Université de Montréal

Optimisation des interactions au sein d'un réseau de connaissances

par

Simon Bélanger

Département d'Informatique et de Recherche Opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la Faculté des Études Supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc)

en informatique

Décembre, 2000

Université de Montréal

Faculté des Études Supérieures

Ce mémoire intitulé :

Optimisation des interactions au sein d'un réseau de connaissances

Présenté par :

Simon Bélanger

A été évalué par un jury composé des personnes suivantes :

Président-rapporteur :	Jean Vaucher
Directrice de recherche :	Esma Aïmeur
Co-directeur de recherche :	Claude Frasson
Membre du Jury :	Marc Kaltenbach

Mémoire accepté le : 27 février 2001

Sommaire

Nous présenterons dans ce mémoire le système *Alice* qui a pour but de modéliser les différentes connaissances et entités que l'on retrouve à l'intérieur d'une organisation. Par l'intermédiaire de cette modélisation, nous tenterons de visualiser les entités similaires et complémentaires. Ces deux dernières relations (similarité, complémentarité) pourront alors être utilisées afin de suggérer des ressources à un individu en difficulté.

La représentation des connaissances repose sur un formalisme que l'on retrouve dans le domaine des ontologies et des graphes sémantiques. Cette modélisation doit être en mesure de refléter les connaissances de chaque entité. Un éditeur graphique est d'ailleurs proposé afin de construire cette représentation des connaissances. L'analyse de ces connaissances est basée sur la catégorisation des différents profils d'entités sans limitation dans le nombre d'attributs et en tenant compte de leur évolution. Nous utiliserons une approche basée sur la reconnaissance de patrons, plus spécifiquement un réseau de neurones nommé cartes auto-organisées de Kohonen. De plus, en nous basant sur ce réseau de neurones, nous tenterons de représenter de façon graphique la distribution des connaissances dans une organisation.

Enfin, dans le but d'apporter de l'aide à un usager face à une problématique, nous proposerons une méthode intuitive pour la construction de requêtes, ces requêtes devant permettre l'accès aux connaissances.

Mots-clés : représentation des connaissances, réseau de neurones, apprentissage non-supervisé, catégorisation, visualisation des connaissances.

Table des matières

Table des figures

Liste des tableaux

Remerciement

Je tiens à remercier ma directrice, le professeur Esma Aïmeur, ainsi que mon co-directeur, le professeur Claude Frasson, qui m'ont guidé à travers les deux années de ma maîtrise. Leur collaboration lors de la rédaction d'articles et leurs conseils judicieux ont été fort appréciés.

Je tiens aussi à remercier mon ami Marc-André pour son apport scientifique, ma chère traductrice et amie Amal et ma graphiste adorée Marième.

Cette maîtrise a bénéficié d'un support financier de la compagnie Virtuel-Âge International. Je leur en suis fort reconnaissant.

Introduction

Les connaissances de l'humain sont acquises tout au long de sa vie, et se composent aussi bien de ses compétences, de ses capacités, de ses intérêts que de ses expériences personnelles. De ce fait, il est de mise d'affirmer que le savoir premier d'une organisation est au coeur des individus qui la constituent [1, 2, 15, 30, 31, 35, 36]. Ce point de vue est d'ailleurs d'autant plus important que nous vivons dans un monde où les professionnels ont un niveau d'études de plus en plus élevé et qu'ils coûtent, de ce fait, de plus en plus cher aux employeurs. Par ailleurs, la forte mobilité de ce qu'on appelle communément "cerveaux" ne cesse de croître, ce qui provoque une vive compétition, et engendre une file interminable d'embauches et de départs.

Dans ce contexte, plusieurs questions se posent : comment gérer ces connaissances qui deviennent quasi volatiles avec cette rotation d'individus (départs dus à la mise à la retraite, à diverses incitations à quitter l'organisation ou tout simplement au roulement de personnel [15]) ? Comment faire bénéficier un nouveau venu de l'expérience acquise d'un collègue afin qu'il puisse surmonter et résoudre un problème ? Comment rapidement créer, dissoudre ou modifier une équipe de projet en constante évolution ? Comment être convaincu de ne pas réinventer la roue inutilement face à une nouvelle difficulté ? Enfin, comment créer et alimenter une interaction constructive entre les diverses sources de connaissances vivantes que sont les membres d'une même organisation ?

La gestion des connaissances d'une organisation composée de plusieurs entités n'est pas une tâche triviale. Cette gestion peut devenir un facteur primordial pour le développement et pour la progression de cette dernière [35, 36]. D'ailleurs, Charnel Havens, Chief Knowledge Officer de la compagnie *EDS*(Electronic Data Systems) cite les enjeux de la gestion de la connaissance d'une entreprise [15, 17] :

> *With a huge portion of a company's worth residing in the knowledge of its employees, the time has come to get the most out of that valuable corporate resource - by applying management techniques.*

Ainsi, le contrôle de l'information dissimulée dans la structure interne et informelle d'une organisation peut devenir un atout important dans l'évolution de cette dernière.

Nous présenterons dans ce mémoire le système *Alice* qui a pour but de modéliser les différentes connaissances et entités que l'on retrouve à l'intérieur d'une organisation. Nous adopterons dans ce cadre les entités suivantes : les individus, les publications, les projets et les départements. Les raisons de ce choix seront expliquées plus loin. Par l'intermédiaire de cette modélisation, nous tenterons de visualiser les entités similaires et complémentaires. Ces deux dernières relations (similarité, complémentarité) pourront être alors utilisées afin de suggérer des ressources à un individu en difficulté.

Le savoir d'une organisation repose sur la connaissance des individus qui la composent. La fonction même de l'organisation et la diversité de ses ressources humaines engendrent une très grande quantité d'information à traiter. Dans ce contexte, plusieurs difficultés se posent :

- À ce jour, très peu de systèmes génériques [1, 2] existent pour la gestion de cette masse d'information dans une organisation. Par ailleurs, le procédé d'accès à la connaissance dans ces systèmes reste compliqué.
- Dans les systèmes existants, les requêtes doivent être suffisamment précises pour arriver à un résultat, ce qui n'est pas toujours évident. En effet, l'usager ne sait pas forcément où chercher la connaissance ni comment la rechercher.

- Il est difficile de catégoriser de façon adéquate le profil d'un entité, basé sur une représentation des connaissances. Cette difficulté est liée à la grande quantité d'information à inclure ainsi qu'à l'évolution des profils (ces derniers ne sont pas statiques).
- La création de représentations efficaces des connaissances dans une organisation représente une tâche ardue. En effet, il est compliqué de distinguer les informations essentielles ainsi que les forces et faiblesses des différentes entités de l'organisation.
- La distribution des connaissances d'une organisation n'apparaît pas clairement en raison du grand nombre d'entités qui y existent. Leur gestion en est donc complexifiée.

Pour faire face à ces difficultés nous envisageons de :

- Construire un système qui effectue, par des méthodes pouvant s'appliquer à n'importe quelle source d'information (donc génériques), une gestion des connaissances au sein d'une organisation.
- Fournir à l'utilisateur une méthode intuitive pour la construction de requêtes, ces requêtes devant permettre l'accès aux connaissances.
- Lorsqu'une requête n'aboutit pas à un résultat précis, notre objectif est de quand même fournir un espace de solutions à l'usager, ces solutions contenant des connaissances pouvant potentiellement l'aider.
- Catégoriser différents profils d'entités sans limitation dans le nombre d'attributs et en tenant compte de leur évolution. Nous utiliserons une approche basée sur la reconnaissance de patrons, plus spécifiquement les réseaux de neurones.
- Construire une représentation des connaissances selon un formalisme que l'on retrouve dans le domaine des ontologies et des graphes sémantiques. Cette modélisation doit être en mesure de refléter les connaissances de chaque entité.

- Construire une représentation graphique de la distribution des connaissances dans une organisation. L'approche utilisée est basée sur le fonctionnement des réseaux de neurones.

Ce mémoire s'articule autour de quatre chapitres principaux. En premier lieu nous présenterons quelques systèmes reliés de près ou de loin à *Alice*, notre système. En second lieu, nous aborderons au chapitre 2 la représentation et l'analyse des connaissances qui seront utilisées au sein d'*Alice*. L'analyse de la connaissance focalisera sur un algorithme de reconnaissance de patrons utilisé pour catégoriser les différents profils de connaissances. Une fois ces deux approches présentées, nous aborderons au chapitre 3 le système *Alice* lui-même. Chaque module de ce système sera décomposé et détaillé à la fois sous sa forme théorique et pratique. Le chapitre 4 portera sur l'évaluation du système *Alice*. Une conclusion viendra clore la présente recherche en illustrant les forces et les faiblesses de notre système et en indiquant les directions futures.

Ces travaux ont été financés par la compagnie Virtuel-Âge International. Ils ont fait l'objet de plusieurs publications [4, 5, 44].

Chapitre 1

Revue de littérature

1.1 Introduction

Nous présenterons dans cette section quelques systèmes similaires par certains aspects au système *Alice*. Cette similarité peut se retrouver aussi bien en terme d'idéologie qu'en terme d'objectifs. Nous porterons une attention particulière au système *CKBS* puisqu'il est de loin le système qui se rapproche le plus, autant en théorie qu'en pratique, du système *Alice*. Les autres systèmes seront présentés globalement, en focalisant toutefois sur certains modules précis.

1.2 Projet CKBS

Le projet *CKBS* [2, 30, 31] (*C*ompetence *K*nowledge *B*ase *S*ystem) a été développé dans les laboratoires du *German Research Center for Artificial Intelligence(DFKI)*. Son but est d'organiser et de profiter au maximum des connaissances d'une entreprise. Il est l'un des systèmes existants se rapprochant le plus du système développé dans le cadre de cette maîtrise. L'idéologie globale de ce système est dans la même lignée qu'*Alice*. Toutefois, l'approche utilisée afin d'arriver à des résultats satisfaisants est sur plusieurs points différente. Il serait donc intéressant de survoler ce projet en détail.

La philosophie derrière *CKBS* réside dans le fait que les compétences et les expériences personnelles des employés sont à la base des connaissances d'une entreprise [31]. L'employé est vu comme un objet, une entité, de la même façon qu'une autre composante de l'entreprise. C'est par l'entremise de ces entités que l'information est extraite. Le terme utilisé par Abecker et son équipe [2, 30, 31] qui définit ces composantes est *OMIS, Organizational Memory Information System.* Ce système tente de faciliter la recherche d'une personne possédant des connaissances, expériences ou habiletés particulières. Ces recherches sont exécutées sous forme de requêtes bien précises.

Les différentes sources d'information que l'on retrouve dans une entreprise sont représentées par un modèle d'information formel. Ce modèle d'information est basé sur un vocabulaire défini par trois différentes ontologies, soit une ontologie sur l'entreprise, une ontologie sur le domaine d'application et une ontologie dite d'information. L'architecture globale de cette représentation est illustrée à la figure 1.1. L'ontologie d'information décrit le vocabulaire associé aux différentes sources d'information avec leur structure respective. Les concepts de cette ontologie sont indépendants du domaine, on peut prendre comme exemple la structure suivante : un livre possède un titre, est écrit par un auteur, etc. L'ontologie de l'entreprise est utilisée pour décrire le contexte où l'on retrouve cette information. Ainsi, on y retrouve la structure de l'entreprise sous une forme hiérarchique. Enfin, les connaissances liées au domaine d'application et contenues dans les différentes sources d'information sont représentées par l'ontologie du domaine.

La recherche d'information à travers cette masse de connaissances est basée sur une logique de premier ordre. Chaque requête est composée et exécutée sous la forme d'une suite d'inférences logiques. Cette approche permet d'avoir une base fondée sur des mécanismes d'extraction théorique qui sont formulés et supportés par les théories du domaine de la logique [46].

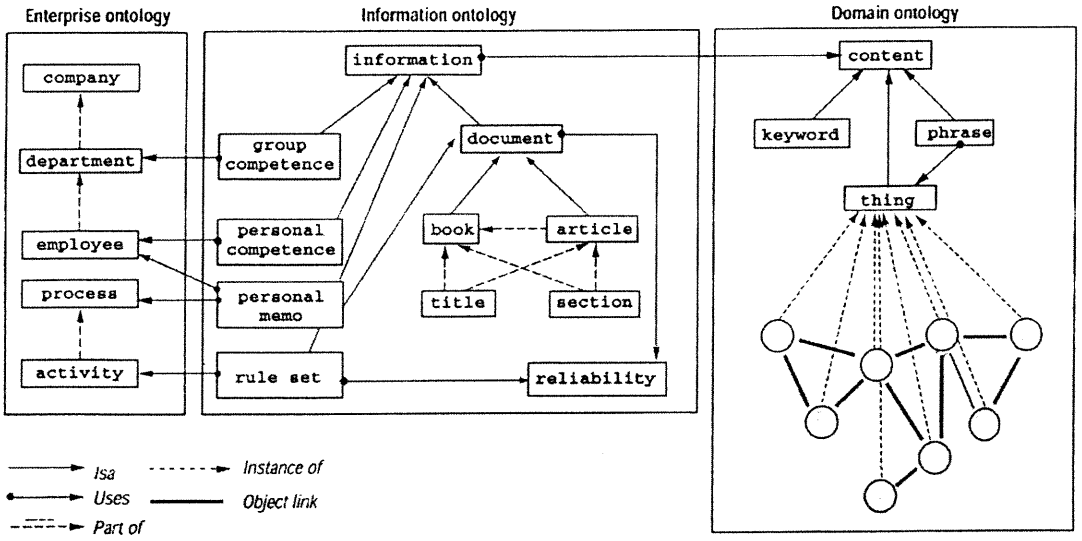

FIG. 1.1 – Architecture de la représentation des connaissances du système *CKBS* [31]

Si nous prenons la figure 1.2, une manière intuitive pour la recherche d'un employé ayant des compétences particulières est de partir de ces compétences et de traverser le graphe dans le but d'arriver à un noeud qui représente un employé. Cette recherche doit se faire en un nombre d'étapes raisonnable. Ce type de requête peut devenir très spécifique au domaine d'application [31]. C'est l'une des raisons pour lesquelles les auteurs du projet *CKBS* ont déterminé un formalisme précis pour la formation de requêtes. Ces requêtes, appelées heuristique de recherche, sont composées d'une séquence de formules de la forme :

$$f_1 \circ f_2 \circ \ldots \circ f_n \tag{1.1}$$

avec

$$f_i \equiv (\lambda)^\gamma \tag{1.2}$$

où λ est un lien ou un lien inverse (λ^{-1}) et où γ indique la fermeture de la relation. Cette fermeture a pour but de déterminer des bornes sur la longueur du chemin suivi

lors de la recherche. Le paramètre γ peut prendre les valeurs suivantes : n (un nombre entier), $\geq n$, $*$ comme abréviation de ≥ 0 ou $+$ comme abréviation de ≥ 1.

Chaque formule utilise le vocabulaire défini dans le cadre des différentes ontologies. Une telle formule prend en considération un ensemble de noeuds comme entrée. Pour chacun de ces noeuds, elle suit les liens spécifiés par la formule f dans un ordre de droite à gauche. Chaque étape retourne un nombre de noeuds qui seront à leur tour les noeuds appartenant à l'ensemble d'entrée pour la prochaine formule. Évidemment, cette recherche n'a du sens que si l'ensemble des résultats finaux ne contient que des employés.

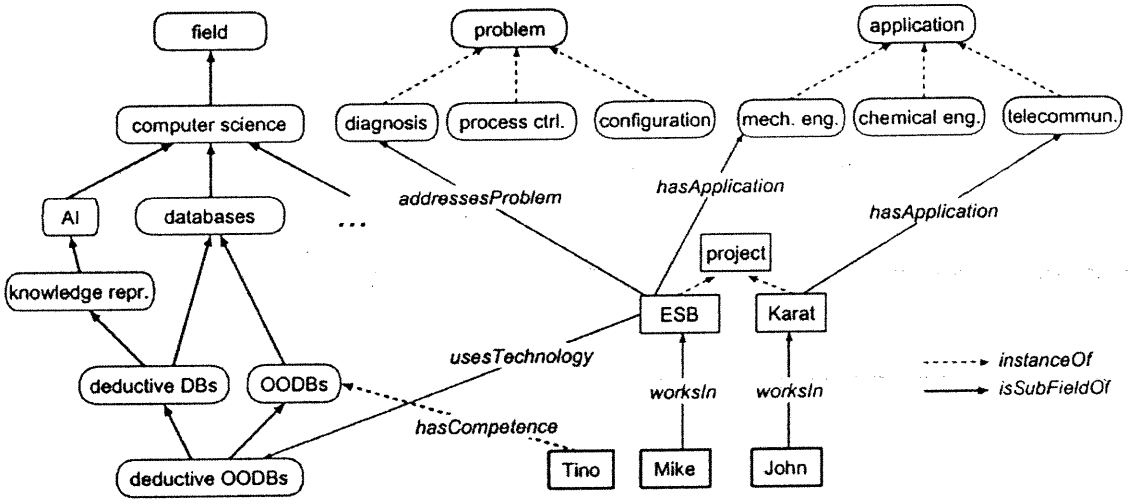

FIG. 1.2 – Exemple d'une ontologie du système *CKBS* [31]

Nous présenterons deux exemples de recherche heuristique basés sur une ontologie, ces exemples sont tirés de [31]. Dans le cas où nous cherchons un employé ayant des compétences dans le domaine des bases de données, une simple recherche à partir de mots clés résulterait en un échec, puisqu'aucun n'employé est associé directement au concept bases de données. Cependant, en tenant compte du fait qu'une personne ayant des connaissances dans un domaine plus spécifique que le domaine initial peut

également avoir les compétences requises, la recherche heuristique est en mesure de suggérer l'employé *Tino*. La spécification de cette recherche est la suivante :

1. $(hasCompetence^{-1})^1$

   "First search for people directely linked to a search concept."

2. $(hasCompetence^{-1})^1 \circ (isSubFieldOf^{-1})^+$

   "Then look for people competent in some subfield."

Le second exemple illustre le principe qu'une personne travaillant sur un projet utilisant une certaine technologie, peut posséder des compétences reliées à cette technologie. C'est ainsi qu'à partir des technologies utilisées par un projet, nous pouvons inférer la connaissance de ces technologies pour les personnes associées au projet. Donc, si nous cherchons un expert en base de données déductives orientées-objet (*DOODB*), nous trouvons le projet *ESB*, et via le lien *worksIn*, nous trouvons *Mike* qui est, selon le principe mentionné ci-dessus, compétent dans le domaine des *DOODB*. Avec une autre heuristique, énonçant comme principe qu'un expert travaillant dans un domaine plus spécifique peut également posséder des connaissances dans le domaine recherché, nous retrouvons une seconde fois l'employé *Tino*. Cependant, cette conclusion (expert *Tino*) semble plus risquée et moins précise que la conclusion qui vise l'employé *Mike*. La spécification de cette recherche est la suivante :

1. $(hasCompetence^{-1})^1$

   "First search for people directely linked to a search concept."

2. $(worksIn^{-1})^1 \circ (usesTechnology^{-1})^1$

   "Then look for people working in a project applying the technology in quest."

3. $(hasCompetence^{-1})^1 \circ (isSubFieldOf)^1$

   "Finally look for people experienced in the direct superconcept of the topic in quest."

En résumé, la description formelle des connaissances sous le modèle d'une ontologie permet de modéliser plusieurs types de relations existantes au niveau de l'entreprise.

De plus, la spécification des requêtes se basant sur une logique formelle apporte une approche intuitive et générale pouvant être appliquée à d'autres domaines d'application.

1.3 Projet Watson

Le système *Watson* [8, 28] a comme caractéristique d'être un système d'information de type *just-in-time*. Cette appellation indique qu'il tente de fournir à l'usager de l'information en temps réel selon le contexte, et ceci sans requête de ce dernier. Par exemple, si l'usager rédige un document, le système tentera d'extraire de l'information de ce document pour ensuite proposer par l'intermédiaire du *Web* des liens qui pourraient lui être utiles dans sa rédaction.

Ce système, de la classe des *Information Management Assistants (IMA)*, repose sur une architecture à quatre niveaux. Le premier niveau, *Anticipator*, utilise un modèle d'anticipation afin d'interpréter à partir du contexte les informations susceptibles d'intéresser l'usager. Le second niveau *Content Analyser*, qui est celui qui nous intéresse le plus, emploie le contenu du document d'une quelconque application sur laquelle l'usager travaille et tente de le représenter sous un modèle précis. Ce modèle est par la suite transmis à la couche *Resource Selector* qui elle s'occupe d'envoyer le modèle à des ressources externes (engins de recherche). Le résultat, habituellement sous forme *HTML*, est filtré par la dernière couche *Result Processor* utilisant des métriques de similarité entre le profil d'intérêt et les résultats retournés. L'usager a également la possibilité de faire des requêtes sur demande.

Comme mentionné ci-dessus, la couche qui nous intéresse le plus est celle qui extrait l'information des documents associés au contexte (*Content Analyser*) pour ensuite les transmettre à la couche *Resource Selector*. La construction du modèle est basée sur des méthodes associées au domaine de l'extraction d'information. Plusieurs étapes

sont suivies, premièrement, il y a élimination des mots peu importants formant un ensemble communément appelé *stop list* (the,a,is,that,etc). Ensuite, les mots ayant une fréquence élevée sont considérés comme représentatifs du document et donc incorporés au modèle. Les mots sous une forme particulière (caractère gras, italique) sont également considérés comme des termes importants. En dernier lieu, la position des mots dans le texte est importante, les mots en haut de page ou début de section sont considérés comme étant plus importants, ainsi que les mots présents dans des structures telles que des listes ou des énumérations de concepts clés. Ensuite, chaque mot se voit associé à un poids selon les critères mentionnés ci-dessus. Le modèle final est composé des 20 mots ayant les poids les plus élevés. Ce modèle est à la base des requêtes acheminées vers les autres couches du processus de gestion de la connaissance. Nous ne nous étendons pas sur les autres couches, car elles ne sont pas directement liées à notre projet.

1.4 Projet RefferelWeb

L'intérêt du système *RefferelWeb* [23, 24, 25] est qu'il aide à la recherche d'informations pouvant mener vers une ressource humaine appropriée. Il utilise une approche complètement différente de celle d'*Alice* et se base uniquement sur la proposition d'experts associés au domaine d'application. Seulement, cette approche tente de stimuler une interaction au sein d'un domaine d'application.

Ce projet a été développé dans les laboratoires de *AT&T*. Il repose sur la création d'un réseau social lié à une communauté d'experts affiliés à un domaine d'application. Le réseau est à la base des recherches menant vers des personnes (experts) pouvant satisfaire à une requête précise. La notion de réseau social repose sur le travail de S. Milgram qui stipule que deux personnes aux États-Unis sont séparées au maximum par un lien de six personnes, ces dernières formant une chaîne de références [33]. Ainsi, la distance entre deux individus est relativement petite en terme de personnes.

De cette manière, une personne peut demander la relation (chaîne de références) qui existe entre elle et un expert, sachant que ce dernier pourra répondre à ses interrogations. Un rayon minimum définissant le nombre de liens maximum entre elle et l'expert peut être spécifié. Une autre possibilité réside dans la recherche d'un expert en spécifiant un sujet particulier afin éventuellement de trouver de l'aide.

Cette chaîne de références permet de rendre les résultats obtenus à la suite d'une requête plus fiables pour les personnes ayant demandé de l'aide justement grâce aux experts qui la constituent. La plupart des systèmes de recommandation sont basés sur des opinions anonymes. L'auteur [23, 24, 25] affirme que lorsqu'une personne doit prendre une décision importante, elle peut difficilement faire confiance à des opinions ou informations venant d'une source anonyme. L'idéal est donc d'être aidé par un expert qu'elle connaît. Seulement, si elle n'en connaît pas directement, elle cherchera à se faire référencer par un ou une série de collègues en qui elle peut avoir confiance. Du coté de l'expert, il sera peut-être plus enclin à répondre à une requête si on lui présente la chaîne de référence qui lui est associée, et qui nous a mené à lui.

Le réseau social est exprimé par un graphe où chaque nœud représente un individu, tandis que les liens sont les relations entre individus. Le réseau est construit sans la coopération explicite des individus apparaissant dans le réseau. Pour ce faire, le système explore diverses sources d'information publique [23] :

- La notion d'auteur et de co-auteur dans les différents communiqués scientifiques est considérée comme un lien entre deux individus.
- Au niveau des entreprises, les relations qui sont déterminées selon l'organisation interne, les projets actuels ou passés et les publications internes sont également considérés.
- La construction d'araignées (processus mobiles) qui scrutent le *WWW* à la recherche de relations basées sur l'occurrence de noms à l'intérieur de mêmes pages *Web*.

- L'analyse du courrier électronique serait un indicateur de choix pour la découverte de relations inédites. Seulement, le respect de la confidentialité mène à l'élimination de cette méthode d'analyse.

La création du réseau social est jumelée à la construction d'une base de connaissances pour chacun des individus présents dans le réseau. *RefferelWeb* tente d'automatiser le processus d'extraction des connaissances du domaine d'expertise pour chacun d'eux. Deux stratégies sont alors mises en oeuvre [23] :

- Générer une base de données d'expertise en même temps que le réseau social est créé. Les titres de publications sont la principale source de connaissances. De plus, les profils sont complétés par la recherche d'informations (plus précisément *data mining*) à partir des documents publics que l'on retrouve sur le *Web*. Une conséquence de cette recherche est l'addition d'individu ne s'étant pas enregistré explicitement au réseau social .
- Générer une requête divisée en différentes sections sous forme de conjonction entre le nom de l'expert et le ou les sujets recherchés. Elle est ensuite soumise à un moteur de recherche du type *AltaVista* qui détermine la fréquence des associations entre le nom et les sujets donnés. Ces statistiques sont d'abord normalisées pour être ensuite utilisées comme profils d'expertise. Cette dernière méthode est plus lente, mais elle assure que les informations rapportées sont à jour.

