Navigateurs Internet Intelligents : Algorithmes de Fourmis Artificielles pour la diffusion d'informations dans un réseau P2P

Christophe Guéret

UNIVERSITÉ FRANÇOIS RABELAIS
TOURS

École Doctorale : Santé, Sciences
et Technologies

Année Universitaire : 2005-2006

THÈSE POUR OBTENIR LE GRADE DE
DOCTEUR DE L'UNIVERSITÉ DE TOURS

Discipline : Informatique

présentée et soutenue publiquement

par :

Christophe GUÉRET

le 8 décembre 2006

Navigateurs Internet Intelligents : Algorithmes de fourmis artificielles pour la diffusion d'informations dans un réseau P2P

Directeur de thèse : Professeur Mohamed Slimane

JURY :

Nom	**Prénom**	**Qualité**	**Lieu d'exercice**
CAPPELLO	Franck	Directeur de Recherches	INRIA Futurs - Orsay
DROGOUL	Alexis	Directeur de Recherches	IRD - Hanoï (Vietnam)
GUINAND	Frédéric	Professeur	Université du Havre
MONMARCHÉ	Nicolas	Maître de Conférences	Université François Rabelais Tours
RAMAT	Eric	Professeur	Université du Littoral
SLIMANE	Mohamed	Professeur	Université François Rabelais Tours

UNIVERSITÉ FRANÇOIS RABELAIS
TOURS

École Doctorale : Santé, Sciences
et Technologies

Année Universitaire : 2005-2006

THÈSE POUR OBTENIR LE GRADE DE
DOCTEUR DE L'UNIVERSITÉ DE TOURS

Discipline : Informatique

présentée et soutenue publiquement

par :

Christophe GUÉRET

le 8 décembre 2006

Navigateurs Internet Intelligents : Algorithmes de fourmis artificielles pour la diffusion d'informations dans un réseau P2P

Directeur de thèse : Professeur Mohamed Slimane

JURY :

Nom	**Prénom**	**Qualité**	**Lieu d'exercice**
CAPPELLO	Franck	Directeur de Recherches	INRIA Futurs - Orsay
DROGOUL	Alexis	Directeur de Recherches	IRD - Hanoï (Vietnam)
GUINAND	Frédéric	Professeur	Université du Havre
MONMARCHÉ	Nicolas	Maître de Conférences	Université François Rabelais Tours
RAMAT	Eric	Professeur	Université du Littoral
SLIMANE	Mohamed	Professeur	Université François Rabelais Tours

Remerciements

En premier lieu, je tiens à remercier Alexis Drogoul et Frédéric Guinand pour avoir accepté d'être rapporteurs et de consacrer une partie de leur temps à l'examen de mes travaux. Malgré la contrainte de temps qui leur était imposée, le sérieux de la relecture et les nombreuses remarques pertinentes formulées dans leurs rapports m'ont permis d'améliorer ma présentation pour le jour « J » et le manuscrit pour les jours à venir.

Je remercierai également Eric Ramat pour avoir accepté de participer à ce jury ainsi que Franck Capello pour avoir lui aussi accepté de venir assister à la présentation de mes travaux et dont les félicitations personnelles m'ont particulièrement touché.

Et puisqu'une thèse c'est 45 minutes de soutenance mais surtout plusieurs années de préparation, je tenais à remercier tous ceux qui m'ont soutenu depuis le début de l'aventure et sans qui rien n'aurait été possible. En particulier, je remercie Mohamed Slimane et Nicolas Monmarché pour avoir cru en moi en acceptant de m'encadrer et pour tout ce qu'ils m'ont appris. Un grand merci à Christian Proust pour son amitié, sa confiance et son entrain à faire que le Laboratoire d'Informatique de Tours et Polytech'Tours soient une grande famille dont je suis fier de faire partie. Merci également au membres de l'équipe Handicap et Nouvelles Technologies ainsi qu'aux doctorants, enseignants et personnels du département Informatique de Polytech'Tours pour ce cadre de travail agréable qu'ils ont tous contribué à entretenir.

Je remercie également Sébastien, Ameur et Christophe pour l'aide qu'ils m'ont apporté dans la « dernière ligne droite » ainsi que pour leur remarques sur le contenu du manuscrit. À ces remarques se sont ajoutées les corrections de ma famille qui ont pris le temps de relire le manuscrit, merci à eux.

Merci enfin à Jennifer pour avoir fait irruption dans ma vie et pour y être resté malgré mes sautes d'humeur ainsi que les nombreuses soirées et fins de semaines passées à travailler sur cette thèse.

Table des matières

Introduction **2**

1 Les Navigateurs Internet Intelligents **4**

1.1 Introduction : Il était une fois le Web... 4

1.2 Mise en scène 5

1.2.1 Une histoire de manque d'informations... 5

1.2.2 ... dans un contexte d'informatique pervasive 8

1.3 Problèmes rencontrés par les utilisateurs 9

1.3.1 Acquisition d'informations 9

1.3.2 Échanges d'informations 11

1.3.3 Gestion et exploitation d'informations 12

1.4 Solutions apportées par un navigateur intelligent 12

1.4.1 Aide à la navigation 13

1.4.2 Aide à la recherche d'informations 17

1.4.3 Aide à la gestion et à l'exploitation 21

1.4.4 Aide à la communication 26

1.5 Discussion 32

1.5.1 Critères de comparaison 32

1.5.2 Appréciation de l'intelligence d'un assistant 35

1.6 Conclusion 36

2 Du navigateur intelligent au Web intelligent **37**

2.1 Une seconde histoire 37

2.2 L'apparition de nouvelles technologies 40

2.2.1 Nouveaux langages liés au Web sémantique 41

2.2.2 Nouveaux langages de service 44

2.2.3 Architecture réseau Pair à Pair 44

2.2.4 Réseaux sociaux 46

2.3 Web intelligence : les prémices d'un nouveau Web 49

2.3.1 Modèle proposé par le « Wisdom Web » 50

2.3.2 Le bureau sémantique social 51

2.3.3 Le Web communautaire (Web 2.0) 52

2.4 Conclusion 53

3 Circulation d'informations dans un réseau P2P **55**

3.1 Introduction 55

3.2 Modélisation du système 57

3.2.1 Modélisation des contenus 57

3.2.2 Modélisation du réseau 61

3.2.3 Modélisation des utilisateurs 67

3.3 Circulation d'informations 69

3.3.1 Recherche d'informations (« pull ») 69

3.3.2 Diffusion active d'informations (« push ») 75

3.3.3 Diffusion passive d'informations (non dirigée) 79

3.4 Gestion de la topologie du réseau 83

3.4.1 Ajout de raccourcis 84

3.4.2 Ajustement dynamique de la topologie 87

3.5 Conclusion 88

4 Quelques modèles biologiques pour les systèmes complexes **90**

4.1 Introduction . . . 90

4.2 Activité de fourragement des fourmis . . . 91

4.2.1 Observations biologiques et modèle mathématique . . . 91

4.2.2 Adaptation des fourmis artificielles au problème de routage de paquets dans les réseaux . . . 93

4.2.3 Autre modélisation à base de fourmis artificielles . . . 94

4.3 Systèmes immunitaires . . . 95

4.4 Systèmes épidémiques . . . 96

4.4.1 Susceptible-Infecté (Susceptible-Infected - SI) . . . 96

4.4.2 Susceptible-Infected-Susceptible (SIS) . . . 97

4.4.3 Susceptible-Infected-Removed (SIR) . . . 98

4.4.4 Susceptible-Exposed-Infected-Removed (SEIR) . . . 98

4.4.5 D'autres modèles... . . . 99

4.4.6 Adaptation des modèles épidémiques à la diffusion de messages dans les réseaux . . . 99

4.5 Conclusion . . . 101

5 Description de la plateforme PIAF **103**

5.1 Introduction . . . 103

5.2 Description du système de gestion de communications de PIAF . . . 106

5.2.1 Contenu des informations en circulation dans le réseau . . . 106

5.2.2 Architecture interne d'un PIAF . . . 107

5.2.3 Mise en réseau de plusieurs PIAFs . . . 108

5.3 Dynamique de fonctionnement . . . 111

5.3.1 Echelles de temps . . . 111

5.3.2 Fourmi : Déplacement d'une information . . . 113

5.3.3 Carnet d'adresse : Mise à jour des informations . . . 116

5.3.4 Gestionnaire de connexions : Vérification des connexions . . . 117

5.4 Etude des paramètres des algorithmes . . . 118

5.5 Conclusion . . . 121

6 Etude expérimentale de la plateforme PIAF **123**

6.1 Introduction 123

6.2 Implémentation de PIAF 123

6.2.1 Réalisation d'un environnement destiné à la « production » 124

6.2.2 Nouvelle implémentation à l'aide d'un outil de simulation 126

6.3 Jeu de données mis en œuvre 128

6.3.1 Base de données d'informations 128

6.3.2 Topologie initiale 129

6.4 Etude des paramètres de l'algorithme 132

6.4.1 Paramétrage de s_{min} 132

6.4.2 Paramétrage de n^+ 138

6.4.3 Paramétrage de η 144

6.4.4 Discussion 148

6.5 Comparaison avec une diffusion aléatoire 149

6.6 Autres applications de PIAF 151

6.6.1 Classification non supervisée 154

6.6.2 Répartition de charge 158

6.7 Conclusion 159

Conclusion **160**

A Annexes **173**

A.1 Algorithmes de génération de données 173

Introduction

Comment rendre intelligent un navigateur Internet ?

Un « navigateur Internet », également dénommé « navigateur Web », a pour principale fonctionnalité l'interprétation d'une page écrite dans un language de balisage hypertexte (HTML) afin d'en produire un affichage graphique. Ces pages sont fournies par des serveurs dont le navigateur est client. La plus importante concentration de ces serveurs est sans aucun doute celle utilisant le réseau Internet : le World Wide Web (WWW ou Web). Avec un nombre de serveurs et d'utilisateurs en constante augmentation, cette toile géante fournit aux internautes qui s'en servent un moyen de communication et une source d'information non négligeable.

Pour trouver une information sur le Web, il suffit de trouver un serveur contenant la page contenant l'objet de la recherche et de la lui demander. Entretenir une discussion autour d'un sujet donné avec d'autres utilisateurs peut, par exemple, se faire par l'utilisation de sites dediés à cette activité. Si l'on désire (re)consulter un site Web connu, il suffit de saisir son adresse pour que le navigateur aille aussitôt interroger le serveur et afficher son résultat. Des exemples d'utilisation simples en théorie mais complexes dans la pratique : trouver un serveur contenant une page intéressante devient vite un problème assimilable à la recherche d'une aiguille dans une bôte de foin, la recherche d'un site de discussion autour d'un sujet précis est d'une complexité similaire et, pour saisir l'adresse d'un site, encore faut-il pouvoir s'en souvenir. Heureusement, plusieurs outils existent déjà pour assister l'utilisateur dans ses diverses activités. Les moteurs de recherche et le système de signets proposé par les navigateurs Internet en sont deux exemples.

Pour savoir ce qui permettrait de rendre un navigateur internet intelligent, nous pouvons nous intéresser aux propos de John McCarthy. Selon lui l'intelligence artificielle se définit comme la faculté qu'a une machine à se comporter comme le ferait un être humain qualifié d'intelligent[1]. Comparé à un navigateur Internet classique, un navigateur Internet « Intelligent » aurait donc pour caractéristique un comportement mimant celui d'un autre utilisateur. En cela, l'apport d'une aide lors de la navigation sur le Web fait partie de ces caractéristiques de l'intelligence d'un navigateur. Mais, comme le fait remarquer Rodney Brooks, l'intelligence est également une notion subjective dépendante de l'utilisateur. Le navigateur ne saurait être intelligent dans l'absolu mais uniquement au yeux des personnes à qui il fournit de l'aide.

[1] "making a machine behave in ways that would be called intelligent if a human were so behaving" - John McCarthy, Dartmouth conference, 1956

Sujet de la thèse

Ce travail est axé sur la conception d'outils permettant la mise en place de services qui amèneraient l'utilisateur à penser qu'il est en présence d'un navigateur « intelligent ». Nos travaux s'orientent sur une étude des différentes activités des utilisateurs d'Internet, seuls ou en groupe, afin d'identifier leurs besoins ainsi que les stratégies d'aide envisageables. Nous proposons une plateforme pour le développement d'outils d'aide intelligents. L'élément central de cette plateforme est un système de circulation d'informations conçu autour de trois principes :

- fonctionner de façon autonome ;
- être non intrusif pour ne pas déranger l'utilisateur ;
- ne nécessiter aucun paramétrage de la part de l'utilisateur.

