t. Le critère choisi pour juger la qualité d'une simulation est une fonction linéaire pondérée du volume total $V_{déb}$ débordé et de la durée totale de la simulation D_t : $\boldsymbol{C = V_{déb} + \alpha.D_t}$ (voir 7.1.2), qu'il faut bien sûr minimiser. Cela permet de comparer l'effet de chacune des actions à partir d'un état donné du système. Le choix de la pondération α est fonction du type de problèmes étudiés, mais demeure arbitraire. Cependant, une étude des configurations testées montre qu'un choix raisonnable est de considérer que 1 m^3 débordé est sensiblement équivalent à 20 secondes perdues, soit $\alpha = 1/20$. On choisit donc l'action qui minimise le critère C. Si cette action n'est pas celle qui maximise G, alors il faut modifier les poids p_i pour que ce soit le cas. Il s'agit donc ici de mettre en place une procédure d'apprentissage de paramètres numériques [Rich 87]. Une façon simple de faire est d'augmenter les p_i (par exemple $p_i\,(t) = p_i\,(t\text{-}1) + \varepsilon.(p_{i-1}\,(t\text{-}1) - p_i\,(t\text{-}1))$ avec $p_i\,(t)$ le poids p_i après apprentissage, $p_i\,(t\text{-}1)$ le poids p_i avant apprentissage, et $\varepsilon \in]0;1[$), afin de respecter la contrainte $p_1 > p_2 > p_3 > p_4$) si le terme G_i de l'action choisie (critère C) est supérieur au terme G_i correspondant de l'action qui maximise G, et inversement de diminuer p_i (par exemple $p_i\,(t) = p_i\,(t\text{-}1) - \varepsilon.(p_i\,(t\text{-}1) - p_{i+1}\,(t\text{-}1))$) si le terme G_i de l'action choisie est inférieur au terme G_i correspondant de l'action qui maximise G. Il faut ensuite normaliser les p_i pour respecter la contrainte $\sum_i p_i = 1$.

Remarque : il serait utile, comme dans la situation précédente, de conserver toutes les contraintes, mais cela n'a pas été implémenté.

3.2.5. Tests des agents qualitatifs par des expérimentations informatiques

Nous avons implémenté les deux agents (*NQA* et *RQA*) en langage C et les avons testé avec diverses configurations du micro-monde (voir figure 2).

● Nous avons tout d'abord expérimenté les agents qualitatifs lors de la résolution, et il ressort que les agents qualitatifs possèdent une stratégie très générale, souvent locale (prise en compte d'un seul réservoir dans une situation avec alarme), n'anticipent pas suffisamment, et ne savent pas gérer des situations spécifiques :

- Par exemple, dans une situation multi-alarme, il peut être nécessaire d'appliquer une stratégie spécifique tenant compte de la configuration de la partie concernée.

- L'opérateur qualitatif fait parfois de mauvais choix d'actions dans une situation avec une seule alarme : par exemple, il peut arriver que cet agent ferme d'abord la vanne de plus petit diamètre en amont d'un réservoir en alarme, et ensuite la vanne de gros diamètre lorsque l'alarme persiste. Cet ordre d'action entraîne un débordement qui aurait pu être évité si l'ordre des actions avait été inversé. Ainsi, dans une telle situation, des valeurs des paramètres différents permettraient peut-être de résoudre ce problème. Ces modifications de valeurs pourraient être mises en œuvre grâce à une des procédures d'apprentissage citées ci dessus. Il est à noter que nous avons traité ce problème avec l'agent artificiel basé sur des cas, décrit au chapitre 4, en adoptant un comportement consistant à fermer la vanne de plus gros diamètre si la tendance du réservoir en alarme est *INCL* et à fermer la vanne de plus petit diamètre si la tendance du réservoir en alarme est *INCS* (voir 5.2.3. : index n° 3).

- Une autre situation est celle où il faut anticiper un débordement lorsqu'il y a une conduite sans vanne :

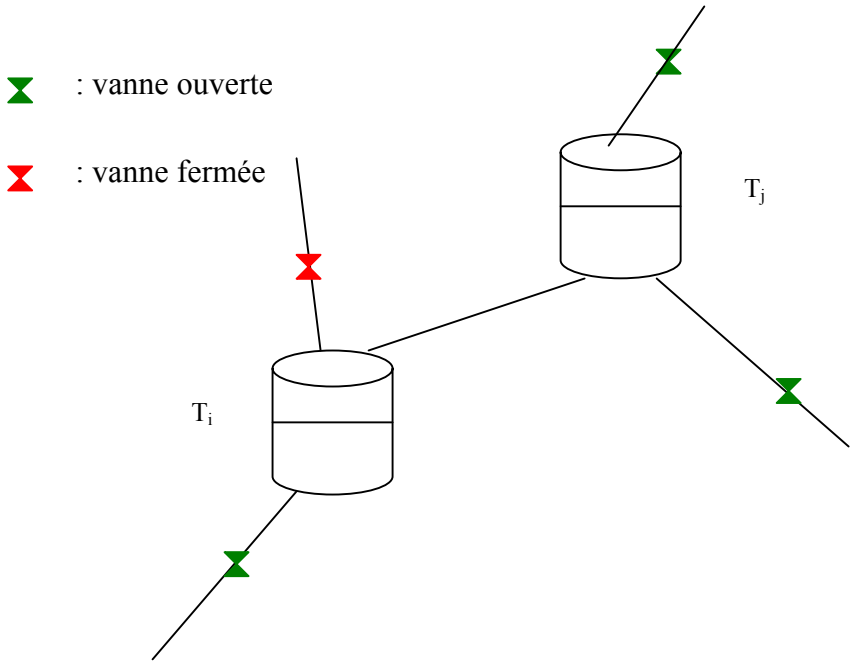


Figure 8 : Exemple de configuration nécessitant une stratégie globale

On voit clairement que si l'on veut empêcher le débordement du réservoir T_i, cela nécessite de fermer la vanne en amont du réservoir T_j suffisamment à l'avance, car on ne peut pas empêcher l'écoulement de T_j vers T_i.

- Enfin, minimiser la durée totale de la simulation peut nécessiter des stratégies fines prenant en compte un ensemble de réservoirs.

Nous détaillerons ces trois situations nécessitant des stratégies spécifiques dans le chapitre consacré au raisonnement à base de cas, et nous montrerons que ce type de raisonnement peut pallier les inconvénients du raisonnement qualitatif mis en œuvre ici. Nous effectuerons également une comparaison entre les différents agents artificiels.

● Nous avons testé l'apprentissage non supervisé, et cela a montré également des défauts.

- Il ressort tout d'abord que les quatre gains intermédiaires G_i ne discriminent pas suffisamment les actions possibles. Il n'est pas rare en effet que les G_i relatifs à l'action a qui maximise le gain global ($G = \Sigma_i\, p_i.G_i$) soient tous supérieurs ou égaux aux G'_i correspondant et relatifs à une autre action a' de score global inférieur ($G' < G$; $G' = \Sigma_i\, p_i.G'_i$ et $G_i \geq G'_i$ pour i = 1 à 4). On observe aussi régulièrement que deux actions a et a' possèdent non seulement le même gain global ($G = G'$), mais également les mêmes gains intermédiaires ($G_i = G'_i$ pour i = 1 à 4). Dans ces deux cas, l'apprentissage par modification des poids p_i est impossible, puisque si l'on souhaite que G' devienne strictement supérieur à G, il est nécessaire (mais non suffisant) qu'au moins un terme G'_i soit strictement supérieur au terme G_i correspondant. Ainsi, il faudrait approfondir le calcul des G_i afin qu'ils discriminent davantage les actions. Par exemple, les G_i ne tiennent pas suffisamment compte des diamètres des conduites, notamment au niveau de la vanne actionnée, car il a été remarqué que deux

actions similaires (fermer en amont d'un réservoir en alarme mais relatives à des conduites de diamètres très différents) n'engendraient cependant pas de différences au niveau des G_i.

- Il ressort également lors d'un test de l'opérateur RQA que les valeurs choisies pour les paramètres q_i ne sont pas forcément les meilleures. Donc là aussi, il serait judicieux de mettre en place une procédure d'apprentissage de ces trois paramètres.

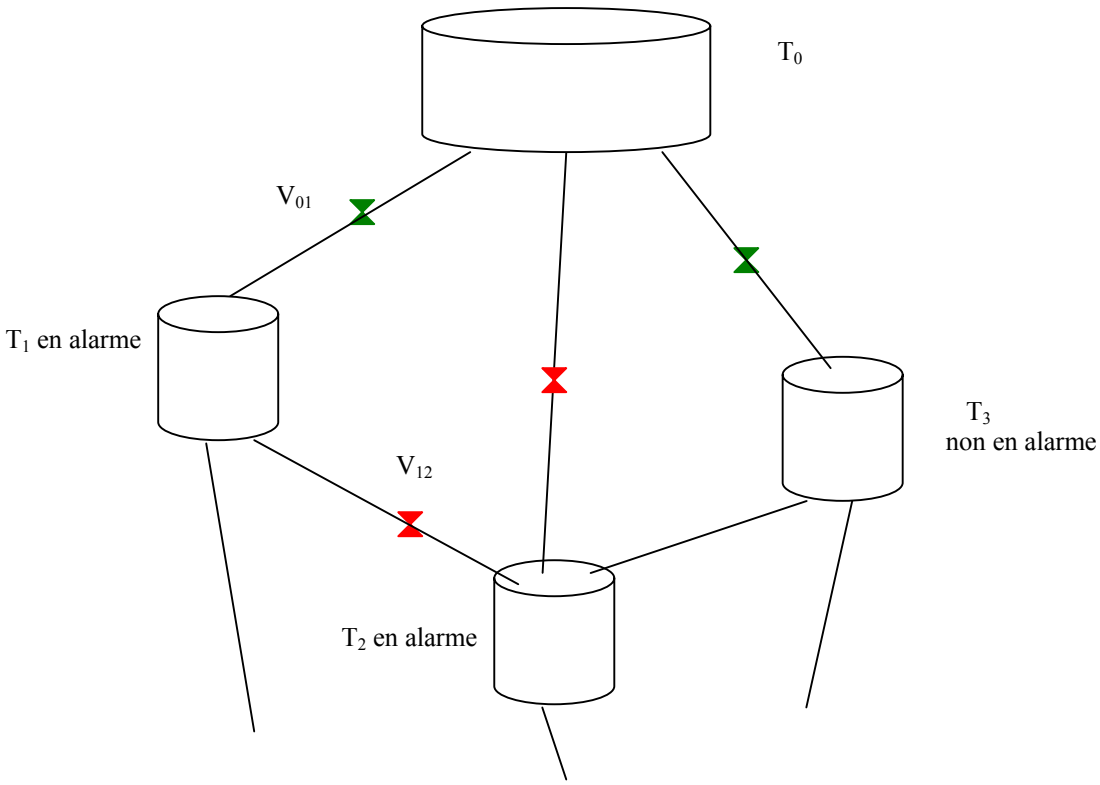


Figure 9 : Etat du réseau associé à une mauvaise action de l'opérateur RQA

En effet, dans la situation de la figure 9 comportant deux réservoirs en alarme, l'opérateur qualitatif RQA calcule deux actions possibles (fermer V_{01} et ouvrir V_{12}) avec :

	G_1	G_2	G_3	G_4
V_{12}	0	1	2	1
V_{01}	0	0	2	-1

Tableau 8 : Gains associés aux valeurs q_i = (1/3 ; 1 ; 3) des paramètres de l'opérateur RQA

L'opérateur qualitatif choisit d'ouvrir V_{12} (gain G maximum), mais il serait préférable de fermer V_{01}, ce qui est impossible à la vue des valeurs des gains G_i (i = 1 à 4).
Pour V_{01}, $G_2 = 0$ car il y a réduction de l'alarme sur T_1 et augmentation de l'alarme sur T_2 par influence indirecte. Pour V_{12}, $G_2 = 1$ car il y a réduction de l'alarme sur T_1 mais il n'y a pas augmentation de l'alarme sur T_2 car la tendance prédite de T_2 (rapport $\Sigma\Phi_e/\Sigma\Phi_s$) reste *INCS* malgré l'ouverture de V_{12}. Il y a donc clairement un manque de cohérence entre les prédictions de tendance par calcul des influences indirectes et les prédictions de tendance par calcul du rapport des diamètres entrant et sortant (avec écoulement). Il semble donc que la

valeur $q_3 = 3$ qui représente la limite *INCS* ↔ *INCL* soit trop élevée. Un test avec $q_3 = 2$ a donné les résultats suivants :

	G_1	G_2	G_3	G_4
V_{12}	-1	0	2	1
V_{01}	0	0	2	-1

Tableau 9 : Gains associés aux valeurs q_i = (1/3 ; 1 ; 2) des paramètres de l'opérateur RQA

On observe alors qu'avec $q_3 = 2$, un apprentissage des p_i est possible afin que l'opérateur choisisse de fermer V_{01} (par exemple, $p_1 = 0.5$ $p_2 = 0.25$ $p_3 = 0.2$ $p_4 = 0.05$).

Pour résumer, lorsqu'il s'agit de discriminer deux actions, si l'apprentissage par modification des valeurs des poids p_i (i = 1 à 4) est impossible, alors il serait judicieux de tenter préalablement un apprentissage par modification des valeurs des paramètres q_i (i = 1 à 3). Bien sûr, les valeurs de ces trois paramètres doivent rester dans certaines limites, puisque les valeurs ($q_1 = 3$ $q_2 = 1$ $q_3 = 1/3$) proposées par [Trave-Massuyes *et al.* 99] semblent correspondre à celles d'un humain pour discriminer les situations.

- Il est à noter enfin que la manière de juger chaque action potentielle pose un problème : simuler une action puis laisser le réseau se vidanger peut s'avérer impossible dans la configuration suivante :

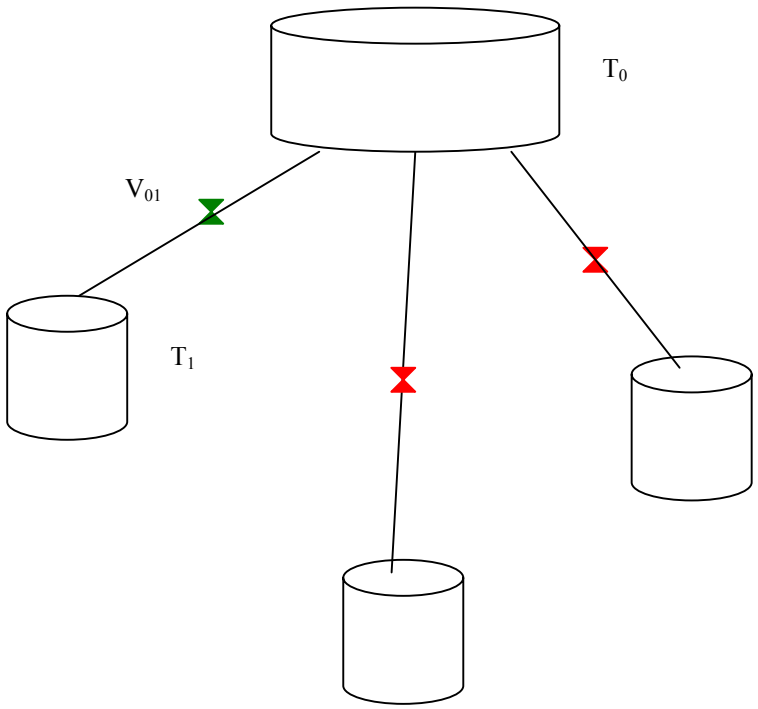


Figure 10 : Simulation de la vidange après fermeture de la vanne V_{01} impossible

Si on doit simuler l'action consistant à fermer V_{01}, alors toutes les vannes en aval du réservoir du haut sont fermées et la vidange est impossible.

De manière générale, dans le cas de l'apprentissage non supervisé, le problème de savoir à posteriori, c'est-à-dire à la fin d'une simulation (un test d'un agent ayant reçu une consigne :

voir p. 7 et chap. 8.1), éventuellement avec l'aide d'un signal de renforcement (positif ou négatif), quelles sont les actions du système qui n'ont pas été bonnes est un problème très délicat, connu en Intelligence Artificielle sous le nom de *problème d'attribution du mérite* [Rich 87], mis en évidence dans les programmes de jeu : à la fin d'une partie, le système reçoit un signal (partie gagnée ou perdue), et si la partie est perdue, alors il doit trouver quelles actions ont contribué à ce mauvais résultat. A ma connaissance, ce problème reste ouvert.

	$V_{déb}$ (m3)	D_t (sec.)	C
Test 1 : Config 3 - NQA ΔV^{alarme} =10 p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	7.98	576	36.8
Test 2 : Config 3 - RQA ΔV^{alarme} =10 q_i = (1/3 ; 1 ; 3) p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	0	586	29.3
Test 3 : Config 3 - RQA ΔV^{alarme} =10 q_i = (1/3 ; 1 ; 2) p_i = (0.5 ; 0.25 ; 0.2 ; 0.05)	0	586	29.3
Test 4 : Config 3 - NQA ΔV^{alarme} =1.5 p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	4.35	582	33.45
Test 5 : Config 3 - RQA ΔV^{alarme} =1.5 q_i = (1/3 ; 1 ; 2) p_i = (0.5 ; 0.25 ; 0.2 ; 0.05)	4.48	586	33.8
Test 6 : Config 1 - NQA ΔV^{alarme} =10 p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	0.95	774	39.7
Test 7 : Config 1 - RQA ΔV^{alarme} =10 q_i = (1/3 ; 1 ; 3) p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	7.37	726	43.7
Test 8 : Config 1 - RQA ΔV^{alarme} =10 q_i = (1/3 ; 1 ; 2) p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	2.53	752	40.1
Test 9 : Config 1 - RQA ΔV^{alarme} =10 q_i = (1/3 ; 1 ; 2) p_i = (0.5 ; 0.25 ; 0.2 ; 0.05)	0	780	39
Test 10 : Config 1 - NQA ΔV^{alarme} =1.5 p_i = (0.4 ; 0.3 ; 0.2 ; 0.1)	9.11	722	45.2
Test 11 : Config 1 - RQA ΔV^{alarme} =1.5 q_i = (1/3 ; 1 ; 2) p_i = (0.5 ; 0.25 ; 0.2 ; 0.05)	4.41	754	42.1

Config i : une configuration du micro-monde
NQA : naïve qualitative agent
RQA : refined qualitative agent
$V_{déb}$ = volume total débordé
D_t = durée de la simulation
$C = V_{déb} + \alpha.D_t$ = objectif global à minimiser ($\alpha = 1/20$)
ΔV^{alarme} = volume d'alarme : si $V_i(t) > (V_i^{max} - \Delta V^{alarme})$ alors alarme du réservoir T_i à l'instant t
Δt = pas de temps de calcul des agents artificiels = 2 secondes

Tableau 10 : Résultats des tests des opérateurs qualitatifs

Chaque ligne du tableau 9 correspond à une simulation d'un agent qualitatif (NQA ou RQA), avec des valeurs de paramètres fixées, et dans une configuration de réseau donné.

Le tableau 9 montre que pour une même configuration de réseau et une même valeur du paramètre ΔV^{alarme}, l'opérateur qualitatif RQA avec les paramètres q_i = *(1/3 ; 1 ; 2)* et p_i = *(0.5 ; 0.25 ; 0.2 ; 0.05)* obtient des résultats meilleurs ou comparables à ceux de l'opérateur NQA ou de l'opérateur RQA avec des valeurs des paramètres q_i (i = 1 à 3) et/ou p_i (i = 1 à 4) différentes (voir les lignes en gris clair par rapport aux lignes blanches situées au dessus).
L'opérateur qualitatif *NQA* n'obtient pas de bons résultats sur la configuration 3 avec $\Delta V^{alarme} = 10$ car il gère mal les situations multi-alarmes qui se présentent. Nous verrons que l'agent basé sur des cas sait gérer ces situations. En effet, dans la même configuration du réseau, aucun débordement ne se produit. Si $\Delta V^{alarme} = 1.5$, alors les situations multi-alarmes sont beaucoup moins fréquentes, d'où de meilleurs résultats. Mais on note que l'influence du seuil d'alarme est inversée, pour cet opérateur, dans la configuration 1 qui comporte une conduite sans vanne : un seuil trop faible entraîne une incapacité à anticiper le débordement du réservoir en aval de la conduite sans vanne.
L'opérateur RQA obtient quant à lui de meilleurs résultats lorsque le volume d'alarme est élevé, quelle que soit la configuration du réseau.

En conclusion, nous pouvons effectuer une analogie entre les agents qualitatifs et le modèle de Rasmussen (voir 2.2) qui est essentiellement réactif. Nous allons maintenant détailler le développement d'une autre approche : l'agent à base de cas. Auparavant, présentons les principes du raisonnement à base de cas.

Chapitre 4

RAISONNEMENT A BASE DE CAS

Nous présentons les principes et un état de l'art du raisonnement à base de cas : cela consiste à utiliser ses expériences passées (des cas) pour résoudre un problème, puis à apprendre à partir du problème résolu, et enfin à stocker en mémoire cette nouvelle expérience.

4.1. INTRODUCTION

Le raisonnement par analogie consiste, pour résoudre un problème, à aller rechercher en mémoire, sur la base d'indices (les mots d'un texte, les formes visuelles, …), les expériences passées qui permettront de comprendre puis résoudre ce problème. L'intérêt de cette approche est d'être proche d'une certaine forme du raisonnement humain. Nous allons examiner une approche possible de ce type de raisonnement : le raisonnement à base de cas (CBR *: Case-Based Reasoning*).

Un système de raisonnement à base de cas intègre deux aspects essentiels : il est susceptible d'apprendre à partir de ses expériences (un problème résolu correspond à un "cas" et est conservé en mémoire), et il utilise ses expériences passées pour résoudre des problèmes (le système recherche en mémoire un cas passé jugé similaire et le réutilise après l'avoir adapté à la nouvelle situation en recherchant les différences entre les deux situations). Le terme résolution de problèmes est vu ici au sens large : il peut s'agir de justifier un raisonnement ou un argument, de critiquer ou expliquer une solution fournie par l'utilisateur, d'interpréter une situation.

Plus précisément, voyons quelles sont les tâches d'un système CBR (figure 11) : un système peut être analysé selon trois perspectives : les tâches, les méthodes et les connaissances du domaine [Aamodt et Plaza 94]. Les tâches sont liées aux buts à atteindre, les méthodes permettent d'accomplir les tâches, et les méthodes nécessitent des connaissances (les connaissances générales du domaine et les informations sur le problème courant et son contexte).
Du point de vue des tâches, on peut décomposer le CBR comme indiqué sur la figure 11. Les tâches sont en gras et les méthodes en italique. Les méthodes reliées à une même tâche (en traits pointillés) sont alternatives. L'ensemble des méthodes n'est pas exhaustif, et certaines d'entre elles peuvent être combinées. On retrouve les quatre phases principales : recherche, réutilisation, révision et apprentissage. Chacune de ces phases peut être à nouveau décomposée.

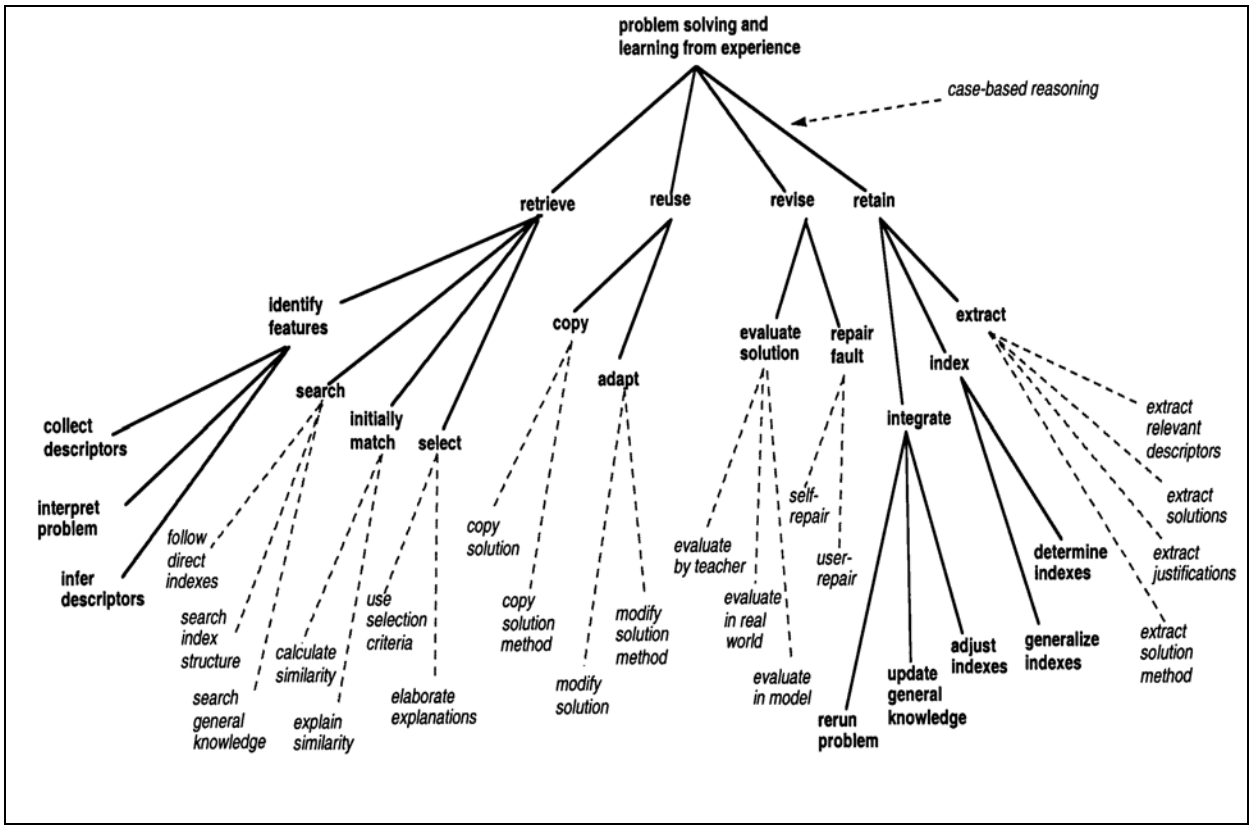


Figure 11 : Analyse des tâches et des méthodes associées pour un système CBR [Aamodt et Plaza 94]

Concernant la représentation d'un cas, celui-ci peut être enregistré comme un objet à part entière, ou bien être séparé en plusieurs unités distribuées au sein du système CBR. En outre, il est supposé que le cas possède une certaine complexité quant à son organisation interne. Par exemple, un cas sous la forme d'un vecteur de valeurs numériques ne correspond généralement pas au paradigme du CBR. Un autre aspect important est l'indexation du cas : un index représente les aspects essentiels permettant de retrouver un cas en mémoire. Les cas peuvent être indexés grâce à un vocabulaire pré-fixé ou bien ouvert, ou grâce à une structure d'index, hiérarchique ou non.

L'apprentissage d'un nouveau cas résolu consiste généralement à retenir le cas concret résolu, ce qui nécessite d'identifier les caractéristiques importantes et discriminantes du problème. En outre, certains travaux, par exemple ceux de [Bergmann et Wilke 97], ont tenté de généraliser à partir du nouveau cas.

En cas d'échec de résolution, la raison de cet échec peut être identifiée et remémorée pour ne plus commettre la même erreur dans le futur.

De plus en plus, les systèmes CBR sont combinés avec d'autres types de raisonnement [Mantaras et Plaza 97], par exemple un système à base de règles (RBR), un système à base de modèles (MBR), … Ces derniers constituent alors les connaissances générales, profondes, utilisées par le système CBR lors des phases de recherche de cas en mémoire, d'adaptation, ou d'apprentissage. On peut également concevoir toutes ces techniques comme alternatives, l'échec d'une d'entre elles conduisant à la tentative d'utilisation d'une autre.

Enfin, un système CBR opère en plusieurs phases bien distinctes (figure 12). Il y a un ordre à respecter entre ces phases, ainsi que des cycles possibles à partir de la phase de validation vers la phase d'indexation, ou de recherche, ou d'adaptation. En outre, une intervention de l'utilisateur peut être nécessaire si le système n'a pas les connaissances suffisantes pour résoudre un problème nouveau :

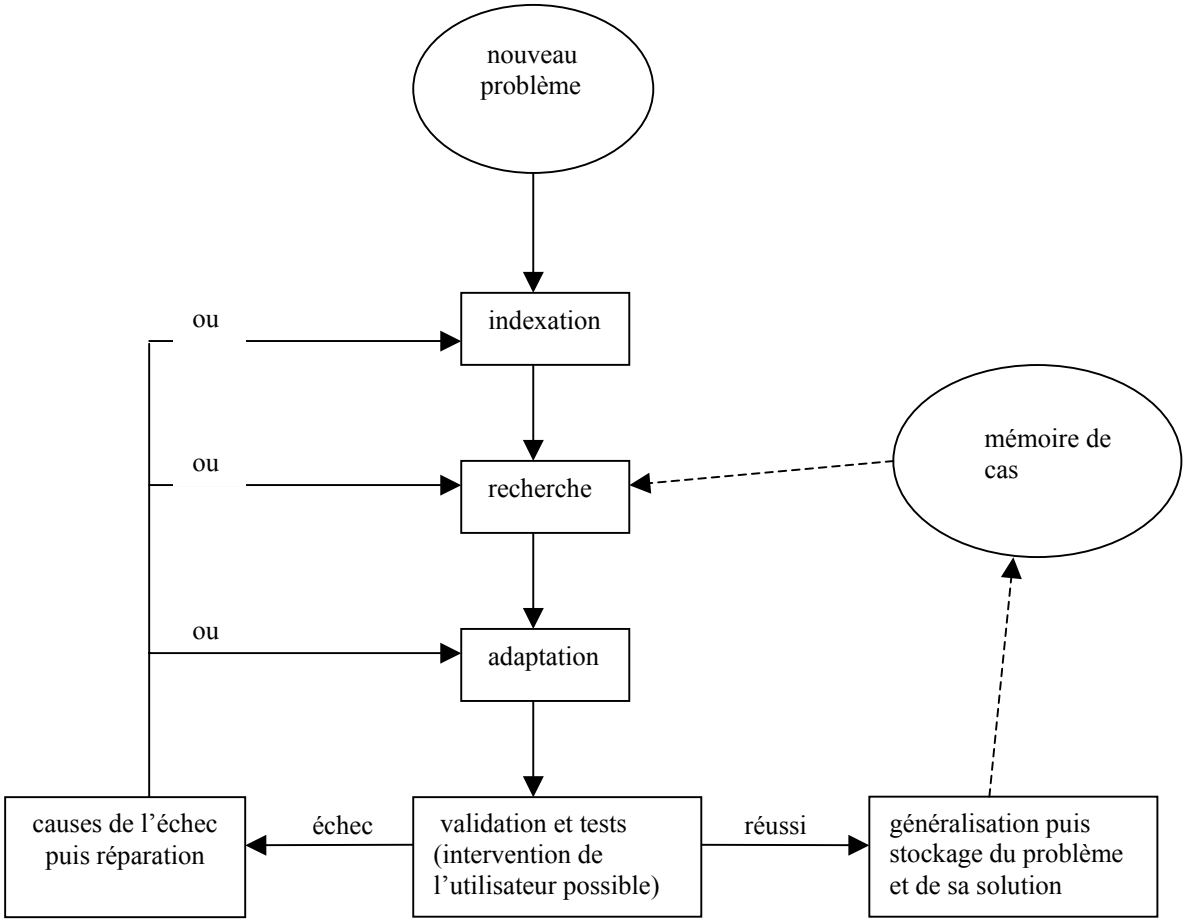


Figure 12 : Schéma général de fonctionnement d'un système CBR

4.2. DESCRIPTION GENERALE

4.2.1. Modélisation

4.2.1.1. Les cas

Un cas est représenté par les données du problème (état initial), la solution (état final), les contraintes et le chemin (les pas intermédiaires) permettant de passer de l'état initial à l'état final. Il est clair que plus le cas contient d'informations, plus le raisonnement sera fin, mais aussi plus il sera long et délicat à mettre en œuvre.

Un cas représente donc l'énoncé d'un problème et sa solution. L'énoncé et sa solution peuvent être considérés comme deux entités séparées. Cependant, il est préférable que l'énoncé soit relié par un "chemin" à sa solution car cela permettra une adaptation plus fine

(notion d'"analogie dérivationnelle" introduite par Jaime Carbonnell [Schmid et Carbonell 99] [Blumenthal et Porter 94]).

On peut également imaginer que soient présentes des méta-connaissances relatives au mode d'emploi et à des explications, ainsi qu'au contexte du problème, afin de faciliter les phases de recherche et d'adaptation. Ces méta-connaissances sont intégrées au sein du cas ou de l'index. De plus, il peut être fort utile d'ajouter des connaissances négatives indiquant dans quelles situations ne pas utiliser un cas.

Ainsi, à chaque cas est associé un ***index*** qui est composé d'indices, représentant les caractéristiques importantes et discriminantes du problème, son contexte, les contraintes, le but atteint, la manière d'atteindre ce but, des explications, et permettant de retrouver, utiliser et adapter de façon appropriée le cas en mémoire. Les cas et les index ne sont pas forcément représentés dans le même formalisme. Si les deux formalismes sont différents, il peut être nécessaire dans certains cas de définir "l'adéquation" entre les deux. Par ailleurs, les caractéristiques des cas et des index peuvent également être associées à des poids (numériques, flous) reflétant leur importance et qui seront utilisés lors de la mise en correspondance (phase de recherche de cas similaires).

Il est donc important de faire la distinction entre le cas (expérience, procédure, scénario, règle) et l'index du cas (informations déclaratives pour organiser, rechercher et utiliser le cas).

4.2.1.2. L'organisation de la base de cas

Les cas peuvent être regroupés en concepts, ces concepts étant organisés sous forme d'une liste ou d'une hiérarchie. Chaque concept est défini en extension (par l'ensemble de ses membres) et éventuellement par intention (par l'ensemble de ses propriétés représentant des conditions nécessaires et suffisantes)

Des modifications peuvent avoir lieu lors de l'apprentissage, soit par ré-organisation de la hiérarchie des index lors de l'intégration du nouveau cas, soit par abstraction de deux cas similaires (fusionnement d'un cas ancien et du cas nouveau).

Historiquement, citons deux approches possibles :

● ***le modèle basé sur des exemples*** [Aamodt et Plaza 94] (figure 13) : un concept est défini en extension et en intention. Ainsi, à chaque catégorie (concept) sont associés des caractéristiques (les indices) et des exemples (les cas). A chaque caractéristique est associé un poids reflétant son importance au sein de la catégorie. A chaque exemple est associée son importance en tant que prototype de la catégorie. En outre, on peut éventuellement associer à un exemple des caractéristiques propres (donc discriminantes par rapport à d'autres exemples de la même catégorie). De plus les exemples proches (dans le sens où ils ne diffèrent que par quelques caractéristiques) d'une catégorie sont reliés. En outre les catégories sont reliées au sein d'un réseau sémantique, qui peut également contenir d'autres connaissances (les buts, le contexte, …) et peut donc être vu comme un réseau de connaissances générales du domaine.

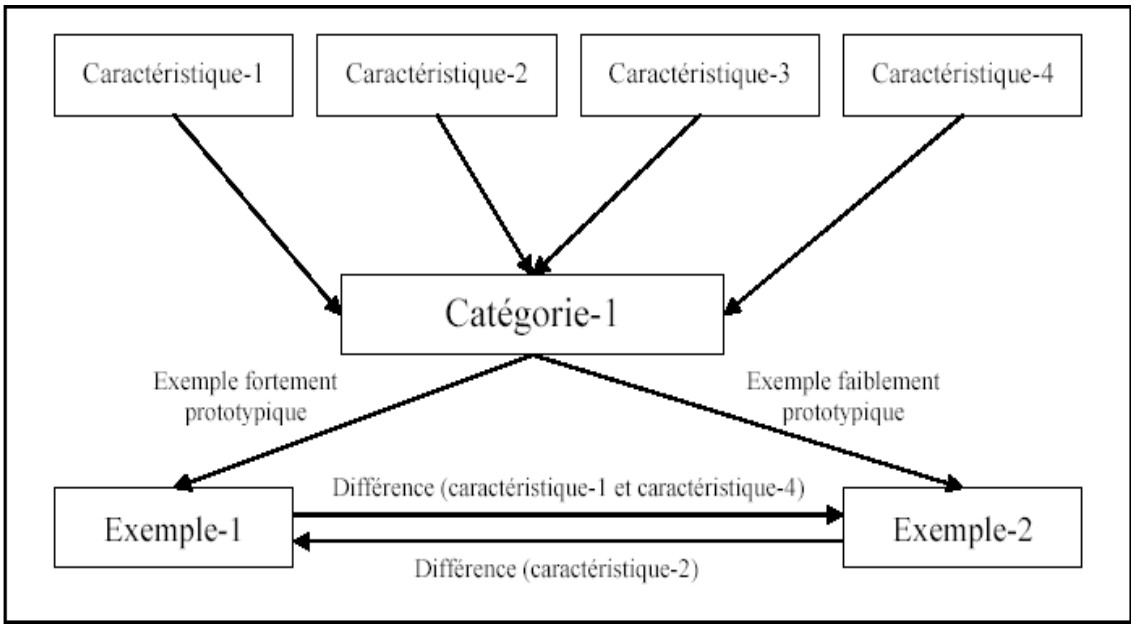


Figure 13 : Modèle d'organisation de la mémoire basé sur les exemples [Aamodt et Plaza 94]

● ***le modèle de la mémoire dynamique*** [Mantaras et Plaza 97] [Aamodt et Plaza 94] (figure 14) : il s'agit à l'origine d'un concept défini par Roger Schank [Schank 99][1]. Les cas possédant des similarités sont placés sous une même structure générale (un épisode généralisé ou EG). Un EG contient des cas, des *normes* (données communes à tous les cas organisés sous cet EG) et des *index* (données discriminantes relatives aux divers cas d'un même EG). Un *index* est composé d'un nom et d'une valeur. Ainsi, la structure des *index* est un réseau de discrimination et un cas est enregistré sous un *index* qui le discrimine des autres cas.
La recherche de cas similaires se fait à partir de la racine de l'arbre : on recherche l'EG contenant le plus de *normes* en commun avec le nouveau cas, puis on parcourt les *index* de cet EG pour trouver le(s) cas le(s) plus similaire(s).
Lors de l'ajout d'un cas nouveau (on utilise les résultats de la recherche précédente si le cas n'a pas été modifié entre temps), un EG est créé si il existe un cas passé possédant des données similaires avec le cas nouveau. Il s'agit ensuite d'indexer (discriminer) les deux cas sous le nouvel EG. Si, lors de l'ajout d'un nouveau cas, deux cas (ou deux EG) sont deux feuilles associées à un même index, alors un EG est créé.
Ainsi, cette structure de mémoire dynamique peut être vue comme l'intégration de connaissances issues d'épisodes spécifiques (les cas) et de connaissances généralisées à partir de ces mêmes épisodes.

[1] Il s'agit de la seconde version, la première datant de 1982.

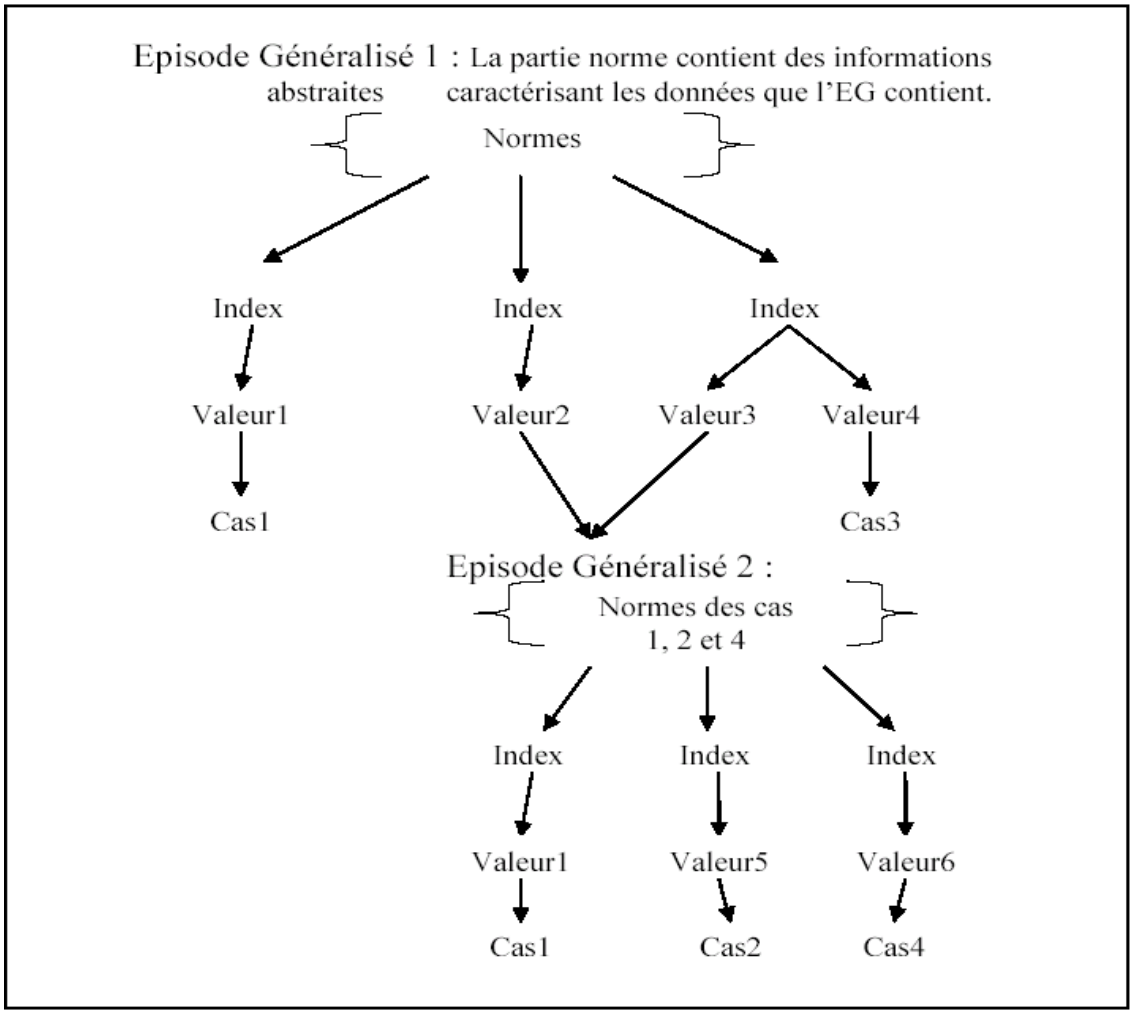


Figure 14 : Modèle d'organisation basé sur une mémoire dynamique

4.2.1.3. L'indexation des cas

Le but principal d'un index est de prédire l'utilité d'un cas et d'aider lors de la phase de recherche de cas similaires. Il doit représenter les buts que le cas peut atteindre ou aider à atteindre, ainsi que les circonstances, le contexte dans lequel le cas est utile pour chacun de ces buts. Un cas peut posséder plusieurs index, s'il existe par exemple plusieurs contextes à partir desquels le cas peut être utilisé. Il s'agit ensuite de généraliser cette description pour qu'elle soit applicable dans diverses situations futures. Un index comporte donc :

- des éléments descriptifs abstraits ne comportant que les caractéristiques essentielles

- des explications, utiles lors de la phase d'adaptation

- des conaissances liées au mode d'emploi, utiles lors de la phase de ré-utilisation

- des connaissances négatives issues d'échecs dans le passé

- le contexte (certains éléments de l'état initial, les contraintes)

- et des éléments discriminants (des éléments qui le différencient de cas similaires)

Une manière simple et usuelle de représenter le vocabulaire d'indexation est une modélisation d'un ensemble de dimensions descriptives (des attributs), et un ensemble de valeurs associées à chaque dimension [Kolodner et Leake 96]. Il est à noter qu'il ne s'agit pas forcément d'une liste de descripteurs, mais que ces derniers peuvent être combinés d'une certaine façon (par exemple sous la forme de relations ou prédicats).
Les index peuvent être organisés sous forme d'une hiérarchie de concepts indices. Un index peut comporter des liens vers d'autres index, ainsi que des liens vers des cas.

Il est à noter pour finir qu'un index peut être vu comme une abstraction d'un cas : [Bergmann et Wilke 97] représentent non seulement des cas concrets mais également des cas abstraits, éventuellement à plusieurs niveaux d'abstraction. La similarité peut alors être définie comme une égalité à un certain niveau d'abstraction. Les cas abstraits peuvent servir d'index, et permettent aussi d'obtenir une solution abstraite qui peut être ensuite adaptée (au même niveau d'abstraction) puis raffinée (vers le niveau concret).

4.2.1.4. Formalismes de représentation des cas et des index

Le choix du formalisme est lié au type de problème.

● ***Représentation objet***

Examinons dans un premier temps les caractéristiques de cette représentation pouvant être utiles dans le cadre du raisonnement à base de cas. Un objet possède diverses caractéristiques :

- des propriétés, sous la forme de couples (attribut-valeur) ou de formules logiques, et qui représentent donc des caractéristiques d'un cas ou de son index (état initial, but, contraintes, explications, éléments du contexte, …).

- des méthodes qui permettent de passer d'un état initial à un état final, et sont paramétrées par des éléments de l'état initial, des contraintes et des éléments du chemin entre les deux états (les deux derniers points permettent l'adaptation).

- des liens entre objets : entre les index (généralisation/spécialisation), entre les cas (différence) et entre les cas et les index (appartenance).

La notion d'objet en Intelligence Artificielle peut en outre faire appel à des concepts tels que les *prototypes* [Barsalou 91], ou les *frames* [Minsky 74] [Barsalou 92] (voir également les chapitres précédents 2.2.2.5 et 2.2.2.9) :

- un *prototype* possède des propriétés typiques. La relation entre un objet et un prototype n'est pas du type "tout ou rien" comme celle qui existe entre une instance et sa classe, mais du type "plus ou moins". En outre, toute instance est potentiellement un prototype pouvant être spécialisé à son tour [Haton 91], à la différence d'une instance d'une classe. Kerber [Kerber *et al.* 92] [Kerber *et al.* 93] a formalisé le raisonnement par analogie avec des exemples typiques.

- selon L. Barsalou, un *frame* est un objet composé de couples (attribut → valeur) représentant les propriétés de cet objet. Un attribut peut être représenté à son tour par un *frame*, ce qui permet de représenter les propriétés de l'attribut. Ainsi, un *frame* peut incorporer des *frames* à

l'intérieur d'elle-même, de manière récursive [Barsalou 93]. En outre, divers types de relations peuvent apparaître entre les attributs ou entre les valeurs [Barsalou 92]. Selon [Minsky 74] un *frame* est une structure de données représentant une situation stéréotypée, sous la forme d'un réseau de nœuds et de relations. Les *niveaux hauts* du *frame* représentent des propositions toujours vraies, les *niveaux bas* comportent des *terminaux* (ou *slots* ou attributs) qui doivent être affectés à des instances spécifiques ou à des données. Chaque *terminal* peut spécifier des conditions que doivent remplir les affectations (habituellement des *sous-frames*) sous la forme de contraintes ou de relations. Les attentes sont représentées par des valeurs par défaut. L'idée nouvelle de Minsky est l'organisation de collections de *frames* reliés entre eux au sein d'un *système de frames*, afin de représenter un scénario, ou bien différentes vues associées à une scène ou un objet physique. En outre, ces *systèmes de frames* sont intégrés dans un réseau représentant des liens de similarité, de différence, de généralité (hiérarchie), ainsi que d'autres liens nécessaires à la compréhension (conditions d'un but, effets d'une action, …). [Clark et Porter 01] ont appliqué la théorie des *frames* pour la construction d'un langage de représentation des connaissances : *KM* (*Knowledge Machine*). L'unité de base, le *frame*, contient des slots (attributs) et des valeurs. Une valeur peut être atomique, ou bien une expression qui réfère à d'autres *frames*. Cette expression peut être très simple (par exemple dans le cas où l'attribut *âge* est lié à l'expression *un entier*), ou plus complexe (par exemple l'attribut *sous-événement* du frame *achat* est lié à l'expression *un don dont l'agent est l'acheteur, l'objet est l'argent de l'achat et le réceptionneur est le vendeur*). Un attribut est une relation binaire entre une instance d'un *frame* et une autre instance représentée au niveau de la valeur. Les *frames*, ainsi que les attributs, peuvent être organisés en hiérarchies. Il est possible de représenter des contraintes sur une valeur et des expressions comportant des conditions ou des quantificateurs universels, d'*attacher* une procédure à un attribut, … Ainsi, le formalisme de représentation est équivalent à une logique du premier ordre, avec des mécanismes supplémentaires, notamment l'unification "lazy", qui reporte les détails d'unification à plus tard, lorsque les besoins de l'application rendront nécessaires l'obtention de ces détails. En effet, l'unification complète de deux *frames*, qui comportent des attributs qui sont eux-mêmes des *frames*, et ainsi de suite, peut être très coûteuse. Signalons pour finir les travaux de Anderson sur l'utilisation des *frames* dans le raisonnement, au sein de la théorie *PUPS* qui succède à la théorie *ACT* [Anderson 89]. A partir des faits provenant de la situation courante et déposés dans la mémoire de travail, un mécanisme d'activation progressive (*spreading activation process*) permet de retrouver dans la mémoire à long terme les *frames* (structure de connaissance déclarative) associés à ces faits. Au cours du raisonnement, les *frames* actifs (le degré d'activation décroît au cours du temps si le *frame* n'est pas utilisé) rendent les connaissances associées (notamment des règles qui sont en fait des *frames* tels que les attributs *forme* et *précondition* conjointement impliquent l'attribut *fonction*) disponibles pour une mise en adéquation (par exemple entre des *frames* et les conditions d'une règle). La sélection d'une règle dépend du coefficient de renforcement associé (fonction des dates d'utilisation de la règle) et des degrés d'activation des *frames* associés aux conditions. Ces principes ont été repris par Holyoak et Thagard lors de la mise en œuvre de l'architecture *PI* (*Processes of Induction*) qui permet de résoudre des problèmes par analogie à l'aide d'une représentation des connaissances orientée *frames* [Holyoak et Thagard 89]. Anderson présente également un mécanisme pour généraliser (grâce à deux principes inductifs permettant de substituer une constante une fonction par une variable appropriée) et discriminer (afin de restreindre la généralisation grâce à la recherche de contraintes et éviter ainsi la sur-généralisation) un *frame*.

Enfin, des mécanismes supplémentaires (par rapport à un langage objet classique) doivent généralement être mis en œuvre :

- la notion d'appartenance à une classe : l'appartenance à un prototype est du type plus ou moins et non du type tout ou rien [Haton 91]. En outre, il arrive que certains concepts doivent être définis par des propriétés de définitions (qui, si elles sont satisfaites, permettent de conclure à l'appartenance d'une instance à la classe), et des propriétés secondaires (qui ne permettent pas d'affirmer l'appartenance à une classe).

- la notion de classification et de reconnaissance, liée à la notion de similarité entre objets.

- la notion d'héritage : les liens d'héritage entre concepts peuvent être sûrs, probables, du type exception ou inhibition, et peuvent également faire appel à la notion de similarité.

Enfin, un avantage de cette représentation est que lorsque les index associés aux cas sont décrits par des conjonctions de couples (attribut-valeur), on peut organiser la base grâce aux techniques classiques de classification supervisée (induction en logique des proposition, arbre de décision), hiérarchique ou non [Cornuéjols et Miclet 02]. Dans le cas d'une hiérarchie, on utilise les notions de généralisation minimale (subsumant le plus spécifique) et de spécialisation maximale (subsumé le plus général) pour classer un concept [Haton 91].

Nous allons maintenant citer deux exemples intéressants d'approches à base d'objets :

- Signalons tout d'abord l'approche de J. Lieber et A. Napoli [Lieber et Napoli 99] (annexe 1) qui proposent une représentation par objets sous la forme de conjonction de couples (attribut-valeur) : ils définissent une classe (l'énoncé d'un problème) comme la conjonction $C = (a_1, s_1) \wedge \ldots \wedge (a_n, s_n)$ où les a_k désignent des propriétés (ou attributs) et les s_k des spécifications attachées à la propriété précisant par exemple le type, le domaine et la cardinalité des valeurs de la propriété. Par exemple a_k peut désigner l'âge d'une personne et s_k une condition $a_k > 40$. Ils définissent ensuite une hiérarchie de classes par le biais d'une relation de généralité : l'énoncé du problème P est plus général que l'énoncé du problème P' (noté $P \sqsubseteq P'$) si pour chaque couple attribut-spécification (a, s) dans P, il existe un couple attribut-spécification (a, s') dans P' tel que $s' \Rightarrow s$.

- Signalons également les travaux de A. Mille et B. Fuchs [Mille *et al.* 99] (annexe 2) qui appliquent le raisonnement à partir de cas dans le cadre de la supervision industrielle. Les connaissances sont représentées par :

· des objets de supervision qui constituent une cible pour l'opérateur dans ses choix.

· les concepts liés à l'expérience : les cas. Un cas est un épisode, et comporte un environnement de supervision de début (ES_d), une séquence d'actions, puis un environnement de supervision de fin (ES_f). On lui associe une *signature événementielle*, c'est-à-dire un ensemble d'évènements ayant précédé le début de l'épisode de supervision. On lui associe également une *signature conceptuelle*, c'est-à-dire l'ensemble des objets de supervision intervenant dans l'épisode, par l'intermédiaire des descripteurs des tableaux de bord, ou de la signature événementielle. Un cas est discriminé par un *objet focal* : si l'opérateur n'a pas de cible particulière en début d'épisode, une procédure d'exploitation de la signature événementielle fournit un objet focal potentiel (un objet de supervision très présent dans la signature événementielle et peu supervisé dans l'environnement courant de supervision).

● ***Formules logiques dans une représentation du type logique des prédicats***

Ce formalisme convient pour représenter des cas complexes ou bien pour représenter des connaissances de façon expressive et fine.
Si les énoncés des problèmes ou les index sont décrits par des formules de la logique des prédicats, la classification est nettement plus délicate que dans le cas précédent (couple attribut-valeur) car la notion de distance ou de similarité n'est pas naturelle, mais des travaux ont été réalisés. Citons trois approches possibles :

- la programmation logique inductive [Cornuejols et Miclet 02], basée sur un sous-ensemble de la logique du premier ordre. Il s'agit d'une technique de classification symbolique non hiérarchique pour l'apprentissage de concept.

- la logique temporelle [Haton 91] [Vila 92], ou des formalismes issus des travaux sur la planification en IA [Rich 87] pour décrire des cas sous forme de plans [Leake 93].

- la logique des descriptions, issue des systèmes KL-ONE et CLASSIC (extension de KL-ONE) [Haton 91] [Salotti *et al.* 99] [Kayser 97] possède deux intérêts principaux :

  • la représentation des connaissances facilitant la définition de concepts (ensembles d'objets).

  • deux mécanismes de raisonnement : la classification hiérarchique de concepts (insérer un concept dans la hiérarchie), et la reconnaissance d'un objet (c'est-à-dire trouver les concepts les plus spécifiques auxquels appartient cet objet).

Ce langage se situe à la frontière entre les réseaux sémantiques (un graphe dont les nœuds représentent des concepts et dont les arcs représentent des relations entre les concepts) et les langages de *frames*.

Le formalisme permet une certaine expressivité. Ainsi, à partir :

- d'un ensemble R de rôles (les attributs d'un concept, sous forme de relations binaires entre concepts)

- d'un ensemble P de concepts

- et d'un ensemble I d'individus (objets instances de concepts)

On peut définir des concepts C en utilisant les règles syntaxiques suivantes non exhaustives :

$C \rightarrow$ *one_of* $\{I_1, I_2, ..., I_n\}$ concept donné en extension
$C \rightarrow D \wedge E$ conjonction des concepts D et E
$C \rightarrow \forall R : D$ restriction de valeur d'un rôle définie par le concept D
$C \rightarrow R$ *fills* $\{I_1, I_2, ..., I_n\}$ valeurs du rôle
$C \rightarrow \min u$ ensemble des réels $> u$
$C \rightarrow \max u$ ensemble des réels $\leq u$
$C \rightarrow R$ *at_least* n cardinalité minimale de R
$C \rightarrow R$ *at_most* n cardinalité maximale de R
….

Les rôles d'un concept sont hérités par ses spécialisations sauf s'ils sont redéfinis par un lien *mod* (masquage de valeur) ou un lien *diff* (rôle différencié, c'est-à-dire héritage subdivisé en plusieurs sous-ensembles).
Il est à noter que l'idée sous-jacente est de restreindre le langage (par rapport à la logique du premier ordre) afin d'avoir un algorithme de classification hiérarchique complet et polynomial.

Ainsi, la logique des descriptions peut être utilisée pour représenter les index des cas.

S. Salotti et V. Ventos [Salotti *et al.* 99] utilisent la logique des descriptions pour organiser la base de cas et définir une représentation des similarités et des différences (voir annexe 3).