Il existe également une version de *RefferelWeb* développée par *AT&T* en interne permettant d'augmenter la communication et la diffusion d'information à l'intérieur même de la corporation.

1.5 Projet InfoFinder

Le projet *InfoFinder* est similaire au projet *Alice* en raison de la construction de profils d'intérêts associés aux usagers. Ces profils sont basés sur les documents que l'usager consulte sur le *Web*. Nous allons décrire brièvement ce système.

L'agent *InfoFinder* [7] est destiné à découvrir et à construire le profil d'intérêt d'un usager. Pour ce faire, l'agent analyse des documents choisis par l'usager lorsque ce dernier navigue sur le *Web*. L'analyse est basée principalement sur une suite d'heuristiques qui extraient certaines phrases selon des critères syntaxiques. Ces phrases serviront par la suite à représenter le document dans son ensemble. Ensuite, un algorithme d'apprentissage est utilisé à la construction d'un arbre de recherche. Cet arbre sera transformé en une ou plusieurs requêtes booléennes qui seront transmises par la suite à un moteur de recherche. L'agent sera donc en mesure de soumettre de nouvelles pages correspondant au profil d'intérêt de l'usager.

Lors de la navigation sur le *Web*, l'usager indique à l'agent, par la sélection d'une icône, les documents qui répondent à ses intérêts. L'agent demande alors un nom générique qu'il associera au document. Le nom donné servira simplement à classer les documents et à une communication future entre l'agent et l'usager. Chacun des documents sera analysé afin d'extraire les phrases clés déterminées par un ensemble d'heuristiques. Les heuristiques sont basées sur l'observation de certaines techniques visuelles pour mettre l'emphase sur des éléments importants. Ces techniques utilisent des polices différentes ou des structures particulières pour certains mots par exemple. Le document sera représenté dans le système par les quelques phrases choisies par l'agent.

Lorsqu'une catégorie est composée d'un nombre suffisant de documents il y a construction d'un arbre de décision par une variante de l'algorithme *ID3*. Ce dernier décrit alors les intérêts de l'usager envers cette catégorie prédéterminée par l'usager. Cet

arbre sera transformé en requêtes booléennes qui seront utilisées ensuite par des engins de recherche pour découvrir de nouveaux documents. Ces documents devraient répondre aux intérêts de l'usager. Le processus d'apprentissage devient par la suite interactif, l'usager ayant la possibilité de donner son appréciation sur les nouveaux documents. Ce processus est facilité par deux icônes, l'une qui approuve et la seconde qui désapprouve la qualité d'un document. Les documents refusés par l'usager sont souvent affiliés à un sujet d'intérêt mais ne répondent pas aux intérêts particuliers de l'usager [7]. De cette façon, l'agent peut redéfinir et clarifier les frontières pour chaque catégorie.

1.6 Projet CiteSeer

Le système *CiteSeer* [6] recommande les articles scientifiques en se basant sur le profil d'intérêts d'un usager. Ce profil contient un ensemble de caractéristiques qui peuvent être manuellement modifiés par l'usager ou par le système. Dans ce dernier cas, le système se base sur les commentaires (*feedback*) que l'usager énonce suite à une recommandation. Nous pouvons alors parler d'une adaptation du profil de l'usager. Ce profil d'intérêts est principalement composé de citations et d'articles intéressants aux yeux de l'usager. La recherche d'un nouvel article se fait sur la base de ce profil. Donc si dans le contenu d'un nouvel article, on retrouve des citations apparaissant dans le profil de l'usager ou référant à des articles également dans ce profil, il sera recommandé.

Certaines métriques basées sur les citations présentes dans un profil d'usager sont appliquées aux documents trouvés afin d'associer un poids représentant l'intérêt d'un nouvel article. Dans le cas où une citation contenue dans le profil de l'usager est très peu fréquente, elle sera considérée comme étant plus importante qu'une citation qui revient fréquemment. Cette approche est similaire à la valeur de discrimination [40, 41] associée à un mot présent dans un corpus.

1.7 Comparaison

Les systèmes présentés ont tous un objectif commun, celui de guider un usager à travers une masse d'information qui, à priori, n'est pas facile à gérer. Nous présenterons dans cette section les approches qui nous ont influencé dans l'élaboration du système *Alice*.

1.7.1 CKBS

Le système *CKBS* se résume en deux grandes approches. La première est le formalisme de représentation basé sur une ontologie. Cette ontologie permet, contrairement à une simple base de mots clés, de modéliser les différentes relations qui existent entre différents concepts. Cette approche sera retenue pour le projet *Alice* car nous croyons qu'elle est une structure simple et efficace.

La seconde approche est celle de la logique de premier ordre et des requêtes de recherches qui lui sont associées. Nous ne doutons en aucun cas de la puissance de recherche de cette technique, toutefois elle nous semble difficile à utiliser sans une connaissance exacte du domaine d'application. En effet, chaque requête doit être précise autant dans sa forme que dans son contenu. Il nous semble donc difficile, pour un usager peu habitué à ce type de manipulations, de construire une requête qui répondra à ses attentes. Cette approche sera donc laissée de côté.

1.7.2 Watson

Le projet *Watson* a comme objectif la construction d'un profil de l'usager, composé de mot clés sélectionnés dans un document en relation avec ce dernier. La méthode de sélection est basée sur la fréquence d'occurrence d'un terme et de certaines règles syntaxiques ou graphiques. Cette approche ne prend pas en compte les relations existantes entre ces différents termes. L'approche basée sur les statistiques d'un terme

dans un document sera utilisée dans *Alice*, seulement les relations entre différents termes seront également prises en considération.

1.7.3 RefferelWeb

Le projet *RefferelWeb* apporte l'idée que chaque individu est porteur de connaissances et est donc capable d'aider son semblable lorsque ce dernier doit résoudre une difficulté donnée. Nous retiendrons de ce système l'idée de suggérer à un usager en difficulté, l'aide d'un second usager pouvant peut-être répondre à ses problèmes. Par contre, même si l'idée d'une chaîne de référence nous parait intéressante, nous ne développerons pas cette approche dans le cadre du projet *Alice*.

1.7.4 InfoFinder

La question de construction d'un profil associé à l'usager revient dans le système *InfoFinder*. Dans ce projet, le profil est construit principalement sur la base de caractéristiques syntaxiques et graphiques retrouvées dans les documents auxquels l'usager semble intéressé. Seulement, il est de mise que ces documents aient certaines caractéristiques syntaxiques et graphiques. Nous pensons que ce type d'analyse peut apporter des bienfaits mais seulement pour certains types de documents. C'est pourquoi nous n'utiliserons pas cette approche mais plutôt une technique applicable à tous les documents. Par contre, comme pour le système *Watson*, l'usager a la possibilité d'émettre certains commentaires dans le but de permettre au système d'adapter le profil de l'usager. Cette approche ne sera pas retenue dans cette présente version du système *Alice*, seulement elle pourrait devenir un ajout important dans des versions ultérieures.

1.7.5 CiteSeer

Enfin, le système *CiteSeer* se caractérise par l'utilisation d'une méthode apparentée à la théorie de la valeur de discrimination de Salton [40]. Elle permet de cibler des énoncés, que ce soit des citations ou des concepts, ayant un degré de caractérisation important. Par l'intermédiaire de cette théorie nous établirons une méthode pour suggérer de nouvelles connaissances ayant échappé à une première modélisation.

1.8 Conclusion

Deux difficultés majeures se retrouvent dans les systèmes décrits ci-dessus. Ces dernières sont également présentes au sein du système *Alice*. La première de ces difficultés concerne la représentation des connaissances d'un individu par rapport à un domaine en particulier. Cette représentation joue un rôle important dans la cohérence et la qualité des informations retournées aux usagers. Le projet *CKBS* est celui qui propose la structure la plus formelle. C'est d'ailleurs une source d'inspiration importante pour le système *Alice* en ce qui concerne la représentation des connaissances. Les autres systèmes reposent sur des représentations simplistes basées sur des bases de mots-clés.

La seconde difficulté concerne le processus d'analyse de ces connaissances. Pour éclaircir ce processus, il serait judicieux de récapituler les points suivants :

- À partir des connaissances, nous devons créer un profil des entités présentes dans notre système.
- L'objectif est de trouver une ressource en terme de connaissances pouvant aider une entité ayant un besoin d'information.
- Pour ce faire, trois points doivent être abordés :

- Recherche de la connaissance dans une structure de données. C'est ce que fait le système *CKBS* qui parcourt une arborescence selon une logique de premier ordre.
- Utilisation de métriques de similarité entre deux types d'entités. C'est ce que font globalement les autres systèmes.
- Catégorisation des entités pour faciliter cette recherche de connaissances.

Nous allons utiliser la troisième approche (catégorisation) qui est la plus évoluée puisqu'elle utilise les deux premières et retourne une information plus précise. Pour effectuer cette catégorisation, nous nous baserons sur la reconnaissance de patrons.

Nous aborderons ces deux difficultés (représentation et analyse de la connaissance) dans le chapitre suivant.

Chapitre 2

Traitement des Connaissances

2.1 Introduction

On retrouve des systèmes à base de connaissances dans des domaines aussi variés que l'ingénierie, les sciences, la médecine ou le domaine de l'administration [42]. Dans chacun de ces domaines, ces systèmes modélisent des situations et résolvent des problèmes. Comme leur nom l'indique (systèmes à base de connaissances), ils reposent sur la connaissance en elle même.

Pour qu'une connaissance soit analysée, elle doit tout d'abord avoir une représentation. Nous aborderons donc, en premier lieu, la problématique de la représentation des connaissances. Une structure basée sur les principes d'une ontologie et représentée de façon graphique par un graphe sémantique sera présentée. Cette représentation a été choisie pour sa simplicité et son intérêt actuel dans le domaine des systèmes à base de connaissances [2, 9, 16].

Une fois la modélisation de cette connaissance effectuée, le second défi réside dans son utilisation dans la résolution de problèmes. La méthode utilisée dans notre système repose sur la reconnaissance de patrons au sein de cet ensemble de connaissances. Nous présenterons donc un aperçu des différentes techniques utilisées dans la reconnaissance

de patrons avant de nous intéresser à une technique bien précise (les cartes auto-organisées de Kohonen). Mais tout d'abord, nous aborderons comme mentionné ci-dessus la représentation des connaissances.

2.2 Représentation des connaissances

La connaissance, qui est à la base de notre raisonnement, se retrouve structurée sous un certain format quel que soit son contexte. Le moyen habituel pour exprimer cette connaissance est la langue naturelle. Elle constitue le moyen le plus simple et le plus universel pour décrire des faits, exprimer des observations, transmettre des connaissances, avec le degré de précision ou d'imprécision souhaité.

La puissance de la langue naturelle crée en même temps un obstacle à son utilisation pour le traitement de l'information. De fait, les systèmes informatiques éprouvent des difficultés en présence de la paraphrase ou de la construction de nouveaux concepts, omniprésents dans l'emploi de la langue. Ils ont tendance à buter sur l'ambiguïté de certains énoncés, pourtant clairs dans le contexte dans lequel ils ont été écrits. En résumé, ils manipulent des symboles formels, et ne peuvent, pour cette raison, appréhender directement des textes [50]. Pour traiter l'information avec une machine, il faut lui fournir un modèle formel. Nous présenterons donc quatre différents types de représentation des connaissances [32] pour ensuite nous focaliser sur une représentation en particulier.

2.2.1 Les représentations logiques

Cette classe de représentations utilise des expressions en logique formelle afin de représenter la base de connaissances. Des règles d'inférence appliquent cette connaissance à des instances de problèmes. La logique des prédicats est la représentation logique la plus usitée.

2.2.2 Les représentations procédurales

Ces dernières représentent la connaissance comme un ensemble d'instructions devant servir à résoudre un problème, ce qui contraste avec les représentations déclaratives des réseaux logiques et sémantiques. Un système de production peut être vu comme un exemple de la représentation procédurale.

2.2.3 Les représentations structurées

Ces types de représentations étendent les réseaux en permettant à chaque noeud de représenter une structure de données complexe. Des exemples de représentations structurées incluent les scripts, les *frames*, et les objets.

2.2.4 Les représentations basées sur les réseaux

Ces représentations capturent la connaissance sous forme de graphes dans lesquels les noeuds représentent les objets ou les concepts du domaine, et les arcs les relations ou association entre eux. Dans les représentations en réseaux, on peut citer les réseaux sémantiques, les dépendances conceptuelles, et les graphes conceptuels. Les réseaux sémantiques ont d'ailleurs été utilisés dans le cadre du présent projet. Nos détaillerons les composantes de notre graphe sémantique en terme de structuration d'une base de connaissances dans la prochaine section.

2.2.5 Composantes d'une base de connaissances

Un domaine d'application peut être découpé en deux différents ensembles qui sont complémentaires : l'ensemble des concepts pertinents qui composent le domaine abordé, et celui des relations qui permettent de relier ces concepts sous différentes sémantiques. Un exemple commun est la relation entre des concepts généraux et des concepts plus spécifiques, ou la construction de concepts complexes à partir de concepts plus simples

[50]. Ces concepts et relations se retrouvent habituellement dans la littérature associée au domaine ou même à l'intérieur d'une base de données conventionnelle. L'ingénieur des connaissances entre alors en jeu avec mission d'extraire de ces sources d'information les concepts du domaine ainsi que les relations qui articulent le tout. Il doit construire une terminologie du domaine, c'est-à-dire recenser les concepts et les termes qui représentent le domaine d'application. Parmi les facteurs qui peuvent rendre cette tâche d'extraction non triviale, nous citerons :

- La cohérence entre concepts : cela peut représenter un problème s'il existe un manque de consensus sur la définition d'une notion entre l'ingénieur des connaissances et les usagers qui utilisent le système. Cette cohérence peut également être mise à l'épreuve lorsqu'un concept évolue avec le temps. L'une des façons les plus simples pour remédier à ce problème est de lier chaque concept à une définition précise.
- La notion de polysème, qui se résume à utiliser un mot ayant plusieurs significations. Une ambiguïté peut également survenir lorsque l'association de deux concepts par une relation peut prendre plusieurs significations. Il faut dans ce cas formuler un nouveau concept général représentant le concept formé par la paire de concepts.
- Le problème des concepts similaires. La synonymie est le cas où deux concepts potentiels désignent la même chose. La synonymie ou paraphrase correspond à une situation dans laquelle un concept unique serait désigné par plusieurs termes différents. L'adhésion à une terminologie de référence supprime ce cas de figure, chaque concept se voyant associer un terme unique. Cependant, certaines terminologies incluent des termes supplémentaires pour désigner un même concept [50]. On conserve ainsi une possibilité de synonymie, représentée par des relations de similarités (voir section 3.3.1.1). Le principal problème avec la synonymie est que deux concepts peuvent être à la fois différents et synonymes en fonction du domaine d'application.

L'ingénieur des connaissances est en grande partie responsable de la qualité du savoir associé au système (qui est stocké dans une base de connaissances). Il faut toutefois souligner que cette base est en constante évolution de même que le domaine d'application. Une approche basée sur les travaux de Salton [40, 41] est utilisée au sein du système *Alice* dans le but de proposer de nouveaux concepts non présents dans la base de connaissances. Elle sera détaillée à la section 3.3.3.2 lors de la description du système.

L'ensemble de la base de connaissances telle que définie dans la section précédente est déterminé par la notion d'ontologie. C'est cette notion que nous allons présenter.

2.2.6 Notion d'ontologie

Nous allons maintenant présenter le formalisme de représentation des connaissances choisi dans le cadre du système *Alice*. La notion d'ontologie sera utilisée pour expliquer le contenu, les propriétés essentielles, et les relations entre les termes de la base de connaissances. La plupart des ontologies ont été construites en utilisant un modèle à base de texte (définition précise pour chaque concept et relation) [29, 37]. Ce modèle facilite la construction d'une ontologie mais néglige l'aspect visuel des relations entre les concepts. C'est pourquoi nous utiliserons les réseaux sémantiques afin de visualiser de façon graphique la base de connaissances.

2.2.6.1 Définition d'une ontologie

En philosophie, le terme *Ontologie* désigne la spéculation sur l'être en tant qu'être, sur l'être en soi [27]. On désigne donc *Ontologie* (à remarquer la lettre majuscule) comme une discipline philosophique à part entière. Cependant, dans le domaine de l'intelligence artificielle, le terme ontologie, issu de la philosophie de la connaissance, désigne généralement l'ensemble des concepts d'un domaine. Dans le cadre de la représentation des connaissances, ce terme est employé plus particulièrement

pour décrire la structure d'un ensemble de connaissances : les concepts, relations et contraintes effectivement utilisés pour modéliser un domaine donné [21, 34, 50].

La théorie derrière ces ontologies est considérée comme une théorie du contenu (*content theory*) car elle permet d'identifier les classes d'objets spécifiques dans un domaine et de les mettre en relation [9]. Dans le cas le plus simple, une ontologie décrira une hiérarchie de concepts, construite sur des relations de généralisation et de spécialisation. Ensuite, des relations pourront être ajoutées afin d'exprimer des sémantiques plus complexes. C'est alors que nous pourrons utiliser le mot ontologie pour désigner une ontologie du domaine x ou du domaine y.

2.2.6.2 Rôle d'une ontologie

Une ontologie joue un rôle important dans la clarification de la structure de connaissance. Cette structure est au cœur du bon fonctionnement d'un système d'information. Cependant, il ne faut pas négliger la conceptualisation derrière le vocabulaire. Une mauvaise analyse du domaine d'application peut mener vers une base de connaissances incohérente. Nous pouvons faire ici un parallèle avec l'analyse orientée-objet ou la modélisation d'une base de données. Dans ces deux domaines, une analyse peu approfondie mènera tôt ou tard à des problèmes souvent très coûteux [9].

2.2.6.3 Visualisation d'une ontologie

La visualisation d'une ontologie peut être réalisée sous forme de réseau sémantique [18, 34]. Les réseaux sémantiques permettent de représenter de façon naturelle le formalisme d'une ontologie. En effet, ils sont composés de noeuds (concepts) et de liens pouvant modéliser toutes les relations présentes dans l'ontologie. Un exemple simple de réseau sémantique est illustré à la figure 2.1. Par exemple, on retrouve un lien *is a* entre le concept B et le concept A, ce dernier indiquant que B est une spécialisation de A.

L'une des caractéristiques importante des réseaux sémantiques est qu'il est aisé de représenter un modèle d'héritage. Ainsi, un concept peut hériter des caractéristiques ou des propriétés d'un autre concept plus général, ce qui simplifie la représentation [45].

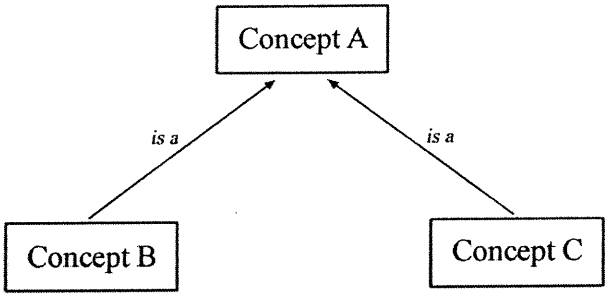

FIG. 2.1 – Exemple simple d'un graphe sémantique

La représentation des connaissances étant expliquée nous allons passer à la procédure d'analyse de ces connaissances. Dans notre système, cette dernière sera effectuée par une technique de reconnaissance de patrons. Nous présenterons donc dans la section suivante les principales méthodes de reconnaissance de patrons pour ensuite nous attarder sur une technique en particulier (cartes auto-organisées de Kohonen).

2.3 La reconnaissance de patrons

La faculté de reconnaître des similitudes entre différentes entités, que ce soit des caractères, des formes ou même des personnes est acquise en très bas âge chez l'être humain. Cette capacité qui semble évidente à première vue, devient vite complexe lorsqu'on tente de la simuler par ordinateur. La discipline que l'on nomme reconnaissance de patrons fait l'étude de cette simulation. Par l'observation de son environnement, elle tente de distinguer des patrons parmi une masse d'information qui semble à première vue chaotique [19].

De la médecine à la sociologie, de la reconnaissance des formes au contrôle de procédés industriels en passant par l'analyse géopolitique, la reconnaissance de patrons, qui vise à simplifier une masse de données complexes, est l'une des démarches fondamentales pour l'analyse d'un phénomène [13]. En effet, cette opération représente pour l'homme, une étape critique dans le processus de prise de décision. Plus le nombre de patrons observés ou connus est élevé, plus la décision choisie risque d'être appropriée [38]. L'utilisation et le perfectionnement de cette discipline dans le domaine de l'intelligence artificielle pourrait se révéler très intéressants.

Les définitions liées au terme *patron* que l'on retrouve dans la littérature diffèrent malheureusement les unes des autres [19]. Dans notre recherche nous nous baserons sur la définition de Watanabe [48] qui définit un patron comme une entité vaguement définie, opposée au chaos, à qui il est possible d'associer un nom ou certaines caractéristiques.

La reconnaissance de patrons peut se faire selon deux différentes formes d'apprentissage, un apprentissage supervisé ou non supervisé.

2.3.1 Apprentissage supervisé

Dans le cas d'un apprentissage supervisé, l'algorithme de reconnaissance de patrons choisi reçoit une série de donnés sous la forme de paires (x_i, y_i) où x_i est une entrée possible et y_i une sortie à laquelle l'entrée x_i est associée. Ces paires (x_i, y_i) sont communément appelées exemples d'entraînement et l'on assume qu'elles proviennent d'une distribution inconnue où le nombre de sorties possibles est fini. L'algorithme tentera de converger vers une fonction $f(x_i) = y_i$ pour tous les i. Ainsi, il sera possible d'associer des entrées encore inconnues à l'une des i sorties prédéterminées [11]. Chaque élément de sortie représente donc un type de patron. Un entraînement adéquat consiste à présenter des exemples connus couvrant tous les champs des sorties

possibles. Ceci résulte en une analyse discriminante pour laquelle chacune des entrées est identifiée comme un membre d'une classe prédéterminée.

2.3.2 Apprentissage non-supervisé

Ce type d'apprentissage consiste à extraire d'un nuage d'information des ensembles ayant des corrélations, des dépendances, des liens de cause à effet, et à tenter de simplifier un grand nombre d'observations en un petit nombre de paramètres pouvant former une catégorie, un patron. Il n'y donc pas de présentation, comme dans l'apprentissage supervisé, d'un ensemble d'exemples constitués de paires (x_i, y_i) représentant un patron précis. Les objectifs de ce type d'apprentissage sont très variés selon le type du problème traité. Un des types de problème qui nous intéresse est la catégorisation, qui tente de partitionner plusieurs entrées en un nombre fixe de sous-ensembles appelés catégories. Chaque entrée d'un même sous-ensemble a des attributs similaires selon une certaine métrique.

2.3.3 Différentes approches

La littérature portant sur la reconnaissance de patrons fait état de plusieurs approches. Nous en présenterons brièvement quatre des plus répandus et reconnus [19]. Parmi ces approches, on retrouve :

- Le jumelage de modèle (*Template matching*) ;
- La classification statistique (*Statistical classification*) ;
- L'approche syntaxique et structurelle (*Syntactic or structural matching*) ;
- Les réseaux de neurones (*Neurals network*).

2.3.3.1 Jumelage de modèle

Cette méthode est l'une des plus anciennes et par le fait même l'une des plus simples. Le terme jumelage (*matching*) est, dans le domaine de la reconnaissance de patrons,

assez générique, et il désigne un calcul de similarité entre deux entités (typiquement des formes à deux dimensions) qui sont du même type. Dans ce cas, un modèle ou prototype du patron recherché est obligatoirement fourni. Lors de la reconnaissance d'une entité, on calcule la similarité de l'entité par rapport à tous les modèles compris dans le système. Cette mesure de similarité, basée sur une forme de corrélation, peut être optimisée lors de l'entraînement du système à partir d'un ensemble de formes. Cette méthode fait appel à un apprentissage supervisé du fait que l'ensemble des patrons à rechercher est prédéterminé.

2.3.3.2 Classification Statistique

Cette approche représente chaque patron prédéterminé sous la forme de x caractéristiques ou x mesures. Ces patrons sont visualisés comme des points dans un espace à x dimensions. Le but est de choisir adéquatement les caractéristiques qui permettront aux patrons d'appartenir à différentes catégories et d'occuper une région compacte et disjointe dans l'espace multi-dimensionnels. L'efficacité de cette approche est déterminée par la qualité de la répartition des patrons dans l'espace. À partir d'un ensemble d'entraînement, constitué de cas appartenant à différents patrons, l'objectif est d'établir des frontières par lesquelles nous pourrons séparer, dans cette espace multi-dimensionnels, les patrons appartenant à différentes catégories. Selon l'approche théorique entourant la classification statistique, ces frontières de décision sont déterminées par une probabilité de distribution des patrons appartenant à différentes catégories [19].

2.3.3.3 Approche syntaxique et structurelle

L'approche syntaxique et structurelle repose sur la structuration sous forme hiérarchique d'un patron. En effet ce dernier est analysé comme une composition de sous-patrons qui sont eux-mêmes composés de sous-patrons encore plus simples. Les patrons élémentaires, les plus simplistes, sont appelés des primitives. Ainsi, un patron complexe

peut être vu en terme d'interrelations entre ces primitives. Il est possible de faire une analogie entre la structure des patrons et la syntaxe d'un langage. Les patrons sont représentés par des phrases appartenant à un langage, les primitives comme l'alphabet de ce langage. De plus, les phrases sont générées selon une certaine grammaire. Ainsi, un patron peut être décrit par un petit nombre de primitives et de règles de grammaires. La grammaire pour chaque classe de patron doit être déduite à partir d'un ensemble d'entraînement. Cette approche est intéressante puisqu'en plus d'effectuer une classification, elle permet de visualiser la composition de ces patrons par la combinaisons de primitives [19]. Seulement, cette approche peut facilement créer une explosion combinatoire des possibilités à analyser, et par conséquent, requérir un ensemble d'entraînement important en nombre et des efforts de calculs énormes.