Structure du document

- le chapitre 1 identifie le cadre de notre travail ainsi que la problématique posée. Ce cadre est celui de la conception d'outils intelligents destinés à fournir une aide aux utilisateurs. Cette conception soulève des problèmes multiples tant au niveau de l'identification des besoins que de la mise au point des programmes assistances eux même. Dans ce chapitre, nous dressons un panorama des solutions existantes et discutons de leur intelligence ;
- le chapitre 2 situe la problématique de la conception d'un navigateur intelligent relativement aux travaux menés dans le cadre de la Web intelligence ;
- le chapitre 3 présente un état de l'art des outils et méthodes destinés à la circulation de messages dans un réseau pair à pair ;
- le chapitre 4 dresse un panorama de modèles bio-mimétiques pouvant être utilisés pour concevoir des algorithmes de diffusion de message dans les réseaux informatiques ;
- le chapitre 5 contient une présentation détaillée de la plateforme PIAF (Personnal Intelligent Agent Framework) : un environnement d'échange d'informations pour agents personnels. Cette description commence par des considérations générales sur la plateforme et son positionnement par rapport aux outils existants avant de se poursuivre par la présentation des différents algorithmes développés ;
- le chapitre 6 décrit une l'étude expérimentale menée sur les algorithmes de circulation de messages proposés. Celui-ci commence par une brève présentation des prototypes réalisés avant de se poursuivre sur la présentation et la discussion des résultats obtenus ;
- ce document se termine par une conclusion sur l'ensemble des travaux présentés ainsi que les perspectives envisagées.

Chapitre 1

Les Navigateurs Internet Intelligents

Ce premier chapitre est consacré à la présentation de la problématique étudiée dans cette thèse. Nous y verrons notamment pourquoi un navigateur Internet Intelligent est un outil qui, en soit, n'existe pas (ou, du moins, pas encore). Plutôt que de fournir une définition générale de l'intelligence d'un navigateur, nous en proposons une estimation basée sur l'étude des fonctionnalités d'assistance fournies.

1.1 Introduction : Il était une fois le Web...

Dans son texte visionnaire de 1945, « As we may think » [Bush 45], Vannevar Bush présentait le « Memex » : un outil permettant l'indexation de différents types de documents et la mise en place de liens entre eux pour en faciliter l'utilisation. En effet, alors que les mécanismes proposés à l'époque étaient basés sur le principe d'indexation, l'auteur fait remarquer que le fonctionnement de l'esprit humain est basé sur l'association d'idées. En se basant sur cette réflexion, Vannevar Bush avait doté son hypothétique « Memex » d'une gestion de liens entre les éléments stockés.

Au fil des années, cette idée a été développée pour finalement devenir l'hypertexte puis l'hypermédia. L'hypertexte est un concept permettant de représenter des documents accompagnés des liens existants entre eux (les hyperliens). L'hypermédia est une extension de l'hypertexte intégrant l'idée de liens entre des ressources textuelles mais également des vidéos et des images. L'exemple le plus connu d'utilisation de l'hypermédia est le World Wide Web (WWW) : un ensemble de serveurs de données reliés entre eux pour produire une vaste source d'informations. Malheureusement, du « Memex » n'a finalement été gardée que l'idée de liens entre documents. L'idée selon laquelle des listes de liens ainsi que des annotations portant sur les documents pourraient être échangées entre les utilisateurs n'a pas rencontré le même succès.

L'hypermédia tel que nous le connaissons actuellement est un modèle à la fois simple et limité de gestion de l'information : il permet de lier entre eux n'importe quel type de document et de les publier au moyen de protocoles et de langages relativement simples à appréhender. Cependant il ne permet pas de prendre en compte le travail collaboratif ou la gestion d'informations concernant les sites visités pour ne citer que ces deux exemples. Durant ces dernières

années, le développement d'outils proposant ces fonctionnalités « manquantes » a été le sujet de recherche de bon nombre de laboratoires de part le monde. Le fruit de leurs travaux est la réalisation de programmes tierce partie chargés d'augmenter le contenu du Web ou de faciliter la communication et l'échange de données entre utilisateurs.

1.2 Mise en scène

1.2.1 Une histoire de manque d'informations...

Alors que le nombre de serveurs qui le constitue ne cesse de croître, le World Wide Web (WWW) tient une place essentielle dans le cadre de la recherche d'informations. Avec les 450 millions de domaines qui le constituent [Consortium 06], le Web constitue une source importante de données mais il en existe d'autres.

Afin de mettre en évidence ces différentes sources et leurs implications, le lecteur est invité à imaginer un scénario simpliste de recherche d'informations : un touriste, de passage dans la ville de Tours, se demande comment occuper sa soirée. Son objectif se résume à une requête : « quelle est la meilleure façon de passer la soirée à Tours en cette période de l'année ? ». Il lui faut donc interroger diverses sources d'informations pour trouver une réponse à son problème. Nous considérerons que les solutions qui s'offrent à cette personne sont au nombre de trois :

1. **Effectuer des recherches sur le WWW**
   Cette première solution est celle qui, grâce à la démocratisation de l'accès à Internet, a tendance à devenir un véritable réflexe pour de plus en plus d'utilisateurs, informaticiens ou non. Chercher une information dans le WWW revient généralement à interroger un moteur de recherche. Qu'ils se présentent sous la forme d'un logiciel tel que **Copernic Agent**[1] ou d'un site Web comme **Google**[2], **Altavista**[3], **Yahoo**[4] ou tant d'autres, ces « moteurs » indexent le contenu des serveurs WWW présents sur le réseau. Pour les interroger, l'utilisateur saisit quelques mots clés (2 ou 3 en moyenne) représentatifs de sa recherche. Notre utilisateur fictif pourrait, par exemple, utiliser le moteur de recherche Google et saisir les mots clés « sortir nuit Tours ». Après une courte attente, il ne lui restera plus qu'à faire le tri parmi les quelques 5 260 000 liens qui constituent la réponse du-dit moteur (voir figure 1.1). On peut d'ailleurs remarquer sur cet exemple le biais introduit par la prise en compte des mots eux-mêmes et non de leur sens contextuel : « Tours » n'a pas toujours pu être identifié par le site comme désignant une ville. Les premiers résultats obtenus concernent essentiellement la ville de Paris et font allusion à des tours. L'étude du traitement automatisé des requêtes de l'utilisateur est un sujet complexe du traitement automatisé du langage naturel. La perte de la sémantique, illustrée ici par la confusion sur le sens du terme « tours » dans la requête, est un des problèmes de ce domaine de recherches.
   Le principal problème de cette source d'informations est justement ce nombre important de réponses. L'utilisation d'autres moteurs, ou d'autres mots clés, aurait pu fournir des

[1] `www.copernic.com`, dernier accès le 7 octobre 2006
[2] `www.google.fr`, dernier accès le 7 octobre 2006
[3] `www.altavista.fr`, dernier accès le 7 octobre 2006
[4] `www.yahoo.fr`, dernier accès le 7 octobre 2006

FIG. 1.1 – Exemple de recherche sur le site `google.fr`

réponses en moindre nombre et/ou plus ciblées mais il reste à l'utilisateur à effectuer un travail final de fouille de données.

2. **Demander conseil à une (des) tierce(s) personne(s)**
   Une seconde solution consiste à interroger d'autres personnes : amis, passants, personnel des « point I » dans les villes, *etc* sont autant de candidats potentiels susceptibles d'apporter une réponse. Par ce biais, l'utilisateur peut espérer des informations plus ciblées qu'avec un moteur de recherche. Ceci dit, selon le nombre de personnes interrogées et de réponses fournies, le travail de fouille de données demeurera plus ou moins conséquent. La principale difficulté reste de savoir *qui* interroger. Dans un environnement de taille limitée, il est possible de connaître les centres d'intérêt ou les compétences de chacun. Pour les structures de taille plus importante, cela devient une tâche plus difficile et le risque de passer à côté de la bonne personne plus grand.
3. **Consulter de la documentation précédemment accumulée**
   Il se peut enfin, et c'est la troisième solution, que l'utilisateur dispose déjà de l'information qu'il cherche. Annoncée dans une brochure publicitaire, décrite dans un guide touristique voire notée sur un pense-bête à la va-vite : il se peut que la soirée idéale soit finalement à portée de main. Reste à s'en souvenir et, plus particulièrement dans le cas de documents électroniques, à retrouver le document intéressant. Les disques durs actuels permettent d'accumuler une quantité conséquente de documents de toute sorte ayant pour unique identifiant un nom défini par l'utilisateur. Ainsi est-il possible, un certain manque d'organisation aidant, de ne pas être en mesure de retrouver un document en ne connaissant que son contenu.

Il est d'ores et déjà possible de tirer une première conclusion de cette mise en scène : quelle que soit la source d'informations interrogée, il reste à l'utilisateur à effectuer une tâche de filtrage parmi l'ensemble des réponses récupérées. Le schéma de la figure 1.2 représente l'enchaînement de ces trois activités. On y remarque que la résolution du besoin initial (trouver une occupation) peut engendrer un second besoin (chercher un moyen de locomotion et/ou un plan de la ville, par exemple, afin de se rendre sur le lieu choisi).

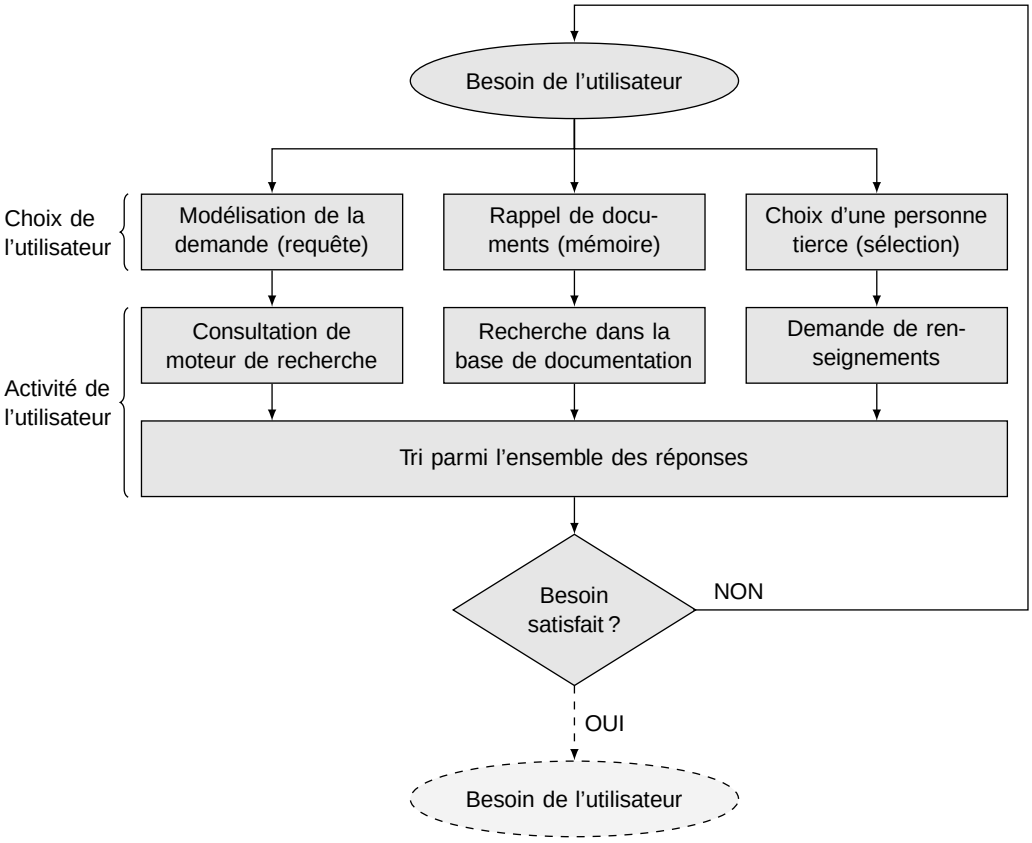


FIG. 1.2 – Processus de recherche mené par l'utilisateur

1.2.2 ... dans un contexte d'informatique pervasive

Anciennement synonyme d'appareillage lourd, l'informatique se miniaturise et s'intègre dans notre environnement sous diverses formes : téléphone portable, assistant électronique personnel (« Personnal Digital Assistant » - PDA), ordinateur portable, *etc.* Le développement récent de technologies sans fils et filaires font que ces outils informatiques, en plus de s'intégrer dans leur environnement, sont également capables de dialoguer entre eux, sans intervention de l'utilisateur.

Le terme d'informatique ubiquiste (« Ubiquitous Computing ») [Weiser 99] est utilisé pour désigner ce principe d'ordinateur se fondant dans l'environnement de l'utilisateur. Le projet **UbiMAS** [Bagci 03] est un exemple d'application de ce principe à la localisation d'employés dans une entreprise. Des ordinateurs sont placés au niveau des portes de bureaux d'une entreprise ainsi que dans les poches de chaque employé sous la forme de badges électroniques. En suivant les allers et venues de ces badges, les portes peuvent aider un utilisateur cherchant un collègue en lui indiquant la porte ayant « vue » son badge pour la dernière fois. Cette faculté témoigne d'une intelligence pervasive (« Pervasive intelligence » - PI) dûe au fait que l'ensemble des portes constitue un écosystème d'agents régit selon des règles inspirées de la biologie, des systèmes physiques ou chimiques [Drogoul 02].