● ***Formalismes issus de la logique floue***

On peut imaginer d'utiliser la logique floue pour modéliser les poids reflétant l'importance de certains éléments d'un cas et/ou de son index, en fonction du contexte du problème à résoudre, comme des quantités floues.
Mais on peut également imaginer de modéliser les caractéristiques des cas ou index de manière floue en utilisant les notions de variables linguistiques (triplet (*V, X, Tv*) où *V* est une variable définie sur *X* et $Tv = \{V_1, V_2, ...\}$ contient des sous-ensembles flous normalisés de *X* caractérisant *V*, par exemple la variable *taille* définie sur {*petite*, *moyenne*, *grande*}), de modificateurs linguistiques (un opérateur *mod* qui, à toute caractérisation floue V_1 de *V* associe une nouvelle caractérisation $mod(V_1)$, par exemple le modificateur *très* qui peut être appliqué à *grand*), de propositions floues (éventuellement quantifiées) ou encore de règles floues [Bouchon-Meunier 95]. Ces caractéristiques seront ensuite mises en adéquation (*fuzzy pattern-matching*) avec le problème courant. La logique floue peut donc être également utilisée pour la définition de la notion de similarité (relations de similarité floues, agrégation floue : voir 4.2.2.2), et pour la phase d'adaptation (transformations floues, modus ponens généralisé, règles graduelles : voir 4.2.2.3).

4.2.2. Les différentes phases

Le système, face à un problème nouveau, procède généralement en cinq étapes :

4.2.2.1. L'indexation du problème à résoudre

Il s'agit ici de comprendre le problème et d'identifier des caractéristiques pertinentes (état initial, but à atteindre, …), que l'on décrit dans le même formalisme que celui dans lequel sont exprimés les cas et/ou les index en mémoire. Cette phase peut donc être extrêmement délicate, par exemple si l'on souhaite extraire les descripteurs pertinents à partir d'un énoncé en langage naturel.

Il s'agit de déterminer les descripteurs importants du problème à résoudre et éliminer les descripteurs non pertinents à la vue du *contexte*, et éventuellement inférer des descripteurs non présents en entrée grâce aux *connaissances générales du domaine* [Bergmann *et al.* 96], afin notamment que la situation nouvelle soit décrite dans le même vocabulaire que les cas en mémoire (*situation assessment* [Kolodner et Leake 96]). En outre, les procédures de description du problème nouveau (*situation assessment procedures* [Kolodner et Leake 96]), qui déterminent son index, sont utilisées lors de la phase de recherche en mémoire de cas

similaires, ainsi que lors de la phase de stockage (apprentissage), car des adaptations ou des échecs peuvent rendre nécessaires la modification de la description initiale.
Il est à noter que puisque cette phase de description du problème doit utiliser le vocabulaire déjà présent dans la librairie de cas, il peut être utile de rechercher des cas ou index dans la mémoire pour modéliser le nouveau problème. Ainsi, selon [Kolodner et Leake 96], les index en mémoire peuvent être utilisés comme guide afin de déterminer quels sont les caractéristiques du problème nouveau sur lesquelles porter l'attention : on utilise donc les caractéristiques de l'index comme descripteurs attendus du nouveau problème, et cela signifie que la phase d'indexation est couplée avec la phase de recherche.

Une façon d'identifier des caractéristiques pertinentes est d'utiliser l'apprentissage par examen de preuve (EBL : *explanation-based learning*) [Cornuejols et Miclet 02] [Maclin et Shavlik 89]. L'idée est la suivante : à partir d'un petit nombre d'exemples (un seul peut suffire) d'un concept, il s'agit d'apprendre ce concept (sous une forme générale) en utilisant une théorie du domaine. Ainsi, on peut apprendre un concept grâce à un seul exemple si celui-ci est bien expliqué et si on dispose de connaissances (la théorie du domaine) permettant de raisonner. Dès lors, en expliquant au système pourquoi tel ou tel indice est pertinent dans telle ou telle situation, celui-ci généralisera ces propositions (généralisation au niveau des situations et des indices) qui pourront alors être utilisées pour sélectionner des indices dans l'énoncé d'un problème nouveau.

On peut également chercher à modéliser l'importance des indices en fonction du contexte du problème à résoudre : fonction dynamique d'évaluation des poids, intégration de méta-connaissances sous la forme de règles ou de procédures, ou encore utilisation de la logique floue.

Deux techniques issues du data-mining et de la reconnaissance des formes peuvent être intéressantes [Cornuejols et Miclet 02] : les méthodes de sélection d'attributs qui sélectionnent, parmi un ensemble donné d'attributs, les plus pertinents, et les méthodes d'extraction d'attributs qui appliquent des transformations linéaires ou non à un ensemble donné d'attributs afin de réduire leur nombre. On peut également inclure à ce niveau divers traitements, lorsque l'application visée est le data-mining : suppression du bruit dans les données (data-cleaning [Bart et Frigot 02]), vérification de la cohérence des données du problème.

4.2.2.2. La recherche de cas similaires

Elle se fait à partir des index, et impose de définir la notion de similarité.

● Notion de similarité

1- La similarité syntaxique est basée sur des critères superficiels, et est définie soit sous la forme d'une distance (entre deux vecteurs), soit par le biais d'une classification dans une hiérarchie d'index (notion de plus petit généralisant : [Salotti *et al.* 99] et annexe 3), soit encore sous la forme d'une procédure de mise en correspondance (voir chapitre 5).

Une distance est appropriée si les cas ou index sont représentés par des vecteurs de couples (attribut-valeur). Il peut être judicieux d'accorder des poids aux descripteurs caractérisant leur importance au sein du problème, éventuellement fonction du contexte. Ces poids servent donc

à retrouver des cas similaires, et peuvent éventuellement être modifiés lors de la phase d'apprentissage. On peut imaginer d'utiliser la logique floue pour modéliser ces poids.

Un autre intérêt de la logique floue est de permettre de modéliser la similarité entre deux caractéristiques (d'un cas ou un index) par une *relation de similarité floue* ou par un *filtrage flou* [Bouchon-Meunier 95] [Gacôgne 97], et de modéliser ensuite la similarité entre deux cas par une opération d'*agrégation floue* (t-norme, t-conorme, produit cartésien de sous-ensembles flous, intégrales floues) [Bouchon-Meunier 95]. [Gacôgne 97] définit des relations floues de similarité, ainsi que des relations d'ordre floues.

Généralement, on définit la similarité par :

> *SIM (x, y)* = propriétés communes à *x* et *y*. Il peut être utile, voire nécessaire, de définir une mesure de similarité pour chaque type de propriété.

Et la dis-similarité par :

> *DISS (x, y)* = propriétés de *x* qui ne subsument pas une propriété de *y* [Salotti *et al.* 99] (opération non symétrique)

Plus spécifiquement, des exemples de mesure de similarité sont :

> *SIM (x, y)* = nombre de propriétés (ou une pondération) communes à *x* et *y*.
>
> *SIM (x, y) = f (x* ∩ *y, x – y, y – x)* qui combine similarité et dis-similarité,
> ou bien, dans le cas où les deux individus *x* et *y* sont décrits par des ensembles d'attributs *A* et *B* respectivement :
> $SIM(x, y) = f(A \cap B) / (f(A \cup B) + \alpha.f(A - B) + \beta.f(B - A))$ avec α et β positifs.
>
> *SIM (x, y)* = longueur (nombre d'arêtes) entre les deux concepts *x* et *y* au sein d'une hiérarchie.

A. Mille et B. Fuchs [Mille *et al.* 99], définissent deux mesures de similarité et de dis-similarité dans le cadre de la supervision industrielle, où un cas est représenté par un épisode (voir annexe 2). Leur définition de la dis-similarité entre deux séquences d'événements temporels est assez générale et représente un intérêt pour notre étude.

2- La similarité sémantique [Aamodt et Plaza 94] est basée sur des caractéristiques profondes, en utilisant par exemple des connaissances générales du domaine, afin de prendre en compte le contexte du problème, expliquer pourquoi deux cas sont similaires et expliquer le degré de similarité entre ces deux cas. On peut imaginer qu'un ensemble de méta-connaissances sous la forme de règles du premier ordre et/ou de procédures puissent permettre de juger la similarité entre deux caractéristiques d'un index ou entre deux cas. Cela pourrait permettre de représenter la similarité de manière fine, mais on voit bien que cela nécessite beaucoup de connaissances.

Il est clair que la notion de similarité sera d'autant plus pertinente qu'elle fait intervenir non seulement les caractéristiques de l'état initial du problème à résoudre, mais aussi les contraintes et le but à atteindre.

Peu de travaux ont été développés dans ce sens. On peut citer l'approche de D. Leake [Leake *et al.* 96 (b)] : le cas le plus facilement applicable à la nouvelle situation (donc le plus similaire) est celui dont le "coût d'adaptation" estimé est le plus faible (annexe 4). On peut également citer l'approche de Lieber [Lieber et Napoli 99] (voir annexe 1 : *recherche guidée par l'adaptabilité*).

● Recherche de cas en mémoire

En réalité, deux types de procédures sont intégrés dans cette phase [Kolodner et Leake 96] :

- la procédure de mise en correspondance et de classement : il s'agit ici de comparer deux cas et de juger de leur degré d'adéquation (ou de similarité). Cet algorithme prend donc en entrée la description et le but à atteindre du problème nouveau.

- la procédure de recherche de cas similaires dans la librairie de cas (ou d'index). Cet algorithme dépend donc de l'organisation de la base de cas/index (liste, arbre, ou graphe de concepts).

D'après [Aamodt et Plaza 94], on peut scinder la procédure de recherche en deux :

1- recherche d'un *ensemble* de cas similaires, ce qui peut correspondre à un dégrossissage.
Dans le cas où on a une hiérarchie des index, la recherche se fait par classification du problème modélisé (la similarité est ici liée aux relations de généralisation et de spécialisation qui ont été définies pour mettre en œuvre la hiérarchie). Généralement, on recherche les cas anciens appartenant aux index généralisant l'index du cas nouveau, et les plus spécifiques (concepts *subsumants les plus spécifiques*).

Mais il est à noter que cette recherche par classification correspond à un critère de similarité fort, donc cette recherche peut échouer. Pour pallier cet inconvénient, diverses approches ont été tentées afin de relaxer la classification :

L'une est décrite dans l'annexe 1, proposée par [Lieber et Napoli 99]. Leur approche mêle en outre les phases de recherche et d'adaptation (*recherche guidée par l'adaptabilité*).

Il est possible d'utiliser des techniques d'optimisation pour améliorer cette phase si la base de cas est très grande. Parmi les méthodes utilisées, on peut citer les algorithmes génétiques (voir [Charon *et al.* 96] pour une description des concepts de base), l'algorithme A^* (voir [Rich 87] pour une description de l'algorithme, et voir [Lieber et Napoli 99] en annexe 1 pour un exemple d'utilisation dans la phase de recherche d'un système CBR).

Une autre solution est de définir différents liens d'héritage entre concepts, par exemple : sûrs, probables, typiques ou exceptionnels. Ainsi, cela permet d'identifier des concepts dont le nouveau cas est une instance au sens strict, mais aussi des concepts dont il est une instance probable, ainsi que des concepts classés en tant qu'exception mais potentiellement intéressants.

On peut également rechercher les cas anciens appartenant aux index spécialisant l'index du cas nouveau, et les plus généraux (concepts *subsumés les plus généraux*).

2- puis sélection dans l'ensemble précédent du (ou des) *meilleur cas*, par exemple en se basant sur une distance évaluant la similarité (voir point précédent) :

De manière générale, si les concepts sont organisés en une hiérarchie, une méthode classique et simple est de calculer une *distance* entre chaque ancien cas retrouvé et le nouveau cas, basée sur la longueur du chemin reliant les concepts dans la hiérarchie. Une façon de procéder est décrite en annexe 3 [Salotti *et al.* 99].
Il est également possible de discriminer les cas candidats grâce à une heuristique visant à *minimiser l'effort d'adaptation*, par exemple en choisissant le cas ancien qui comporte le moins de buts à atteindre non présents dans le cas nouveau. En effet, il est utile de s'intéresser ici au but atteint, voire aussi à la manière d'atteindre ce but, afin d'améliorer la sélection et de préparer la phase suivante (adaptation du cas sélectionné). [Leake 96 (a)] définit la similarité en se basant sur une estimation des coûts d'adaptation (voir annexe 4).

Il peut être utile de se baser sur les caractéristiques *non identiques* entre les deux cas (en utilisant les liens de discrimination), les poids associés aux descripteurs, les éventuels prototypes associés à la base de cas.

Pour finir, notons que les connaissances générales du domaine (*background knowledge* - 4.2.2.3) peuvent également fournir une aide précieuse.

4.2.2.3. L'adaptation du cas retrouvé

Il s'agit de réutiliser le ou les cas sélectionnés et proposer une solution. Pour cela, il faut tenir compte des différences entre le cas remémoré et le problème à résoudre. Il faut agir sur une ancienne solution, c'est-à-dire insérer un élément nouveau, effacer un élément, substituer un élément par un autre. En outre, il s'agit éventuellement de savoir quelle(s) partie(s) du cas retrouvé sera transférée au cas à résoudre. Cette phase a lieu lors de la formulation d'une solution, et éventuellement après un *feedback* de l'utilisateur (phase de révision) ayant pointé des problèmes.

On distingue, de manière non exhaustive, quatre approches :

● ***substitution*** : on peut par exemple re-instancier une ancienne solution avec de nouveaux objets, ou rechercher dans une mémoire auxiliaire (réseau sémantique ou autre) une structure d'objet ou une valeur de paramètre pour remplacer une ancienne inadéquate, cette recherche pouvant d'ailleurs être guidée (par des heuristiques par exemple), ou même mettre en œuvre un système d'adaptation par substitution à base de cas [Kolodner et Leake 96]. [Smith et Cunningham 93] opère par modification des attributs des objets de la solution.
Si le système possède des connaissances générales sous la forme d'un graphe causal ou un réseau sémantique, une possibilité est de lancer un processus de recherche d'objets jouant les mêmes rôles (par exemple possédant le même type de relations) que ceux du cas nouveau. Les objets ainsi découverts sont candidats pour une substitution. Il faut alors définir la similarité d'objet et la similarité de relations.

● ***transformation*** : on s'intéresse ici uniquement à la solution, et pas à la manière d'atteindre cette solution (à rapprocher de la notion d'analogie transformationnelle). Il s'agit de définir et utiliser des opérateurs qui appliquent des transformations sur la solution. Ces opérateurs sont sensibles aux différences entre les deux états initiaux (celui du cas nouveau et celui du cas retrouvé similaire), et transforment la solution du cas retrouvé en fonction de ces différences.

[Smith et Cunningham 93] utilise quatre opérateurs possibles (substituer, effacer, insérer et ré-ordonner) qui sont appliqués à la solution.
On peut utiliser des heuristiques de bon sens pour remplacer, ajouter ou effacer des composants d'une ancienne solution, ou bien utiliser un raisonnement à base de modèle causal.

La logique floue est un outil intéressant pour modéliser des transformations floues, par exemple le modus ponens généralisé, le mécanisme d'inférence floue (si on a la règle : *V est A* $\Rightarrow$ *W est B* et si on a le fait : *V est A'* alors on en déduit : *W est B'*), et également le concept de règles graduelles (*plus V est* A_i*, moins W est* B_j) [Bouchon-Meunier 95].

● ***dérivation*** : on cherche davantage à analyser la manière dont a été trouvée la solution. Les cas doivent donc contenir des explications / justifications sur les méthodes utilisées, les opérateurs considérés, les buts atteints, … Ainsi, le plan lié au cas ancien et permettant d'atteindre la solution sera ré-instancié et les pas élémentaires sont "re-joués" dans le cadre du nouveau cas. Cette notion d'analogie dérivationnelle a été introduite par Jaime Carbonell [Schmid et Carbonnell 94] [Blumenthal et Porter 94]. [Smith et Cunningham 93] utilise une approche voisine consistant à ré-instancier la trace de raisonnement du cas passé - chaque pas contient une action et une justification – en utilisant le cas à résoudre.
Cette façon de procéder peut nécessiter une importante base de connaissances générales du domaine.
Il est également possible de mettre en œuvre un système d'adaptation à partir de cas stockés sous la forme d'une dérivation de l'état initial à l'état final [Leake 93].

● ***heuristiques d'adaptation*** : on peut utiliser des heuristiques spécialisées et indexées par le contexte dans lequel elles sont utiles.

On peut également citer les travaux de [Bergmann *et al.* 96] qui utilise des règles de complétion pour étendre la description du cas retrouvé, et des règles d'adaptation pour modifier le cas retrouvé s'il y a des différences avec le problème.

4.2.2.4. La validation du cas ré-utilisé

Cette étape comporte deux phases :

1- *l'évaluation de la solution* proposée détermine si la solution est correcte. Cela peut être fait par l'utilisateur, ou par un simulateur, ou en exécutant la solution dans le cadre réel. Si cette évaluation prend un certain temps, alors le cas peut être enregistré dans la base, mais il doit être marqué comme non évalué. Si l'évaluation est réussie, on passe à la phase d'apprentissage.

2- Si ce n'est pas le cas, nous sommes alors dans le cadre d'un apprentissage à partir d'échec : il faut *réparer la solution*, c'est-à-dire :

● détecter les erreurs. Cette phase peut nécessiter l'intervention de l'utilisateur s'il faut modifier certaines connaissances du système ou lui fournir de nouvelles connaissances.

● corriger la solution et inclure des connaissances de façon à ne plus commettre ces erreurs. Cette phase peut également nécessiter l'intervention de l'utilisateur. Mais il est important de noter que l'on peut en grande partie automatiser cette phase :

- si le bon cas a été retrouvé, alors c'est l'adaptation qui est en cause. On boucle sur la phase d'adaptation (les informations éventuellement fournies par l'utilisateur vont être utilisées), puis on modifie, par apprentissage, les stratégies d'adaptation [Leake 96 (a)] (annexe 5).

- si le système a utilisé un cas inopportun alors qu'il existe dans la base un cas qui permettrait de résoudre le problème, alors on boucle sur la phase de recherche pour tester un autre cas. Mais il peut être utile, si le système possède suffisamment de connaissances, de ré-indexer le cas nouveau (le problème à résoudre) ainsi que certains cas passés (ceux qui ont été injustement jugés similaires et ceux qui ont été injustement jugés non similaires), puis de modifier la hiérarchie des cas/index en conséquence. Enfin, il peut être utile de modifier les connaissances liées aux notions de similarité et d'indexation.

- si le système a utilisé un cas inopportun et qu'il n'existe pas dans la base un cas qui permettrait de résoudre le problème, alors on peut soit tenter d'adapter le cas inopportun, ce qui peut nécessiter un certain travail si le problème à résoudre et la cas retrouvé sont dis-similaires, soit enregistrer le problème et sa solution (grâce à l'intervention de l'utilisateur) car il s'agit là d'un problème vraiment nouveau que le système ne pouvait résoudre avec sa base de cas.

● expliquer les erreurs. Certains travaux ont tenté d'automatiser la tâche d'apprentissage à partir d'explications. Citons les travaux de [Maclin et Shavlik 89] et de [Cornejols et Miclet 02] sur la technique nommée *Explanation-Based-Learning*. D'autres travaux ([Leake 94] et [Leake 95 (b)]) se sont attachés à analyser les motivations des explications : la technique nommée *Goal-Driven Explanation*, basée sur le fait que la poursuite d'explications par un individu est contrôlée par des interactions dynamiques entre le monde extérieur, et les connaissances, les besoins en informations et les buts actifs de cet individu, est mise en œuvre grâce à un système CBR.

L'apprentissage par l'échec entre en jeu ici et peut être réalisé par exemple grâce à la création de connaissances négatives (du type "si on est dans telle situation, alors ne pas faire telle chose"). [Minsky 94] insiste sur le fait que nos connaissances nous indiquent certes que faire dans telle situation, mais qu'une grande partie d'entre elles indiquent ce qu'il ne faut pas faire. Atteindre un but nécessite d'éviter les actions pouvant entraîner des problèmes. Il assimile le raisonnement de bon sens à "un îlot de consistance délimité par des bornes de sécurité", et ces bornes sont l'expertise négative. Leur apprentissage repose sur un observateur de niveau méta. Selon [Minsky 85] deux processus nous empêchent de faire des erreurs ou d'avoir des pensées fausses, erronées, négatives : les surpresseurs qui attendent qu'une mauvaise idée apparaisse mais empêchent de passer à l'acte et bloquent donc le cours de la pensée, et les censeurs qui interviennent plus tôt et interceptent les états d'esprit précédant cette idée, et donc détournent la pensée.

4.2.2.5. L'apprentissage

Le but est de rendre le système plus efficace (si un cas a été adapté d'une nouvelle façon, a été résolu par une nouvelle méthode ou grâce à la combinaison de solutions de plusieurs cas, alors lors d'un rappel futur le travail réalisé n'aura pas à être répété) et plus compétent (l'expérience permet au système d'anticiper et d'éviter les erreurs commises dans le passé).

Le premier problème est de savoir si l'on doit effectivement enregistrer le cas nouveau : le système a-t-il appris quelque chose d'utile ? La notion d'utilité doit donc être définie, notamment par rapport à l'atteinte de buts, mais également en terme de contexte (si un problème connu a été résolu dans un contexte très différent, alors il est utile de généraliser le contexte). Selon [Kolodner et Leake 96], si ce qui est différent dans une nouvelle situation nous enseigne quelque chose qui n'aurait pas pu être *facilement* inféré à partir des cas existants, alors ce nouveau cas est utile.

Un problème potentiel lié à l'apprentissage par accumulation est qu'en enregistrant un nouveau cas après chaque résolution d'un problème, la base de cas peut rapidement devenir ingérable du fait de sa taille incontrôlée. Il faut donc enregistrer un cas de manière sélective, c'est-à-dire juger de l'intérêt du cas nouveau par rapport au contenu de la base de cas, comme cela vient d'être souligné précédemment. On peut également, à l'occasion, supprimer des cas dont la fréquence d'utilisation est devenue très faible, ce qui équivaut à un oubli. La généralisation de cas (fusionner le cas appris avec un autre cas similaire en mémoire) est aussi un moyen d'éviter d'enregistrer tous les cas. On peut citer à ce sujet les opérateurs de généralisation décrits dans l'annexe 6 à propos de la méthode "espace des versions" [Cornuejols et Miclet 02] : pour un attribut numérique, on généralise à l'aide d'un intervalle englobant les différentes valeurs, ou à l'aide d'une valeur moyenne avec un écart-type, et pour un attribut qualitatif hiérarchique, on peut généraliser en utilisant l'attribut "père".

Il s'agit également de savoir quelles sont les informations pertinentes que l'on va sauvegarder [Kolodner et Leake 96] (toutes les différences observées ne sont pas forcément importantes), sous quelle forme, c'est-à-dire quel sera le contenu du cas appris et de son index. Les informations à conserver au sein du nouveau cas sont les descripteurs pertinents, la solution et la manière de l'atteindre, mais également des méta-connaissances du type explications, modes d'emploi, ainsi que des connaissances négatives issues d'un éventuel échec (voir phase de révision). L'indexation du cas est un problème central car c'est ce qui va permettre de réutiliser ce cas dans le futur. Il est clair que les résultats des phases précédentes de recherche des cas similaires, d'adaptation et de révision sont d'une grande aide pour l'indexation :

- les index des cas similaires trouvés lors de la phase de recherche sont utiles pour apprendre les index du nouveau cas.

- en s'appuyant sur les phases de recherche et d'adaptation, on peut savoir si le nouveau cas doit être intégré dans une structure générale existante (par exemple un index) contenant des cas similaires, ou au contraire intégré en tant qu'entité à part entière.

- la phase d'adaptation peut aussi donner des renseignements précieux pour discriminer le cas nouveau (donc pour l'indéxer) et le cas ancien similaire qui a été adapté.

- la phase de révision permet de créer des connaissances négatives : dans tel contexte, tel indice n'est pas pertinent.

Si les éléments des index ont des poids numériques, alors il est possible de développer une procédure de renforcement [Rich 87] afin de modifier ces poids en fonction de la qualité de la solution fournie : si la solution est correcte, alors on augmente les poids des descripteurs qui ont permis de retrouver le cas qui a permis de construire une bonne solution. Si la solution n'est pas correcte, alors on diminue ces poids. D'ailleurs, même si les descripteurs ne sont pas

associés à des poids, il peut être utile de leur associer des méta-connaissances indiquant dans quels contextes ils sont utiles (et dans quels contextes ils ne le sont pas).

Les techniques d'apprentissage artificiel provenant de l'IA peuvent être utilisées au cours des différentes phases d'un système CBR. L'apprentissage au sein de cette discipline est généralement vu sous l'angle de la généralisation, par le biais de l'induction à partir d'exemples aboutissant à la création de concepts, et fonctionne soit sans connaissance *a priori* [Cornuejols et Miclet 02], soit avec des connaissances du domaine (EBL : *Explanation Based-Learning* [Maclin et Shavlik 89] [Leake 95 (b)] [Cornuejols et Miclet 02]). Ainsi, dans le cadre du CBR, si plusieurs cas sont indexés de façon similaire et prédisent une solution similaire, alors une généralisation inductive à partir de ces cas peut être mise en place.

En conclusion, l'apprentissage au sein d'un système CBR peut être mis en œuvre de multiples manières :

● Apprentissage par accumulation (sélective) d'expériences (stockage de cas).

● Apprentissage des index (l'index d'un cas stocké peut être modifié dans le futur en fonction des nouvelles expériences), ou des connaissances d'indexation (pour rechercher les indices pertinents du problème à résoudre - phase de modélisation).

● Apprentissage de l'organisation de la mémoire [Leake 95 (a)], afin notamment d'améliorer la phase de recherche. Il s'agit également, en utilisant les informations issues des différentes phases, de réorganiser la mémoire de cas et d'index :

- trouver la place du cas résolu dans la hiérarchie, et éventuellement réorganiser cette hiérarchie (voir par exemple le chapitre sur la classification dans [Haton 91]),

- généraliser le nouveau cas avec des cas passés,

- créer de nouveaux index regroupant des cas similaires.

● Apprentissage de stratégies d'adaptation, sous la forme de règles de substitution ou de transformation, ou encore grâce à la génération de procédures de recherche en mémoire [Leake 93] (annexe 5).

● Apprentissage de connaissances négatives afin de mieux définir la similarité, l'indexation ainsi que l'adaptation.

● Apprentissage de la notion de similarité, par exemple en modifiant les poids associés à une distance, ou encore en estimant un coût lié à l'obtention ou à l'adaptation de la solution [Leake 96 (a)] : en effet, lorsque plusieurs cas similaires ont été retrouvés pour résoudre un problème nouveau, il est judicieux de choisir le cas dont la solution est la plus facile à mettre en œuvre.

● Améliorer le modèle des connaissances générales du domaine (voir 4.2.3.), utilisé pour définir la similarité, l'indexation, l'adaptation.

4.2.3. Approches intégrées

La plupart des systèmes CBR utilisent des connaissances générales (*background domain knowledge*) en plus des connaissances représentées par les cas, généralement sous la forme :

● de systèmes à base de règles (voir [Pitrat 93] pour une bonne étude des concepts clés : détermination des règles déclenchables, choix d'une règle, intérêt des méta-connaissances et des connaissances procédurales),

● de modèles de raisonnement causal (*Model-Based Reasoning* – MBR) (voir [Torasso 04] et [Karamouzis et Feyock 92] pour des exemples d'intégration CBR-MBR),

● de réseaux sémantiques (voir [Kayser 97] et [Sabah 88] pour une présentation des concepts de base),

● ou encore de hiérarchies d'objets / *frames*.

Il s'agit donc de connaissances spécifiques à une phase du raisonnement (similarité, adaptation, …) et également au domaine d'étude.

Les intérêts sont multiples :

● servir de guide au cours des différentes phases : l'indexation (d'une part indexer un cas nouveau et ré-indexer un cas ancien, et d'autre part modifier la hiérarchie des index lors de l'insertion d'un cas nouveau), la définition de la notion de similarité, les stratégies d'adaptation, la réparation en cas d'échec et la généralisation de cas sont des concepts d'une grande diversité qui nécessitent donc beaucoup de connaissances pour être appliqués de façon pertinente ou pour fournir des explications,

● mais également permettre de résoudre un problème dans le cas où aucune situation similaire n'a été retrouvée ou bien s'il y a eu un échec de résolution, par exemple en utilisant une méthode générale de type analyse moyens-fins (*General Problem Solver* – Allen Newell dans [Rich 87]), le raisonnement qualitatif, ou encore le raisonnement temporel pour un problème de planification, ….

On peut alors voir apparaître deux niveaux de traitements enchevêtrés : le niveau des connaissances, composé de méta-connaissances et de connaissances spécifiques au domaine, et le niveau des expériences, composé de la base de cas (un ensemble de méthodes pour résoudre des problèmes).

Une autre façon de voir une approche intégrée est d'utiliser, à l'intérieur d'un système CBR, d'autres systèmes CBR, par exemple pour la phase d'adaptation et pour définir la similarité, éventuellement en les combinant avec des systèmes RBR (*Rule-Based Reasoning*) [Leake 96 *et al.* (b)] (voir annexe 4).

En conclusion, cette approche intégrée offre une vision globale afin d'attaquer les problèmes de l'IA : résolution de problèmes grâce à des expériences passées jugées similaires en utilisant de connaissances générales et spécifiques pour définir les concepts de base (indexation, similarité, adaptation, …), combinaison de différents types de raisonnement (à base de cas, à base de règles, à base de modèles, …), combinaison de différents types d'apprentissage

(accumulation d'expériences, organisation de la mémoire, connaissances stratégiques (indexation, similarité, adaptation), connaissances négatives, généralisation).

4.2.4. Conclusion

Le raisonnement à base de cas offre une grande souplesse et une grande flexibilité dans la définition et l'utilisation d'un cas :

● souplesse dans la modélisation d'un cas, dans le sens où l'on peut faire un couplage avec d'autres technologies issues de l'IA. Un cas peut être représenté par un système de faits et de règles, un plan, une procédure heuristique, une théorie, un modèle, un graphe, …

● capacité d'un cas à modéliser des connaissances typiques, ou des exemples.

● les conditions d'application d'un cas, sous la forme d'index, peuvent être modélisées de diverses façons (logique floue, objets complexes au sein d'une hiérarchie, …). En outre, la mise en correspondance peut n'être que partielle dans un système CBR, alors qu'elle doit être totale dans le cas d'un système à base de règles, mais la contre partie est alors la nécessité d'une adaptation. Enfin, dans un système à base de règles, l'instanciation des règles ne nécessite qu'un *matching process* très simple, c'est-à-dire vérifier que l'objet testé est identique ou appartient à la classe de l'objet de la condition, alors que dans un système CBR, le *mapping process* nécessite davantage de travail afin de juger de la similarité des deux objets malgré des différences observées.

● on peut intégrer des méta-connaissances et des connaissances procédurales à un cas, afin notamment de représenter des connaissances du type *mode d'emploi* et *explications* qui facilitent l'adaptation.

● récursivité/compositionnalité dans la définition d'un cas : un cas peut être défini à l'aide d'autres cas, ce qui correspond bien à une caractéristique cognitive humaine. Cependant, il semble que dans la majorité des travaux, un cas représente l'unité de base.

● on peut associer différents points de vues à un cas (cela rejoint le point précédent).

Nous allons maintenant présenter une application du raisonnement à base de cas dans le cadre de notre étude.

Chapitre 5

AGENT A BASE DE CAS

Nous détaillons l'agent artificiel dédié aux principes du raisonnement à base de cas, où un cas représente un petit nombre de réservoirs reliés par des conduites et permet de gérer une situation spécifique, et la mise en correspondance permet de retrouver ce cas au sein du réseau courant, puis nous analysons les résultats d'expérimentations informatiques.

5.1. INTRODUCTION

Le CBR est utilisé de la façon suivante : il s'agit, à partir de l'état courant du système, de déterminer les informations pertinentes sur la situation (alarmes, positions des vannes, configuration spatiale des réservoirs, diamètres des conduites, …). Ces informations sont les indices permettant de retrouver des cas en mémoire. Ces cas sont associés, au niveau stratégique, à des buts et contraintes, et au niveau planification, à des actions (ou séquences d'actions) (voir figure 16).

Un des intérêts du raisonnement à base de cas est de modéliser divers types de stratégies, qu'elles soient simples ou complexes. On peut par exemple définir une stratégie de base comme le contrôle d'un ou plusieurs chemins grâce à un ensemble de cas. P. Ponsa parle de modèles de comportement [Ponsa *et al.* 02] : dans le cadre de la configuration représentée par la figure 6, on a les stratégies de base S1 (contrôle d'un chemin ouvert passant par un seul réservoir intermédiaire, par exemple $T_1 \rightarrow T_2 \rightarrow T_5$, $T_1 \rightarrow T_3 \rightarrow T_5$ ou $T_1 \rightarrow T_4 \rightarrow T_5$), S2 (contrôle d'un chemin ouvert passant par deux réservoirs intermédiaires, par exemple $T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_5$ ou $T_1 \rightarrow T_4 \rightarrow T_3 \rightarrow T_5$), …, jusqu'à la stratégie de base qui contrôle les cinq chemins possibles. Une stratégie globale peut être vue alors comme une séquence de stratégies de base.

Mais on peut aussi définir, toujours sous forme de cas, des stratégies locales. Par exemple, la stratégie locale consistant à vidanger un réservoir en alarme dans un autre réservoir qui n'est pas en alarme peut s'exprimer à l'aide d'un cas.

La définition de la similarité entre états sera facilitée si l'on définit préalablement l'abstraction d'état. L'index permet justement d'associer à un cas les informations pertinentes. L'indexation consiste donc ici à abstraire un état du système (par exemple dans un cas avec alarme, on pourrait avoir des informations qualitatives concernant la présence de vanne ou pas, les positionnements relatifs des réservoirs, et l'existence des conduites reliant ces réservoirs, ceci au voisinage d'un réservoir en alarme).

La représentation d'un cas peut inclure, des données qualitatives concernant l'état du système (hauteurs d'eau, tendances, alarmes, position des vannes ouvertes ou fermées), ainsi que des

données qualitatives concernant le réseau (positions relatives des réservoirs – grâce par exemple à un graphe orienté, diamètres relatifs des conduites, conduites avec ou sans vanne). En effet, les diamètres des conduites, liés aux débits circulant dans ces conduites, sont des paramètres importants qui vont influencer les stratégies. D'autre part, un cas peut être relatif non pas à l'ensemble du réseau mais à un sous-ensemble de quelques réservoirs (un à trois), afin que ce cas soit plus facilement et plus souvent applicable. Dès lors, le système CBR est applicable à tous types de modèles, et donc indépendant de la configuration du réseau. Ceci constitue un avantage majeur. De plus, une action (ou un ensemble d'actions) est conditionnée par un état du système qu'il faut définir de manière assez générale pour qu'elle puisse s'appliquer à partir de divers états "voisins". Ainsi, une façon de définir des états typiques est de les définir partiellement.
Ainsi, on peut procéder en deux niveaux : le premier niveau concerne la configuration du réseau (ou une partie du réseau) et son état. Un cas est donc représenté à ce niveau uniquement avec ce type de données. Ensuite, au second niveau, on associe à ce cas une action (voire une séquence).

5.1.1. Modélisation du problème à résoudre

Deux éléments sont à prendre en compte :

● la configuration du réseau est un graphe orienté dont les nœuds sont les réservoirs et leurs caractéristiques, les arcs sont les conduites et leurs caractéristiques. Il s'agit donc d'extraire à chaque pas de temps la partie intéressante du réseau à la vue de l'état de celui-ci.

● l'état du système à chaque pas de temps, est modélisé par des attributs associées aux nœuds (hauteur d'eau, tendance) et aux arcs (position *ON/OFF* des vannes).

5.1.2. Cas en mémoire

Un cas est un exemple permettant de gérer une situation concrète, représenté par trois éléments :

● une partie de réseau concrète, modélisé par un graphe orienté dont on associe à chaque nœud un réservoir, avec comme attribut la capacité du réservoir associé, et à chaque arc une conduite avec comme attributs le diamètre de la conduite et l'existence d'une vanne ou non sur cette conduite.

● l'état du système relatif à cette partie de réseau et à un instant donné contient d'autres informations, dynamiques, et représentées par des attributs des nœuds et des arcs du sous-graphe (hauteur d'eau, tendance qualitative, et état d'alarme pour chaque nœud, position de l'éventuelle vanne pour chaque arc).

● une séquence d'actions (ou une seule action) afin de satisfaire un but.

5.1.3. Index en mémoire

Un index est relié à un ou plusieurs cas et est composé de trois parties :

● une partie de réseau abstraite, modélisé par un graphe orienté, étiqueté uniquement par l'existence de vanne sur les conduites (on ne représente pas les caractéristiques des réservoirs, ni les diamètres des conduites). Il s'agit du même graphe que celui du cas auquel il est relié.

● l'état abstrait (uniquement l'état d'alarme de chaque réservoir – les autres caractéristiques de l'état sont représentées au niveau du cas).

● un but de haut niveau (gestion d'un ou plusieurs alarmes, gestion sans alarme, accélérer le processus, minimiser la durée totale, ...).

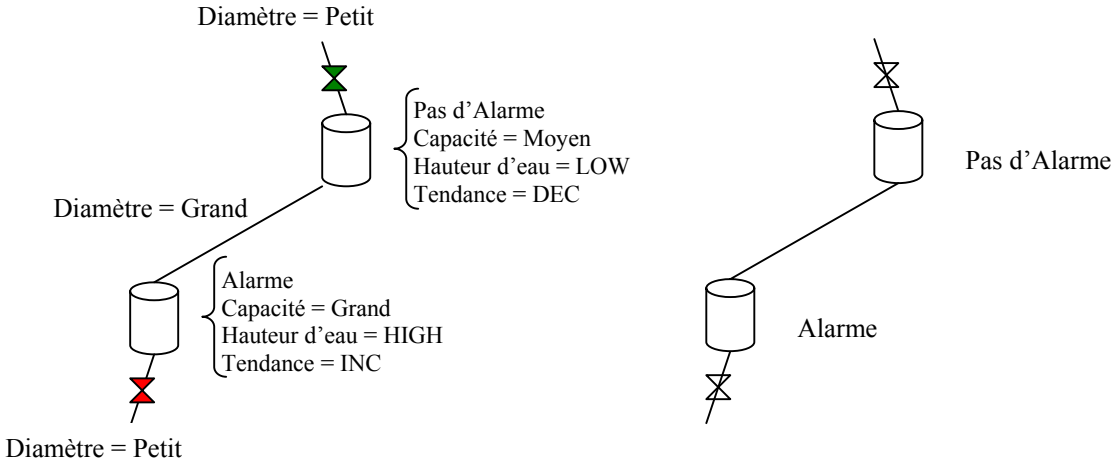
● ainsi que d'autres informations intéressantes et discriminantes (nombre de réservoirs modélisés, présence de conduites sans vanne, connaissances négatives).

Ainsi, un index est une classe regroupant des cas. Les cas placés sous le même index possèdent tous une *abstraction commune* qui est cet index. Il s'agit alors d'éviter que deux cas similaires soient issus d'index différents. Pour cela il faut que les index soient suffisamment distincts, et que les cas issus d'index différents soient suffisamment distincts. Cela est possible grâce à des pénalisations numériques *spécifiques* (voir 5.3.1.2). Si cela venait tout de même à se produire, il faut alors discriminer les index en question (voir 5.5.5).

: vanne ouverte

: vanne fermée

: existence d'une vanne



exemple d'un cas en mémoire *exemple d'un index en mémoire*

Figure 15 : différenciation entre un cas et un index

5.1.4. Organisation de la mémoire

En outre, une hiérarchie à plusieurs niveaux est constituée : tout d'abord une hiérarchie de buts de haut niveau, telle que chaque but contient des index, éventuellement eux aussi organisés en une hiérarchie, chaque index étant lui-même éventuellement organisé en une hiérarchie de cas : l'action (ou la séquence d'actions) associée à chaque cas est une façon d'atteindre concrètement le but de haut niveau dans une situation donnée.

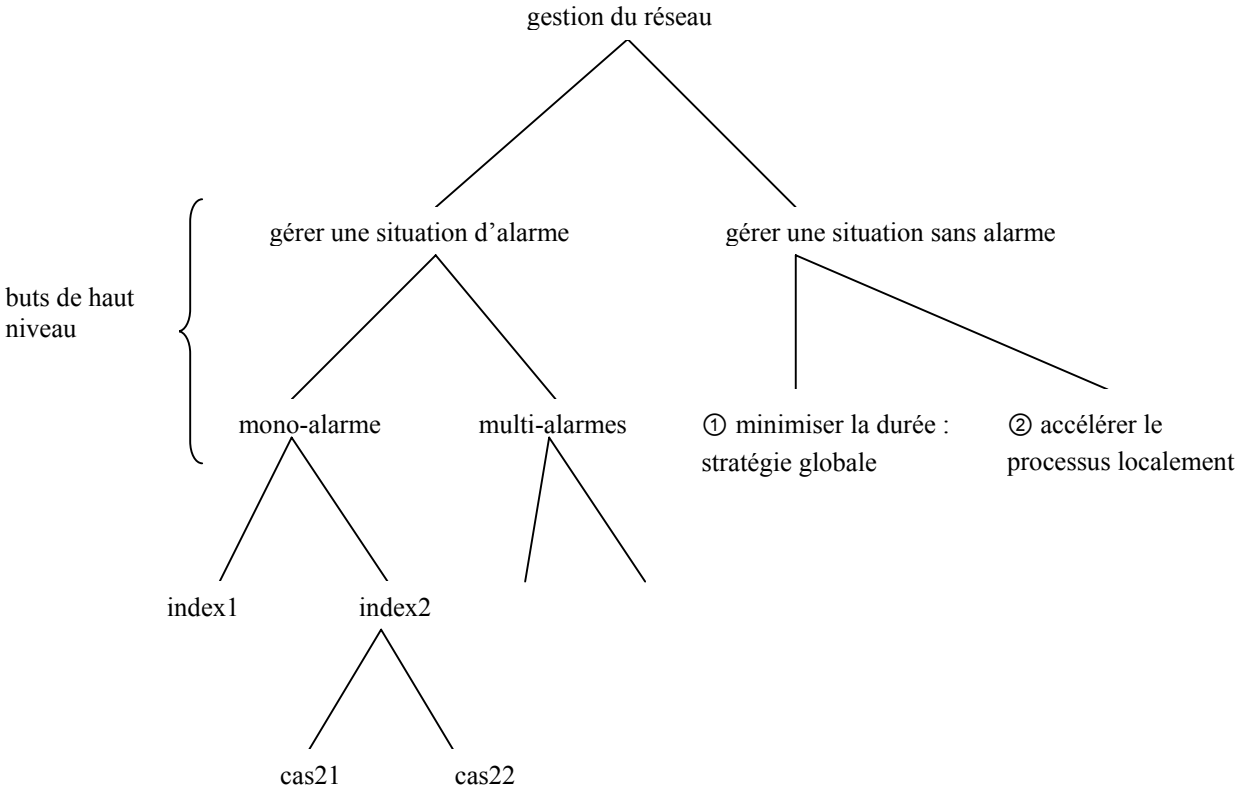


Figure 16 : Organisation générale des cas et index

Nous pouvons alors détailler encore la modélisation des cas :

Dans la situation où au moins un réservoir est en alarme, les cas et index n'intègrent que la partie concernée, c'est-à-dire le (ou les) réservoir en alarme, ainsi que ce qui influence cette alarme :

● s'il y a un seul réservoir en alarme, le cas intègre ce réservoir et les conduites en amont et en aval. Mais si une des conduites est sans vanne, alors on pourrait intégrer également le réservoir non en alarme reliant cette conduite au réservoir en alarme, ainsi que les conduites reliées directement à ce réservoir non en alarme, car cela permettrait une gestion plus fine de l'alarme.

● s'il y a plusieurs réservoirs en alarme, deux situations se présentent :

- les réservoirs en alarme ne sont pas reliés directement. Dans ce cas, on cherche simplement à privilégier d'abord le réservoir qui débordera avant les autres. On se ramène alors au cas mono alarme. Il est cependant possible de mettre en place une stratégie afin de stocker si nécessaire un cas comportant des réservoirs en alarme non reliés directement, ainsi que les réservoirs reliant ces premiers.

- les réservoirs en alarme sont reliés directement. Dans ce cas, il faut prendre en compte tous ces réservoirs en même temps, car réduire l'alarme d'un réservoir peut augmenter l'alarme d'un autre réservoir. On enregistre alors les réservoirs en alarme, les conduites en amont et en aval de chacun de ces réservoirs, avec la même remarque que dans le cas mono-alarme si une des conduites est sans vanne.

● dans la situation où aucun réservoir n'est en alarme, deux cas se présentent :

- on cherche à accélérer le processus localement, c'est-à-dire que l'on cherche à ouvrir un chemin entre le réservoir source et le réservoir puit. Le cas représente donc un chemin comportant au moins une vanne fermée. Diverses sous-stratégies sont alors possibles : privilégier le chemin comportant le moins de vannes fermées, ouvrir d'abord les vannes du haut, puis celles du bas, ou l'inverse.

- on cherche à minimiser la durée totale de la simulation (voir index n° 9 – 5.2.3.), ce qui nécessite une stratégie. Cela ne signifie pas qu'il faille enregistrer tout le réseau. Le cas représente une couche de réservoirs se videangeant dans un autre réservoir.

NB : Il y a un ordre d'application des cas. On essaie d'appliquer la stratégie de minimisation de la durée, sinon on essaie d'accélérer le processus localement.

5.2. BASE DE CAS ET BASE D'INDEX

Tout d'abord, détaillons les structures de données utilisées : la structure Str^{new} = (Gr^{new} ; $Tank^{new}$) représente le problème nouveau à résoudre, et la structure Str^{old} = (Gr^{old} ; $Tank^{old}$) représente un élément stocké en mémoire dans la base de cas ou d'index.

5.2.1. Structure de données du problème à résoudre

● les conduites sont représentées à l'aide d'un tableau Gr^{new} (*Gr* = graphe), par des objets $Gr^{new}[i][j]$ comportant les attributs suivants :

- existence d'une conduite : $Gr^{new}[i][j].\exists\_conduite \in \{0,1\}$
- état de la vanne : $Gr^{new}[i][j].position\_vanne \in \{0,1\}$
- existence d'une vanne : $Gr^{new}[i][j].\exists\_vanne \in \{0,1\}$
- diamètre de la conduite : $Gr^{new}[i][j].\Phi \in$ {P, M, G} (P = Petit, M = Moyen, G = Grand)

avec $i \in [1, ..., n]$ et $j \in [1, ..., n]$ et n = nombre de réservoirs.

● les réservoirs sont représentés à l'aide d'un tableau $Tank^{new}$, par des objets $Tank^{new}[k]$ (notés T_k dans le chapitre 3.2. sur l'agent qualitatif) comportant les attributs suivants :

- état d'alarme ou non : $Tank^{new}[k].état \in \{0,1\}$
- capacité : $Tank^{new}[k].capacité \in$ {P, M, G}
- tendance : $Tank^{new}[k].tendance \in$ {INCL, INCS, DECL, DECS, STD}
- hauteur $Tank^{new}[k].hauteur \in$ {EMPTY, LOW, HIGH, FULL}

avec $k \in [1, ..., n]$.

Remarque 1 : Les diamètres des conduites et les capacités des réservoirs sont représentés de manière qualitative dans un domaine de valeurs D = {P, M, G} (P = Petit, M = Moyen, G = Grand) grâce à la méthode des k-moyennes (voir annexe 7). L'ensemble des valeurs numériques correspondant aux données du réseau sont donc classées à l'avance.

Remarque 2 : Nous aurions pu de façon alternative modéliser ces trois variables qualitatives par des nombres flous. Ainsi, une valeur numérique donnée est représentée par la variable qualitative avec laquelle le degré d'appartenance est le plus élevé. De plus, l'opposé flou ($op(x) = 1 - x$) du degré d'appartenance de cette valeur numérique à l'un de ces ensembles pourrait alors définir la valeur de pénalisation (voir 5.3.1.2) associée à la mise en correspondance entre cette valeur numérique et la variable qualitative considérée. Mais il n'est pas évident que cette méthode soit plus efficace que la précédente. En effet toutes deux comporte une part d'arbitraire.

Remarque 3 : Les hauteurs et tendances des réservoirs sont également représentées de manière qualitative, de la même façon que pour l'agent qualitatif (voir 3.2.1).

5.2.2. Structure de données du cas et de l'index en mémoire

On ne représente qu'une partie du réseau : nombre n' de réservoirs de l'ordre de 2 ou 3 avec $n' < n$.

Pour prendre en compte la possibilité de disjonctions au niveau des valeurs possibles d'un attribut (après une généralisation de deux cas ou deux index par exemple), les domaines de valeurs des attributs sont ici représentés par des tableaux de valeurs (et non une valeur), par exemple ($Tank^{old}[k].capacité$ = [P, M]) ≡ ($Tank^{old}[k].capacité$ = P ∨ M).

Certains attributs représentent des **C**onnaissances **N**égatives (objets **CN**) qui définissent les valeurs interdites.

Les structures de données sont les suivantes :

● nombre de réservoirs :

valeur : n'
valeurs interdites sur le nombre de réservoirs : *CN_nb_Tank*

● présence de conduite sans vanne ou non

● les conduites sont représentées par un tableau (graphe) d'objets $Gr^{old}[i'][j']$,
avec $i' \in [1, ..., n']$ et $j' \in [1, ..., n']$ et les attributs :

$Gr^{old}[i][j].\exists\_conduite \in \{0,1\}$
$Gr^{old}[i'][j'].position\_vanne$ et $Gr^{old}[i'][j'].CN\_position\_vanne \in \{0,1\}$
$Gr^{old}[i'][j'].\exists\_vanne$ et $Gr^{old}[i'][j'].CN\_\exists\_vanne \in \{0,1\}$
$Gr^{old}[i'][j'].\Phi$ et $Gr^{old}[i'][j'].CN\_\Phi \in \{P, M, G\}$

● les réservoirs sont représentés par un tableau d'objets $Tank^{old}[k']$ avec $k' \in [1, ..., n']$ et les attributs :

$Tank^{old}[k'].état \in \{0,1\}$
$Tank^{old}[k'].capacité$ et $Tank^{old}[k'].CN\_capacité \in \{P, M, G\}$
$Tank^{old}[k'].tendance$ et $Tank^{old}[k'].CN\_tendance \in \{INCL, INCS, DECL, DECS, STD\}$
$Tank^{old}[k'].hauteur$ et $Tank^{old}[k'].CN\_hauteur \in \{EMPTY, LOW, HIGH, FULL\}$

$Tank^{old}$[k'].CN_conduite_sup_amont.état_vanne
$Tank^{old}$[k'].CN_conduite_sup_amont.Φ
$Tank^{old}$[k'].CN_conduite_sup_amont.$\exists$_vanne

$Tank^{old}$[k'].CN_conduite_sup_aval.état_vanne
$Tank^{old}$[k'].CN_conduite_sup_aval.Φ
$Tank^{old}$[k'].CN_conduite_sup_aval.$\exists$_vanne

Remarque 1 : les connaissances négatives *CN_conduite_sup_amont* et *CN_conduite_sup_aval* concernent des conduites supplémentaires, c'est-à-dire non présentes explicitement au sein de l'index ou du cas mais qui pourraient être présentes dans la configuration du problème à résoudre. Chacun des attributs est représenté par un ensemble d'ensembles : on peut alors représenter des disjonctions de valeurs, mais il faut bien sûr veiller à la correspondance entre les différents ensembles de chacun des paramètres. Ainsi, on peut par exemple interdire qu'un réservoir donné comporte en amont deux conduites supplémentaires de diamètres Φ_1 et Φ_2 et avec des vannes ouvertes, ou bien comporte une conduite supplémentaire de diamètre Φ_3 avec une vanne fermée, ce qui serait modélisé par :

CN_ conduite_sup_amont.Φ = [[Φ_1, Φ_2], [Φ_3]],
CN_conduite_sup_amont.$\exists$_vanne = [[1, 1], [1]],
et CN_conduite_sup_amont.état_vanne = [[1, 1], [0]].

Remarque 2 : pour les paramètres *CN_conduite_sup_amont* et *CN_conduite_sup_aval*, il est possible de mettre en place une généralisation (voir 5.5.1. généralisation de deux cas). Elle concerne chaque ensemble de conduites supplémentaires de même cardinalité.

Remarque 3 : concernant les connaissances négatives relatives aux autres attributs, on peut soit avoir un seul ensemble de valeurs, soit avoir comme pour les conduites supplémentaires plusieurs ensembles de valeurs, c'est-à-dire un ensemble d'ensembles. Nous nous sommes limités à un seul ensemble de valeurs.

Remarque 4 : concernant les paramètres autres que connaissances négatives (diamètre, capacité, hauteur, …), on peut de la même façon soit avoir un seul ensemble de valeurs, soit avoir plusieurs ensembles de valeurs pour représenter des disjonctions. Nous nous sommes limités pour le moment à un seul ensemble de valeurs.

● *CN_bijection_ϕ* et *CN_bijection_capacité* comportent des ensembles de valeurs interdites et sont associés à chaque cas (voir 5.4 pour la signification de la bijection).

● *pénalisation* : il s'agit d'un objet dont les attributs sont les valeurs numériques associées aux différences observées entre le cas et la structure courante du réseau. Il est à noter qu'il s'agit de valeurs spécifiques, qui masquent donc les valeurs par défaut (voir mise en correspondance 5.3.1.2.), et qui ne sont créées et utilisées que si nécessaire, par exemple en cas d'échec de résolution (voir 5.5.5.). Ainsi, on a :

un objet *pénalisation*, associé à chaque cas, et dont les attributs sont :

- nombre de réservoirs
- bijection entre les valeurs de diamètre des conduites
- bijection entre les valeurs de capacité des réservoirs

un objet *pénalisation_conduite*, associé à chaque conduite de chaque cas, et dont les attributs sont :

- diamètre
- existence d'une vanne
- position d'une vanne

un objet *pénalisation_Tank*, associé à chaque réservoir de chaque cas, dont les attributs sont :

- capacité
- hauteur
- tendance
- conduite_sup_amont
- conduite_sup_aval

Les deux attributs *conduite_sup_amont* et *conduite_sup_aval* comportent eux-mêmes chacun des attributs : *∃_vanne, état_vanne*, *Φ*, *valeur* (de la pénalisation). On pourrait se contenter du seul attribut *valeur*, pour une gestion moins lourde, mais cela pourrait entraîner des conflits dus à un manque de finesse.

● enfin des attributs définissant les liens hiérarchiques entre les index, les liens d'appartenance de chaque cas à un index, et les liens hiérarchiques entre les cas d'un même index.

5.2.3. Cas et Index modélisés

Nous rappelons (voir chapitres 5.1.2. et 5.1.3.), qu'un cas représente une structure détaillée alors qu'un index représente une structure abstraite, même si les schémas ci-dessous ne modélisent pas explicitement ces différences.

Index n° 1 et cas n° 1 :

Etat : Pas d'alarme et il existe un chemin fermé.
Action : ouvrir ce chemin (c'est-à-dire ouvrir une vanne fermée).

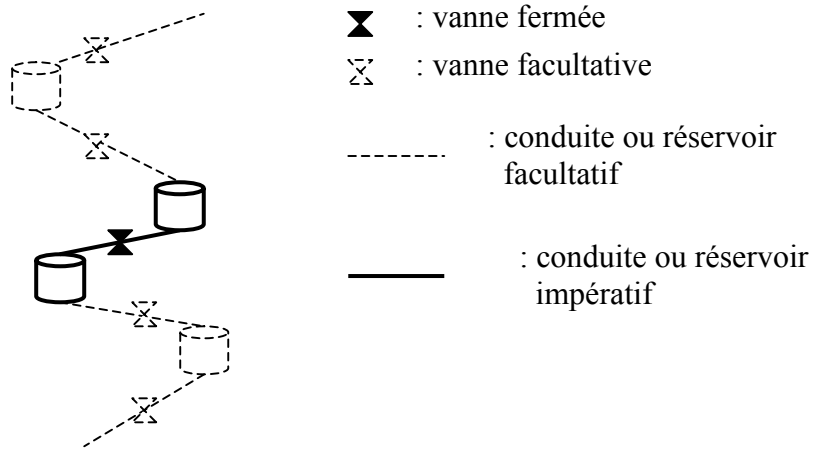


Figure 17 : Cas n° 1

Remarque : il est à noter que ce cas privilégie les chemins comportant le moins de vannes fermées, en ajoutant des pénalités lorsqu'il y a plus d'une vanne fermée, lors de la mise en correspondance.

Sans détailler l'ensemble des données, nous avons :
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].état = 0$
$\forall i',\ j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].capacité = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].tendance = INCL \vee INCS \vee DECL \vee DECS \vee STD$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].hauteur = LOW \vee HIGH$

Index n° 2 et cas n° 2 :

Etat : réservoir T_i en alarme et il existe une vanne fermée en aval de T_i.
Action : ouvrir cette vanne.