2.3.3.4 Réseaux de Neurones

Chez l'humain, le siège de l'intelligence et de l'apprentissage est le cerveau. Ce dernier a d'ailleurs grandement inspiré la réalisation d'algorithmes d'apprentissages de type neuronaux, plus communément appelés réseaux de neurones. Ce type de réseau peut être vu comme une centrale de calcul usant de parallélisme massif et constituée de plusieurs processeurs inter-connectés. De façon plus formelle, un réseau de neurones est composé de noeuds, ou d'unités de calcul représentant des neurones artificiels. Des liens dirigés, auxquels des poids numériques sont associés, agissent comme connexions entre les neurones de sortie et ceux en entrée. La principale caractéristique des réseaux de neurones est qu'ils sont en mesure de déchiffrer des relations non-linéaires complexes entre des données d'entrée-sortie [19, 39].

Les réseaux de neurones les plus populaires en ce qui concerne la reconnaissance de patrons font partie de la familles des *feed-forward networks*, incluant le *multilayer perceptron* et le *Radial-Basis Function* [20]. Ces réseaux sont organisés en plusieurs couches de neurones et possèdent des liens unidirectionnels entre ces couches. Un autre type

de réseau de neurones populaire est représenté par les *cartes auto-organisées de Kohonen*, communément appelé *Self-Organizing Map (SOM)* [12, 22, 26]. Ce réseau est plus particulièrement utilisé lors de problèmes de catégorisation et visualisation des données. C'est d'ailleurs l'approche que nous utiliserons à des fins de catégorisation au sein du système *Alice*.

2.3.4 Cartes auto-organisées de Kohonen

Les cartes auto-organisées de Kohonen sont obtenues à partir d'un réseau de neurones artificiels dont la fonction est de révéler les dépendances non linéaires entre plusieurs variables. Il est essentiellement un algorithme de catégorisation, pratiquement indépendant du domaine d'application. Les données en entrée sont présentées sous la forme d'un vecteur de poids appelé *codebook*. Ce réseau repose sur un apprentissage non-supervisé et de type compétitif, c'est-à-dire qu'à chaque itération, un échantillon (*codebook*) de base est présenté à un réseau compétitif composé d'une couche, dont le rôle est de trouver un *gagnant* et de rapprocher les poids de ce gagnant du vecteur présenté en entrée. Seulement, contrairement aux algorithmes purement compétitifs, le neurone gagnant n'est pas l'unique bénéficiaire de l'apprentissage. En effet tout son voisinage, compris sous une certaine distribution, en est affecté. Il en résulte une couche de sortie composée de neurones représentant diverses catégories.

Avant d'expliquer en détail la théorie et le fonctionnement des cartes auto-organisées de Kohonen, nous présenterons, un bref aperçu de l'historique du concept des cartes auto-organisées.

2.3.4.1 Petit historique

Les cartes auto-organisées de Kohonen sont basées sur les recherches de Von der Malsburg [47] qui a étudié les vertébrés supérieurs, et plus particulièrement l'organisation de leurs cartes sensorielles ou topiques. Ces cartes participent, entre autre,

à la simplification d'un ensemble de données. Ainsi à partir de représentations simplifiées, les décisions sont beaucoup plus faciles à prendre. En cherchant à comprendre la formation de ces cartes topiques, Von der Malsburg a proposé son modèle d'auto-organisation fondé sur l'observation de la rétinotopie [49].

Le modèle qu'il a proposé afin de représenter les cartes rétinotopiques est un réseau de la forme illustrée à la figure 2.2. Ce réseau comporte deux couches, soit une couche inférieure et une seconde couche dite supérieure. La couche inférieure représente des capteurs visuels organisés en grille bidimentionnelle $n = w_i \text{x} h_i$. Par ailleurs, la couche supérieure est représentée par un réseau de neurones sous forme d'une grille de $N = w_o \text{x} h_o$ neurones. Ces deux couches sont complètement interconnectées, chaque neurone est composé d'un vecteur-poids d'une dimension n puisque chaque capteur visuel est connecté directement avec chacun des neurones.

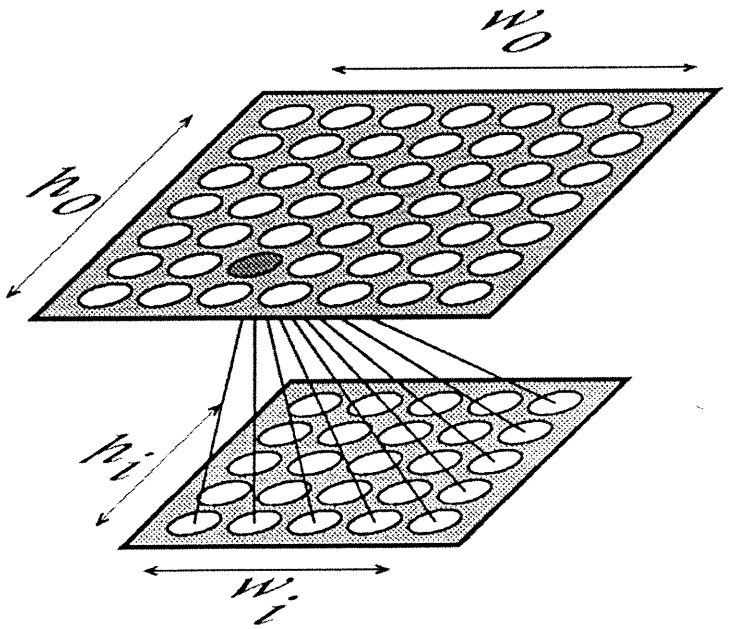

FIG. 2.2 – Premier modèle des cartes auto-organisées de Von der Malsburg [13]

Au sein de la grille de neurones, il existe des liens entre chacun d'eux. Ces liens sont de nature excitatrices à courte distance et inhibitrices à longue distance selon un profil de chapeau *Mexicain*, illustré à la figure 2.3. Lorsqu'un motif d'activité ξ des capteurs est présenté, les neurones répondent par une activité plus ou moins forte selon la proximité de leur vecteur-poids à ce motif ξ. Une coopération/compétition a lieu entre les neurones qui s'activent ou s'inhibent mutuellement au moyen des connections latérales. Il émerge de cette compétition non pas un seul gagnant mais une bulle d'activité centrée sur des neurones voisins. L'adaptation est faite selon une règle de Hebb suivie d'une re-normalisation des poids. Il en résulte un rapprochement des poids des neurones de la bulle vers ξ [13]. Ce modèle d'auto-organisation repose sur la volonté que les unités dont les vecteurs-poids sont proches dans l'espace d'entrée soient aussi proches par leur emplacement physique. Cette notion d'emplacement est transposée sur la couche de neurones par un apprentissage non seulement du neurone gagnant mais également des neurones voisins.

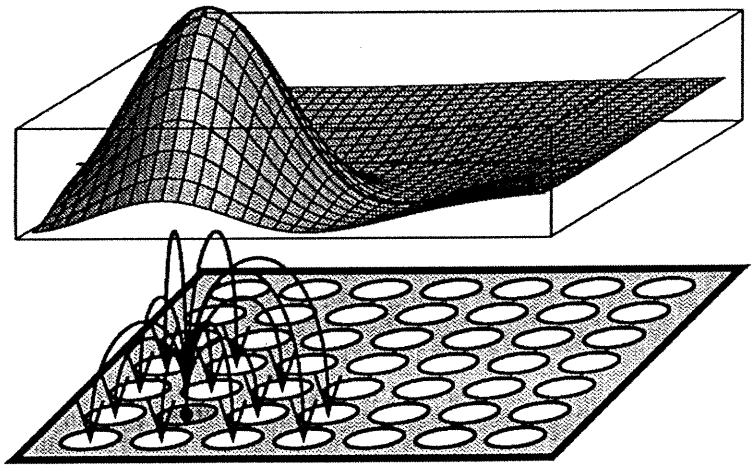

FIG. 2.3 – Profil du chapeau *Mexicain* [13]

Dans le cas du problème de rétinotopie, le but est d'apprendre, d'une façon non supervisée, à projeter un stimulus localisé à un endroit du plan de capteur vers

un endroit correspondant dans la carte de neurones. C'est donc une cartographie position-position et par le fait une catégorisation qui permettra une simplification d'un ensemble de données [13].

2.3.4.2 Algorithme De Kohonen

Les cartes auto-organisées de Kohonen ont été créées suite à deux simplification du modèle de Von der Malsburg. En premier lieu, la couche inférieure représentant les capteurs visuels est remplacé par un vecteur à composantes continues. La structure prend donc la forme de la figure 2.4. D'autre part, le mécanisme de formation de la bulle par le jeu des connexions latérales ou profil de chapeau *Mexicain* est remplacé par la prise en compte de l'effet de cette bulle, il n'y a plus de connexions latérales. On procède à un calcul de gagnant, mais l'adaptation se fait pour tous les neurones dans le voisinage du gagnant.

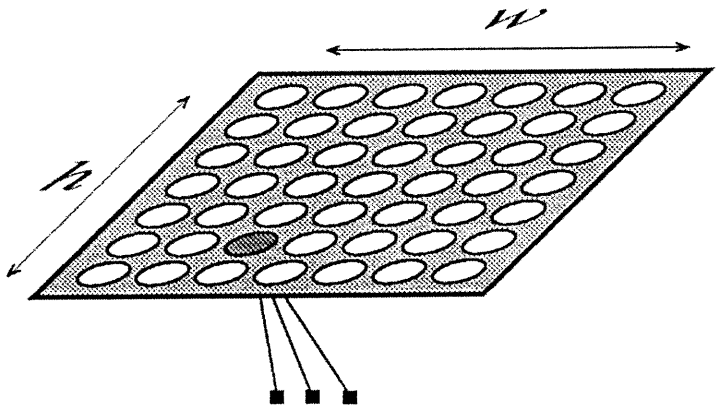

FIG. 2.4 – Modèle des cartes auto-organisées de Kohonen [13]

Les cartes auto-organisées de Kohonen se composent d'un vecteur d'entrées

$$x(t) = [x_1(t), x_2(t), \ldots, x_j(t)] \tag{2.1}$$

appelé vecteur-poids. Il est représenté par une couche d'entrées de dimension j. Ce vecteur $x(t) \in \Re^n$ est connecté à une couche de neurones, en sortie, via des liens auxquels des poids u_{ij} sont associés. Ainsi, chacun des neurones qui composent cette couche de sortie est représenté par un vecteur de référence défini par :

$$m_i(t) = [u_{i1}(t), u_{i2}(t), \ldots, u_{ij}(t)] \in \Re^n \tag{2.2}$$

Initialement, les valeurs u_{ij} associées aux liens entre les deux couches sont générées de façon aléatoire. De plus, la couche de neurones peut-être de forme rectangulaire, hexagonale ou bien même irrégulière. Par contre, il est démontré que de meilleurs résultats ont été obtenu avec une représentation hexagonale [26]. Une fois que la dimension et que la forme de la couche de neurones sont déterminés, la première étape consiste à entraîner le réseau de neurones à partir d'un ensemble d'échantillons.

Pour chaque vecteur d'entrées $x(t)$ présenté au réseau, il y a détermination d'un neurone dit gagnant. Le neurone choisi est celui qui possédera le vecteur référence le plus similaire au vecteur d'entrées. Nous pouvons traduire cette notion de similarité par une notion de distance entre vecteurs. Les distances le plus couramment utilisées dérivent de la distance généralisée de Minkowski $d_m(a,b) = (\sum_{k=1}^{n} \|a_k - b_k\|^p)^{\frac{1}{p}}$ et sont la distance Euclidienne ($p = 2$), la distance de Manhattan ($p = 1$) et la distance Chessboard ($p = \infty$). Ces trois mesures ne donnent pas des résultats similaires et les deux dernières (Manhattan, Chessboard) peuvent mener à des effets pervers sur le processus d'auto-organisation [13]. C'est pourquoi la métrique choisie lors du calcul de similarité est basée sur la distance euclidienne minimum entre deux vecteurs. Soit les vecteurs $x = (\xi_1, \ldots, \xi_n)$ et $y = (\eta_1, \ldots, \eta_n)$. La distance euclidienne est déterminée par

$$d_E(x,y) = \|\xi - \eta\| = \sqrt{(\xi_1 - \eta_1)^2 + (\xi_2 - \eta_2)^2 + \ldots + (\xi_n - \eta_n)^2} \tag{2.3}$$

Le neurone gagnant c est alors identifié par l'équation

$$c = \|x(t) - m_c(t)\| = \min_i \|x(t) - m_i(t)\| \tag{2.4}$$

Le neurone gagnant ainsi que son voisinage auront la possibilité de modifier leurs vecteur-référence afin de se rapprocher, selon une certaine mesure, de l'échantillon $x(t)$ présenté. En effet, l'algorithme derrière les cartes auto-organisées de Kohonen peut être vu comme une quantification vectorielle sous contraintes d'une structure de voisinage entre les unités. A priori, la quantification vectorielle ne s'intéresse pas à la disposition physique des unités. L'apprentissage compétitif, dont la quantification vectorielle est à la base, est un processus d'adaptation dans lequel les neurones deviennent graduellement sensibles à différentes catégories de vecteurs d'entrées. Le groupe sélectionné ayant droit à un certain apprentissage, soit le neurone gagnant et son voisinage, sont modifiés selon l'équation

$$m_i(t+1) = m_i(t) + \alpha(t) h_{ci}[x(t) - m_i(t)] \tag{2.5}$$

Ainsi, lors de l'apprentissage, les neurones de la couche de sortie proches topographiquement (jusqu'à une certaine distance) seront activés dans le but d'apprendre, de se rapprocher, de la même entrée. Ceci aura un effet de *relaxation* sur les neurones du même voisinage, résultant en une couche de neurones ordonnée à la fin de l'apprentissage. L'équation 2.5 est composée de deux termes jouant un rôle primordial, le facteur d'apprentissage $\alpha(t)$ et la fonction du voisinage $h_{ci}(t)$.

Le nombre d'échantillons lors de l'entraînement du réseau est fini et s'effectue donc à l'intérieur d'un temps t. Le facteur d'apprentissage appliqué aux neurones sélectionnés diminue en fonction de l'augmentation du temps t. $\alpha(t)$ est donc inversement proportionnel à t. L'échelle des valeurs de $\alpha(t)$ est telle que $0 < \alpha < 1$. Il en résulte donc un apprentissage important au début de l'entraînement du réseau dont l'importance décroît jusqu'à devenir infime à la fin de l'apprentissage.

Le terme $h_{ci}(t)$ présent dans la formule 2.5 est un facteur de pondération en fonction de la distance dans la grille autour du gagnant. Cette fonction, que l'on retrouve souvent sous l'appellation *voisinage gaussien*, détermine la mesure de l'apprentissage selon la distance du neurone par rapport au neurone gagnant. Le voisinage gaussien est défini par

$$h_{ci} = \exp(-\frac{\|r_c - r_i\|^2}{2\sigma^2(t)}) \tag{2.6}$$

où $r_c \in \Re^2$ et $r_i \in \Re^2$ sont les vecteurs de location sur la grille de sortie du neurone c et i respectivement . Donc lorsque $\|r_c - r_i\|$ augmente $h_{ci} \rightarrow 0$. Par ailleurs, la variable σ est le rayon du voisinage qui décroît graduellement avec le temps au même titre que le facteur d'apprentissage. Le cartes auto-organisées de Kohonen sont très sensibles aux différents paramètres mentionnés ci-dessus, tous les détails concernant l'utilisation de ces derniers seront présentés à la section 3.6.1.

À la fin de l'apprentissage, chaque neurone représente une catégorie. Ainsi, pour un nouveau vecteur en entrée, le réseau détermine le neuron le plus similaire à ce dernier et lui associe la catégorie correspondante.

2.4 Conclusion

Nous avons abordé dans ce chapitre deux aspects reliés aux systèmes à base de connaissances, la représentation et l'analyse de ces dernières. En première lieu, nous avons brièvement présenté les différents types de représentation des connaissance. La représentation choisie est la représentation basée sur les réseaux. Nous avons jumelé, à cette représentation, la notion d'ontologie qui désigne le contenu d'une base de connaissances. Cette ontologie est représentée de façon graphique sous forme de graphe sémantique.

En second lieu, nous avons présenté différentes techniques de reconnaissance de patrons. Parmi celles-ci le modèle basé sur les réseaux de neurones a été retenu. De façon plus précise, les cartes auto-organisées de Kohonen seront à la base de l'analyse des différentes entités constituées de connaissances. Ce type de réseau de neurones a été choisi pour deux raisons principales. Le première est qu'il est essentiellement un algorithme de catégorisation performant et bien établi. La seconde raison réside dans le fait qu'il utilise un apprentissage non-supervisé, obligatoire dans notre cas du fait qu'il n'existe pas de classes définies au sein des entités que nous voulons catégoriser.

L'implantation des concepts présentés jusqu'à maintenant a résulté en la création du système *Alice*. C'est ce que nous allons aborder dans le chapitre suivant.

Chapitre 3

Le système Alice

3.1 Introduction

La connaissance est à priori un terme difficile à cerner. Nous la définissons comme un ensemble de compétences, d'expériences, de savoirs et parfois même d'intérêts. Cette notion de connaissance est également une difficulté quand il s'agit de la gérer, de l'utiliser de façon optimale ou même de la formaliser. Elle se retrouve sous diverses formes et différents niveaux, elle peut être concrète, pratique, ou complètement théorique. Cette même connaissance se situe dans un espace flou, il n'existe pas de connaissance absolue ou minimale, ni même de connaissance de bas niveau (qui ne vaudrait pas la peine que l'on s'y attarde). Cet espace flou est constitué de dépendances directes et indirectes, de hiérarchies et d'enchaînements logiques ou non logiques.

La connaissance est emmagasinée par l'homme au moyen de différents stimulus. Son niveau de maîtrise est différent pour chacun. Nous croyons que la seule façon d'utiliser cette connaissance à son maximum est d'en faire une coordination à l'échelle humaine. Cette hypothèse se base sur le principe bien connu de *diviser pour régner*. Chaque personne possède des compétences, des connaissances qui lui sont propres, qu'elle maîtrise à un certain niveau. Personne ne peut posséder une connaissance *totale*. Si

deux personnes ont les mêmes connaissance ou presque, il serait intéressant d'analyser les expériences associées à ces dernières. Ceci dit, la connaissance totale ou parfaite pourrait être obtenue en mettant en parallèle tous les cerveaux humains. Remettons maintenant ces principes dans un contexte un peu plus réaliste et focalisons sur le système *Alice*.

L'idéologie derrière le système *Alice* est de permettre d'utiliser les connaissances d'un ensemble d'individus par un autre individu. Nous avons choisi le modèle d'une organisation comme ensemble d'individus. Dans une organisation, l'échange d'informations et de connaissances entre les individus est primordial. Cependant, cette communication n'est pas toujours possible, ni facile, plusieurs contraintes font souvent barrage à cette coopération qui pourrait sembler naturelle. De plus, il n'est pas évident de diriger une personne ayant un besoin de connaissances vers un individu pouvant potentiellement l'aider. Le système *Alice* a pour objectif d'orienter les membres d'une organisation dans cet ensemble de connaissances. Ces principales tâches dans ce cadre seront :

- Aider les personnes à formuler des requêtes à partir d'un vocabulaire qui leur sont commun.
- Présenter les entités de l'organisation susceptibles de répondre aux requêtes formulées.
- Analyser de façon graphique la distribution des connaissances à l'intérieur de l'organisation.

Nous présenterons dans ce chapitre les différents modules du système *Alice* en commençant par son architecture globale. Précisons que nous avons conçu et fait l'implantation de tous les modules du système *Alice*.

3.2 Architecture du système Alice

L'architecture d'*Alice*, illustrée à la figure 3.1, se compose de six modules semi-indépendants. L'analyse et la modélisation orientée-objet préalable à la construction du système sont responsables de cette indépendance. Nous retrouvons donc les modules suivants :

- La gestion des connaissances
- La structure de l'organisation
- La base de profils des entités
- La catégorisation des profils
- La visualisation des connaissances
- L'interface des requêtes

Chaque module de cette architecture sera analysé en détail autant du point de vue pragmatique que théorique.

3.3 La gestion des connaissances

L'un des objectifs premiers de ce travail est d'optimiser l'utilisation des connaissances réparties sur un ensemble d'individus, que ce soit dans le cadre d'une organisation ou dans le cadre de tout autre regroupement de spécialistes. Cette connaissance doit être extraite de chaque individu et représentée sous une forme choisie, sans en altérer la forme ou le sens. La forme de modélisation choisie est basée sur le principe d'une ontologie représentée par un graphe sémantique.

3.3.1 Ontologie des connaissances

Nous croyons que du fait du formalisme apporté par ce type de structure (ontologie), il est possible de modéliser la connaissance dans toute sa complexité. Chaque connaissance est représentée par un concept du domaine d'application. Le rôle de l'on-

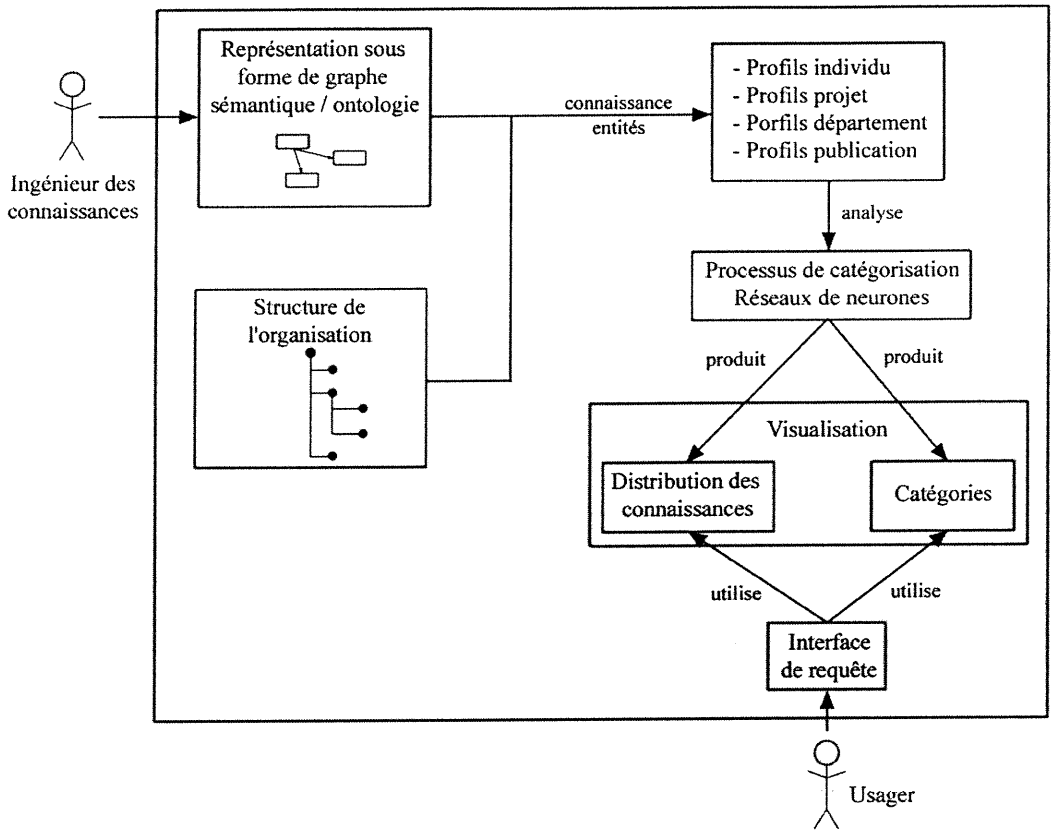

FIG. 3.1 – Architecture du système *Alice*

tologie est de représenter les relations existantes entre les différentes connaissances ou concepts. Ces relations sont transposées sous forme de liens, au nombre de quatre dans notre système :

- Lien de similarité ;
- Lien de généralisation ;
- Lien de spécialisation ;
- Lien d'association.

Derrière chaque lien se retrouve une sémantique particulière qui sera prise en considération lors de la construction des profils d'individus, basés sur cette ontologie des connaissances. Nous présenterons maintenant plus en détail chaque type de lien.

3.3.1.1 Lien de similarité

Le lien de similarité, comme son nom l'indique, associe deux concepts similaires. Il est utilisé lorsque deux concepts similaires sont très représentatifs du domaine d'application. Un synonyme de moindre importance sera jumelé à un concept au niveau de la base de données et non dans le graphe de connaissances. Notons ici qu'un lien de similarité ne veut pas nécessairement indiquer une synonymie. Par exemple, les termes *browser* et *client* (si l'on parle d'une application client-server internet) sont similaires sans être des synonymes. C'est d'ailleurs la raison pour laquelle un degré de similarité est associé à ce type de lien. Ce degré de similarité peut prendre trois valeurs différentes : *faible*, *modéré* ou *élevé*. Il est déterminé manuellement par le concepteur du graphe de connaissances et est donc sujet à la subjectivité de ce dernier.

Par l'intermédiaire de ce lien, on peut faire l'hypothèse qu'un individu ayant une connaissance x possède des chances d'avoir dans son bagage la connaissance y selon le degré de similitude entre x et y. Cette dernière observation peut ne pas être détectée explicitement lors de la modélisation des connaissances de cet individu.

3.3.1.2 Lien de généralisation, spécialisation

Nous pouvons mettre dans la même catégorie les liens de généralisation et de spécification puisque ces deux liens sont tout simplement inverses. Un lien de généralisation aura comme point de départ une connaissance plus générale que celle du point d'arrivée et vice-versa pour le lien de spécification. Il est donc évident, contrairement au lien de similarité, que ces deux types de liens sont obligatoirement dirigés.