Pouvoir accéder à n'importe quelle information depuis n'importe quel endroit est un avantage qui nécessite la mise en place de nouveaux moyens de communication et de gestion de l'information. Le projet **Collaborator** [Bergenti 04] répond à ces besoins en proposant un système d'informations accessible depuis n'importe quel appareil disposant d'un navigateur Internet. Le choix du navigateur Internet comme terminal d'accès est motivé par plusieurs atouts du Web : (1) le Web est facilement accessible depuis un appareil « communiquant » (2), un serveur Web permet d'intégrer différents services dans une seule structure uniforme (3), l'accès au Web est indépendant de la plate-forme de développement. L'emplacement des serveurs Web n'a quant à lui que peu d'importance du moment qu'ils puissent être contactés.

Le développement de ces environnements pervasifs nous permet d'envisager l'exemple précédent dans un cadre « tout informatisé » où toutes les sources d'informations seraient accessibles *via* un navigateur Internet. Plus spécifiquement, il est possible de distinguer quatre acteurs :

- les utilisateurs qui sont à la recherche d'informations ou susceptibles d'en fournir ;
- les données dont dispose chaque utilisateur ;
- le réseau autorisant la communication entre utilisateurs ;
- le navigateur Internet comme moyen d'accès à ce réseau.

La figure 1.3 représente leurs interactions[5]. Sur cette figure, on peut remarquer que le navigateur Internet n'interagit ni avec l'utilisateur ni avec le réseau des serveurs Web. Ceci est dû au fait que le navigateur n'est considéré qu'en tant qu'outil permettant la consultation du Web par l'utilisateur.

[5]Les icônes utilisées pour ce schéma ainsi que d'autres illustrations font partie du « Tango Desktop Project »(`tango.freedesktop.org`) et sont disponibles sous licence « Creative Commons Attribution Share-Alike » (Cc by-sa)

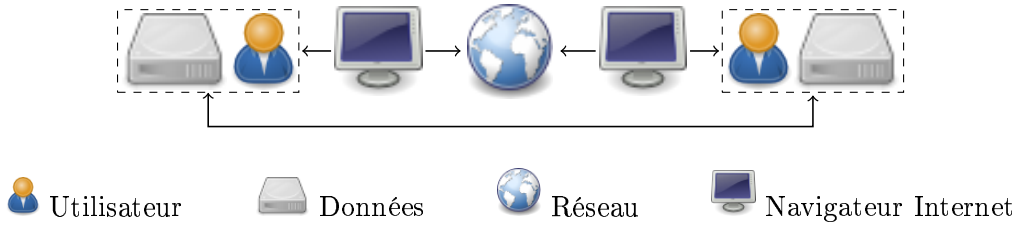


FIG. 1.3 – Le Navigateur Internet comme intermédiaire dans la démarche d'accès à l'information

1.3 Problèmes rencontrés par les utilisateurs

Les trois solutions de notre exemple introductif peuvent être ramenés aux trois activités qu'un utilisateur, face à sa machine, est susceptible d'effectuer :

- **acquisition d'informations** : « effectuer des recherches sur le WWW » revient à chercher à acquérir de l'information en consultant divers sites Web.
- **échange d'informations** : « demander conseil à une(des) personne(s) tierce(s) » est une solution qui consiste, d'une manière générale, en l'échange d'informations entre utilisateurs.
- **gestion et exploitation d'informations** : « consulter de la documentation précédemment accumulée » est une tâche qui se rapporte aux problèmes de la gestion de l'information accumulée.

Pour chaque activité, nous donnons ci-après une définition plus précise de la problématique posée ainsi que des difficultés rencontrées par l'utilisateur.

1.3.1 Acquisition d'informations

L'activité de consultation d'un ensemble de sites par l'intermédiaire d'un navigateur Internet peut se décomposer en deux sous-activités :

- la **navigation** durant laquelle il est courant d'aller de site en site de manière opportuniste. La sélection des liens et l'acquisition d'informations suit alors des préférences définies sur le long terme.
- la **recherche** qui vise à satisfaire un besoin plus ciblé et ponctuel. L'objectif est alors de trouver une information particulière.

Alors qu'en français il n'existe pas de réelle distinction (on parle, plus généralement, de « navigation »), les correspondances anglophones sont respectivement « browsing the Web » et « searching the Web ». Il est également à noter que la limitation entre ces deux activités reste assez vague dans la mesure où une recherche ponctuelle n'empêche pas l'utilisateur de faire de la navigation. Ce dernier peut, alors qu'il recherche une information précise, trouver d'autres information intéressantes.

Recherche et navigation sont à rapprocher des deux classes d'accès à l'information identifiées par Mizzaro et Tasso [Mizzaro 02] :

- **la recherche d'informations (RI)** qui caractérise un besoin précis concernant une base de données de documents définie. Le cas le plus courant est l'utilisation de moteurs de

recherche sur Internet. Il s'agit d'une activité complexe à traiter automatiquement car l'utilisateur doit pouvoir exprimer un besoin ;

- **le filtrage d'informations (FI)** qui exprime une notion contraire où la base de données de documents est plus progressive et le besoin exprimé de manière plus générale. Les utilisateurs de systèmes de filtrage d'informations sont plus à même d'exprimer leur préférences et d'attendre plus longtemps la réponse.

Ainsi, naviguer sur Internet signifie filtrer les informations intéressantes parmi l'ensemble des sites parcourus. Cette notion de filtrage se retrouve également dans les outils de filtrage collaboratif que nous verrons ultérieurement.

Un sondage effectué en 1997 [Team 97] et dont quelques résultats sont présentés dans le tableau 1.1, avait permis de visualiser plusieurs problèmes liés à l'utilisation du Web. On y remarque que les utilisateurs ont principalement une attitude les conduisant à sauter de site en site. Bien qu'il puisse s'agir d'un choix de leur part, le rapprochement avec la proportion d'utilisateurs éprouvant des difficultés pour retrouver une page déjà visitée met en évidence un problème lié au repérage dans les sites. Si les utilisateurs sont opportunistes, c'est peut être tout simplement parce qu'ils sont perdus ! Un an auparavant, une autre étude du même organisme avait également mis en évidence une certaine difficulté éprouvée lors de la question du stockage et de la réutilisation des pages récupérées [Pitkow 96].

Utilisateurs	**Résultat**
17.8 %	Difficultés pour retrouver une page déjà visitée.
8.8 %	Difficultés pour visualiser les emplacements passés et futurs dans le WWW
6.4 %	Difficultés pour déterminer l'emplacement courant
87.7 %	Naviguent sur le Web d'une manière opportuniste

TAB. 1.1 – Résultats d'une étude sur le Web datant de 1996 (adapté de [Bouras 01])

Bien que ces études datent des années 1990 dans un cadre où tout évolue très vite, les problèmes relevés restent paradoxalement d'actualité en 2000. Sur la base des résultats fournis par la première étude que nous avons cité, Bouras *et al.* [Bouras 01], ont identifié précisément les problèmes liés à la navigation. La recherche d'informations apporte également son lot de difficultés spécifiques. Au final, les problèmes rencontrés par l'utilisateur sont nombreux :

- le sentiment de **désorientation** et l'impression de se perdre lors de la navigation dans un environnement non linéaire ;
- l'**effort cognitif** que l'utilisateur doit faire pour pouvoir gérer mentalement le parcours effectué dans un site et le tri dans l'information accumulée ;
- le **manque de contexte** car une seule page est consultée à la fois ;
- le **besoin d'indices sur le contenu**. Une fois arrivé sur un site, il n'est pas toujours aisé de se faire une idée précise de son contenu ;
- le **manque de contrôle sur l'accès aux pages** provenant du fait que le lecteur est susceptible d'arriver sur une même page de différentes façons ;
- la difficulté à **définir un besoin**, particulièrement dans le cadre de l'utilisation de moteur de recherche.

1.3.2 Échanges d'informations

L'échange d'informations est une activité importante et couramment pratiquée dans un groupe d'utilisateurs. Cet échange consiste en fait en un transfert unilatéral d'informations depuis son détenteur vers une autre personne, à l'initiative de l'une ou l'autre des parties concernées. Ces échanges constituent une certaine forme de collaboration entre utilisateurs qui peut avoir deux aspects différents [Taher 04] :

- dans le cadre d'un **travail collaboratif**, par exemple la conduite d'un projet en groupe, les échanges auront pour objectif la diffusion globale de l'information. Il s'agit de s'assurer que tous les participants obtiennent les informations nécessaires. Typiquement, cela implique l'utilisation de logiciels de gestion de travail de groupe (« *groupware* » ou « collecticiels ») tels que **Egroupware**[6] ou **Hula**[7]. Ces outils permettent à un groupe engagé dans un travail collaboratif d'échanger de l'information ;
- le **filtrage collaboratif**[8] correspond à une coopération plus informelle. A la différence du filtrage classique, le filtrage collaboratif est effectué selon des critères personnels mais également en fonction de ceux des autres membres du groupe. Les outils permettant ce filtrage collaboratif sont divers, mais une différenciation est faite entre le filtrage collaboratif actif et son opposé passif [Lueg 98]. Cet adjectif qualifie le besoin, pour une personne ayant trouvé une information, d'avoir l'intention de la partager avec le groupe.

Selon [Lueg 98], ces technologies peuvent être considérées comme similaires. Cependant, deux points restent discriminants. La nature du groupe premièrement : contrairement au filtrage collaboratif, le collecticiel sert de support à un groupe préétabli. Ce groupe pouvant être défini par la nature d'une organisation ou par l'existence d'un projet commun aux membres. Deuxièmement, le niveau de participation requis : un *groupware* nécessite l'implication de tous les membres pour communiquer leurs information alors que le filtrage collaboratif peut se révéler significatif même dans le cas où une certaine asymétrie est observée. Cette asymétrie se présente entre les personnes fournissant de l'information et ceux l'utilisant.

Les travaux de [Grudin 88] puis plus récemment ceux de [Lueg 98] et [Taher 04] mettent en évidence les problèmes communs aux collecticiels et au filtrage collaboratif actif :

- le nécessaire **besoin d'acceptation de l'utilisateur**. Il ne peut y avoir collaboration qu'avec la participation active des utilisateurs. Soit un effort supplémentaire que ces derniers ne seront pas nécessairement enclin à fournir. Une des conditions d'acceptation peut être le bénéfice obtenu à l'utilisation : l'effort supplémentaire pourrait être consenti si le service en retour est intéressant ;
- conséquence directe du point précédent, le **lien entre avantages tirés et utilisation**. Pour être efficaces, ces outils doivent être utilisés par tous les membres du groupe. Or, pour qu'ils les utilisent, les utilisateurs ont besoin de trouver ces outils efficaces. Pour qu'il soit utilisé, l'outil doit se montrer efficace et pour être efficace, l'outil doit être utilisé : c'est une spirale dont il n'est pas forcément facile de sortir ;
- en conséquence de ces deux problèmes, le **démarrage à froid** est particulièrement délicat. Il faut que dès sa première utilisation l'outil se montre efficace et puisse facilement être accepté par les utilisateurs.

[6] `www.egroupware.org`, dernier accès le 2 octobre 2006

[7] `hula-project.org/Hula_Project`, dernier accès le 2 octobre 2006

[8] Il est également possible de rencontrer le terme de « filtrage social » (*social filtering*) pour désigner cette même notion.

1.3.3 Gestion et exploitation d'informations

La gestion d'informations et l'exploitation constituent la dernière classe d'interactions que nous avons pu identifier. Celles-ci définissent la relation qui existe entre l'utilisateur et ses données. La façon dont l'utilisateur a accès aux information qu'il possède relève de la gestion alors que l'exploitation relève de l'utilisation pratique de ces données. Il n'existe pas de frontière nette entre gestion et exploitation. Seule l'échelle de temps est discriminante : l'exploitation se fait dans l'instant alors que la gestion implique des processus à plus long terme. Aussi, nous avons ici choisi de présenter ensemble ces deux notions.

Le sondage effectué par [Team 97] avait déjà permis de mettre en évidence la difficulté qu'ont 17.8 % des utilisateurs à retrouver une page déjà consultée, soit à retrouver une information qu'ils possèdent déjà, celle de la localisation de cette page. Plus récemment, la « perte » de fichier dans les supports de stockage a pu être observée. De plus, dans une autre optique, la lecture même d'un document récupéré et l'accès aux information qu'il contient (ou, du moins, est susceptible de contenir) peuvent être compliquées par les limitations physiques et/ou cognitives de l'utilisateur.