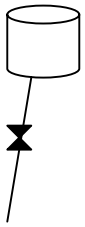


Figure 18 : Cas n° 2

Sans détailler l'ensemble des données, nous avons :
$Tank^{old}[0].état = 1$
$Gr^{old}[0][1].\Phi = P \vee M \vee G$
$Tank^{old}[0].capacité = P \vee M \vee G$
$Tank^{old}[0].tendance = INCL \vee INCS$

Index n° 3 :

Etat : réservoir T_i en alarme et il existe une vanne ouverte en amont de T_i.

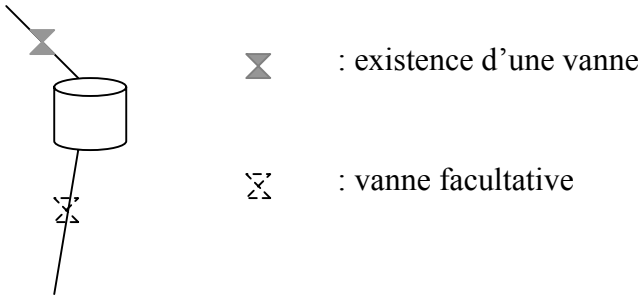


Figure 19 : Index n° 3

Sans détailler l'ensemble des données, nous avons :
$Tank^{old}[0].état = 1$
$\forall i', j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$Tank^{old}[0].capacité = P \vee M \vee G$
$Tank^{old}[0].tendance = INCL \vee INCS$

Cas n° 3-0 :

Etat : réservoir T_i en alarme et tendance de T_i = *INCL* (increase a lot) et toutes les conduites en aval de T_i = (vanne ouverte ou sans vanne) et il existe une vanne ouverte en amont de T_i.
Action : fermer la vanne de plus gros diamètre en amont de T_i.

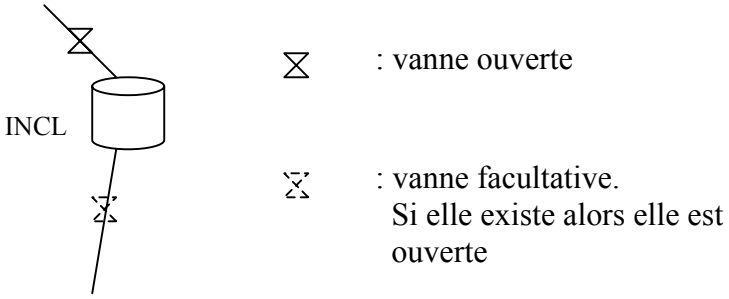


Figure 20 : Cas n° 3-0

Remarque 1 : on peut éventuellement ajouter une connaissance négative interdisant d'avoir en aval une conduite supplémentaire avec vanne fermée. Cela permet de privilégier le cas n° 2.

Remarque 2 : l'action peut être mise en oeuvre soit grâce à une procédure (ce qui ne nécessite qu'un seul cas modélisé avec une seule conduite en amont du réservoir en alarme), soit à l'aide de plusieurs cas, avec chacun une action représentée explicitement au sein du graphe, et représentant différentes configurations (plusieurs conduites en amont du réservoir en alarme, avec vannes ouvertes, et à chaque fois action sur la vanne de plus gros diamètre) (voir 5.5.3. pour plus de détails).

Cas n° 3-1 :

Etat : réservoir T_i en alarme et tendance de T_i = *INCS* (increase small) et toutes les conduites en aval de T_i = (vanne ouverte ou sans vanne) et il existe une vanne ouverte en amont de *Ti.*
Action : fermer la vanne de plus petit diamètre en amont de T_i.

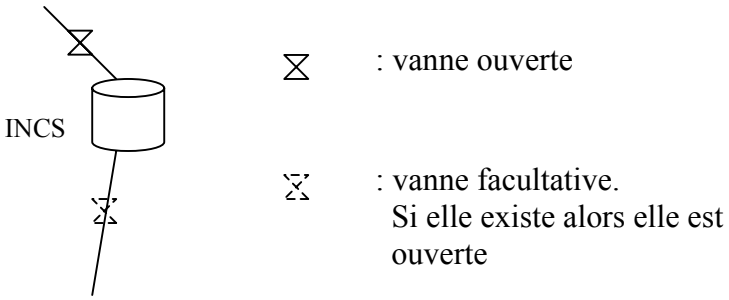


Figure 21 : Cas n° 3-1

Les remarques au sujet du cas n° 3-0 s'appliquent également ici.

Index n° 4 et cas n° 4 :

Etat : réservoir T_i en alarme et toutes les conduites en aval de T_i = (vanne ouverte ou pas de vanne) et il existe une conduite sans vanne en amont de T_i et les autres vannes (en amont de T_i) sont fermées.
Action : fermer une vanne en amont du réservoir en amont de T_i et relié à T_i par une conduite sans vanne.

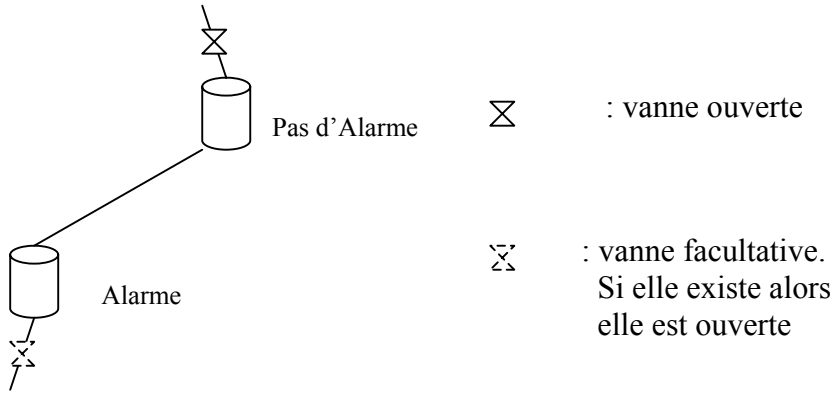


Figure 22 : Cas n° 4

Sans détailler l'ensemble des données, nous avons :
$Tank^{old}[0].état = 0$ et $Tank^{old}[1].état = 1$
$\forall i', j' \in [1, ..., n']$ $Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']$ $Tank^{old}[k'].capacité = P \vee M \vee G$
$Tank^{old}[0].tendance = INCL \vee INCS \vee DECL \vee DECS \vee STD$
$Tank^{old}[1].tendance = INCL \vee INCS$

Remarque 1 : on peut éventuellement ajouter deux connaissances négatives relatives au réservoir en alarme interdisant d'avoir en aval une conduite supplémentaire avec vanne fermée (privilégier le cas n° 2) et interdisant d'avoir en amont une conduite supplémentaire avec vanne ouverte (privilégier les cas n° 3-0 et 3-1).

Remarque 2 : Ce cas est une généralisation du cas 8 qui comporte des conditions liées à des tests hydrauliques permettant d'appliquer ce cas de manière plus judicieuse. Le cas n° 4 n'est donc pas utilisé si le cas n° 8 l'est.

Index n° 5 et cas n° 5 :

Etat : réservoir T_i en alarme et toutes les conduites en aval de T_i = (vanne ouverte ou pas de vanne) et il existe une conduite sans vanne en amont de T_i et les autres sont fermées.
Action : ouvrir une vanne en aval du réservoir en amont de T_i et relié à T_i par une conduite sans vanne.

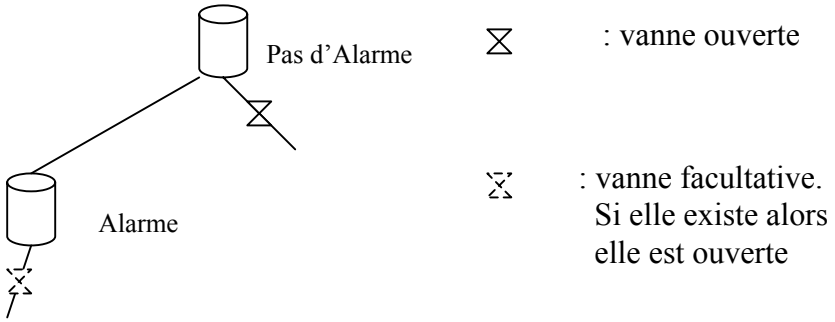


Figure 23 : Cas n° 5

Sans détailler l'ensemble des données, nous avons :
$Tank^{old}[0].état = 0$ et $Tank^{old}[1].état = 1$
$\forall i', j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].capacité = P \vee M \vee G$
$Tank^{old}[0].tendance = INCL \vee INCS \vee DECL \vee DECS \vee STD$
$Tank^{old}[1].tendance = INCL \vee INCS$

La remarque 1 pour le cas n° 4 s'applique également pour le cas n° 5.

Index n° 6 et cas n° 6 :

Etat : réservoirs T_i et T_j en alarme et reliés directement.
Action : ouvrir une vanne en aval de T_i ou T_j (sauf celle sur la conduite $i \rightarrow j$) sinon fermer une vanne en amont de T_i ou T_j (sauf celle sur la conduite $i \rightarrow j$).

En effet, ouvrir V_{ij} (si elle était fermée) va certes diminuer l'alarme sur T_i mais va augmenter l'alarme sur T_j. De même, fermer V_{ij} (si elle était ouverte) va certes diminuer l'alarme sur T_j mais va augmenter l'alarme sur T_i.

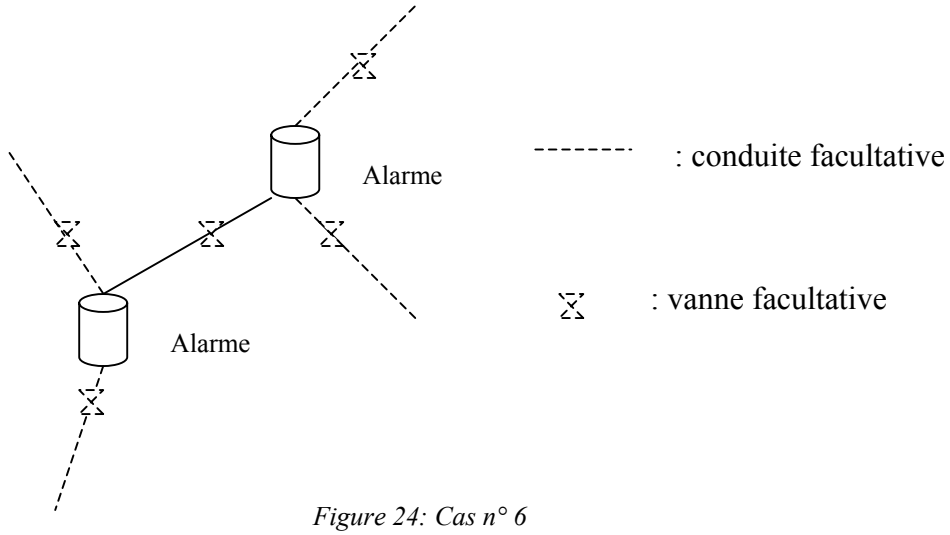


Figure 24: Cas n° 6

Sans détailler l'ensemble des données, nous avons :
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].état = 1$
$\forall i', j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].capacité = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].tendance = INCL \vee INCS$

Remarque : l'action peut être mise en oeuvre soit grâce à une procédure, soit à l'aide de quatre cas représentant les quatre actions possibles sur une vanne fermée en aval d'un des deux réservoirs et sur une vanne ouverte en amont d'un des deux réservoirs.

Index n° 7 et cas n° 7 :

Etat : m réservoirs T_1 à T_m en alarme (reliés directement ou non) ; $2 \leq m \leq$ nombre de réservoirs intermédiaires du problème à résoudre.
Action : sélectionner le réservoir qui a le plus gros rapport $\Sigma\Phi_e / \Sigma\Phi_s$ (Φ_e et Φ_s sont les diamètres entrant et sortant avec écoulement), puis lui appliquer un cas de gestion d'alarme (n° 2 à 5).

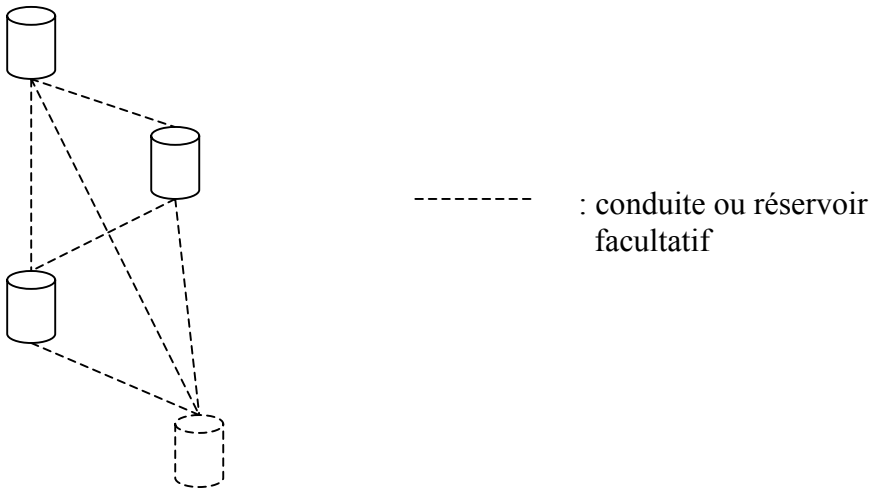


Figure 25 : Cas n° 7

Les réservoirs et conduites facultatives du schéma ci-dessus servent uniquement à calculer les rapports $\Sigma\Phi_e / \Sigma\Phi_s$. L'action est donc mise en œuvre grâce à une procédure.

Sans détailler l'ensemble des données, nous avons :
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].état = 1$
$\forall i', j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].capacité = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].tendance = INCL \vee INCS$

Index n° 8 et cas n° 8 : stratégie d'anticipation à long terme du débordement.

Etat : il s'exprime par quatre conditions.

● il existe un réservoir T_i (en alarme ou non) relié par une conduite amont sans vanne à un réservoir T_j non en alarme,
● il existe une conduite en aval de T_i (car le réservoir puit ne peut déborder),
● le diamètre de la conduite sans vanne (Φ_{ji}) est supérieur au diamètre en aval de T_i (Φ_{ir}),
● la dernière condition est plus complexe :

On se place à l'instant t courant de la simulation, et on prédit l'évolution du système dans un futur proche (instant $t + k.dt$, avec dt le pas de temps de calcul, à ne pas confondre avec le

pas de temps de calcul Δt de l'agent artificiel). La dernière condition s'exprime théoriquement à l'aide des quatre inéquations suivantes :

1. $V_i(t) + (q_{ji} - q_{ir}).kdt > V_i^{max}$

q_{ji} = débit circulant de T_j vers T_i (on suppose pour simplifier tous les débits constants entre t et $t + k.dt$)
q_{ir} = débit en aval de T_i (les vannes fermées en aval de T_i doivent être ouvertes à l'instant t).
On ne tient pas compte de l'écoulement éventuel à l'instant t en amont de T_i (autre que celui de T_j vers T_i), car s'il y a écoulement et si les toutes conditions sont remplies, alors les vannes en amont de T_i doivent être fermées à l'instant t (voir partie Action ci-dessous).

2. $V_i(t) + (q_{ji} - q_{ir}).(k-1)dt < V_i^{max}$

Les conditions 1 et 2 signifient un débordement de T_i **à partir** de l'instant $t + k.dt$.

3. $V_j(t) > (q_{jp} + q_{ji} - q_{hj}).k.dt$ (avec q_{hj} = débit entrant dans T_j)

Cela signifie que T_j ne sera pas vidangé à l'instant $t + k.dt$ si on laisse ouvert en amont de T_j.

4. $V_j(t) < (q_{jp} + q_{ji}).k.dt$

Cela signifie que s'il n'y avait pas d'apport en amont de T_j, c'est-à-dire si on fermait maintenant (à l'instant t) la vanne en amont de T_j, alors ce dernier se vidangerait avant l'instant $t + k.dt$ (q_{ji} est nul avant $t + k.dt$) et on éviterait donc le débordement de T_i.

Mais le but n'étant pas ici d'exprimer par des calculs hydrauliques complexes les conditions exactes permettant d'anticiper le débordement, mais plutôt d'approcher le raisonnement qualitatif mis en œuvre par un opérateur humain. En effet, un humain n'a pas accès au valeurs de débits et il exprime cette condition intuitivement et approximativement par : si à un instant t la somme des volumes des réservoirs T_i et T_j diminuée des volumes estimés sortant de T_i (conduite i→r) et de T_j (conduite j→p) pendant la durée de la vidange de T_j dans T_i (on ferme en amont de T_j) est supérieure à la capacité de T_i alors il y aura un débordement de T_i.

Nous avons cherché à exprimer la condition :
$(V_i(t) - \text{Volume estimé sortant de } T_i + \text{Apport estimé de } T_j) > V_i^{max}$
grâce à des calculs hydrauliques simples sous la forme :

$\boldsymbol{(V_i(t) + \alpha(\Phi_{ji}, \Phi_{ir}, \Phi_{jp}).V_j(t)) \geq V_i^{max}}$ (les détails de ce calcul sont présentés en annexe 8).

Action : fermer la (ou les) vanne en amont du réservoir T_j et fermer la (ou les) vanne en amont de T_i et ouvrir la (ou les) vanne en aval de T_i et ouvrir la (ou les) vannes en aval de T_j.

NB : cette action a été mise en œuvre grâce à une procédure testant la condition d'application ci-dessus.

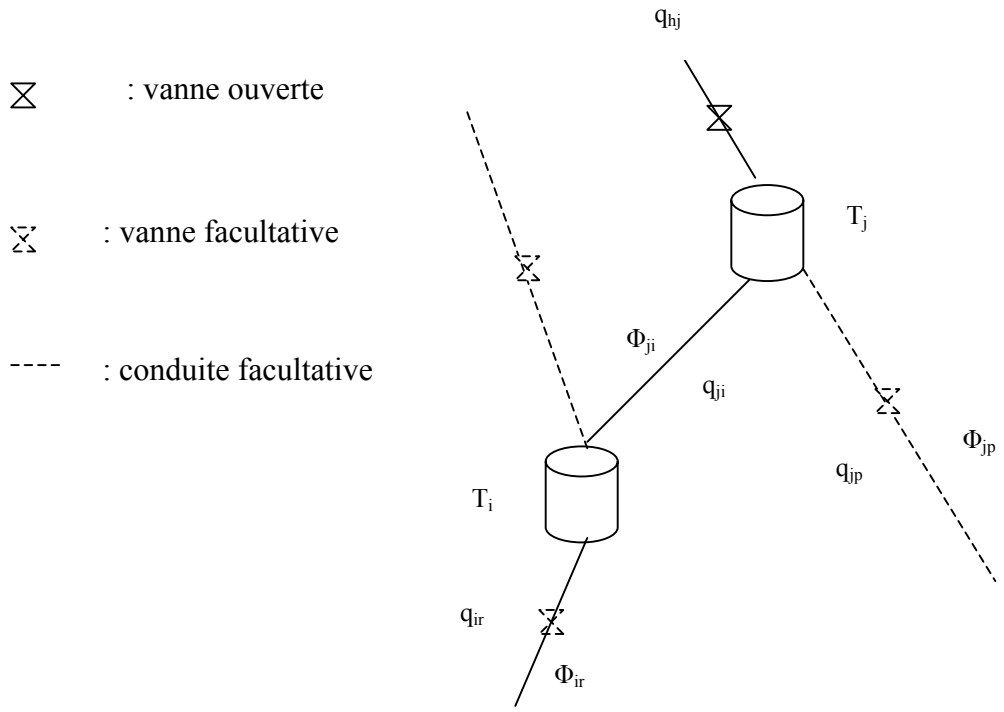


Figure 26 : Cas n° 8

Sans détailler l'ensemble des données, nous avons :
$\forall k' \in [1, ..., n']$ $Tank^{old}[k'].état = 0 \vee 1$
$\forall i', j' \in [1, ..., n']$ $Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']$ $Tank^{old}[k'].capacité = P \vee M \vee G$
$\forall k' \in [1, ..., n']$ $Tank^{old}[k'].tendance = INCL \vee INCS \vee DECL \vee DECS \vee STD$

Remarque : cette règle peut aller à l'encontre de l'objectif de minimisation de la durée totale.

Index n° 9 : stratégie de minimisation de la durée totale de la simulation.

Etat : m réservoirs T_1 à T_m (en alarme ou non) reliés au réservoir puit $T_{n'}$. $2 \leq m \leq$ nombre de réservoirs intermédiaires du problème à résoudre. Si on se trouve dans la situation telle que :

$V_i \approx$ *Volume Optimal* (T_i) $\forall\ i \in [1 ; m]$ (pour le calcul du *Volume Optimal* (T_i), voir annexe 9)

Alors les réservoirs T_1 à T_m peuvent se vidanger dans $T_{n'}$ selon la même durée, si ces vidanges soient indépendantes, c'est-à-dire que l'on ferme les vannes sur les conduites reliant directement ces réservoirs.

Action : si $\exists\ i \in [\ 1 ; m\ ]$ $V_i \neq$ *Volume Optimal* (T_i) alors il faut agir sur T_i pour ramener son volume à la valeur optimale.

NB : comme cela a été dit, l'action n'est pas représentée au niveau de l'index, mais au niveau des cas, en l'occurrence les cas n° 9-0 et 9-1 suivants.

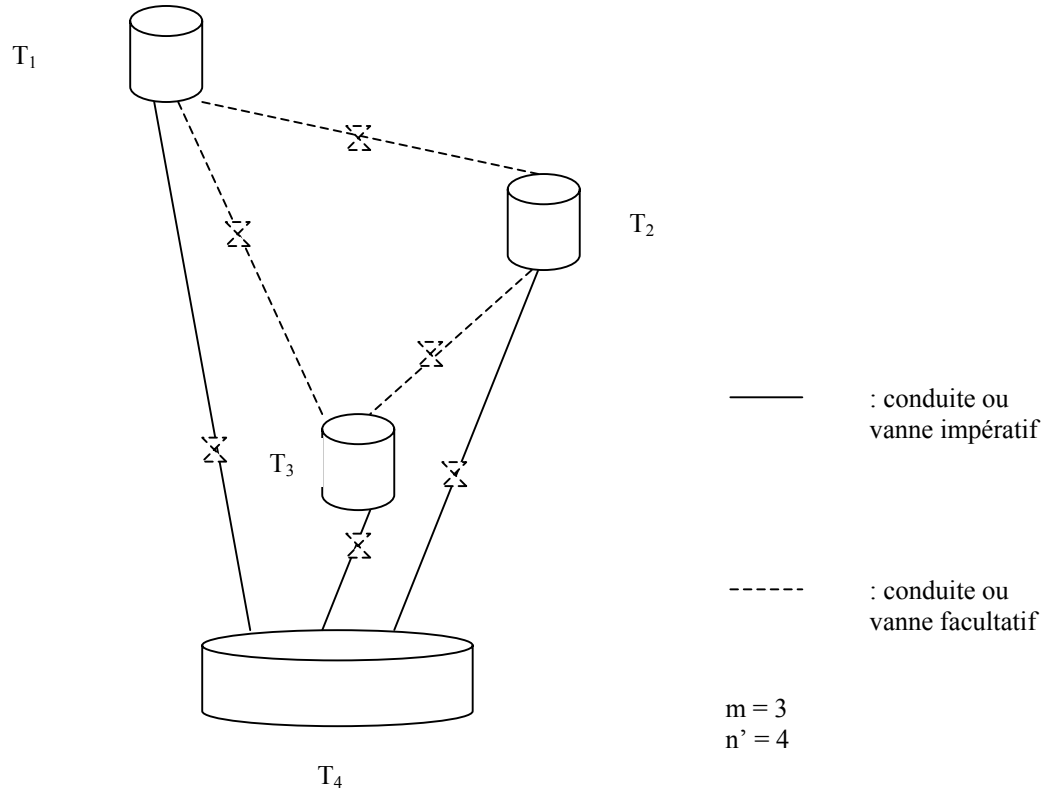


Figure 27 : Index n° 9

Sans détailler l'ensemble des données, nous avons :
$\forall k' \in [1, ..., n'-1]\ \ Tank^{old}[k'].état = 0 \vee 1$
$Tank^{old}[n'].état = 0$
$\forall i', j' \in [1, ..., n']\ \ Gr^{old}[i'][j'].\Phi = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].capacité = P \vee M \vee G$
$\forall k' \in [1, ..., n']\ \ Tank^{old}[k'].tendance = INCL \vee INCS \vee DECL \vee DECS \vee STD$

Cas n° 9-0 :

Etat : réservoir T_i (en alarme ou non) et relié directement à $T_{n'}$ (réservoir puit) et $V_i < Volume\ Optimal\ (T_i)$.
Action : remplir T_i (c'est-à-dire ouvrir juste en amont et/ou fermer juste en aval).

Remarque 1 : il est préférable de remplacer la condition $V_i < Volume\ Optimal\ (T_i)$ par $V_i < (Volume\ Optimal\ (T_i) - V\varepsilon)$, $V\varepsilon$ définissant une marge autour de la valeur $Volume\ Optimal\ (T_i)$ (voir aussi Remarque 1 relative au cas n° 9-1), car le but ici est de s'approcher du comportement d'un humain qui ne peut définir ce volume de manière exacte et constante tout au long de la simulation, mais plutôt comme une zone (floue).

Remarque 2 : l'action "remplir T_i" peut être réalisée grâce à une procédure, ou bien à l'aide de deux cas représentant les situations où l'on ouvre juste en amont et où l'on ferme juste en aval. D'autre part, l'action "fermer juste en aval" doit être appliquée avec précaution car elle va à l'encontre du but qui est de vidanger le plus rapidement possible : elle n'est appliquée que si il est également préconisé de vidanger le réservoir se situant à l'autre extrémité (avale) de la vanne à actionner. D'autre part, l'action "ouvrir juste en amont" n'est pas appliquée s'il

est également préconisé de remplir le réservoir se situant à l'autre extrémité (amont) de la vanne à actionner.

Remarque 3 : il est tout de même nécessaire d'avoir une procédure de calcul des volumes optimaux.

Remarque 4 : une exception existe pour les deux cas n° 9-0 et 9-1, si tous les réservoirs situés en amont des *m* réservoirs T_1 à T_m reliés au réservoir puit $T_{n'}$ sont vides. Dans cette situation, l'action consiste à fermer les vannes sur les conduites reliant directement ces *m* réservoirs afin que la vidange finale se fasse selon la même durée.

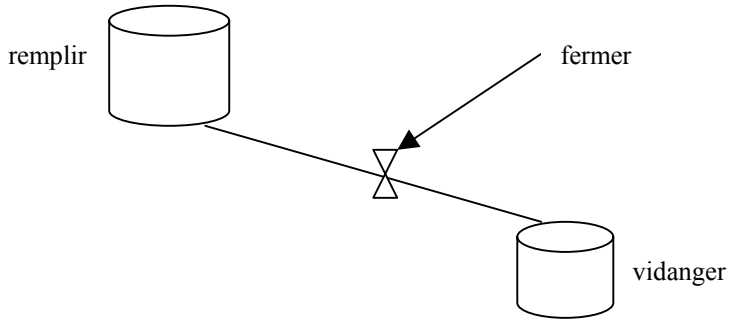


Figure 28 : Action "remplir un réservoir" du cas n° 9-0

Cas n° 9-1 :

Etat : réservoir T_i (en alarme ou non) et relié directement à $T_{n'}$ (réservoir puit) et $V_i > Volume\ Optimal\ (T_i)$.
Action : vidanger T_i (c'est-à-dire fermer juste en amont et/ou ouvrir jute en aval et/ou agir plus en amont ou plus en aval si T_i est relié par des conduites sans vanne).

Remarque 1 : il est préférable de remplacer la condition $V_i > Volume\ Optimal\ (T_i)$ par $V_i > (Volume\ Optimal\ (T_i) + V\varepsilon)$, $V\varepsilon$ définissant une marge autour de la valeur *Volume Optimal* (T_i). Ainsi, si $(Volume\ Optimal\ (T_i) - V\varepsilon) < V_i < (Volume\ Optimal\ (T_i) + V\varepsilon)$, alors on se situe dans la zone optimale, donc on n'a pas à agir. On définit donc la zone optimale ***ZO*** de volume $\Delta V\_ZO = 2.V\varepsilon$ autour de la valeur *Volume Optimal* (T_i). Dans cette zone, on considère que l'on est proche du volume optimal et donc on n'agit pas.

Remarque 2 : l'action "vidanger T_i" peut être réalisée grâce à une procédure, ou bien à l'aide de deux cas représentant les situations où l'on ferme juste en amont et où l'on ouvre juste en aval. L'action "fermer juste en amont" n'est pas appliquée s'il est également préconisé de vidanger le réservoir se situant à l'autre extrémité (amont) de la vanne à actionner, et l'action "ouvrir juste en aval" n'est pas appliquée s'il est également préconisé de vidanger le réservoir se situant à l'autre extrémité (aval) de la vanne à actionner.

Remarque 3 : si plusieurs volumes optimaux calculés sont faibles, alors cette stratégie a peu d'intérêt, car le but est ici de répartir le volume total dans ces trois réservoirs pour les vidanger selon la même durée.

Remarque 4 : lorsque cette stratégie est judicieuse, on peut même l'appliquer à un ensemble de réservoirs constituant une couche intermédiaire. Dans le cas ci-dessous, on pourrait l'appliquer d'une part à l'ensemble (T_2, T_3, T_4), et d'autre part à l'ensemble (T_5, T_6, T_7). Il

s'agit donc de repérer les différentes couches de réservoirs, afin que les réservoirs de la couche *k-1* se vidangent selon la même durée dans les réservoirs de la couche *k*.

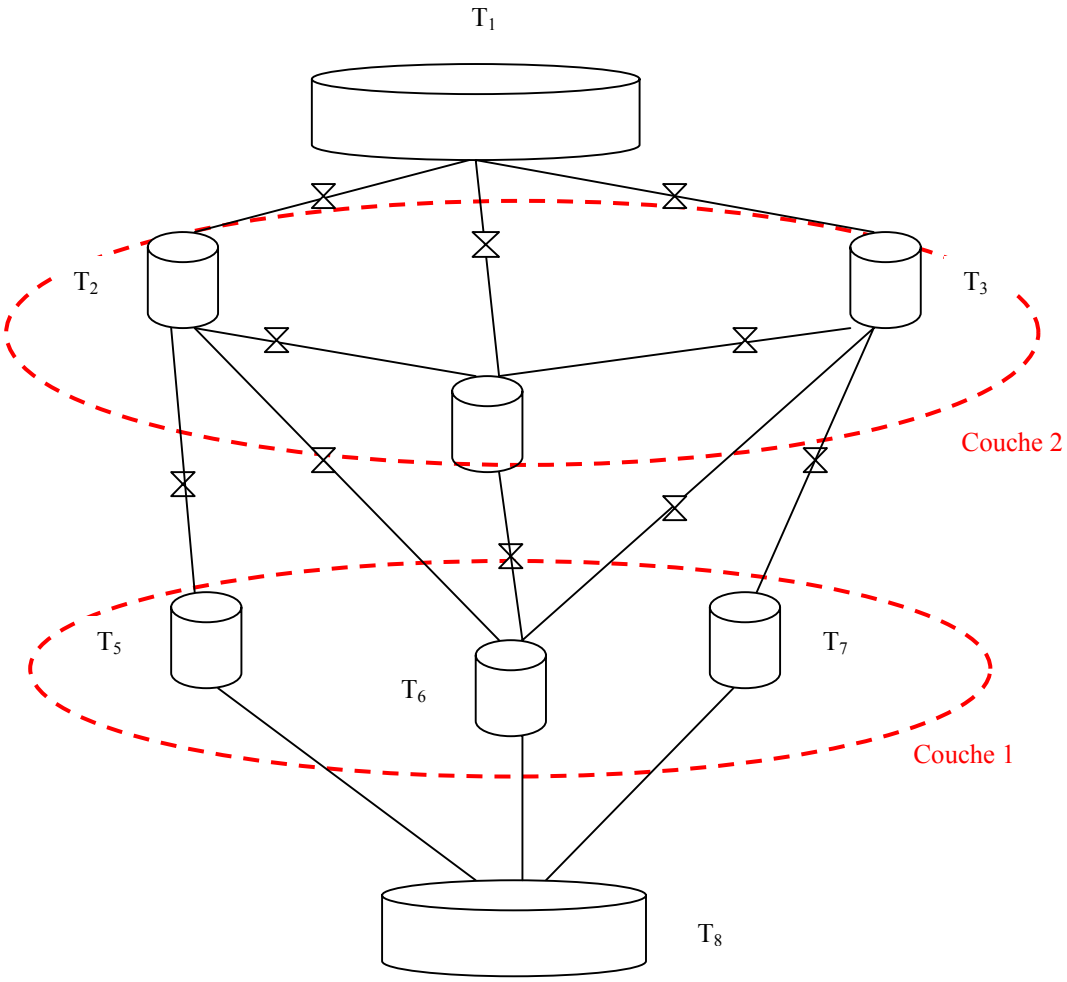


Figure 29 : Application généralisée du cas n°9 grâce à une organisation du réseau en couches de réservoirs

Remarque générale : concernant tout ou partie des règles de gestion des alarmes, on pourrait mettre en place une stratégie semblable à celle mise en place dans la méthode d'optimisation par recherche *Tabou* : les conduites correspondant aux actions effectuées pour gérer les alarmes sont placées dans une *liste Tabou* et y restent pendant un certain nombre de pas de temps (2 ou 3.Δt par exemple). Toute vanne sur une conduite qui se trouve dans la liste Tabou ne peut pas être actionnée. Cela permet d'ajouter de la cohérence dans les actions par effet de mémoire.

5.3. PHASE DE RECHERCHE DE CAS SIMILAIRES

La phase de recherche est basée sur la mise en correspondance de deux graphes, celui représentant le problème à résoudre et celui représentant le cas/index en mémoire. Il s'agit donc de définir la similarité entre deux graphes orientés étiquetés.

L'algorithme actuel opère en deux phases :

● recherche dans la hiérarchie (figure 16) d'un but de haut niveau (accélérer le processus, ...) correspondant à la situation courante à gérer, puis première discrimination grossière, au

niveau des index placés sous ce but, sur deux paramètres : le nombre de réservoirs représentés et la présence ou non de conduite sans vanne.

● recherche dans la hiérarchie correspondant au but de haut niveau à atteindre (figure 16) d'une configuration partielle de réseau (structure et état), sous la forme d'un graphe orienté étiqueté, correspondant à la situation à gérer. On effectue cette recherche en deux temps : recherche d'un index similaire, puis recherche d'un cas similaire.

- recherche d'un (ou plusieurs) index de manière approfondie (voir 5.3.1.1) en se basant sur la configuration partielle abstraite et l'état abstrait (graphe orienté étiqueté). Concernant le cas "sans alarme", un ordre a été mis en place, car il s'agit de chercher si possible à appliquer une stratégie de minimisation globale de la durée de la simulation. Si on ne trouve pas de cas à appliquer, alors on se tente une accélération locale du processus.

- puis, au sein de cet (ces) index, recherche d'un (ou plusieurs) cas en se basant sur la configuration partielle concrète et l'état concret (graphe orienté comportant des informations supplémentaires par rapport à celui de l'index) et en calculant un coût (voir 5.3.1.2).

Remarque : même si le système n'applique qu'un seul cas à chaque pas de temps, il pourrait être intéressant de conserver un classement des cas retrouvés, notamment lorsque le système commet une erreur (selon un professeur) dû à un mauvais choix de cas, alors que le bon cas existait en mémoire (voir 5.5.5. et 5.5.6.).

5.3.1. Mise en correspondance de deux graphes

Il s'agit de manière générale d'effectuer un "mapping" entre le problème à résoudre et un cas en mémoire (ou un index) à chaque pas de temps. Il y a deux procédures de mise en correspondance : la première procédure utilise le problème à résoudre et un cas (ou un index) en mémoire, et il s'agit alors de savoir si la structure en mémoire est présente dans le problème, puis elle attribue un score numérique reflétant les différences observées, qui peut être vu comme une *mesure de similarité*. La seconde, très voisine, utilise deux cas en mémoire, et est utilisée lorsqu'il s'agit de discriminer ces deux cas afin de créer des connaissances négatives lors de l'apprentissage par l'échec (voir 5.5.6.).

Nous présentons la procédure de recherche du meilleur cas correspondant à la situation présente, qui se déroule en deux étapes :

5.3.1.1. Mise en correspondance au niveau de l'index

Il s'agit tout d'abord de trouver des index dont on observe la structure quelque part au sein de la situation présente. Cette procédure va non seulement chercher les index potentiels mais également va tester les différentes façons de mettre en correspondance l'index et la situation courante (le problème à résoudre à l'instant t). Il s'agit là du point délicat de l'algorithme.

Un test des connaissances négatives associé à l'index est effectué pour chaque mise en correspondance possible entre l'index et la situation courante. Si le test est positif, alors l'index est enregistré comme potentiellement utile et la mise en correspondance est également enregistrée.

On a ainsi deux structures de données en sortie :

● *cout (i, n)* le coût associée à la *n* ième mise en correspondance entre l'index *i* et le problème à résoudre (donc *n* est fonction de *i*), utilisée lors de la seconde phase (voir 5.3.1.2)

● *mec (i,n)* le *n* ième mise en correspondance (mec), décomposée en deux sous-structures :

- *mec_tank (i, n, k)* : mise en correspondance entre deux réservoirs (la *k* ième au sein de la *n* ième mise en correspondance entre les deux structures, donc *k* est fonction de *n*)

- *mec_pipe (i, n, m)* : mise en correspondance entre deux conduites (la *m* ième au sein de la *n* ième mise en correspondance entre les deux structures, donc *m* est fonction de *n*)

De manière plus précise, on effectue tout d'abord une mise en correspondance des caractéristiques de base (buts notamment), afin de trouver un premier ensemble d'index potentiellement intéressants :

● but de haut niveau (réduire une situation mono alarme, accélérer le processus, …)

● nombre de réservoirs en alarme

● nombre de réservoirs concernés par la gestion de la situation courante

● présence de conduite sans vanne

Ensuite on effectue une mise en correspondance entre la structure de la situation courante ($Tank^{new}$, Gr^{new}) et la (sous) structure de l'index ($Tank^{old}$, Gr^{old}). Il s'agit donc de rechercher la structure ($Tank^{old}$, Gr^{old}) d'un index au sein de la structure ($Tank^{new}$, Gr^{new}) de la situation courante. On doit donc trouver *I, J, K* des parties de l'ensemble [1, ..., *nT*], où *nT* est le nombre de réservoirs du problème à résoudre, tels que :

- $|I| = |J| = |K| = nT'$ (nT' = nombre de réservoirs de l'index)
- $\forall (i \in I, j \in J, k \in K)\ \exists (i' \in I', j' \in J', k' \in K')\ Gr^{new}[i, j] = Gr^{old}[i'][j']$ et $Tank^{new}[k] = Tank^{old}[k']$

Les égalités ci-dessus doivent être interprétées de la façon suivante : il y a correspondance entre l'objet $Gr^{new}[i, j]$ et l'objet $Gr^{old}[i'][j']$ si la valeur de chaque attribut de $Gr^{new}[i, j]$ appartient à l'ensemble représentant l'attribut correspondant de $Gr^{old}[i'][j']$ (idem pour $Tank^{new}[k]$ et $Tank^{old}[k']$).

Nous avons distingué la situation sans alarme de l'index n° 1 des autres situations.

● Pour l'index n° 1 (une chaîne de réservoirs avec au moins une vanne fermée),

n = 1; // initialisation : nombre de mise en correspondance entre les deux structures
Pour chaque conduite c_0 ($Gr^{new}[i_0, j_0]$) avec une vanne fermée de la situation courante
m = k = 1 ; // initialisations : nombre de mises en correspondance pour les conduites
// et pour les réservoirs (voir notations p. 76)
mec_pipe (1, n, m) = mise en correspondance entre c_0 et la conduite comportant la vanne fermée de l'index n° 1 ;
mec_tank (1, n, k) et *mec_tank (1, n, k+1)* = mises en correspondance entre les deux réservoirs en amont et en aval de ces conduites ;
m++ ;
k+=2 ;
Pour toute conduite c_1 de la situation courante en amont de la conduite c_0
mec_pipe (1, n, m) = mise en adéquation entre c_1 et la conduite correspondante en amont de l'index n° 1 ;
mec_tank (1, n, k) = mise en correspondance entre les deux réservoirs en amont de ces conduites ;
Si c_1 comporte une vanne fermée Alors {*cout (1, n)* += pénalité ;} // on privilégie // les chemins comportant le moins de vannes fermées en ajoutant des pénalités lorsqu'il y a // plus d'une vanne fermée
m++ ;
k++ ;
$c_0 = c_1$;
Pour toute conduite c_2 de la situation courante en aval de la conduite c_0
mec_pipe (1, n, m) = mise en adéquation entre c_1 et la conduite correspondante en aval de l'index n° 1 ;
mec_tank (1, n, k) = mise en correspondance entre les deux réservoirs en aval de ces conduites ;
Si c_2 comporte une vanne fermée Alors {*cout (1, n)* += pénalité ;}
m++ ;
k++ ;
$c_0 = c_2$;
n++ ;
Choix de la mise en correspondance n_0 qui minimise *cout (1, n)* ;

Cet algorithme est donc exhaustif quant à la recherche des différentes mises en correspondance possibles.

● Pour les index avec au moins un réservoir en alarme, on a préalablement organisé les réservoirs en couches hiérarchiques, afin de définir un ordonnancement de haut en bas et faciliter la mise en correspondance. Par exemple, pour la configuration de la figure 30, il y a trois couches : la première pour le réservoir *Tank[0]*, la seconde pour *Tank[1]* et *Tank[2]*, et la dernière pour *Tank[3]*.

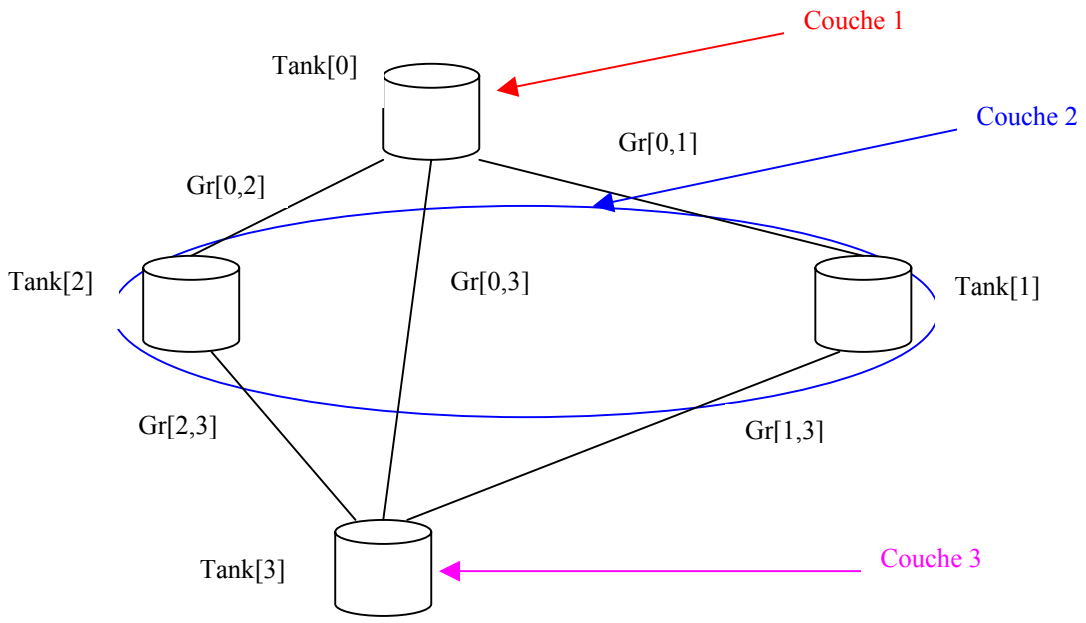


Figure 30 : Numérotation des éléments du réseau en fonction de l'organisation en couches

L'algorithme se déroule en trois phases.

Phase 1 :

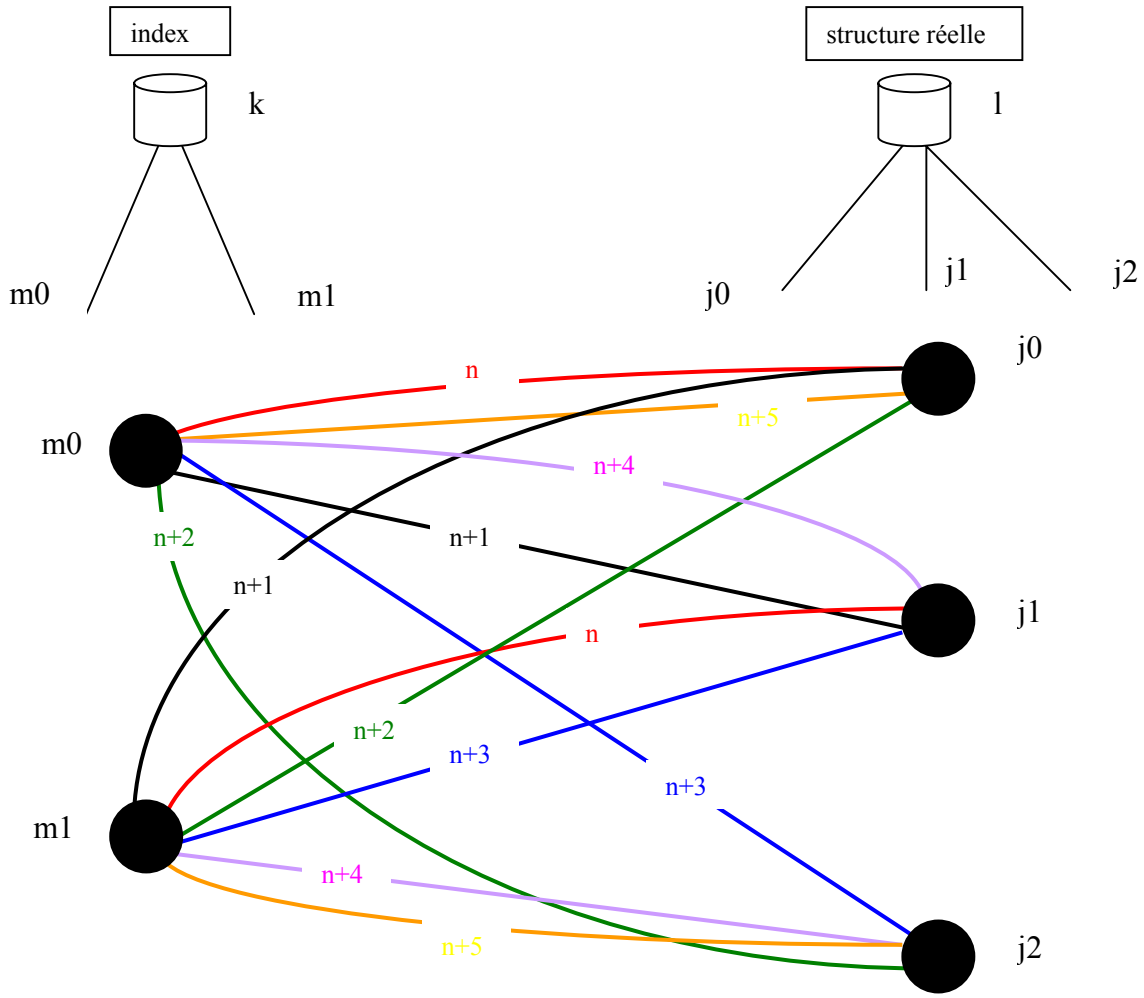
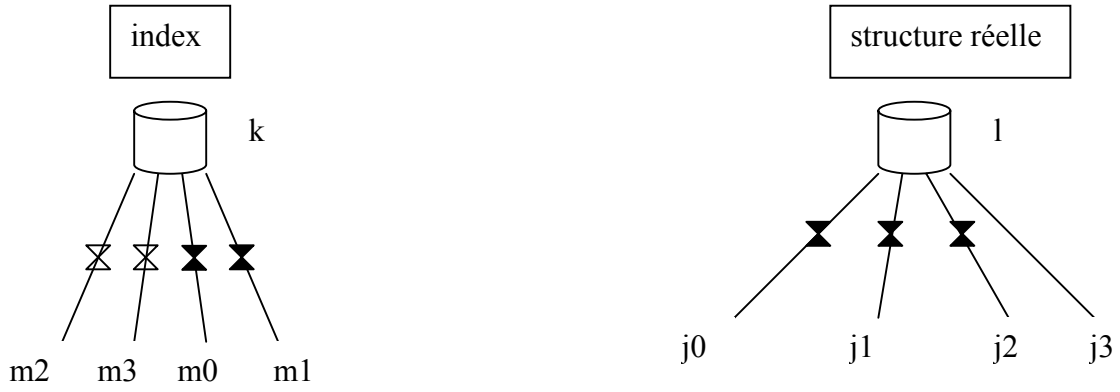


Figure 31 : Mises en correspondance des conduites de même type de deux réservoirs

Le schéma figure 31 montre à travers un exemple le principe de base : les réservoirs *k* et *l* sont mis en adéquation. Il y a ensuite six façons (numérotées de *n* à *n+5*) de mettre en correspondance les deux conduites *k→m0* et *k→m1* avec les trois conduites *l→j0*, *l→j1* et *l→j2*.

Le processus est en fait plus complexe que ce schéma de base car il faut tenir compte des caractéristiques des conduites (avec vanne, sans vanne, …), et également tenir compte des mises en correspondance faites lors des étapes précédentes : il s'agit notamment, lorsque l'on évalue les différentes façons de mettre en adéquation les conduites, de savoir s'il faut créer une nouvelle mise en correspondance (incrémentation de *n* dans la figure 31 et dans *mec_pipe(i,n,m)*) ou si l'on ajoute des correspondances entre conduites (incrémentation de l'indice *m* dans *mec_pipe(i,n,m)*) à une mise en adéquation existante (indice *n*). Ceci est assuré par différents tests.



⧗ : conduite avec vanne

⧖ : conduite avec ou sans vanne

Figure 32 : Mises en correspondance des conduites de types différents de deux réservoirs

D'après le schéma figure 32, les conduites avec vanne *k→m0* et *k→m1* sont mises en adéquation avec les conduites avec vanne *l→j0*, *l→j1* et *l→j2*, puis les conduites "avec ou sans" vanne *k→m2* et *k→m3* sont mises en correspondance avec l'une des conduites avec vanne et avec la conduite sans vanne *l→j3*, soit au total six possibilités.

Détails de l'algorithme (les notations sont celles décrites p. 86) :

Pour i = 2 à nombre d'index
    $n = k = m = 0$; // *initialisations des nombres de mises en correspondance*
    Trouver les deux réservoirs en alarme situés les plus hauts (un dans l'index noté $Tank^{old}[k_1]$ et un dans le réseau courant noté $Tank^{new}[k_2]$) ;
    mec_tank (i, n+1, k+1) = mise en correspondance entre ces deux réservoirs ;
    n++ ;
    k++ ;
    mec(amont, k_1, k_2*)* ;
    mec(aval, k_1, k_2*)* ;

mec(x, $Tank^{old}$, $Tank^{new}$)
// x = amont ou aval
// effectue toutes les mises en correspondance possible entre les conduites situés en *x* des 2
// réservoirs $Tank^{old}$ et $Tank^{new}$ mis en adéquation précédemment
{
Pour toute conduite c_1 avec vanne de l'index *i* en *x* de $Tank^{old}$
Pour toute conduite c'_1 avec vanne du réseau courant en *x* de $Tank^{new}$
Si *test_mec_pipe ( )* = vrai
Si *test_nouvelle_mec ( )* = vrai
Alors *{n++ ; m = 0 ; mec_tank (i, n, k) = mec_tank (i, n-1, k) ;}*
Sinon {*(n , m) = recherche_mec_existantes ( ) ;}*
mec_pipe (i, n, m+1) = mise en adéquation entre c_1 et c'_1 ;
m++ ;
Pour toute conduite c_2 sans vanne de l'index *i* en *x* de $Tank^{old}$
Pour toute conduite c'_2 sans vanne du réseau courant en *x* de $Tank^{new}$
Si *test_mec_pipe ( )* = vrai
Si *test_nouvelle_mec ( )* = vrai
Alors *{n++ ; m = 0 ; mec_tank (i, n, k) = mec_tank (i, n-1, k) ;}*
Sinon {*(n , m) = recherche_mec_existantes ( ) ;}*
mec_pipe (i, n, m+1) = mise en correspondance entre c_2 et c'_2 ;
m++ ;
Pour toute conduite c_3 avec ou sans vanne (les deux valeurs peuvent être possibles pour certains index) de l'index *i* en *x* de $Tank^{old}$
Pour toute conduite c'_3 non encore mise en correspondance du réseau courant en *x* de $Tank^{new}$
Si *test_mec_pipe ( )* = vrai
Si *test_nouvelle_mec ( )* = vrai
Alors *{n++ ; m = 0 ; mec_tank (i, n, k) = mec_tank (i, n-1, k) ;}*
Sinon {*(n , m) = recherche_mec_existantes ( ) ;}*
mec_pipe (i, n, m+1) = mise en correspondance entre c_3 et c'_3 ;
m++ ;
Pour toute conduite c_4 sans vanne de l'index *i* en *x* de $Tank^{old}$
Pour toute conduite c'_4 non encore mise en correspondance du réseau courant en *x* de $Tank^{new}$
Si *test_mec_pipe ( )* = vrai
Si *test_nouvelle_mec ( )* = vrai
Alors *{n++ ; m = 0 ; mec_tank (i, n, k) = mec_tank (i, n-1, k) ;}*
Sinon {*(n , m) = recherche_mec_existantes ( ) ;}*
mec_pipe (i, n, m+1) = mise en correspondance entre c_4 et c'_4 ;
m++ ;
Pour toute conduite c_5 avec vanne de l'index *i* en *x* de $Tank^{old}$
Pour toute conduite c'_5 non encore mise en correspondance du réseau courant en *x* de $Tank^{new}$
Si *test_mec_pipe ( )* = vrai
Si *test_nouvelle_mec ( )* = vrai
Alors *{n++ ; m = 0 ; mec_tank (i, n, k) = mec_tank (i, n-1, k) ;}*
Sinon {*(n , m) = recherche_mec_existantes ( ) ;}*
mec_pipe (i, n, m+1) = mise en correspondance entre c_5 et c'_5 ;
m++ ;
}

test_mec_pipe ( )
*{*renvoie faux si les connaissances négatives associées à l'index i ne sont pas respectées ou si les deux réservoirs en amont des conduites mises en adéquation ne sont pas dans le même état (alarme ou non) ou si les deux réservoirs en aval des conduites mises en adéquation ne sont pas dans le même état.*}*

test_nouvelle_mec ( )
*{*renvoie vrai si la mise en correspondance entre les deux conduites introduit une nouvelle mise en correspondance possible entre les deux structures, et renvoie faux si la mise en correspondance entre les deux conduites peut être insérée dans une mise en correspondance existante entre les deux structures (voir figure 31)*}*

recherche_mec_existantes ( )
*{*recherche les mises en correspondance existantes entre les deux structures (notées *n* : voir p. 76) telles que la mise en adéquation entre les deux conduites testées puisse y être insérée, et recherche le nombre de mises en adéquation existantes entre conduites (notée *m* : voir p. 76) correspondant à chaque *n*.*}*

Phase 2 : Ensuite on met en adéquation deux autres réservoirs en alarme, si l'index et la configuration du problème à résoudre mettent en avant tous deux un second réservoir en alarme, puis on effectue comme précédemment les mises en correspondance au niveau des conduites en amont et en aval de ces réservoirs.

Pour i = 2 à nombre d'index
Si on trouve deux autres réservoirs en alarme // notés $Tank^{old}[k'_1]$ et $Tank^{new}[k'_2]$ et // situés en aval ou au même niveau (voir figure 30) que les 2 réservoirs précédents // $Tank^{old}[k_1]$ et $Tank^{new}[k_2]$
Pour n = 1 à nombre de mises en correspondance déjà réalisées entre les 2 structures
Si (les conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$ existent et il existe m tel que *mec_pipe (i, n, m)* = mise en correspondance entre les deux conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$) ou (les conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$ n'existent pas)
Alors {
mec_tank (i, n, K+1) = mise en correspondance entre les deux réservoirs $Tank^{old}[k'_1]$ et $Tank^{new}[k'_2]$ (K est le nombre de mises en correspondance entre réservoirs) ;
mec(aval, k'_1, k'_2) ;
K++ ;
Si les conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$ n'existent pas
Alors *mec(amont, k'_1, k'_2) ; }*

Remarque : l'algorithme concerne également les index n° 8 et 9 (qui s'appliquent à des réservoirs en alarme ou non), où on remplace, lors des deux premières phases précédentes, les mises en adéquation entre deux réservoirs en alarme par des mises en adéquation entre deux réservoirs non en alarme.

Phase 3 : Enfin, on met en adéquation tous les couples possibles de réservoirs qui ne sont pas en alarme, et qui sont reliés par une conduite sans vanne à un des réservoirs en alarme mis en correspondance précédemment (phases 1 et 2). Ces deux réservoirs non en alarme sont bien sûr tous deux en amont ou tous deux en aval des réservoirs en alarme (figure 33). Puis on effectue comme précédemment les mises en correspondance au niveau des conduites en amont et en aval de ces réservoirs non en alarme.