L'utilisation de ces liens peut permettre d'augmenter l'efficacité d'une requête qui aurait pu mener vers une information moins riche, si ces liens n'avaient pas existé. En effet, il n'est pas erroné de supposer qu'un individu ayant des connaissances associées à un concept plus spécialisé que le concept recherché ait également des connaissances,

ou une partie des connaissances, sur le concept plus général. Cependant, l'inverse ne peut être vérifié. Il est alors possible d'élargir l'espace de recherche sans diminuer la qualité des solutions proposées par le système.

3.3.1.3 Lien d'association

Un lien d'association représente une relation entre deux connaissances à priori distinctes. L'utilisation d'un concept(A) par un autre concept(B) donnera lieu à une connexion de ces deux concepts par un lien d'association. Ce type de lien est souvent très proche du domaine d'application. Ainsi, si nous reprenons l'exemple ci-dessus, un individu ayant une connaissance au niveau du concept B a de forte chance de connaître le concept A.

3.3.2 Représentation graphique de l'ontologie

Comme nous avons pu le constater, une ontologie est composée de concepts (connaissances) et de relations. La forme de représentation qui semble toute naturelle est l'utilisation d'un graphe sémantique. Les concepts formeront les noeuds du graphe tandis que les relations(liens) seront décrites par des arêtes, dirigées ou non dirigées, entre ces noeuds.

Ce graphe est sujet à plusieurs modifications, que ce soit par l'ajout ou la suppression de connaissances ou de relations. De plus, la taille du graphe peut devenir importante en très peu de temps, il serait donc intéressant d'avoir la possibilité de ne travailler que sur une section de ce dernier. Ces observations nous ont convaincu de construire un éditeur graphique (figure 3.2) permettant de manipuler avec facilité cette représentation des connaissances.

La tâche première de cet éditeur est de faciliter la construction d'un graphe de connaissances. Nous décrirons les principales fonctions suivantes :

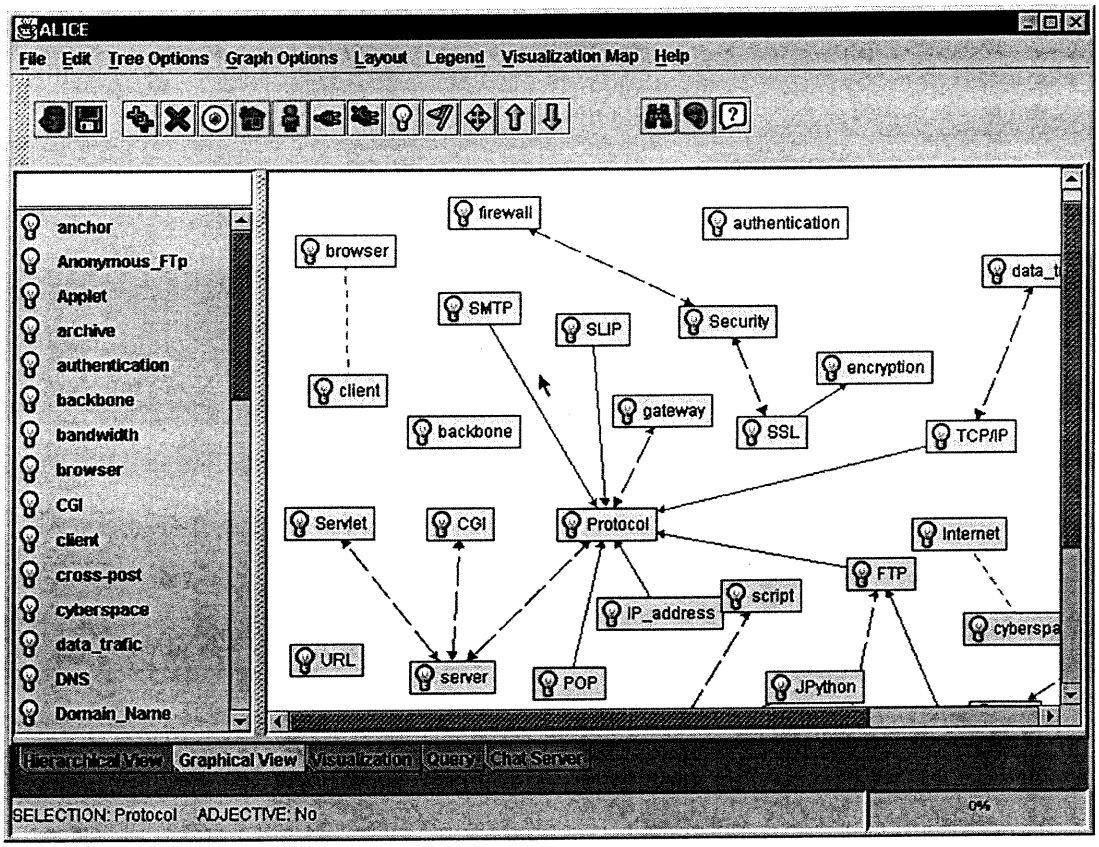

FIG. 3.2 – Ontologie des connaissances sous la forme d'un graphe sémantique

- Ajout et suppression d'une connaissance
- Ajout et suppression d'un lien
- Positionnement et visualisation des connaissances

3.3.2.1 Ajout et suppression d'une connaissance

L'ajout d'une connaissance se fait par la sélection de l'icône appropriée dans la barre d'outils ou la barre de menus. D'ailleurs, il en est de même pour toutes les opérations que nous allons présenter. Chaque fonction est accessible soit par la barre d'outils, soit par la barre des menus. Une fois l'icône sélectionnée, une fenêtre, illustrée à la figure 3.3, apparaît à l'usager.

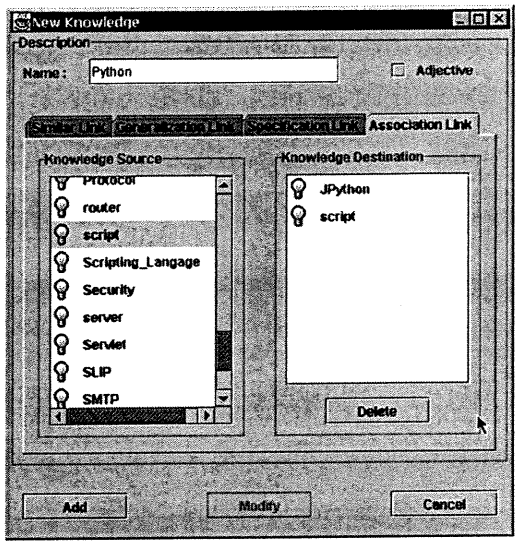

FIG. 3.3 – Ajout d'une nouvelle connaissance

Par l'intermédiaire de cette fenêtre l'usager doit spécifier les différents attributs de la nouvelle connaissance. Il est également possible de définir les liens qui seront attachés à cette connaissance. Il suffit de glisser (*Drag and Drop*) les connaissances désirées dans les listes dédiées aux différents types de liens. Lorsque l'usager décide d'ajouter la connaissance, il ne lui reste qu'à appuyer sur le bouton *add* situé dans le bas de la fenêtre. Dans le cas où un champ obligatoire est manquant, exemple le nom de la connaissance, un message indiquant l'erreur est lancé et permet à l'usager de corriger la situation. Afin de ne pas reprendre cette description pour chaque fonction de l'éditeur, nous aimerions préciser que le système est en mesure de toujours indiquer à l'usager si une erreur s'est introduite. Un message d'erreur permettra à ce dernier de reprendre la situation sous contrôle ou tout simplement d'arrêter(*cancel*) la procédure. La connaissance ajoutée, nous pourrons voir apparaître la nouvelle connaissance ainsi que les connaissances reliées à cette dernière par des liens.

Cette même fenêtre (figure 3.3) sera présentée à l'usager lorsqu'il voudra analyser les propriétés d'une connaissance. Il pourra à ce moment modifier les différents attributs

ainsi que les liens qui composent le sous-graphe associé à cette connaissance. La seule différence est représentée par le texte du bouton *add* qui sera remplacé par le texte *modify*.

La suppression d'une connaissance débute par la sélection de celle-ci. Une fois la sélection effectuée, le processus de suppression est lancé par la barre des menus, ou par la barre d'outils ou encore par un menu flottant accessible en *cliquant* sur le bouton droit de la souris. Ce processus éliminera la connaissance sélectionnée de même que tous les liens qui y sont attachés. Pour toute suppression d'une entité, que ce soit une connaissance, un lien ou autres, un message de confirmation sera affiché, et l'usager devra l'approuver pour pouvoir continuer.

3.3.2.2 Ajout et suppression d'un lien

Pour faire l'ajout d'un lien il faut tout d'abord sélectionner deux connaissances (source, destination). Ensuite, les différents types de liens sont présentés à l'usager afin qu'il puisse faire son choix. Une vérification est alors effectuée afin d'éviter les relations circulaires ou non autorisées. Dans ce dernier cas, un message donnant des précisions sur le problème est lancé. La suppression d'un lien se fait par le même procédé, c'est-à-dire la sélection des connaissances puis le lancement du processus de suppression.

Notons que la représentation graphique de chaque type de lien est différente. En effet, nous retrouvons dans le système :

- Un trait rouge en pointillés (petits traits) pour le lien de similarité.
- Un trait bleu dirigé pour les liens de généralisation et de spécialisation (indiquant l'origine et la destination).
- Un trait noir pointillés avec une alternance de grands et de petits traits pour le lien d'association. Ce lien est bidirectionnel (flèche à chaque extrémité).

3.3.2.3 Positionnement et visualisation des connaissances

Lors de l'enregistrement du graphe des connaissances, il est à noter que le système *Alice* garde la position de l'emplacement de ces dernières. Aussi, lors du démarrage du système, le graphe de connaissance est-il toujours sous sa dernière configuration. Ceci permet pratiquement d'éliminer l'utilisation d'algorithmes qui gèrent la position des connaissances au sein du graphe. Ce type d'algorithme peut rapidement devenir complexe pour ce type de représentation.

Les connaissances visibles, sous la forme d'un graphe sont celles sélectionnées dans la liste de gauche. Il est d'ailleurs possible de sélectionner ou dé-sélectionner les connaissances de cette liste afin de n'afficher qu'un sous-graphe. Lorsqu'il y a une sélection dans la liste, toutes les connaissances rattachées à celle qui est sélectionnée sont également affichées.

Plusieurs autres petits détails et fonctions existent dans notre système mais ils ne seront pas décrits dans ce mémoire. Ils ont comme tâche première de faciliter la construction du graphe et d'augmenter la qualité de l'aspect visuel. Ils n'interviennent pas dans les fonctions de base obligatoires.

3.3.3 Découverte de nouvelles connaissances

La construction de l'ontologie des connaissances est en grande partie réalisée par un ingénieur des connaissances spécialisé dans le domaine d'application. Le bon fonctionnement du système est en grande partie relié à cette ontologie. Elle doit avoir une représentation adéquate et complète du domaine d'application. Dans le cas contraire les performances du système en seront considérablement affectées. Cette ontologie se doit d'être en constante évolution, à l'image du domaine d'application. L'ajout de nouvelles connaissances peut alors se faire soit manuellement comme lors de la

construction initiale ou par un processus semi-automatique. Nous décrirons dans cette section l'approche dite semi-automatique dédiée à l'ajout de nouvelles connaissances.

3.3.3.1 Fréquence des Mots

La découverte de nouvelles connaissances ou concepts est basée sur des théories que l'on retrouve dans le domaine de la recherche d'information (RI). Ce domaine de la RI est assez vaste. Dans notre cas, nous nous concentrerons sur les méthodes souvent utilisées lors de l'indexation de documents. En effet, le but derrière les méthodes d'indexation est de retrouver les concepts les plus importants dans un document. Ces concepts sont représentés sous la forme d'un mot ou de plusieurs mots (habituellement deux mots, mots-composés). Un index est un mot clé qui possède son propre sens et est habituellement sous la forme d'un nom [3].

Une approche intuitive repose sur l'hypothèse selon laquelle un mot ayant une occurrence élevée est un concept qui représente bien le document qui le contient [3]. Par le même raisonnement, on peut conclure que les mots ayant une faible occurrence sont peu intéressants. Toutefois, les mots ayant une occurrence très élevée sont souvent des mots que l'on retrouve dans les *stop lists* tels que (a,the,that,is, etc.), c'est pourquoi un seuil maximal du nombre d'occurrences doit être déterminé afin d'éviter ces derniers. Ainsi, après avoir déterminé un seuil supérieur et inférieur, les mots les plus représentatifs seront ceux situés au centre de la distribution.

Toutefois, un mot ayant une fréquence centrée par rapport à la distribution mais se retrouvant dans la plupart des documents n'est pas nécessairement représentatif du domaine d'application. En effet, un mot se trouvant dans tous les documents est peu discriminant. Nous nous attarderons donc sur la valeur de discrimination associée à un concept, une connaissance.

3.3.3.2 Valeur de Discrimination

La notion de discrimination réfère au fait qu'un terme permet de distinguer un document des autres documents. Un terme ayant une valeur de discrimination élevée apparaît donc, selon cette logique, dans un nombre de documents réduit. Un terme qui apparaît dans tous les documents n'est pas discriminant. Cette notion peut également s'appliquer aux connaissances d'un individu. Par exemple, chaque domaine d'application se base sur certaines notions qui sont connues par tous les individus du même domaine. Ainsi, ces notions ne sont pas représentatives ou particulières à un individu. Elles seront donc peu utiles à la catégorisation de ces derniers. Les connaissances étant extraites des documents qui leurs sont associés, ce facteur de discrimination peut devenir à la fois un élément de précision et une aide à la découverte de nouvelles connaissances.

Le calcul de cette valeur repose sur le degré de densité associé à un ensemble de documents [40]. Prenons deux documents au sein de cet ensemble, D_i et D_j, représentés sous la forme vectorielle

$$\overrightarrow{D_i} = (p_{1,i}, p_{2,i}, \ldots, p_{k,i}) \tag{3.1}$$

où $p_{k,i}$ est le poids du terme k dans le document d_i. Une mesure de similarité $sim(D_i, D_j)$ peut être utilisée pour calculer la similarité de ces deux documents. La fonction de similarité que nous utilisons est le cosinus de l'angle entre les deux vecteurs représentant les documents [3]. Cette formule est normalisée, c'est pourquoi elle retourne une valeur entre 0 et 1.

$$sim(D_i, D_j) = \frac{\sum_{m=1}^{k} w_{m,i} \times w_{m,j}}{\sqrt{\sum_{m=1}^{k} w_{m,i}^2} \times \sqrt{\sum_{m=1}^{k} w_{m,j}^2}} \tag{3.2}$$

La densité d'un ensemble de documents est définie par la matrice composée des mesures de similarité entre tous les couples possibles des documents (D_i, D_j) excepté lorsque $i = j$. Elle représente le degré *d'entassement* dans *l'espace* des documents

[40]. Ce degré de densité est défini par

$$d = C \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} sim(D_i, D_j) \tag{3.3}$$

C étant une constante égale à $\frac{1}{n(n-1)}$. En effet lorsque les n documents sont identiques, $sim(D_i, D_j) = 1$ pour tous les couples et D atteint alors son maximum d'où

$$d = C \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} 1 = C \cdot n(n-1) \tag{3.4}$$

Ce degré de densité peut être calculé de façon plus efficace en construisant un document appelé *centroïde* et noté $\overline{D}$ [41, 40]. Ce document artificiel est composé de la fréquence moyenne de chacun des termes présents dans le corpus.

$$FreqMoy_k = \frac{1}{n} \sum_{i=1}^{n} Freq_{ik} \tag{3.5}$$

$Freq_{ik}$ étant la fréquence du terme k dans document i. Le degré de densité est alors calculé comme la somme des mesures de similarité entre chaque document et le centroïde.

$$d = \frac{1}{n(n-1)} \sum_{i=1}^{n} sim(\overline{D}, D_i) \tag{3.6}$$

Notons d_k la densité du corpus où le terme k est éliminé de tous les documents. Si ce terme k est initialement répandu uniformément dans la majorité des documents avec un taux de fréquence important, la similarité entre chaque document aura donc tendance à diminuer. La densité du corpus augmentera et donc la valeur d_k diminuera par rapport à la valeur de la densité originale. Dans le cas contraire, si le terme k ne possède une forte fréquence que dans quelques documents, la valeur de similarité entre chaque document augmentera, la densité de l'espace du corpus diminuera et la valeur d_k augmentera. Si un terme est peu fréquent à la fois dans le corpus de documents et dans les documents où il apparaît, il aura le même effet qu'un terme uniformément distribué.

Cette valeur de discrimination sera donc calculée par la soustraction de d à d_k.

$$dv_k = d_k - d \tag{3.7}$$

Ainsi, nous pouvons catégoriser en trois groupes l'utilité des différents termes d'un document selon leur taux de discrimination :

1. Les termes possédant un taux de discrimination dv_k élevé sont des termes qui représentent bien le document.
2. Les termes possédant un taux de discrimination dv_k proche de 0 sont des termes plutôt neutres apportant peu d'information sur le document.
3. Les termes possédant un taux de discrimination dv_k négatifs sont des termes pratiquement inutiles.

3.3.3.3 Proposition d'une connaissance

Par l'intermédiaire de la valeur de discrimination, il nous est possible, à chaque analyse d'un document, de vérifier si de nouvelles connaissances peuvent être proposées à l'ingénieur de connaissances. L'entrée en jeu de l'ingénieur explique pourquoi la méthode utilisée dans la découverte de nouvelles connaissances est semi-automatique. Il revient à ce dernier de prendre la décision finale. Si le système propose une connaissance et que celle-ci est rejetée par l'ingénieur de connaissances, le système ne générera plus cette proposition. Il en est de même pour une proposition acceptée.

Pour qu'une proposition soit lancée, un certain seuil associé à la valeur de discrimination doit être respecté. Ce seuil est calculé selon la moyenne des valeurs de discrimination de certaines connaissances choisies manuellement. Évidemment, ces connaissances doivent être connues pour avoir un effet discriminant sur le domaine d'application. Ce seuil n'est pas fixe et peut-être modifié en tout moment par l'ingénieur des connaissances.

3.4 La structure de l'organisation

Le terme organisation est ici synonyme d'un ensemble d'individus ayant des connaissances particulières dans un champ d'application restreint. La structure de l'organisation, illustré à la figure 3.4, a pour but de modéliser cette dernière sous tous ces angles. Cette modélisation est basée sur un nombre fini d'entités que l'on retrouve dans la plupart des organisations, quel que soit leur domaine d'application. Ces entités sont au nombre de quatre dans notre système :

- Département ;
- Individu ;
- Document ;
- Projet.

Plusieurs relations existent au sein de ces quatre entités. Ces relations sont fondées sur une hiérarchie particulière. Cette dernière doit répondre aux critères suivants :

- Un département est composé d'un ou plusieurs individus.
- Un individu peut appartenir à aucun ou plusieurs départements. Il peut être l'auteur ou le co-auteur de plusieurs documents.
- Un document est associé à un ou plusieurs individus
- Un projet est associé à un ou plusieurs individus. Il peut également être associé à un ou plusieurs documents relatant une information associée au projet.

Les règles mentionnées ci-dessus sont nécessaires à l'analyse d'une entité.

3.4.1 Construction de la structure de l'organisation

La structure de l'organisation peut être visualisée sous la forme d'une hiérarchie d'entités. Nous proposons deux représentations possibles au sein du système *Alice*. La première est une arborescence de type *Windows (système d'exploitation)* illustrée à la figure 3.4. La seconde est un graphe sémantique.

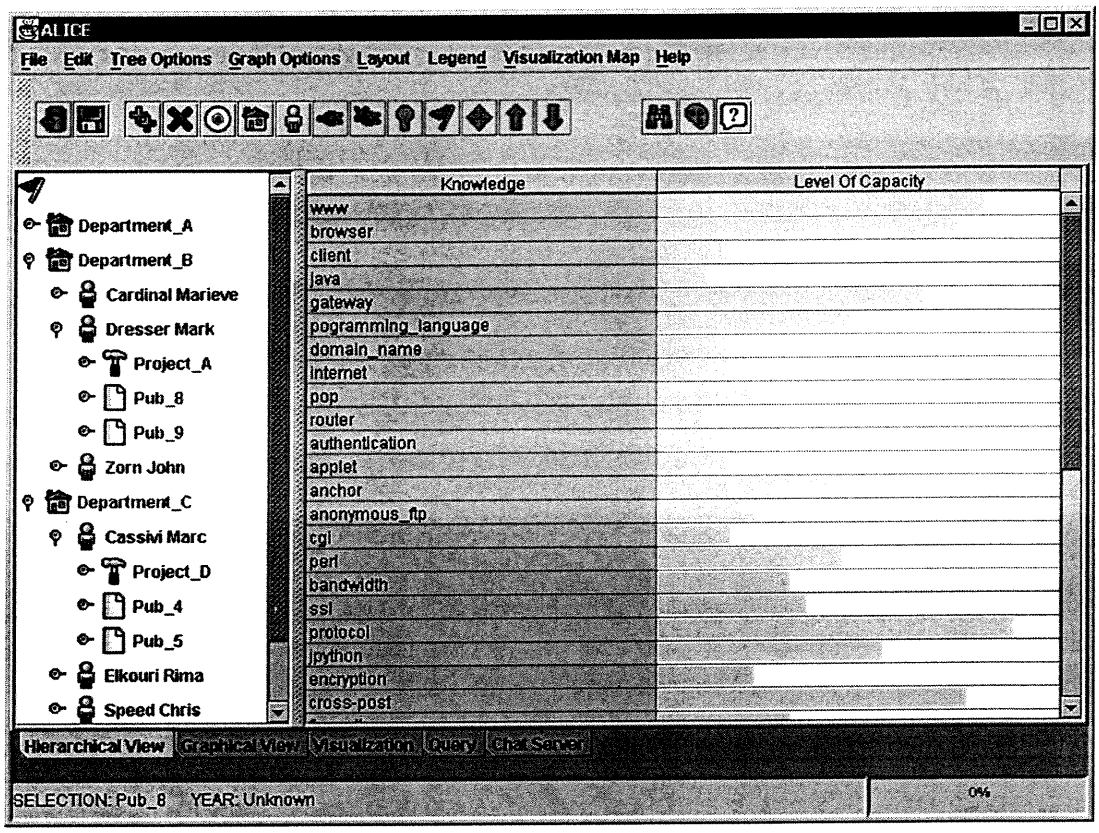

FIG. 3.4 – Structure de l'organisation sous la forme d'une arborescence

3.4.1.1 Forme Arborescente

Dans la première représentation, l'ajout d'une entité est basé sur la procédure utilisée sous le système d'exploitation *Windows*. C'est-à-dire qu'il faut faire la sélection d'un noeud (parent) pour ensuite ajouter un nouveau noeud qui sera alors l'enfant. Le lien entre les deux entités est créé automatiquement. Dans le cas où plusieurs types d'entités peuvent être l'enfant d'un même parent, un choix sera présenté à l'usager. C'est alors qu'une fenêtre, associée au type de l'enfant choisi, sera affichée. Dans cette fenêtre les attributs obligatoires et optionnels peuvent être spécifiés. La fenêtre à la figure 3.5 montre l'ajout d'un projet. Il est par la suite toujours possible d'accéder

et de modifier ses attributs. La même fenêtre est utilisée pour ces deux dernières fonctionnalités.

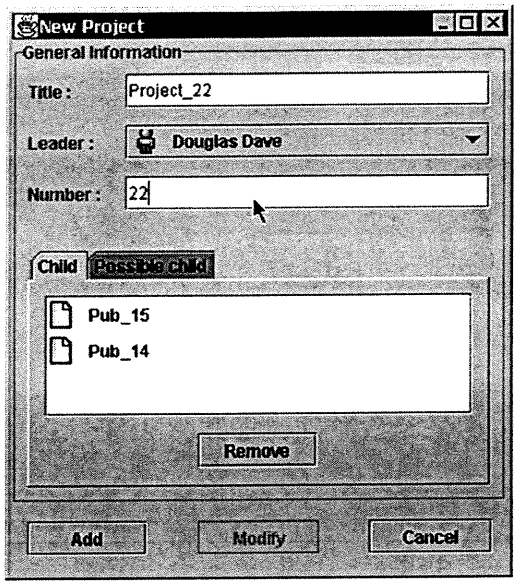

FIG. 3.5 – Ajout d'un nouveau Projet

La suppression d'une entité se fait par une sélection de celle-ci avant de l'éliminer. Tous les liens qui mènent directement aux enfants de cette entité sont également supprimés. Par contre, les entités considérées comme des enfants ne sont pas éliminées puisqu'elles peuvent se retrouver à un autre emplacement dans l'arborescence.

3.4.1.2 Graphe sémantique

La structure de l'organisation peut aussi être représentée par un graphe sémantique (figure 3.6). Les liens entre chaque entité sont des relations de composition. Les différentes fonctionnalités offertes à l'usager sont les mêmes que celles associées au graphe sémantique des connaissances. La différence majeure avec l'arborescence précédente est apparente lors de la suppression d'une entité. Cette entité est éliminée de façon définitive, car elle n'est représentée qu'une seule fois dans le graphe d'entités.

Encore une fois, nous avons laissé de côté diverses petites fonctions, en nous focalisant sur les interactions générales et indispensables au bon fonctionnement de notre système.