1.4 Solutions apportées par un navigateur intelligent

Pour aider l'utilisateur à faire face à tous ces problèmes, diverses solutions ont été proposées. Toutes ont pour but de rendre le navigateur plus « intelligent ». A travers ces outils, le navigateur passe du statut de simple intermédiaire à celui d'assistant (voir figure 1.4).

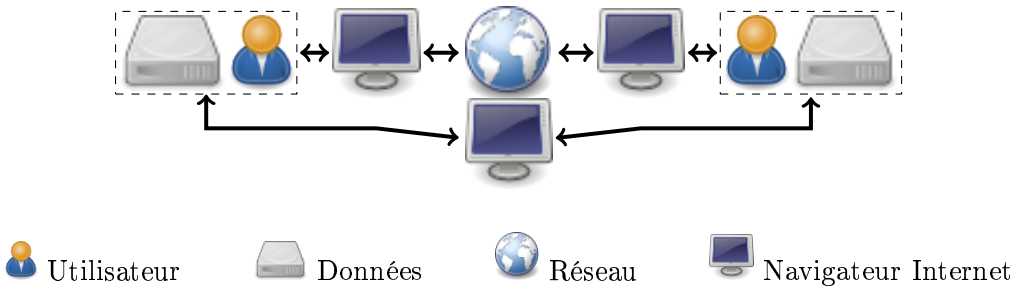


FIG. 1.4 – Nouveau schéma d'intéractions et nouvelle place du navigateur Internet

Avec, pour chacune des interactions observées, une stratégie d'aide ciblée :

- **Aide à la navigation**
  Cette aide est principalement axée autour de la découverte de nouveaux sites où la suggestion de pages à consulter. Parmi les aides proposées vont se trouver les outils permettant de suggérer de nouveaux sites, les cartes de navigation aidant l'utilisateur à se souvenir du chemin parcouru de site en site ou encore les systèmes de suggestion de produits sur les sites commerciaux ;
- **Aide à la recherche**
  Contrairement aux activités de navigation, en période de recherche l'utilisateur exprime un besoin précis et ponctuel. Les aides proposées sont dans ce cas articulées autour des principes de (re)formulation automatique de requêtes, l'indication ciblée de sites à consulter ou l'interrogation automatique de bases de données d'informations.

- **Aide à la communication**
  Cette aide se trouve dans tous les outils facilitant l'échange d'informations entre utilisateurs. L'objectif est de faciliter la circulation de l'information entre les membres d'un même groupe, prédéfini ou non.
- **Aide à la gestion d'informations**
  Le but est d'aider l'utilisateur à mieux gérer les informations dont il dispose. Plusieurs solutions peuvent être proposées parmi lesquelles l'indexation automatique de fichiers par le contenu ou le rappel de l'existence de certaines sources d'informations.

Les sous-sections qui suivent sont consacrées à une présentation d'un certain nombre d'outils destinés à aider l'utilisateur. Sans avoir la prétention d'être exhaustif, cet état de l'art présente un panorama de solutions significatives dans chaque domaine. Les outils présentés ici sont classés selon leur cadre d'utilisation idéal mais sont généralement utilisables de façon plus large.

1.4.1 Aide à la navigation

La navigation sur Internet se caractérise pas un processus actif de filtrage d'informations de la part de l'utilisateur. Alors qu'il navigue de site en site, essentiellement de manière opportuniste, ce dernier glane des informations au hasard des pages consultées. La sélection des informations retenues se base sur des critères à long terme tels que les centres d'intérêt de l'utilisateur et ne répond pas nécessairement à un besoin précis et ponctuel. Dans ce cadre, les assistants sont essentiellement dédiés à aider l'utilisateur à parcourir un (des) site(s) et/ou lui suggéreront d'autres adresses à visiter. Selon que la portée de leurs suggestions se limite au voisinage du site consulté ou au Web dans son ensemble, il est possible de distinguer deux catégories d'assistants à la navigation :

- **Construction de cartes de navigation et guides pour site Internet**
  Les cartes de navigation et les guides ont pour vocation d'aider l'utilisateur à ne pas se perdre dans le « cyber espace ». L'aide fournie par une carte est de nature essentiellement passive, comme le serait celle fournie par le plan d'une ville ou d'un parc d'activités. Elles indiquent à l'utilisateur un parcours recommandé et/ou le parcours effectué (notion de tours guidés - « *guided tours* »). Les guides de site sont, quant à eux, comparables à ceux que l'on pourrait trouver dans un musée : ces derniers recommandent aux utilisateurs en visite les liens les plus intéressants à suivre. Ce sont des guides de tours (« *tour guide* ») actifs.
- **Exploration et suggestion d'adresses**
  La suggestion d'adresses effectuée par un guide qui, à la manière d'un guide touristique ou de musée, va recommander à l'utilisateur les liens à suivre, permet à un utilisateur de découvrir de nouveaux sites. L'idée principale est, partant du site qu'il consulte, de lui indiquer quelques adresses au contenu similaire. Le guide de tour va conseiller un Internaute sur le meilleur parcours à effectuer pour profiter pleinement du site consulté.

Il est possible de remarquer que, bien qu'étant particulièrement adaptés à la navigation, ces outils peuvent aussi se révéler efficaces dans le cadre de recherches.

1.4.1.1 Cartes de navigation et guides pour site Internet

Les pages Internet consultées sont le plus souvent composées d'un nombre conséquent de liens hypertexte pouvant mener soit vers une autre page, soit vers un autre site. Le langage de programmation actuellement employé (HTML4) ne permet pas d'associer de notion sémantique aux liens, l'Internaute ne peut donc faire son choix qu'en se basant sur l'intitulé du lien. L'objectif de ces deux outils est de guider l'utilisateur dans le choix des liens à suivre. Ce guidage peut s'effectuer soit avec l'aide d'une carte, soit en indiquant directement dans la page quels sont les liens les plus judicieux à suivre.

Les cartes de navigation sont construites et affichées sur l'écran de façon à ne pas entraver la navigation de l'utilisateur. Ainsi sont-elles généralement situées en dehors de la zone d'affichage de la page. Deux exemples de carte de navigation sont visibles sur les captures d'écran des figures 1.5 et 1.6, sur le côté gauche du navigateur. Elles correspondent respectivement aux cartes produites par **Footprints** [Wexelblat 99] et **Nestor** [Bélisle 99a, Bélisle 99b] et correspondent à deux stratégies différentes pour la construction de carte :

- les cartes produites par **Footprints** se basent sur l'idée qu'à l'instar de traces de pas laissées sur des chemins par des randonneurs, l'intéraction entre internautes et sites Internet peut être mémorisée et constituer un historique d'intéraction [Wexelblat 99]. Chaque carte est associée à un site en particulier. Toutes les nuits, un serveur analyse les parcours effectués par les utilisateurs ayant consulté le site. Cette information est utilisée afin de calculer une importance relative des différents liens entre les pages ;

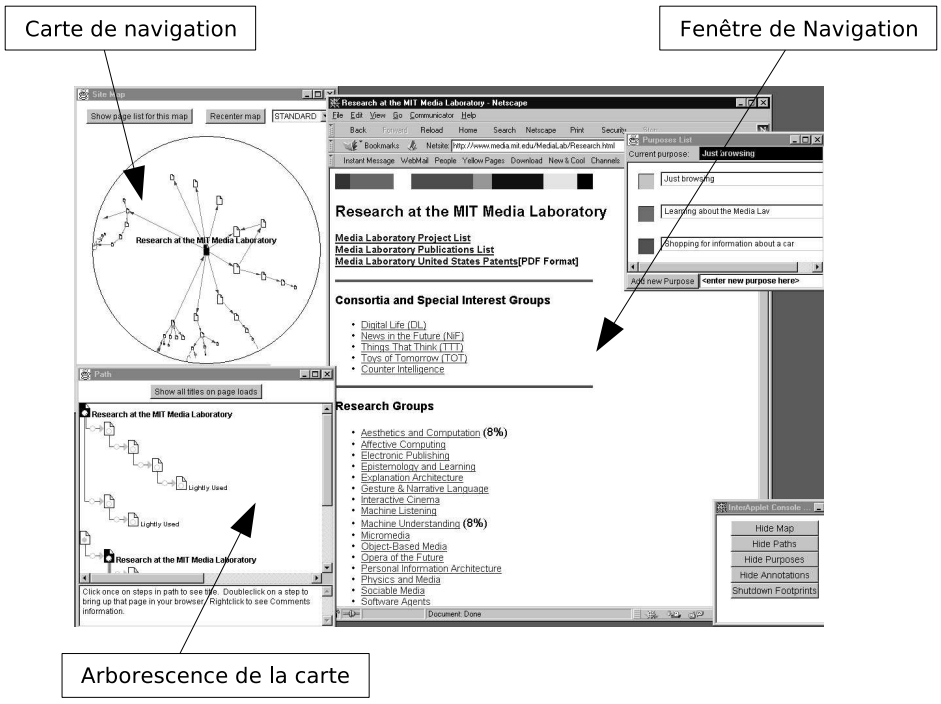


FIG. 1.5 – Exemple d'utilisation de Footprints [Wexelblat 99]

- **Nestor** [Bélisle 99a, Bélisle 99b] n'est pas axé sur les sites mais sur l'utilisateur. Alors que ce dernier navigue de site en site au gré des liens suivis, **Nestor** établit une carte du parcours effectué. Cette carte peut être utilisée comme un historique et peut permettre de revenir directement à un site ou une page sans avoir à cliquer sur le bouton « retour arrière » du navigateur. Grâce à ses possibilités d'annotation et d'import/export sous forme de fichier XML, la carte peut également se changer en guide et tour commenté pour un ensemble de site.

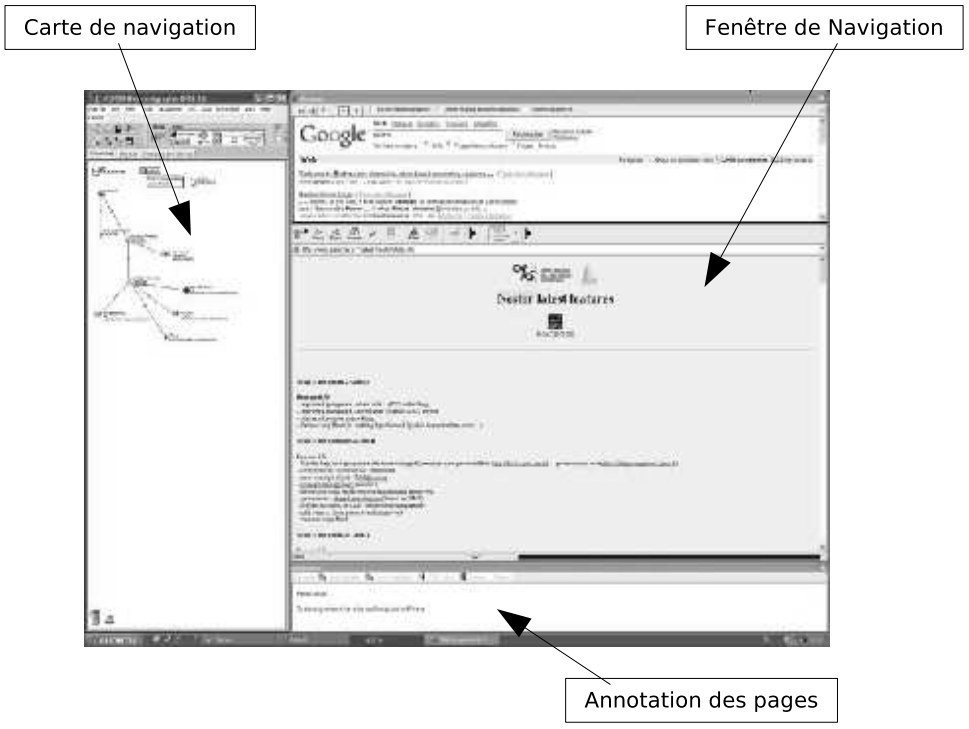


FIG. 1.6 – Exemple d'utilisation de Nestor [Zeiliger 05]

Si, au lieu d'utiliser des cartes afin de chercher son chemin, l'utilisateur préfère être conseillé, il existe des guides de tours. Ces guides, agissant comme des agents explorateurs [Lieberman 01], effectuent des recherches pour le compte de l'utilisateur et affichent leur résultat sous la forme de suggestions de liens à suivre.