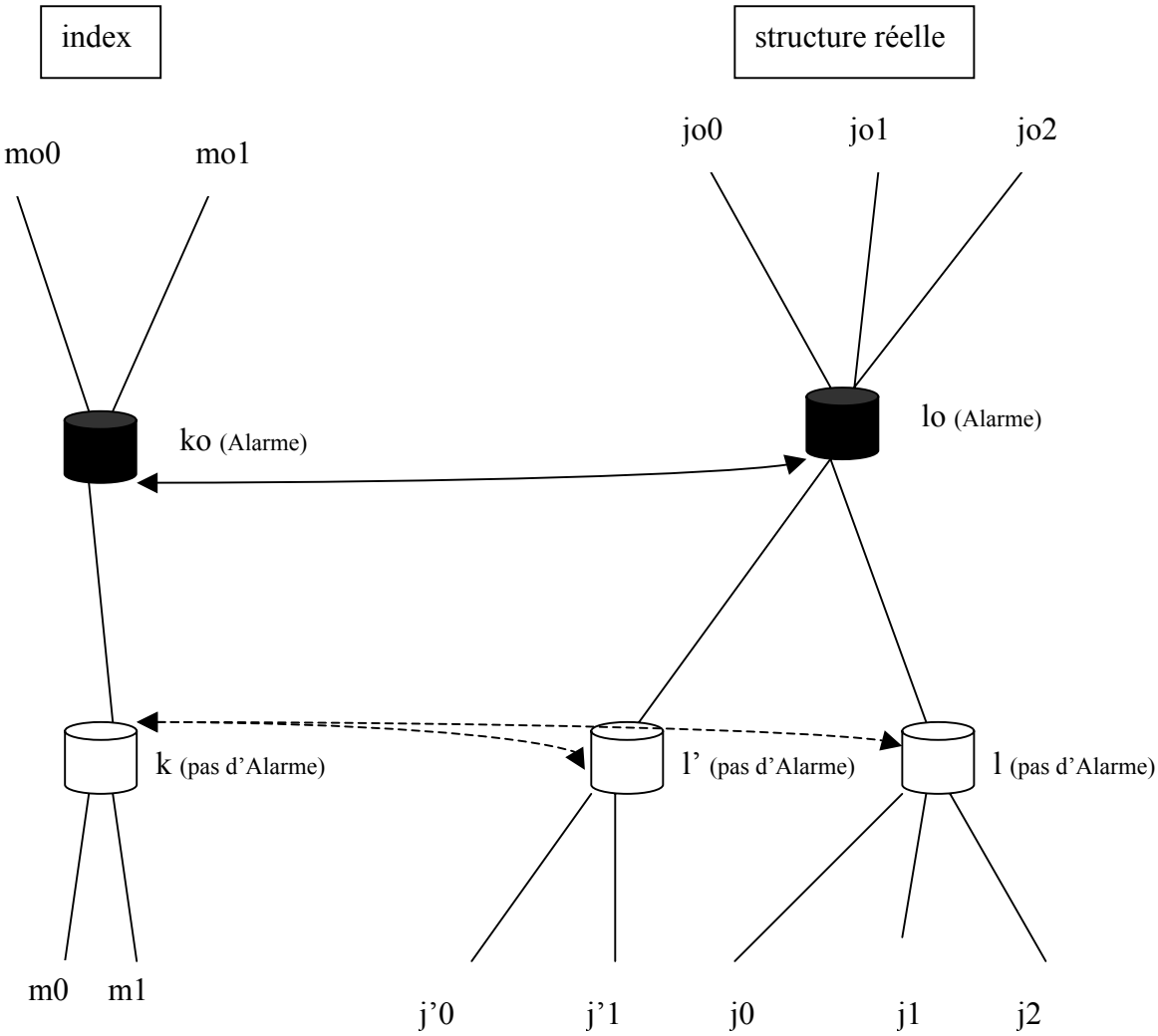


Figure 33 : Mises en correspondance entre des réservoirs

On voit d'après ce schéma qu'il peut y avoir plusieurs réservoirs non en alarme (*l* et *l'*) du problème à résoudre qui peuvent être mis en adéquation avec le réservoir *k* non en alarme de l'index. Ensuite, les mises en adéquation entre les conduites $k \rightarrow mi$ (i = 0 à 1) et $l \rightarrow ji$ (i = 0 à 2) ou $l' \rightarrow j'i$ (i = 0 à 1) doivent tenir compte des mises en correspondance précédentes, notamment entre les conduites $ko \rightarrow moi$ (i = 0 à 1) et $lo \rightarrow joi$ (i = 0 à 2) liées aux réservoirs en alarme (*ko* et *lo*), afin de savoir s'il faut créer une nouvelle mise en correspondance ou si l'on ajoute des adéquations entre conduites à une mise en correspondance existante. Ceci nécessite des tests supplémentaires insérés dans les procédures *test_nouvelle_mec( )* et *recherche_mec_existantes( )*.

Pour i = 2 à nombre d'index
Pour n = 1 à nombre de mises en correspondance entre les 2 structures
Pour k = 1 à K (nombre de mises en correspondance entre 2 réservoirs)
$K' = K$;
Si il existe *mec_tank (i, n, k)* une mise en correspondance entre deux réservoirs en alarme (notés $Tank^{old}[k_1]$ et $Tank^{new}[k_2]$)
Pour tout réservoir non en alarme $Tank^{old}[k'_1]$ de l'index i en aval de $Tank^{old}[k_1]$ relié par une conduite sans vanne $Gr^{old}[k_1][k'_1]$
Pour tout réservoir non en alarme $Tank^{new}[k'_2]$ de la structure courante en aval de $Tank^{new}[k_2]$ relié par une conduite sans vanne $Gr^{new}[k_2][k_2']$
Si il existe m tel que *mec_pipe (i, n, m)* = mise en correspondance entre les 2 conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$ Alors {
mec_tank (i, n, K'+1) = mise en correspondance entre les deux réservoirs $Tank^{old}[k'_1]$ et $Tank^{new}[k'_2]$;
mec(aval, k'_1, k'_2) ;
K'++ ;}
Pour tout réservoir non en alarme $Tank^{old}[k'_1]$ de l'index i en amont de $Tank^{old}[k_1]$ relié par une conduite sans vanne $Gr^{old}[k'_1][k_1]$
Pour tout réservoir non en alarme $Tank^{new}[k'_2]$ de la structure courante en amont de $Tank^{new}[k_2]$ relié par une conduite sans vanne $Gr^{new}[k'_2][k_2]$
Si il existe m tel que *mec_pipe (i, n, m)* = mise en correspondance entre les 2 conduites $Gr^{old}[k_1][k'_1]$ et $Gr^{new}[k_2][k'_2]$ Alors {
mec_tank (i, n, K'+1) = mise en correspondance entre les deux réservoirs $Tank^{old}[k'_1]$ et $Tank^{new}[k'_2]$;
mec(amont, k'_1, k'_2) ;
K'++ ;}
$K = K'$;

Limitations de l'algorithme

On voit donc apparaître une faiblesse de cet algorithme : il n'est pas exhaustif quant à la recherche des différentes mises en correspondance possibles. En effet, lors des deux premières phases, on n'effectue pas toutes les adéquations possibles entre les réservoirs en alarme, mais uniquement entre les deux réservoirs situés les plus hauts, puis éventuellement entre deux autres réservoirs en alarme. Il est en fait exhaustif si la structure courante du problème à résoudre comporte un ou deux réservoirs en alarme situés sur des couches différentes. S'il y en a plus de deux, il ne prend donc pas en compte la combinatoire. Ceci n'a pas d'influence actuellement à la vue des différents cas et index enregistrés en mémoire (comportant deux réservoirs en alarme au maximum) qui peuvent être traités par cet algorithme. Il est à noter d'une part que la situation avec au moins trois réservoirs en alarme sera tout de même traitée par l'algorithme (les index n° 2 à 7 s'appliquent) mais celui ci ne testera pas toutes les possibilités de mise en adéquation entre les réservoirs en alarme, d'autre part que dans les problèmes traités qui comportent en général trois ou quatre réservoirs intermédiaires, il est très rare d'avoir plus de deux réservoirs en alarme. L'exhaustivité consisterait ici à considérer les couches de réservoirs (figure 30).

Cependant l'algorithme est exhaustif concernant les mises en correspondance au niveau des conduites et au niveau d'éventuels réservoirs non en alarme. Il est à noter tout de même concernant ce dernier point que l'algorithme ne prend en compte que les réservoirs non en alarme et situés directement en amont ou en aval d'un réservoir en alarme, donc il ne remonte pas à des niveaux supérieurs.

Ainsi, un travail futur serait de rendre cet algorithme exhaustif.

5.3.1.2. Sélection du meilleur cas

Ensuite, pour chaque index enregistré (c'est-à-dire dont on a trouvé une adéquation possible avec la situation courante et qui respecte les connaissances négatives), on teste la (ou les) mise en correspondance associée, au niveau de chaque cas stocké sous l'index, en calculant un coût détaillé cette fois-ci, puisqu'un cas correspond à la même structure de base que son index, mais avec tous les détails concernant les diamètres, hauteurs d'eau, positions de vanne, … On ne retient finalement que le cas de coût minimum, qui sera appliqué.

Le calcul du coût associé à une mise en correspondance se fait grâce à des pénalisations numériques et des connaissances négatives (pénalisations infinies) relatives à chaque différence observée : en effet, d'une part la correspondance peut ne pas être parfaite, et d'autre part il se peut que la structure ($Tank^{old}$, Gr^{old}) puisse être retrouvée de différentes façons au sein de la configuration présente. On ajoute un certain nombre de points dès qu'une différence est trouvée, en fonction de cette différence.

Il est donc important de noter que l'on peut associer à chaque nœud et à chaque arc de la structure du cas ou de l'index en mémoire ($Tank^{old}$, Gr^{old}) des attributs représentant des connaissances négatives, par exemple on peut avoir un cas représentant un réservoir en alarme tel que les conduites en aval ne doivent pas être fermées. Ainsi, une connaissance négative non respectée entraîne un rejet.

De manière générale, pour chaque paramètre de la structure ou de l'état, trois situations se présentent lorsqu'une différence est observée : différence possible (pénalisation faible), ou tolérée (pénalisation forte) ou interdite (connaissance négative, donc rejet).

Pour certaines différences, on ajoute un nombre de points importants : par exemple, la condition d'alarme sur les réservoirs est très importante, de même que les conduites reliant deux réservoirs, et il serait même préférable de rejeter un cas s'il y a ce type de différence. Dans certains situations, la condition sur l'existence d'une vanne ou sur la position d'une vanne est également très importante : nous pouvons soit pénaliser fortement cette différence, soit même rejeter le cas grâce à une connaissance négative associée à l'arc correspondant.
Concernant les valeurs des diamètres des conduites (ou des capacités des réservoirs), on n'ajoute qu'une faible pénalisation s'il y a plusieurs différences mais s'il existe une bijection (voir 5.4.) entre les deux ensembles de valeurs relatif à ce paramètre. Le fait que le nombre de conduites soit différent entraîne là aussi globalement une faible pénalisation, sauf si une conduite supplémentaire est sans vanne. En fait dans cette dernière situation, la pénalisation est fonction du cas et de l'emplacement de cette conduite sans vanne au sein de la structure. Enfin le fait que le nombre de réservoirs soit différent ne doit pas toujours être fortement pénalisé puisque le système peut dans certaines situations s'adapter à une variation de ce paramètre.

Il est évident que l'on doit toujours associer une connaissance négative à la position de la vanne actionnée. Ceci ne doit pas être oublié lors de l'apprentissage d'un cas nouveau. En outre, cette conduite comportant la vanne actionnée doit être repérable explicitement (par un attribut par exemple), surtout s'il s'agit d'une action instanciée, car si on ne parvient pas à établir une correspondance entre cette conduite d'un cas ancien et une conduite du cas nouveau, alors il y a rejet du cas ancien.

Le paramètre $Gr^{old}[i][j].\exists\_conduite$ est implicitement une connaissance nécessaire. Le non respect de cette valeur entraîne un rejet du cas/index correspondant. On peut bien sûr rendre ce paramètre facultatif en indiquant la valeur $0 \vee 1$.

La procédure de mise en correspondance attribue à chaque type de différence un nombre de points par défaut, sachant également que des attributs spécifiques peuvent être affectés à chaque cas indiquant des pénalisations différentes qui masquent alors les valeurs par défaut. En effet, pour certains cas, les niveaux d'eau des réservoirs et les diamètres de certaines conduites peuvent être primordiaux pour juger de leur application.

Voici un jeu de valeurs de **pénalisations par défaut** pour chaque type de différence :

- nombre de réservoirs : 8
- nombre de réservoir en alarme : 100
- conduite sans vanne : 5
- tendance d'un réservoir : 0.1
-capacité d'un réservoir : 0.1 (sauf si bijection : voir point suivant)
- bijection entre les valeurs de capacité des deux cas : 0.1
- hauteur d'un réservoir : 0.2
- diamètre d'une conduite : 0.3 (sauf si bijection : voir point suivant)
- bijection entre les valeurs de diamètre des deux cas : 0.1
- existence d'une vanne : 7
- position d'une vanne : 4
- conduite_sup_amont_tank : 0.1
- conduite_sup_aval_tank : 0.1
- conduite_inf_tank : 6

Remarque 1 : les différences du type diamètre d'une conduite, bijection entre les valeurs de diamètre des deux cas, existence d'une vanne et position d'une vanne ne concernent pas les éventuelles conduites supplémentaires (qui sont traitées avec les pénalisations *conduite_sup_amont_tank* et *conduite_sup_aval_tank*), mais les conduites sur lesquelles se fait explicitement la mise en correspondance entre les deux structures.

Remarque 2 : les deux paramètres *conduite_sup_amont_tank* et *conduite_sup_aval_tank* réfèrent à des conduites supplémentaires (en amont ou en aval d'un réservoir) présentes au sein de la structure du problème à résoudre, et donc absentes de la structure de l'index.
Ainsi, un nombre de conduites différent en amont (ou en aval) de tel réservoir serait considéré comme une seule différence, même s'il y a plusieurs conduites supplémentaires.

Remarque 3 : le paramètre *conduite_inf_tank* réfère à la situation où il y a davantage de conduites au niveau de l'index qu'au niveau de la structure du problème à résoudre, donc la situation inverse de celle représentée pour les deux paramètres précédents (*conduite_sup_amont_tank* et *conduite_sup_aval_tank*). Il faut alors que les caractéristiques des conduites du cas nouveau soient en adéquation avec celles de *certaines* conduites du cas ancien (caractéristiques positives et négatives), et que les connaissances négatives associées aux conduites non mises en adéquation du cas ancien soient respectées. De plus, si certaines conduites du cas nouveau ne sont pas en adéquation avec celles du cas ancien, il faut que leurs caractéristiques soient en accord avec les paramètres *CN_conduite_sup_amont* et *CN_conduite_sup_aval* du cas ancien.

Ainsi, dans cette situation, certaines conduites de l'index ne sont pas mises en adéquation. Cela n'est pas forcément un problème, mais si c'est le cas, on peut l'indiquer grâce à une valeur de pénalisation spécifique importante du paramètre *conduite_inf_tank* au sein d'un index et pour le réservoir concerné, ou bien au sein du paramètre $Gr^{old}[i][j].\exists\_conduite$ de l'index pour la conduite $i \rightarrow j$ concernée (la valeur 1 pour ce paramètre indique aussi comme cela a été dit ci-dessus que la valeur 0 est interdite).

5.3.2. Classification des graphes

Nous faisons ici une proposition d'amélioration de la phase de recherche de cas similaires, mais qui n'a pour l'instant pas été implémentée. Il s'agit de classifier le graphe représentant la structure du problème à résoudre au sein d'une hiérarchie de graphes représentant la mémoire du système. R.A. Levinson (dans [Haton 91]) propose une méthode de classification dans une hiérarchie de graphes non orientés dont nous nous sommes inspiré, car il faut tenir compte ici du fait que les graphes sont étiquetés. La classification pourrait être utile lorsque le nombre d'index au sein d'un même but de haut niveau est important, ou lorsque le nombre de cas au sein d'un même index est important.

A un premier niveau, elle peut concerner des index au sein d'un même but de haut-niveau : il s'agit alors d'un moyen de faciliter la recherche de graphes correspondant à des index. Cette classification peut alors être vue comme une recherche grossière d'un ensemble d'index similaires en mémoire, auxquels on appliquerait ensuite la procédure de mise en correspondance afin de juger plus finement la similarité avec le problème à résoudre.

A un second niveau, elle peut concerner des cas au sein d'un index : il s'agit alors d'un moyen de faciliter la recherche de graphes correspondant à des cas. Cette classification peut alors être vue comme une recherche grossière d'un ensemble de cas similaires en mémoire, auxquels on appliquerait ensuite la procédure de calcul de coût. Cependant la procédure de classification est la même dans les deux situations.

Nous définissons donc quatre critères de similarité :

- au niveau de l'index : classification et procédure de mise en correspondance,

- au niveau du cas : classification et procédure de calcul de coût.

La méthode fonctionne de la manière suivante : il s'agit dans un premier temps de relier, au sein d'un même but, chaque index à ses prédécesseurs et successeurs immédiats par des pointeurs, et au sein d'un index, de relier chaque cas à ses prédécesseurs et successeurs immédiats par des pointeurs. Un graphe prédécesseur est un sous-graphe (la relation "est un sous-graphe de" est une relation d'ordre partiel). Les graphes situés les plus bas dans la hiérarchie sont donc les plus généraux, et le (ou les) graphe le plus haut est le plus spécifique.

L'algorithme de R.A. Levinson [Haton 91] est le suivant :

Soit un graphe *G* à classer.

1ère étape : recherche des prédécesseurs (sous-graphes) immédiats de *G* = ensemble *IP (G)*.
On effectue un parcours en largeur à partir des feuilles (graphes les plus généraux), et on remonte. Le graphe courant examiné est noté *g*.

IP(G) = nul
Tous les graphes à examiner sont non marqués
Tant que il existe un graphe g non marqué tel que chaque élément de IP (g) est marqué vrai ou IP(g) = nul
Si g est un prédécesseur de G
Alors :
    g est marqué vrai
    IP(G) = (IP(G) – IP(g)) ∪ g
Sinon
g est marqué faux

2ème étape : recherche des successeurs immédiats de G = ensemble IS (G)

IS(G) = nil
P = IP(G)
Tant que (P ≠ nul)
Faire :
    Chaque successeur s d'un élément p de P est marqué m(p)
    Les successeurs comportant les marques de tous les éléments de P sont des successeurs potentiels de G.
    On effectue un test (G est-il un prédécesseur de ces éléments ?) pour le vérifier, et on ajoute ces éléments à l'ensemble IS(G).
    En outre on marque les successeurs de ces éléments comme faux.

P = (successeur(P) – IS(G)) et dont les successeurs ne sont pas marqués faux

Si on a *IP(G) = IS(G)*, alors *G* est l'unique élément de ces deux ensembles et *G* fait partie de la base de cas, sinon on a encadré *G* par ses prédécesseurs immédiats et ses successeurs immédiats. Si on doit, lors de l'apprentissage, intégrer le cas (ou l'index) dans la hiérarchie, il le sera donc entre ces deux ensembles.

En outre, la recherche des ensembles *IP(G)* et *IS(G)* peut constituer un moyen de rechercher des cas (ou index) similaires à *G*.

Il faut également implémenter une procédure *test_prédécesseur (g, G)* qui teste si le graphe *g* est un prédécesseur du graphe *G*, c'est-à-dire si *g* est contenu dans *G*. Elle pourrait être basée sur la procédure de mise en correspondance ($Tank^{old}$ et Gr^{old} seraient associées à *g*, $Tank^{new}$ et Gr^{new} seraient associées à *G*. Une adaptation de cette procédure permettrait de juger par un score numérique la qualité de la relation "sous-graphe de" entre *g* et *G*, et pourrait d'ailleurs être conservée en mémoire (étiquette au niveau du lien entre les deux graphes), mais dans ce cas il ne faudrait pas oublier de mettre à jour cette valeur si l'un des deux graphes venait à être modifié dans le futur (apprentissage, généralisation, ...).

5.3.3. Conclusion

La classification et la mise en correspondance de deux graphes sont utiles *pour la phase de recherche* de cas similaires. En outre, la mise en correspondance est utile *pour l'adaptation* du cas nouveau au cas en mémoire jugé le plus similaire. En effet, même s'il y a des différences, la correspondance au niveau des réservoirs et des conduites reliant ces réservoirs permet ensuite de déterminer la correspondance entre la conduite comportant la vanne actionnée au sein du cas en mémoire et une conduite du cas nouveau, ce qui permet d'effectuer une action. Ceci est surtout utile si l'action est instanciée, c'est-à-dire localisée explicitement au sein de la structure du cas (voir 5.5.3.). Par opposition, une action procédurale (voir 5.5.3.) permet une adaptation nettement plus poussée, ce qui est très utile si la détermination de l'action nécessite des calculs logiques et numériques au sein de la structure du réseau.

Enfin, suite à cette phase de recherche, il s'agit de ré-utiliser le cas considéré comme le plus similaire à l'état actuel du problème à résoudre. La phase de ré-utilisation consiste à instancier/exécuter l'action du cas sélectionné. Mais s'il y a des différences entre les deux cas, une adaptation est nécessaire.

5.4. ADAPTATION

Dans notre étude, il s'agit de savoir s'adapter lorsque l'on ne trouve pas un index identique (c'est-à-dire le graphe abstrait de l'index est différent de celui du cas à résoudre), et également à s'adapter lorsqu'on a trouvé un index identique, mais on n'a pas trouvé de cas identique. L'adaptation est largement facilitée si l'action à réaliser au sein du cas est paramétrée par la structure du réseau (voir 5.5.3.). En effet, une façon de généraliser un cas afin que l'action associée soit applicable et adaptable dans de nombreuses situations est que le calcul d'action comporte des paramètres relatifs à la configuration du réseau et à l'état du système. Il s'agit donc là d'une action procédurale. L'adaptation peut concerner :

● le nombre de réservoirs présents dans le graphe de l'index. Par exemple, dans certaines situations, si un cas/index est relatif à deux alarmes simultanées, alors le système peut peut-être gérer un problème similaire mais avec trois alarmes simultanées.
L'adaptation peut être tentée si le nombre de réservoir dans la situation courante appartient à l'ensemble représentant l'ensemble des valeurs possibles de ce paramètre au niveau de l'index.
Une autre situation dans laquelle le nombre de réservoirs diffère est liée à l'existence d'une conduite sans vanne. Comme cela a été dit plus haut (5.1.4), on intègre au sein de la structure du cas les réservoirs reliés à cette conduite sans vanne. L'adaptation peut cependant échouer dans cette situation car la présence d'une conduite sans vanne peut nécessiter une stratégie spécifique.
Par exemple le schéma ci-dessous montre une situation où le cas à résoudre ne peut être résolu à l'aide du cas en mémoire. On peut empêcher ce type d'adaptation en associant une connaissance négative soit à une conduite avec vanne indiquant qu'elle ne doit pas être sans vanne, soit au niveau du nombre de réservoirs du cas, soit au niveau des conduites supplémentaires d'un réservoir.

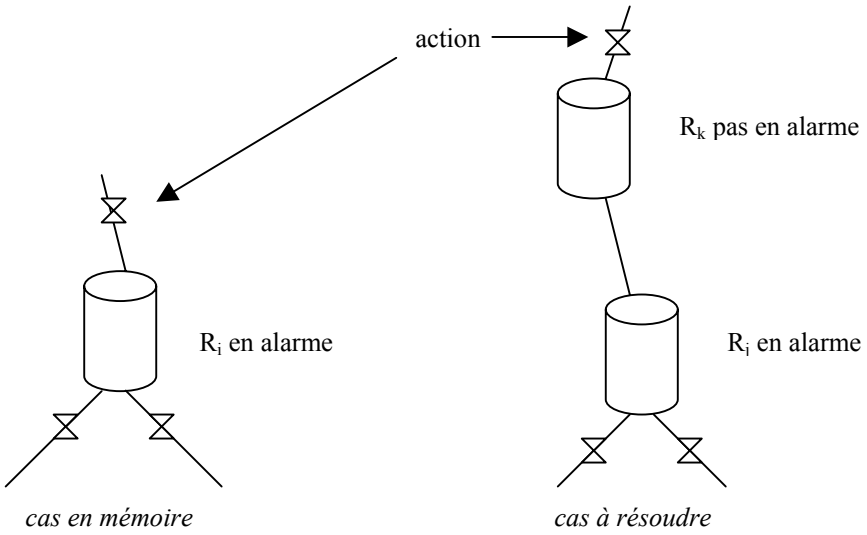


Figure 34 : Adaptation sur le nombre de réservoirs

L'adaptation sur le nombre de réservoirs a été mis en œuvre au sein des index n° 1, 7 et 9 de la base de connaissances.

● l'ensemble des capacités des réservoirs et les diamètres des conduites. Il est possible de s'adapter en définissant une ***bijection*** entre l'ensemble des valeurs des capacités (ou bien des diamètres) du cas/index en mémoire et l'ensemble des valeurs du même paramètre du cas à résoudre. Ces deux paramètres sont modélisés de façon qualitative par un domaine de valeurs possibles D = {P, M, G} (P = Petit, M = Moyen, G = Grand). Par exemple, si un cas fait référence à l'un de ces deux paramètres avec un domaine de valeurs {P, M}, et le cas nouveau fait référence au même paramètre mais avec le domaine de valeurs {M, G}, alors le système peut tenter d'appliquer la substitution entre les deux domaines : P ↔ M et M ↔ G.

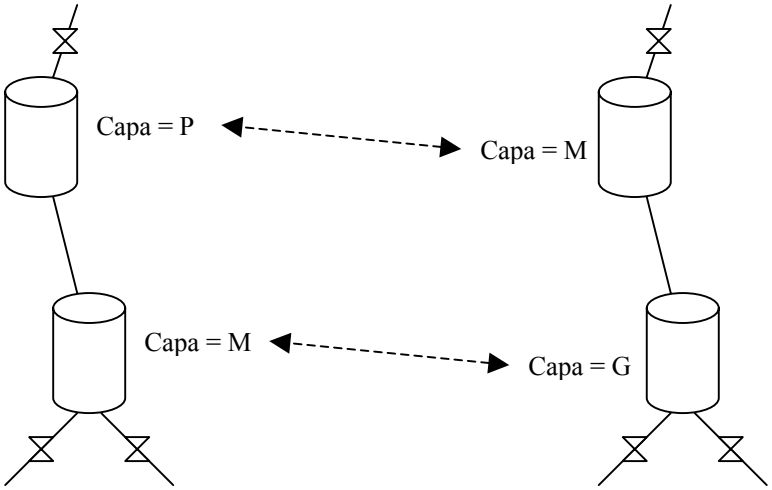


Figure 35 : Adaptation sur les valeurs des capacités des réservoirs

● les divers paramètres de la structure et de l'état du réseau : position d'une vanne, capacité d'un réservoir, diamètre d'une conduite, existence ou non d'une vanne (sans introduire de réservoir supplémentaire), tendance qualitative et hauteur d'eau d'un réservoir. Ces paramètres peuvent être adaptés de façon isolée.

● l'existence de conduites supplémentaires sur l'une des deux structures à comparer.

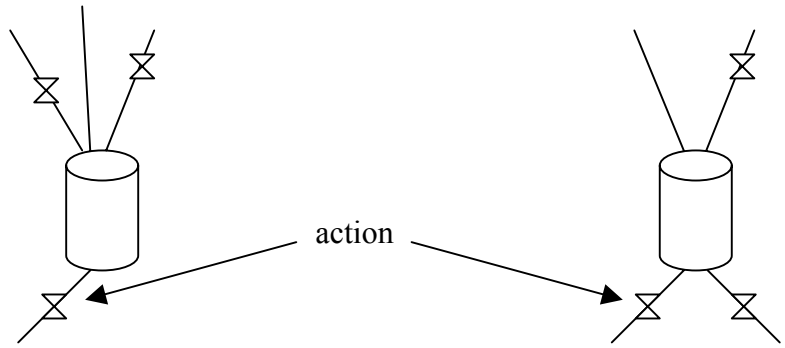


Figure 36 : Adaptation par rapport à des conduites supplémentaires au niveau d'un réservoir

Comme nous l'avons vu précédemment, l'adaptation est possible dans les deux sens, c'est-à-dire dans la situation où le cas à résoudre comporte au niveau d'un réservoir davantage de conduites que le cas en mémoire, et dans la situation inverse où le cas en mémoire comporte au niveau d'un réservoir davantage de conduites que le cas à résoudre

Conclusion

L'adaptation consiste donc, malgré des différences observées entre le cas à résoudre et le cas retrouvé en mémoire, à appliquer l'action indiquée par ce cas retrouvé, qu'elle soit procédurale (paramétrée par la structure courante du réseau) ou bien instanciée (voir 5.5.4).

Mais bien entendu, dans toutes ces situations, l'adaptation sur un paramètre entraîne une pénalisation au niveau de la mise en correspondance, voire un rejet. Dans ce dernier cas, lorsque l'adaptation ne doit pas être tentée, on l'indique au sein du cas/index grâce à une connaissance négative, fournie soit par l'utilisateur/professeur, soit par le système lui-même (voir apprentissage par l'échec). Les capacités d'adaptation sont donc étroitement liées aux valeurs des pénalisations (plus une différence entraîne une pénalisation forte, ce qui reflète une adaptation délicate, moins le cas en mémoire a de chances d'être sélectionné) et aux connaissances négatives (qui signifient l'impossibilité d'adaptation). L'adaptation est donc liée à la procédure de mise en correspondance entre les deux graphes étiquetés. Ainsi, la création de connaissances négatives et la possibilité d'avoir des valeurs de pénalisations spécifiques à chaque cas/index assurent une certaine finesse au niveau des possibilités d'adaptation.

Comme nous allons le voir, si elle réussit, et selon l'importance des différences entre les deux cas, on peut soit enregistrer le cas nouveau, soit l'éliminer (après avoir éventuellement préalablement généralisé, c'est-à-dire enregistré des informations relatives au cas nouveau au sein du cas ancien). Si elle ne réussit pas (selon le jugement d'un professeur expert), alors ce professeur indique ce que le système aurait dû faire (voir apprentissage 5.5.).

5.5. SPECIFICATIONS D'UNE METHODE D'APPRENTISSAGE

En premier lieu, précisons qu'il s'agit ici d'une proposition concernant la phase d'apprentissage, et que l'implémentation n'a pas été réalisée, mais pourrait faire l'objet de travaux futurs.

Après la résolution d'un problème, le système analyse tous les nouveaux cas créés. Cette décomposition en cas aura été réalisée lors de la résolution : il s'agit des configurations partielles du réseau qui à chaque pas de temps ont été mises en correspondance avec un cas choisi dans la mémoire.
En outre, un "professeur" (un expert humain) intervient en cas d'échec pour corriger le système et lui indiquer l'action qu'il aurait dû effectuer à un instant donné. L'utilisation du contrôle optimal pourrait être très utile à ce niveau au "professeur" car cela permettrait de savoir si la résolution opérée par l'opérateur artificiel est de bonne qualité ou non, d'extraire le cas échéant des stratégies optimales, et de savoir à quel(s) instant(s) le système à base de cas a effectué une mauvaise action (problème d'attribution du mérite cité précédemment).

5.5.1. Généralisation de deux cas ou index au niveau de la structure et de l'état du réseau

Elle est utilisée lorsque le cas ancien a été appliqué avec succès pour résoudre le cas nouveau. Il s'agit donc ici d'une généralisation au niveau des ***conditions d'application*** d'un cas ou d'un index, concernant les attributs des objets "conduite" ($Gr^{old}[i][j]$) et "réservoir" ($Tank^{old}[k]$).

Une procédure simple (notée *Po*), mais qui ne s'applique que dans la situation où les deux cas ne possèdent qu'un seul paramètre (un attribut) dont la valeur diffère, tous les autres ayant des valeurs identiques (notamment l'action est la même), consiste à modifier l'attribut en introduisant une disjonction entre les valeurs possibles. Si l'attribut contient la disjonction de l'ensemble des valeurs du domaine, alors il n'intervient plus :

Si $[(x_k = a_k)_{k \in K,\, k \neq i}\ ;\ (x_i = b)] \in C \wedge [(x_k = a_k)_{k \in K,\, k \neq i}\ ;\ (x_i = c)] \in C$
Alors $[(x_k = a_k)_{k \in K,\, k \neq i}\ ;\ (x_i = b \vee c)] \in C$

où C est un cas, x_i un paramètre avec Domaine $(x_i) = \{b, c, \ldots\}$, les x_k pour $k \in K$, $k \neq i$ représentent tous les autres paramètres, de valeur constante ici.

A la limite, lorsque $[(x_k = a_k)_{k \in K,\, k \neq i}\ ;\ (x_i = Domaine\ (x_i))] \in C$, alors x_i n'intervient plus dans le description du cas.

Rappelons que les paramètres d'un cas sont la capacité des réservoirs (D_R = {Petit, Moyen, Grand}), les diamètres des conduites (D_Φ = {Petit, Moyen, Grand}), l'existence d'une vanne sur chacune de ces conduites, la position de cette vanne, le test d'alarme, la tendance qualitative et la hauteur d'eau de chaque réservoir, ainsi que l'action permettant de résoudre un but de haut niveau. La généralisation peut porter sur chacun de ces paramètres, sauf sur le test d'alarme, car il représente le paramètre essentiel permettant de distinguer les situations, ni sur l'action, car d'une part nous avons supposé que cet apprentissage s'applique lorsque le cas ancien a été appliqué avec succès (donc l'action est bonne), et d'autre part la généralisation d'une action instanciée nécessite un processus spécifique (voir remarque 3 ci-dessous).

La généralisation peut aussi porter sur les connaissances négatives *CN_ conduite_sup_amont* et *CN_conduite_sup_aval* (paramètres qui concernent des conduites supplémentaires), et éventuellement sur d'autres connaissances négatives, mais il s'agit là d'un apprentissage à partir d'échec (voir 5.5.5.).

Concernant la généralisation sur l'existence d'une vanne sur une conduite, si c'est le cas ancien (en mémoire) qui comporte une conduite sans vanne (et le cas nouveau comporte une

vanne au niveau de la conduite correspondante), alors la généralisation est possible, et elle se répercute sur l'index de la même manière. Cependant, si c'est le cas nouveau qui comporte une conduite sans vanne (et le cas ancien comporte une vanne au niveau de la conduite correspondante), mieux vaut intégrer le cas nouveau comportant un réservoir supplémentaire relié à cette conduite sans vanne, même si l'adaptation a réussi, et créer un nouvel index le contenant (voir 5.5.2.). En effet, comme cela a été dit précédemment, la présence d'une conduite sans vanne représente une situation spécifique à prendre en compte explicitement.

Il est à noter enfin que le cas nouveau est généralisé non seulement avec le cas ancien (choisi), mais également avec tous les cas anciens situés sous le même index. Par exemple, supposons que le cas nouveau et le cas ancien choisi diffèrent par une seule et même conduite de caractéristiques *{Φ = P ; position_vanne = 0}* pour le cas nouveau et *{Φ = M ; position_vanne = 0 ∨ 1}* pour le cas ancien. Si un autre cas, présent dans l'index du cas choisi, ne diffère lui aussi que par cette même conduite avec les caractéristiques *{Φ = P ; position_vanne = 1}*, alors la généralisation provoque la modification suivante : les trois cas de l'index sont fusionnés en un seul avec l'attribut *{Φ = P ∨ M}* pour la conduite en question. Ainsi, même si une généralisation n'est pas possible à un instant donné, elle le sera peut être dans le futur.

Si les deux cas possèdent plusieurs paramètres de valeurs différentes et donc la généralisation à l'aide de *Po* n'a pas été possible, il semble préférable d'enregistrer le nouveau cas. Deux situations se présentent alors :

● parmi les paramètres dont les valeurs diffèrent, un seul est un paramètre de l'index, alors on généralise l'index (disjonction de valeurs au niveau de ce paramètre), et on enregistre les deux cas sous cet index généralisé. Il est à noter qu'il n'y a qu'un seul paramètre de l'index qui peut être soumis à une généralisation par disjonction de valeurs : l'existence d'une vanne sur une conduite.

● aucun paramètre dont les valeurs diffèrent entre les deux cas n'est un paramètre de l'index, alors on n'a pas à créer d'index et on ne fait qu'enregistrer le nouveau cas sous l'index existant.

Remarque 1 : en fait, on pourrait de manière alternative fusionner deux cas même si plusieurs paramètres ont des valeurs différentes, afin de limiter le nombre de cas en mémoire, en modélisant les valeurs possibles d'un paramètre comme un ensemble d'ensembles (voir 5.2.2). Par exemple, si deux cas sont identiques sauf deux paramètres : un cas possède les paramètres *ϕ[i][j] = a* et *état_vanne[k][l] = b*, et l'autre cas possède les paramètres *ϕ[i][j] = a'* et *position_vanne[k][l] = b'*, alors on ne modélise qu'un seul cas de paramètres *ϕ[i][j] = [[a] ; [a']]* et *état_vanne[k][l] = [[b] ; [b']]*. Comme cela a été dit, il faut alors veiller à la correspondance entre ces ensembles de valeurs. Il est à noter qu'il ne s'agit pas là d'une généralisation, mais juste d'un moyen d'économiser de la place mémoire.

Remarque 2 : le problème de cette procédure simple de généralisation *Po* est qu'il faut beaucoup d'exemples avant de pouvoir généraliser. On pourrait donc mettre en place une procédure d'induction plus puissante mais plus lourde du type arbre de décision (voir annexe 2), avec comme vecteur d'entrée les paramètres (capacité, diamètre, hauteur, tendance, existence d'une vanne, position de la vanne) pour chaque réservoir et conduites. Ce type de procédures permet une généralisation plus poussée, donc nécessitant moins d'exemples pour exhiber un concept, en se basant par exemple sur des probabilités.

Exemple	Paramètre : x_1	Paramètre : x_2	Paramètre : x_3	Paramètre : x_4	cas
E1	a	b	c1	d1	C
E2	a	b	c2	d2	C

Tableau 11 : Induction à partir d'exemples

Une procédure d'induction puissante pourrait induire, d'après ces exemples :

$(x_1 = a \text{ et } x_2 = b) \in C$

Une façon d'améliorer cette procédure simple *Po* est d'ajouter un autre type de généralisation qui porte sur l'ensemble des valeurs des capacités des réservoirs ou des diamètres des conduites. Il s'agit ici donc d'une généralisation sur l'ensemble des valeurs et non sur une valeur, conditionnée par l'existence d'une bijection entre les deux ensembles. Par exemple, un cas peut être généralisé si les valeurs des diamètres des conduites sont dans {P, M} dans un cas, et dans {M, G}dans l'autre cas, et s'il existe une bijection :

$P \rightarrow M$

$M \rightarrow G$

Remarque 3 : concernant la généralisation au niveau de l'action, si l'action au niveau du cas ancien est une procédure paramétrée, alors elle est déjà généralisée. Si il s'agit d'une action instanciée, alors elle peut être généralisée dans des cas simples grâce à une procédure spécifique (voir 5.5.4.).

Remarque 4 : si le cas nouveau et le cas ancien utilisé ne possèdent pas le même index, alors il est possible de généraliser les deux index (ou tout au moins de regrouper les deux index) mais cela doit être explicitement indiqué par le professeur. Dans le cas contraire, il y a création d'un index intégrant uniquement le cas nouveau (voir 5.5.2 situation 3).

5.5.2. Création d'un nouvel index

L'index créé doit être suffisamment général (pour qu'il puisse être appliqué dans le futur) et suffisamment spécifique (pour ne pas englober trop de situations).
Cet index créé est une abstraction du cas correspondant : tous deux comportent des nœuds représentant les réservoirs et des arcs représentant les conduites. On représente l'état abstrait en ne tenant compte que des états d'alarme ou non des réservoirs, et la structure abstraite du réseau en ne tenant compte que de l'existence ou non d'une vanne sur les conduites.
Le but associé à l'index créé est soit indiqué par le professeur, soit trouvé par le système en fonction de son état. Il s'agit ensuite de placer ce nouvel index (et le cas associé) à la bonne place dans la hiérarchie, c'est-à-dire sous le but, voire sous un index existant si ce nouvel index est plus spécifique qu'un index existant (voire classification des cas et index).

Il y a création d'un index dans trois situations. Cela se produit si, lors de la résolution :

1- on n'a pas réussi à trouver un index (tous ont été rejetés),

2- on a trouvé un index qui a été jugé similaire, mais à tort (il y a eu échec de résolution),

3- on a trouvé un index "peu similaire", mais qui a permis de trouver un cas en mémoire et une action associée jugée satisfaisante par le professeur (voir remarque 1 ci-dessous). Le terme "peu similaire" signifie que des différences "importantes" ont été trouvées entre le cas

nouveau et le cas ancien. Le terme “différence importante” réfère à une différence correspondant à un attribut représenté au niveau de l’abstraction, c’est-à-dire de l’index. Or, les cas regroupés sous un même index possèdent par hypothèse tous la même abstraction. Ainsi, si le cas nouveau a été résolu grâce à un cas ancien qui ne possède pas la même abstraction, alors les deux cas possèdent des index différents, et il est donc préférable de créer un index associé au cas nouveau, sauf indication contraire de la part du professeur (voir 5.5.1 remarque 4). Une seconde raison est que même si l’adaptation a réussi, des situations futures pourraient rendre nécessaire la distinction explicite de ces deux cas.

Dans chacune de ces situations, l’index créé ne regroupe que le cas nouveau.

Si l’ancien cas et le nouveau cas possèdent la même abstraction, alors ils sont bien sur regroupés sous le même index déjà existant, et il n’y a pas de création d’index. On classe ce nouveau cas au sein de la hiérarchie existante sous l’index, mais ici cette opération est facilitée par le fait que l’on a observé une similitude avec un cas ancien. Donc le cas nouveau sera probablement proche du cas ancien. En outre, on peut aller plus loin en regroupant sous cet index ces deux cas similaires bien sûr, mais aussi d’autres cas faisant déjà partie de l’index existant et qui sont similaires à ces deux cas, en se basant sur les scores de mise en correspondance avec le nouveau cas. Cette idée permet alors la création de groupements de cas au sein d’un index (donc des sous-index), ce qui peut être utile si la taille de la base devient importante.

Les différentes situations entraînant la création ou non d’un index sont également présentées dans la partie bilan de l’apprentissage ci-dessous (5.5.6.2.).

Remarque 1 : s’il y a eu adaptation sur l’existence d’une vanne avec variation du nombre de réservoirs (voir schéma ci-dessous), alors, que cette adaptation ait réussi ou non, il est préférable d’intégrer le réservoir relié à cette conduite sans vanne, et donc de créer un nouvel index associé au nouveau cas.

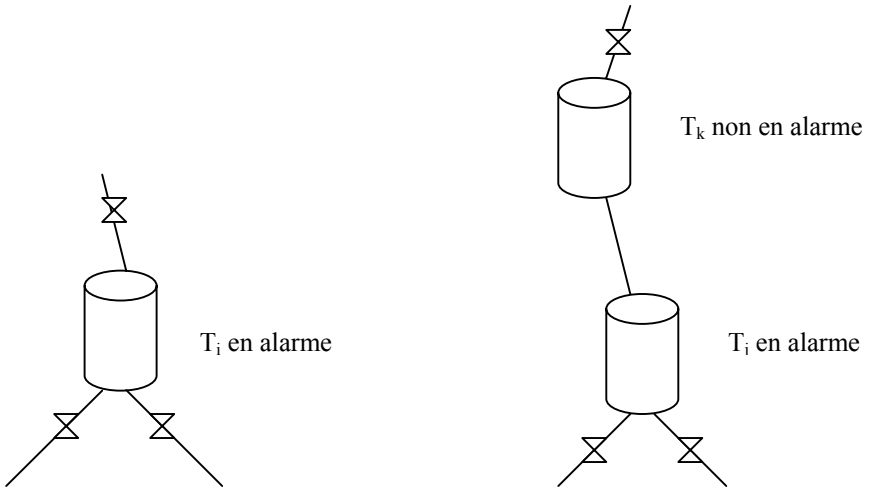


Figure 37 : Création d’un index - situation 1

Remarque 2 : si par contre il y a eu adaptation sur l’existence d’une vanne sans variation du nombre de réservoirs (voir schéma ci-dessous), alors, si cette adaptation a réussi, on

généralise les deux cas sans créer d'index, et si cette adaptation a échoué, on crée un index intégrant uniquement le cas nouveau, car il y a une différence importante entre ces deux cas au niveau de la structure (et en outre l'action peut être différente).

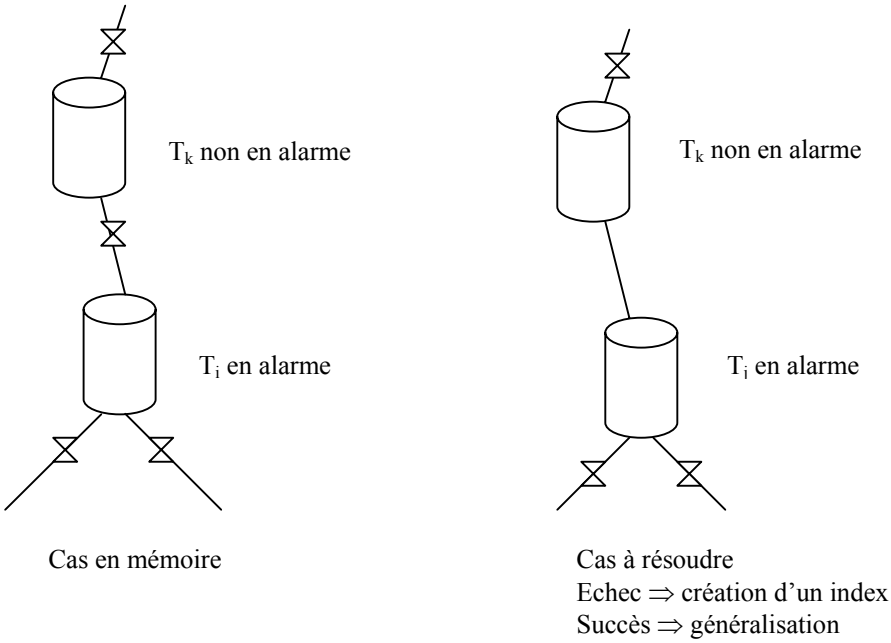


Figure 38 : Création d'un index - situation 2

Remarque 3 : de la même façon que l'on classe un cas nouveau intégré au sein d'un index comportant déjà plusieurs cas, il faut classer un nouvel index au sein d'un but de haut niveau comportant déjà plusieurs index.

5.5.3. Calcul de l'action correspondant à un cas : procédure ou instanciation explicite ?

Le principe de l'action procédurale est que si le calcul de l'action dépend des structures *Tank* et *Gr*, alors le système peut s'adapter lorsque la configuration liée au problème à résoudre est légèrement différente. Il s'agit donc d'une procédure qui prend en paramètre les deux structures $Tank^{new}$ et Gr^{new} (dont on aura préalablement établi la correspondance avec les structures $Tank^{old}$ et Gr^{old} du cas en question). L'intérêt est de pouvoir généraliser au niveau des conditions d'application du cas. En outre, un seul cas bien choisi peut suffire à représenter les conditions générales dans lesquelles la situation s'applique.

Une solution plus simple est d'instancier l'action en identifiant de manière explicite au sein du graphe du cas la vanne à actionner, au lieu de la calculer grâce à une procédure fonction de la structure du problème à résoudre. Ceci est possible de façon simple si :

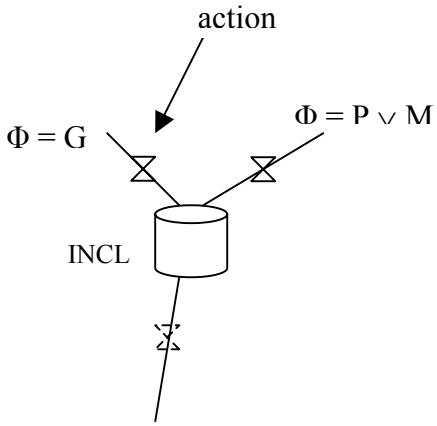
● le choix de l'action ne nécessite pas de calcul numérique ou logique complexe (comme par exemple pour l'index n° 7).

● les divers éléments du réseau (réservoirs et conduites) contiennent suffisamment d'informations (sous la forme d'attributs) pour qu'il n'y ait pas d'ambiguïté quant à la mise en correspondance entre la structure d'un cas en mémoire et celle du problème à résoudre, de façon à ce que l'action instanciée soit déterminée là aussi sans ambiguïté.

Mais dans les situations où une instanciation de l'action est possible, cela peut tout de même amoindrir les capacités d'adaptation.

Examinons sur des exemples les différences entre ces deux modes de représentations :

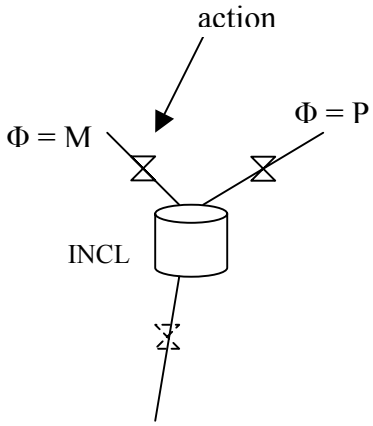
● Pour coder l'action correspondant à l'index n° 3, comportant un réservoir en alarme et une seule conduite en amont, la procédure consiste à ajouter des pénalités aux différentes mises en correspondance, décroissantes en fonction du diamètre de la conduite en amont du réservoir en alarme. Grâce à cette procédure, un seul cas est suffisant. Mais on pourrait aboutir au même niveau de généralité avec plusieurs cas représentant les différentes situations possibles. Par exemple, pour le cas n° 3-0, on pourrait de façon alternative enregistrer (ou faire apprendre au système avec l'expérience) les trois cas suivants (Φ représente le diamètre d'une conduite) :



: vanne ouverte

: vanne facultative.
Si elle existe alors elle est ouverte

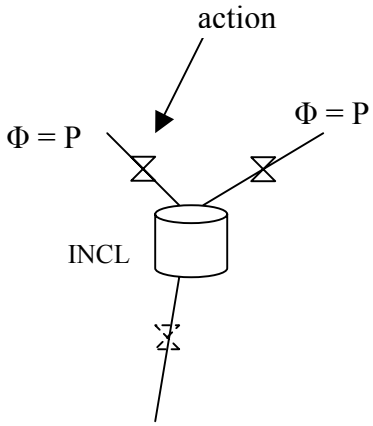
Cas 3-1



: vanne ouverte

: vanne facultative.
Si elle existe alors elle est ouverte

Cas 3-2



: vanne ouverte

: vanne facultative.
Si elle existe alors elle est ouverte

Cas 3-3

Figure 39 : Alternative à la procéduralisation du cas n°3

● Pour coder l'action correspondant à l'index n° 7, où il faut rechercher le réservoir qui a le plus gros rapport $\Sigma\Phi_e / \Sigma\Phi_s$ (Φ_e et Φ_s sont les diamètres entrant et sortant avec écoulement), une procédure simple peut être mise en œuvre :

```
Procédure Index7 (Gr^new, Tank^new)
{
trouver i et j dans [1 ; ... ; n'] tel que (Tank^new [i].état = 1 et Tank^new[j].état = 1)

// teste l'écoulement sur les conduites l→ k en amont et en aval de Tank^new [i] et Tank^new [j]
δ[l][k] = 0 si (Tank^new[l].H = 0 ou Gr^new[l][k].état_vanne = 0), = 1 sinon

pour k=i et k=j :
rapport_Φ[k] =  Σ l=1 to n' Gr^new[l][k].Φ × δ[l][k] / Σ l=1 to n' Gr^new[k][l].Φ × δ[k][l]

// sélection du réservoir i ou j (indice_tank_al)
si (rapport_Φ[i] > rapport_Φ[j])
        indice_tank_al  = i;
sinon
        indice_tank_al = j;

// recherche d'un cas s'appliquant à la situation où seul le réservoir choisi est en alarme
lancer_recherche_cas (indice_tank_al) ;
}
```

Pour cet index, sans l'aide d'une procédure, on voit bien qu'il faut alors enregistrer de nombreux cas représentant les différentes situations possibles.

Conclusion

Il est clair que la procéduralisation de l'action liée à un cas est d'une certaine façon une solution de facilité puisque d'une part le système peut s'adapter à une grande variété de situations avec très peu de cas en mémoire, et d'autre part la procédure est réalisée par l'utilisateur et non par le système. Mais on peut voir la procéduralisation comme une capacité d'abstraction du système.

Il serait intéressant de tester les deux manières de coder une action :

● Action procédurale. Très peu de cas seront nécessaires pour modéliser une grande variété de situations pouvant se présenter et qui nécessitent l'application de cette action.

● Action instanciée. Beaucoup de cas seront nécessaires pour avoir le même comportement. Ces cas pourront être fournis soit par un professeur/utilisateur, soit grâce à l'apprentissage.

Ceci permettrait de juger de manière plus approfondie les avantages et inconvénients de chacune.

Le système est alors basé sur deux comportements distincts et complémentaires :

● comportement prototypique, abstrait (un cas avec une action procédurale)

● comportement basé sur les instances (beaucoup de cas avec chacun une action instanciée)

5.5.4. Généralisation au niveau de l'action d'un cas

Tout d'abord, remarquons qu'il s'agit d'une généralisation sur le but associé à un cas, par opposition à la généralisation sur les conditions d'application d'un cas (voir 5.5.1.).
Nous avons vu que l'instanciation d'une action au sein d'un cas engendrait une perte de généralité au niveau du calcul d'action, et une perte au niveau de l'adaptabilité. On peut tout de même tenter de généraliser au niveau de cette action instanciée, ce qui permettrait alors de rejoindre le premier point de vue correspondant à l'action procédurale. L'idée est de représenter l'action de manière abstraite et si possible de façon unique en découvrant le concept causal sous-jacent au cas.
Nous sommes dans la situation suivante : un cas ayant été appris avec une action instanciée, si, dans le futur, un cas à résoudre assez similaire se présente mais est mal résolu à cause du manque d'adaptation (on se place ici dans la situation où c'est l'action du cas appris, ayant été mal généralisée, ou non généralisée, qui a provoqué l'erreur et donc c'est la généralisation au niveau de l'action qui permettra de résoudre le problème), alors le professeur fait apprendre au système un nouveau cas avec une nouvelle action instanciée.
Le système tente alors de généraliser ces deux actions. Pour cela, il commence par situer la conduite comportant la vanne actionnée par rapport à un des réservoirs (par exemple il s'agit d'une action sur une conduite en amont du réservoir en alarme). Ensuite il essaie de discriminer cette conduite par rapport à celles qui sont positionnées de la même façon par rapport au réservoir précédemment sélectionné (par exemple toutes les conduites en amont du réservoir en alarme), grâce aux caractéristiques des conduites (existence d'une vanne, diamètre de la conduite, position *ON/OFF* de la vanne). Il est à noter qu'il est préférable d'avoir plusieurs exemples avant de discriminer sur le diamètre de la conduite.

Le problème est le suivant :

Données : dans une configuration *c*, l'action associée est *a*, et dans une configuration *c'*, l'action est *a'*.
But : trouver une action *A* qui soit une abstraction de *a* et *a'* et telle que dans les configurations *c* et *c'*, l'action associée soit *A*.

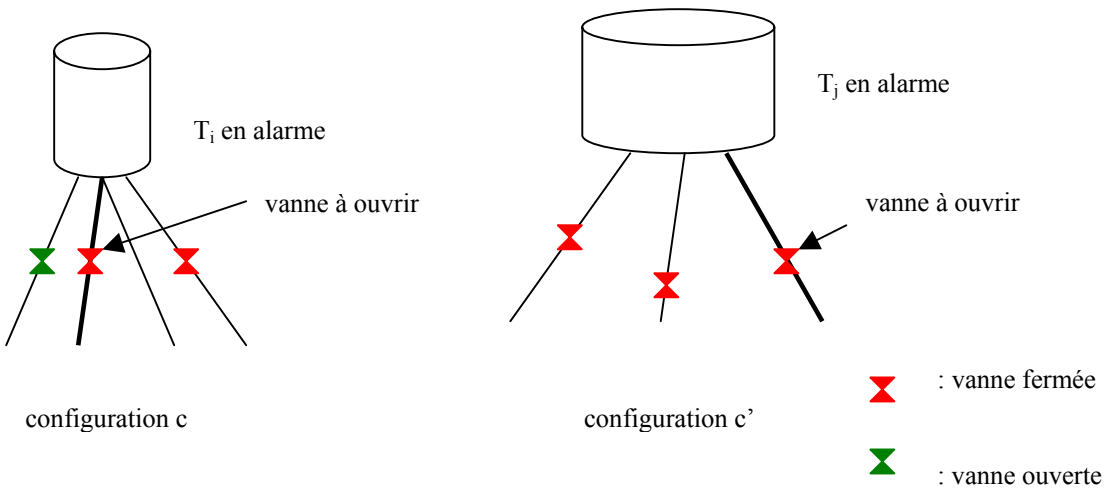


Figure 40 : Généralisation d'une action à partir d'exemples

D'après le schéma ci-dessus,

Configuration *c* :
abstraction de l'action *a* = ouvrir une vanne en aval de T_i avec T_i en alarme.