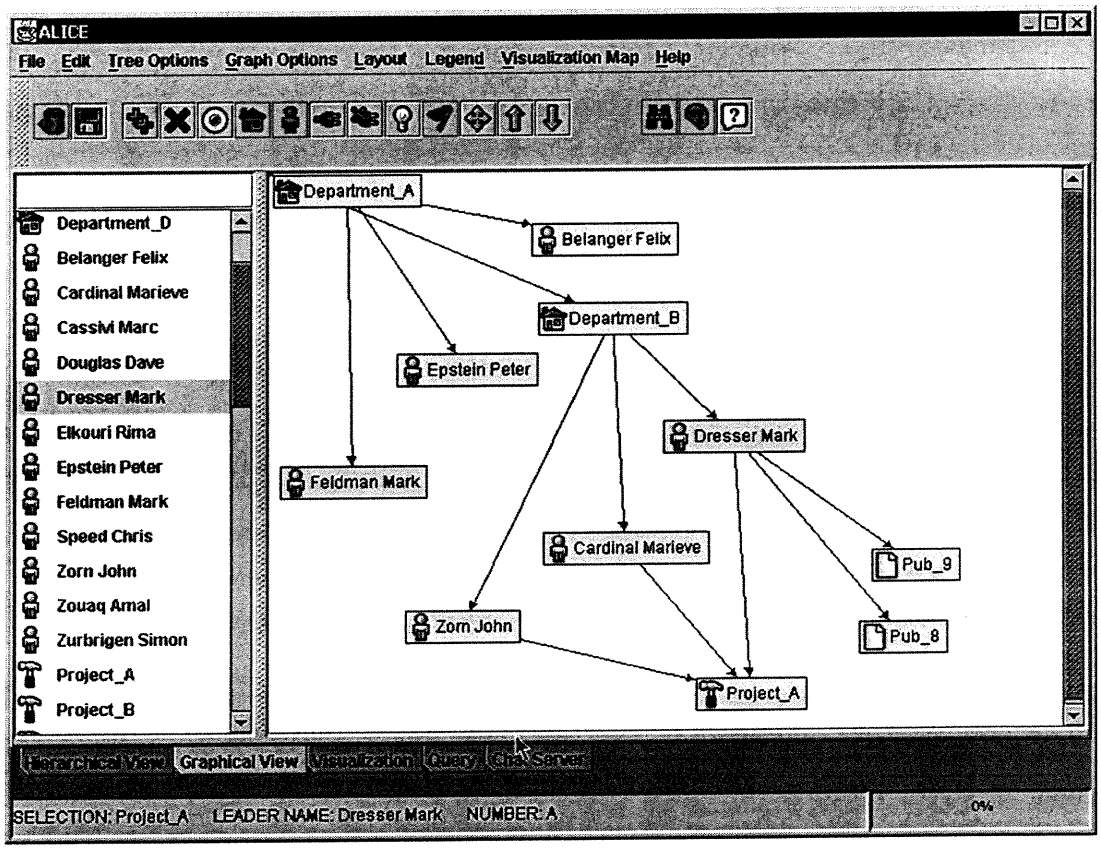

FIG. 3.6 – Structure de l'organisation sous la forme d'un graphe sémantique

3.5 Base des profils des entités

Le lien qui unit la structure de l'organisation au graphe des connaissances est représenté par les profils d'entités. Ainsi, chaque entité est associée à un profil. Ces profils sont en quelque sorte les empreintes digitales des entités. Ils sont composés de x valeurs numériques, x étant le nombre de connaissances représentées dans l'ontologie,

oscillant entre 0 et 1. Deux types de profils ont été élaborés, le profil de compétences et le profil d'intérêts. Ces profils peuvent être visualisés dans la figure 3.4 pour l'entité sélectionnée.

3.5.1 Profil de compétences

Le profil de compétences tente de définir les capacités, les compétences d'une entité en se basant sur les connaissances présentes dans l'ontologie. Par exemple, un individu qui est l'auteur d'un document sur les techniques orientées-objet est supposé être compétent dans ce domaine. Son profil de compétence devra donc refléter cet aspect, qui sera d'ailleurs découvert lors de l'analyse du document en question. Nous verrons dans la section 3.5.4 comment ce profil est construit.

Les publications sont les entités qui donnent le plus d'informations sur la compétence des individus. En effet, ces publications sont analysées dans le but d'en extraire le contenu sous forme de connaissances. La procédure d'analyse sera présentée à la section 3.5.4. Nous pouvons donc conclure que l'entité porteuse d'information est la publication, qu'elle soit reliée directement à un individu ou un projet. Enfin, contrairement au profil d'intérêts, chaque entité de l'organisation possède un profil de compétence.

3.5.2 Profil d'intérêts

Ce profil, comme son nom l'indique, tente de représenter les intérêts de chaque entité. Ces intérêts sont en grande partie associés directement aux individus ou aux projets. Les profils d'intérêts des individus sont essentiellement créés et mis à jour par l'analyse de la discussion entre individus. Cette analyse n'est pas faite par *Alice* mais plutôt par le système *WhiteRabbit* [44] brièvement décrit à la section 3.9. Par contre, en ce qui concerne les projets, les profils d'intérêts sont basés sur l'analyse des descriptions de ces derniers. Une description de projet comporte les objectifs visés et

les approches qui seront utilisées pour y arriver. Ces objectifs et approches sont en fait les intérêts du projet. Les compétences reliées au projet seront découvertes lors de publications portant sur les résultats associés aux approches et techniques utilisées. Les publications ne peuvent avoir un profil d'intérêts associé. Elles possèdent seulement un profil de compétences.

3.5.2.1 Analyse du courrier électronique

Le courrier électronique représente une source d'information sur les intérêts des usagers d'une même organisation. La destination de ces courriers peut aussi bien être à l'interne qu'à l'externe. Afin d'analyser cette source d'information, nous avons développé un module qui, de façon transparente, intercepte tous les courriers d'un usager de l'organisation. Il en analyse le contenu dans le but d'extraire les mots clés (connaissances) présents dans le graphe de connaissances de l'organisation. La procédure d'analyse est la suivante : pour chaque message, il y a détermination de l'usager associé au message, élimination de toute partie provenant d'un autre message (*reply*) et finalement extraction des connaissances présentes au sein du message. Cette information est ensuite transmise au système *Alice* pour que ce dernier mette à jour le profil d'intérêts de l'usager ou des usagers en question.

Ce module est externe au système *Alice*, il fonctionne comme processus indépendant sur la machine où le serveur du courrier électronique est installé. La communication entre ce processus et *Alice* s'effectue par le réseau (communication port à port). Il possède une certaine autonomie, c'est-à-dire qu'il n'est pas dépendant du système *Alice*. Si ce dernier n'est pas en marche, le processus emmagasinera l'information pour la transmettre au moment où le système *Alice* sera de nouveau en fonction.

Malheureusement, pour des raisons de confidentialité et d'éthique professionnelle, ce module n'a pas été intégré et utilisé pour la construction des profils d'intérêts dans la présente version d'*Alice*.

3.5.3 Dépendances entre profils

La méthode de détermination des profils est basée sur la structure arborescente de l'organisation. Cette méthode est la même que ce soit pour un profil de compétences ou d'intérêts. La seule différence à noter est la source d'analyse pour chacun des profils.

Le profil de compétence est initialement construit à partit d'une analyse de l'entité publication. Une fois le profil créé, il devient une source de calcul pour un profil d'une entité ayant un lien avec cette publication. Ainsi, l'individu qui est l'auteur de trois publications aura comme profil de compétences une moyenne de ces trois profils. La publication de base pour chaque individu est un curriculum vitae obligatoire lors de l'enregistrement de ce dernier. Donc, tous les départements sont assurés de posséder un profil de compétences du fait qu'un département est obligatoirement composé d'un individu au minimum. Cependant, un projet peut ne pas être associé à un profil de compétences si aucune publication concernant ce projet n'existe. D'ailleurs, une publication y d'un individu participant au projet x ne devient pas nécessairement une publication de ce projet. Cette association pourrait être une source d'erreurs puisqu'elle suppose que tous les individus du projet x possèdent les compétences comprises dans la publication y, ce qui peut ne pas être le cas.

Dans le cas du profil d'intérêts, les seules différences à noter sont les sources d'information permettant l'analyse. Ainsi, un projet utilise sa description incluant les objets et approches visés comme source d'analyse, tandis que les individus se hissent sur le système *White Rabbit* qui permet d'analyser leurs discussions.

3.5.4 Construction des profils

La construction des profils repose sur les connaissances qui constituent le domaine d'application. Ce domaine est d'ailleurs représenté par un graphe de connaissances

où les relations entre les connaissances sont représentées par différents liens. Ces liens apportent une information, que nous pouvons qualifier de cachée, qui sera utilisée dans la construction des profils d'usagers.

La construction d'un profil débute par l'ajout d'une publication à un usager, cette étape est illustrée à la figure 3.7. La publication est ensuite analysée par un processus d'extraction qui recherche les occurrences de connaissances. Pour chaque occurrence d'une connaissance, le profil de l'usager est mis à jour.

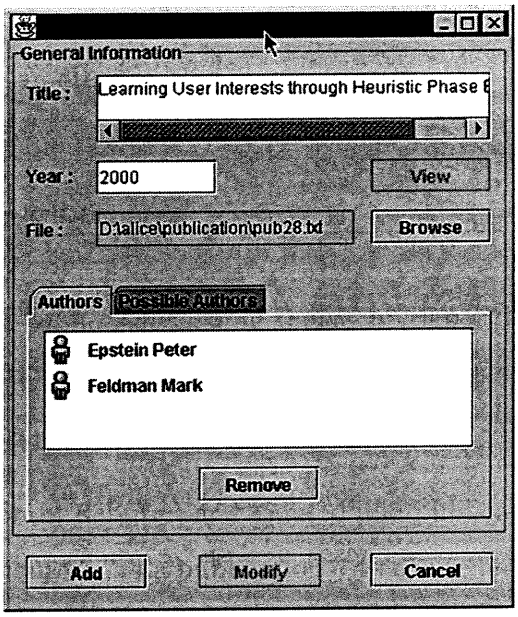

FIG. 3.7 – Ajout d'une nouvelle publication

Ce profil est représenté sous la forme d'un vecteur

$$\overrightarrow{p_j} = (f(x_{1j}), f(x_{2j}), \ldots, f(x_{ij}))$$

où $f(x_{ij})$ est la valeur associée à la i^{eme} connaissance qui compose le profil de la publication j. La valeur de $f(x_{ij})$ se situe entre 0 et 1. Afin que les valeurs du profil soient toujours entre les bornes (0,1), nous utilisons une fonction appelée sigmoïde (figure 3.8), qui est de la forme

$$f(x_{ij}) = \frac{1}{1 + e^{\lambda(x_{ij} - \beta)}} \tag{3.8}$$

La constante λ représente le taux de variation de cette fonction. Donc plus λ est élevé, plus la fonction tendra vers 1 rapidement. Le paramètre β a comme utilité de déplacer la fonction selon l'abscisse. Lorsque $x = \beta$ la valeur de $f(x) = 0.5$, ce qui signifie qu'une valeur appropriée pour la constante β serait la moyenne des valeurs x pour tous les profils, donc

$$\beta = \frac{\sum_{i=0}^{m} \sum_{j=0}^{n} x_{ij}}{mn} \tag{3.9}$$

où m est le nombre de profils existants et n le nombre de connaissances dans le graphe. Cette valeur permet d'avoir des profils équilibrés.

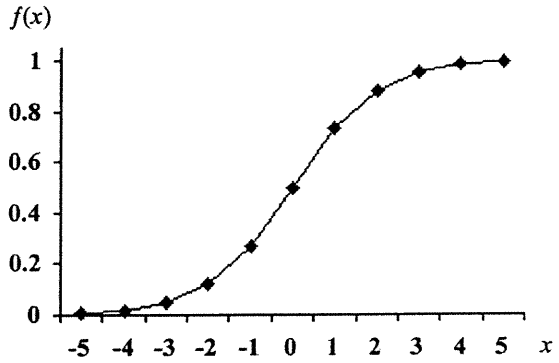

FIG. 3.8 – Représentation graphique de la fonction sigmoïde

Pour chaque occurrence de la connaissance i dans une publication j, la valeur $f(x_{ij})$ du profil correspondant est modifiée à la hausse. Afin d'arriver à la nouvelle valeur $f(x_{ij})$, la procédure suivante doit être suivie :

- Calculer la valeur x_{ij} précédente qui est à l'origine de la valeur $f(x_{ij})$ actuelle. Ceci consiste à calculer l'inverse de la sigmoïde. Cette fonction est décrite par l'équation 3.10.

- Augmenter x_{ij} d'une valeur k (par défaut $k = 1.0$). Cette nouvelle valeur est égale à $x_{ij}^{'}$.
- Transformer la valeur $x_{ij}^{'}$ par la fonction sigmoïde pour retrouver une valeur entre 0 et 1 $(f(x_{ij})^{'})$ qui remplacera l'ancienne valeur $f(x_{ij})$

$$x = -\frac{\ln(\frac{1}{f(x)} - 1)}{\lambda} + \beta \tag{3.10}$$

L'opération d'augmentation du poids de la connaissance sera ensuite répétée pour toutes les connaissances reliées directement ou indirectement à la connaissance de départ. Nous appelons, niveau, la distance qui sépare la connaissance de départ des connaissances qui lui sont reliées. Ce point est illustré à la figure 3.9. L'augmentation de chacune de ces connaissances est en fonction du type de lien. La valeur qui sera ajoutée à x_{ij} est définie par la variable k. Pour chaque type de lien est associé un k différent, les valeurs (k) par défaut sont :

- Pour le lien de similarité, $k = \frac{\nu}{3.0}(1 \leqslant \nu \leqslant 3)$ où ν est la valeur du degré de similarité associée au lien de similarité (*faible* $= 1$, *modéré* $= 2$, *élevé* $= 3$). Donc, plus ν est élevé plus k tend vers 1 et plus le poids de la connaissance similaire est augmenté.
- Pour le lien de généralisation $k = 2.0$.
- Pour le lien d'association $k = 1.0$.
- Pour le lien de spécialisation $k = 0.5$.

Chaque valeur de k tente de représenter la sémantique derrière ces liens. Le poids d'une connaissance reliée par un lien de généralisation est davantage augmenté que celui d'une connaissance reliée par un lien de spécialisation. En effet, un individu à davantage de chance de connaître la connaissance plus générale que celle plus spécifique. Les individus font plus facilement des associations (lien d'association) entre connaissances que des spécialisations. À l'inverse, ils ont plus de facilité à faire des généralisations que des associations, d'où la valeur de k pour chaque type de lien.

Au fur et à mesure que l'on s'éloigne de la connaissance de départ, une constante $0 \leqslant \varphi < 1$ (par défaut $\varphi = 0.8$ initialement) est multipliée à la valeur k. En effet, lorsqu'une entité possède une connaissance A, il est possible que les connaissances reliées à cette connaissance A soit également connues de cette entité. Toutefois, plus on s'éloigne de la connaissance de départ, plus cette probabilité diminue 3.9. Ainsi, pour chaque niveau, la valeur ajoutée au poids de la connaissance est toujours plus petit puisque $\varphi < 1$ (figure 3.9).

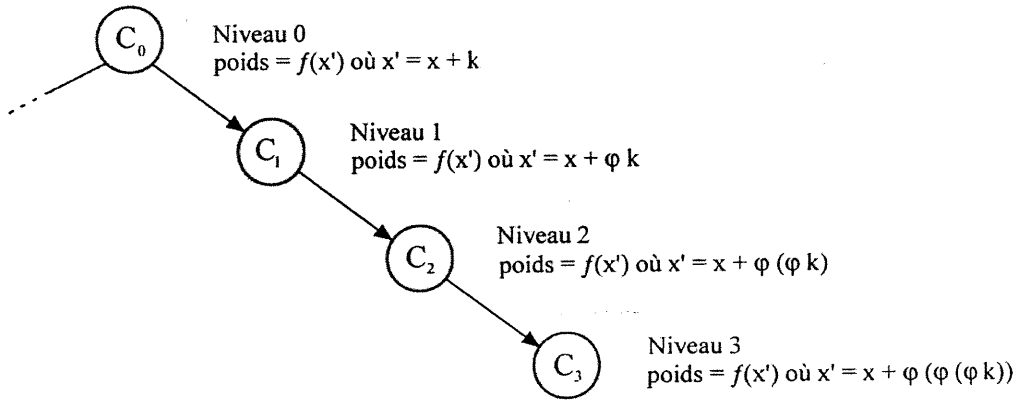

FIG. 3.9 – Représentation des différents niveau dans le graphe de connaissances

Notons que dans le cas où une publication est éliminée du système *Alice*, toutes les occurrences des connaissances extraites de cette dernière sont éliminées des profils. Il y a donc une mise à jour de tous les profils ayant une référence vers cette publication.

3.6 Catégorisation des profils

Le module de catégorisation repose sur le réseau de neurones décrit à la section 2.3.4, soit les cartes auto-organisées de Kohonen. Nous cherchons à catégoriser en n classes les différents profils présents dans le système. Rappelons que ces profils sont,

en quelque sorte, l'image des connaissances de chaque entité du système. La valeur de discrimination décrite à la section 3.3.3.2 sera introduite dans cet algorithme afin d'améliorer la catégorisation produite. Nous présenterons tout d'abord les différents paramètres utilisés dans l'initialisation et la phase d'apprentissage du réseau de neurones pour ensuite nous attarder sur l'ajout de la valeur de discrimination au réseau de neurones de base.

3.6.1 Initialisation du réseau de neurones

Le choix des paramètres associés aux cartes auto-organisées de Kohonen est déterminant pour la qualité des résultats. À vrai dire, plus la taille du réseau de neurones est importante, plus ces paramètres deviennent cruciaux [26].

La dimension de la couche de neurones en sortie détermine le nombre de catégories parmi lesquelles les entités seront distribuées. Cette couche est composée initialement de 9 neurones, donc 9 catégories. Ce nombre peut être modifié en tout temps. Il y aura alors une réorganisation des données basée sur la nouvelle dimension de la couche de neurones, qui devra repasser par une nouvelle phase d'apprentissage. Cet apprentissage du réseau de neurones repose sur plusieurs paramètres.

Débutons avec le paramètre $\sigma(t)$ qui représente le rayon du voisinage considéré lors de cet apprentissage. Dans le cas ou l'on choisit une valeur trop petite pour $\sigma(t)$, la couche en sortie de neurones ne sera pas ordonnée de façon globale. Un $\sigma(t)$ associé à une valeur trop grande nuira à une spécialisation accrue des neurones, nous aurons donc des catégories plus ou moins vagues et donc peu intéressantes. Une valeur initiale adéquate pour $\sigma(t)$ est représenté par environ la moitié du diamètre de la couche de neurones. Lorsque $\alpha(t)$ (facteur d'apprentissage) est assez grand, $\sigma(t)$ peut diminuer selon une fonction linéaire. Cependant, vers la fin de l'apprentissage, lorsque $\alpha(t)$ tend vers 0, nous pouvons restreindre le $\sigma(t)$ aux neurones voisins directs afin de peaufiner la spécialisation de la carte de neurones. La valeur initiale de $\sigma(t)$ ($radiusI$) est de 0.9

fois le rayon initial de la couche de neurones. La fonction linéaire qui fait décroître sa valeur lors de l'apprentissage est

$$\sigma(t) = radiusI - \frac{t * radiusI}{cycle}; \tag{3.11}$$

où un cycle est la variable représentant le nombre de cycles qui doivent être accomplis lors de l'apprentissage.

Le paramètre $\alpha(t)$ est lié au taux d'apprentissage tout au long de la phase d'entraînement. Sa valeur initiale est proche de 1. Nous retrouvons dans la littérature [12, 13, 22, 26] la fonction

$$\alpha(t) = 0.9 * 1 - \frac{t}{1000} \tag{3.12}$$

associée à l'évolution de $\alpha(t)$. Cette fonction implique donc une valeur initiale de 0.9 à $t = 0$. Lorsque cette valeur tend vers zéro, il est important d'arrêter sa décroissance afin de permettre une spécialisation dans un petit rayon. Cette valeur d'arrêt se trouve, dans notre cas, égale à 0.2.

3.6.2 Introduction de la valeur de discrimination

La valeur de discrimination a pour but d'identifier les connaissances les plus représentatives d'un domaine d'application. Elle est, comme décrit à la section 3.3.3.2, utilisée dans la présentation de nouvelles connaissances potentielles. Nous utiliserons également cette valeur dans la catégorisation des profils d'entités. En effet, l'algorithme de Kohonen considère chaque valeur du vecteur en entrée comme ayant la même importance. Ce qui veut dire qu'un terme avec un degré de discrimination élevé sera traité de la même façon qu'un autre avec un faible degré de discrimination. Afin d'expliquer comment prendre en considération le degré de discrimination, nous allons résumer comment l'algorithme de Kohonen fait la sélection du neurone gagnant associé au vecteur en entrée.

L'apprentissage du réseau de neurones consiste à déterminer, pour chaque catégorie, un vecteur modèle. Pour chaque cycle de cet apprentissage, un vecteur en entrée représente la couche supérieure du réseau de neurones. Chaque composante du vecteur en entrée est connectée à toutes les neurones qui constituent la couche de sortie. Le neurone gagnant est celui dont la somme des liens entre lui et le vecteur en entrée est minimale. Ce calcul de similarité est équivalent à la distance euclidienne (équation 2.3). Rappelons que les composantes du vecteur en entrée sont les poids de chaque entité du graphe de connaissances. Donc, afin de prendre en compte le degré de discrimination, il faut donner une importance plus grande aux liens faisant référence aux poids associés aux connaissances ayant un degré de discrimination élevé. Ce facteur d'importance pourrait être la valeur de discrimination elle même. Dans ce cas, la valeur de chaque lien serait multipliée par le degré de discrimination de la connaissance associée au poids en entrée.

Prenons par exemple le vecteur en entrée $x = (x_1, x_2, \ldots, x_n)$, le vecteur des valeurs de discrimination $d = (d_1, d_2, \ldots, d_n)$ où $d_i (1 \leq i \leq n)$ est la valeur de discrimination de la connaissance associée au poids x_i, et le vecteur modèle d'un neurone $v = (v_1, v_2, \ldots, v_n)$. La nouvelle distance, qui reflète la similarité, entre le vecteur x et le vecteur v est :

$$distance = \sqrt{d_1(x_1 - v_1)^2 + d_2(x_2 - v_2)^2 + \ldots + d_n(x_n - v_n)^2} \tag{3.13}$$

La valeur de discrimination d_i peut être négative comme elle peut être positive. Une valeur négative ou égale à 0 n'a pas lieu d'être car elle élimine complètement l'effet de la connaissance associée à cette valeur. C'est pourquoi nous avons choisi d'utiliser une fonction qui transforme la valeur d_i en une valeur positive. Cette fonction représentée à la figure 3.10 est

$$\phi_i = f(d_i) = e^{\lambda d_i} \tag{3.14}$$

où λ représente le facteur de progression vers l'infini. Plus ce facteur est grand, plus la pente de la courbe augmente de façon drastique. Ainsi, la modification du facteur λ modifie l'importance du degré de discrimination. Un faible λ implique qu'il y aura une légère augmentation ou diminution de ϕ lorsque le taux de discrimination changera d'une connaissance à une autre.

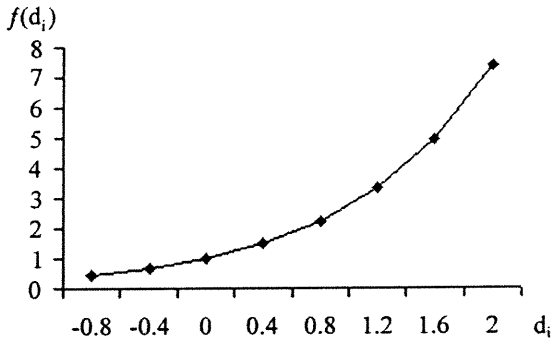

FIG. 3.10 – Représentation graphique de la fonction exponentielle (3.14)

Nous pouvons démontrer cette dernière affirmation en calculant la dérivée seconde de la fonction 3.14. La dérivée seconde représente la fonction que suit l'augmentation de la pente de la courbe à la figure 3.10.

$$\phi_i' = f(d_i)' = d_i e^{\lambda d_i}$$

$$\phi_i'' = f(d_i)'' = e^{\lambda d_i} + d_i^2 e^{\lambda d_i} = (1 + d_i^2) e^{\lambda d_i} \tag{3.15}$$

Plus la valeur du λ est élevée, plus la différence du degré de discrimination entre deux termes sera importante (lorsque $\lambda \longrightarrow \infty, \phi_i'' \longrightarrow \infty$) . Enfin, la fonction utilisée pour calculer la distance entre deux vecteurs prenant en considération le degré de discrimination est

$$distance = \sqrt{\phi_1(x_1 - v_1)^2 + \phi_2(x_2 - v_2)^2 + \ldots + \phi_n(x_n - v_n)^2} \tag{3.16}$$

3.7 Visualisation des données

Comme nous l'avons précisé à la section 2.3.4, l'algorithme de catégorisation utilise une forme de régression non-linéaire lors de son apprentissage. Cette régression est effectuée sur une base de vecteurs de références qui, à la fin de l'apprentissage, sont ordonnés. On peut donc imaginer que la couche de neurones associée aux vecteurs de références forme un réseau élastique à deux dimensions. La visualisation de cette couche de sortie facilitera l'analyse de la catégorisation, et par le fait même, l'analyse de la distribution des données.

3.7.1 Similarité entre catégories

L'un des principes fondamentaux des cartes auto-organisées de Kohonen est la prise en considération du voisinage lors de l'apprentissage du réseau. Ceci implique que la distribution des données dans la couche de neurones n'est pas sans signification. Au contraire, cette distribution peut être une source d'information très utile. Une faiblesse liée aux algorithmes de catégorisation est représentée par le fait qu'une catégorie peut être vide (aucune entité associée à cette dernière). Toutefois, si un ou plusieurs autres neurones sont similaires, donc proches en terme de distance au neurone vide (catégorie vide), il est possible de découvrir des entités similaires à cette catégorie.

Chaque neurone possède un vecteur de références qui sera obligatoirement plus similaires aux neurones avoisinants qu'aux neurones à l'extérieur de ce voisinage. Ceci, est dû à la prise en compte du voisinage lors de l'apprentissage. Donc, un neurone voisin peut être très similaire en terme de vecteur de référence et donc potentiellement en mesure de fournir une information utile à l'usager ayant obtenu un neurone gagnant vide. Par contre, il n'est pas dit que tous les neurones voisins possèdent un haut taux de similarité. C'est pourquoi une visualisation graphique des neurones et de leur distance en terme de similarité peut-être très utile.