Letizia [Lieberman 95] fut un outil précurseur de ce domaine. Développé par Henry Lieberman au « Massachusetts Institute of Technology », sa première version date de 1995 et utilisait le navigateur Internet « Mosaic » [9]. Son principe est de surveiller le comportement de l'utilisateur au fur et à mesure de ses recherches sur le réseau. Les données sont analysées en temps réel afin de permettre au navigateur de suggérer des liens parmi ceux listés sur les pages Internet parcourues. Le modèle adopté est une collaboration entre l'utilisateur et Letizia qui recherchent tous deux de l'information sur la même page. La différence se situe sur la fiabilité du critère de choix qui est plus efficace pour l'homme et sur la vitesse de traitement qui est cette fois en faveur de la machine. Letizia effectue une recherche en largeur et en temps réel dans la page

[9] Ce navigateur a été développé par le NCSA, jusqu'en 1993. Un historique est visible sur `http://www.ncsa.uiuc.edu/News/MosaicHistory/`

vue par l'utilisateur. Les pages candidates sont filtrées grâce à des profils appris en observant l'activité de recherche conduite par l'utilisateur. Letizia profite du temps passé par l'utilisateur à lire la page pour effectuer ses recherches, le programme explore les liens de la page courante afin d'anticiper le comportement de l'utilisateur. Plutôt que d'ouvrir une à une les pages derrière les liens, Letizia indique à l'utilisateur quels sont les liens potentiellement intéressants. L'analyse d'une page et les suggestions sont faites en fonction de mots clés précisés par l'utilisateur.

Un autre projet, **Webwatcher** [Joachims 97], met en avant un aspect de navigation collaborative d'un site. A l'instar de Footprints, il permet de recommander des liens selon la fréquentation qu'ils reçoivent. La différence principale avec ce dernier se situe au niveau du tri que fait Webwatcher afin d'affiner les suggestions selon le contexte de la navigation en cours. Contrairement à Letizia, au lieu de s'installer sur un poste client et apprendre les préférences d'un utilisateur en particulier sur sa navigation de plusieurs sites, Webwatcher est installé sur le serveur Web qui héberge le site dont le parcours doit être optimisé. Un autre projet, Personnal Webwatcher, développé par la même équipe se rapproche plus du projet Letizia. L'utilisateur commence par indiquer à Webwatcher quel est son but et doit à la fin indiquer au programme si les aides qui lui ont été prodiguées se sont révélées utiles ou non.

Le site de vente en ligne **Amazon**[10] est représentatif d'une autre forme de guidage à l'intérieur d'un site Internet. Lorsqu'un utilisateur choisi de commander un produit, ce dernier se voit recommander un certain nombre d'autres articles. Pour ce faire, Amazon utilise l'historique des articles commandés ainsi que le contenu du panier en cours qu'il compare avec les informations obtenues pour d'autres utilisateurs.

1.4.1.2 Exploration et suggestion d'adresses

Un guide de navigation peut également étendre ses suggestions à l'ensemble du Web au lieu de les restreindre au voisinage du site consulté. Ainsi, l'utilisateur se voit proposé une liste de site Internet connexes et dont il ignorait probablement l'existence. Dans une approche similaire aux guides, les suggestions faites peuvent être le fruit d'une recherche autonome de l'agent ou de l'exploitation des données provenant d'autres utilisateurs.

L'assistant **PowerScout** est représentatif de la première approche. Alors que l'utilisateur consulte les pages, cet outil extrait une liste de mots clés. Ceux-ci, ainsi qu'une liste complémentaire paramétrée par l'utilisateur, sont utilisés afin d'interroger différents moteurs de recherche. Les liens ainsi récupérés sont présentés à l'utilisateur classés selon leur similarité avec la page affichée. Les sites se retrouvant en haut de la liste étant ceux les plus susceptibles de détenir des informations dont le thème est proche de la page actuellement consultée. Les auteurs introduisent la notion de « navigation conceptuelle » pour décrire l'utilisation de ces liens : les pages ne sont plus parcourues selon les liens placés par les auteurs mais selon la similarité entre les documents selon les mots qu'ils contiennent.

Lira [Balabanovic 97] propose une approche similaire à la différence que celle-ci est asynchrone. Une fois par jour, les utilisateurs de cet outil se voient proposer une liste de sites à visiter et à évaluer avec une note allant de 1 à 5. Cette liste est actualisée toutes les nuits en interrogeant des moteurs de recherche. La requête utilisée est composée en fonction des évaluations

[10] `www.amazon.com`

précédentes (profil de l'utilisateur) et des sites consultés dans la journée. Cette évaluation des pages pouvant s'avérer fastidieuse, **WebMate** [Chen 98] propose de remplacer la notation par un test booléen. *via* un simple clic, l'utilisateur indique si oui ou non la page qu'il consulte lui plaît (« I like it »). WebMate présente également la particularité d'utiliser différents profils pour un même utilisateur afin de modéliser ses différents centres d'intérêt.

Le site **Alexa**[11], édité par la société du même nom, propose une autre approche de la recommandation de liens. Les recommandations sont faites en fonction de la popularité des sites. Pour profiter de ce service, il suffit pour l'utilisateur d'installer une barre d'outils disponible sur le site d'Alexa. Cette barre d'outil vient s'ajouter à celles déjà présentes dans le navigateur. Par l'intermédiaire de cette barre, le moteur d'Alexa recommande des sites à consulter. La recommandation est calculée, pour chaque site, sur la base de trois critères :

1. le « reach rank » qui indique le nombre de personnes ayant visité le site sur une population ramenée à un million d'Internautes ;
2. la moyenne du nombre de pages consultées dans le site ;
3. l'« Alexa rank » calculé en fonction des deux précédents critères et qui détermine le classement du site.

La barre d'Alexa se charge également d'effectuer des remontées d'informations pour indiquer quels sont les sites consultés par l'utilisateur.

Let's browse [Lieberman 99], une extension de Letizia, a pour objectif de tirer partie d'une action collaborative de recherche sur Internet. En considérant un groupe d'utilisateurs, celui-ci est capable de suggérer à un des membres ou au groupe dans son ensemble des liens intéressants pour lui/eux. Ces liens sont choisis parmi ceux présents sur les pages personnelles de chaque membre du groupe. Letizia est utilisé afin de parcourir l'ensemble de ces sites. « Let's browse » effectue ensuite l'analyse des profils des personnes présentes à un instant donné devant l'écran de la machine et la suggestion de liens adéquats. Sa première utilisation fut à la conférence « Media Lab's Digital Life Consortium » en Octobre 1997. Lors de cet événement, tous les intervenants étaient munis de badges électroniques, ce qui permis, en les utilisant conjointement avec un récepteur, de détecter quelles personnes étaient devant l'écran de l'ordinateur à un instant donné. Une fois qu'une personne était détectée, son profil était calculé en fonction des mots clés sur sa page internet personnelle ou, le cas échéant, sur la page internet de son équipe de recherche. Disposant des mots clés de chaque intervenant susceptible de voir les messages sur l'écran, l'ordinateur peut ainsi parcourir le Web et suggérer des liens selon les mots clés trouvés dans les pages et leur concordance avec ceux de chaque personne.

1.4.2 Aide à la recherche d'informations

Le World Wide Web est formé par un grand nombre de serveurs proposant chacun un ensemble de sites. Ces sites sont eux-mêmes composés d'un certain nombre de pages. La recherche d'informations consistant à trouver une ressource particulière se décompose donc en deux étapes : trouver un site susceptible de répondre favorablement à la requête puis trouver l'information dans ce site. A chacune de ces étapes correspond un type d'aide particulier :

- **Recherche de sources d'informations**

[11] `www.alexa.com`, dernier accès le 7 décembre 2006

Afin de répertorier les sites Internet existants, certains autres sites constituent des annuaires. Ces annuaires sont par la suite interrogés par les utilisateurs en quête d'une source d'informations.

- **Filtrage d'informations depuis une ou plusieures sources**
  Permet de consulter un certain nombre de sources d'informations dont on sait qu'elles sont susceptibles d'apporter une réponse.

1.4.2.1 Recherche de sources d'informations

Une première solution pour trouver une source d'informations est de partir d'une première source connue et d'en explorer son voisinage. Les **Webrings**[12], ou anneaux de sites, illustrent ce principe en liant des sites de sujet similaire entre eux. Il est possible d'effectuer le tour de l'anneau en suivant des liens vers le site qui suit ou précède celui en cours de consultation. Ce mécanisme d'anneaux permet de trouver d'autres sites potentiellement intéressants sous deux conditions : connaître un premier site et que celui-ci fasse partie d'un anneau.

Les moteurs de recherche constituent une alternative plus polyvalente aux anneaux. Plus besoin de partir d'un site intéressant ; il suffit de connaître l'adresse d'un moteur de recherche et de lui soumettre sa requête. Ces outils maintiennent une liste des sites Web les plus intéressants et proposent un certain nombre de méthodes pour les interroger. Selon le niveau d'intervention humaine requis pour maintenir cette liste, il est possible de faire la distinction entre les moteurs de recherche entièrement automatisés (**Google**), semi-automatiques (**Yahoo**) ou entièrement manuelle (**Dmoz**[13]). Le dernier annuaire cité présente la particularité d'être entièrement saisi manuellement, par des bénévoles, sur une idée similaire à celle de l'encyclopédie **Wikipedia**[14]. A ces premiers exemples de moteurs et annuaires dit « généralistes » viennent s'ajouter des sites spécialisés. Ceux-ci, au lieu d'accumuler des adresses de sites traitant de sujets divers, sont axés sur un sujet particulier. Le site **Citeseer**[15] en est un exemple appliqué à l'indexation de la littérature scientifique en ligne.

Afin de faciliter la consultation de différents moteurs, l'utilisateur peut faire appel à un métamoteur de recherche. Au lieu de n'utiliser qu'une base de données de liens qui leur est propre, ces outils vont interroger un ensemble de moteurs de recherche et effectuer une synthèse des informations obtenues. La plupart des méta-moteurs (**Dogpile**[16], **Metacrawler**[17], **WebCrawler**[18], ...) présentent leurs résultats à l'utilisateur sous la forme classique d'une liste de liens (voir figure 1.7).

Kartoo[19], un métamoteur édité par *Kartoo Technologies*, constitue quant à lui une carte afin de représenter les résultats de la recherche (voir figure 1.8). Chaque feuille représente un site Internet, la taille du pictogramme permet d'avoir une idée de l'importance du site. Entre ces feuilles sont représentés des halos avec en leur centre un mot clé. Ce mot est celui qui est le

[12] `dir.Webring.com`, dernier accès le 7 octobre 2006
[13] `www.aef-dmoz.org`, dernier accès le 7 octobre 2006
[14] `wikipedia.org`, dernier accès le 7 octobre 2006
[15] `www.citeseer.com`, dernier accès le 7 octobre 2006
[16] `www.dogpile.com`, dernier accès le 7 octobre 2006
[17] `www.metacrawler.com`, dernier accès le 7 octobre 2006
[18] `www.webcrawler.com`, dernier accès le 7 octobre 2006
[19] `www.kartoo.com`, dernier accès le 7 octobre 2006

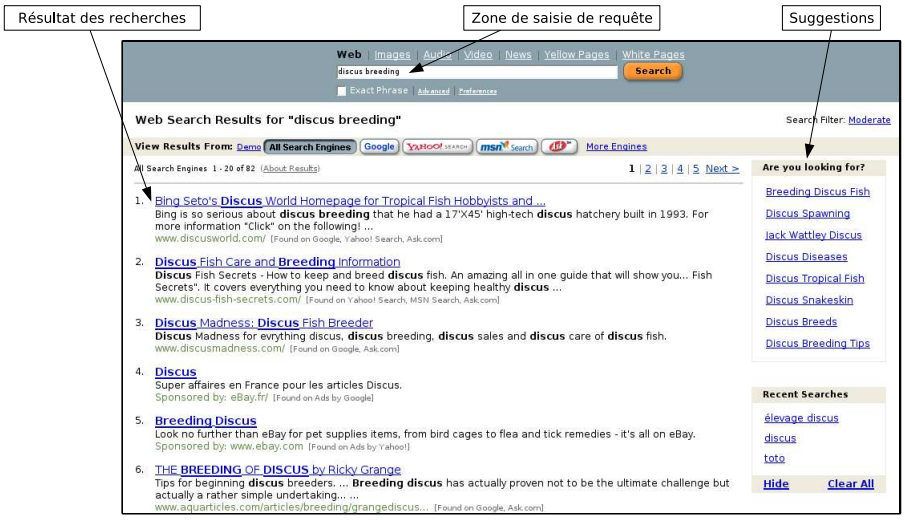


FIG. 1.7 – Capture d'écran du métamoteur de recherche « Webcrawler »

plus présent dans les pages concernés. Sur la capture d'écran de la figure 1.8, on peut remarquer que « eau » et « poissons » sont deux mots clés obtenus lors de la recherche d'informations sur l'élevage de « discus »[20]. Cette représentation permet de mettre en évidence une information qui n'est que sous-jacente lors de l'utilisation de listes.