Configuration *c'* :
abstraction de l'action *a'* = ouvrir une vanne en aval de T_j avec T_j en alarme.

1ère abstraction commune : ouvrir *une* vanne en aval d'un réservoir en alarme.

Mais cela ne suffit pas car cette vanne n'est pas déterminée de façon unique. Donc il faut trouver pour chaque configuration ce qui distingue cette vanne fermée des autres vannes fermées. L'analyse des différences au niveau des caractéristiques des conduites montre que la caractéristique discriminante est le diamètre, et plus précisément qu'il s'agit de la vanne de plus gros diamètre.
On a alors l'abstraction *A* : ouvrir *la* vanne de plus gros diamètre en aval d'un réservoir en alarme. Cette abstraction peut être vue comme une procéduralisation, ce qui permet en quelque sorte d'unifier les deux modes de représentation d'une action.
Mais il est à noter que l'on ne pourra pas toujours déterminer de façon unique cette vanne. Ce n'est pas un problème car il se peut que l'action généralisée soit du type : ouvrir une vanne fermée en aval de tel réservoir, sans qu'il soit nécessaire d'ajouter plus de précision.

Après une généralisation réussie au niveau de l'action, le système tente de généraliser *au niveau de la structure* (voir 5.5.1.). En outre, concernant d'éventuelles différences au niveau du nombre de conduites en amont ou en aval de chaque réservoir, on ne conserve que la partie concernée comportant le moins de conduites, ou bien l'intersection, par souci de simplicité, tout en notant bien sûr certaines informations utiles au sein des attributs "conduites supplémentaires".

Comme on le voit, ce type de généralisation ne convient que pour des situations simples. Par exemple, si un cas fait référence à une action sur une conduite dont le diamètre a la valeur Φ = P, alors peut-on généraliser ce cas avec la valeur Φ = M voire Φ = G ? On voit bien que ce type de généralisation dépend de la situation et de conditions et nécessite beaucoup de connaissances de bon sens. Si l'on souhaite faire apprendre au système une généralisation à partir d'exemples pour les cas/index n° 7, 8 et 9, il faut mettre au point une méthode de classification à partir d'exemples fonctionnant à partir de connaissances, notamment en mécanique des fluides et en mathématiques, afin de découvrir les concepts causaux permettant de relier des conditions et des explications à l'action associée à un cas. Mais il faut un grand nombre de connaissances car dans ces situations complexes, la généralisation nécessite la mise en œuvre d'un raisonnement délicat. Le raisonnement qualitatif pourrait permettre d'intégrer et utiliser ces connaissances (voir 9. Conclusion).
La mise en place d'une procédure d'action paramétrée permet de manière simple de couvrir tous ces niveaux, mais bien évidemment on en revient au problème de l'apprentissage de cette procédure.
Une autre possibilité afin d'apprendre à gérer des situations complexes comme celles représentées par les cas n° 8 et 9 est, comme cela a été dit plus haut (voir 5.5.1 remarque 2), de stocker puis regrouper tous les cas similaires (structure et état) et dont l'action est identique (et localisée de façon unique), pour ensuite leur appliquer une procédure d'induction fonctionnant sans connaissances *a priori*, mais le système ne découvre pas le concept

physique lié au cas, et il s'agit d'une généralisation au niveau des conditions et pas de l'action.
Il semble en outre délicat de donner la capacité au système de déterminer lui-même quand il est opportun de généraliser une action et quand il ne l'est pas. En effet, il est difficile pour le système de juger si une erreur est due à un problème de manque de généralisation au niveau de l'action, ou à un problème de sur-généralisation au niveau des conditions, ou à un problème nouveau qui n'est pas représenté dans la mémoire de cas. Le professeur devra intervenir à ce niveau pour indiquer si le système généralise ou non l'action.

Si le système ne généralise pas au niveau de l'action, il enregistre le nouveau cas avec son action instanciée, et diverses situations se présentent (voir 5.5.6.2. Bilan de l'apprentissage – problème mal résolu).

Conclusion

Ainsi, concernant l'action à réaliser au sein d'un cas, soit le professeur intervient de manière poussée pour incorporer une procédure d'action, soit l'action reste instanciée. Dans cette dernière situation, il est nécessaire d'enregistrer un grand nombre de cas afin de couvrir l'ensemble des situations pouvant se produire. Si la situation est relativement simple, le système peut tenter une généralisation au niveau de l'action, mais si la situation est complexe (index n° 7, 8 et 9 par exemple), alors il semble préférable (en tout cas dans un premier temps) de ne pas réaliser de généralisation au niveau de l'action. Quoi qu'il en soit, que l'action soit procédurale ou instanciée, tout se passe comme indiqué dans les différents chapitres concernant l'adaptation, la généralisation au niveau de la structure, la création d'un nouvel index et l'apprentissage par l'échec.

5.5.5. Apprentissage par l'échec

Dans la situation où le professeur corrige une action, il s'agit pour le système de ne plus commettre cette erreur dans le futur. Pour cela, on peut ajouter des connaissances négatives au cas qui a été utilisé et qui n'a pas été jugé opportun par le professeur, ou bien modifier les valeurs des pénalisations au niveau des différences observées entre le cas nouveau et le cas ancien inopportun.

Une connaissance négative peut se placer au niveau de l'attribut d'un objet de la structure. Il s'agit donc d'attributs (par exemple $Gr^{old}[i][j].CN\_\phi$ avec Gr^{old} désignant un cas /index en mémoire contenant des objets dont un des attributs est l'objet CN_ϕ qui est en l'occurrence un tableau de chaînes de caractères comportant les valeurs interdites du diamètre ϕ de la conduite $i \rightarrow j$).
Il est à noter que si une adaptation a porté sur l'ensemble des valeurs des diamètres ou des capacités (bijection entre deux cas) et a échoué, alors on l'indique grâce à un paramètre du cas ancien : *CN_bijection_*ϕ ou *CN_bijection_capacité* qui comportera l'ensemble interdit. De la même façon, si l'adaptation du cas ancien concernant une conduite supplémentaire au niveau d'un réservoir i a provoqué un échec, on introduit dans la structure $Tank^{old}$ une connaissance négative $Tank^{old}[i].CN\_conduite\_sup\_amont/aval$.

Deux situations se présentent :

● le cas remémoré est jugé inopportun car l'action correspondante n'est pas bonne (l'adaptation qui a été tentée a échouée), et il n'existe pas de cas en mémoire qui aurait permis

de résoudre le problème, alors le professeur indique l'action qui aurait due être effectuée, et il s'agit ensuite de discriminer le cas nouveau et le cas ancien inopportun.

● le cas remémoré est jugé inopportun mais il existe au sein de la base un cas qui n'a pas été choisi mais qui est le bon cas. Il s'agit alors de discriminer le cas ancien inopportun et le bon cas ancien *par rapport au cas nouveau*.

La discrimination consiste en premier lieu à identifier l'endroit précis au niveau du cas ancien (remémoré) qui a provoqué l'échec. Une façon de procéder est de comparer les deux structures et états de réseau des deux cas à discriminer. Pour cela, on utilise la procédure de mise en correspondance qui est sensible aux différences. On rappelle en outre que les valeurs des pénalisations correspondant à des différences entre deux cas (ou deux index) ont des valeurs par défaut qui sont fixes, mais peuvent être modifiées par spécialisation au sein d'un cas/index. Ainsi, toute modification des valeurs de pénalisation au sein d'un cas/index entraîne la création de valeurs de pénalisation spécifiques, donc les pénalisations au sein de ce cas/index passent du statut de valeurs par défaut au statut de valeurs spécifiques.

● Pour la première situation ci-dessus, on utilise la mise en correspondance (noté "*mec*" par la suite) entre le cas nouveau et le cas ancien inopportun, et l'idée est alors d'augmenter le score de mec afin d'éviter que le cas ancien soit de nouveau choisi dans le futur dans une même situation. Pour ce faire, on augmente les valeurs de pénalisations au sein du cas ancien et correspondant aux différences observées. Eventuellement on peut baisser ces valeurs au sein du cas nouveau pour accentuer l'écart.

● Pour la seconde situation, on utilise la *mec* entre d'une part le cas nouveau et le cas ancien inopportun, et d'autre part entre le cas nouveau et le bon cas ancien, et il s'agit d'inverser le rapport des deux scores. Avant, on avait :
score_mec (cas nouveau, cas inopportun) < score_mec (cas nouveau, bon cas),
on veut maintenant :
score_mec (cas nouveau, cas inopportun) > score_mec (cas nouveau, bon cas).
Une stratégie simple consiste à baisser, au sein du bon cas (qui aurait dû être choisi), les valeurs des pénalisations correspondant à des différences observées entre le bon cas et le cas nouveau mais non observées entre le cas nouveau et le cas inopportun. Inversement, on peut également augmenter les valeurs des pénalisations, au sein du cas inopportun, correspondant à des différences observées entre le cas inopportun et le cas nouveau mais non observées entre le cas nouveau et le bon cas.
A la suite de cette discrimination, le cas nouveau et le bon cas ancien sont suffisamment similaires et l'on peut tenter une généralisation (voir 5.5.1 et 5.5.4).

Remarque : on s'aperçoit qu'il peut tout de même apparaître des *conflits*, si on a augmenté auparavant certaines valeurs de pénalisation afin de discriminer deux cas (ou index), et on se trouve maintenant dans une nouvelle situation où on veut baisser ces mêmes valeurs. Il faudrait alors en théorie stocker en mémoire toutes les opérations de modification de valeurs de pénalisation lors des discriminations de couples de cas/index, de façon à chercher des valeurs de pénalisations qui ne provoquent aucun conflit. On a donc un problème de satisfaction de contraintes :
soit $D_{i,j}$ la j ième discrimination lors de la i ième simulation, entre les deux cas C^{new} et C^{old} (par exemple il s'agit de discriminer un cas nouveau C^{new} et un cas ancien inopportun C^{old}).
On veut par exemple : $new\_score\_mec\ (C^{new}, C^{old}) > old\_score\_mec\ (C^{new}, C^{old})$
avec $new\_score\_mec\ (C^{new}, C^{old}) = \Sigma_{\neq}\ (P_i^{new} \neq)$ et $old\_score\_mec\ (C^{new}, C^{old}) = \Sigma_{\neq}\ (P_i^{old} \neq)$

Le score de mise en correspondance est la somme des pénalités relatives à des différences observées. $P_i^{*}{}_{\neq}$ (* = new ou old) représentent donc les valeurs de pénalisation correspondant à des différences observées entre deux cas/index. Ces valeurs doivent être indicées par le numéro *i* de la simulation afin de tenir compte d'une contrainte supplémentaire : pour une différence donnée $\neq_k$, la valeur de pénalisation nouvelle (après discrimination) lors de la *i* ième simulation $P_i^{new}{}_{\neq_k}$ est la même que l'ancienne (avant discrimination) $P_{i+1}{}^{old}{}_{\neq_k}$ lors de la *i+1* ième simulation.

Par ailleurs, sur les deux situations précédentes, trois phénomènes viennent se greffer :

● Tout d'abord, le système doit en outre être capable de juger les différences importantes de celles qui ne le sont pas. Les valeurs numériques des pénalisations peuvent alors être utilisées comme caractérisant l'importance associée à une différence. Deux stratégies complémentaires sont possibles :

- s'il n'y a qu'une seule différence, alors on peut de manière sûre de discriminer les deux cas en incorporant une connaissance négative au sein du cas inopportun au niveau de l'élément correspondant à cette différence, plutôt que de modifier les valeurs des pénalisations comme indiqué ci-dessus. Cette stratégie peut être étendue à la situation où il y a plusieurs différences mais dont une a une importance beaucoup plus grande que les autres (donc une valeur de pénalisation beaucoup plus grande). Mais rien ne garantit que cette dernière stratégie soit toujours juste.

- s'il y a plusieurs différences dont les valeurs de pénalisation sont comparables, alors le problème est plus délicat car on ne peut pas savoir quelle est (ou quelles sont) la différence qui est vraiment discriminatoire. On pourrait classifier les différences en deux groupes (Importantes, Non Importantes) par la méthode des k-moyennes en fonction des valeurs de pénalisation pour chaque type de différence. Pour les différences appartenant au groupe *I* (Importantes), les valeurs des pénalisations correspondantes seraient alors augmentées au sein du cas ancien inopportun (et éventuellement diminuées au sein du cas nouveau).

● *Discrimination des cas similaires situés sous le même index*. Il ne s'agit pas seulement de discriminer le cas nouveau et le cas retrouvé qui a échoué, mais également de discriminer le cas nouveau et les cas situés sous le même index (que le cas ayant échoué), si ces derniers ont obtenu un score de mise en correspondance comparable à celui du cas ayant échoué.

● *Discrimination des index similaires*. Il est à noter que dans les deux situations, la discrimination se fait en réalité soit au niveau des cas si leurs index sont les mêmes, soit au niveau des index si ceux-ci sont différents. En effet, il est possible que l'échec soit du à un mauvais choix d'index. Un index a donc été jugé similaire à tort (voir 5.5.6.2. bilan de l'apprentissage – problème mal résolu). La discrimination se fait de la même façon que pour les cas, grâce à la procédure de mise en correspondance.

5.5.6. Bilan de l'apprentissage

5.5.6.1. Bilan relatif à l'incorporation d'un nouveau cas

Le cas qui sera stocké correspond à la configuration partielle du réseau et à l'état du système à l'instant indiqué par le professeur. Comme il a été dit plus haut, on n'enregistre pas toute la configuration réseau, mais uniquement la partie utile en fonction du but à atteindre (éliminer

une alarme, accélérer le processus, ...), et éventuellement mise en évidence grâce au cas ancien choisi pour faire face à la nouvelle situation. Ce cas aura été créé lors de la résolution, mais il est à noter que la modélisation du cas stocké peut être différente de celle du cas créé lors de la résolution, notamment si l'échec est du à un mauvais choix d'index ou du but à atteindre (on peut très bien imaginer que des index ou des buts puissent être en concurrence).
S'il y a eu échec de résolution, le professeur indique l'action qui aurait du être effectuée par le système, et éventuellement le but associé (minimiser la durée, gestion d'une situation de double alarme, …) s'il y a eu erreur au niveau du choix du but à atteindre. Le système doit alors de lui-même créer et modifier ses connaissances afin :

● d'intégrer ce nouveau cas, ré-organiser la mémoire (création d'index, classification des cas et des index),

● de ne plus commettre ce genre d'erreur (apprentissage par l'échec),

● le cas échéant de généraliser au niveau de l'action,

● et de généraliser au niveau de la structure et de l'état du réseau.

L'enregistrement d'un nouveau cas consiste à placer la configuration (partielle) du réseau, soit sous un nouvel index pour intégrer uniquement ce cas, soit sous l'index qui avait été utilisé pour résoudre ce cas (voir 5.5.2 : création d'index). Dans cette dernière situation, il s'agit ensuite de classer ce nouveau cas au sein de la hiérarchie existante sous l'index.

Le problème qui se pose à propos de l'enregistrement du cas nouveau (ou de l'index créé) est de savoir comment *intégrer les connaissances négatives et les valeurs de pénalisation* correspondant au cas ancien qui a été utilisé :

● **si l'adaptation a réussi**, alors on copie les connaissances négatives et les valeurs de pénalisation dans le nouveau cas, sauf là où se situent les différences entre les deux cas : à ce niveau, on copie les connaissances négatives, et on abaisse les valeurs de pénalisation *fortes* s'il y en a, puis on copie ces valeurs au sein des deux cas. S'il s'agit de pénalisations par défaut, alors elles prennent le statut de pénalisations spécifiques. L'explication est que l'on a tenté une adaptation malgré un score de mise en correspondance élevé (dû à la pénalisation forte), et l'abaissement de ces valeurs vise à diminuer le score de mise en correspondance afin de conforter, renforcer le cas qui a été choisi en mémoire. Mais il faut veiller à gérer d'éventuels conflits lors des modifications des valeurs (voir la remarque du chapitre apprentissage par l'échec).

● **si l'adaptation a échoué**, alors on copie les connaissances négatives et les valeurs de pénalisation dans le nouveau cas, sauf là où se situent les différences entre les deux cas : à ce niveau, on copie les connaissances négatives s'il y a plusieurs différences (une modification des pénalisations s'est produite – voir apprentissage par l'échec), ou on copie les valeurs de pénalisations s'il n'y a qu'une seule différence (car la création d'une connaissance négative s'est produite au niveau de cette différence – voir apprentissage par l'échec).

5.5.6.2. Bilan des diverses situations d'apprentissage

Si l'opérateur artificiel a bien résolu le problème (c'est-à-dire de façon proche de l'optimal ou de façon satisfaisante selon le professeur), alors deux situations se présentent :

1- si le nouveau cas a été jugé identique (au niveau de la structure et au niveau de l'état) à un cas en mémoire, et a donc atteint le sous-but fixé *sans adaptation*, alors il n'est pas nécessaire d'enregistrer ce nouveau cas.

2- mais si le nouveau cas a été jugé similaire à un cas en mémoire et a atteint le sous-but fixé *avec adaptation*, alors trois situations peuvent se présenter :

2.1- s'il est possible de généraliser / fusionner ces deux cas (généralisation au niveau de la structure et de l'état du réseau, et/ou au niveau de l'action), alors on modifie le cas en mémoire afin de tenir compte du cas nouveau mais on n'enregistre pas le cas nouveau.

2.2- si les deux structures ne diffèrent que par le nombre de conduites (voir remarque 2), alors on ne généralise pas et on n'enregistre pas le cas nouveau. En effet, les nombres de conduites en plus ou en moins en amont ou en aval d'un réservoir sont des attributs de l'objet correspondant à ce réservoir, mais en tant que connaissances négatives. Donc ces attributs ne sont modifiés qu'en cas d'échec. Dans l'exemple ci-dessous, ces variations n'introduisent pas de modification de l'action, et l'on s'adapte facilement du cas 1 au cas 2 ou inversement. Si cela pose problème, on introduit des connaissances négatives (voir ci-dessous apprentissage lorsque le problème est mal résolu).

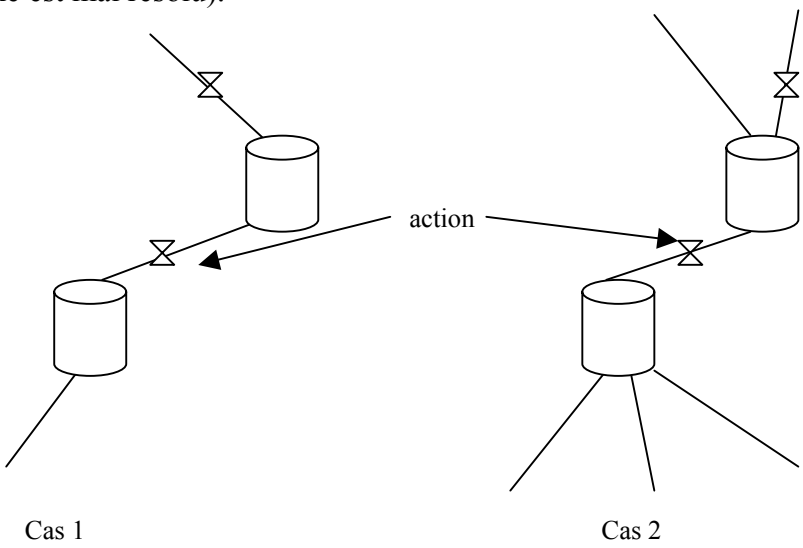


Figure 41 : Situation n'engendrant pas le stockage du cas nouveau

Remarque 1 : si les deux structures diffèrent par le nombre de conduites, et par un ou plusieurs autres paramètres, alors on ne tient compte que des variations au niveau des autres paramètres et on se ramène donc à la situation précédente (tentative de fusion), ou à la suivante (fusion impossible).

Remarque 2 : il est à noter que cela ne concerne pas les conduites reliant des réservoirs représentés explicitement au sein de la structure. D'après le schéma ci-dessous, si les deux cas diffèrent par la présence d'une conduite entre deux réservoirs, alors il semble préférable, comme cela a déjà été dit, d'enregistrer le nouveau cas car il s'agit là d'une différence importante, et de créer un index intégrant ce nouveau cas.

Cas en mémoire

Cas à résoudre : à stocker sous un nouvel index

Figure 42 : Situation engendrant le stockage du cas nouveau

2.3- si on ne peut pas fusionner les deux cas, il est alors nécessaire d'enregistrer le cas nouveau. Deux situations se présentent :

2.3.1- il y a des différences importantes, c'est-à-dire au niveau de l'abstraction / index (nombre de réservoirs différent, conduite supplémentaire entre deux réservoirs représentés explicitement, existence d'une vanne, …). Il faut créer un nouvel index intégrant uniquement le cas nouveau.

2.3.2- il y a plusieurs petites différences sur les valeurs de paramètres propres au cas mais non à l'index (capacité d'un réservoir, diamètre d'une conduite, …), et la généralisation au niveau de ces paramètres n'a pas été possible, mais les deux cas sont tout de même similaires. On intègre alors le nouveau cas au sein de l'index du cas ancien.

Si l'opérateur artificiel a mal résolu le problème, le professeur indique quelles situations n'ont pas été bien gérées par le système et indique pour chacune d'elles, soit l'action que le système aurait dû faire à cet instant précis (selon le pas de temps de l'agent artificiel), soit le cas (existant) qui aurait du être choisi. Comme cela a été dit précédemment, le système conserve en mémoire tous les cas nouveaux créés pour résoudre le problème (à partir de l'état du réseau et l'action réalisée par le système). Un cas nouveau est créé à chaque pas de temps de calcul de l'agent artificiel. Trois situations se présentent au cours desquelles le système peut alors incorporer un nouveau cas grâce au professeur :

1- le cas nouveau n'a pas été jugé similaire à un index existant (tous les index ont été rejetés). Le système devra donc créer un nouvel index qui intégrera ce nouveau cas, et classer cet index. Cela peut se produire par exemple lorsqu'une situation vraiment nouvelle est rencontrée par le système.

2- le cas nouveau a été jugé similaire à un index existant, mais n'a pas été jugé similaire à un cas existant sous cet index (tous les cas ont été rejetés). Il faudra donc intégrer ce nouveau cas puis le classer.

3- le cas nouveau a été jugé similaire à un index existant, puis à un cas existant, mais l'action réalisée n'est pas jugée bonne par le professeur, soit parce que l'adaptation a échoué, soit parce que le bon cas existait mais n'a pas été retrouvé. Quatre situations se présentent alors :

3.1- l'action que le professeur indique appartient au même but de haut niveau (accélérer le processus, double alarme, …). Les deux structures possèdent une différence importante, représentée au niveau de l'index (nombre de réservoirs, présence d'une conduite sans vanne, conduite supplémentaire), et l'adaptation a échoué, alors on crée un nouvel index qui intègre uniquement le nouveau cas (voir création d'un index), et on classe ce nouvel index. Il s'agit également d'ajouter une étiquette à l'index du cas inopportun en mémoire sous la forme d'une connaissance négative (ou bien de modifier des valeurs de pénalisation des différences observées – voir apprentissage par l'échec).

3.2- l'action que le professeur indique appartient au même but de haut niveau. Les deux structures possèdent une ou plusieurs petites différences relatives à la structure ou à l'état du réseau mais qui ne sont pas représentées au niveau de l'index (position de vanne, diamètre d'une conduite, hauteur d'un réservoir, ...), et l'adaptation a échoué, alors on intègre le cas nouveau sous l'index du cas ancien, on le classe, et on ajoute une étiquette au cas inopportun en mémoire sous la forme d'une connaissance négative (ou bien on modifie des valeurs de pénalisation des différences – voir apprentissage par l'échec).

3.3- l'action que le professeur indique n'appartient pas au même but de haut niveau. Ce but doit alors être explicitement indiqué par le professeur. Il s'agit d'ajouter une étiquette au cas qui a été retrouvé à tort, ainsi qu'à son index, sous la forme d'une connaissance négative (ou bien de modifier des valeurs de pénalisation des différences – voir apprentissage par l'échec). Ensuite, on crée un index (intégrant uniquement le nouveau cas) au niveau du but indiqué par le professeur. Enfin, l'index doit être classé parmi les index déjà existants au sein de ce but.

3.4- il existe un cas/index en mémoire qui aurait permis de bien résoudre le problème, mais celui-ci n'a pas été choisi. Le professeur indique le cas/index en mémoire qui aurait dû être choisi, et le système doit discriminer le cas/index retrouvé qui a entraîné l'échec et le cas/index en mémoire qui aurait permis la résolution par rapport au cas/index nouveau, puis le système tente de généraliser (voir apprentissage par l'échec).

Remarque : si le système ne trouve pas d'index ou de cas en mémoire lui permettant d'agir, cela ne signifie pas forcément une erreur, mais un *cas (implicite) de non-action*. Il s'agit d'une erreur seulement si le professeur l'indique.

5.6. EXPERIMENTATIONS INFORMATIQUES

Nous avons testé deux agents à base de cas : l'un, nommé *Tactic 1*, utilise tous les index/cas développés sauf l'index n° 9 (stratégie de minimisation de la durée) car cette stratégie particulière nécessite une analyse assez fine du problème et n'est donc pas forcément utilisée par un opérateur humain. L'autre agent, nommé *Tactic 2*, utilise tous les index/cas.

	$V_{déb}$ (m3)	D_t (sec.)	Critère C
Config 3 Tactic=1 ΔV^{alarme}=1.5 Δt = 2	3.34	586	32.6
Config 3 Tactic=2 ΔV^{alarme}=1.5 Δt = 2 ΔV_ZO= 3	0	450	22.5
Config 3 Tactic=1 ΔV^{alarme} = 2 Δt = 3	4.53	586	33.8
Config 3 Tactic=2 ΔV^{alarme} = 2 Δt = 3 ΔV_ZO= 6	0	456	22.8
Config 3 Tactic=1 ΔV^{alarme} = 10 Δt = 2	0	586	29.3
Config 3 Tactic=2 ΔV^{alarme} = 10 Δt = 2 ΔV_ZO= 3	0	464	23.2
Config 1 Tactic=1 ΔV^{alarme}=1.5 Δt = 2	3.88	737	40.7
Config 1 Tactic=2 ΔV^{alarme}=1.5 Δt = 2 ΔV_ZO= 3	1.12	675	34.9
Config 1 Tactic=1 ΔV^{alarme} = 2 Δt = 3	4.26	740	41.3
Config 1 Tactic=2 ΔV^{alarme} = 2 Δt = 3 ΔV_ZO= 6	1.72	698	36.6
Config 1 Tactic=1 ΔV^{alarme} = 10 Δt = 2	2.44	747	39.8
Config 1 Tactic=2 ΔV^{alarme} = 10 Δt = 2 ΔV_ZO= 3	0.70	679	34.6

Config i : une configuration du micro-monde
$V_{déb}$ = volume total débordé
D_t = durée de la simulation
$C = V_{déb} + \alpha.D_t$ = objectif global à minimiser (α = 1/20)
Δt = pas de temps de calcul de l'agent artificiel
ΔV^{alarme} = volume d'alarme : si ($Capacité_i - V_i$) > ΔV_{al} alors alarme du réservoir i
ΔV_ZO = zone floue (en m3) autour du volume optimal = $2.V_\varepsilon$ (voir 5.2.3. index n° 9)

Tableau 12 : Résultats des tests de l'opérateur basé sur des cas

Les résultats sont nettement améliorés par rapport aux agents qualitatifs, grâce à l'utilisation de diverses stratégies de gestion de situations typiques pouvant se produire. Par exemple, pour la configuration 3 avec ΔV^{alarme} =1.5, le critère *C* est passé de 33.45 (agent qualitatif) à 22.5 (agent à base de cas). Pour la configuration 1 avec ΔV^{alarme} =1.5, le critère *C* est passé de 42.1 (agent qualitatif) à 34.9 (agent à base de cas).

On note comme prévu l'impact important sur la durée de l'emploi de la tactique n° 2 (*Tactic 2*). Cet impact est tout de même moins important lorsque la configuration du réseau comporte au moins une conduite sans vanne (configuration n° 1). En effet, dans ce cas, la stratégie de minimisation de la durée est plus délicate à appliquer car il est plus difficile de maintenir dans une certaine zone les volumes des réservoirs en aval des conduites sans vanne, et cela nécessiterait la mise en place d'une stratégie supplémentaire d'anticipation. Il est à noter en outre que cette stratégie de minimisation de la durée a également un impact important sur les

débordements des réservoirs. Ceci est dû au fait qu'en cherchant constamment à maintenir les volumes des réservoirs dans une certaine zone, on évite alors de se trouver dans la zone d'alarme.

Pour la configuration 1, les simulations avec la tactique n° 1 (*Tactic 1*) et un volume d'alarme ΔV^{alarme} faible montrent des débordements. Ceux-ci sont dûs au fait que le seuil d'alarme est calé très près du sommet des réservoirs. Cela correspond à un comportement risqué, mais qui permet de gagner un peu de temps. En augmentant le volume d'alarme, on diminue fortement les débordements, mais la durée totale augmente un peu, ce qui correspond à un comportement plus prudent.
On note également que le calage des seuils d'alarme ΔV^{alarme} et des zones floues ΔV_ZO autour des volumes optimaux est spécifique à chaque configuration de réseau. Mais on peut peut-être dégager une relation : en effet, pour la configuration n° 3, les meilleurs résultats sont obtenus avec un *comportement risqué* (ΔV^{alarme} faible, donc on considère qu'un réservoir est en alarme lorsque le niveau d'eau est proche de la limite du réservoir) et *attentif* (ΔV_ZO faible, donc on cherche à maintenir les volumes des réservoirs dans des zones de faible amplitude). Pour les configuration n° 1, les meilleurs résultats sont observés pour un *comportement prudent* (ΔV^{alarme} important). Ainsi, lorsque le réseau comporte au moins une conduite sans vanne (configuration 1), mieux vaut avoir un comportement prudent, et lorsque le réseau ne comporte pas de conduites sans vanne (configuration 3), on peut se permettre un comportement risqué (ΔV^{alarme} faible) mais attentif (ΔV_ZO faible). Il est tout de même à noter que pour un certain nombre de configurations, ces deux paramètres ont une influence plus limitée.

Remarque 1 : tous les résultats précédents sont confirmés par les tests plus nombreux correspondant à la coopération entre agents artificiels (voir 8.2.1.).

Remarque 2 : la configuration 6 n'a pas été testée car il s'agit simplement d'une variante de la configuration 1.

5.7. CONCLUSION

Un agent à base de cas a été implémenté et testé. Un cas représente une partie de réseau (un petit nombre de réservoirs connectés), qui, si il est reconnu comme applicable et le plus similaire (parmi les autres cas présents en mémoire) à la situation courante, ce qui nécessite une mise en correspondance entre le graphe représentant ce cas et le graphe représentant la situation courante, provoque l'exécution d'une action.
Cet agent donne des résultats meilleurs que l'agent qualitatif, car les cas peuvent représenter des situations complexes et globales (gestion de plusieurs réservoirs avec un but à long terme : anticiper le débordement, minimiser la durée totale).
Nous allons maintenant examiner si le raisonnement par classification peut être appliqué à notre problème.

Chapitre 6

CLASSIFICATION ET RECONNAISSANCE DE FORMES

Nous présentons le raisonnement par classification à partir d'exemples, qui consiste à apprendre des classes en généralisant la description des exemples de chaque classe, puis la tentative d'application de ces modèles à notre étude : en effet, les problèmes soulevés nous ont empêché de mettre au point un agent à base de classification.

6.1. CLASSIFICATION SANS CONNAISSANCES *A PRIORI* A PARTIR D'EXEMPLES

6.1.1. Présentation

Cette méthode fonctionne en deux phases :

● Apprentissage de classes à partir d'exemples.

Toutes les méthodes utilisent une entrée représentant les caractéristiques d'un exemple, et fournissent une classe en sortie. Un exemple est généralement représenté sous la forme d'un vecteur de couples attribut-valeur. Les attributs peuvent être :

- binaires,

- numériques (éventuellement séquentiels),

- nominaux (éventuellement organisés en hiérarchie),

Une classe est alors représentée selon un formalisme du type logique des propositions.

Mais un exemple peut également être sous la forme d'un vecteur de prédicats, une classe est alors représentée selon un formalisme du type logique du premier ordre (Programmation Logique Inductive [Cornuejols et Miclet 02]).

Une classe peut être représentée par l'ensemble de ses membres, ou par un membre typique, abstrait et idéalisé, c'est-à-dire un prototype [Haton 91].

L'apprentissage consiste à généraliser la description des attributs des exemples de chaque classe. Cette généralisation peut s'appuyer sur des critères statistiques ou sur des opérateurs

de généralisation. L'apprentissage artificiel, issu de l'IA, propose plusieurs méthodes de classification, qui se différencient en général selon trois caractéristiques :

- les attributs des exemples sont de nature *numérique* ou *symbolique*,

- création d'une *hiérarchie* de classes ou plus simplement d'une *liste* de classes,

- apprentissage *supervisé*, c'est-à-dire qu'à chaque exemple, la catégorie associée est donnée, ou *non supervisé*, et il s'agit par exemple de partitionner des exemples comme dans la méthode des k-means (voir annexe 7). On parle souvent de découverte automatique à propos de l'apprentissage non supervisé [Cornuejols et Miclet 02].

Le problème est de fournir suffisamment d'exemples pour que les capacités de généralisation s'expriment, mais pas trop car il y a risque de surapprentissage : dans ce cas, la classe apprise est d'une trop grande complexité, et représente trop exactement l'ensemble d'apprentissage, c'est-à-dire réalise en pratique un apprentissage par cœur, au détriment de sa qualité de généralisation.

Parmi les méthodes de classification, on peut citer de manière non exhaustive l'espace des versions, les arbres de décision probabilistes, les réseaux connexionnistes et les méthodes de *clustering* statistique ou flou parmi lesquelles se situe la méthode LAMDA mise au point au LAAS-CNRS par J. Aguilar-Martin [Piera *et al.* 89] [Waissman 00] (voir annexe 6 pour une présentation de ces méthodes).

Les travaux de A. Cornuéjols [Cornuejols et Miclet 02] définissent un panorama assez complet des différentes méthodes de classification à partir d 'exemples. J.P. Haton [Haton 91] insiste lui davantage sur les méthodes de classification symboliques. H. Abdi [Abdi 94] propose une bonne introduction aux réseaux connexionnistes (théorie mathématique, principaux modèles).

● Reconnaissance d'un élément nouveau, c'est-à-dire trouver sa classe d'appartenance, par exemple en calculant une distance (ou un degré d'adéquation/similarité) soit entre cet élément et les exemples de chaque classe (puis en effectuant une moyenne pour chaque classe), soit entre cet élément et chaque classe, ou encore en insérant cet élément dans la hiérarchie des classes.

6.1.2. Application dans le cadre de notre étude : agent artificiel basé sur la classification description → action

De manière simple, on peut imaginer un vecteur d'entrée constitué par la *description courante du système*, qui peut inclure par exemple l'alarme dans chaque réservoir qui est donc un attribut binaire (ou bien les hauteurs d'eau et les tendances mais il s'agit d'attributs numériques réels), ainsi que la position des vannes (qui est également un attribut binaire). Si la sortie correspond à l'action envisagée sur une des vannes, l'apprentissage consiste alors à fournir des couples (description → action). Pour résoudre ce problème, on peut appliquer une méthode classique du type espace de versions ou arbre de décision.

Un problème se pose avec ce type de description car il faut effectuer un apprentissage propre à chaque configuration du réseau. En effet, la taille du vecteur d'entrée est fonction du nombre de réservoirs et du nombre de conduites, ce qui représente un inconvénient sérieux.

Remarque 1 : la représentativité de la description choisie influe sur l'efficacité de la méthode. Théoriquement, la description optimale est l'observation de *l'état du système dynamique* analysé. En général, celui-ci n'est pas totalement observable et des approches sous-optimales sont accessibles.

Remarque 2 : pour une configuration donnée, le nombre de descriptions est potentiellement infini, si l'on considère la hauteur d'eau (un nombre réel) comme un paramètre de la description.

Remarque 3 : à une classe, on peut de manière indifférenciée lui associer une action ou une séquence d'actions. Donc on peut également imaginer une sortie sous la forme d'une séquence de quelques actions, correspondant à une stratégie spécifique.

Nous avons étudié le problème de l'apprentissage par classification pour une configuration donnée du réseau (configuration n° 3 : voir figure 2).

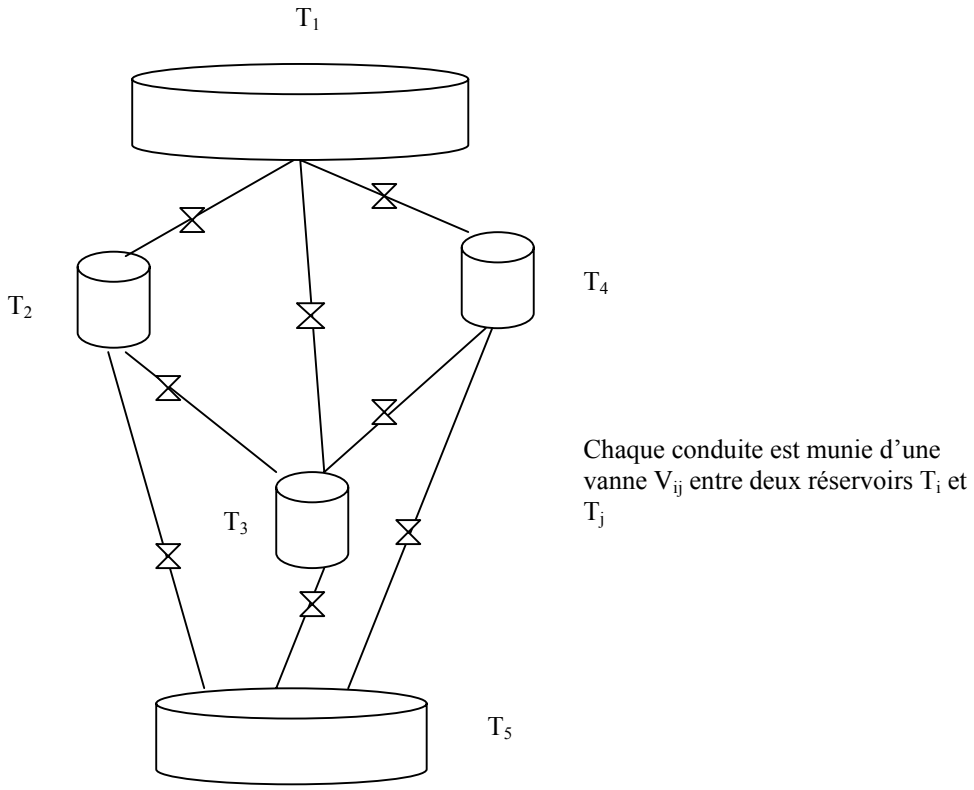


Figure 43 : Apprentissage d'actions dans une configuration donnée

Examinons tout d'abord un exemple. Supposons que l'on veuille que le système apprenne dans quels cas ouvrir V_{35} (la vanne reliant T_3 à T_5). Il s'agirait d'apprendre par exemple la règle très simple : si T_3 est en alarme et V_{35} est fermée alors ouvrir V_{35}. Les positions de toutes les autres vannes n'ont pas d'importance pour cette règle, et on voit bien qu'il faudrait un grand nombre d'exemples (des descriptions) dont l'action associée (la classe) serait "ouvrir V_{35}" pour que l'algorithme d'apprentissage puisse généraliser, c'est-à-dire ici juger que tous les attributs des vecteurs d'entrée fournis en exemples pour cette classe sauf deux (l'état d'alarme de T_3 et la position de V_{35}) ne sont pas pertinents. En théorie, avec un vecteur d'entrée comportant n attributs binaires, il faudrait 2^{n-2} exemples (n-2 attributs non pertinents) pour apprendre la règle précédente. S'il y en a moins, il n'est pas évident que l'algorithme généralise suffisamment.

Le problème devient encore plus important si l'on souhaite faire apprendre au système une abstraction de la règle précédente : si T_i est en alarme et V_{ij} est fermée alors ouvrir V_{ij}.

L'apprentissage consisterait à fournir un grand nombre de descriptions, et ceci pour chaque action possible (nombre de vannes multiplié par deux). On peut alors procéder de deux façons. Soit un expert humain définit à l'avance les différentes descriptions pour chaque action, ce qui semble irréalisable car cela nécessiterait un travail trop important, soit on utilise les séquences d'actions effectuées par des opérateurs artificiels ou humains lors de tests, mais dans ce dernier cas il faut s'assurer que la séquence est de bonne qualité et cohérente avec l'apprentissage que l'on veut réaliser, et cela nécessiterait là aussi un travail très important.

La méthode de l'espace des versions ne permettrait pas d'exhiber un ensemble *S* d'hypothèses maximalement spécifiques (voir annexe 6) de bonne qualité (sauf si on fournit beaucoup d'exemples positifs), mais on pourrait fournir beaucoup d'exemples négatifs pour chaque classe (toutes les descriptions qui ne sont pas associées à la classe "ouvrir V_{35}") et donc peut être exhiber un ensemble *G* d'hypothèses maximalement générales de bonne qualité.

La méthode des arbres de décision (voir annexe 6) fonctionnerait peut-être un peu mieux car elle est basée sur la notion de probabilité, et donc on pourrait contrôler la taille de l'arbre en limitant le nombre de nœuds, par exemple en se contentant d'arrêter la division d'un nœud avec un critère légèrement supérieur à 50 % (valeur à fixer). Mais il n'est pas sûr que cela soit suffisant. Et quoi qu'il en soit, on sera toujours confronté au problème des exemples à fournir en entrée.

Nous avons testé la méthode *LAMDA* (voir annexe 6) avec la configuration n° 3. Des exemples ont été fournis en entrée du logiciel. Un exemple est modélisé par un vecteur composé des proportions des 5 réservoirs ($h_i(t)/H_i \in [0\ ;1]$) et des positions binaires des 8 vannes. Ces exemples sont issus de simulations réalisées par l'opérateur artificiel *NQA*. La classification fournie par *LAMDA* n'est pas bonne car les classes ne sont pas nettement caractérisées : dans l'espace à deux dimensions (exemple-classe), on observe des nuages de points non distincts. Cela est dû au trop faible nombre d'exemples fournis pour chaque action possible. En effet, l'agent qualitatif ayant toujours le même comportement sur une configuration donnée de réseau, les exemples fournis par ces quelques simulations ne peuvent prétendre donner une vue suffisamment vaste permettant de généraliser toutes les actions possibles. Les résultats pourraient être améliorés d'une part en utilisant en entrée des données qualitatives (à domaine de valeurs fini) concernant les réservoirs (et non des hauteurs d'eau dont les domaines de valeurs sont infinis) et en fournissant davantage d'exemples, mais cette fois par l'intermédiaire de sujets humains.

Conclusion

Toutes ces méthodes de classification qui fonctionnent sans connaissance *a priori* nécessiteraient beaucoup d'exemples pour réussir à abstraire un concept de façon satisfaisante, donc un travail important à fournir en amont, et ceci pour chaque configuration du réseau.

Nous avons donc choisi de ne pas utiliser ce type d'agent pour la définition d'une coopération entre raisonnements.

D'autre part, il serait intéressant d'étudier la possibilité d'avoir un vecteur d'entrée plus abstrait, décrivant les caractéristiques importantes de la situation courante afin de ne plus dépendre de la configuration du réseau. Le paragraphe suivant décrit une méthode fonctionnant à partir de descriptions abstraites.

6.2. CLASSIFICATION HEURISTIQUE

Une approche très différente de la précédente est la classification heuristique [Haton 91] :

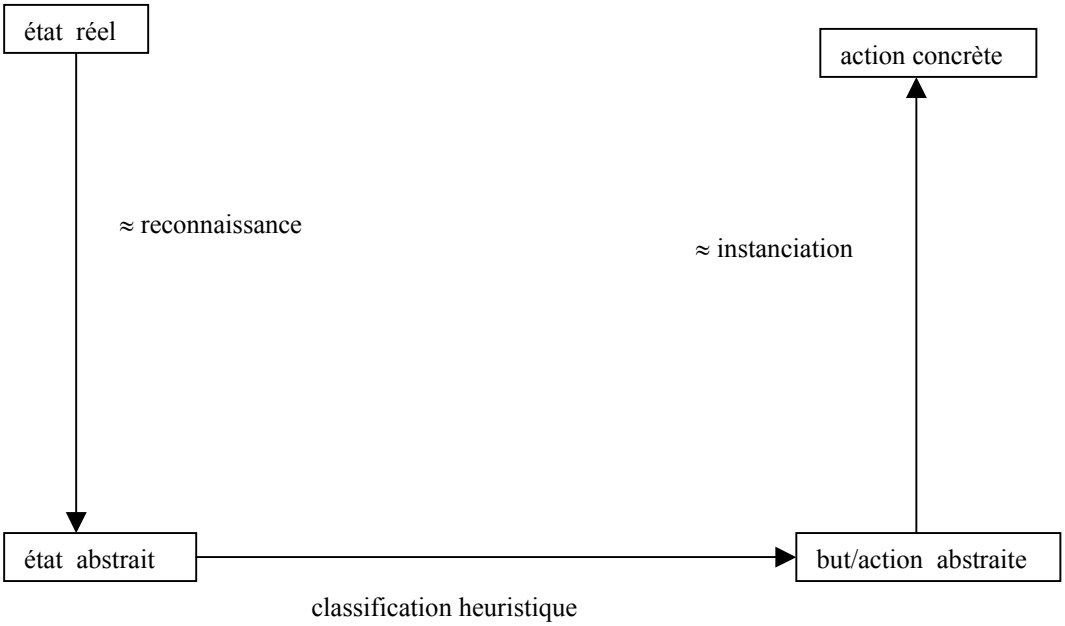


Figure 44 : Classification heuristique

L'idée est la suivante : à partir de la description réelle (concrète) du système, il s'agit de reconnaître l'application de règles générales (abstraites) de classification heuristiques (du type : C_1 *et* C_2 *et ... et* $C_n \Rightarrow A_1$) qui, à une description abstraite (C_1 *et* C_2 *et ... et* C_n) associent une action abstraite (A_1). Ensuite, on instancie cette action abstraite dans le cadre de la configuration du réseau.

La reconnaissance (tester si les conditions C_i de la règle générale sont satisfaites) et l'instanciation (trouver une action concrète, dans le cadre du problème à résoudre, à partir de la conclusion générale de la règle) se font grâce à des procédures spécifiques (une pour chaque C_i i = 1 à n, et une pour l'action A_1).

Il peut y avoir plusieurs règles déclenchables à partir d'une même description réelle. Pour résoudre ce problème, on peut définir un ordre sur les règles de classification, ou bien définir des méta-règles ou des procédures pour gérer les conflits éventuels.

Dans le cadre de notre étude, il est aisé d'appliquer cette méthode. Chaque procédure, qu'il s'agisse d'une condition ou d'une action, prend en paramètre la structure du réseau et sa description à un instant *t*.

Voici quelques exemples de règles de classification, simples à implémenter :

R1 : pas d'alarme **et** il existe un chemin fermé ⇒ ouvrir ce chemin

R2 : réservoir T_i en alarme **et** il existe une vanne fermée en aval de T_i ⇒ ouvrir cette vanne.
Remarque : on pourrait spécialiser cette règle en ouvrant la vanne de plus gros diamètre.

R3 : réservoir T_i en alarme **et** tendance de T_i = *INCL* (increase a lot) **et** toutes les conduites en aval de T_i = (vanne ouverte ou sans vanne) **et** il existe une vanne ouverte en amont de T_i ⇒ fermer la vanne de plus gros diamètre en amont de T_i

R4 : réservoir T_i en alarme **et** tendance de T_i = *INCS* (increase small) **et** toutes les conduites en aval de T_i = (vanne ouverte ou sans vanne) **et** il existe une vanne ouverte en amont de T_i ⇒ fermer la vanne de plus petit diamètre en amont de T_i

R5 : réservoir T_i en alarme **et** toutes les conduites en aval de T_i = (vanne ouverte ou pas de vanne) **et** il existe une conduite sans vanne en amont de T_i **et** les autres sont fermées ⇒ fermer une vanne en amont du réservoir en amont de T_i et relié à T_i par une conduite sans vanne

R6 : réservoirs T_i et T_j en alarme **et** reliés directement ⇒ ouvrir une vanne en aval de T_i ou T_j (sauf celle sur la conduite $i \rightarrow j$) sinon fermer une vanne en amont de T_i ou T_j (sauf celle sur la conduite $i \rightarrow j$)
Remarque : ouvrir V_{ij} (si elle était fermée) va certes diminuer l'alarme sur T_i mais va augmenter l'alarme sur T_j. De même, fermer V_{ij} (si elle était ouverte) va certes diminuer l'alarme sur T_j mais va augmenter l'alarme sur T_i.

R7 : réservoirs T_i et T_j en alarme (reliés directement ou non) ⇒ privilégier le réservoir qui a le plus gros rapport $\Sigma\Phi_e / \Sigma\Phi_s$ (Φ_e et Φ_s sont les diamètres entrant et sortant avec écoulement)
Remarque : "privilégier" doit être interprété comme sélectionner un des réservoirs en alarme puis lui appliquer une règle de gestion d'alarme (R2 à R9).

Conclusion

Cette méthode est susceptible de donner de bons résultats puisqu'il est tout à fait possible d'incorporer des règles générales de classification heuristique correspondant à des stratégies fines afin de gérer les situations d'alarme et minimiser la durée totale de vidange. Les procédures correspondant aux différentes conditions et à l'action de chaque règle doivent être fournies par l'utilisateur. Cependant, il est également possible, en théorie, d'inclure un module d'apprentissage de règles à partir d'exemples (à l'aide par exemple de la méthode des arbres de décision), mais comme on l'a vu, ce type de classification semble délicat à appliquer pour notre problème.

Ce modèle est proche d'un système de raisonnement à base de règles (RBR), mais la différence est qu'ici les règles sont abstraites dans le sens où des procédures sont nécessaires pour les instancier, alors que dans un système RBR, fondé sur la logique des propositions ou la logique des prédicats, l'instanciation des règles ne nécessite qu'un "*matching process*" simple (vérifier que l'objet testé est identique ou appartient à la classe de l'objet de la condition), mais l'avantage est un nombre réduit de règles par rapport à un système RBR.

Nous avons vu au chapitre 5 que le raisonnement à base de cas permet, d'une façon différente, d'implanter ce type de règles heuristiques, mais avec plus de flexibilité au niveau de l'application, grâce aux notions de similarité et de mise en correspondance, et en outre il possède des capacités d'apprentissage très développées et applicables à notre problème. Un système CBR constitue donc une bien meilleure approche que la classification heuristique pour notre problème.

6.3. CLASSIFICATION ACTION → BUT

On peut également utiliser le raisonnement par classification pour catégoriser des actions (ou des séquences d'actions) par l'intention (le but associé à l'action). On peut même aller plus loin et définir des classes représentant des intentions de manière plus précise en associant des caractéristiques comportementales de l'opérateur humain en activité de supervision, par exemple telle action a été réalisée dans le but de réduire un état d'alarme, mais *selon un comportement prudent*, une autre action dans le but d'accélérer le processus mais *en prenant des risques*.

Un opérateur (humain ou artificiel) qui résout le problème cherche à atteindre des buts successivement en fonction de la description du système (voir Intentions et buts – 7.1.1.2.1.). L'opérateur qualitatif possède deux buts principaux (en cas d'alarme, retourner à une situation sans alarme, et en cas de non alarme, accélérer le processus). L'opérateur basé sur des cas associe chaque cas à un but.

Il ne s'agit donc pas ici de résoudre le problème de supervision. Cette approche est utile pour comparer les raisonnements d'un opérateur humain avec celui d'un agent artificiel (en mode Hors ligne - voir 7.1). En effet, comparer deux séquences d'actions peut être vu comme la reconnaissance des intentions associées aux actions, puis la comparaison de ces intentions. Il semble plus aisé de comparer deux séquences de buts que deux séquences d'actions.

Un algorithme de classification doit donc associer une action, réalisée par un opérateur humain, au but de cette action. Ainsi, la structure du problème change par rapport aux chapitres précédents. Plus exactement, cet algorithme utilise en entrée un couple description-action. En effet, il faut prendre en compte non seulement l'action, mais également la description du système juste avant l'action, sachant que la description est un vecteur (positions des vannes et état d'alarme des réservoirs par exemple). On peut également représenter un couple description-action par un couple (plus exactement une séquence) de descriptions (la description juste avant chaque action et la description juste après chaque action). La sortie représente la classe (l'intention/but de l'action).

Ainsi, il faut développer une méthode de classification, qui à un couple de descriptions (ou un couple description-action) associe une classe (l'intention). Il s'agit ensuite de fournir à l'algorithme des exemples de couples description-action pour chaque intention. La méthode de classification doit prendre en compte l'aspect séquentiel de l'entrée (une séquence de deux descriptions), et pourrait s'appuyer par exemple sur les réseaux connexionnistes (les réseaux récurrents et les réseaux d'ordre deux) ou sur une méthode d'apprentissage d'*automates*. Un problème potentiel serait que l'algorithme considère une description du système comme un symbole. Or, dans notre étude nous définissons une description de façon détaillée (sous forme d'un vecteur), et le nombre de descriptions est potentiellement infini. Cela nécessite donc un travail supplémentaire consistant à définir des *descriptions types* du système (qui seront les symboles) et avoir une procédure (basée sur la classification ou sur la similarité) qui, en

fonction de la description réelle du système, associe une *description type*. Nous retombons sur les problèmes de classification précédemment cités : beaucoup d'exemples (sous forme de descriptions) sont nécessaires pour réussir à abstraire un concept de façon satisfaisante, et le vecteur d'entrée dépend de la structure du réseau, ce qui nécessite un apprentissage pour chaque configuration. Nous avons donc décidé de ne pas implémenter cette méthode, et nous avons développé et implémenté une autre approche permettant de découvrir les intentions associées aux actions (voir 7.1.1.2.2.).

6.4. CONCLUSION

La classification à partir d'exemples ne permet pas de mettre au point un agent artificiel, car cette technologie fonctionne sans connaissance *a priori* et ne permet pas une généralisation suffisante à partir de quelques exemples fournis en entrée.
La classification heuristique pourrait permettre, grâce à des règles de classification abstraites, d'obtenir de bons résultats, mais ce type de raisonnement est assez proche du raisonnement à base de cas, sans en posséder toute la puissance, et n'a donc pas été implémenté.
Une autre possibilité serait d'utiliser la technologie dite de généralisation à partir d'explication (*Explanation Based-Generalization* [Cornuejols et Miclet 02]) qui est une méthode de généralisation fonctionnant à partir d'un seul ou très peu d'exemples, grâce à des connaissances (une théorie du domaine généralement sous le formalisme de la logique du 1er ordre). Cela pourrait constituer un travail futur à explorer.

Chapitre 7

COMPARAISON DES RAISONNEMENTS DE L'HUMAIN ET DE L'AGENT ARTIFICIEL

Il existe deux façons de comparer les raisonnements, le mode En ligne (les deux opérateurs sont face au même test simultanément et à chaque fois que l'opérateur humain effectue une action, l'agent artificiel indique quelle action il aurait fait, sans la réaliser, et il s'agit ensuite d'effectuer une comparaison action par action) et le mode Hors ligne (les deux opérateurs effectuent le test de manière indépendante, et il s'agit ensuite d'effectuer une comparaison globale de deux séquences d'actions) [Ponsa et Catala 99 (b)].
La comparaison doit tenir compte des caractéristiques de l'agent artificiel. Mais il s'agit également de définir une distance qui soit générique, c'est-à-dire qui s'applique à tous les agents artificiels.
En outre, divers critères sont utilisés pour la comparaison : la durée du test, l'évitement ou non de situations de danger (c'est-à-dire ici le nombre de situations de débordement de réservoir ou le volume total débordé), et bien sûr la séquence d'actions. Ce dernier point est le plus délicat à prendre en compte, et peut être mis en œuvre par une comparaison locale, action par action, ou bien de manière plus globale, par exemple en analysant les actions de l'opérateur humain par rapport au comportement, en terme d'intentions, de l'agent artificiel.

7.1. MODE HORS LIGNE

7.1.1. Comparaison globale

7.1.1.1. Généralités

L'opérateur humain et l'agent artificiel effectuent le test de manière totalement indépendante. Dans ce cas, une comparaison directe entre les actions de l'opérateur humain et de l'agent artificiel est extrêmement délicate car il s'agit de comparer deux séquences, ce qui pose de nombreux problèmes :

- les actions (ouvrir/fermer une vanne) peuvent être effectuées à n'importe quel instant,
- la durée du test peut être différente, et donc le nombre d'actions également,
- la séquence d'action peut être très différente (c'est-à-dire l'utilisation de stratégies très différentes) et conduire à la même réussite de supervision,
- l'ordre des actions peut différer, bien que les stratégies soient les mêmes,
- le pas de temps de calcul de l'agent artificiel a une influence importante sur la comparaison.