3.7.2 Visualisation des catégories

La visualisation des catégories est représentée par deux types d'hexagones. Les hexagones ayant dans leur centre un point noir représentent un neurone donc une catégorie. Chacune des paires de ces hexagones (avec point noir) est obligatoirement séparée par un hexagone sans point noir que nous nommerons hexagone de distance ou un *hexD*(figure 3.11). Un *hexD* indique le niveau de similarité entre deux neurones (deux catégories) par des variations de couleurs. Un *hexD* d'une teinte foncée signifie un écart important entre deux neurones, une teinte claire une forte similitude. La forme hexagonale a été choisie parce qu'elle élimine toute confusion dans l'identification des deux neurones qu'elle associe. En effet, un *hexD* ne peut être en contact qu'avec seulement deux neurones. Toutes les ambiguïtés sont donc éliminées.

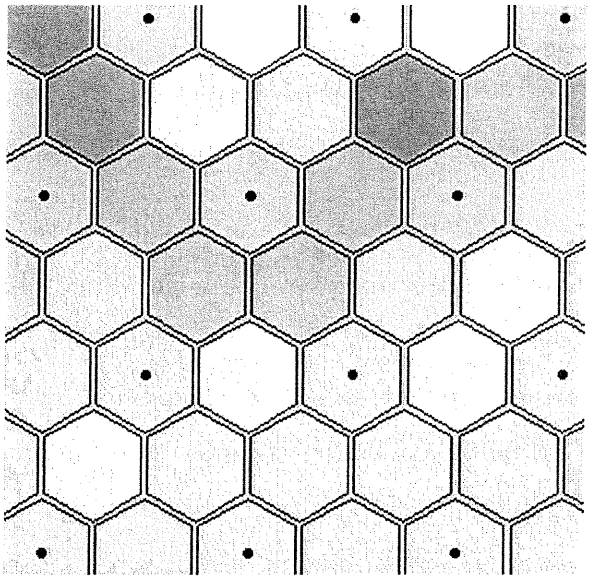

FIG. 3.11 – Structure hexagonale de la carte de visualisation

La coloration des *hexD* est basée sur une échelle de couleurs. Cette échelle est bornée par deux valeurs extrêmes, soit 0 pour deux neurones possédant des vecteurs de références identiques et $\sqrt{nombreConnaissancesDansVecteur}$ dans le cas opposé.

En effet, les valeurs des attributs d'un vecteur de référence se situent entre 0 et 1. De plus, le nombre d'attributs est égal au nombre de connaissances qui constituent l'ontologie. Donc, la distance euclidienne entre deux vecteurs complètement opposés soit $\overrightarrow{a} = (0, 0, \ldots, 0)$ et $\overrightarrow{b} = (1, 1, \ldots, 1)$, est égale à

$$\sqrt{\sum_{n=1}^{n}(a_n - b_n)^2} = \sqrt{\sum_{n=1}^{n}(0-1)^2} = \sqrt{n} \tag{3.17}$$

où n représente le nombre de connaissances.

Pour chaque entité, les profils de capacités ainsi que les profils d'intérêts s'il y a lieu, peuvent être visualisés sur une carte construite à base de neurones et de *hexD*. Un exemple de cette visualisation est présenté à la figure 3.12. Il est possible de préciser le type des entités et des profils que nous voulons analyser, ce qui entraîne une vue moins chargée. La dimension de la carte peut-être également modifiée par un zoom sur une section en particulier, ou une réduction de la carte permettant d'en avoir une vue globale si cette dernière est composée de plusieurs neurones. La dimension dépend du nombre de neurones qui forment la couche de sortie du réseau de neurones. Cette manipulation se fait par l'intermédiaire d'une barre de défilement (*sliderBar*) qui se trouve au bas de la figure 3.12.

La coloration de la carte de visualisation nous permet entre autres de faire une méta-catégorisation. C'est-à-dire qu'il est simple d'observer par les codes de couleurs les regroupement de profils ainsi que les séparations marquées entre deux segments de la carte. Ceci se résume à catégoriser les x neurones en n catégories, $n < x$ selon la distribution. De plus, la densité des entités par neurone nous permet de faire apparaître les forces et spécialités ou même les faiblesses d'une organisation.

Le détail de chaque neurone peut être obtenu par la sélection (double clic de la souris) de ce dernier. Une fenêtre apparaît alors (figure 3.13), qui contient l'image du vecteur de référence du neurone sous forme de profil . Chaque entité contenu dans ce neurone

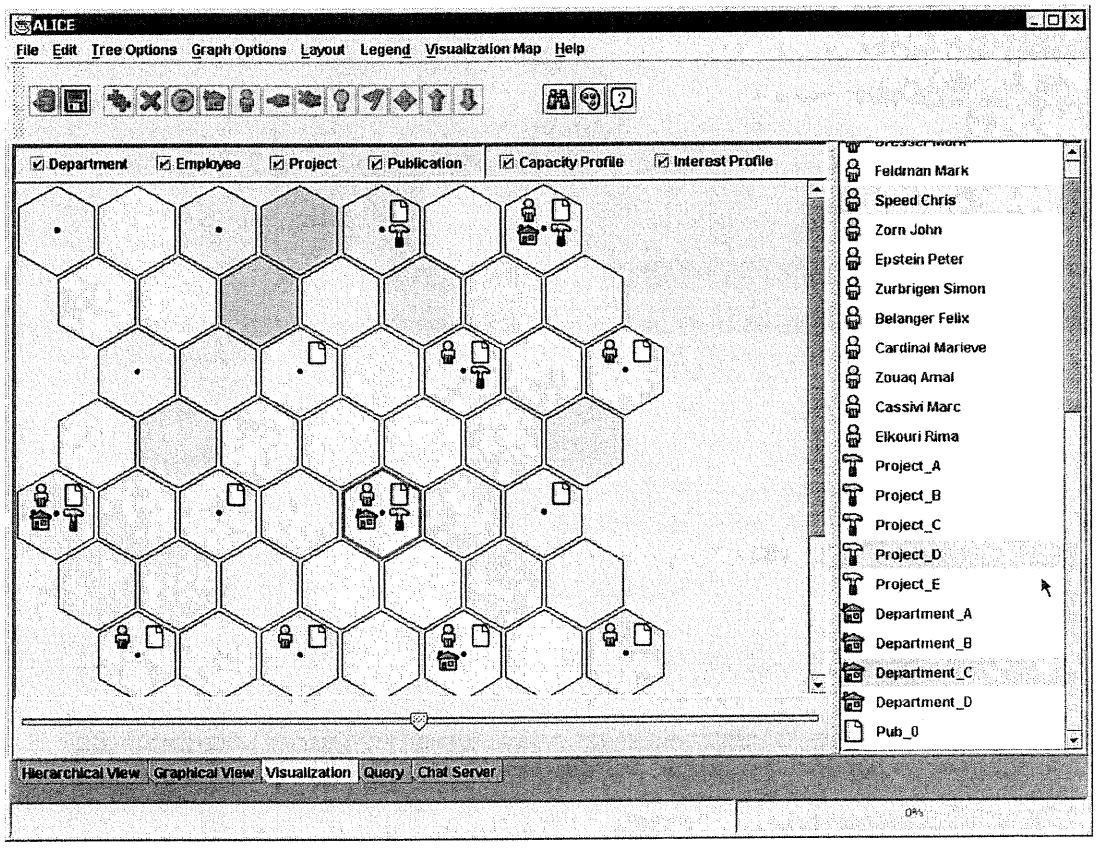

FIG. 3.12 – Carte de visualisation des entités de l'organisation

possède un profil similaire à ce vecteur de référence. De plus, en analysant, par le même procédé, le vecteur de référence d'un neurone voisin, nous pouvons en déduire les caractéristiques communes ou différents.

Pour avoir plus d'information sur la distribution des entités présentes sur la carte de visualisation (figure 3.12) deux possibilités sont offertes à l'usager :

- Faire la sélection d'un neurone (hexagone avec point noir au centre) avec la souris. Les entités qui y sont catégorisées seront alors sélectionnées dans la liste à gauche de la carte.
- À l'inverse, si on sélectionne une entité dans la liste de gauche, le neurone qui contient cette entité sera encadré d'un trait rouge.

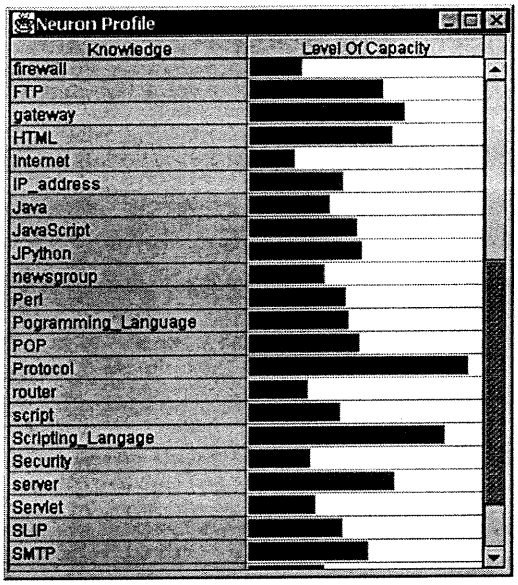

FIG. 3.13 – Profil d'un neurone

Ces deux fonctions permettent donc de faire une analyse plus approfondie de la distribution des entités.

3.8 Requêtes

Une fois l'ensemble de connaissances emmagasiné, structuré et analysé, il est important de permettre à l'usager de lancer des requêtes sur cette structure d'information. Cet usager peut ainsi, par une simple requête, être en mesure de se diriger vers une information potentiellement utile. Nous présenterons donc dans la prochaine section le fonctionnement du module de requête, qui tente de répondre aux objectifs mentionnés ci-dessus.

3.8.1 Construction d'une requête

Le module de requête est représenté par une interface, illustrée à la figure 3.14, qui permet à l'usager de construire une requête, de la soumettre au système, et d'accéder

aux résultats. Deux types de requêtes ayant chacun un but différent sont proposés à l'usager. Les deux buts visés sont :

- Rechercher des entités ayant des compétences (connaissances) précises. Ces requêtes retournent des entités complémentaires par rapport aux connaissances de l'usager.
- Rechercher des entités ayant des intérêts précis.

La construction d'une requête se fait en trois étapes :

- Sélectionner le ou les types d'entités (département, projet, individu, publication) désirés comme résultats.
- La seconde étape consiste à spécifier si la recherche doit se faire à partir des profils de compétences ou d'intérêts.
- La dernière étape est la sélection des connaissances qui formeront le profil associé à la requête.

Selon les connaissances choisies lors de la dernière étape, un profil est automatiquement construit et soumis au réseau de neurones. La construction du profil se fait de la même façon que lors de l'analyse d'une publication. Les liens du graphe de connaissances sont pris en considération. Suite à la catégorisation de ce profil, nous sommes en mesure de proposer des entités qui répondent au profil associé à la requête.

3.8.2 Analyse des résultats

Les résultats associés à la requête sont présentés dans la partie droite de l'interface (figure 3.14). On y retrouve les entités qui correspondent à la même catégorie que le profil ayant comme base les connaissances de la requête. Il est alors possible de sélectionner (par un double *clic* de la souris) une entité afin d'avoir de plus amples informations. Par exemple, l'usager peut accéder directement au contenu d'une publication ou même avoir des renseignements sur un individu en mesure de l'aider (exemple pour une publication, figure 3.7).

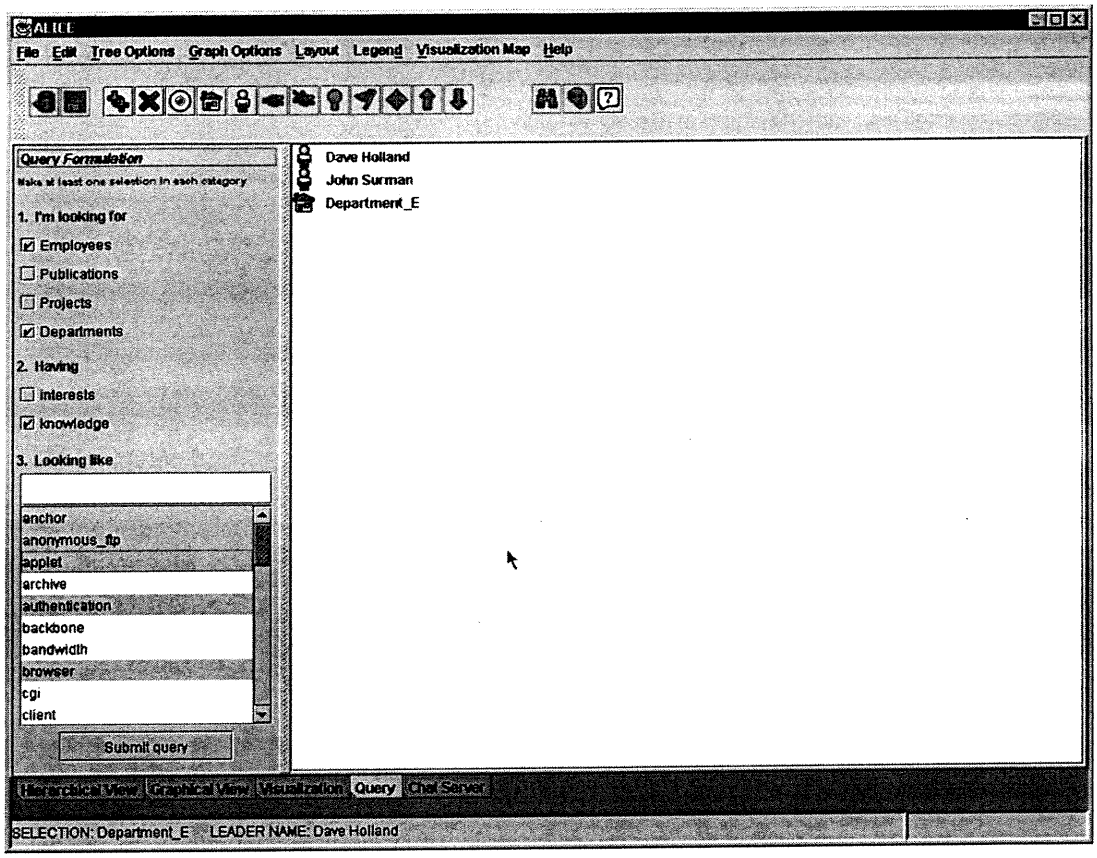

FIG. 3.14 – Interface de requête pour l'usager

Il est possible que la catégorie associée au profil de la requête soit vide. Dans ce cas, aucun résultat ne sera retourné. La visualisation des connaissances au sein de l'organisation entre alors en jeu . Comme nous l'avons décrit à la section 3.7, la couche de neurones en sortie, représentant les catégories, est ordonnée. Ceci implique que les neurones voisins du neurone contenant le profil de la requête peuvent peut-être aider l'usager dans sa recherche d'information. Ces neurones possèdent les vecteurs les plus similaires au profil de la requête, après, bien sûr, le neurone choisi initialement. Ces neurones (sous-ensemble de la couche de neurones) ainsi que leur contenu peuvent être analysés par le biais de la carte de visualisation illustrée à la figure 3.15.

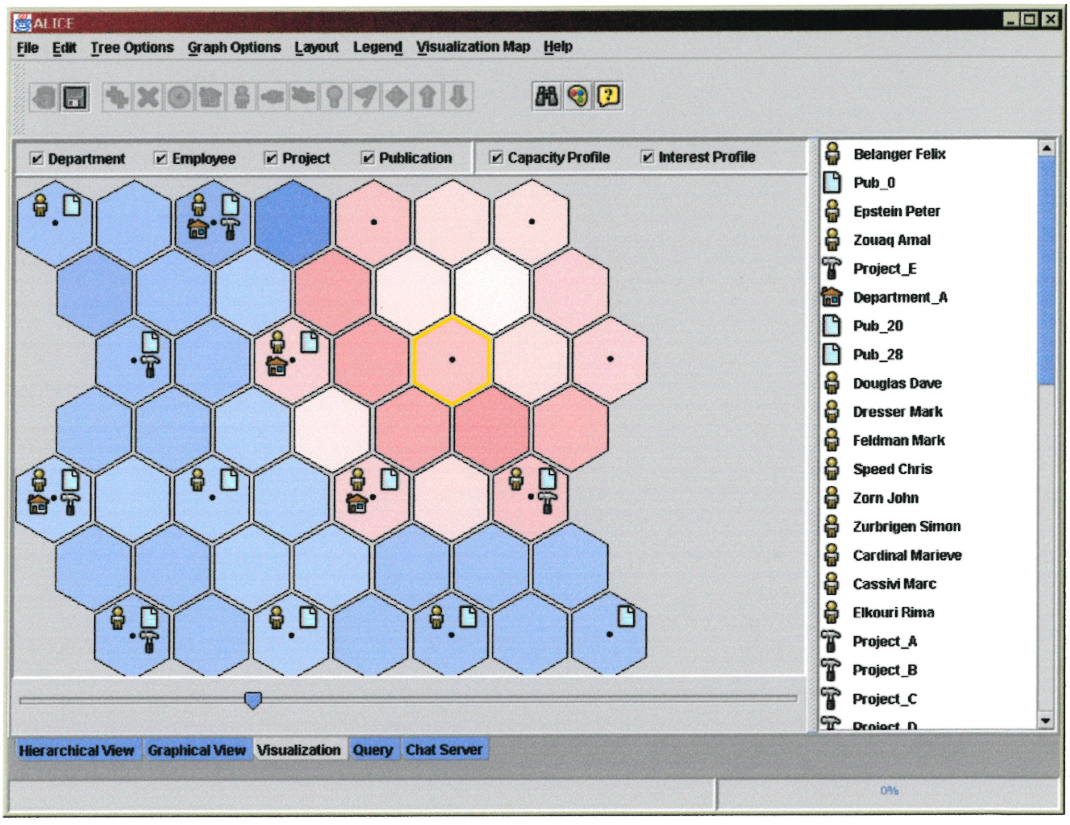

FIG. 3.15 – Visualisation du voisinage d'une requête

Ce sous-ensemble de neurones est représenté par une couleur rouge qui permet de bien le distinguer. Le neurone initialement sélectionné possède un cadre jaune. Ainsi, les entités à l'intérieur de ce sous-ensemble sont susceptibles d'aider l'usager. Les différents tons de couleurs peuvent donner un indice de la distance en terme de connaissances entre deux neurones. De la même manière il est donc possible de savoir quelles sont les entités les plus proches du neurone gagnant.

3.9 Système annexe - White Rabbit

Le système *White Rabbit* [43, 44] a pour objectif de maximiser la coopération au sein d'un groupe de personnes au moyen de l'analyse de la conversation entre ces personnes. Chaque usager est assisté par un agent intelligent ayant la capacité d'établir le profil de ses intérêts. Par la suite, grâce à ses propriétés d'autonomie et de mobilité, l'agent est en mesure de visiter ses confrères (individu de l'organisation) afin de mettre en contact les usagers possédant des intérêts communs. Un agent médiateur facilite la communication entre les agents personnels et effectue la catégorisation des différents profils apportés par ceux-ci. La conversation se tient dans un environnement de discussion en temps réel, adapté aux besoins du système. Le module de catégorisation et l'utilisation de graphe de connaissances sur lesquels sont basés les profils d'usagers se retrouvent dans notre système (*Alice*), on voit donc que ces deux systèmes sont complémentaires. D'ailleurs, ils sont devenus deux modules intégrales d'un système nommé *VAI Connection* créé au sein de l'entreprise Virtuel-Âge International.

Le système ayant été décrit, nous allons présenter son évaluation.

Chapitre 4

Évaluation et discussion

4.1 Introduction

Le système *Alice* pourrait être évalué par deux façons différentes. La première serait de l'intégrer dans une organisation où le nombre de ressources humaines serait important. De cette façon, nous pourrions observer si le système est en mesure d'apporter une aide aux employés à la recherche d'informations. De plus, les résultats obtenus pourraient être analysés par des experts du domaine d'application. Cependant, ce type d'expérimentation est difficilement accessible car il exige beaucoup de temps ainsi qu'un partenaire industriel. C'est pourquoi cette approche ne sera pas adoptée dans le présent travail.

La seconde approche, utilisée dans ce mémoire, est une analyse basée sur des données expérimentales fictives. L'évaluation portera principalement sur trois concepts qui sont à la base du fonctionnement du système *Alice*, soit :

- La catégorisation des différentes entités.
- L'influence du degré de discrimination associé aux connaissances.
- L'impact des différents liens que l'on retrouve dans le graphe des connaissances.

Des résultats satisfaisants devraient laisser présager une gestion adéquate des connaissances, par le système *Alice*, au sein d'une organisation. Avant de nous focaliser sur chaque concept individuellement, nous allons présenter, dans la section suivante, les données utilisées pour cette évaluation.

4.2 Base de données expérimentale et fictive

Les données utilisées pour l'évaluation sont divisées en deux parties, celle qui constitue le graphe de connaissances et celle qui est associée à la structure de l'organisation (soit les entités).

4.2.1 Graphe de connaissances

Le graphe de connaissances que nous avons construit traite du domaine de l'*Internet*, figure 4.1. On y retrouve donc un ensemble de mots spécialisés liés à *Internet*, ayant des relations particulières entre eux. Les différents types de liens définis par l'ontologie décrite à la section 3.3.1 sont au nombre de trois :

- Lien de similarité.
- Lien de généralisation et spécialisation.
- Lien d'association.

Ce graphe se compose de 42 connaissances, reliées au domaine de l'*Internet*, que nous avons nous-mêmes choisies. Ce nombre est assez important pour permettre de générer des profils d'une taille raisonnable (42 attributs). Il est évident que dans une situation réelle, ce graphe serait maintes fois plus important, toutefois, pour une évaluation, nous pensons que ce nombre est suffisant.

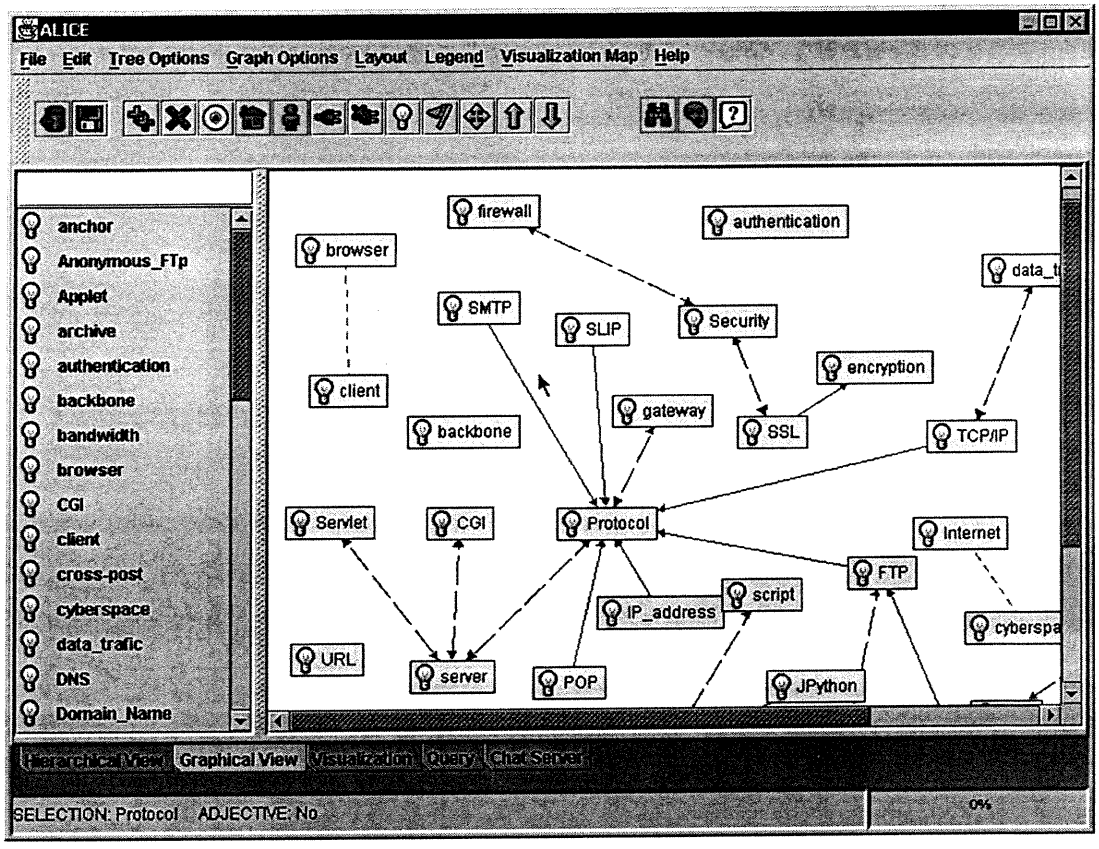

FIG. 4.1 – Graphe des connaissances associé au domaine de l'*Internet*

4.2.2 Structure de l'organisation

La structure de l'organisation (figure 4.2), des départements jusqu'aux publications, est fictive à tous les niveaux. La structure est composée de 4 départements, 5 projets, 12 individus et d'une banque de 29 publications associées aux différents individus.