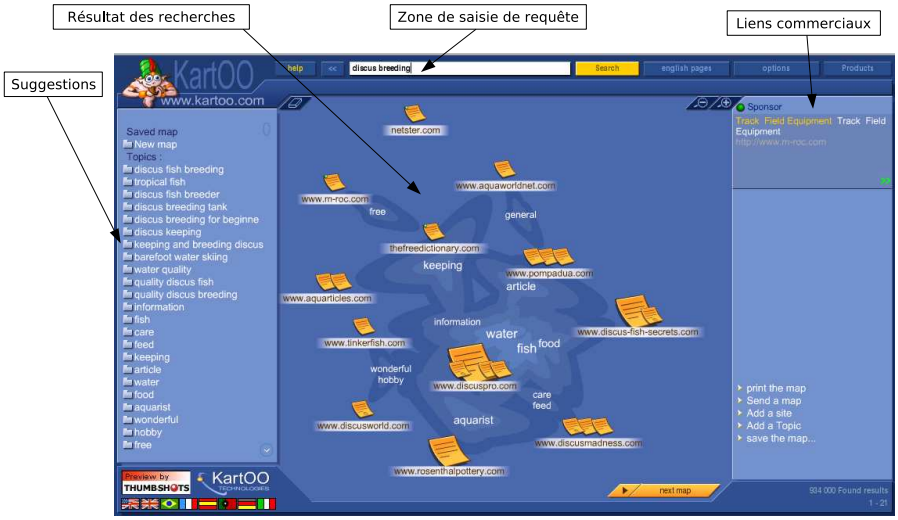


FIG. 1.8 – Capture d'écran du métamoteur de recherche « Kartoo »

Il est intéressant de remarquer que les deux moteurs présentés proposent des fonctionnalités d'affinage de requête. En voyant la liste des requêtes passées les plus proches de celles qu'il vient de formuler, l'utilisateur peut penser à de nouveaux mots clés à utiliser et ainsi interroger le moteur de façon plus efficace.

[20]Poisson exotique d'eau douce

1.4.2.2 Filtrage d'informations depuis une ou plusieur sources

Après avoir trouvé un ensemble de sources d'informations, il convient de les consulter afin de trouver l'information recherchée. Nous pouvons par exemple faire ici le parallèle entre l'activité qui consiste à trouver un annuaire téléphonique et celle d'en parcourir les pages à la recherche d'un nom particulier.

La consultation des ressources peut être facilitée par l'utilisation d'une page de souscription s'il s'agit de sites Web le proposant. Cette page, formalisée dans un langage tel que le RSS 1.0 (*Rich Site Summary*), permet de résumer le contenu d'un site sous la forme de plusieurs « billets ». Chacun de ces billets correspond à une information disponible. Le flux RSS d'un journal en ligne, par exemple **Lemonde.fr**[21], permet de se maintenir au courant du contenu de la « une ». L'avantage des flux RSS est qu'ils peuvent être consultés en dehors du site de publication en utilisant des outils dédiés. La figure 1.9 est une capture d'écran d'un de ces outils.

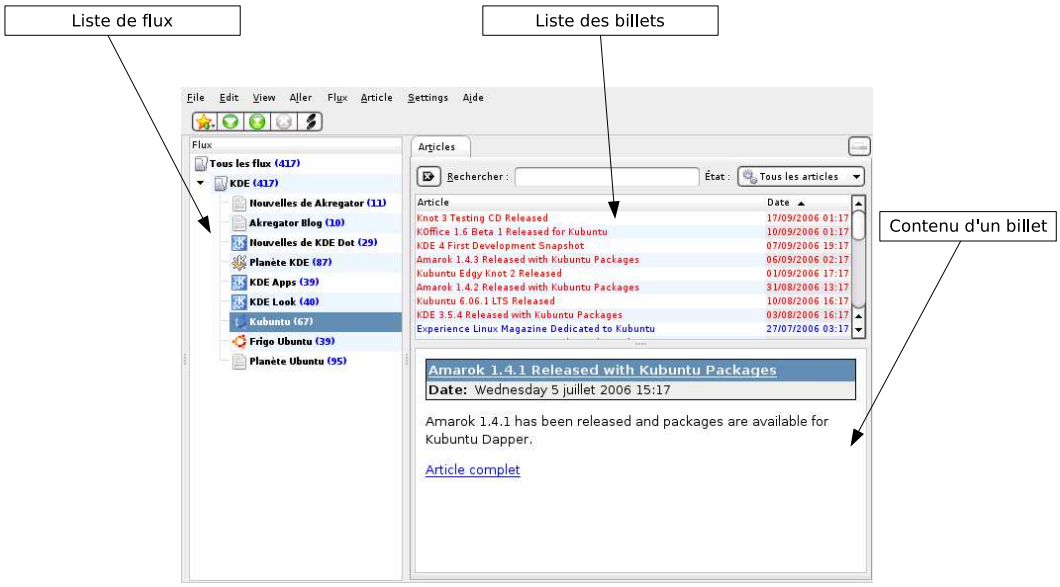


FIG. 1.9 – Capture d'écran du logiciel de lecture de flux RSS « Akregator »

Cependant, si le nombre de ressources est important, la consultation peut s'avérer fastidieuse pour l'utilisateur. Le logiciel **Amalthaea** [Moukas 96] développé par Moukas *et al.* est un exemple d'agent permettant l'automatisation de ce processus de recherche. Ce dernier fonctionne selon un modèle économique mettant en œuvre des agents de filtrage d'informations (IFA) et des agents de découverte d'informations (IDA). Les premiers ont pour tâche la consultation des ressources disponibles alors que les seconds effectuent l'interface avec l'utilisateur. Un IDA est associé à chaque ressource disponible et chaque IFA correspond à un centre d'intérêt de l'utilisateur. Selon le principe de l'offre et de la demande, ces IFA sont créés et détruits afin de s'adapter à la variation des besoins de l'utilisateur. Afin de différencier centres d'intérêt et besoins ponctuels, les IFA disposent d'une variable booléenne indiquant s'ils sont temporaires ou non.

L'idée de la différenciation entre fournisseurs d'informations et consommateurs se retrouve

[21] `www.lemonde.fr`, dernier accès le 7 octobre 2006

également dans le système **ImmuneSearch** [Ganguly 04, Di Caro 05]. Celui-ci se compose de 3 éléments : des profil d'informations (P_i), des profil de recherche (P_s) et des messages M. Il est supposé que, d'une manière générale, un utilisateur peut avoir des connaissances sur un domaine particulier (la recherche scientifique par exemple) et faire des recherches sur un autre domaine (le football). Chaque pair dispose donc d'un profil de recherche et un profil d'informations respectivement représentatifs de l'information cherchée et des connaissances. La circulation des messages dans le réseau utilise un modèle inspiré des systèmes immunitaires.

1.4.3 Aide à la gestion et à l'exploitation

Ces deux notions couvrent des besoins sur le court et le long terme. Nous présentons ici trois catégories d'outils répondant à ces besoins :

- **Mémorisation et récupération de contenu**
  Se souvenir d'une adresse d'un site et être capable d'y retourner est une tâche complexe pour un utilisateur. Il est notamment difficile de se rappeler d'un grand nombre d'adresses de site Web. Les outils d'aide à la mémorisation et à la récupération de contenu vont aider l'utilisateur dans cette tâche.
- **Appropriation du contenu**
  Contrairement aux documents papier, les documents éléctroniques ne peuvent faire l'objet d'une appropriation par l'utilisateur. Corner la page d'un livre ou y apposer des remarques manuscrites sont notamment deux activités altérées par la dématérialisation du support.
- **Modification de la page**
  Les pages Internet peuvent parfois être très chargées en éléments graphiques ou avoir une mise en page complexe. Ces caractéristiques peuvent devenir gênantes et rendre plus difficile la navigation pour des personnes peu habituées à utiliser l'outil informatique ou ayant des déficiences (visuelles, par exemple). Un assistant capable de modifier la page va pouvoir en changer le contenu afin de le rendre plus accessible.

1.4.3.1 Mémorisation d'adresses de sites Web

Outre le fait de se sentir perdu lors de la navigation sur Internet, les utilisateurs éprouvent également des problèmes quant à la mémorisation des pages consultées. Retrouver un site que l'on avait apprécié mais dont l'adresse n'a pas été notée est une tâche parfois complexe.

Une première solution est disponible sur tous les navigateurs Internet. Il s'agit de signets (« bookmarks ») permettant de stocker l'adresse du site dans une barre d'outils prévue à cet effet. Pour noter l'adresse d'un site, un signet est enregistré associant l'URL (Uniform Ressource Location) et une description. Ces signets peuvent être ensuite organisés en arborescence et utilisés à la fois comme méthode d'accès rapide aux sites et comme catalogue de sites appréciés. Bien que ce système soit très répandu il présente bon nombre de limitations :

- pas de contexte : un signet ne permet pas de stocker l'information du contexte de recherche ayant conduit à sauvegarder l'adresse du site. L'utilisateur ne peut retrouver un site dans ses signets par association d'idées ;
- pas de méta information : seule une description basique du site peut être mémorisée.

Une seconde solution est l'utilisation de l'historique, également présent sur tous les navigateurs. Il se présente sous la forme d'une liste enregistrant toutes les pages affichées par le navigateur. Il peut être classé par ordre chronologique et/ou par provenance. Cet outil est quelque peu basique et comparable aux signets en terme de fonctionnalités : il ne permet pas de gérer de méta-données mais propose une notion de contexte (limitée). Alors qu'il n'est pas possible de lier entre elles deux entrées de l'historique, il est néanmoins possible de voir le contexte de navigation en utilisant un tri chronologique.

Le rappel d'adresses de sites déjà visités se place dans le cadre de la problématique de la gestion de l'information acquise. Alors que, tel que présentés aux chapitre précèdent, les outils passifs vont permettre à l'utilisateur de plus facilement stocker cette information, les outils actifs vont chercher à l'exploiter. L'exploitation la plus courante se fait par le biais d'agents se basant sur la nature associative de la mémoire humaine : les « agents de souvenir » (« Remembrance Agents ») [Rhodes 04]. En effet, alors que les ordinateurs peuvent lier deux documents en se basant sur des mots clés ou les noms des fichiers, le cerveau humain fonctionne par associations d'idées pour aller rechercher un souvenir dans la mémoire. C'est la consultation d'un document ou d'un groupe de documents qui déclenche un processus de rappel lié au contexte. Les agents de souvenir aident à ce déclenchement en affichant de manière continue et non intrusive un ensemble de documents connus en relation avec le document consulté. Les agents d'aide à la mémoire associative se démarquent des outils tels que WebWatcher et Letizia sur deux aspects : (1) le mode de suggestion se base sur un contexte de recherche plutôt que sur un profil de l'utilisateur et (2) la source de données utilisée est constituée d'un ensemble de documents issus de l'historique plutôt que des liens contenus dans la page consultée.

Les premiers travaux de Rhodes *et al.* [Rhodes 96] portent sur un agent s'intégrant dans l'éditeur de texte EMACS et composé de deux éléments. Le premier s'intègre dans l'éditeur et se charge de l'acquisition d'informations concernant le document en cours de saisie (mots clés) ainsi que de la présentation des documents rappelés. Le second, extérieur à l'éditeur, utilise les informations fournies par le premier élément pour effectuer ses recherches parmi les ressources qu'il est capable d'exploiter. Il peut ainsi trouver d'anciens messages électroniques, des fichiers de notes personnelles, et des documents en ligne.

Margin notes [Rhodes 00] transpose ce système aux navigateurs Web. Un serveur de proximité (« proxy ») est utilisé pour incruster dans chaque page Web consultée, un cadre noir contenant une liste de documents recommandés (voir figure 1.10). Ce projet a permis de tirer plusieurs conclusions concernant le rapport qu'entretiennent les utilisateurs vis à vis des systèmes de remembrement. La première implémentation du système bloquait l'affichage de la page en attendant que les calculs soient fait. Les délais d'attente engendrés n'ont pas été acceptés par les utilisateurs. Une seconde version générant les recommandations « à la volée », c'est-à-dire pendant que l'utilisateur consulte la page, a eu plus de succès. Il a été également démontré que même si une suggestion se trouve particulièrement pertinente, il n'est pas certain que l'utilisateur la prenne en compte. Pour être appréciés, ces systèmes doivent être non intrusifs et ne pas perturber le fonctionnement des applications dans laquelle ils sont embarqués.