Il s'agit de savoir si le raisonnement de l'humain peut être considéré comme appartenant ou non à telle ou telle classe d'agents artificiels. On essaie donc de détecter, au sein des séquences d'actions effectuées par l'humain, des régularités qui soient propres à une classe donnée d'agents artificiels, ce qui nécessite également un calage des paramètres de cette classe d'agents. Plutôt que de faire une comparaison au niveau des actions, il semble préférable de passer à un niveau plus abstrait et de comparer les buts de l'opérateur humain et de l'agent artificiel. Il faut donc réussir à *reconnaître les buts* de l'opérateur humain à partir de ses actions sur les vannes. Les travaux de P. Ponsa [Ponsa *et al.* 02], relatif à ce même problème, sont basés sur la définition de modèles (patterns) comportementaux : le but "minimiser le temps" est lié à la notion de *chemins à ouvrir* entre le réservoir source et le réservoir puit, et le but "minimiser les débordements" est lié au *contrôle des niveaux d'eau* des réservoirs. L'auteur définit ainsi 7 modèles comportementaux (notés M_i i = 1 à 7), pour la gestion de configurations de réseaux type 3, 4 ou 5 (voir figure 6). Par exemple le modèle M_1 est : un seul chemin est ouvert (toutes les autres vannes sont fermées) et un seul réservoir intermédiaire (celui appartenant au chemin ouvert) est contrôlé, le modèle M_2 est : deux chemins sont ouverts (toutes les autres vannes sont fermées) et deux réservoirs intermédiaires (ceux appartenant aux chemins ouverts) sont contrôlés, et le modèle M_6 est : tous les chemins sont ouverts et tous les réservoirs intermédiaires sont contrôlés. Dès lors, une stratégie d'un opérateur est définie comme une séquence de modèles comportementaux. Les modèles relatifs au contrôle d'un seul chemin sont effectivement intéressants car ils dénotent bien un type de comportement, mais les modèles relatifs au contrôle de plusieurs chemins et réservoirs ne sont pas assez détaillés. Par exemple, il y a un grand nombre de façons de gérer deux réservoirs simultanément, comme en témoignent les différents cas de l'agent CBR. Or l'étude de Ponsa ne définit que deux modèles (M_2 et M_4) relatifs à la gestion de deux réservoirs. En outre, le lien de causalité entre un chemin ouvert et le contrôle d'un réservoir n'est pas clair car on peut très bien contrôler un réservoir sans qu'il n'y ait de chemin ouvert le traversant, par exemple si une vanne en amont du réservoir est ouverte et les vannes en aval fermées (ou le contraire). L'auteur n'a pas pris en compte ce type de situations pourtant très fréquentes.

Nous avons souhaité développer une approche différente (figure 45) : ainsi, la comparaison des raisonnements peut se faire de manière indirecte en passant par la reconnaissance des buts : dans une situation concrète donnée, un opérateur humain effectue une action, liée à un but. Nous avons choisi de développer une approche de reconnaissance de but à base de logique floue, qui évite donc la comparaison d'actions, et dans laquelle on tente de découvrir l'intention qui est associée à chaque action de l'opérateur humain, à condition que cette intention fasse partie de l'ensemble des buts possibles de l'agent artificiel. En effet, pour chaque agent artificiel, dans une situation générique donnée, il cherche à appliquer une stratégie (un cas par exemple), liée à un but.

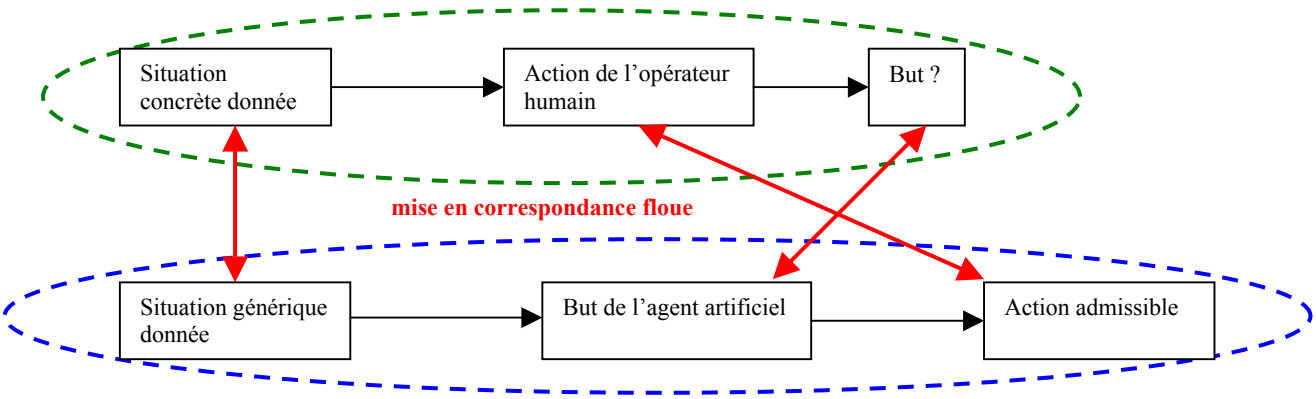


Figure 45 : Principe de la reconnaissance de l'intention associée à une action de l'opérateur humain

7.1.1.2. Reconnaissance floue de l'intention associée à une action

7.1.1.2.1. Intentions et buts

Il paraît nécessaire de faire ici référence à des auteurs ayant étudié de près les concepts de but et d'intention, afin de placer notre approche dans ce contexte.

Selon [Minsky 85], on ne peut pas apprendre le sens de quelque chose uniquement en le reliant à des noms. Chaque mot/idée doit associer des causes/explications, des objectifs/buts, des actions effectives permettant d'atteindre les objectifs. Par exemple, une action doit être associée à des conditions préalables à sa réalisation (contexte, ressources, …), à des causes possibles (pourquoi faire cette action, dans quelle intention ?), et à des effets (l'état résultant de la réalisation effective de l'action). En effet, un mot n'a de sens que parce qu'il est relié à d'autres concepts. Par exemple, tout objet (notamment un objet physique) comporte au moins deux types de descriptions : structurale et fonctionnelle, ainsi que les liens entre les deux descriptions. L'auteur affirme en outre que nos connaissances sont "orientées but" (toute connaissance n'est utile que si elle est reliée à un but, une utilisation), et avance l'idée selon laquelle l'avenir de la programmation informatique est la "programmation orientée but" (*goal oriented programing*).
D'autre part, le concept d'intention comporte deux conditions d'existence : la persistance, et une image ou une description de l'état voulu. On peut ajouter une autre condition qui est la ressource ou la capacité pour atteindre ce but. D'un point de vue législatif, l'intention est liée à la notion de responsabilité. De plus, l'intention est associée au contrôle de soi. En effet, parfois nous nous sentons obligé d'agir d'une certaine façon, de manière irrésistible, comme si une force nous oppressait, et il s'agit là plus d'une réaction que d'une action intentionnelle.

[Minsky 03] affirme qu'un humain est presque toujours en train de poursuivre des buts : même lorsqu'il n'a pas l'impression d'avoir un but bien défini, il se sent obligé de penser ou faire quelque chose. En outre, il serait très difficile de montrer que certaines activités mentales ont lieu sans motivation, car certaines de nos activités mentales possèdent la fonction de cacher les buts de certaines autres. D'autre part, nous réagissons souvent non pas aux situations elles-mêmes, mais plutôt aux buts sous-jacents que ces situations mettent en évidence.

Selon [Schank 99], chaque aspect du comportement humain implique la poursuite de buts. Parfois, ils sont assez simples, comme se brosser les dents, parfois ils sont plutôt inconscients, comme rechercher en mémoire une expérience similaire lorsque l'on est face à une situation nouvelle, et parfois ils sont complexes, comme la réalisation d'un logiciel informatique. En outre, on ne peut comprendre une histoire, ou ce que quelqu'un nous dit, sans avoir accès aux buts sous jacents et à leur interaction. Nous cherchons à apprendre uniquement pour nous aider à poursuivre des buts, nous cherchons à comprendre quelque chose dans le but d'apprendre à partir de cette compréhension. Il s'agit donc dans tous les cas d'activités dirigées par les buts. Schank introduit également le concept de scénario dirigé par les buts (*goal-based scenario*) qui concerne donc l'apprentissage d'une capacité (*skill*) sous la forme d'un scénario en vue d'accomplir un but.

[Barsalou 02] affirme que la fonction d'une entité dépend de quatre domaines : la perspective intentionnelle (pourquoi un agent souhaite avoir accès aux connaissances liées à la fonction, et quel est le point de vue associé - le sien, celui d'un autre, celui de l'inventeur -), l'histoire de l'entité (le rôle standard, le processus d'invention, le processus de fabrication et l'histoire

de l'utilisation d'une instance spécifique), l'environnement physique qui comprend l'entité (une structure physique et des buts internes), des agents externes, et les objets et lieux environnants centraux pour comprendre la fonction, et la séquence d'événements (les comportements de l'entité) produisant un résultat (le but atteint). Puis l'auteur étudie les conditions sous lesquelles la fonction standard peut être réalisée ou non, et une fonction non standard peut être réalisée ou non.

Enfin, Ram et Leake pensent que tout individu apprend dans le but d'améliorer ses performances dans une tâche [Ram et Leake 94], rejoignant ainsi le point de vue de Schank. Pour cela, il doit décider de quelle information il a besoin, la trouver (ce qui nécessite la formation de sous-buts et la planifications d'actions pour les atteindre), quelles sont les stratégies alternatives en fonction du contexte, puis ensuite reformuler ou modifier ce qui était déjà connu, par la formation de généralisations ou une ré-organisation de la mémoire.

7.1.1.2.2. Approche proposée

Tout d'abord, remarquons qu'une comparaison de deux actions qui ne tiendrait pas compte des intentions associées mais uniquement des influences des actions sur le système (par exemple en terme d'accroissement ou d'abaissement des niveaux d'eau, de manière plus ou moins directe) pose le problème suivant : il est très difficile de comparer deux actions lorsqu'elles sont détachées de leur but (par exemple le but d'une action peut être de réduire une alarme, ou bien d'accélérer le processus, …).

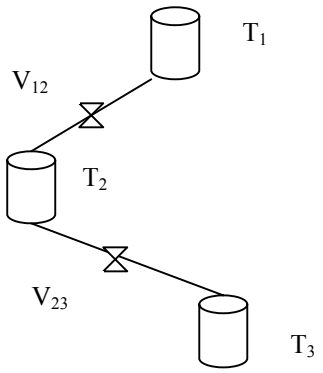


Figure 46 : Distance entre deux actions

En effet, supposons que l'action a_1 consiste à fermer V_{12} et l'action a_2 consiste à ouvrir V_{23} (voir figure 46). Si les deux actions servent à réduire une alarme du réservoir T_2, alors elles sont "proches". Mais si a_1 sert à remplir T_1 et a_2 sert à remplir T_3, alors ces deux actions sont liées à des stratégies différentes et donc sont "distantes". On voit donc clairement que la comparaison de deux actions passe par leur interprétation en terme de but/intention.

Ensuite, remarquons que la flexibilité est un atout important lorsqu'il s'agit de comparer les raisonnements d'un opérateur humain et d'une machine. Si les paramètres de l'agent artificiel (le seuil d'alarme ΔV^{alarme}, les pondérations des sous-buts p_1, p_2, p_3, p_4 de l'opérateur qualitatif, les zones optimales ΔV_ZO , …) sont fixés, il s'en suit un manque de flexibilité du système. Par exemple, si le seuil en volume ΔV^{alarme} est fixe, et si l'opérateur humain considère un seuil d'alarme légèrement au-dessus ou en dessous, alors certaines de ses actions seront considérées comme non admissibles, ce qui ne semble pas judicieux. En outre, un opérateur humain considérera un seuil d'alarme grossier (une zone plutôt qu'une valeur fixe)

puisqu'il est incapable de repérer précisément les niveaux. Il serait donc préférable que l'opérateur artificiel s'adapte à l'humain par le biais de ces paramètres. Bien sûr, cette adaptation doit posséder certaines limites : si par exemple une action de l'opérateur humain afin de réduire une alarme d'un réservoir est effectuée alors que le niveau d'eau dans le réservoir est très éloigné du seuil d'alarme de l'agent artificiel, alors cette action de l'opérateur humain ne peut pas être considérée comme proche du comportement de l'agent artificiel.

Analysons plus précisément la figure 45 : toute action humaine est dirigée par un but (7.1.1.2.1.). Ainsi, dans notre étude, une action peut donc être vue avec l'intention de gérer une situation. Nous allons chercher ici à analyser et interpréter chacune des actions (**H_Act**) de l'opérateur humain dans le cadre du comportement de l'agent artificiel (**AA**), c'est-à-dire de voir s'il est possible de relier une action de l'opérateur humain à un but prédéfini de l'agent artificiel.
L'étude préalable du comportement de l'AA a permis d'exhiber des classes d'intention associées chacune à une classe de situation (ou situation générique), c'est-à-dire que dans telle situation, il agit avec telle intention : l'opérateur qualitatif dans une situation de non alarme cherche à accélérer le processus et dans une situation d'alarme cherche à la réduire, et l'opérateur à base de cas reconnaît une situation puis agit dans un but précis (par exemple il reconnaît une situation comportant une conduite sans vanne et éventuellement agit pour éviter un débordement futur).
En outre, à chaque but/intention de l'agent *AA* correspond un ensemble d'actions admissibles (**Adm_Act**), dont une est choisie selon divers critères. Une condition nécessaire est donc l'appartenance de l'action *H_Act* à cet ensemble *Adm_Act*.
Il s'agit alors d'évaluer la plausibilité de l'action *H_Act* de l'opérateur humain pour chaque classe d'intention, car même dans le cadre du comportement de l'agent artificiel, il existe plusieurs interprétations possibles d'une action. Cette évaluation est réalisée à partir des concepts de base de la logique floue [Bouchon-Meunier 95] [Gacôgne 97].

Nous allons maintenant analyser le cas où l'opérateur humain agit (points 1 ci-dessous) et celui où il n'agit pas (point 2 ci-dessous) :

1. Intention correspondant à une action (H_Act ≠ No_Act)

Lorsque l'opérateur humain agit, il s'agit de savoir quelle situation générique *Sj* (*Sj* ∈ {alarme, non alarme, hors zone optimale, anticipation débordement}) il a cherché à gérer. L'intention associée à l'action *int (H_Act)* est liée aux degrés d'appartenance μ_{Sj_Ti} aux situations *Sj* (*j* = 1 à 4). Ainsi, nous allons appliquer le principe général suivant qui sera cependant adapté aux spécificités de chacune des situations *Sj* :

Pour chaque situation *Sj* (*j* = 1 à 4) :
Pour chaque réservoir intermédiaire T_i :
Si ***(H_Act ≠ No_Act et Adm_Act_Sj (Ti) ≠ Ø et H_Act ∈ Adm_Act_Sj (Ti))***
Alors *H_Act* peut être interprétée avec l'intention de gérer la situation *Sj* relativement au réservoir T_i avec un degré d'intention (ou de plausibilité) fonction du niveau d'eau $h_i(t)$ dans le réservoir T_i :
$\mathbf{int_{Sj}(H\_Act, T_i) = \mu_{Sj\_Ti}(h_i)}$
Sinon $\mathbf{int_{AA\_Sj}(H\_Act, T_i) = 0}$
Bilan sur l'ensemble du réseau à un instant *t* :
$\mathbf{int_{Sj}(H\_Act) = max_{\ i}(int_{Sj}(H\_Act, T_i))}$ car on retient l'interprétation la plus plausible.

No_Act représente l'absence d'action concrète et *Adm_Act_Sj* (T_i) représente l'ensemble des actions admissibles afin de gérer la situation *Sj* relativement au réservoir T_i .

1.1. Intention correspondant à une situation d'alarme (situation S1 = Al)

On examine ici si l'action a été établie dans l'intention de réduire l'alarme d'un réservoir. On rappelle que la notion d'alarme est liée à la notion de risque de débordement.

Pour chaque réservoir intermédiaire T_i :
Si ***(H_Act ≠ No_Act et Adm_Act_Al*** (T_i) ***≠ Ø et H_Act ∈ Adm_Act_Al*** (T_i) ***)***
Alors *H_Act* peut être interprétée avec l'intention de réduire l'alarme avec un degré d'intention (ou de plausibilité) fonction du niveau d'eau $h_i(t)$ dans le réservoir T_i et mesuré par : $int_{S1}(H\_Act, T_i) = \mu_{Al\_Ti}(h_i)$
Sinon $int_{S1}(H\_Act, T_i) = 0$
Bilan sur l'ensemble du réseau à un instant *t* :
$int_{S1}(H\_Act) = max_i(int_{S1}(H\_Act, T_i))$ car on retient l'interprétation la plus plausible.

No_Act représente l'absence d'action concrète et *Adm_Act_Al* (T_i) représente l'ensemble des actions admissibles en supposant le réservoir T_i en état d'alarme (voir 3.2.2.).
Du fait que l'opérateur humain (**HA**) n'a pas forcément la même notion d'alarme que l'agent artificiel, nous modélisons une zone floue fonction de la hauteur d'eau $h_i(t)$ du réservoir T_i.

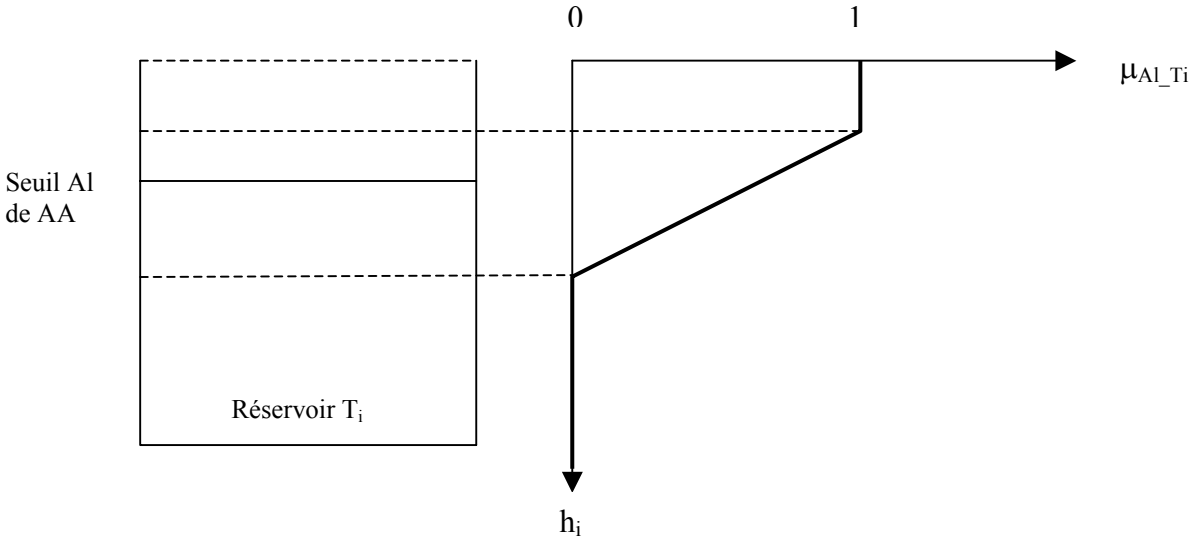


Figure 47 : Degré d'intention correspondant à une situation d'alarme d'un réservoir en fonction du niveau d'eau

Si *H_Act* ∈ *Adm_Act_Al* (T_i) et $\mu_{Al\_Ti}(h_i)$ est strictement positif, cela signifie que l'action de l'humain peut être interprétée comme réduisant l'alarme, et l'adéquation avec le (ou encore la plausibilité par rapport au) comportement de l'agent artificiel est mesurée par $\mu_{Al\_Ti}(h_i)$. Plus cette valeur est proche de 1, plus l'interprétation de cette action selon cette intention est cohérente avec le comportement de l'agent artificiel.

Le choix d'une zone floue englobant le seuil d'alarme fixe de l'agent artificiel *AA* est motivé par la raison suivante : si le niveau d'eau dans le réservoir T_i est légèrement supérieur ou légèrement inférieur au seuil d'alarme de l'agent *AA*, et si l'action *H_Act* appartient à l'ensemble des actions admissible dans une situation d'alarme du réservoir T_i et appartient

également à l'ensemble des actions admissibles dans une situation sans alarme (par exemple ouvrir une vanne en aval de T_i), alors il est difficile de trancher quant à l'intention de l'opérateur humain. Celui-ci peut tout autant avoir effectué cette action pour réduire l'alarme de T_i ou bien pour accélérer le processus dans une situation sans alarme. La zone floue ainsi définie permet de refléter cette indétermination.

1.2. Intention correspondant à une situation sans alarme (situation S2 = No_Al)

On examine si l'action a été établie dans l'intention d'accélérer le processus (c'est-à-dire ouvrir une vanne), et si chaque réservoir peut être vu comme étant dans un état de non alarme.

Pour le système global (l'ensemble du réseau) :
Si ***(H_Act ≠ No_Act et Adm_Act_No_Al ≠ Ø et H_Act ∈ Adm_Act_No_Al)***
Alors *H_Act* peut être interprétée comme accélérant le processus dans une situation sans alarme avec un degré d'intention fonction des niveaux d'eau $h_i(t)$ dans chacun des réservoirs T_i et mesuré par *:* $\boldsymbol{int_{S2}(H\_Act) = \min_i(\mu_{No\_Al\_Ti}(h_i))}$
Sinon $\boldsymbol{int_{S2}(H\_Act) = 0}$

Adm_Act_No_Al représente l'ensemble des actions admissibles en supposant qu'aucun réservoir n'est dans un état d'alarme (voir 3.2.2.).

On se trouve dans une situation de non alarme si *tous* les réservoirs sont dans une situation de non alarme, et le minimum reflète bien la conjonction "et" (on pourrait également utiliser des opérateurs d'agrégation floue de type norme triangulaire ou t-norme afin de définir l'intersection de sous-ensembles flous [Bouchon-Meunier 95]).

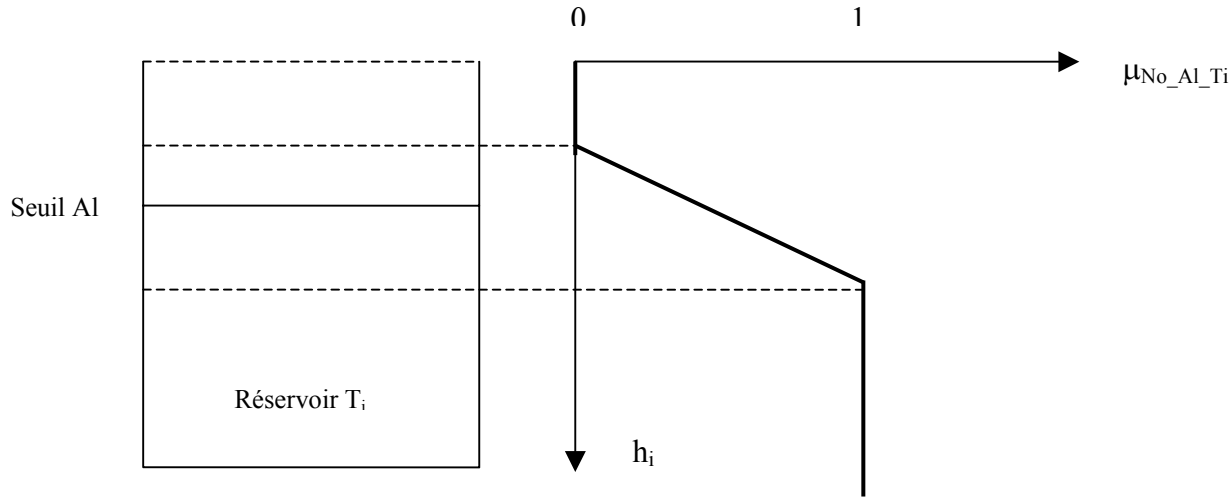


Figure 48 : Degré d'intention correspondant à une situation sans alarme en fonction du niveau d'eau

On peut poser, pour des raisons de cohérence et de symétrie : $\mu_{No\_Al\_Ti}(h_i) = 1 - \mu_{Al\_Ti}(h_i)$.

1.3. Intention correspondant à une situation hors zone optimale (stratégie de minimisation du temps)(situation S3= Hors_ZO)

On rappelle qu'il s'agit d'une stratégie visant à conserver le niveau d'eau de certains réservoirs dans une certaine zone, dite zone optimale (**ZO**), afin que ces réservoirs se vidangent selon la même durée, qui est minimale globalement (voir 5.2.3. index n° 9).

Pour chaque réservoir intermédiaire *Ti* pour lesquels l'agent artificiel applique cette stratégie (ensemble ***I***), deux cas se présentent, selon que l'on doit vidanger ou remplir le réservoir :

Si ***(H_Act ≠ No_Act et Adm_Act_Hors_ZO_Vidanger (T_i) ≠ Ø et H_Act ∈ Adm_Act_Hors_ZO_Vidanger (T_i))***

Alors *H_Act* peut être interprétée comme permettant de vidanger T_i afin qu'il se retrouve dans sa zone optimale avec un degré d'intention fonction du niveau d'eau dans le réservoir T_i et mesuré par *:* $\mathbf{int_{S3}(H\_Act, T_i) = \mu_{Hors\_ZO\_Vidanger\_Ti}(h_i)}$

Sinon si ***(H_Act ≠ No_Act et Adm_Act_Hors_ZO_Remplir (T_i) ≠ Ø et H_Act ∈ Adm_Act_Hors_ZO_Remplir (T_i))***

Alors *H_Act* peut être interprétée comme permettant de remplir T_i afin qu'il se retrouve dans sa zone optimale avec un degré d'intention fonction du niveau d'eau dans le réservoir T_i et mesuré par : $\mathbf{int_{S3}(H\_Act, T_i) = \mu_{Hors\_ZO\_Remplir\_Ti}(h_i)}$

Sinon $\mathbf{int_{S3}(H\_Act, T_i) = 0}$

Bilan sur l'ensemble du réseau à un instant *t* :

$\mathbf{int_{S3}(H\_Act) = max_{i \in I}(int_{S3}(H\_Act, T_i))}$**.** On retient l'interprétation la plus plausible.

Adm_Act_Hors_ZO_Vidanger (T_i) est l'ensemble des actions admissibles en supposant le niveau d'eau du réservoir T_i au dessus de la zone optimale (voir 5.2.3 : cas 9-1 partie Action). *Adm_Act_Hors_ZO_Remplir* (T_i) représente l'ensemble des actions admissibles en supposant le niveau d'eau du réservoir T_i en dessous de la zone optimale (voir 5.2.3 : cas n° 9-0 partie Action).

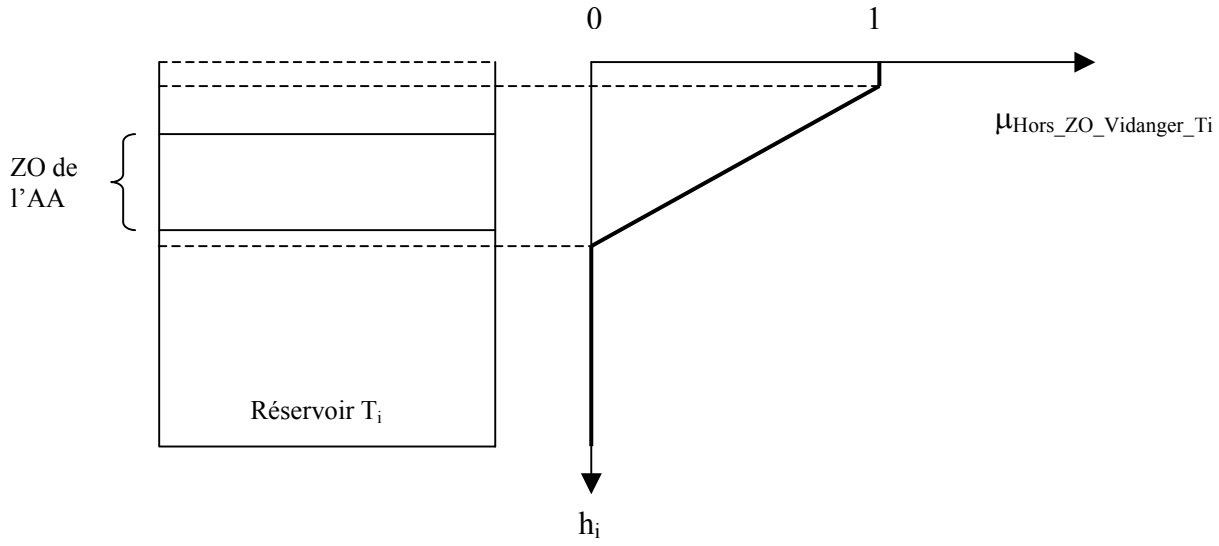


Figure 49 : Degré d'intention correspondant à une situation hors zone optimale où il faut vidanger un réservoir en fonction du niveau d'eau

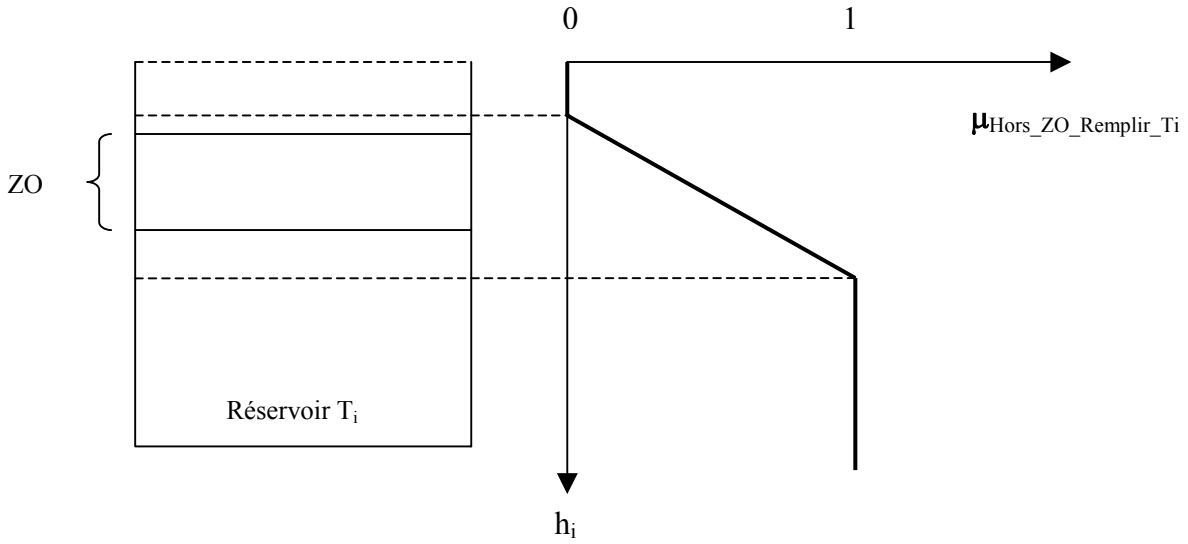


Figure 50 : Degré d'intention correspondant à une situation hors zone optimale où il faut remplir un réservoir en fonction du niveau d'eau

On peut poser, pour des raisons de symétrie et de cohérence :
$\mu_{Hors\_ZO\_Remplir\_Ti}(h_i) + \mu_{Hors\_ZO\_Vidanger\_Ti}(h_i) = 1$.

1.4. Intention correspondant à une situation d'anticipation de débordement (situation S4 = Ant_Déb)

Pour chaque couple de réservoirs T_i et T_j reliés par une conduite sans vanne (ensemble ***IJ***),
Si ***(H_Act ≠ No_Act et Adm_Act_Ant_Déb(T_i, T_j) ≠ Ø et H_Act ∈ Adm_Act_Ant_Déb(T_i, T_j))***
Alors *H_Act* peut être interprétée comme permettant d'anticiper le débordement de T_i avec un degré d'intention fonction du niveau d'eau dans les réservoirs T_i et T_j et mesuré par : $int_{S4}(H\_Act, T_i, T_j) = \mu_{Ant\_Déb\_Ti\_Tj}(h_i, h_j)$
Sinon $int_{S4}(H\_Act, T_i, T_j) = 0$
Bilan sur l'ensemble du réseau à un instant *t* :
$int_{S4}(H\_Act) = max_{(i,j) \in IJ}(int_{S4}(H\_Act, T_i, T_j))$**.** On retient l'interprétation la plus plausible.

Adm_Act_Ant_Déb(T_i, T_j) représente l'ensemble des actions admissibles correspondant à cette situation (voir index n° 8 et cas n° 8 : partie Action).

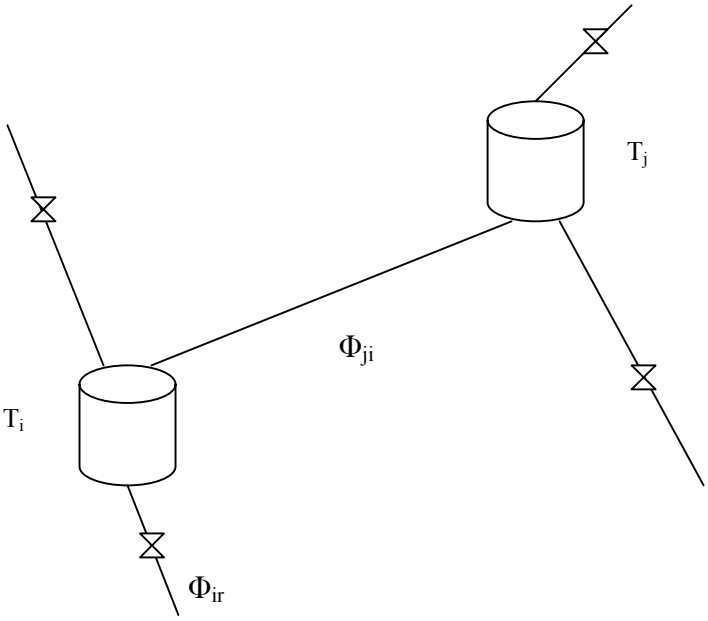


Figure 51 : Situation d'anticipation de débordement

Pour le calcul de $\mu_{Ant\_Déb\_Ti\_Tj}$, remarquons tout d'abord que l'*AA* applique le cas correspondant à cette situation si $(V_i + \alpha.V_j) \geq V_i^{max}$ (on note $V_i = V_i(t)$ = volume de T_i à l'instant t et $\alpha = f(\Phi_{ji}, \Phi_{ir}, ...) \in]0;1[$: voir annexe 9). Nous allons alors définir le degré d'intention $\mu_{Ant\_Déb\_Ti\_Tj}$ en décrivant le nombre α de manière floue. En effet, la somme des volumes $(V_i + \alpha.V_j)$ correspond à la somme $(V_i + V_j)$ diminuée de la somme (volume sortant de T_i pendant la durée de vidange de T_j + volume sortant de T_j par une conduite autre que $j \rightarrow i$ pendant cette même durée) (voir annexe 9). C'est essentiellement le second terme (la somme des volumes sortant) qui est délicat à apprécier pour un opérateur humain. Cette somme est reflétée par le terme α.

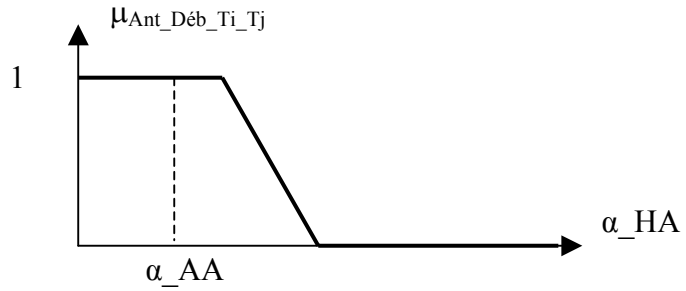


Figure 52 : Degré d'intention correspondant à une situation d'anticipation de débordement

α_AA est le nombre α correspondant à l'agent *AA*, c'est-à-dire $\alpha\_AA = f(\Phi_{ji}, \Phi_{ir}, ...)$ qui est une valeur fixe pour une configuration donnée et appartient à] 0 ; 1 [(voir annexe 9).
α_HA est le nombre α correspondant à l'opérateur humain *HA* en se plaçant à la limite au niveau du test précédent, c'est-à-dire $(V_i + \alpha\_HA.V_j) = V_i^{max}$, ce qui permet ensuite de calculer $\mu_{Ant\_Déb\_Ti\_Tj}$ (voir figure 52).
Si $\alpha\_HA \leq \alpha\_AA$ alors par définition le test de l'agent *AA* est vrai ($(V_i + \alpha\_AA.V_j) \geq V_i^{max}$), donc l'action de l'opérateur humain peut être interprétée avec l'intention d'éviter un débordement futur (ou de le limiter s'il est inévitable) et le degré d'intention est maximum, c'est-à-dire 1. Par contre, plus α_H_Act augmente, moins l'action peut être interprétée avec cette intention, d'où un degré de plausibilité qui décroît.

Si $\Phi_{ji} < \Phi_{ir}$
Alors $\mu_{Ant\_Déb\_Ti\_Tj} = 0$;
Sinon
$\alpha\_AA = f(\Phi_{ji}, \Phi_{ir}, ...)$;
$\alpha\_HA = (V_i^{max} - V_i)/ V_j$;
$\mu_{Ant\_Déb\_Ti\_Tj} = f(\alpha\_HA, \alpha\_AA)$;

1.5. Situation hors comportement AA (situation S5)

Si l'opérateur humain a effectué une action ($H\_Act \neq No\_Act$) et si cette action H_Act n'est pas dans au moins une des quatre classes d'intention précédentes, c'est-à-dire que les conditions associées à ces classes d'intentions ($Adm\_Act\_Sj \neq \emptyset$ et $H\_Act \in Adm\_Act\_Sj$) ne sont pas vérifiées, alors H_Act ne peut être interprétée comme appartenant au comportement de l'agent artificiel. Donc $\boldsymbol{int_{S5}(H\_Act) = 0}$.

2. Explication d'une absence d'action (*H_Act = No_Act*)

Nous sommes désormais dans la situation où l'opérateur humain *HA* n'a pas agi et nous cherchons à expliquer cette "non action". Les explications sont fournies par l'analyse du comportement de l'agent artificiel *AA* : celui-ci cherche toujours à appliquer un cas en mémoire, sinon à appliquer les stratégies liées au raisonnement qualitatif, donc cherche à effectuer une action. Ceci est représenté par les quatre situations précédentes *Sj* (alarme, non alarme, hors zone optimale, anticipation débordement). Si cela n'est pas possible, alors il n'agit pas.
Il s'agit également de savoir si la "non action" de l'opérateur humain est cohérente avec le comportement de l'agent *AA* lorsque ce dernier n'agit pas. Nous allons donc mesurer l'adéquation *int (H_Act)* avec le comportement de l'agent artificiel *AA*. Il est impératif de distinguer le cas "action" du cas "non action" car dans le cas "non action" il faut tenir compte d'une situation et de sa négation en même temps, et l'interprétation de la "non action" sera la plus plausible parmi les deux interprétations selon la situation *Sj* et selon la situation $\neg Sj$ (la négation de *Sj*). Nous considérons donc les situations $S'j = (Sj ; \neg Sj)$ (avec $Sj \in$ {alarme, hors zone optimale, anticipation débordement})

Pour chaque situation $S'j$ (j = 1 à 3) :
Pour chaque réservoir intermédiaire T_i :
Si ***(No_Action_Sj_Ti et No_Action_¬Sj_Ti)***
Alors il n'y a aucune action possible pour gérer *Sj* ni $\neg Sj$ relativement à T_i, et on choisit donc l'interprétation la plus plausible entre : on n'a pas agi sur *Ti* car on était dans la situation *Sj* mais il n'y avait pas d'action possible ou : on n'a pas agi sur *Ti* car on était dans la situation $\neg Sj$ mais il n'y avait pas d'action possible.
Soit : $\boldsymbol{int_{S'j}(H\_Act, T_i) = max(\mu_{Sj\_Ti}(h_i), (\mu_{\neg Sj\_Ti}(h_i))}$
Sinon si ***(No_Action_Sj_Ti)*** alors $\boldsymbol{int_{S'j}(H\_Act, T_i) = \mu_{Sj\_Ti}(h_i)}$
Sinon si ***(No_Action_¬Sj_Ti)*** alors $\boldsymbol{int_{S'j}(H\_Act, T_i) = \mu_{\neg Sj\_Ti}(h_i)}$
Sinon $\boldsymbol{int_{S'j}(H\_Act, T_i) = 0}$
Bilan sur l'ensemble du réseau à un instant *t* :
$\boldsymbol{int_{S'j}(H\_Act) = min_i (int_{S'j}(H\_Act, T_i))}$**.** L'opérateur humain n'a pas agi car il ne pouvait agir sur aucun réservoir.

No_Action_Sj_Ti est un test binaire vrai si, en supposant qu'on est dans la situation *Sj*, alors il n'y a aucune action possible pour gérer *Sj* relativement au réservoir T_i.

Nous allons appliquer ce principe général en l'adaptant aux spécificités des situations *S'j* (*j* = 1 à 3) auxquelles l'agent *AA* se trouve confronté.

Finalement, la "non action" se produit si *conjointement*, dans les trois classes de situations *S'j* (*j* = 1 à 3), aucune action n'est possible (voir point 3. Récapitulatif ci-dessous).

Remarque : si un ensemble *Adm_Act_Sj* = *Ø* alors *H_Act (= No_Act)* $\in$ *Adm_Act_Sj*.

2.1. Situation d'alarme ou non (situation S'1 = (Al ; No_Al))

Si pour chaque réservoir intermédiaire, l'*HA* juge soit que ce réservoir est dans un état de non alarme et qu'il n'a pas d'action à faire localement pour accélérer le processus au niveau de ce réservoir, soit que ce réservoir est dans un état d'alarme et qu'il n'a pas d'action afin de réduire cette alarme, alors la non action de l'*HA* est cohérente avec le comportement de l'*AA*. De façon plus formelle, pour chaque T_i intermédiaire, nous calculons le degré d'intention associé à la "non action" correspondant à la situation d'alarme ou non : $int_{S'1}(H\_Act, T_i)$.

Pour chaque T_i intermédiaire,
Si ***(No_Action_No_Al(T_i) et No_Action_Al(T_i))***
Alors $\mathbf{int_{S'1}(H\_Act, T_i) = max(\mu_{No\_Al\_Ti}(h_i), \mu_{Al\_Ti}(h_i))}$
Sinon si ***No_Action _No_Al (T_i)***
Alors $\mathbf{int_{S'1}(H\_Act, T_i) = \mu_{No\_Al\_Ti}(h_i)}$ (*HA* n'a pas agi sur T_i car il a considéré que celui-ci n'était pas en alarme et qu'il n'y avait pas d'action pour accélérer le processus localement)
Sinon si ***No_Action _Al (T_i)***
Alors $\mathbf{int_{S'1}(H\_Act, T_i) = \mu_{Al\_Ti}(h_i)}$ (*HA* n'a pas agi sur T_i car il a considéré que celui-ci était en alarme et qu'il n'y avait pas d'action pour réduire cette alarme)
Sinon $\mathbf{int_{S'1}(H\_Act, T_i) = 0}$
Bilan sur l'ensemble du réseau (à un instant donné) :
$\mathbf{int_{S'1}(H\_Act) = min_i\ (int_{S'1}(H\_Act, T_i))}$**.** En effet, si l'opérateur humain n'agit pas, c'est parce qu'il n'y avait pas d'action à faire sur chaque réservoir, et la conjonction est traduite par le minimum.

No_Action_No_Al (T_i) est un test binaire qui est vrai s'il n'y a aucune action à faire en considérant le réservoir T_i non en alarme, c'est-à-dire si le réservoir T_i est relié immédiatement à des conduites sans vanne ou avec vannes ouvertes (ce qui correspond en quelque sorte à une condition de non action en cas de non alarme locale au réservoir T_i), ou bien il existe dans cet ensemble des vannes fermées mais qui soit doivent le rester (les actions consistant à ouvrir ces vannes sont dans la liste *Tabou*, ou bien il s'agit de besoins liés à la stratégie de minimisation de la durée), soit sont reliées en amont à un réservoir vide. Ce test n'est donc fonction que des positions des vannes.

No_Action_Al (T_i) est un test binaire qui est vrai s'il n'y a aucune action à faire en considérant le réservoir T_i en alarme, c'est-à-dire si l'ensemble des actions admissibles pour réduire cette alarme est vide (*Adm_Act_Al* (T_i) = *Ø*) ou bien les actions de cet ensemble correspondent soit à des fermetures de vannes situées en amont de T_i mais reliées à un autre réservoir vide, soit à

des vannes fermées en aval de T_i et qui doivent le rester pour les besoins de la stratégie de minimisation de la durée. Ce test n'est donc fonction que des positions des vannes.

2.2. Situation hors zone optimale ou non (situation S'2 = (ZO ; Hors_ZO))

Si pour chaque réservoir pour lesquels l'*AA* applique la stratégie de minimisation de durée, soit l'*HA* juge que ce réservoir se trouve dans sa zone optimale (ce qui est donc une raison de ne pas agir - voir remarque ci-dessous), soit ne s'y trouve pas auquel cas l'*HA* juge que le réservoir est au dessus de la zone mais il n'y a pas d'action à faire pour vidanger, ou le réservoir est en dessous de la zone mais il n'y a pas d'action à faire pour remplir, alors la "non action" de l'*HA* est cohérente avec le comportement de l'*AA*. De façon plus formelle :

Pour chaque T_i concerné par la stratégie de minimisation de la durée (ensemble ***I***), nous calculons le degré d'intention associé à la "non action" correspondant à la situation hors zone optimale ou non : $int_{S'2}(H\_Act, T_i)$.

Si $\boldsymbol{int_{ZO\_Ti}(h_i) \geq max\ (int_{Hors\_ZO\_Vidanger\_Ti}(h_i), \mu_{Hors\_ZO\_Remplir\_Ti}(h_i))}$
Alors $\boldsymbol{int_{S'2}(H\_Act, T_i) = \mu_{ZO\_Ti}(h_i)}$ (*HA* n'a pas agi sur T_i car il a considéré que celui-ci était dans sa zone optimale, donc par définition il n'y a pas d'action à faire : voir remarque 2)

Sinon Si ***Adm_Act_Hors_ZO_Vidanger = Ø et Adm_Act_Hors_ZO_Remplir = Ø***
Alors $\boldsymbol{int_{S'2}(H\_Act, T_i) = max\ (\mu_{Hors\_ZO\_Vidanger\_Ti}(h_i), \mu_{Hors\_ZO\_Remplir\_Ti}(h_i))}$
HA n'a pas agi sur T_i car il a considéré que T_i était hors de sa zone optimale, qu'il fallait le vidanger (ou remplir en fonction du max) mais qu'il n'y avait pas d'action à faire.

Sinon Si ***Adm_Act_Hors_ZO_Vidanger = Ø***
Alors $\boldsymbol{int_{S'2}(H\_Act, T_i) = \mu_{Hors\_ZO\_Vidanger\_Ti}(h_i)}$
Sinon Si ***Adm_Act_Hors_ZO_Remplir = Ø***
Alors $\boldsymbol{int_{S'2}(H\_Act, T_i) = \mu_{Hors\_ZO\_Remplir\_Ti}(h_i)}$
Sinon $\boldsymbol{int_{S'2}(H\_Act, T_i) = 0}$

Bilan sur l'ensemble du réseau (à un instant donné) :
$\boldsymbol{int_{S'2}(H\_Act) = min_{i \in I}\ (int_{S'2}(H\_Act, T_i))}$.

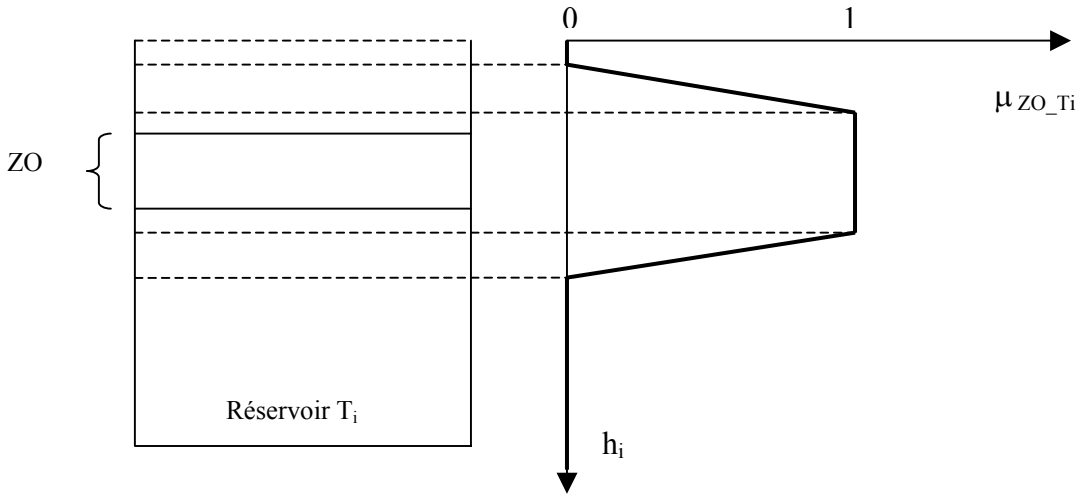


Figure 53 : Degré d'intention correspondant à une situation d'un réservoir dans sa zone optimale en fonction du niveau d'eau

Remarque 1 : μ_{ZO_Ti} est défini à partir de $\mu_{Hors\_ZO\_Vidanger\_Ti}$ et de $\mu_{Hors\_ZO\_Remplir\_Ti}$, notamment pour les limites des zones floues.

Remarque 2 : par définition, lorsque l'on se trouve dans la zone optimale d'un réservoir T_i, il n'y a pas d'action à effectuer sur ce réservoir, dans le cadre de la stratégie de minimisation de la durée totale de la simulation ($Adm\_Act\_ZO\ (T_i) = \emptyset\ \forall T_i$).

2.3. Situation d'anticipation de débordement ou non (situation S'3 = (Ant_Déb ; No_Ant_Déb))

Si pour chaque couple de réservoirs reliés par une conduite sans vanne, soit l'*HA* juge que la stratégie d'anticipation de débordement ne s'applique pas, soit qu'elle s'applique mais qu'il n'y a aucune action à faire pour anticiper le débordement futur, alors la non action de l'*HA* est cohérente avec le comportement de l'*AA*. De façon plus formelle :

Pour chaque couple (T_i, T_j) reliés par une conduite sans vanne (ensemble ***IJ***), nous calculons le degré d'intention associé à la "non action" correpondant à la situation anticipation de débordement ou non : $int_{S'3}\ (H\_Act, T_i, T_j)$:

Si $\mathbf{(1 - \mu_{Ant\_Déb\_Ti\_Tj}(h_i, h_j)) \geq \mu_{Ant\_Déb\_Ti\_Tj}(h_i, h_j)}$

Alors $\mathbf{int_{S'3}(H\_Act, T_i, T_j) = 1 - \mu_{Ant\_Déb\_Ti\_Tj}(h_i, h_j)}$ (*HA* n'a pas agi sur T_j car il a considéré que l'on n'était pas dans une situation d'anticipation de débordement : voir remarque ci-dessous)

Sinon si $\mathbf{Adm\_Act\_Ant\_Déb(T_i, T_j) = \emptyset}$

Alors $\mathbf{int_{S'3}(H\_Act, T_i, T_j) = \mu_{Ant\_Déb\_Ti\_Tj}(h_i, h_j)}$ (*HA* n'a pas agi sur T_j car il a considéré que l'on était dans une situation d'anticipation de débordement mais qu'il n'y avait pas d'action à faire)

Sinon $\mathbf{int_{S'3}(H\_Act, T_i, T_j) = 0}$

Bilan sur l'ensemble du réseau (à un instant donné) :

$\mathbf{int_{S'3}(H\_Act) = \min_{(i,j) \in IJ} (int_{S'3}(H\_Act, T_i, T_j))}$.

On retient donc l'interprétation de plausibilité maximale entre les deux choix *Anticipation Débordement* et *non Anticipation Débordement*, mais le choix *Anticipation Débordement* est soumis à des conditions.

Remarque : si l'on n'est pas dans une situation d'anticipation de débordement, alors par définition il n'y a aucune action à effectuer dans ce cadre ($Adm\_Act\_\neg Ant\_Déb(T_i, T_j) = \emptyset\ \forall$ T_i et T_j).

3. Récapitulatif et conclusion

Dressons un tableau récapitulatif montrant l'ensemble des situations traitées.

	H_Act ≠ No_Act
∃ i Adm_Act_Al (T_i) ≠ ∅ et H_ACT ∈ Adm_Act_Al (T_i)	**int_{S1} (H_Act)**
Adm_Act_No_Al ≠ ∅ et H_ACT ∈ Adm_Act_No_Al	**int_{S2} (H_Act)**
∃ i Adm_Act_Hors_ZO (T_i) ≠ ∅ et H_ACT ∈ Adm_Act_Hors_ZO (T_i)	**int_{S3} (H_Act)**
∃ i, j Adm_Act_Ant_Déb (T_i, T_j) ≠ ∅ et H_ACT ∈ Adm_Act_Ant_Déb (T_i, T_j)	**int_{S4} (H_Act)**
sinon (les 4 conditions ci-dessus non vérifiées)	**H_Act ∉ comportement de AA int_{S5} (H_Act) = 0**

Tableau 13 : Bilan dans le cas où l'opérateur humain effectue une action

	H_Act = No_Act
Tests locaux associés à la situation (alarme ; non alarme)	**$int_{S'1}$ (H_Act)**
Tests locaux associés à la situation (zone optimale ; hors zone optimale)	**$int_{S'2}$ (H_Act)**
Tests locaux associés à la situation (anticipation débordement ; non anticipation débordement)	**$int_{S'3}$ (H_Act)**

Tableau 14 : Bilan dans le cas où l'opérateur humain n'effectue pas d' action

Le comportement global de l'opérateur humain par rapport à l'agent artificiel sur une simulation est donné par une mesure de similarité globale effectuée sur l'ensemble de ses actions :

Global_Sim = (1/nb_H_Act). Σ_{H_Act} ***int (H_Act)*** avec :

int (H_Act) = min $_{j=1à3}$ ***(int$_{S'j}$(H_Act))*** **si *H_Act = No_Act.*** En effet, on n'agit pas si dans les trois situations *S'j* qui se présentent, il n'y avait aucune action possible à faire, ceci dans le cadre du comportement de l'agent artificiel *AA,* et le minimum reflète bien la conjonction "et".

int (H_Act) = max $_{j=1à5}$ ***(int$_{Sj}$ (H_Act))*** **si *H_Act ≠ No_Act.*** En effet, on recherche ici l'intention de plausibilité maximale associée à l'action *H_Act* car il s'agit d'une décision liée

à un choix (connecteur “ou”). On restreint bien sûr le maximum sur les situations *Sj* telles que les conditions sont vérifiées.

Le problème de cette approche est qu’il y a parfois des actions litigieuses, c’est-à-dire sujettes à plusieurs interprétations possibles, les degrés d’adéquation avec certaines des situations définies ci-dessus ayant donc des valeurs proches. En effet, comme cela a déjà été dit, une même action peut être vue comme satisfaisant plusieurs buts ou stratégies différentes. Si l’on souhaite seulement juger la cohérence du comportement de l’opérateur humain par rapport au comportement de l’agent artificiel, ce n’est pas gênant. Cela le devient si l’on souhaite analyser les différentes stratégies mises en œuvre par l’opérateur humain au cours de la simulation. Une façon d’éliminer certaines interprétations peut être :

- de définir les zones floues précédentes de manière plus étroite (zones de faible amplitude). On peut alors comparer l’opérateur humain à plusieurs agents artificiels avec chacun des zones floues étroites et différentes les unes des autres. Cela peut néanmoins nécessiter un nombre important d’agents si l’on souhaite balayer les différentes situations possibles.

- d’examiner l’ensemble de la séquence des actions de l’opérateur humain de manière à dégager certaines caractéristiques de son comportement. On peut ainsi :

  • tenter de dégager ses propres seuils pour chaque réservoir, de façon plus précise que les seuils flous ci-dessus définis de manière générique et assez tolérante. Le but est donc de caler les classes de situations précédentes en fonction des actions non litigieuses de l’opérateur humain (les actions qui ne sont sujettes qu’à une seule interprétation). On peut alors interpréter de manière plus restrictive les actions litigieuses. La zone d’indétermination (ou floue), notée [*h1* ; *h2*], correspond à la zone où l’opérateur humain a effectué des actions non litigieuses mais contradictoires (par exemple à un niveau d’eau *h1* il a considéré qu’un réservoir était en alarme et plus tard, à un niveau d’eau *h2* légèrement supérieur à *h1*, il a considéré que ce réservoir n’était pas en alarme), ou bien à une zone d’actions litigieuses, telle que pour $h > h2$ il a toujours considéré que le réservoir était en alarme, pour $h < h1$ il a toujours considéré que ce réservoir n’était pas en alarme, et pour $h1 < h < h2$ il s’agit d’actions litigieuses.

  • privilégier une certaine logique (ou continuité) de comportement : par exemple supposons qu’une action interprétée de façon non litigieuse comme réduisant une alarme d’un réservoir est précédée ou suivie d’une action interprétée de façon litigieuse. On peut alors éliminer le litige en supposant que l’intention associée à cette action est également de réduire l’alarme du même réservoir.