Chacune des publications est générée de façon automatique. Pour ce faire, nous avons implanté un générateur de publication qui sélectionne, de façon aléatoire, un mot parmi une banque de mots pour ensuite l'ajouter à la fin d'un fichier texte. Cette banque de mots est composée des connaissances préalablement établies portant sur le domaine de l'*Internet* et de mots n'ayant aucun lien avec ce domaine. Ces derniers sont

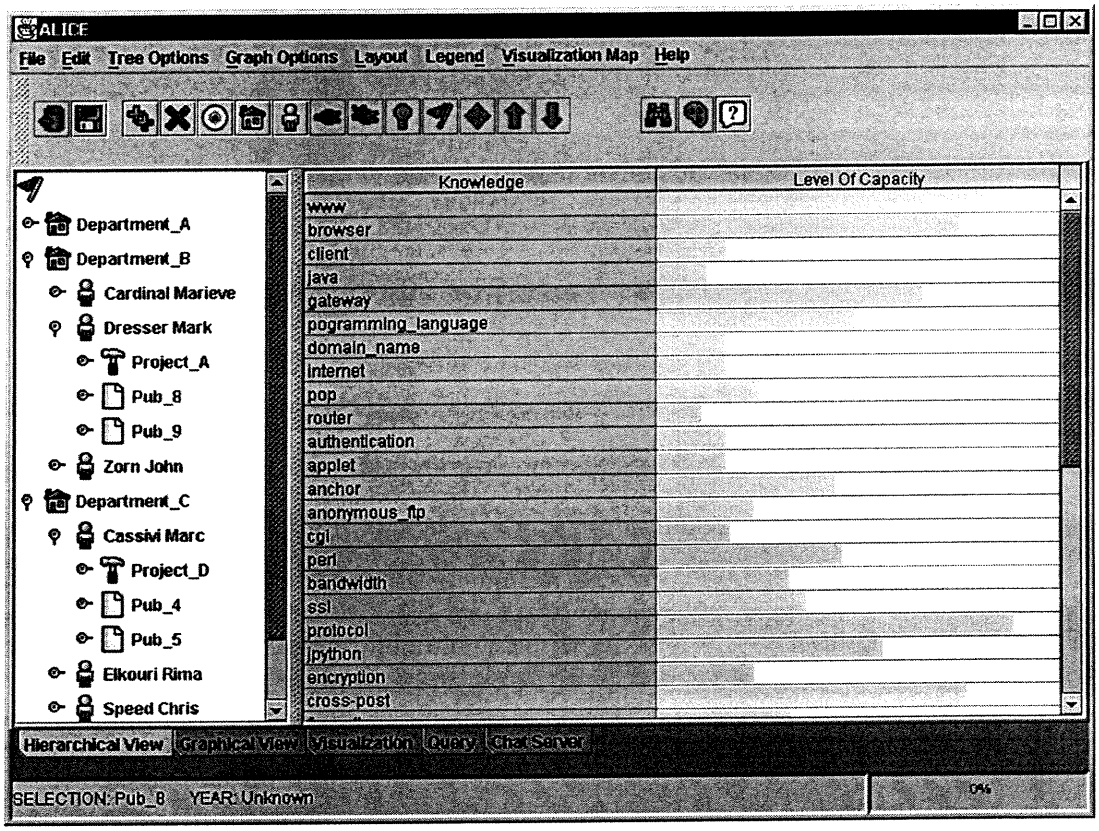

FIG. 4.2 – Structure fictive de l'organisation.

utilisés pour augmenter la diversification et l'hétérogénéité au sein des publications. Le nombre de mots au sein d'une publication est déterminé par une variable que nous définissons. De cette façon, chaque fichier texte créé correspond à une publication.

Cette génération implique que les publications n'ont pas de sens puisqu'elles sont construites de façon aléatoire à partir d'une base de mots où les connaissances du domaine d'application représentent un sous-ensemble. Ceci n'affecte en rien l'évaluation du système puisque même avec de vraies publications, cette version d'*Alice* ne tient pas compte de la sémantique linguistique. En effet, le système fait la recherche de mots précis et utilise des données statistiques.

4.3 Analyse de la catégorisation

Comme vu à la section 3.6, l'approche utilisée pour trouver les entités similaires est celle de la catégorisation. Cette catégorisation est effectuée par un réseau de neurones basé sur les cartes auto-organisées de Kohonen que nous avons implanté. Une manière de vérifier si la catégorisation est efficace est d'abord de calculer le degré de similarité entre une entité d'une catégorie et le vecteur modèle de cette même catégorie. Ce calcul du degré de similarité est répété sur chacune des entités comprises dans la même catégorie. Ensuite, les résultats obtenus sont compilés pour en extraire une moyenne qui représente un degré de similarité moyen des entités par rapport à la catégorie correspondante. Plus cette moyenne est élevée, plus nous avons la preuve que la catégorisation est performante. En effet, si une ou plusieurs entités sont catégorisées sous une mauvaise catégorie, leur degré de similarité avec cette dernière sera peu élevé et par conséquent la moyenne en souffrira.

4.3.1 Configuration du réseau de neurones

Il est important de préciser la configuration de la couche de sortie du réseau de neurones car ce paramètre est relié à l'interprétation des résultats. En effet, la couche en sortie est composée d'un nombre de neurones correspondant au nombre de catégories désirées. Donc, plus la dimension de cette couche de sortie sera grande plus la catégorisation sera spécialisée. Nous tenterons de démontrer cette hypothèse dans le processus d'évaluation de la catégorisation. Deux configurations seront testées, la première sera de 3x3 soit 9 neurones et la seconde de 5x5 ce qui équivaut à 25 neurones.

Par ailleurs, l'entraînement du réseau de neurones repose sur 10000 cycles d'apprentissage et sur un ensemble de 46 vecteurs en entrée. Le nombre de cycles lors de l'apprentissage a été déterminé selon des données statistiques préalablement établies [12, 26]. Le tableau qui suit (4.1) est composé des résultats de la catégorisation de

différentes entités selon une configuration de 3x3 (9 neurones, donc 9 catégories). Le graphique de la figure 4.3 qui suit est la représentation visuelle des donnés présentes dans ce tableau.

Catégorie/Neurone	Nb. Entités	Dist. Moyenne	% Similarité
1	0	-	-
2	0	-	-
3	6	0.76877949	88.13
4	7	0.76797678	88.14
5	5	0.75144458	88.40
6	7	0.76035504	88.26
7	6	0.83442840	87.12
8	13	0.73950487	88.58
9	6	0.95235408	85.30

TAB. 4.1 – Distance moyenne des entités d'une même catégorie (9 neurones)

L'analyse des résultats présents dans le tableau 4.1, visualisée par le graphique de la figure 4.3, montre que la catégorisation semble être efficace. La droite représentant la somme des moindres carrés montre un degré de similarité d'environ 88%. La distribution des entités (histogramme) par catégorie est assez uniforme même si les deux premières ne contiennent pas d'entités. Par ailleurs, le nombre d'entités par neurone ne semble pas influencer la qualité du degré de similarité. Nous allons maintenant reprendre le même test avec comme différence, une couche de neurones d'une dimension de 25 neurones (5x5). Les résultats de ce test sont représentés dans le tableau 4.2 et le graphique de la figure 4.4.

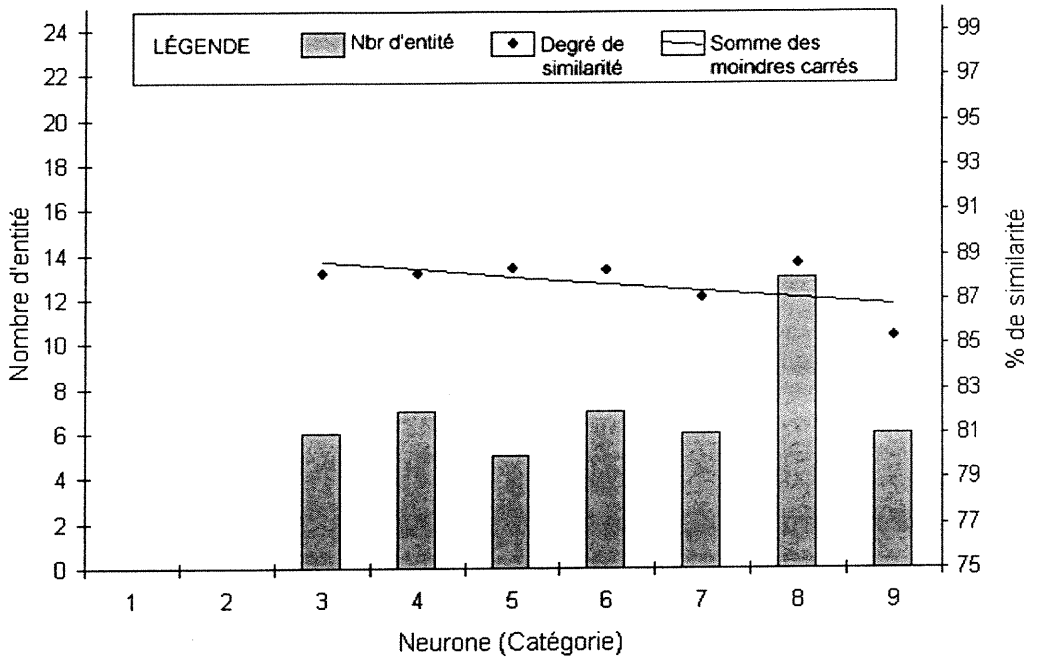

FIG. 4.3 – Graphique du nombre d'entités et du degré de similarité des entités en fonction des neurones (3x3).

Catégorie/Neurone	Nb. Entités	Dist. Moyenne	% Similarité
1	1	0.08355370740	98.71
2	1	0.10390400060	98.40
3	4	0.66378596894	89.75
4	2	0.46094927762	92.88
5	2	0.75891087212	88.28
6	3	0.89828907255	86.13
7	1	0.03087356985	99.52
8	3	0.59183241952	90.86

suite à la page suivante

suite de la page précédente			
Catégorie/Neurone	Nb. Entités	Dist. Moyenne	% Similarité
9	1	0.07380262261	98.86
10	2	0.50105135211	92.26
11	2	0.90636010665	86.01
12	2	0.23928496440	96.30
13	4	0.64470193621	90.05
14	2	0.23466801326	96.37
15	5	0.65380711943	89.91
16	2	0.00451926569	99.93
17	2	0.41610113461	93.57
18	2	0.00350318296	99.94
19	0	-	-
20	0	-	-
21	3	0.65184367497	89.94
22	4	0.49223070021	92.40
23	2	0.82632872959	87.24
24	0	-	-
25	0	-	-

TAB. 4.2 – Distance moyenne des entités d'une même catégorie (25 neurones)

Le graphique de la figure 4.4 vérifie l'hypothèse mentionnée ci-dessus, stipulant qu'une augmentation de la dimension de la couche de sortie, donc plus de catégories, apporte une spécialisation de la catégorisation. Il en résulte une distribution uniforme des entités et une hausse du degré de similarité moyen par rapport au vecteur modèle (environ 93%). Le graphique de la figure 4.5 fait la comparaison des deux droites (somme des moindres carrés) représentant la moyenne du degré de similarité entre les entités et le vecteur modèle associé à la catégorie prise dans les graphiques des

figures 4.3 et 4.4. On peut rapidement voir que la catégorisation est plus spécialisée avec la configuration de 25 neurones.

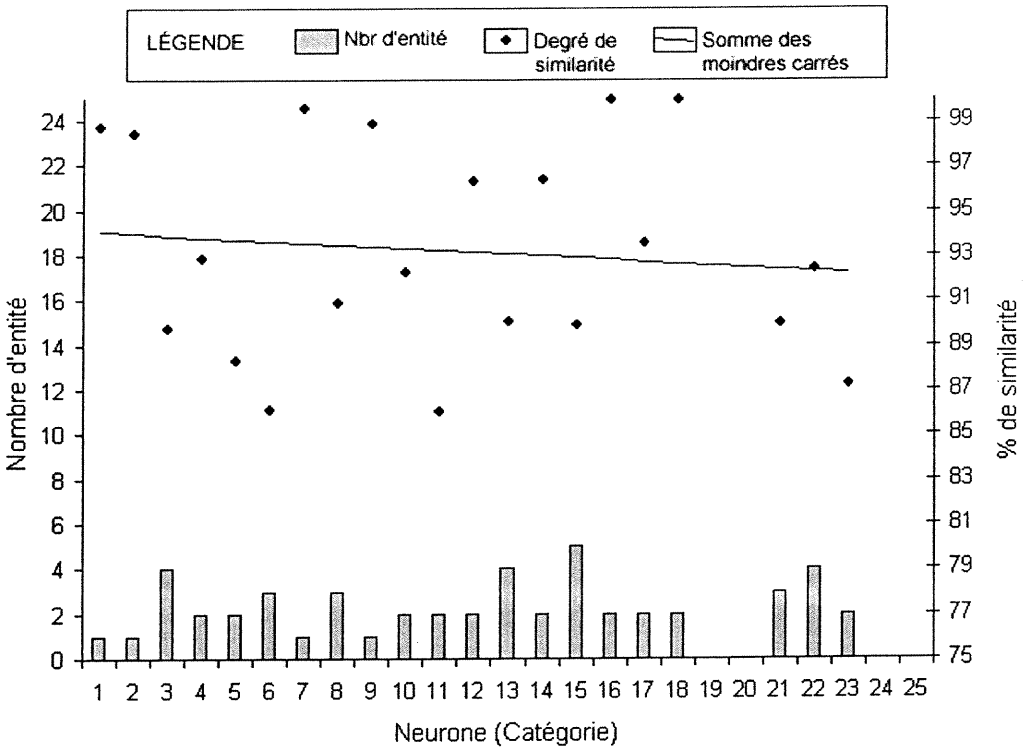

FIG. 4.4 – Graphique du nombre d'entités et du degré de similarité des entités en fonction des neurones (5x5).

Quelque soit le type de configuration, la moyenne du degré de similarité associée aux catégories est assez constante. Ceci montre que chaque entité est jumelée à un neurone qui la représente adéquatement. Ces résultats nous amènent à conclure que la catégorisation des entités s'effectue avec succès. Nous allons maintenant nous attarder sur le degré de discrimination appliqué à cette catégorisation.

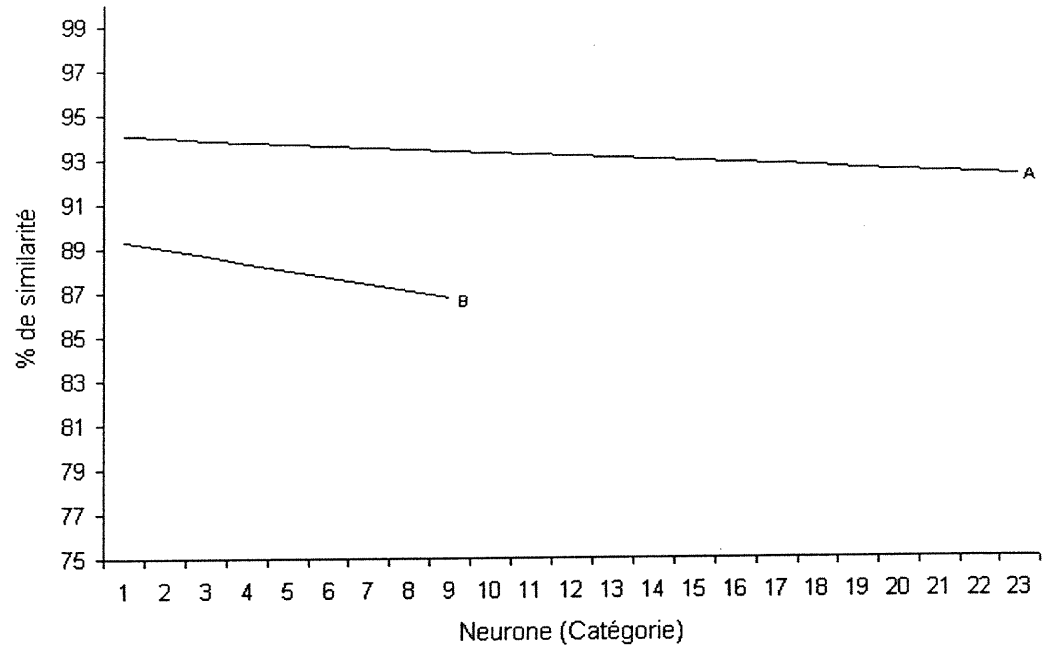

FIG. 4.5 – Graphique du degré de similarité moyen des entités en fonction des neurones. Courbe A : configuration 5x5, courbe B : configuration 3x3.

4.4 Degré de discrimination

Chaque connaissance est associée à un degré de discrimination qui est calculé lors de l'extraction des connaissances. Plus ce degré est élevé, plus cette connaissance est représentative du domaine d'application. Comme décrit à la section 3.6.2, la valeur du degré de discrimination peut être intégrée à l'algorithme de catégorisation. L'ajout de ce paramètre peut sensiblement augmenter la spécialisation de la catégorisation des différents vecteurs de connaissances. Le tableau 4.3 et le graphique de la figure 4.6 montrent les résultats de la catégorisation des mêmes données que dans la première configuration (3x3) (section 4.3), en ajoutant une valeur de discrimination.

Catégorie/Neurone	Nb. Entités	Dist. Moyenne	% Similarité
1	0	-	-
2	0	-	-
3	6	0.79458743	88.17
4	7	0.76239844	88.23
5	5	0.77349034	88.46
6	7	0.73983453	88.58
7	6	0.93432343	87.26
8	13	0.75434346	89.03
9	6	0.82343443	87.29

TAB. 4.3 – Distance moyenne des entités d'une même catégorie en tenant compte du degré de discrimination

Nous observons que les degrés de similarité sont supérieurs à ceux dans le tableau 4.1. Par ailleurs, le graphique de la figure 4.7 montre les courbes moyennes du degré de similarité selon que la valeur de discrimination est prise en considération (courbe A) ou non (courbe B). Il est facilement observable que la valeur de discrimination augmente légèrement la performance de la catégorisation. Cette observation peut être expliquée par la composition des profils de connaissances. En effet, la catégorisation sera légèrement meilleure si les poids dominants sont ceux ayant un fort degré de discrimination.

Ce degré de discrimination prend toute son importance lors d'une requête faite au système par l'usager. Rappelons qu'une requête, quelque soit son type, repose sur un vecteur de connaissances. Elle aura donc un rendement maximal si les poids les plus élevés du vecteur sont ceux dont les connaissances possèdent un degré de discrimination également élevé. Dans le cas contraire, il n'y aura pas d'effet inverse (effet négatif), toutefois le degré de discrimination aura peu ou même aucune influence sur la requête. Le tableau 4.4 fait la compilation de 6 requêtes différentes sur la même

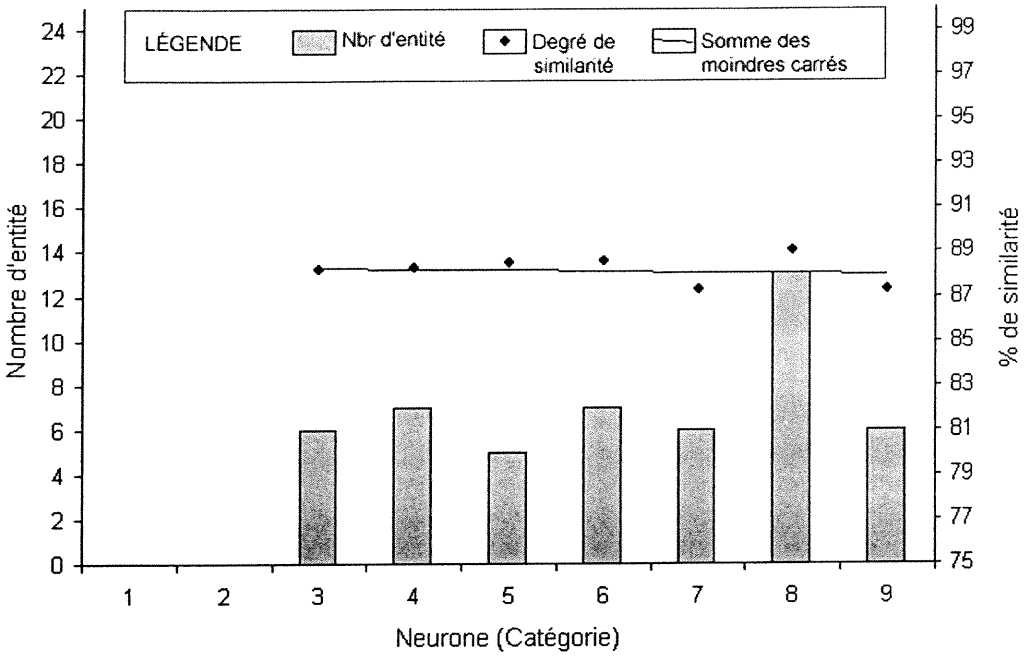

FIG. 4.6 – Graphique du nombre d'entités et du degré de similarité des entités en fonction des neurones (3x3), avec valeurs de discrimination.

base de données. Chaque requête est exécutée deux fois, avec et sans valeur de discrimination associée aux connaissances. Pour chacune de ces requêtes, la moyenne du degré de similarité entre les entités de la catégorie gagnante est calculée. Il est important de remarquer qu'ici, le degré de similarité n'est pas calculé entre les entités d'une catégorie et le vecteur modèle de cette dernière mais plutôt entre les différentes entités d'une même catégorie. De cette manière, nous pourrons vérifier si la valeur de discrimination permet d'associer à une requête des entités plus spécialisées. Selon cette hypothèse, la prise en considération de la valeur de discrimination devrait augmenter le degré moyen de similarité entre les mêmes entités d'une catégorie.

Notons que cette évaluation s'est faite en deux étapes. Nous avons, en premier lieu, lancé l'apprentissage du réseau de neurones sans tenir compte du degré de discrimination et nous avons effectué nos différentes requêtes. Ensuite, nous avons refait

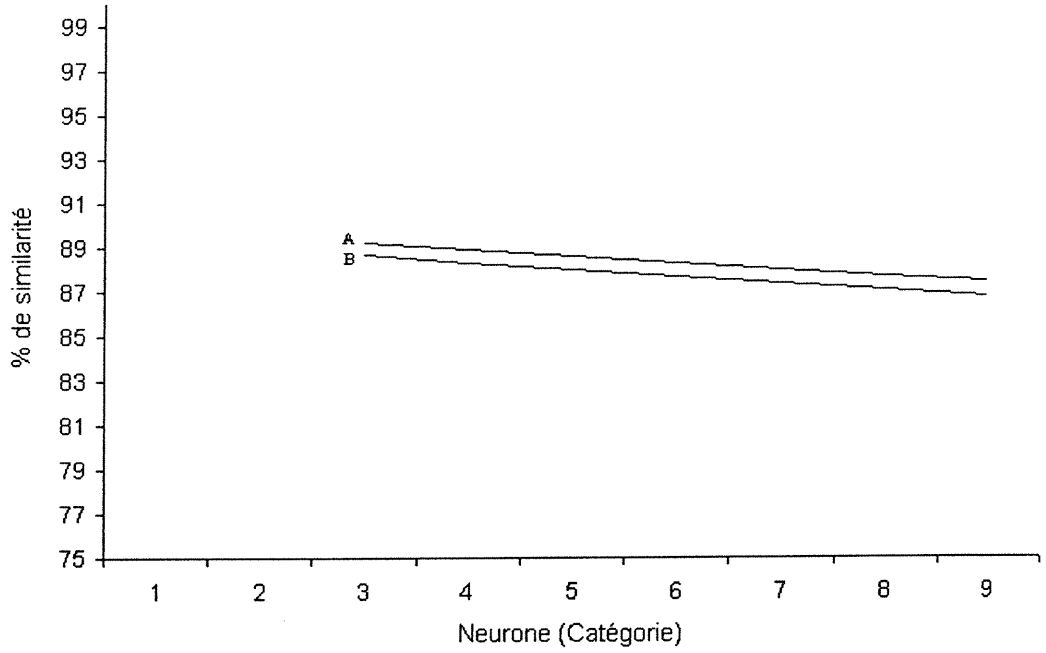

FIG. 4.7 – Graphique du degré de similarité moyen des entités en fonction des neurones(3x3). Courbe A : avec valeur de discrimination, courbe B : sans valeur de discrimination.

l'apprentissage du réseau de neurones, cette fois-ci en considérant le degré de discrimination puis nous avons exécuté les mêmes 6 requêtes. Le tableau 4.4 ainsi que le graphique de la figure 4.8 reflètent les résultats de cette expérimentation.

Num. Req.	Valeur Disc.	Catégorie	Dist. Moyenne	% Similarité
1	oui	2	0.69495673	89.27
1	non	3	0.81567564	87.41
2	oui	5	0.64463846	90.05
2	non	7	0.82677688	87.24

suite à la page suivante

suite de la page précédente				
Num. Req.	Valeur Disc.	Catégorie	Dist. Moyenne	% Similarité
3	oui	5	0.77349034	88.23
3	non	5	0.76956466	88.12
4	oui	4	0.80346374	88.30
4	non	4	0.76797678	88.14
5	oui	1	0.81435654	87.43
5	non	9	0.95235408	85.30
6	oui	9	0.61437343	90.51
6	non	8	0.78797678	87.84

TAB. 4.4 – Distance moyenne des entités retournées par une requête avec ou sans le degré de discrimination

L'analyse du graphe 4.8 démontre à partir des deux courbes moyennes A et B que le degré de similarité entre les entités d'une même catégorie est supérieur lorsque la valeur de discrimination est prise en considération. Dans tous les cas, les requêtes jumelées au degré de discrimination retournent des entités plus similaires au profil de la requête.

Par ailleurs, nous remarquons dans le tableau 4.4 que pour certaines requêtes, la catégorie retournée n'est pas la même selon que la valeur de discrimination est prise en considération ou non. Cependant, si l'on observe la figure 4.9, les catégories retournées sont dans tous les cas une catégorie voisine.

Cette dernière observation peut être expliquée par un exemple où la dimension des profils est réduite. Prenons le profil de requête $req = (2, 6, 9)$ et deux vecteurs modèles représentant chacun une catégorie, $cat_1 = (2, 3, 6)$ et $cat_2 = (2, 4, 5)$. Le vecteur $d = (1, 2, 1)$ représente les degrés de discrimination, ce qui indique que la connaissance en deuxième position dans les profils est plus représentative du domaine que les autres.