Sur un principe similaire à Margin Notes, **Calvin** [Bauer 01] constitue un exemple de système pro-actif d'aide à la mémoire. Son architecture s'oriente autour de plusieurs éléments chargés de surveiller les activités de l'utilisateur et d'un agent émettant des suggestions. Dans le cadre de la gestion d'adresses de sites Web, un agent se greffant au navigateur Web indexe automatiquement

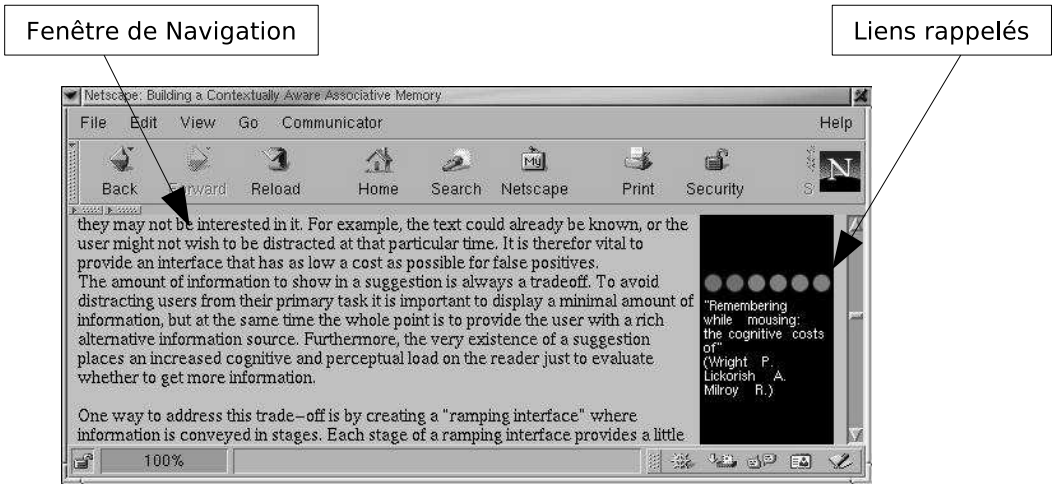


FIG. 1.10 – Exemple d'affichage de Margin Notes [Rhodes 00]

chaque page consultée alors que, parallèlement à cette indexation, d'autres pages sont indiquées à l'utilisateur. Cette limitation à la gestion d'adresses de sites rend le cadre d'application de Calvin plus limité que celui de Margin notes et le rapproche des outils de suggestions de liens tels qu'Alexa.

1.4.3.2 Appropriation du contenu

Le premier système d'annotations qui fut proposé était intégré dans le navigateur Mosaic développé par la NCSA [NCSA 97]. Il permettait à l'utilisateur de créer des annotations personnelles qui étaient stockées localement sur la machine. Ces annotations globales à une page et associées au navigateur pouvaient être utilisées comme aide-mémoire ou pour enregistrer des notes globales sur la page consultée. Bien qu'étant incluse comme une fonctionnalité « de base », cette annotation de page est ensuite tombée en désuétude auprès des éditeurs des navigateurs Internet qui ont succédé à Mosaic. Cependant l'idée des annotations comme aide-mémoire a ensuite été développée pour devenir une aide à la discussion entre utilisateurs.

CoNote permet d'ajouter des annotations à n'importe quelle page internet en respectant cependant des points d'ancrage particuliers définis soit par l'auteur du document soit par l'administrateur. La conception de CoNote a été faite en considérant qu'un système d'annotation peut jouer le rôle d'un forum et même agir plus efficacement dans bien des cas. En effet le document annoté produit un certain contexte aux discussions suivies au fur et à mesure des annotations, ce dont ne disposent pas les forums classiques ou les « newsgroup ». Les points d'ancrage des annotations étant prédéfinis, cela permet de fixer des points de départ de débat, chaque intervention étant une nouvelle annotation indentée afin de pouvoir suivre le fil de la discussion. Ce programme dispose également d'un outil de recherche afin de pouvoir parcourir un document à la recherche de nouvelles annotations. Il est construit comme un *proxy* se plaçant entre le navigateur internet et les serveurs envoyant les pages demandées. Ainsi les pages sont interceptées, les annotations insérées dans le code et le tout renvoyé à l'application cliente pour être affiché.

Dans la même lignée, **PageSeeder**[22] propose aux utilisateurs d'annoter les documents qui sont présents sur le serveur de Page. Les points d'ancrage de ces annotations sont fixes et définis au moment où le document est ajouté sur le serveur, il s'agit des fins de paragraphes. Outre la possibilité de distinguer différents types d'annotations comme par exemple « question », « réponse » et « discussion générale », le système gère l'utilisation d'un modérateur qui vérifie et accepte chaque annotation avant son ajout effectif sur le serveur et un système d'alertes par courrier électronique permet aux utilisateurs de se tenir au courant des nouvelles annotations sans avoir à revenir visiter la page. Les notes sont directement insérées dans la page HTML, à l'endroit où elles ont été spécifiées.

Comme l'ont mis en évidence les travaux effectués sur l'application CoNote, un système d'annotation de documents peut avantageusement remplacer les méthodes classiques de discussion asynchrone sur le réseau Internet. Cette analogie peut être poussée à l'extrême en concevant un système centré sur l'aspect d'outil de dialogue, faisant alors passer les annotations en tant que telles en second plan. Dans ce cadre, l'application **ComMentor** [Röscheisen 94] est construite de façon à pouvoir être utilisée en tant qu'outil de gestion de discussions, de tours guidés de documents ou encore d'indicateurs d'usages. Sa structure est organisée autour de trois composants : un serveur d'annotations, les serveurs Web contenant les pages visitées et le navigateur Internet.

Dans **CaMILE** [Guzdial 97], ce sont également à des points spécifiques que sont placées les annotations. Il ne s'agit pas cette fois d'un placement automatique selon la structure HTML du document mais d'un placement manuel. CaMILE ayant été développé dans un objectif de support de cours pour des discussions entre étudiants, ce sont les professeurs mettant en ligne le document qui y définissent également les points d'ancrage pour les départs de discussions. L'ajout et la lecture des notes se fait dans une fenêtre séparée, ce qui permet de les dissocier de la page elle-même.

Contrairement aux outils précédemment présentés, **DocReview** n'a pas pour but de permettre aux utilisateurs de placer des notes n'importe où dans le document mais de commenter des sections entières. Ainsi il est proposé à l'administrateur qui met en place le document sur le système d'annotation de découper son document en plusieurs « zones de relecture » de différentes tailles. Une fois ceci effectué, les utilisateurs disposent d'un bouton R et d'un bouton W disponibles au début de chaque zone. Ces deux boutons servent respectivement à consulter et à ajouter une annotation à la section. Ces actions s'effectuent dans une fenêtre à part de celle contenant le document. Le système affiche également un résumé indiquant le nombre de commentaires effectués sur le document, ce qui permet d'avoir rapidement un aperçu de l'état de l'activité de relecture.

Enfin, le navigateur **Amaya** développé par le World Wide Web Consortium (W3C) afin de valider toutes les dernières recommandations émises par ce même organisme vient récemment d'être doté d'un système d'annotation de pages HTML : **Annotea**. Cet outil est développé dans l'optique d'utiliser les dernières technologies en matière de gestion de liens et de méta-données, ce qui constitue sa principale différence avec les nombreux autres outils d'annotation existants.

[22] `http://ps.pageseeder.com/ps/topic/ps`, dernier accès le 23 octobre 2006

1.4.3.3 Modification de la page

Indépendamment de l'utilisateur qui le consulte, le Web offre la même apparence. Un moyen d'aider l'utilisateur est d'y ajouter une couche de personnalisation. Dans [Barrett 97], Barett et Maglio proposent un système d'aide dénommé **WBI** (Web Browser Intelligence). Il s'agit d'un système de personnalisation de pages se situant entre le navigateur Internet de l'utilisateur et les serveurs Web (*proxy*). Ce positionnement lui permet de modifier dynamiquement le contenu des pages et de proposer des fonctionnalités telles que le sur-lignage de mots ou l'ajout d'informations supplémentaire. Un résultat intéressant de ce projet est la mise en évidence et l'application pratique d'un guide de comportement pour agent assistant. Ce guide se résume en cinq points :

1. l'agent doit maintenir l'interface Web que l'utilisateur a l'habitude d'utiliser. Ce dernier doit intéragir principalement avec le Web et non avec l'agent ;
2. l'agent doit pouvoir agir de manière pro-active et réactive. C'est-à-dire qu'il doit pouvoir témoigner d'un comportement actif de conseil quand la situation le requiert tout en répondant aux demandes de l'utilisateur ;
3. l'agent ne doit pas se supplanter à l'utilisateur. Il ne peut que fournir un avis que l'utilisateur est libre de suivre ou non ;
4. l'activité de l'agent doit être périphérique à la tâche de l'utilisateur de sorte que ce dernier puisse ignorer la présence de l'agent s'il le désire ;
5. l'agent doit généraliser, reconnaître et classer les activités de l'utilisateur pour des utilisations futures. L'identification de ces activités ainsi que de leur fréquence permet de mieux cibler les aides à fournir.

Cela peut se généraliser en deux points :

- ne pas perturber l'utilisateur : l'outil doit se fondre dans l'environnement de l'utilisateur et ne pas exiger d'effort supplémentaire de sa part, ni le déranger dans ses tâches habituelles ;
- comprendre, analyser et conseiller : l'outil doit pouvoir témoigner d'un niveau d'analyse minimale.

Une approche plus complexe est proposée dans le projet **I-KnowUMine** [Vlachakis 03] présenté par ses auteurs comme étant une plate-forme de personnalisation du Web basée sur la structure du contenu et les comportements de l'utilisateur. Plus spécifiquement, le système combine des connaissances relatives au comportement de l'utilisateur et à la sémantique des documents consultés pour produire un flux d'informations optimal. Plusieurs domaines de recherche sont touchés par ce projet : *Web mining*, gestion de contenu, personnalisation et portails. Pour les fédérer, Vlachakis *et al.* définissent un serveur d'espace de contextualisation de contenu qui :

- propose des composants d'édition et de publication de contenu ;
- utilise des techniques de fouille du Web (« Web Mining ») pour la collection de données, leur pré-traitement (« Preprocessing »), leur analyse ainsi que la génération de connaissances. Tous ces processus sont menés en vu de l'utilisation pour la personnalisation ;
- met en œuvre des outils de personnalisation pour utiliser les connaissances générées et émettre des recommandations en fonction du comportement de l'utilisateur ;
- fournit un portail de déploiement de contenu contextuel.

L'un des objectifs de ce système est de proposer une représentation uniforme à la fois du contenu et de l'utilisation qui en est faite. Pour se faire, le système est décomposé en deux parties : une en ligne (*online*), et la seconde hors ligne (*offline*) (voir figure 1.11). Une fois installé sur

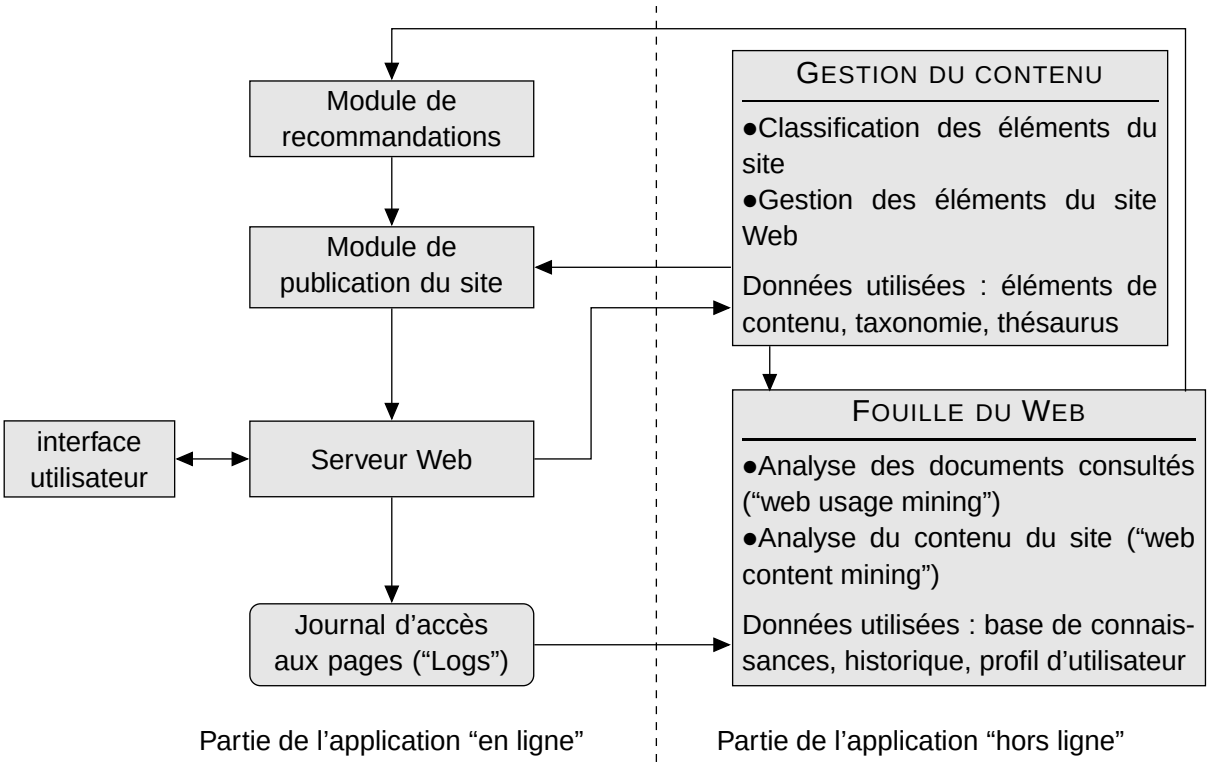


FIG. 1.11 – Représentation simplifiée de l'architecture **I-KnowUMine**. D'après [Vlachakis 03]

un serveur Web, ce système permet de fournir des recommandations à l'utilisateur. Il s'agit de recommandations concernant les liens à suivre dans le site concerné.