On se trouve donc ici dans une situation de prise de décision différée : pour interpréter les actions litigieuses de l’opérateur humain, on attend d’avoir davantage d’informations sur son comportement.

Remarque : une manière de régler le problème de la définition de ces zones floues serait de demander en début de session de test à l’opérateur humain quels sont ses seuils d’alarme par exemple. Mais alors on le pousse implicitement à avoir une stratégie utilisant des seuils fixes, qui est comme on l’a vu une stratégie possible parmi d’autres. On l’influence donc pour qu’il ait la même notion d’alarme que l’agent artificiel. Cela biaise donc les résultats.

7.1.2. Distance globale

De manière plus simple, on peut définir une distance numérique :

● Sans comparer des séquences d'actions, on peut utiliser le critère C^{ij} :

Pour deux simulations S^i et S^j, on a $C^{ij} = (D_t^i - D_t^j) / (D_t^i + D_t^j) + (V_{déb}{}^i - V_{déb}{}^j) / (V_{déb}{}^i + V_{déb}{}^j)$
D_t^i = durée totale de la simulation i
$V_{déb}{}^i$ = volume total débordé pour le simulation i
$C^{ij} > 0 \Rightarrow S^j$ meilleur que S^i
$C^{ij} < 0 \Rightarrow S^i$ meilleur que S^j
$C^{ij} = 0 \Rightarrow S^i$ et S^j "idem"

Le problème de ce critère est qu'il est utile uniquement pour comparer deux simulations (par exemple un opérateur artificiel et un opérateur humain), mais il ne permet pas de juger si une simulation est de bonne qualité. En effet, pour cela il faut connaître l'optimum, ou tout au moins avoir une bonne connaissance de ce que peut être cet optimum.

● On peut également utiliser le critère C^i_{opt} pour juger la qualité d'une simulation :

Pour une simulation S^i, on a $C^i_{opt} = (D_t^i - D_t^{min}) / (D_t^i + D_t^{min}) + V_i / V_{déb}{}^{max}$
D_t^{min} = durée minimale d'une simulation
$V_{déb}{}^{max}$ = volume maximum de débordement

Ainsi, comme on le voit, ce critère nécessite l'intervention du contrôle optimal afin de calculer les paramètres D_t^{min} et $V_{déb}{}^{max}$.

● Le critère choisi pour juger la qualité d'une simulation est finalement une fonction linéaire pondérée du volume total débordé et de la durée totale de la simulation :

Pour une simulation S^i, on a $C^i_{linéaire} = V_{déb}{}^i + \alpha. D_t^i$
Le choix de la pondération α est fonction du type de problèmes étudiés, mais demeure arbitraire (voir discussion précédente au chapitre 3.2.4.).
Mais il est à noter que pour tous ces critères sont assez approximatifs car pour juger si une simulation est de bonne qualité, il faudrait idéalement connaître l'optimum (nous rappelons que la problème traité ici est un problème de commande optimale, avec deux objectifs à optimiser), ou tout au moins avoir une bonne connaissance de ce que peut être cet optimum.

7.2. MODE EN LIGNE

7.2.1. Généralités

Le raisonnement de l'agent artificiel est dépendant du raisonnement de l'humain dans le sens où les deux opérateurs sont face au même test simultanément et à chaque fois que l'opérateur humain effectue une action, l'agent artificiel indique quelle action il aurait fait, mais il ne réalise pas cette action. Dans ce cas, il est possible d'effectuer une comparaison action par action et on parle de ***distance locale***.

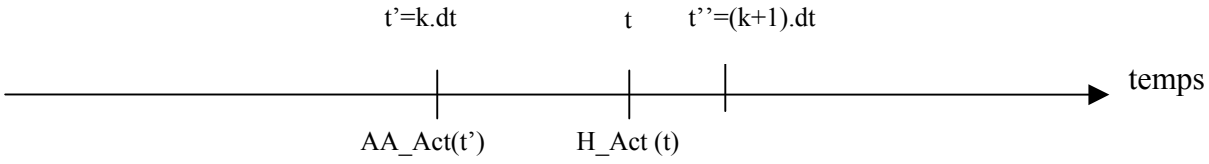


Figure 54 : Compatibilité entre l'instant d'action d'un opérateur humain et le pas de temps de l'opérateur artificiel

Si à un instant *t* l'humain effectue l'action *H_Act(t)*, alors il faut récupérer (donc avoir stocké préalablement) l'état (les positions des vannes et les hauteurs d'eau) au temps *t'* correspondant à un instant de calcul du système (*t' = k.dt* avec *k* entier et *dt* le pas de temps de calcul hydraulique) et précédant immédiatement *t*, puis lancer le calcul de l'action *AA_Act(t')* que ferait l'agent artificiel dans l'état récupéré. L'action *H_Act(t)* a lieu entre *k.dt* et *(k+1).dt* et est affectée à l'instant *t' = k.dt*. Ainsi, tant que l'humain n'effectue pas d'action, l'agent artificiel stocke l'état à chaque pas de temps de calcul *dt*.
Mais l'agent artificiel possède lui aussi un pas de temps Δt ($\Delta t = n.dt$ avec n entier), si bien que si l'humain effectue une action à l'instant *t* (affectée à l'instant de calcul précédent *t' = k.dt*) mais n'en effectue pas d'autre entre *t* et $t'+\Delta t$, alors l'agent artificiel indique l'action qu'il aurait fait à cet instant $t'+\Delta t$ (et l'humain n'a donc effectué aucune action à cet instant).
La distance locale sera alors issue de la comparaison des *H_Act(t)* et des *AA_Act(t')* correspondant sur l'ensemble de la simulation.

7.2.2. Cas d'un opérateur qualitatif

Il s'agit d'une méthode de comparaison action par action spécifique à l'agent qualitatif, développée par [Trave-Massuyes *et al.* 99], et que nous avons implémenté.

Pour chaque action de l'humain, on doit juger si cette action est admissible au sens du modèle qualitatif (voir 3.1.2). Si c'est le cas, on peut effectuer une comparaison avec l'action qu'aurait effectué l'agent qualitatif. Si le nombre d'actions non admissibles (inconsistantes avec les buts du modèle) effectuées par l'humain est supérieur à un seuil, alors le raisonnement de l'humain est considéré comme n'appartenant pas à la même classe que celle des agents qualitatifs.

Cas sans alarme : pour chaque action admissible de l'humain, on définit un score associé à l'humain G_H et un score associé à l'agent artificiel G_A ($G_A = 0$ par référence). Trois cas peuvent alors se présenter :

Action de l'agent artificiel	Action de l'humain	G_H
Pas d'action	Pas d'action	$G_H = 0$
Vanne ouverte	Vanne ouverte	$G_H = G_1$
Vanne ouverte	Pas d'action	$G_H = G_2$

Tableau 15 : Distance locale

Les valeurs de G_1 et G_2 sont calculées de la façon suivante :

Comme au paragraphe 3.2.2, nous supposons que les chemins sont étiquetés par le nombre de vannes fermées.

Soit D = (étiquette minimale sur les chemins traversant la vanne actionnée par l'humain) – (étiquette minimale sur les chemins traversant la vanne actionnée par l'agent qualitatif)
Si $D \neq 0$ alors $G_1 = D$.
Si $D = 0$ alors on considère tous les chemins d'étiquette minimale. Soit $l + 1$ le nombre de classes d'équivalence définies par le diamètre minimum (min_Φ_i) sur un chemin i et classées par ordre décroissant. La classe 0 inclut le chemin traversant la vanne actionnée par l'agent qualitatif. Si le chemin qui traverse la vanne actionnée par l'humain et qui possède la valeur maximale de min_Φ_i appartient à la classe j, alors $G_1 = j / l$.
G_2 = 1 + (étiquette maximale parmi tous les chemins arrivant au réservoir puit) – (étiquette minimale parmi les chemins traversant la vanne actionnée par l'agent qualitatif).

Cas avec alarme : comme précédemment, pour chaque action admissible de l'opérateur humain, on définit un score associé à l'humain G_H et un score associé à l'agent artificiel G_A. On utilise pour cela la méthode vue au paragraphe 3.2.2 (étape 3) pour calculer un score en fonction de la réussite de chacun des quatre buts ($G = \sum_i p_i.G_i$). On utilise les mêmes poids p_i pour les deux opérateurs.

Comparaison : soit $D_t = G_A(t') - G_H(t)$ calculé pour chaque action admissible de l'humain aux instants $t_1, t_2, \ldots, t_f$ (correspondant aux instants $t'_1, t'_2, \ldots, t'_f$ de l'agent artificiel comme cela a été indiqué plus haut).
On peut alors mesurer la similarité entre les deux raisonnements à l'aide d'une norme dans $\mathbb{R}^f$ (f = nombre d'actions admissibles de l'humain) :

$$d_1 = \sqrt{\frac{\sum_{i=1}^{i=f} (D_{t_i})^2}{f}} \qquad \text{ou} \qquad d_2 = \frac{\sum_{i=1}^{i=f} \left| D_{t_i} \right|}{f}$$

L'interprétation de ces distances se fait à l'aide d'une borne supérieure γ :

$$\gamma = \max\left\{ y - x, \binom{n}{2} \right\} - (x - y)$$

avec $x = -(n-2).(p_1 + p_2) - p_4.n^*$ *et* $y = p_2.(n-2) + 2.p_3 + p_4.n^*$

et n = nombre de réservoirs

n^* = nombre maximum de chemins pouvant être ouverts en même temps par action sur une vanne.

D'où l'interprétation de la distance calculée en la situant dans l'intervalle [0 ; γ] :

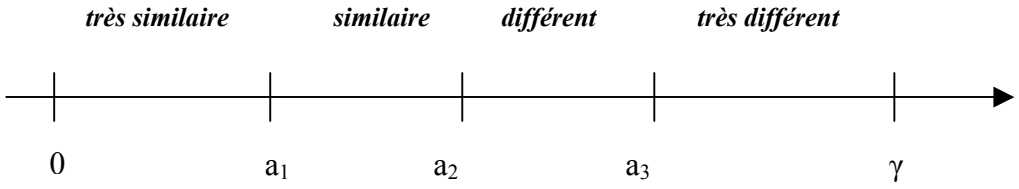


Figure 55 : Interprétation qualitative de la distance

On peut prendre par exemple : $a_1 = \gamma/4$ $a_2 = \gamma/2$ et $a_3 = 3\gamma/4$

Il est à noter que les paramètres du modèle (ΔV^{alarme} et les coefficients de pondération pour définir le score associé à chaque action admissible) influencent la comparaison avec le raisonnement d'un opérateur humain.

7.2.3. Distance locale générique

Nous définissons ici une méthode de comparaison directe de l'action de l'opérateur humain avec l'action de l'agent artificiel, et qui a été implémentée.

L'action de l'agent artificiel appartient à un ensemble d'actions admissibles *Adm_Act* connu. Deux cas se présentent :

● ***si H_Act $\notin$ Adm_Act***

ce qui correspond à trois sous cas :
H_Act $\neq$ *No_Act et Adm_Act* $\neq \emptyset$ *(et H_Act* $\notin$ *Adm_Act)*
H_Act = No_Act et Adm_Act $\neq \emptyset$
H_Act $\neq$ *No_Act et Adm_Act* $= \emptyset$

Alors la similarité entre les deux actions est $s(AA\_Act, HA\_Act) = 0$

● ***si H_Act $\in$ Adm_Act***

ce qui correspond à deux sous cas :
H_Act $\neq$ *No_Act et Adm_Act* $\neq \emptyset$ *(et H_Act* $\in$ *Adm_Act)*
H_Act = No_Act et AA_Act = No_Act

Alors la similarité entre les deux actions est $s(AA\_Act, HA\_Act) = 0$

Bilan : le résultat de la comparaison action par action entre l'opérateur humain et l'agent artificiel sur une simulation est donné par un degré de similarité locale (1 – distance locale) :

$$Local\_Sim = (1/nb\_H\_Act).\Sigma_{H\_Act}\, s(AA\_Act, HA\_Act)$$

7.2.4. Conclusion

Le calcul des similarités *s(AA_Act, HA_Act)* mériterait d'être rafiné, mais la méthode a l'avantage de comparer l'action de l'humain au but sous-jacent à l'action de l'agent artificiel, représenté par l'ensemble des actions admissibles.
Un inconvénient de la distance locale est qu'il est possible que deux opérateurs humains différents soient à la même distance locale d'un agent artificiel alors que les deux opérateurs humains ont un comportement différent. Cela peut être dû par exemple à des plans utilisés qui sont différents mais qui mènent au même but. Un autre inconvénient est que si deux opérateurs humains effectuent les mêmes actions mais dans un ordre différent (mais on est dans une situation où l'ordre est indifférent) et aboutissent tous deux au même succès (même $V_{déb}$ et D_t par exemple), alors il serait bon qu'ils soient à la même distance d'un agent artificiel, ce qui ne sera pas le cas avec une comparaison action par action au même instant.

Nous allons maintenant détailler les tests effectués sur des sujets humains et les agents artificiels par le biais d'expérimentations informatiques.

Chapitre 8

SESSIONS DE TESTS

Nous analysons ici les résultats d'expérimentations informatiques : d'une part la comparaison entre les agents artificiels permet de définir une coopération, d'autre part la comparaison entre des opérateurs humains et artificiels.

8.1. GENERALITES

Une session de tests a été menée sur des sujets humains (des doctorants issus de disciplines scientifiques et agés de 25 à 30 ans), prenant en compte un ensemble de micro-mondes de difficulté croissante.
Il s'agit d'expliquer au sujet, en début de session, quels sont les objectifs et comment il peut interagir avec le micro-monde. Aucune phase d'entraînement n'est prévue, car d'une part les connaissances nécessaires à la compréhension de la tâche et des phénomènes physiques sont relativement simples et supposées acquises par les sujets, et d'autre part un des buts des expérimentations est d'étudier l'évolution du comportement des sujets (plus précisément l'apprentissage de stratégies de gestion) au cours des tests (le même test est passé plusieurs fois par chaque sujet).
Une ***consigne*** est donc donnée en début de session, qui reflète les objectifs à optimiser et les contraintes. Par exemple, une instruction peut être : *transférer l'eau du réservoir le plus haut vers le réservoir le plus bas en un minimum de temps et en évitant les débordements des réservoirs intermédiaires, en agissant sur les vannes binaires*.

Le protocole est le suivant [Ponsa et Catala 99 (a)] :

- expliquer au sujet humain qu'il peut agir grâce à la souris du PC (→10 secondes)
- l'opérateur observe le micro-monde et lit l'instruction (→ 30 secondes)
- début du test ($t = 0$)
- première action ($t = t_1$)
- dernière action ($t = t_f$)
- fin d'évolution du micro-monde ($t = t_{ff}$)

8.2. EXPERIMENTATIONS INFORMATIQUES

8.2.1. Comparaison des agents artificiels et coopération

L'agent basé sur des cas utilise des stratégies heuristiques relativement évoluées correspondant aux index/cas 6 à 9. Ceci lui permet d'obtenir des résultats nettement meilleurs

que l'agent qualitatif lors des tests, car ce dernier ne possède pas ce type de stratégie. Il semble donc raisonnable de conserver ces stratégies lors de la définition de la coopération.

Par ailleurs, l'agent qualitatif semble avoir une bonne gestion des situations d'ouverture de vanne dans un cas sans alarme (recherche d'ouverture de chemins selon des critères à optimiser), et également des situations de gestion mono alarme (prédiction à un pas de temps des tendances des différentes actions possibles et choix d'une action selon des critères à optimiser). Ainsi, dans ces deux types de situations, l'agent qualitatif semble plus efficace que l'agent basé sur des cas qui utilise des index/cas qui ne sont pas optimisés. Donc dans les situations sans alarme et les situations de gestion mono-alarme, nous allons conserver les stratégies de l'agent qualitatif.

L'idée de coopération entre ces deux agents est donc la suivante : à chaque pas de calcul de l'agent artificiel, on cherche tout d'abord à utiliser l'agent à base de cas afin que celui-ci tente d'appliquer les index 6 à 9 correspondant aux stratégies heuristiques (gestion des alarmes multiples, anticipation des débordements et minimisation de la durée totale). Si un de ces index est applicable dans la situation courante, alors l'action correspondante au cas sélectionné est effectuée, et on passe au pas de calcul suivant. Mais si aucun de ces index/cas n'est applicable, alors le système passe la main à l'agent qualitatif qui va tenter d'agir (dans les situations sans alarme et de gestion mono-alarme). Cette coopération est réalisée par deux agents nommés *NQA-Tactic i* (i = 1 à 2), *NQA* étant l'agent qualitatif et *Tactic i* étant l'agent CBR : *Tactic 1* utilise les index 6 à 8, car l'index 9 (stratégie de minimisation de la durée) correspond à une stratégie particulière qui nécessite une analyse assez fine du problème et n'est donc pas forcément utilisée par un opérateur humain. L'autre agent, nommé *Tactic 2*, utilise les index 6 à 9 (voir 5.6.). La collaboration est davantage marquée lorsque l'agent CBR (le premier agent qui tente d'agir) sélectionne l'index n° 7 (m réservoirs en alarme) : dans cette situation, la tâche de l'agent CBR est de sélectionner le réservoir en alarme qui possède le plus gros rapport $\Sigma\Phi_e$ / $\Sigma\Phi_s$ (Φ_e et Φ_s sont les diamètres entrant et sortant avec écoulement). C'est alors au tour de l'agent qualitatif de trouver une action en utilisant l'information fournie : il applique sa stratégie de gestion mono-alarme (voir 3.2.2) en se focalisant sur le réservoir sélectionné et en considérant que seul ce réservoir est en alarme.

Remarque 1 : par rapport à la classification des situations d'interactions définies par Ferber [Ferber 95] (voir chapitre 2.4.2.), on peut considérer ici que l'on se trouve dans une situation de collaboration simple dans le cas de l'index n° 7 et d'indépendance pour les autres index.

Remarque 2 : ce type de stratégie rejoint celle mise en œuvre par [Torasso 04], pour qui la coopération est mise en œuvre de la façon suivante : si le système CBR échoue à trouver une solution au problème, alors le système de raisonnement à base de modèles (*Model-Based Reasoning* - MBR) tente d'en trouver une. Si c'est le cas, alors la solution fournie par le module MBR est ajoutée à la mémoire de cas. De plus, au lieu d'utiliser une stratégie fixe (CBR d'abord, et MBR ensuite si échec du premier), l'auteur a expérimenté des stratégies opportunistes qui combinent les systèmes CBR et MBR de différentes façons en fonction de la difficulté estimée du problème à résoudre.

	$V_{déb}$ (m3)	D_t (sec.)	Critère C
Config 3 NQA-Tactic 1 ΔV^{alarme}=1.5 Δt = 2	4.34	582	33.4
Config 3 NQA-Tactic 2 ΔV^{alarme}=1.5 Δt = 2 ΔV_ZO= 3	0	432	21.6
Config 3 NQA-Tactic 1 ΔV^{alarme}= 2 Δt = 3	5.13	585	34.4
Config 3 NQA-Tactic 2 ΔV^{alarme}= 1 Δt = 2 ΔV_ZO= 1	0	420	21
Config 3 NQA-Tactic 2 ΔV^{alarme}= 2 Δt = 3 ΔV_ZO= 6	0	444	22.2
Config 3 NQA-Tactic 1 ΔV^{alarme}= 10 Δt = 2	0	584	29.2
Config 1 NQA-Tactic 1 ΔV^{alarme}=1.5 Δt = 2	3.59	754	41.3
Config 1 NQA-Tactic 2 ΔV^{alarme}=1.5 Δt = 2 ΔV_ZO= 3	0	658	32.9
Config 1 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 2 ΔV_ZO= 1	0	662	33.1
Config 1 NQA-Tactic 1 ΔV^{alarme}= 2 Δt = 3	2.38	762	40.5
Config 1 NQA-Tactic 2 ΔV^{alarme}= 2 Δt = 3 ΔV_ZO= 6	0	669	33.4
Config 1 NQA-Tactic 1 ΔV^{alarme}= 10 Δt = 2	0.55	776	39.4
Config 4 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 2 ΔV_ZO= 1.5	1.37	197	11.2
Config 4 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 1 ΔV_ZO= 1.5	0	177	8.8
Config 4 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 0.5 ΔV_ZO= 1.5	0	164	8.2
Config 4 NQA-Tactic 2 ΔV^{alarme}= 3 Δt = 1 ΔV_ZO= 2	0	175	8.7
Config 4 NQA-Tactic 1 ΔV^{alarme}= 3 Δt = 1	2.68	213.5	13.4
Config 2 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 2 ΔV_ZO= 1.5	0.33	190	9.8
Config 2 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 0.5 ΔV_ZO= 1.5	0	183.5	9.2
Config 2 NQA-Tactic 1 ΔV^{alarme}=1 Δt = 1	1.15	196	11
Config 2 NQA-Tactic 2 ΔV^{alarme}=3 Δt = 1 ΔV_ZO= 2	0	189	9.4
Config 2 NQA-Tactic 1 ΔV^{alarme}=3 Δt = 1	0	201	10
Config 6 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 2 ΔV_ZO= 1.5	0.04	199.5	10
Config 6 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 0.5 ΔV_ZO= 1.5	0	194.5	9.7
Config 5 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 2 ΔV_ZO= 1.5	2.53	144	9.7
Config 5 NQA-Tactic 2 ΔV^{alarme}=1 Δt = 1 ΔV_ZO= 1.5	0.27	140	7.3
Config 5 NQA-Tactic 1 ΔV^{alarme}=1 Δt = 1	2.83	166.5	11.2
Config 5 NQA-Tactic 2 ΔV^{alarme}=3 Δt = 1 ΔV_ZO= 2	0	142	7.1
Config 5 NQA-Tactic 1 ΔV^{alarme}=3 Δt = 1	1.09	168.5	9.5

Config i : une configuration du micro-monde
$V_{déb}$ = volume total débordé D_t = durée de la simulation $C = V_{déb} + \alpha.D_t$ = objectif global à minimiser ($\alpha = 1/20$)
Δt = pas de temps de calcul de l'agent artificiel
ΔV^{alarme} = volume d'alarme : si $V_i(t) > (V_i^{max} - \Delta V^{alarme})$ alors alarme du réservoir T_i à l'instant t
NQA-Tactic i (i = 1 à 2) : coopération CBR-NQA
ΔV_ZO = zone floue (en m3) autour du volume optimal = $2.V_\varepsilon$ (voir 5.2.3. index n° 9)

Tableau 16 : Résultats des tests effectués sur l'agent basé sur une coopération

Les résultats observés lors des tests de l'agent artificiel à base de cas et les remarques correspondantes (voir 5.6) sont valables ici, à savoir : les résultats sont meilleurs que pour un agent qualitatif seul, notamment avec l'emploi de la tactique n° 2 (*NQA-Tactic 2*) qui réduit nettement la durée et les débordements.

Les simulations des configurations n° 4 et 5 sont caractérisées par des vitesses élevées et donc des remplissages rapides des réservoirs. Cela nécessite un pas de temps de calcul Δt réduit afin de bien gérer la situation. Les résultats montrent qu'avec un pas de temps de 2 secondes, des débordements se produisent. Ils sont dus à un volume important entrant dans chaque réservoir entre deux actions consécutives, entraînant de nombreuses alarmes, ce qui empêche d'appliquer la stratégie de minimisation de la durée (*NQA-Tactic 2*). La diminution du pas de temps permet d'une part de réduire voire éliminer les débordements, et d'autre part de réduire la durée totale de la simulation. En effet, dans ce cas, l'agent artificiel peut appliquer de manière efficace la stratégie de minimisation de la durée (*NQA-Tactic 2*).

La coopération entre les deux agents peut avoir des effets négatifs, notamment lorsqu'une action effectuée par l'agent CBR afin d'appliquer la stratégie de minimisation de la durée ou la stratégie d'anticipation de débordement est annulée (action contraire) par l'agent NQA quelques pas de temps plus tard, ce qui peut même entraîner un débordement d'un réservoir. Un moyen d'empêcher ces effets est d'avoir une liste d'actions Tabou dont la taille est importante (de l'ordre de 10), ce qui ne semble pas judicieux. Ce problème n'est d'ailleurs pas spécifique à la coopération car il intervient également avec l'agent artificiel basé sur des cas : l'index correspondant à l'ouverture d'une vanne dans une situation sans alarme annule la stratégie de minimisation de la durée. Il s'agirait donc, pour les réservoirs concernés par la stratégie de minimisation de la durée, de ne gérer ces réservoirs que par cette stratégie, donc sans faire intervenir la notion d'alarme.

Les résultats de ces tests (à comparer avec les tests de l'agent à base de cas seul) montrent que l'amélioration attendue n'a pas eu lieu. Les résultats sont sensiblement les mêmes pour l'agent à base de cas et l'agent basé sur une coopération. Ceci est du au fait que la stratégie de gestion mono-alarme de l'agent qualitatif n'est pas excellente, et elle pourrait peut-être, comme cela a déjà été dit, être améliorée grâce à une modification des valeurs des paramètres de cet agent (par exemple par l'apprentissage). Il semble en outre que la stratégie de recherche d'un chemin à ouvrir (cas sans alarme) de l'agent qualitatif a globalement peu d'influence, car en pratique cette stratégie est utilisée principalement en début de simulation lors de l'ouverture d'un ensemble de vannes.

L'amélioration de la coopération pourrait consister en une *amélioration de l'agent qualitatif*, tant au niveau des valeurs de ses paramètres (par exemple avec une procédure d'apprentissage : voir 3.2.4.) qu'au niveau de ses stratégies de contrôle. Mais il s'agirait surtout de définir des stratégies pour l'agent qualitatif qui soient complémentaires de celles de l'agent à base de cas.
En effet, pour notre problème, l'agent à base de cas et l'agent qualitatif sont actuellement plus en situation d'indépendance (voir remarque 1 ci-dessus) que complémentaires (ce qui aboutirait à une situation de collaboration simple ou coordonnée), car ils opèrent actuellement tous deux au même niveau, celui de la résolution (dans tel état du système, effectuer telle action). Une piste qui semble prometteuse serait de placer les agents à des niveaux conceptuels différents afin de véritablement imbriquer les agents : l'agent CBR opérerait toujours au niveau de la résolution, mais l'agent qualitatif opérerait maintenant au niveau de la représentation des connaissances et du raisonnement, sous la forme de modèles qualitatifs

(voir chapitre 9 : conclusion) et fournirait une aide à l'agent CBR lors des différentes phases du processus de résolution.

8.2.2. Comparaison entre un opérateur humain et un agent artificiel

		$V_{déb}$ (m3)	D_t (sec.)	C	Comparaison Sujet vs. NQA-Tactic 1 **Global_Sim**	Comparaison sujet vs. NQA-Tactic 1 **Local_Sim**	Comparaison sujet vs. NQA-Tactic 2 **Global_Sim**	Comparaison Sujet vs. NQA-Tactic 2 **Local_Sim**
Sujet 1	Config 5	11.5	137.5	18.4	0.32	0.4	0.38	0.4
	Config 5	1.9	184.5	11.1	0.45	0.43	0.18	0.13
	Config 5	0	140	7	0.17	0.25	0.56	0.56
	Config 2	3.15	168.5	11.6	0.15	0.24	0.37	0.43
	Config 2	0	176.5	8.8	0.13	0.29	0.44	0.51
Sujet 2	Config 5	0	180.5	9	0.16	0.35	0.44	0.59
	Config 5	0	142	7.1	0.22	0.27	0.53	0.52
	Config 5	0	144.5	7.2	0.23	0.3	0.56	0.63
	Config 2	0	180.5	9	0.18	0.33	0.42	0.61
	Config 6	0.37	200.5	10.4	0.62	0.74	0.43	0.47
	Config 4	0	173.5	8.7	0.19	0.42	0.51	0.54
Sujet 3	Config 5	1.97	192.5	11.6	0.37	0.4	0.12	0.07
	Config 5	0	175	8.6	0.28	0.61	0.22	0.18
	Config 2	0.25	211	10.8	0.38	0.51	0.21	0.22
	Config 2	0.23	211	10.8	0.37	0.53	0.18	0.15
	Config 6	0	201.5	10	0.46	0.62	0.38	0.44
Sujet 4	Config 5	0	150.5	7.5	0.14	0.18	0.48	0.48
	Config 2	0.002	199	10	0.31	0.44	0.19	0.2
	Config 6	0	213	10.7	0.53	0.67	0.48	0.56
Sujet 5	Config 5	17.8	216.5	28.7	0.08	0.32	0.32	0.31
	Config 5	19.5	208	29.9	0.07	0.37	0.1	0.05
	Config 5	5.9	169	14.4	0.18	0.21	0.12	0.08
	Config 2	12	191	21.6	0.34	0.58	0.13	0.12
	Config 2	5.5	230.5	17	0.37	0.47	0.15	0.16
Sujet 6	Config 5	0.84	210.5	11.4	0.45	0.48	0.16	0.17
	Config 5	0	201.5	10.1	0.46	0.5	0.16	0.08
	Config 2	0	201	10.1	0.14	0.28	0.39	0.50
Sujet 7	Config 5	2.04	168	10.4	0.36	0.39	0.33	0.32
	Config 5	0	170	8.5	0.43	0.47	0.2	0.17
	Config 2	0.6	197.5	10.5	0.33	0.5	0.17	0.2
	Config 2	0	192	9.6	0.27	0.32	0.16	0.2

Config i représente une configuration du réseau
Les doubles traits horizontaux séparent les résultats des tests des différents sujets humains
Δt = pas de temps de calcul des agents artificiels = 2 secondes
$V_{déb}$ = volume total débordé
D_t = durée de la simulation
$C = V_{déb} + \alpha.D_t$ = objectif global à minimiser ($\alpha = 1/20$)
NQA-Tactic 1 : coopération CBR-NQA sans utiliser le cas "stratégie de minimisation de la durée"
NQA-Tactic 2 : coopération CBR-NQA en utilisant toutes les stratégies de l'agent CBR
Global_Sim = similarité globale (adéquation floue entre les intentions : voir 7.1.1.2.2.)
Local_Sim = similarité locale (comparaison action par action : voir 7.2.3.)

Tableau 17 : Résultats des tests des sujets humains

Même dans les cas où le comportement d'un sujet humain s'inscrit dans le cadre du comportement d'un agent artificiel, c'est-à-dire que les intentions associées aux situations dans lesquelles se produisent les actions correspondent effectivement à des couples (intention – situation génériques) de l'agent artificiel, les scores numériques reflétant la similarité entre les deux opérateurs ne sont pas très élevés (*Global_sim* ou *Local_sim* autour de 0.5). Ceci est dû au fait que les seuils flous de certains sujets humains sont très différents des seuils flous définis pour les agents artificiels : certaines actions de sujets humains correspondent par exemple à la gestion d'un réservoir en alarme mais ne peuvent être interprétées selon cette intention ($\mu_{Al\_Ti}(h_i)$ est nul ou très faible), car l'agent artificiel considère que la hauteur d'eau h_i est trop faible pour déclarer le réservoir T_i en état d'alarme. D'où l'intérêt d'améliorer la méthode de comparaison en calculant les propres seuils d'un opérateur humain lors de la simulation.

Il est d'ailleurs à noter que la rapidité de la vidange pour les configurations testées peut en partie expliquer ces scores de similarité peu élevés. En effet, le sujet humain, pressé par le temps, a souvent des seuils d'alarme soit très élevés (auquel cas il aurait dû agir avant selon l'agent artificiel), soit très faibles (auquel cas il aurait dû agir plus tard selon l'agent artificiel). Lors des tests de débuggage, la configuration testée était liée à une vidange lente des réservoirs, et les résultats de ces simulations ont montré une grande similarité de comportement avec l'agent artificiel (*Global_sim* de 0.7 à plus de 0.9), car le sujet humain qui effectuait les tests agissait de façon non pressée par le temps, donc utilisait un seuil d'alarme stable et souvent proche de celui de l'agent artificiel (l'opérateur humain avait d'ailleurs connaissance des seuils flous de l'agent artificiel, donc agissait, de manière consciente ou non, en fonction de ces seuils).

On note une bonne corrélation entre les scores de similarité *Global_sim* et *Local_sim*, ce qui laisse à penser que la mesure locale de comparaison action par action donne un estimation grossière mais souvent assez fiable.

La configuration n° 6, comportant un grand nombre de conduites sans vanne, engendre généralement un comportement du sujet humain similaire à celui de l'agent artificiel dont une des stratégies principales est de gérer ces situations avec des conduites sans vanne (*NQA-Tactic1*). Ainsi, la stratégie de minimisation de la durée n'est pas appliquée par les opérateurs humains pour cette configuration. On observe même chez certains sujets un changement de stratégie pour cette configuration, ces sujets étant en proches de l'agent artificiel *NQA-Tactic2* pour toutes les configurations, sauf pour la configuration n° 6 où ils sont proches de l'agent artificiel *NQA-Tactic 1*.

On note des comportements propres à chaque sujet. Pour le sujet humain 1, on observe un changement de stratégie correspondant à un apprentissage de la stratégie de minimisation de la durée (au cours des premiers tests il est plus proche de *NQA-Tactic1* que de *NQA-Tactic2*, puis la tendance s'inverse ensuite). Ce sujet a un comportement risqué (ΔV^{alarme} faible) et obtient de très bons résultats grâce à cette nouvelle stratégie. Comme cela a déjà été dit, les scores de similarité moyens sont dus à des zones floues (alarme, zone optimale) différentes pour le sujet 1 et l'agent artificiel. Pour les sujets 2 et 4, les changements de stratégies correspondent aux changements de configuration du réseau. Ces deux sujets ont montré un comportement moins risqué que le sujet 1, et ils employaient également la stratégie de minimisation de la durée, d'où la même remarque que précédemment à propos des scores de similarité moyens. Les sujets 3 et 7 n'ont pas employé la stratégie de minimisation de la durée, mais ont un comportement cohérent avec celui de l'agent *NQA-Tactic1*, donc le score

de similarité moyen est encore du à des seuils flous différents. Les sujets 5 et 6 ont un comportement relativement éloigné de ceux des agents artificiels, du à un manque de stratégie. Ici, les scores de similarités moyens reflètent donc bien ces comportements.

	Apprentissage de stratégies de gestion	Comportement risqué (ΔV^{alarme} faible)	Résultats
Sujet 1	Oui (évolution)	+ +	Très bon (proche NQA-tactic 2)
Sujet 2	Non (déjà connu)	+	Très bon (proche NQA-tactic 2)
Sujet 3	Non	-	Assez bon (proche NQA-tactic 1)
Sujet 4	Non (déjà connu)	+	Très bon (proche NQA-tactic 2)
Sujet 5	Non	- -	Médiocre (éloigné des agents artificiels)
Sujet 6	Non	- -	Médiocre (éloigné des agents artificiels)
Sujet 7	Non	-	Assez bon (proche NQA-tactic 1)

Tableau 18 : Bilan des tests des sujets humains

8.3. CONCLUSION

Les expérimentations informatiques montrent d'une part que la coopération que nous avons définie, qui est une mise en série des agents (l'agent CBR tente d'agir ; s'il ne le peut pas, alors il passe la main à l'agent qualitatif) n'apparaît pas judicieuse (l'agent basé sur la coopération obtient des résultats équivalents à l'agent CBR seul) car l'agent qualitatif est nettement plus faible que l'agent à base de cas et ils agissent tous deux au même niveau de résolution (dans telle situation, appliquer telle action). Il faudrait donc redéfinir l'agent qualitatif à un autre niveau, par exemple celui de la représentation (qualitative causale) des connaissances.

Les comparaisons entre les agents artificiels et des sujets humains permettent de caractériser le comportement des sujets et elles montrent que la méthode de comparaison passant par la reconnaissance des intentions pourrait être nettement améliorée en calculant les propres seuils d'un opérateur humain lors de la simulation.

Chapitre 9

CONCLUSION

L'idée générale de ces travaux est d'étudier la coopération et d'évaluer la plausibilité cognitive de modèles issus de l'IA et utilisés par le groupe DISCO du LAAS-CNRS, dans le cadre de la supervision de processus, un des thèmes prépondérant du groupe DISCO. En outre, le choix des trois types de modèles d'IA a été motivé par le fait que le groupe travaille par ailleurs à la mise en place d'une coopération entre ces trois approches dans le cadre du laboratoire AUTODIAG, commun avec la société ACTIA S.A.

● Trois types de modèles de raisonnement ont été testés sur un micro-monde lié à la physique hydraulique, avec une tâche de conduite composée d'un double critère de temps minimal et de volume débordé minimum :

- Les modèles de classification de données, qui fonctionnent à partir d'exemples mais sans connaissance *a priori*, ne donnent pas de bons résultats car il est délicat d'obtenir un grand nombre d'exemples permettant d'exhiber les concepts correspondant aux stratégies à mettre en place. Un travail futur serait de mettre au point une méthode d'apprentissage par examen de preuve (*Explanation Based-Learning* [Cornuejols et Miclet 02] [Maclin et Shavlik 89]) qui est une méthode de généralisation fonctionnant à partir de très peu d'exemples, grâce à une théorie du domaine sous le formalisme de la logique du premier ordre. Nous verrons ci-dessous que cette technologie pourrait être intégrée au sein d'un modèle qualitatif jouant le rôle de connaissances profondes utilisées par un système de raisonnement à base de cas.

- Le modèle de raisonnement qualitatif donne de bons résultats dans le cas de situations simples à gérer (un seul réservoir en alarme, ouvrir un chemin en cas de non alarme), mais ce modèle n'est pas doté de stratégies complexes visant à gérer par exemple des situations d'anticipation de débordement sur le long terme, ou à minimiser la durée totale de façon plus globale. Une amélioration consisterait à mettre en place une procédure d'apprentissage supervisé des paramètres du modèle (le seuil d'alarme ΔV^{alarme}, les coefficients de pondération p_1, p_2, p_3 *et* p_4 permettant de calculer le score associé à chaque action admissible, et les coefficients q_1, q_2 *et* q_3 permettant de définir la tendance des réservoirs). En effet, nous avons constaté qu'une approche non supervisée ne donne pas de bons résultats, notamment parce que les quatre gains intermédiaires G_i correspondant aux sous buts associés à une situation d'alarme ne discriminent pas suffisamment les actions possibles.

- Le modèle de raisonnement à base de cas donne de meilleurs résultats car il permet de mettre en place des stratégies heuristiques, simples ou complexes, générales ou spécifiques, grâce à des cas modélisant une partie de réseau utile pour gérer une situation donnée. Les travaux futurs seraient d'une part d'améliorer l'algorithme de mise en correspondance entre un graphe et un sous-graphe afin de le rendre exhaustif, d'autre part d'implémenter les procédures d'apprentissage (stockage de cas nouveau, apprentissage par

l'échec, généralisation), ce qui permettrait de tester les deux manières de modéliser une action au sein d'un cas : une action procédurale qui a l'intérêt d'être adaptable et donc de ne nécessiter qu'un petit nombre de cas (un seul *a priori*) mais dont l'apprentissage semble très délicat, et une action instanciée qui nécessite plusieurs cas pour arriver au même niveau de généralité que l'action procédurale mais dont l'apprentissage semble plus aisé.

● Cependant, d'autres approches pourraient être utiles pour notre problème :

Contrôle optimal

L'intérêt de cette approche est de connaître la séquence optimale d'actions. Cela serait très utile afin d'évaluer la performance des opérateurs humains ou artificiels par rapport à un référent absolu, et d'autre part d'exhiber des stratégies optimales. Cependant, d'un point de vue cognitif, l'approche n'est pas très intéressante. L'approche n'a pas encore été développée, car il se pose plusieurs problèmes : prise en compte de l'aspect optimisation multi-critères, et d'un aspect système hybride dû à la coexistence de phénomènes continus (lois hydrauliques régissant les écoulements) et à évènements discrets (actions sur les vannes).

Système à base de règles

On peut tout à fait imaginer qu'un système de règles puisse permettre de résoudre ce problème, bien que cela nécessiterait un certain nombre de règles (par opposition à la classification heuristique qui utilise des règles abstraites, et au raisonnement à base de cas qui utilise un petit nombre de cas).

On peut citer, dans le cadre de notre étude, quelques exemples de règles, dont certaines, très spécifiques, utilisent la logique des propositions (R2 et R3) (voir figure 6), et d'autres plus générales utilisent la logique des prédicats (R1) :

R1 : ***si*** $\exists\, i \neq \{1\ ;\ 5\}$ tel que $T_i(t\text{-}1)$ en alarme ***et*** $\exists\, j \neq \{1\ ;\ 5\}$ ***et*** $j \neq i$ tel que $T_j(t\text{-}1)$ $\neg$en alarme ***et*** $\exists\, V_{ij}$ ***et*** $V_{ij}(t\text{-}1)$ fermée ***alors*** $V_{ij}(t)$ peut être ouverte.

R2 : ***si*** $T_2(t\text{-}1)$ $\neg$en alarme ***et*** $V_{25}(t\text{-}1)$ est fermée ***alors*** $V_{25}(t)$ peut être ouverte.

R3 : ***si*** $T_2(t\text{-}1)$ en alarme ***et*** $V_{12}(t\text{-}1)$ est ouverte ***alors*** $V_{12}(t)$ peut être fermée.

Les avantages des systèmes à base de règles sont les suivants :

- la facilité de mise en œuvre.

- la déclarativité des connaissances, qui est indispensable pour pouvoir modifier facilement un système comportant une grande quantité de connaissances. On ne donne qu'une seule fois une connaissance sous forme déclarative, un seul changement suffit pour la modifier même si elle intervient en de nombreuses occasions.

- la possibilité d'accroître l'expressivité grâce à l'apport de la logique floue (quantificateurs flous, implications floues, modus ponens généralisé, règles graduelles, agrégation des conclusions des règles, …) [Gacôgne 97] [Bouchon-Meunier 95], mais il en résulte une résolution plus complexe et des temps de calcul plus longs.

- la capacité à fournir des preuves et des rapports sur la manière dont le raisonnement est mené, ce qui est très important dans le cadre de l'interaction homme-machine pour que l'opérateur ait confiance dans le système [Moray 97].

Cependant, les inconvénients d'un système de règles sont nombreux :

- généralement on utilise la logique des propositions pour la simplicité des algorithmes de résolution, mais cela empêche de définir des règles dont les prémisses (c'est-à-dire les états dans lesquels on peut appliquer la règle) soient généralisées. Pour cela, il faut utiliser la logique du 1er ordre (voir par exemple la règle R1) afin d'introduire des variables et des prédicats mais cela complique nettement la résolution [Haton 91].

- un manque de flexibilité et de modularité pour représenter les connaissances.

- un mode uniforme de représentation (lié au point précédent). En effet, une caractéristique importante du raisonnement humain est de changer de formulation d'un problème lorsque celle-ci n'est pas adéquate. Or, la difficulté d'un problème varie beaucoup avec sa formulation.

- la difficulté à être sûr que toute l'expertise pertinente est extraite de l'opérateur humain. En outre un système industriel évolue, ce qui nécessite des modifications des règles, opération d'autant plus délicate que les interactions entre règles sont nombreuses [Moray 97].

- dans des situations de travail, l'expert effectue rarement des raisonements profonds mais procède davantage par des schémas de décision basés sur la reconnaissance d'expériences passées similaires à la situation courante [Moray 97].

- un manque de pertinence car l'essentiel de nos connaissances sont en fait des méta-connaissances indiquant quand et comment utiliser une connaissance, et quand ne pas utiliser une connaissance ("l'expertise négative" [Minsky 94]).

- l'absence de prise en compte du but à atteindre dans les moteurs d'inférence fonctionnant en chaînage avant, ce qui entraîne souvent une asphyxie du système. Une façon de traiter ce problème est de mêler chaînage avant et chaînage arrière.

- quasi-impossibilité de s'adapter à la situation courante si cela n'a pas été explicitement prévu par une règle : un humain, pour s'adapter, va utiliser de nombreuses expériences passées qu'il va combiner.

- le raisonnement à base de règles s'est très peu préoccupé des processus pourtant essentiels que sont la mise en correspondance et l'instanciation. Le *matching process* des conditions des règles consiste simplement à vérifier que l'objet testé est identique ou appartient à la classe de l'objet de la condition.

- les effets de bord : lorsque l'on modifie une règle ou lorsque l'on en ajoute une nouvelle, cela peut parfois avoir des effets négatifs difficilement prévisibles et contrôlables lorsque la base comporte un grand nombre de règles (il s'agit là d'un probème que l'on rencontre également dans les systèmes CBR lorsque le nombre de cas est grand).

Même si une façon de résoudre en partie ces problèmes est d'intégrer des méta-connaissances et des connaissances procédurales ainsi que de modulariser les règles, ce type de modèle de raisonnement ne serait pas d'un grand intérêt pour résoudre notre problème, car il semble être surpassé par le modèle de raisonnement à base de cas, et l'apprentissage pose des problèmes sérieux (voir chapitre 6). Mais le raisonnement à base de règles pourrait cependant permettre, en complément du raisonnement à base de modèles, de modéliser les connaissances profondes du domaine utilisées par le système CBR, et constituer ainsi une coopération intéressante entre ces trois types de raisonnement.

Apprentissage par renforcement

On peut également appliquer la méthode dite d'apprentissage de réflexes par renforcement [Cornuejols et Miclet 02] pour résoudre le problème de supervision de réservoirs.
L'agent est vu comme réalisant une fonction de *E* (ensemble des états) dans *A* (ensemble des actions) : $s_t \rightarrow a_t$. On appelle politique cette fonction de comportement.

Plus généralement, ***une politique*** est une fonction de E x A $\rightarrow \mathbb{R}$ qui à chaque état *s* et à chaque action possible *a* dans *s* associe la probabilité $\pi(s, a)$ de choisir *a* dans *s*. ***L'environnement*** est une fonction de E x A $\rightarrow$ E x $\mathbb{R}$: $(s_t, a_t) \rightarrow (s_{t+1}, r_t)$ où r_t est le signal de renforcement reçu par l'agent à l'instant *t*.
Souvent, on décompose cette fonction en deux fonctions :
une fonction de transition entre états de E x A $\rightarrow$ E : $(s_t, a_t) \rightarrow s_{t+1}$
et une fonction de renforcement immédiat de E x A $\rightarrow \mathbb{R}$: $(s_t, a_t) \rightarrow r_t$.

L'agent va alors chercher à maximiser une certaine espérance de gain. Mais tout le problème est ici, comme nous l'avons déjà signalé (voir 3.2.5), de définir un signal de renforcement de bonne qualité.

● La coopération entre agents donne des résultats sensiblement identiques à ceux de l'agent à base de cas seul. Un travail futur consistera donc à définir une meilleure complémentarité entre le raisonnement qualitatif et le raisonnement à base de cas. Les problèmes actuels sont l'indépendance (voir 2.4.2) entre les agents plutôt que la complémentarité (situation de collaboration simple ou coordonnée : voir 2.4.2), et d'autre part la relative faiblesse de l'agent qualitatif. On pourrait alors chercher à améliorer cet agent, mais le problème de l'indépendance entre les deux demeurera, car ils opèrent tous deux au même niveau. Une autre manière de faire et qui semble plus prometteuse a été évoquée au paragraphe 4.2.2.3. (approche intégrée entre un système CBR et d'autres systèmes d'IA) : un système de raisonnement à base de cas peut tout à fait utiliser des connaissances générales du domaine (*background domain knowledge*) en plus des connaissances représentées par les cas, dans le but évident d'améliorer les différentes phases du système CBR. Il s'agit donc, dans le cadre de notre étude, de placer les agents artificiels à différents niveaux hiérarchiques au sein du système à superviser. Un modèle de raisonnement causal (MBR) pourrait jouer le rôle de connaissances générales du domaine, par exemple en modélisant des concepts physiques liées à l'hydraulique, afin d'aider lors des différentes phases de la résolution du processus CBR, et notamment lors de l'apprentissage de cas, afin de généraliser un cas ou un index (5.5.4.) : expliquer pourquoi un paramètre (diamètre, capacité, …) n'a pas d'influence, trouver un concept commun modélisant les conditions physiques permettant d'appliquer une action. Mais comme cela a été dit, ces problèmes font appel à un grand nombre de connaissances de

sens physique commun qu'il faut combiner d'une certaine façon au sein d'un raisonnement. En outre, le modèle qualitatif causal pourrait être complété par des connaissances représentées sous le formalisme de la logique du premier ordre, qui constitueraient alors un raisonnement à base de règles (RBR), ou bien qui joueraient le rôle de théorie du domaine à partir de laquelle un exemple serait généralisé à partir d'explications, rejoignant ainsi le principe de la classification à partir de connaissances (*Eplanation-Based Learning* EBL). Cette intégration CBR – MBR - RBR permettrait ainsi se rapprocher d'un système de collaboration coordonnée (voir 2.4.2).
A ce sujet nous pouvons citer les travaux de [Karamouzis et Feyock 92] qui s'intéressent au raisonnement sur les systèmes physiques, au sens ensemble de composants interconnectés de façon à remplir une fonction. Les problèmes généralement associés à ce type de raisonnement sont la modélisation du comportement normal, le diagnostic de fautes, et la prédiction du comportement futur. Le système CBR comporte des cas représentés par des symptômes observés, sous la forme de séquences temporelles, ainsi qu'une explication causale. Le système de raisonnement à base de modèles (*Model-Based Reasoning* - MBR) comporte les connaissances profondes du domaine telles que les dépendances physiques et fonctionnelles entre les composants (l'information causale représentant les transitions entre états) et des (sous) modèles dynamiques. Lors du fonctionnement du système, la mise en correspondance entre un cas nouveau et un cas en mémoire est facilitée par l'utilisation du modèle MBR qui peut traiter les symptômes qui apparaissent différemment mais qui sont liés à une même cause, ou qui sont tels que l'un est la cause de l'autre. Le système possède également des règles d'adaptation destinées à combler les éventuels liens causaux manquants au sein du cas nouveau. Il est à noter que le système, en plus d'enregistrer le cas nouveau résolu, peut également augmenter et modifier le modèle MBR. De plus, les liens causaux au sein du cas nouveau qui sont déjà dans le modèle voient leur fréquence augmentée, traduisant le fait que tel lien causal devient plus sûr.
Une autre voie expérimentée par cet auteur est d'utiliser un modèle décrivant le domaine *via* des équations. Le comportement de ces équations est prédit à partir des conditions initiales, qui ici ne sont pas des valeurs mais des intervalles. Le processus de mise en correspondance du système CBR est donc lié à l'appartenance des valeurs des variables initiales à des intervalles. Le fonctionnement du système CBR permet de raffiner les intervalles par apprentissage.

● L'évaluation de la plausibilité cognitive de modèles d'IA passe par une comparaison du raisonnement mené par un opérateur humain et celui mené par un agent artificiel. Cette comparaison nécessite de définir une "distance" inter-raisonnement, qui peut être définie de façon numérique, ou mieux grâce à une méthode permettant de juger de l'adéquation d'une séquence d'actions d'un opérateur humain avec le comportement d'un agent artificiel (voir 7.1.1.2). Dans ce cadre, un travail futur, afin d'améliorer la mise en adéquation floue entre une action de l'opérateur humain et une intention, serait de définir les zones floues (zone d'alarme, zone optimale pour une vidange de réservoirs selon la même durée afin de minimiser la durée totale de la simulation, …) non pas de manière un peu arbitraire en fonction des paramètres de l'agent artificiel, mais uniquement en fonction du comportement de l'opérateur humain, à l'aide notamment de ses actions jugées non litigieuses, c'est-à-dire dont l'interprétation en terme d'intention ne fait *a priori* aucun doute.

● L'étude passe également par la définition du pas de temps de calcul de l'agent artificiel Δt (somme des durées d'observation et de réflexion mêlées et de la durée d'une action) afin d'améliorer encore la cohérence avec le raisonnement humain. Ce Δt est très variable chez un humain selon l'urgence ou non de la situation. Un travail futur pourrait consister à définir ce

pas de temps en fonction de la situation (voir 2.1.3), ce qui nécessiterait une capacité d'anticipation de la part du système.

BIBLIOGRAPHIE

[Aamodt et Plaza 94] AAMODT A., PLAZA E. *Case-based reasoning : foundational issues, methodological variations, and system approaches*. AI Communications IOS Press, 1994.

[Aarts et Lenstra 97] AARTS E., LENSTRA J.K. *Local search in combinatorial optimization*. John Wiley and Sons Inc., 1997, 512 p.

[Abdi 94] ABDI H. *Les réseaux de neurones*. Presses universitaires de Grenoble, 1994, 269 p.

[Amalberti et Cacciabue 90] AMALBERTI R., CACCIABUE P.C., VALOT C., DECORTIS F., DOSDROWITZ B. *Modelling preferences in process control – The importance of metaknowledge*. JRC ISPRA-CERMA, 1990.

[Amalberti 91] AMALBERTI R. *Savoir-faire de l'opérateur : aspects théoriques et pratiques en ergonomie*. Dans : Modèles en analyse de travail. Amalberti R., Montgollin M., Theureau J. (Eds.). Liège, Mardaga, 1991.

[Anderson 89] ANDERSON J.R. *A theory of the origins of human knowledge*. Dans : Artificial Intelligence, volume 40, Elsevier Science Publishers, 1989.

[Bainbridge 78] BAINBRIDGE L. *Le contrôleur de processus*. Bulletin de psychologie, tome XXXIV n° 352, 1978.

[Barsalou 91] BARSALOU Lawrence. *Deriving categories to achieve goals*. Dans : The psychology of learning and motivation, volume 27. Academic Press, 1991.

[Basalou 92] BARSALOU Lawrence. *Frames, concepts and conceptual fields*. Dans : Frames, fields and contrasts New essays in semantic and lexical organization. Lawrence Erlbaum Associates Publishers, 1992.

[Barsalou 93] BARSALOU Lawrence, HALE Christopher. *Components of conceptual representation : from feature lists to recursive frames*. Dans : categories and concepts : theorical views and inductive data analysis. San Diego Academic Press, 1993.

[Barsalou 02] BARSALOU L. *The HIPE theory of function*. Dans : Representing functional features for language and space: insights from perception, categorization and development. Oxford University Press, 2002.

[Bartak 98] BARTAK Roman. *Guide to Constraint Programming*. http://ktiml.mff.cuni.cz/~~bartak/constraints, 1998.

[Bart et Frigot 02] BART Pascal, FRIGOT Eric. *Raisonnement à partir de cas*. Dossier EPITA http://www.calia.org/dossiers/cbr.pdf, 2002.

[Bergmann *et al.* 96] BERGMANN Ralph, WILKE Wolfgang, VOLLRATH Ivo, WESS Stefan. *Integrating general knowledge with object-oriented case representation and reasoning*. Dans : Hans-Dieter Burkhard and Mario Lenz (Eds.), 4th German Workshop on CBR - System Development and Evaluation - , Berlin, 1996.

[Bergmann et Wilke 97] BERGMANN Ralph, WILKE Wolfgang. *On the role of abstraction in case-based reasoning*. Dans : Case-Based Reasoning Research and Development : 2th International Conference on Case-Based Reasoning, Spinger-Verlag, Berlin,1997.

[Blumenthal et Porter 94] BLUMENTHAL Brad, PORTER Bruce. *Analysis and empirical studies of derivational analogy*. Journal of Artificial Intelligence, volume 67, 1994.

[Bouchon-Meunier 95] BOUCHON-MEUNIER Bernadette. *La logique floue et ses applications*. Addison Wesley, 1995, 257 p.

[Cacciabue 90] CACCIABUE P.C., MANCINI G., BERSINI U. *A model of operator behaviour for man-machine system simulation*. Automatica, vol. 26, n° 6, 1990.

[Chandrasekaran et al. 89] CHANDRASEKARAN B., TANNER C., JOSEPHSON J.R. *Explaining control strategies in problem solving*. IEEE Expert, 1989.

[Charon *et al.* 96] CHARON Irène, GERMA Anne, HUDRY Olivier. *Méthodes d'optimisation combinatoire*. Masson, 1996, 268 p.

[Clancey 85] CLANCEY W.J. *Heuristic classification*. Artificial Intelligence vol. 27, 1985.

[Clark et Porter 01] CLARK Peter, POTER Bruce. *KM – the Knowledge Machine 2.0 : users manual*. http://www.cs.utexas.edu/users/mfkb/km/userman.pdf Technical report, AI Lab., University Texas at Austin, 2001.

[Cordier *et al.* 00] CORDIER M.O., DAGUE P., LEVY F., MONTMAIN J., STAROSWIECKI M., TRAVE-MASSUYES L. *AI and automatic control approaches of model-based diagnosis : links and underlying hypotheses*. IFAC'00, 2000.

[Cornuejols et Miclet 02] CORNUEJOLS Antoine, MICLET Laurent. *Apprentissage artificiel – concepts et algorithmes*. Eyrolles, 2002, 591 p.

[De Kleer et Brown 84] DE KLEER J., BROWN J.S. *A qualitative physics based on confluences*. Artificial Intelligence volume 24, 1984.

[Ferber 95] FERBER J. *Les systèmes multi-agents. Vers une intelligence collective*. InterEditions, 1995, 522 p.