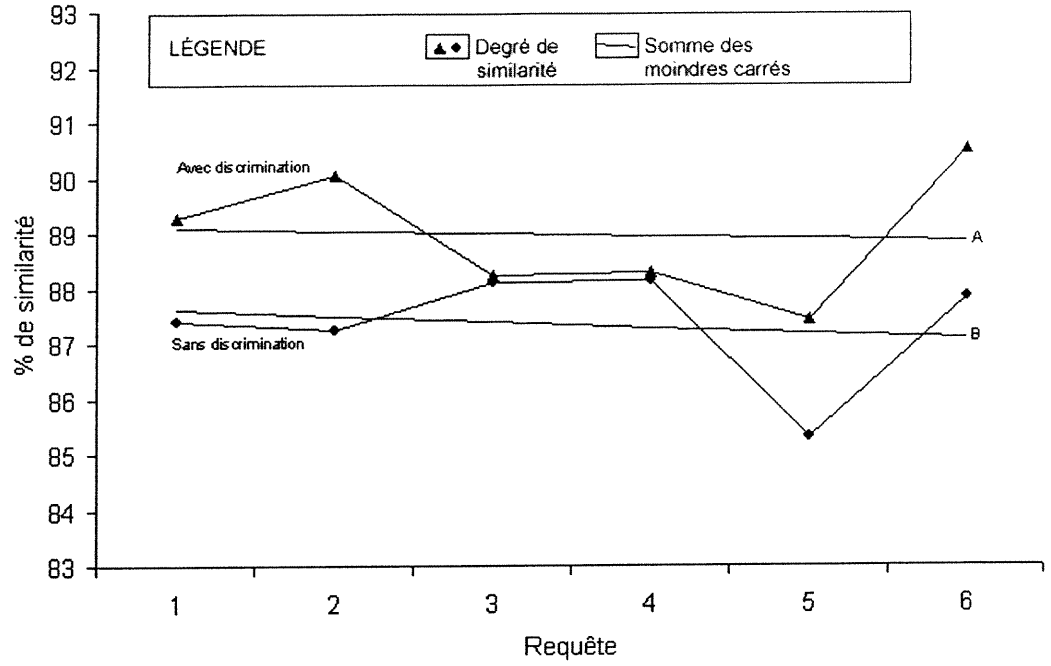

FIG. 4.8 – Graphique du degré de similarité moyen des entités en fonction des requêtes. Courbe A : courbe moyenne avec valeur de discrimination, courbe B : courbe moyenne sans valeur de discrimination

Si nous laissons de côté la valeur de discrimination, la catégorie associée à req est cat_1 puisque $dist(req, cat_1) < dist(req, cat_2)$, $dist()$ étant la fonction qui calcule la valeur euclidienne.

$$dist(req, cat_1) = \sqrt{(2-2)^2 + (6-3)^2 + (9-6)^2} = 4.2426$$

$$dist(req, cat_2) = \sqrt{(2-2)^2 + (6-4)^2 + (9-5)^2} = 4.4721$$

Dans le cas contraire, lorsque la valeur de discrimination est prise en considération, la catégorie cat_2 sera associée à req puisque $dist(req, cat_1) > dist(req, cat_2)$. La fonction $dist()$ permet le calcul de la distance euclidienne associée au degré de discrimination (équation 3.13).

$$dist(req, cat_1) = \sqrt{1(2-2)^2 + 2(6-3)^2 + 1(9-6)^2} = 5.1961$$

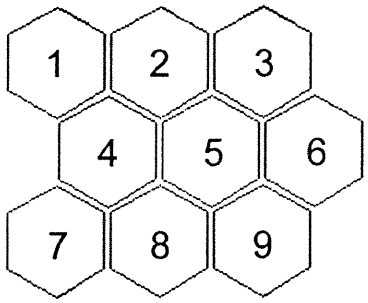

FIG. 4.9 – Disposition des neurones sur la couche de sortie.

$$dist(req, cat_2) = \sqrt{1(2-2)^2 + 2(6-4)^2 + 1(9-5)^2} = 4.8989$$

Ce résultat démontre que le facteur de discrimination peut modifier l'ensemble des entités associées à une requête. Dans l'exemple ci-dessus, la catégorie 2 (cat_2) est plus appropriée à la requête (req) puisque la connaissance en position 2 est plus représentative du domaine d'application et, par le fait même, de la requête. Il est à noter que si l'on tient compte ou non du degré de discrimination, les catégories retournées pourront être différentes mais resteront voisines. En effet, la couche de sortie du réseau de neurones étant ordonnée, le degré de discrimination ne peut pas complètement modifier les résultats.

4.5 Utilisation des liens entre connaissances

Les liens entre les connaissances sont en grande majorité utilisés lors de la construction ou lors de la mise à jour d'un profil. Ils ont comme objectif de représenter les relations existantes entre les connaissances qui constituent les profils d'entités. L'influence de ces liens se retrouve également au sein du module de requêtes.

Dans le cas où un usager recherche entre autres un individu qui maîtrise le langage de programmation C, il construira une requête où cette connaissance sera sélectionnée. Si aucun profil ne possède un poids qui permettrait de croire que ce dernier maîtrise

le langage C, les entités retournées (s'il y en a) ne seront d'aucune aide. Par contre, il serait logique d'affirmer que si un individu connaît très bien le langage C^{++}, il y a de fortes chances qu'il puisse répondre à une requête portant sur le langage C. Pour représenter la relation entre le langage C et C^{++}, un lien d'association est requis entre les deux connaissances. De cette façon, lors de la requête, chaque fois que la connaissance C^{++} sera rencontrée, le poids de la connaissance C sera augmenté selon un facteur attribué au lien d'association. De cette façon, le système sera en mesure de retourner des entités plus en mesure de répondre à la requête.

Lors de notre évaluation, nous avons remarqué que les liens entre les connaissances peuvent également, dans des cas très précis, biaiser les résultats. En effet, le poids d'une connaissance qui est reliée par plusieurs liens à d'autres connaissances a tendance à devenir important rapidement. Ceci est dû à l'augmentation du poids de cette connaissance chaque fois qu'une connaissance reliée à cette dernière est rencontrée. De plus, si la connaissance possède une valeur de discrimination élevée, elle deviendra encore plus représentative du profil de l'usager. Le poids peut alors devenir exagéré et biaiser les résultats. La figure 4.10 montre un exemple de graphe où cette situation pourrait arriver. Afin de l'éviter, deux solutions sont envisageables.
La première solution, la plus simple, serait d'éviter, lors de la construction du graphe, les connaissances qui sont trop génériques ou fortement couplées. La seconde solution serait d'insérer un facteur lié au nombre de liens dans le calcul de la distance entre deux entités. Si nous reprenons l'équation 3.16, l'insertion d'un tel facteur, appelé μ, donnerait

$$distance = \sqrt{\mu\phi_1(x_1 - v_1)^2 + \mu\phi_2(x_2 - v_2)^2 + \ldots + \mu\phi_n(x_n - v_n)^2} \qquad (4.1)$$

La difficulté majeure est la détermination du nombre de liens à partir duquel le facteur μ doit diminuer l'importance d'une connaissance. Cette dernière équation n'a pas été évaluée dans la présente version du système *Alice*.

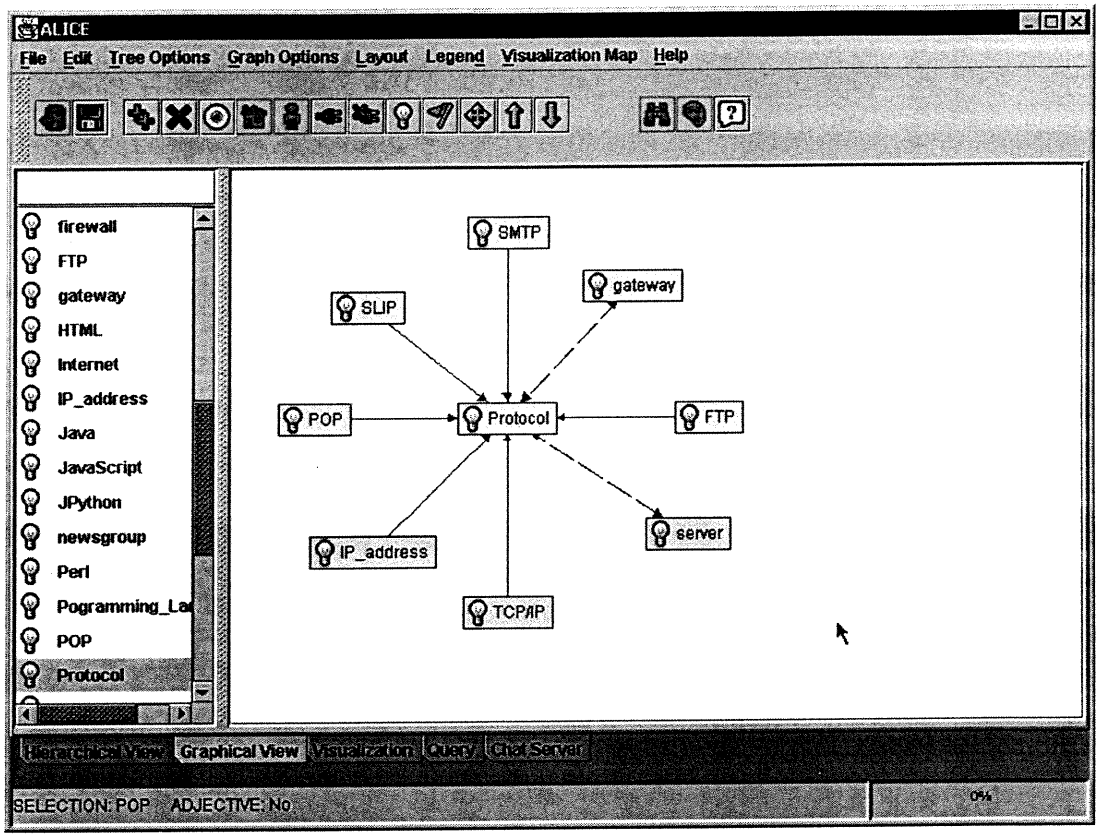

FIG. 4.10 – Exemple d'une section du graphe de connaissances pouvant biaiser les résultats

4.6 Conclusion

Suite aux trois expérimentations menées dans ce projet, nous pouvons résumer l'évaluation du système en trois points. Premièrement, les résultats obtenus lors de la première phase de tests démontrent que la catégorisation effectuée par le réseau de neurones basé sur les cartes auto-organisées de Kohonen est efficace. Nous retrouvons des degrés de similarité variant de 80% à 99% selon la configuration du réseau de neurones. Cette configuration diffère selon le nombre de neurones qui composent la couche de sortie, ce qui correspond au nombre de catégories. Cette configuration joue un rôle important également dans la spécialisation des catégories. En effet, plus la dimension de la couche

de neurones est importante, plus les catégories sont spécialisées. La catégorisation est une étape primordiale dans le bon fonctionnement de notre système *Alice*.

La seconde phase d'expérimentation a été orientée sur l'apport, au système *Alice*, du degré de discrimination associé aux connaissances extraites des publications. Les résultats ont montré que la prise en considération de cette valeur augmente la précision des entités retournées par rapport aux requêtes faites par l'usager. Ainsi, le système *Alice* peut répondre de façon plus précise aux demandes des usagers.

En dernier lieu, nous avons analysé l'effet des différents liens qui composent le graphe de connaissances au sein du système *Alice*. Sans ces liens, notre système *Alice* est plus susceptible de se retrouver sans entité à proposer à l'usager suite à une requête. Les différents liens sont en mesure de permettre au système de faire des corrélations entre les connaissances et ainsi d'augmenter sa flexibilité face à une requête. Une amélioration notable du système *Alice* serait l'insertion d'un facteur μ, qui permettrait de contrer l'effet d'une surabondance de liens sur une connaissance.

Conclusion

L'objectif principal du système *Alice*, que nous avons entièrement conçu et implanté, est de modéliser les différentes connaissances et entités que l'on retrouve à l'intérieur d'une organisation pour ensuite les analyser et en extraire une information. Cette connaissance est, selon nous, principalement constituée du savoir des individus au sein d'une organisation.

La première difficulté rencontrée a été de trouver une représentation des connaissances et des relations qui existent entre ces dernières. La méthode choisie a été une représentation basée sur les réseaux soutenue par les principes d'une ontologie et la représentation graphique d'un réseau sémantique. Nous avons nommé cette structure "graphe de connaissances". D'ailleurs, nous avons développé un éditeur graphique dans le but de faciliter la construction de ce graphe de connaissances. Nous croyons que ce formalisme de représentation ainsi que son éditeur peuvent être ré-utilisés dans la construction de graphes de connaissances s'appliquant à n'importe quel domaine d'application.

Une fois la connaissance représentée, il faut être en mesure de l'analyser pour en extraire une information pouvant prodiguer une aide aux usagers. La méthode utilisée est la reconnaissance de patrons. Cette méthode détermine par un réseau de neurones, et plus particulièrement pour les cartes auto-organisées de Kohonen, les différents patrons (catégories) au sein des entités de l'organisation. Ce type de reconnaissance de patron permet la catégorisation des profils d'entités, constitués de connaissances,

avec un apprentissage non-supervisé. Cette méthode tient compte de l'évolution des profils. De plus, nous avons inclus un degré de discrimination au sein des cartes auto-organisées de Kohonen, ce qui représente un plus pour le type de problème que nous traitons. Ce degré est associé à chaque connaissance du domaine d'application et a pour but de donner un indice sur la valeur de représentation de cette connaissance dans le domaine considéré. Donc, plus le degré d'une connaissance est élevé, plus cette dernière est importante.

La catégorisation des entités au sein de l'organisation nous permet d'avoir une représentation de la distribution des connaissances. Cette représentation tient compte non seulement des catégories découvertes mais également de la notion de distance entre chaque catégorie. Cette notion de distance est basée sur la distance entre les neurones qui forment la couche en sortie du réseau de neurones. La catégorisation permet également la formulation de requêtes pouvant venir en aide aux usager de l'organisation. Une association entre les résultats de la requête et la visualisation graphique de la distribution des connaissances est en mesure d'aider l'usager face à une requête associée à un ensemble de solution vide.

Le système *Alice* répond en grande partie aux objectifs que nous nous étions fixés en début de projet. Premièrement, les différents modules construits sont semi-indépendants les uns des autres, ce qui permet d'adapter le système à différentes sources d'informations. De plus, la méthode de représentation des connaissances peut être utilisée pour n'importe quel domaine. Par ailleurs, le processus par lequel l'usager doit passer pour construire une requête est très simple et intuitif. Lorsqu'une requête est floue ou que les résultats retournés sont peu satisfaisants, l'usager possède toujours la possibilité d'analyser les entités potentiellement aptes à l'aider par la visualisation de la distribution des connaissances. Par cette visualisation, notre objectif de fournir un espace de solutions à l'usager lorsqu'une requête n'aboutit pas à un résultat précis.

La catégorisation des entités par les cartes auto-organisées de Kohonen s'est avérée réussie. Comme les résultats le démontrent, la catégorisation s'effectue avec un fort pourcentage de réussite avant ou sans la prise en considération du degré de discrimination. Par ailleurs, les résultats montrent que lorsque le degré de discrimination est utilisé lors d'une requête, les résultats obtenus peuvent en bénéficier. Cette dernière observation dépend fortement des connaissances choisies lors de la formation de la requête. Si ces connaissances possèdent un degré de discrimination élevé, les résultats seront supérieurs, dans le cas contraire, les résultats en seront peu affectés.

Le module sur la construction du graphe de connaissances est celui qui est le plus susceptible d'évaluer dans un proche avenir. Dans la version actuelle, tous les liens entre les entités sont gérés par l'ingénieur des connaissances. Cependant, cette gestion peut devenir très complexe lorsque le nombre de connaissances devient important. Ceci s'explique par le fait que par le simple ajout d'une nouvelle connaissance, plusieurs relations peuvent relier la connaissance ajoutée à d'autres connaissances. Donc, plus le nombre de connaissances augmente, plus la mise à jour des relations se complique.

La méthode choisie pour tenter de remédier à ce problème, est l'utilisation d'un dictionnaire linguistique. Nous pensons utiliser *WordNet* [10, 14] dans le système *Alice*. *WordNet* donne la possibilité d'exécuter différentes requêtes à partir d'un système distant. Parmi les requêtes proposées, nous retrouvons :

- Les différents sens d'un même mot (s'il y a lieu) et leurs définitions.
- Les termes plus généraux ou plus spécifiques au terme désigné (lien de généralisation et de spécialisation).
- Les synonymes de ce terme (lien de similarité).
- Les termes qui peuvent être associés à ce dernier (lien d'association).

Les résultats des différentes requêtes mentionnées ci-dessus peuvent être ensuite analysés et transformés sous un modèle précis pouvant être utilisé par *Alice*. Le modèle choisi repose sur *XML* puisqu'il sera plus facile de traiter ses résultats (*api XML*

pour Java) et d'assurer la portabilité du système dans des versions futures. Par l'intermédiaire de cette information, une analyse en fonction de la nouvelle connaissance et des connaissances déjà existantes sera faite pour ensuite proposer les liens requis à l'usager. Par cette méthode, nous ne remettons pas en cause la construction de type semi-automatique puisqu'il revient toujours à l'ingénieur des connaissances de prendre la décision finale. Toutefois, cet ajout apportera une plus grande fiabilité dans la cohérence sémantique du graphe de connaissances.

Une amélioration à long terme serait l'intégration d'un module d'analyse de la langue naturelle. En effet, une des faiblesses de notre système est qu'il accomplit l'extraction des connaissances d'un document (en l'occurrence une publication) en ne tenant compte que de l'occurrence d'une connaissance. Dans la majorité des cas, cette connaissance est présente car elle fait partie intégrante du projet. On inclut donc la connaissance dans le profil de l'usager sans tenir compte de la sémantique de la phrase à laquelle elle appartient. Aussi, une analyse sémantique de la phrases serait-elle nécessaire pour éviter des incohérences sémantiques ou des interprétations erronées.

Ces derniers points seront analysés plus en profondeur puisque le système *Alice* continuera à être développé au sein de l'entreprise Virtuel-Âge International.

Bibliographie

[1] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kühn, and M. Sintek. Toward a technology for organizational memories. *IEEE Intelligent System*, 13(3) :40–48, 1998.

[2] A. Abecker and S. Decker. Organizational memory : Knowledge acquisition, integration, and retrieval issues. In Puppe F., editor, *XPS'99 Knowledge-Based Systems*, pages 113–124, Wurzburg, Germany, 1999. Springer Verlag.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley, 1999.

[4] S. Bélanger, M.A. Thibodeau, C. Frasson, and E. Aïmeur. VAI Connection, expertise managing application for a company. In *IASTED, Artificial Intelligence and Applications*, Innsbruck, Austria, 2001. To appear.

[5] S. Bélanger, M.A. Thibodeau, S. Tadié, E. Aïmeur, and C. Frasson. Agent de liaison basé sur la compétence des experts. In *Third Annual Conference on Telelearning NCE (Poster)*, Vancouver, Canada, 1998.

[6] K.D. Bollacker, S. Lawrence, and C.L. Giles. A system for automatic personalized tracking of scientific literature on the web. In *Proceedings of the 4th ACM Conference on Digital Libraries*, pages 105–113, 1999.

[7] B. Brulwich and C. Burkey. The infofinder agent : Learning user interests through heuristic phrase extraction. *IEEE Expert*, 18(2) :22–27, 1997.

[8] J. Budzik and K.J. Hammond. Watson : Anticipating and contextualizing information needsl. In *Proceedings of the Sixty-second Annual Meeting of the American Society for Information Science*, pages 38–43, Medford, NJ, 1999.

[9] B. Chandrasekaran, J.R. Josephson, and V.R. Benjamins. What are ontologies, and why do we need them ? *IEEE Intelligent Systems*, 14(1) :20–26, 1999.

[10] J. Daudé, L. Padró, and G. Rigau. Mapping wordnets using structural information. *Proceedings of the 38th Anual Meeting of the Association for Computational Linguistics*, pages 24–31, 2000.

[11] T. Dean, J. Allen, and Y. Aloimonos. *Artificial Intelligence, Theroy and Practice.* Benjamin/Cummings Publishing Company, Inc, 1995.

[12] G. Deboeck and T. Kohonen. *Visual Explorations in Finance with Self-Organizing Maps.* Springer-Verlag, 1998.

[13] P. Demartines. *Analyse de données par réseaux de neurones auto-organisés.* PhD thesis, Institut National Polytechnique de Grenoble, France, 1994.

[14] C. Fellbaum, editor. *WordNet, An Electronic Lexical Database.* The MIT Press *B*ooks, 1998.

[15] O. Gerbé. *Un modèle uniforme pour la modélisation et la métamodélisation d'une mémoire d'entreprise.* PhD thesis, Université de Montréal, Montréal, Canada, 2000.

[16] N. Guarino. Formal ontology and information systems. In *Proceedings of FOIS'98*, pages 3–15, Trento, Italy, 1998.

[17] C. Havens. Enter, the chief knowledge officer. *CIO Canada*, 4(10) :36–42, 1996.

[18] F. Heylighen. Bootstrapping knowledge representations : from entailment meshes via semantic nets to learning webs. In *Kybernetik*, 2000. In press.

[19] A.K. Jain, R.P.W. Duin, and J. Mao. Statistical pattern reconition : A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1) :4–37, 2000.

[20] A.K. Jain, J. Mao, and K.M. Mohiuddin. Artificial neural networks : A tutorial. *Computer*, pages 31–44, March 1996.

[21] J. Jannink, S. Pichai, D. Verheijen, and G. Wiederhold. Encapsulation and composition of ontologies. In *AAAI Workshop on AI and Information Integration*, 1998.

[22] S. Kaskis. *Data exploration using self-organizing maps.* PhD thesis, Helsinki University of Technology, Finland, 1997.

[23] H. Kautz and B. Selman. Creating models of real-wold communities with refferelwed. In *Working notes of the Workshop on Recommender Systems, held in conjuction with AAAI-98*, Madison, WI, 1998.

[24] H. Kautz, B. Selman, and M. Shah. The hidden web. *AI Magazine*, 18(2) :27–36, 1997.

[25] H. Kautz, B. Selman, and M. Shah. Referralweb : Combining social netwoks and collaborative filtering. *Communications of the ACM*, 40(3) :63–65, 1997.

[26] T. Kohonen. *Self-Organizing Maps.* Springer-Verlag, second edition edition, 1997.

[27] Larousse. Le petit larousse illustré, 1993.

[28] D. Leake, R. Scherle, J. Budzik, and K.J. Hammond. Selecting task-relevant sources for just-in-time retrieval. In *In Proceedings of the AAAI-99 Workshop on Intelligent Information Systems*, pages 38–43, 1999.

[29] D.B. Lenat and R.V. Guha. *Building Large Knowledge-Based Systems : Representation and Inference in the CYC Project.* Addison-Wesley, 1990.

[30] M. Liao, A. Abecker, A. Bernadi, K. Hinkelmann, and M. Sintek. Ontologies for knowledge retrieval in organizational memories. In *Workshop on Learning Software Organizational Memories (LSO) at SEKE'99*, Kaiserslautern, Germany, 1999.

[31] M. Liao, S. Hinkelmann, A. Abecker, and M. Sintek. A competence knowledge base system as part of the organizational memory. In Puppe F., editor, *XPS'99 Knowledge-Based Systems*, pages 125–137, Wurzburg, Germany, 1999. Springer Verlag.

[32] G.F. Lugar and W.A. Stubblefield. *Artificial Intelligence : Structure and strategies for complex problem solving.* Benjamin/Cummings Publishing Company, Inc, 1997.

[33] S. Milgram. The small-world problem. *Psychology Today*, 1(1) :60–76, 1967.

[34] P. Mitra, M. Kersten, and G. Wiederhold. A graph-oriented model for articulation of ontology interdependencies. In C. Zaniolo, P.C. Lockemann, M.H. Scholl, and T. Grust., editors, *Proceedings of Conference on Extending Database Technology (EDBT)*, pages 86–100, Konstanz, Germany, 2000. Springer Verlag.

[35] D.E. O'Leary. Enterprise knowledge management. *IEEE Computer*, 31(3) :54–61, 1998.

[36] D.E. O'Leary. Knowledge-management systems : Converting and connecting. *IEEE Intelligent System*, 13(3) :30–33, 1998.

[37] Ontolingua. http ://www.ksl.svc.stanford.edu :5915/doc/project-papers.html.

[38] P.E. Ross. Flash of genius. *Forbes*, pages 98–104, November 1998.

[39] S. Russell and P. Norvig. *Artificial Intelligence, A modern approach.* Printice Hall, Englewood Cliffs, New Jersey, 1995.

[40] G. Salton. *Automatic Text Processing, The transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[41] G. Salton and M.J. McGill. *Introduction to modern Information Retrieval.* McGraw-Hill, 1983.

[42] M. Stefik. *Introduction to knowledge systems.* Morgan Kaufmann Publishers, Inc, San Francisco, California, 1995.

[43] M.A Thibodeau. White rabbit - agent intelligent d'analyse de discussion pour la reconnaissance de profils d'usagers compatibles. Master's thesis, Université de Montréal, Montréal, Canada, 2001.

[44] M.A Thibodeau, S. Bélanger, and C. Frasson. White rabbit - matchmaking of user profiles based on discussion analysis using intelligent agents. In Gauthier G., Frasson C., and VanLehn K., editors, *Intelligent Tutoring Systems, ITS2000*, pages 113–122, Montréal, Canada, 2000. Springer Verlag.

[45] E. Turban and J.E. Aronson. *Decision support systems and intelligent systems.* Printice Hall, 1998.

[46] C.J. van Rijsbergen. Towards an information logic. In *12th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 77–86, 1989.

[47] C. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14 :85–100, 1973.

[48] S. Watanabe. *Pattern Recognitions : Human and Mechanical.* Wiley, NewYork, 1985.

[49] D.J. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self-organization. In *Proceedings of the Royal Society of London*, pages 431–445, 1976.

[50] P. Zweigenbaum. Encoder l'information médicale : des terminologies aux systèmes de représentation des connaissances. *Innovation Stratégique en Information de Santé*, 2(3) :27–47, 1999.