[Forbus 86] FORBUS K.D. *The role of qualitative dynamics in naive physics*. Qualitative reasoning about physical systems, Elsevier Science, 1986.

[Ginsberg *et al.* 96] GINSBERG Matthew, CRAWFORD James, ETHERINGTON David. *Dynamic Backtracking*. Final Technical Report 8/1/96. CICRL, University of Oregon, 1996.

[Gacôgne 97] GACÔGNE Louis. *Eléments de logique floue*. Hermes, 1997, 253 p.

[Hanafi 00] HANAFI S. *On the convergence of Tabu Search*. Journal of Heuristics, vol. 7, 2000.

[Haton 91] HATON Jean-Paul. *Le raisonnement en intelligence artificielle*. InterEditions, 1991, 480 p.

[Hayes-Roth *et al.* 83] HAYES-ROTH F., WATERMAN D.A., LENAT D.B. *Building Expert systems*. Addison-Wesley, 1983.

[Hoc 98] HOC J.M. *Modèles cognitifs de l'opérateur humain*. Journées "Automatique et Homme". LAMIH Valenciennes, septembre 1998.

[Holyoak et Thagard 89] HOLYOAK K.J., THAGARD P.R. *A computational model of analogical problem solving*. Dans : Similarity and analogical reasoning. Vosniadou S., Ortony A. (Eds.). Cambridge University Press, 1989.

[Hollnagel 93] HOLLNAGEL E. *Models of cognition : procedural prototypes and contextual control*. Dans : Le travail humain, tome 56, n° 1, pp. 27-51, 1993.

[Iwasaki et Simon 86] IWASAKI Y., SIMON H.A. *Causality in device behavior*. Artificial Intelligence, volume 29, 1986.

[Jimenez Molina 00] JIMENEZ MOLINA J.A. *The design and implementation of the fault-isolation module for the causal simulator Ca-EN*. Final project for the computer engineering diploma of the university of Saragosse, Espagne, 2000.

[Karamouzis et Feyock 92] KARAMOUZIS Stamos, FEYOCK Stefan. *An integration of case-based and model-based reasoning and its application to physical system faults*. 5th International Conference IEA/AIE, Paderborn, Allemagne, juin 1992.

[Kayser 97] KAYSER Daniel. *La représentation des connaissances*. Hermes, 1997, 308 p.

[Kerber *et al.* 92] KERBER Manfred, MELIS Erica, SIEKMANN Jörg. *Analogical reasoning with typical examples*. SEKI Report SR-92-13, 1992.

[Kerber *et al.* 93] KERBER Manfred, MELIS Erica, SIEKMANN Jörg. Analogical reasoning with a hybrid knowledge base. IJCAI – workshop on principles of hybrid reasoning and representation, Chambéry, 1993.

[Kolodner et Leake 96] KOLODNER Janet, LEAKE David. *A tutorial introduction to case-based reasoning*. Dans : Case-based reasoning – experiences, lessons & future directions. MIT Press, 1996, 420 p.

[Kuipers 84] KUIPERS B. *Commonsense reasoning about causality : deriving behaviour from structure*. Artificial Intelligence volume 24, 1984.

[Leake 93] LEAKE D. *Learning adaptation strategies by introspective reasoning about memory search*. AAAI Workshop on case-based reasoning, 1993.

[Leake 94] LEAKE D. *Issues in goal-driven explanation*. AAAI Spring Symposium on goal-driven learning, 1994.

[Leake 95 (a)] LEAKE D. *Representing self-knowledge for introspection about memory search*. AAAI Spring Symposium on Mental states and mechanisms, 1995.

[Leake 95 (b)] LEAKE David. *Abduction, experience and goals : a model of everyday abductive explanation*. The Journal of experimental and theorical Artificial Intelligence, 1995.

[Leake 96 (a)] LEAKE David. *Linking adaptation and similarity learning*. 18th annual Conference of the Cognitive Science Society, 1996.

[Leake 96 *et al.* (b)] LEAKE David, KINLEY Andrew, WILSON David. *Multistrategy learning to apply cases for case-based reasoning*. 3th International Workshop on Multistrategy Learning, 1996.

[Lieber et Napoli 99] LIEBER Jean, NAPOLI Amadeo. *Raisonnement à partir de cas et résolution de problèmes dans une représentation par objets*. Dans : RIA volume 13 – Raisonnement à partir de cas, 1999.

[Maclin et Shavlik 89] MACLIN Richard, SHAVLIK Jude. *Using explanation-based learning to acquire programs by analysing examples*. University of Wisconsin Computer Science Technical Report 858, 1989.

[Mantaras et Plaza 97] DE MANTARAS Ramon Lopez, PLAZA Enric. *Case-based reasoning : an overview*. AI Communications Journal.1997

[Mille *et al.* 99] MILLE Alain, FUCHS Béatrice, CHIRON Benoît. *Raisonnement fondé sur l'expérience : un nouveau paradigme en supervision industrielle*. Dans : RIA volume 13 – Raisonnement à partir de cas, 1999.

[Minsky 74] MINSKY Marvin. *A Framework for Representing Knowledge*. MIT-AI Laboratory Memo 306, juin 1974.

[Minsky 85] MINSKY Marvin. *La société de l'esprit*. InterEditions, 1985, 653 p.

[Minsky 94] MINSKY Marvin. *Negative Expertise*. International Journal of Expert Systems, volume 7, n° 1, p. 13-19, 1994.

[Minsky 03] MINSKY M.. *The emotion machine*. http://web.media.mit.edu/~minsky/, 2003.

[Moray 97] MORAY N. *Human factors in process control*. Dans : Handbook of human factors and ergonomics. 2nd edition. Salvendy G. (Eds.). John Wiley and Sons Inc., 1997.

[Newell 90] NEWELL A. *Unified theories of cognition*. Harvard University Press, 1990.

[Piera *et al.* 89] PIERA N., DESROCHES P., AGUILAR MARTIN J. *Lamda : an incremental conceptual clustering method*. Rapport LAAS n° 89420, 1989.

[Pitrat 93] PITRAT Jacques. *Penser autrement l'informatique*. Hermes, 1993, 206 p.

[Ponsa et Catala 99 (a)] PONSA P., CATALA A. *Implementation of Artificial Reasoners within a micro-world generator framework*. Rapport n° 99585 LAAS-CNRS, Toulouse, France, novembre 1999.

[Ponsa et Catala 99 (b)] PONSA P., CATALA A. *Human supervision in industrial process*. 7th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'99), Barcelone, Espagne, 18-21 octobre 1999.

[Ponsa *et al.* 00] PONSA P., CATALA A, TRAVE-MASSUYES. *Matlab utilities : the naïve qualitative agent connected to a micro-world framework*. Rapport n° 00277 LAAS-CNRS, Toulouse, France, novembre 2000.

[Ponsa *et al.* 02] PONSA P., CATALA A, TRAVE-MASSUYES L., DIAZ M. *Assessing human supervision strategies with qualitative reasoning techniques*. 15th IFAC, Barcelone, Espagne, 21-26 juillet, 2002.

[Ram et Leake 94] RAM Ashwim, LEAKE D. *A framework for goal-driven learning*. AAAI Spring Symposium on goal-driven learning, 1994.

[Rasmussen 83] RASMUSSEN J. *Skills, rules, knowledge : signals, signs and symbols and other distinctions in human performance models*. IEEE transactions on systems, man and cybernetics, 1983.

[Reiter 87] REITER R. *A theory of diagnosis from first principles*. Artificial Intelligence, vol. 32, 1987.

[Rich 87] RICH Elaine. *Intelligence Artificielle*. Masson, 1987, 439 p.

[Sabah 88] SABAH Gérard. *L'intelligence artificielle et le langage - représentation des connaissances - volume 1*. 2ième édition. Hermes, 1988, 357 p.

[Salazar-Ferrer 95] SALAZAR-FERRER Pascal. *Raisonnement causal et modélisation de l'activité cognitive d'opérateurs de chaufferie nucléaire navale*. Thèse : Psychologie cognitive, université de Provence, 1995, 228 p.

[Salotti *et al.* 99] SALOTTI Sylvie, VENTOS Véronique. *Une approche formelle du raisonnement à partir de cas dans une logique de descriptions*. Dans : RIA volume 13 – Raisonnement à partir de cas, 1999.

[Schank 99] SCHANK R.C. *Dynamic Memory revisited*. Cambridge University Press, 1999.

[Schmid et Carbonell 99] SCHMID Ute, CARBONELL Jaime. *Empirical evidence for derivational analogy*. 21st Annual Conference of the Cognitive Science Society, Simon Fraser University-Vancouver, British Columbia, august 19-21, 1999.

[Smith et Cunningham 93] SMITH Barry, CUNNINGHAM Padraig. *Complexity of adaptation in real-world case-based reasoning systems*. 6th Irish Conference on Artificial Intelligence and Cognitive Science, 1993.

[Taillard *et al.* 96] TAILLARD E., BADEAU P., GENDREAU M., GUERTIN F., POTVIN J.Y. *A Tabu Search heuristic for the Vehicle Routing Problem with soft time windows*. Rapport technique, CRT, Université de Montréal, 1996, 35 p.

[Thuriot *et al.* 01] THURIOT C., TRAVE-MASSUYES L., CELLIER J.M. *A framework for cognitive validation of Qualitative Reasoning concepts in process supervision Human System Interface*. IFAC-HMS'01, 2001.

[Torasso 04] TORASSO Pietro. *Case-based reasoning in diagnostic problem solving : alternative or complementary to MBR*. DX-04, 15th International Workshop on Principles of Diagnosis, Carcassonne, France, 23-25 juin, 2004.

[Trave-Massuyes et Pons 97] TRAVE-MASSUYES L., PONS R. *Causal ordering for multiple mode systems*. 11th International Workshop on Qualitative Reasoning, Cortona, Italie, 3-6 juin, 1997.

[Trave-Massuyes *et al.* 99] TRAVE-MASSUYES Louise, PRATS Francesc, SANCHEZ Monica, AGELL Nuria, PASTOR Josette. *Qualitative Agents for assessing human reasoning in Process Supervision*. 13th QR'99, Loch Awe, Ecosse, 6-9 juin, 1999.

[Trave-Massuyes *et al.* 01] TRAVE-MASSUYES L., ESCOBET T., PONS R., TORNIL S. *The Ca~En diagnosis system and its automatic modelling method*. Workshop on Diagnosis, Qualitative Reasoning and Socio-Economic Systems, Valladolid, Espagne, juillet 2001.

[Trave-Massuyes et Dague 03] TRAVE-MASSUYES L., DAGUE P. *Modèles et raisonnements qualitatifs*. Hermes, 2003.

[Vanderhaegen 99] VANDERHAEGEN F. *Cooperative system organisation and task allocation : illustration of task allocation in air traffic control*. Dans : Le travail humain, tome 62, n° 3, pp. 197-222, 1999.

[Vanderhaegen et al. 04] VANDERHAEGEN F., MILLOT P., CHALME S. *Towards models of cooperation versus competition to analyze car drivers behaviors*. IEEE International Conference on Systems, Man and Cybernetics. La Hague, Pays-Bas, octobre 2004.

[Vila 92] VILA L. *A survey of temporal reasoning in Artificial Intelligence*. Draft, 2nd version, 1992.

[Waissman 00] WAISSMAN-VILANOVA J. *Construction d'un modèle comportemental pour la supervision de procédés : application à une station de traitement des eaux*. Thèse : systèmes automatiques, INPT, 2000, 119 p.

[Weld et De Kleer 90] WELD D., DE KLEER J. *Readings in qualitative reasoning about physical systems*. Morgan Kaufmann, San Mateo, CA, 1990.

[Wielinga *et al.* 92] WIELINGA B.J., SCHREIBER A.Th., BREUKER J.A. *KADS, a modelling approach to knowledge engineering*. Knowledge acquisition volume 4, 1992.

ANNEXES

Annexe 1. Recherche guidée par l'adaptabilité

Il s'agit de l'approche proposée dans [Lieber et Napoli 99].

Un cas source K est représenté sous la forme d'un problème (énoncé) et de sa solution associée : $K = (source, sol(source))$.
Un index (noté *idx*) de K est défini comme une **généralisation** de l'énoncé (*source*).
L'énoncé d'un problème est représenté sous la forme d'une conjonction $C = (a_1, s_1) \wedge \ldots \wedge (a_n, s_n)$ où les a_k désignent des propriétés (ou attributs) et les s_k des spécifications attachées à la propriété précisant par exemple le type, le domaine et la cardinalité des valeurs de la propriété. Par exemple a_k peut désigner l'âge d'une personne et s_k une condition $a_k > 40$.
Une hiérarchie des index est définie par le biais d'une **relation de généralité** : l'énoncé du problème P est plus général que l'énoncé du problème P' (noté $P \sqsubseteq P'$) si pour chaque couple attribut-spécification (a, s) dans P, il existe un couple attribut-spécification (a, s') dans P' tel que $s' \Rightarrow s$ (implication logique).

Le raisonnement à base de cas s'effectue de la manière suivante :

Soit un problème cible = (*cible, sol(cible)*) à résoudre. On classifie l'énoncé *cible* dans la hiérarchie des index, c'est-à-dire qu'on recherche des énoncés *idx(source)* subsumants (plus généraux que) *cible*, ce qui met en évidence *chemin de similarité* entre la *source* et la *cible* :

sim(*source*, *cible*) = *source* $\sqsubseteq$ *idx(source)* $\sqsupseteq$ *cible*

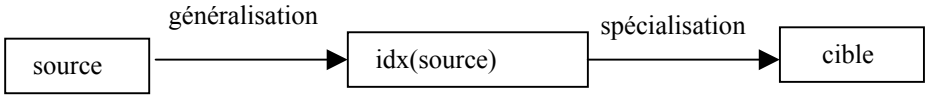


Figure 56 : Chemin de similarité entre un problème source et un problème cible

Si on a trouvé une *source*, alors on adapte sa solution par une suite d'opérations de *généralisation* et de *spécialisation*. Plus précisément **l'adaptation** consiste à généraliser *sol(source)* pour produire la solution *sol(idx(source))*, puis à spécialiser *sol(idx(source))* en *sol(cible).*

Si on n'a pas trouvé de *source* par le processus de classification précédent, alors on va rechercher en mémoire des *sources* telles qu'en appliquant des *déformations* (fonctions Φ et Ψ) à *cible* et à *source*, on puisse se ramener au cas précédent pour appliquer des opérateurs d'adaptation, ce qui met en évidence un autre type de *chemin de similarité* :

sim(*source*, *cible*) = *source* $\sqsubseteq$ *idx(source)* $\rightarrowtail$ Φ*(idx(source))* $\sqsupseteq$ Ψ*(cible)* $\leftarrowtail$ *cible*

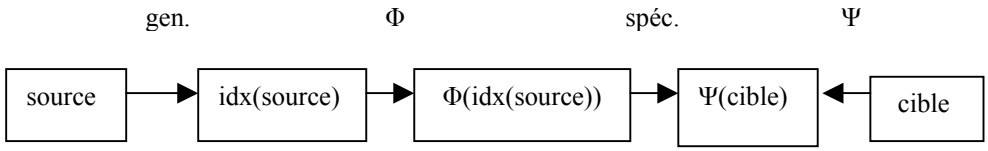


Figure 57 : Chemin de similarité à l'aide de fonctions de déformation

Φ doit posséder la propriété suivante : si on connaît une solution *sol(idx(source))* alors on doit pouvoir construire une solution de *sol(Φ(idx((source)))*. Φ dépend du domaine, mais généralement il s'agit d'une généralisation ou d'une transformation d'état.
Ψ doit posséder la propriété suivante : si on connaît une solution *sol(Ψ(cible))* alors on doit pouvoir construire une solution de *sol(cible)*. Ψ dépend du domaine, mais généralement il s'agit d'une transformation d'état.

Dans la pratique, on a préalablement créé un ensemble de fonctions de déformations Φ_i et Ψ_i et il s'agit de trouver deux séquences, optimales selon un critère, et qui satisfassent la contrainte :

$$A \rightarrowtail \Phi_1(A) \rightarrowtail \ldots \rightarrowtail \Phi_n(A) \sqsupseteq \Psi_m(B) \leftarrowtail \ldots \leftarrowtail \Psi_1(B) \leftarrowtail B$$

Avec $A = idx(source)$ et $B = cible$.

Pour cela, un algorithme *A** [Rich 87] est utilisé afin de rechercher le cas source qui minimise le coût du chemin entre source et cible, en exploitant une fonction d'évaluation et en effectuant une recherche bidirectionnelle dans un espace d'états :

$$eval(X, Y) = coût(A \rightarrowtail \ldots \rightarrowtail X) + coût(Y \leftarrowtail \ldots \leftarrowtail B) + estim(X, Y)$$

L'état (X, Y) est un état but si $X \sqsupseteq Y$ (c'est-à-dire X plus général que Y).

Les auteurs précisent que la recherche de cas similaires ne prend en compte que l'état initial du problème ou l'énoncé (on recherche en mémoire les cas dont l'état initial est plus général que l'état initial du problème à résoudre). Il serait bienvenu de prendre également en compte le but du problème ou sa solution. Les auteurs indiquent une façon de procéder :

Soit un problème $P = (Init_P, Oper, But_P)$ avec :
$Init_P$ est l'état initial (e_0).
$Oper = \{o_i, i = 1 \text{ à } f\}$ est un ensemble d'opérateurs qui permettent de passer d'un état à un autre.
But_P est l'état final (e_f), le but à atteindre.

$$sol(P) = e_0 \xrightarrow{o_1} e_1 \xrightarrow{o_2} \ldots \xrightarrow{o_f} e_f$$

On dit alors qu'un problème $P = (Init_P, Oper, But_P)$ est plus général qu'un problème $P' = (Init_{P'}, Oper, But_{P'})$ si : l'énoncé $Init_P$ est plus général que $Init_{P'}$ et le but But_P est plus spécifique que $But_{P'}$.

Remarque : l'insertion d'un objet (un cas appris) au sein de la hiérarchie existante se fait de manière classique en recherchant les subsumants les plus spécifiques (*SPS*) et les subsumés les plus généraux (*SPG*), puis en effectuant des mises à jour des arcs.
Par exemple, lors de l'insertion de *BC* dans la hiérarchie *Ω*, les *SPS* de *BC* sont *B* et *C*, les *SPG* sont *ABC* et *BCD*, les arcs créés sont en gras, et les arcs *ABC→B* et *BCD→B* sont supprimés.

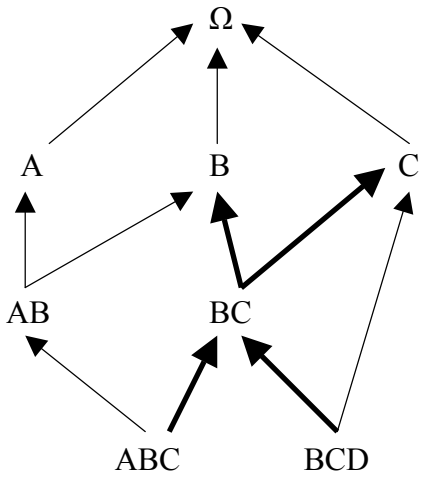


Figure 58 : Insertion de l'objet BC dans la hiérarchie Ω

Annexe 2. Phase de recherche de cas anciens dans le cadre de la supervision industrielle, où un cas est représenté par un épisode

Il s'agit de l'approche proposée dans [Mille *et al.* 99].

1- Sélection des cas compatibles avec la situation courante.

2- Appariement des cas proches par similarité conceptuelle :

$$S_c(x,y) = \frac{1}{2}.\left(\frac{1}{2^{NS(x,P_c(x,y))}} + \frac{1}{2^{NS(y,P_c(x,y))}}\right)$$

$P_c(x,y)$ est le concept parent commun à x et y. Par définition, $P_c(x,x) = x$.
$NS(x,C)$ est le niveau de spécialisation d'une instance x par rapport à un concept C (par exemple le nombre d'arêtes qui les séparent dans la hiérarchie). Ainsi $NS(x,x) = 0$.

On définit alors la similarité conceptuelle de deux cas, c'est-à-dire deux environnements de supervision ES_n et ES_m (n et m sont les nombres d'objets respectifs des deux environnements, et on note $ES_n \cap ES_m$ = ensemble des couples d'objets compatibles dans les deux environnements) :

$$S_{ES}(ES_n, ES_m) = \frac{\sum_{i=1}^{i=n} \sum_{j=1}^{j=|ES_n \cap ES_m|} S_c(x_i, y_j)}{n.|ES_n \cap ES_m|}$$

3- Sélection du meilleur cas par dissimilarité événementielle.
On va rechercher un cas qui a une histoire événementielle similaire.

Soient :
$e^c_1, \ldots, e^c_n$ les événements de la situation courante S^c,
$e^r_1, \ldots, e^r_m$ les événements de la situation retrouvée S^r,
$rg(e)$ le rang dans S^r de l'événement e,
$rg(e^c_i)$ le rang dans S^r de l'événement positionné en i dans S^c.

On a alors :

diss1 = $|S^r| - |S^c|$: il s'agit d'une mesure de la représentativité qui indique à quel point une séquence d'événements est présente dans une autre.

D'autre part :
$diss2 = (rg(e^c_2) - rg(e^c_1) - 1) + (rg(e^c_3) - rg(e^c_2) - 1) + \ldots + (rg(e^c_n) - rg(e^c_{n-1}) - 1)$

diss2 = $rg(e^c_n) - rg(e^c_1) - n + 1$: il s'agit de la dispersion qui indique à quel point la chaîne source se disperse dans la chaîne cible.

la dissimilarité événementielle est ***diss = diss1 + diss2***

Annexe 3. Utilisation de la logique des descriptions pour organiser la base de cas et définir une représentation des similarités et des différences

Il s'agit de l'approche proposée dans [Salotti *et al.* 99].
Les cas sont considérés comme des individus et sont indexés par des concepts appelés *concepts indices* qui sont organisés en une hiérarchie. Ces concepts représentent les catégories significatives pour organiser les cas, et sont définis par des conjonctions de propriétés.

Les auteurs définissent également une *base de connaissances du domaine* grâce à des concepts généraux qui font le lien entre la description concrète d'un nouveau cas et les concepts indices.

Lorsque le système est face à un cas nouveau, il procède de la façon suivante :

La recherche de cas similaires se fait en retrouvant les concepts indices les plus spécifiques qui subsument le nouveau cas, puis en sélectionnant les cas qui appartiennent à au moins un des concepts indices précédents. On obtient donc un premier ensemble de cas E_{sim}.

Une seconde étape permet d'affiner cette sélection, grâce à la définition d'un *critère de similarité* (la similarité entre deux concepts peut être caractérisée par l'ensemble des propriétés qu'ils ont en commun, c'est-à-dire par le plus petit généralisant - *PPG)* et d'un *critère de dissimilarité* (la dissimilarité entre deux concepts *A* et *B*, dans cet ordre car l'opération n'est pas symétrique, est l'ensemble des propriétés du concept *A* qui ne subsument pas une propriété de *B*). Il est à noter que ces deux opérations utilisent en entrée deux cas (individus) et fournissent en sortie un concept. Cette sélection se fait en deux étapes :

1- on calcule le *PPG* de chaque cas de E_{sim} avec le nouveau cas. Les *PPG* étant des concepts, on les organise dans une hiérarchie, et on sélectionne les plus spécifiques car ce sont ceux qui ont le plus de propriétés en commun avec le nouveau cas. A partir de ces *PPG* sélectionnés, on définit l'ensemble des cas les plus similaires E_{sim+} : si un *PPG* correspond à un seul cas, alors ce cas appartient à E_{sim+}. Sinon, on essaye de discriminer les cas (appelés C_{old}) de ce *PPG* grâce au critère de dissimilarité.

2- D'après le théorème : si *PPG(A,B) = PPG(A,C) alors DISS(A,C) = DISS(A,B)*
On déduit que tous les C_{old} précédents ont même $DISS(C_{new},C_{old})$. Ainsi, pour tous les C_{old} ayant même *PPG*, on calcule $DISS(C_{old},C_{new})$. Les concepts *DISS* fournis sont ensuite organisés en une hiérarchie. On met alors à jour E_{sim+} en ajoutant les cas correspondant aux *DISS* les plus généraux car ils minimisent la dissimilarité.

Enfin, une dernière étape consiste à intégrer le nouveau cas dans la base. En outre, si l'expert n'est pas satisfait des cas sélectionnés à la fin de l'étape précédente, alors il partage cet ensemble E_{sim+} en un ensemble d'instances positives d'un concept indice à apprendre auquel on ajoute le nouveau cas, et un ensemble d'instances négatives. Un algorithme d'apprentissage de concept est alors lancé de façon à trouver la définition d'un concept indice qui inclue les instances positives et rejète les instances négatives. Le nouveau concept est ensuite inclus dans la hiérarchie de concepts indices grâce au mécanisme de classification.

Annexe 4. Utilisation de systèmes CBR combinés avec des systèmes RBR (*Rule Based Reasoning*) à l'intérieur d'un système CBR

Cette approche est issue des travaux de [Leake 96 (a)] et [Leake *et al.* 96 (b)].
Leurs travaux portent sur la conception d'un système de planification à base de cas.
Un système CBR transformationnel (adaptation par transformation de la solution) génère un plan réponse à un problème donné, après avoir retrouvé dans une mémoire de plans ceux qui sont les plus similaires et choisi un d'entre eux. A l'intérieur de ce système, on a :

- pour l'adaptation : un système CBR dérivationnel qui utilise la trace d'une adaptation antérieure similaire. Ce système comporte donc une mémoire de cas d'adaptation, et un cas est ici défini par une stratégie pas à pas, car il s'agit ici d'une adaptation par dérivation. Mais si on ne trouve pas de cas d'adaptation, on utilise alors un système RBR comportant des règles générales d'adaptation. Mais il faut dans ce cas rechercher en mémoire ces règles. Pour ce faire, on utilise un système CBR dérivationnel (adaptation par analyse de la trace de la solution) qui recherche un cas similaire (c'est-à-dire une stratégie de recherche en mémoire définie pas à pas). Ce système comporte donc une mémoire de cas de recherche en mémoire. Si on n'en trouve pas, on utilise un système RBR comportant des règles de recherche en mémoire. Par exemple, une règle peut indiquer, lorsqu'il s'agit de vérifier les contraintes associées aux acteurs d'un plan antérieur retrouvé et à adapter (c'est-à-dire ici substituer les acteurs de ce plan par d'autres acteurs), de rechercher en mémoire des objets satisfaisants ces contraintes.

- pour définir la similarité : l'idée est que pour sélectionner les cas les plus facilement applicables à la nouvelle situation, les jugements de similarité doivent refléter l'adaptabilité. Ainsi, le système sélectionne initialement des plans réponses en se basant sur un système RBR comportant des règles spécifiques définissant des critères de similarité. Mais lorsque des cas d'adaptation sont appris (système CBR dérivationnel précédent), le système peut alors utiliser un système CBR transformationnel afin d'estimer les coûts d'adaptation en se basant sur un examen des cas d'adaptation. Ainsi, ce système CBR recherche (dans la mémoire des cas d'adaptation précédemment définie) des cas pouvant s'appliquer pour adapter le problème à résoudre (un plan à élaborer) et estime la quantité d'effort requise lors de précédentes adaptations. Cette quantité peut être mesurée par la longueur de la trace dérivationnelle. Le système sélectionnera donc le cas (plan réponse) nécessitant le moins d'adaptation en terme de coût. Il s'agit donc ici d'un processus de recherche guidée par l'adaptabilité.

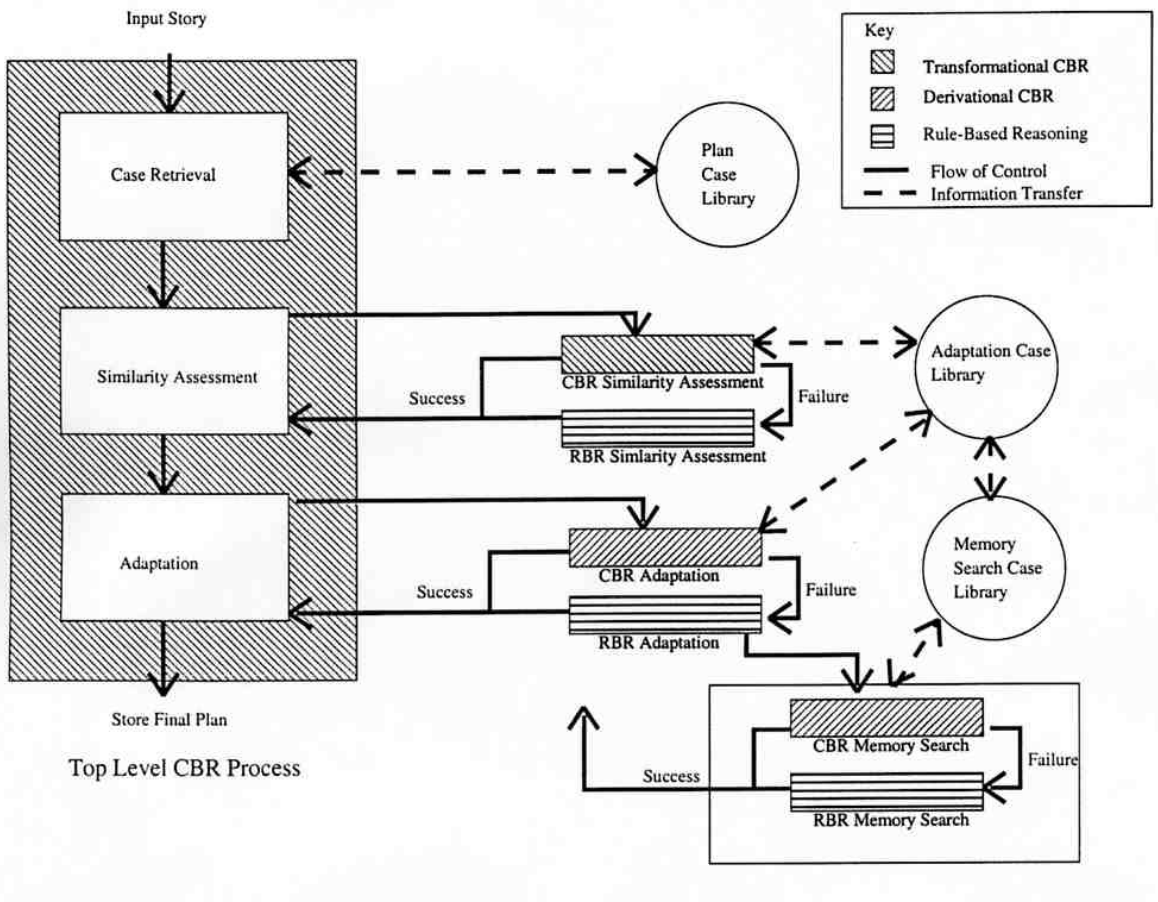


Figure 59 : exemple de combinaison CBR-RBR [Leake 96 et al. (b)]

Annexe 5. Apprentissage de stratégies d'adaptation

Cette approche est issue des travaux de [Leake 93], [Leake 95 (a)] et [Leake 96 (a)].
L'idée de base est la génération et l'apprentissage de procédures de recherche en mémoire. Il s'agit de connaissances sur l'organisation de la mémoire afin de rechercher des informations ne correspondant pas à des liens explicitement pré-codés en mémoire.

Le système démarre avec des règles d'adaptation abstraites et des stratégies simples de recherche en mémoire basées sur l'organisation de cette mémoire. Lorsque la solution au nouveau problème n'est pas adéquate, il s'agit :

- en premier lieu de décrire le point qui nécessite une adaptation. Par exemple, il peut s'agir d'une *mauvaise adéquation au niveau de l'action d'un agent*

- essayer de retrouver une stratégie d'adaptation existante, spécifique et indexée comme pertinente par rapport au point précédent. Initialement, il n'y en a pas puisque ces stratégies sont apprises. Ainsi, si on n'en trouve pas, on recherche alors des règles d'adaptation abstraites applicables (par exemple *remplacer un élément*, *ajouter un élément*, *effacer un élément*). Il s'agit donc également d'avoir un critère afin de décider quelle règle appliquer

- planifier le recherche en mémoire. Par exemple, si le but est de substituer un antécédent causal, on peut appliquer la règle de recherche en mémoire : *pour trouver un antécédent plausible expliquant l'état d'un acteur, rechercher une action plausible que l'acteur peut effectuer afin de causer cet état*. Ceci nécessite alors l'application d'une autre règle : *pour trouver une action plausible effectuée par un acteur, considérer les actions qu'il fait habituellement*, qui génère donc un sous-but. On applique enfin la règle : *pour trouver une action habituelle d'un acteur, parcourir les liens correspondant aux plans et buts associés à cet acteur*. On a alors une chaîne opérationnelle.

- on synthétise ce plan et on l'associe à la règle abstraite sous-jacente. On a alors une nouvelle stratégie d'adaptation. Par exemple, dans le cas précédent, on aurait : *pour remplacer une action effectuée par un acteur, parcourir les liens correspondant aux plans et buts associés à cet acteur*.

Annexe 6. Quelques méthodes de classification par apprentissage et reconnaissance fonctionnant sans connaissances *a priori*

Nous présentons quelques méthodes classiques pouvant être utilisées dans le cadre de notre problème.

l'espace des versions [Cornuéjols et Miclet 02] :

Il s'agit d'induction dans un formalisme de type logique des propositions.
Les attributs peuvent être de type :

- binaires (domaine {vrai, faux}), numériques (par exemple la taille d'un être vivant),

- nominaux hiérarchiques (par exemple le domine hiérarchisé {être vivant, animal, végétal, humain, …})

- nominaux non hiérarchiques (par exemple la couleur dont le domaine est {rouge, vert, bleu, …}).

Dans cette méthode, les exemples sont considérés séquentiellement. Ainsi, à l'étape t, une hypothèse candidate h_t (le concept à exhiber), cohérente avec l'ensemble des exemples déjà passés en revue S_t, a été construite.
Lorsqu'on étudie un nouvel exemple :

- soit il est correctement classé par h_t, donc $h_{t+1} = h_t$,

- soit il est incorrectement classé et deux cas se présentent : si l'exemple est négatif (contre-exemple du concept), il a donc été incorrectement classé comme positif. Il faut donc spécialiser h_t tout en couvrant toujours tous les exemples positifs de S_t (d'où une nouvelle hypothèse h_{t+1}). Par contre, si l'exemple est positif et a donc été incorrectement classé comme négatif, il faut généraliser h_t tout en ne couvrant aucun des exemples négatifs de S_t (d'où une nouvelle hypothèse h_{t+1}).

L'algorithme recherche deux ensembles pour définir le concept : l'ensemble S des hypothèses cohérentes (c'est-à-dire couvrant les exemples positifs et excluant les exemples négatifs) maximalement spécifiques, c'est-à-dire que toute spécialisation de S lui fait perdre sa cohérence, et l'ensemble G des hypothèses cohérentes maximalement générales, c'est-à-dire que toute généralisation de G lui fait perdre sa cohérence.
L'algorithme d'élimination des candidats, détaillée dans [Cornuéjols et Miclet 02], permet de calculer ces deux ensembles.
L'ensemble S constitue donc une borne inférieure de l'ensemble des concepts cohérents, et L'ensemble G constitue donc une borne supérieure de l'ensemble des concepts cohérents. Autrement dit, tout concept plus général qu'un élément de S et plus spécifique qu'un élément de G est cohérent.

On peut définir divers opérateurs de spécialisation et de généralisation qui peuvent être spécifiques à un type d'attribut.
Voici quelques exemples d'opérateurs de généralisation :

$$\left.\begin{array}{l}(x_i = a) \wedge (x_j = v1) \in C \\ (x_i = a) \wedge (x_j = v2) \in C\end{array}\right\} \Rightarrow (x_i = a) \wedge x_j \in [v1 ; v2] \in C$$

$$\left.\begin{array}{l}x_i = a \wedge (x_j = n1) \in C \\ x_i = a \wedge (x_j = n2) \in C\end{array}\right\} \Rightarrow (x_i = a) \wedge (x_j = N) \in C \text{ (N est le père de n1 et n2)}$$

$x_i = a \in C \Rightarrow (x_i = a) \vee (x_j = b) \in C$

$(x_i = a) \wedge (x_j = b) \in C \Rightarrow x_i = a \in C$

$(x_i = a) \wedge (x_j = b) \in C \Rightarrow (x_i = a) \vee (x_j = b) \in C$

$(x_i = a1) \in C \wedge (x_i = a2) \in C \Rightarrow x_i = (a1 \vee a2) \in C,$
et à la limite, lorsque $x_i = \vee_{j \in J} (aj)$ avec Domaine $(x_i) = \vee_{j \in J} (aj)$ alors x_i n'intervient plus.

Il est possible de renverser ces opérateurs pour obtenir des opérateurs de spécialisation.

L'avantage de cette approche est d'une part d'exhiber des concepts dans le langage de la logique propositionnelle, et d'autre part de rester dans le domaine symbolique lors de la phase d'apprentissage, d'où une capacité explicative supérieure par rapport aux méthodes présentées ci-dessous (qui fonctionnent dans le domaine numérique), car l'utilisateur a un contrôle sur cet apprentissage.

Les arbres de décision [Cornuéjols et Miclet 02] :

Il s'agit d'induction dans un formalisme de type logique des propositions. Les attributs peuvent être binaires, numériques ou nominaux non hiérarchiques.

Quand l'arbre est partiellement construit, à chaque nœud correspond un sous-ensemble des exemples d'apprentissage : ceux qui satisfont tous les tests menant à ce nœud. Si ce sous-ensemble n'est pas constitué de points appartenant tous à la même classe, la construction doit se poursuivre. Il faut alors choisir le meilleur attribut (le plus discriminant) à tester.
Plaçons-nous au cours de cette construction à un nœud auquel sont attachés n points de l'échantillon d'apprentissage (c'est-à-dire tous les exemples sauf ceux qui satisfont la séquence de tests ayant mené à ce nœud), répartis en C classes c_j comportant chacune n_j points.

$$\Sigma_{j=1 \text{à} C} n_j = n$$

Considérons pour simplifier un attribut binaire a. Il partage chaque sous-ensemble n_j en deux parties comportant respectivement l_j points (test a = vrai) et r_j points (a = faux).

$$l = \Sigma_{j=1 à C}\, l_j \quad \text{et} \quad r = \Sigma_{j=1 à C}\, r_j \quad et \quad r + l = n$$

On cherche parmi les d attributs restant à tester celui qui possède la plus grande corrélation avec la répartition en classes, c'est-à-dire qui maximise l'information mutuelle avec la distribution en classes des n points d'apprentissage : $I(C,a) = H(C) - H(C|a)$, où H est l'entropie, donc qui minimise $H(C,a)$, avec :

$$H(C|a) = l/n.J(a = vrai) + r/n.J(a = faux)$$

$$J(a = vrai) = \Sigma_{j=1 à C}\, l_j/l.log(l_j/l) \quad \text{et} \quad J(a = faux) = \Sigma_{j=1 à C}\, r_j/r.log(r_j/r)$$

Donc on choisit $i^* \in \{1, \ldots, d\}$ tel que : $i^* = argmin\ H(C|a_i)$

La construction de l'arbre jusqu'à son terme naturel, c'est-à-dire dont les feuilles correspondent à des exemples de la même classe peut ne pas être judicieux : risque de surapprentissage (on est trop calé sur les données : voir 6.1.1.) et donc manque de généralité. Il s'agit donc de contrôler la taille de l'arbre. Une des solutions à ce problème est le pré-élagage : on arrête de diviser un nœud s'il existe une classe "suffisamment majoritaire" sous ce nœud, et ce dernier est donc associé à une feuille qui est cette classe majoritaire. Une autre solution, le post-élagage, plus valide théoriquement et plus efficace en pratique, consiste d'abord à construire l'arbre complètement, puis à l'élaguer en remontant des feuilles vers la racine, en utilisant un critère de qualité qui exprime un compromis entre l'erreur commise et une mesure de la complexité de l'arbre.

Un avantage de cette technique est d'offrir une traduction immédiate en logique des propositions.

Il est à noter également que divers travaux tentent d'étendre cette méthode en introduisant la logique floue (extraction de règles floues à partir d'exemples) mais des problèmes restent en suspens (partitionnement du domaine d'un attribut numérique et détermination de prédicats flous à partir de données numériques) [Gacôgne 97].

LAMDA (Learning Algorithm for Multivariate Data Analysis) :

Il s'agit d'une méthode basée sur une approche probabiliste développée au LAAS-CNRS par J.Aguilar-Martin [Piera *et al.* 89] [Waissman 00].

Les attributs sont quantitatifs (le domaine de valeur est un intervalle) ou qualitatifs (le domaine de valeurs est un ensemble discret).

Les classes soient existent déjà (apprentissage supervisé) soient sont créées à partir d'exemples (apprentissage non supervisé). Chaque classe est définie par un vecteur de paramètres $(\rho_1, \ldots, \rho_p)$ où p est la dimension du vecteur d'entrée.
Dans le cas d'un attribut qualitatif i, ρ_i est un vecteur dont les éléments sont les fréquences d'apparition des différentes valeurs du domaine de l'attribut au sein de la classe. Par exemple, dans le cas d'un attribut Forme de domaine {Triangle, Carré, Rectangle}, ρ_i = (Prob(Triangle), Prob(Carré), Prob(Rectangle)).
Dans le cas d'un attribut quantitatif i, ρ_i est la moyenne des valeurs de cet attribut au sein de la classe.

Il est à noter qu'il existe une classe supplémentaire, appelée classe vide ou résiduelle, qui représente le seuil en dessous duquel il y a rejet afin d'éviter des assignations peu significatives. Pour un attribut qualitatif, les éléments de ρ_i sont des fréquences identiques.
Par exemple pour l'attribut Forme, $\rho_i = (0.333 ; 0.333 ; 0.333)$.
Pour un attribut quantitatif, $\rho_i = 0.5$ (moyenne normalisée).

Pour la reconnaissance de la classe d'une instance $x = (x_1, ..., x_p)$, on calcule les degrés d'adéquation marginale de chaque valeur d'attribut avec chaque classe (y compris la classe vide).
Pour un attribut qualitatif, on a : $\mu (x_i / c_k) = Prob (x_i \text{ dans } \rho_{k,i})$ où $\rho_{k,i}$ désigne le paramètre associé à l'attribut x_i dans la classe c_k.
Pour un attribut quantitatif, on a : $\mu (x_i / c_k) = \rho_{k,i}^{xi}.(1- \rho_{k,i})^{(1-xi)}$ mais la valeur de x_i doit être préalablement normalisée : $x_i = (x_i - x_{min}) / (x_{max} - x_{min})$.

On calcule alors le degré d'adéquation globale $\mu (x / c_k) = f (\mu (x_i / c_k))$.
La fonction f peut représenter la notion linguistique "et", ou la notion "ou", ou encore une notion intermédiaire.
Dans un contexte probabiliste, "et" $\Rightarrow f$ = produit ; "ou" $\Rightarrow f$ = somme probabiliste.
Dans un contexte flou, "et" $\Rightarrow f$ = minimum ; "ou" $\Rightarrow f$ = maximum.
On peut aussi, dans chacun de ces contextes, utiliser une fonction f qui est une pondération des fonctions associées au "et" et au "ou".

Consécutivement à cette phase de reconnaissance, on peut mettre en œuvre un processus d'apprentissage : si l'instance x appartient à la classe vide, alors on crée une nouvelle classe en modifiant les paramètres de la classe vide, et si x appartient à une classe c_k (non vide), alors on modifie les paramètres de cette classe c_k. Les modifications de paramètres se font de la façon suivante :
Pour un attribut qualitatif i, le vecteur de paramètres de la classe c_k étant $\rho_i = (\rho_{i,j})$, nous avons une augmentation de la fréquence correspondant à la valeur x_i (de l'instance x qui vient d'être traitée) : $\rho_{i,j} = \rho_{i,j} + (1 - \rho_{i,j})/(N + 1)$ où N est le nombre d'exemples ayant servis à calculer ces fréquences pour la classe c_k, mais dans le cas de la création d'une nouvelle classe, N peut être choisi arbitrairement (on impose juste $N > 0$). Nous avons aussi une diminution des fréquence correspondant aux valeurs différentes de x_i : $\rho_{i,j} = \rho_{i,j} + (0 - \rho_{i,j})/(N + 1)$.
Pour un attribut quantitatif i, la valeur du paramètre ρ_i de la classe c_k est actualisée selon une formule similaire : $\rho_i = \rho_i + (x_i - \rho_i) / (N + 1)$.

Annexe 7. Classification symbolique de données numériques par l'algorithmes des *k-means*

Prenons l'exemple de la classification des diamètres numériques en valeurs symboliques ou qualitatives.

Φ_{ij} représente le diamètre numérique de la conduite entre les réservoirs T_i et T_j (si elle existe), dont on pourrait imaginer une valeur prise dans le domaine {Petit ; Moyen ; Grand}.

Cette classification par la méthode des k-moyennes (*k-means*) fonctionne de la manière suivante [Cornuejols et Miclet 02] :

• On commence par choisir le nombre de classes, trois dans notre cas (P = Petit ; M = moyen ; G = Grand).

• Puis on calcule un représentant pour chaque classe, par exemple P = min (Φ_{ij}) ; G = max (Φ_{ij}) ; M = Φ_{iojo} tel que $| \Phi_{iojo} - (P + G)/2 | = min_{i,j} | \Phi_{ij} - (P + G)/2 |$.

• On associe ensuite chaque élément Φ_{ij} à la classe dont il est le plus proche (distance euclidienne).

• On calcule ensuite le centre de gravité de chaque classe (qui devient le nouveau représentant de la classe), et on réitère le processus jusqu'à ce que la partition ne change plus.

• On peut calculer la qualité globale de la classification par le biais de la somme des variances intra-classes : $T = 1/m. \Sigma_{j=1àC} \Sigma_{i=1àm} \delta_{ij}.|x_i - \mu_j|^2$ où m est le nombre d'exemples, C le nombre de classes, μ_j le centre de gravité de la classe j, δ_{ij} vaut 1 si x_i appartient à la classe j.

L'algorithme des k-moyennes fait diminuer T, mais rien n'assure que le minimum global soit atteint.
Ce critère T permet également de tester et juger la qualité d'un domaine à une, deux ou trois valeurs. Pour cela, il suffit de lancer l'algorithme des k-moyennes avec une, deux ou trois classes, pour ne retenir finalement que le nombre de classes correspondant à une variance minimale. Mais dans le cas où l'algorithme ne retiendrait qu'une seule classe, il faut également choisir entre les trois valeurs possibles : {P}, {M} ou {G}, et dans le cas où il ne retiendrait que deux classes, il faut choisir entre les trois valeurs possibles : {P, M}, {P, G} ou {M, G}.
Cela nécessite d'utiliser des connaissances supplémentaires, par exemple en fixant des intervalles pour chaque classe de diamètre : P ≈] 0 ; 20 cm [, M ≈ [20 cm ; 30 cm [et G ≈ [30 cm ; +∞ [. Il est clair qu'un tel choix est lié à une classe d'application donnée. Il ne faut pas voir ces intervalles comme étant stricts mais plutôt comme étant flous : si l'algorithme des k-moyennes trouve deux classes C1 = {10 ; 12 ; 18 ; 22 } et C2 = {28 ; 32 ; 35 ; 40} alors on aurait C1 = P et C2 = G bien que les points 22 et 28 appartiennent à l'intervalle relatif à M. Il est clair qu'il y aura toujours des cas litigieux (par exemple C1 = {18 ; 19 ; 21 ; 22 } ∈ P ou M ?).

Pour finir, notons que J.C. Bezdek (dans [Gacôgne 97]) étend la méthode à des classes floues ("*fuzzy k-means*").

Annexe 8. Condition hydraulique d'anticipation de débordement dans le cadre de l'index n° 8 de l'opérateur artificiel basé sur des cas

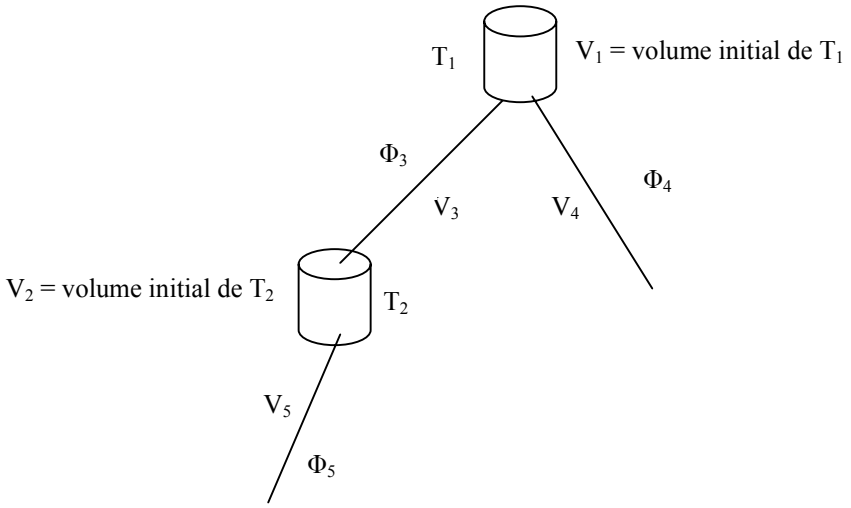


Figure 60 : Anticipation du débordement de T_2

Un certain volume V_1 d'eau dans le réservoir T_1 se vidange dans le réservoir T_2 selon une durée *dt.*

Nous posons :
V_3 = volume écoulé par la conduite de diamètre Φ_3 pendant *dt*
V_4 = volume écoulé par la conduite de diamètre Φ_4 pendant *dt*
V_5 = volume écoulé par la conduite de diamètre Φ_5 pendant *dt*

Nous avons :
$V_1 = V_3 + V_4$
Le débit circulant dans une conduite est approximativement proportionnel au carré du diamètre (formule hydraulique des pertes de charge linéaires dans une conduite : voir détail en annexe 9) :
$Q_3 = \beta.(\Phi_3)^2 = V_3 / dt$
On peut supposer également que le coefficient de débit β est constant (les rugosités des conduites sont les mêmes) :
$Q_4 = \beta.(\Phi_4)^2 = V_4 / dt$

Ainsi, $V_3 = V_1.\Phi_3^2 / (\Phi_3^2 + \Phi_4^2)$

$Q_5 = \beta.(\Phi_5)^2 = V_5 / dt$

Donc $V_5 = (\Phi_5)^2.\beta.dt = (\Phi_5)^2.V_1 / (\Phi_3^2 + \Phi_4^2)$

Le test d'anticipation de débordement est : $V_2 + V_3 - V_5 > V_2^{max}$

Soit : $V_2 + V_1.(\Phi_3^2 - \Phi_5^2) / (\Phi_3^2 + \Phi_4^2) > V_2^{max}$

C'est-à-dire : $\mathbf{V_2 + \alpha . V_1 > V_2^{max}}$ **avec** $\mathbf{\alpha = (\Phi_3^2 - \Phi_5^2) / (\Phi_3^2 + \Phi_4^2)}$

Annexe 9. Calculs des volumes optimaux lors de la stratégie de minimisation de la durée totale de la simulation dans le cadre de l'index n° 8 de l'opérateur artificiel basé sur des cas

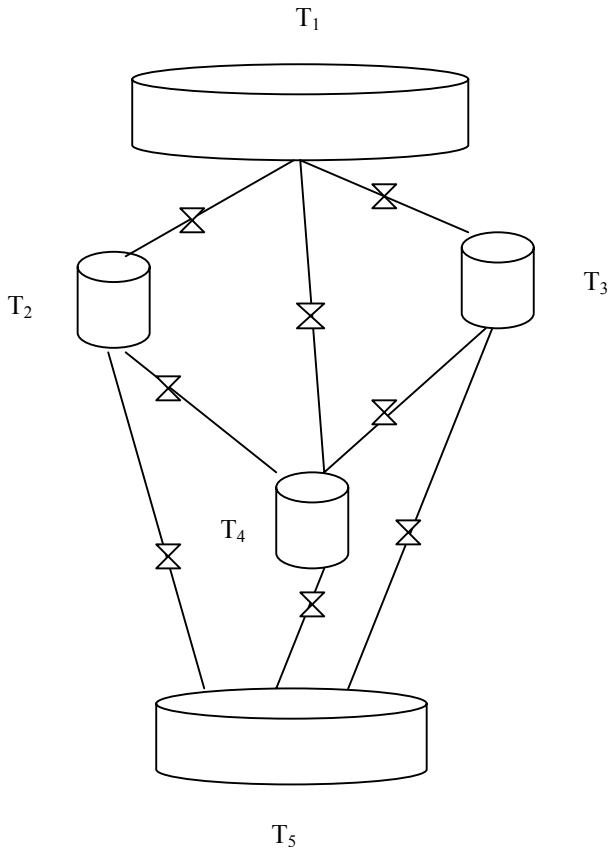


Figure 61 : Minimisation de la durée de vidange de T_1 dans T_5

Prenons un exemple simple pour résoudre ce problème. La généralisation se fait très facilement. D'après le schéma ci dessus, T_2, T_3 et T_4 sont reliés directement à T_5. La stratégie de minimisation de la durée totale s'applique donc à ces trois réservoirs intermédiaires.

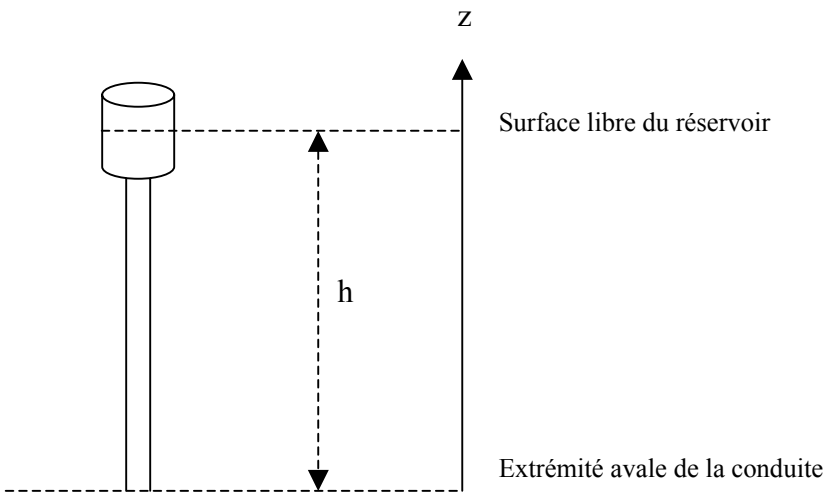


Figure 62 : Réservoir à surface libre relié en aval à une conduite

D'après la figure 62, l'application du théorème de Bernoulli en régime stationnaire entre la surface libre du réservoir et l'extrémité avale (également à l'air libre) de la conduite fournit la relation :
$Po + \rho.g.h = Po + ½.\rho.v^2$
où Po est la pression, atmosphérique et v la vitesse à la sortie de la conduite
Donc le débit q à la sortie de la conduite est :

$q = \sqrt{2.g.h}.\frac{\pi.\phi^2}{4}$ (en fait il faudrait introduire un facteur multiplicatif α représentant le coefficient de rugosité, mais il ne joue aucun rôle dans notre calcul)

En supposant que la hauteur d'eau h soit sensiblement la même pour les trois réservoirs T_2, T_3 et T_4, et que les rugosités des conduites sont identiques, alors on a :

$q = \beta.(\Phi)^2$ avec β = constante

En supposant que ces trois réservoirs se vidangent selon la même durée dt, on a alors une relation entre les trois volumes, sachant que $q = V / dt$:

$V_2 / (\Phi_{T2\text{-}T5})^2 = V_3 / (\Phi_{T3\text{-}T5})^2 = V_4 / (\Phi_{T4\text{-}T5})^2$ où $\Phi_{Ti\text{-}Tj}$ représente le diamètre de la conduite entre T_i et T_j.

Le calcul des volumes optimaux permettant une vidange des trois réservoirs selon la même durée est la solution du problème suivant :

Maximiser : *Volume optimal (T_2) et Volume optimal (T_3) et Volume optimal (T_4)*

Sous contraintes :

Volume Optimal (T_2) / $(\Phi_{T2\text{-}T5})^2$ = Volume Optimal (T_3) / $(\Phi_{T3\text{-}T5})^2$
Volume Optimal (T_3) / $(\Phi_{T3\text{-}T5})^2$ = Volume Optimal (T_4) / $(\Phi_{T4\text{-}T5})^2$

Volume optimal (T_2) $\leq V_2^{max}$ (à V_ε près)
Volume optimal (T_3) $\leq V_3^{max}$ (à V_ε près)
Volume optimal (T_4) $\leq V_4^{max}$ (à V_ε près)

Remarque 1 : l'utilisation d'une procédure du type simplex est inutile car il est très facile de voir qu'une des trois variable atteint sa borne supérieure. En effet, ceci est dû aux contraintes linéaires d'égalité. Il suffit donc de tester pour chacune des trois variables la valeur obtenue par la borne (les valeurs des deux autres variables s'en déduisant immédiatement), c'est-à-dire le respect des deux autres contraintes sur les bornes.

Remarque 2 : il est préférable d'utiliser une marge de sécurité V_ε car une des valeurs *Volume Optimal (T_i)* atteint sa borne qui est la capacité du réservoir correspondant T_i.