D'autres outils permettent de linéariser le contenu de la page afin de rendre les sites accessibles. C'est le cas par exemple d'**AccessiBar**. Un greffon pour navigateur Web permettant, entre autre fonctionnalité, de supprimer la mise en page d'un site afin d'en proposer le contenu sous la forme d'un texte uniforme (pas de colonnes, ...).

1.4.4 Aide à la communication

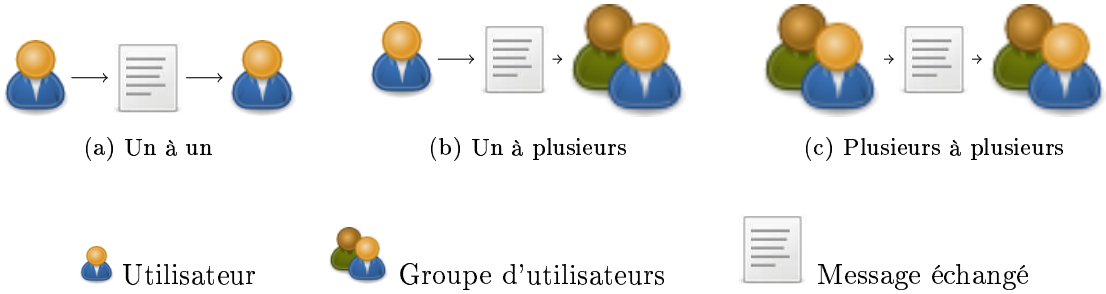
La communication d'informations est un processus impliquant deux catégories d'utilisateurs : les fournisseurs et les demandeurs. Le fournisseur donne aux clients les informations qui lui sont demandées. Selon le nombre de représentants de chaque catégorie, il est possible de distinguer différents schémas de communication pour ces échanges :

- **Un à un (*one to one*)**
  Ce schéma se trouve dans le cas où seuls un serveur et un client sont impliqués. Il s'agit, par exemple, de la situation où un utilisateur pose une question à un autre utilisateur. Ceci correspond à un schéma réseau dit de « client / serveur ». L'envoi des messages y est fait en « unicast » (un seul destinataire).
- **Un à plusieurs (*one to many*)**

Il se peut également que le fournisseur ait besoin d'envoyer une information à plusieurs demandeurs à la fois. Dans ce cas il s'agit d'un modèle client/serveur utilisant un mécanisme de diffusion « multicast » (à plusieurs destinataires). Si l'information doit être diffusée sur l'ensemble du réseau, le message est envoyé en « broadcast ».

- **Plusieurs à plusieurs (*many to many*)**
  Si tous les utilisateurs peuvent envoyer des informations à tous les autres, chacun d'entre eux est à la fois fournisseur de demandeur. Ce schéma de communication correspond au cient/serveur avec pour particularité que chaque participant est client et serveur en même temps.

A chacun de ces types d'échanges correspond un outil privilégié. Nous en présentons ci-après quelques uns représentatifs de la fonctionnalité visée. Ces outils se caractérisent également par la nature du canal de communication utilisé (voir figure 1.12).



(a) Un à un (b) Un à plusieurs (c) Plusieurs à plusieurs

FIG. 1.12 – Illustration des schémas de communication

1.4.4.1 Communications de type « un à un » (*one to one*)

Ce premier schéma, simpliste, met en œuvre un unique détenteur d'informations et un unique demandeur. Un des outils entré dans les habitudes des Internautes et illustrant ce mode de fonctionnement est la messagerie éléctronique. S'il désire faire passer une information, l'utilisateur a juste besoin de connaître l'adresse de courriel du destinataire voulu. Il en va de même lorsque la démarche est celle d'une demande d'informations. Si, pour un courriel donné, plusieurs destinataires sont concernés, ce processus est instancié autant de fois que nécessaire.

La principale difficulté pour un utilisateur réside dans le choix du ou des destinataires de ce courrier à envoyer. Il lui faut choisir parmi ses connaissances les personnes qui seront intéressées par le message et, si ce dernier est de nature confidentielle, censés le recevoir. En supposant que la taille de l'ensemble des connaissances est relativement petit, il est facile de retenir les adresses et centres d'intérêt de chacun. Pour un ensemble plus conséquent, la tâche peut s'avérer complexe voire impossible. Le « carnet d'adresse » permet de stocker les noms et adresses des contacts. La majorité de ces carnets permettent également de saisir d'autres information telles que la date d'anniversaire du contact, son pseudonyme ou encore ses centres d'intérêt. Mais ce type de support reste particulièrement adapté aux dialogues entre utilisateurs ayant une connaissance globale des centres d'intérêt respectifs de chacun des membres de leur entourage. Ces outils montrent leur faiblesse dès que cette connaissance est incomplète ou défaillante. Par exemple, le problème d'oubli de destinataires souvent remarqué dans l'utilisation courante de la messagerie électronique. Soit les envois sont ciblés au risque d'oublier certaines personnes, soit ils sont trop larges au risque d'en importuner d'autres avec des messages qui ne les intéressent pas.

1.4.4.2 Communications de type « un à plusieurs » (*one to many*)

Les échanges de type un à plusieurs concernent un utilisateur désirant dialoguer avec un groupe d'utilisateurs. Contrairement aux échanges de type un à un impliquant plusieurs utilisateurs individualisés, la notion de groupe permet de définir une entité virtuelle avec laquelle l'utilisateur va dialoguer. Un Internaute publiant un site Web va ainsi s'adresser au groupe des abonnés au réseau et non à l'abonné X, l'abonné Y, l'abonné Z, *etc.*

Pour supporter ce type d'échanges, il convient de mettre en place un support de communication commun auquel tous les clients ont accès et d'où le serveur puisse poster ses messages. Les forums de discussions en sont un exemple. Il s'agit d'outils d'échange de messages avec suivi de fil de discussion qui se présentent généralement sous la forme d'un site Web. Chaque utilisateur disposant des droits l'y autorisant peut lancer une nouvelle discussion (on parle d'un « post ») ou participer à une discussion existante. Chaque forum est axé sur un thème de discussion particulier, des « boards » permettent de rassembler plusieurs forums pour constituer un ensemble plus vaste.

Ce moyen de communication uniforme et ciblé représente l'inconvénient de devoir être localisé sur le Web pour être utilisé. En effet, les forums sont des sites comme les autres et l'utilisateur, s'il n'en connaît pas l'emplacement, doit le trouver. C'est paradoxalement un outil plus ancien que les forums qui permet de contourner ce problème. Il s'agit du réseau de discussions **UseNET**. Les forums y sont groupés selon un nommage hiérarchique dans un réseau unique consultable grâce à un protocole dédié (le NNTP - *Network News Transfer Protocol*). Par exemple, `comp.lang.php` est une sous-branche de la partie consacrée à l'informatique (`comp`) consacrée au langage (`lang`) de programmation `php`. Du point de vue collaboratif, les mêmes propriétés sont observées entre ces forums et ceux utilisant un support de type site Web. La différence essentielle tient d'une part au point de vue historique, les forums UseNet sont les plus anciens et donc les plus importants, et d'autre part du point de vue regroupement. Il est assez facile de trouver un forum sur un sujet précis en parcourant l'arbre de nommage. De plus, n'importe quel utilisateur peut créer une nouvelle branche s'il le désire.

Ces deux premiers exemples permettent de mettre en place des discussions de type un à plusieurs asynchrones. Le serveur y émet son message que les clients lisent ultérieurement, sans contrainte de délai. Des discussions synchrones peuvent être tenues dans un réseau de type IRC (Internet Relay Chat). Ce mode de discussion, utilisant un protocole dédié, permet à un ensemble d'utilisateurs de dialoguer en « temps réel ». Les groupes sont identifiés par des canaux de discussions nommés.

Plus récemment sont apparus des sites de support pour réseau social (*online social networking*) tels que **Orkut**[23], **Linkedin**[24] ou **Friendster**[25]. Le principe de ce site est de constituer un réseau social basé sur une relation de connaissance. Pour obtenir un compte sur ce site il est nécessaire de recevoir une invitation de la part d'un membre déjà inscrit. Une fois cette invitation acceptée, il est possible de constituer son réseau social en ajoutant des amis membres du réseau ou en en invitant de nouveaux. Les comptes dont disposent chaque utilisateur permettent de renseigner un certain nombre d'informations personnelles tels que les centres d'intérêt, le statut marital, *etc.* D'un point de vue collaboratif et échange d'informations, ce site propose une liste

[23] `www.orkut.com`, dernier accès le 20 Septembre 2006
[24] `www.linkedin.com`, dernier accès le 20 Septembre 2006
[25] `www.friendster.com`, dernier accès le 20 Septembre 2006

de communautés classées par thème. Un utilisateur joignant une communauté y trouvera, outre la liste des membres, un forum ainsi qu'un outil d'échange de fichiers. Ces sites présentent deux principales limitations : (1) pour bénéficier de l'information de liens entre les personnes et trouver d'autres utilisateurs, il est nécessaire que cette information ait été saisie et rendue accessible. (2) L'utilisation d'un serveur pose les problèmes de la fiabilité et de la montée en charge ainsi que celui de la confidentialité des informations.

1.4.4.3 Communications de type « plusieurs à plusieurs » (*many to many*)

Dans ce dernier schéma de communication, un groupe d'utilisateurs s'adresse à un autre groupe d'utilisateurs. A la différence des deux précédents schémas où un seul utilisateur était à l'origine de l'information échangée, cette fois c'est un groupe dans son ensemble qui émet le message. Pour ces structures existent plusieurs outils généralement identifiés en tant que « Web based community systems », ces outils de support de communauté sont basés sur l'utilisation du Web.

Ce schéma de communications peut être mis en parallèle avec les environnements dirigés par un principe de stigmergie [Parunak 05]. Ce principe désigne un échange d'informations indirecte entre agents basé sur une modification de leur environnement. Chacun dispose de son propre état et dynamique de fonctionnement et est à même d'évoluer indépendamment (figure 1.13).

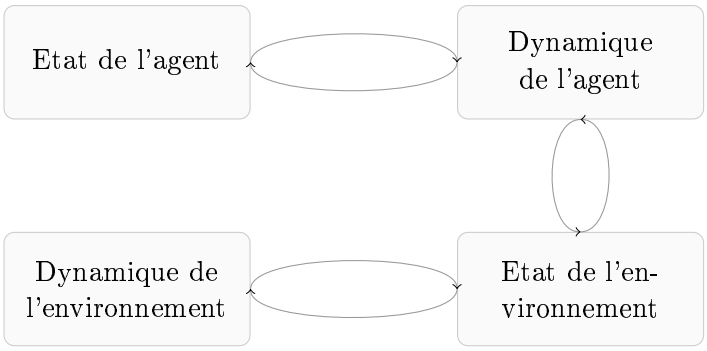


FIG. 1.13 – Architecture d'un système stigmergique (adapté de [Parunak 05])

Dans notre cas, les utilisateurs sont vus comme des agents évoluant dans un environnement qui est leur cadre de travail : c'est en modifiant cet environnement que la communication est mise en place. Afin d'illustrer notre propos, prenons comme exemple l'encyclopédie libre **Wikipedia**[26]. Le site Web de cette encyclopédie est bâtit sur un système de « Wiki » dont le principe de fonctionnement est de permettre à chaque visiteur de modifier le contenu des pages affichées. Chacun est libre d'enrichir le contenu en modifiant un article ou en en créant de nouveaux. L'ensemble de ces pages constitue l'environnement dans lequel le groupe des visiteurs fait part de ses connaissances.

Sur un principe plus large, les gestionnaires de contenu (« **C**content **M**anagment **S**system - CMS ») permettent la mise en place et la gestion de bases de documents partagées. Ces outils ont pour but le stockage et l'indexation centralisée de fichiers électroniques. Par l'intermédiaire d'une interface Web, les utilisateurs peuvent déposer et consulter des documents. Certains de ces

[26] `www.wikipedia.org`, dernier accès le 20 Septembre 2006

outils proposent également des modules permettant l'intégration de pages Wiki. Le nombre de projets de CMS existants témoignent de leur succès, le lecteur est invité à se rendre sur le site OpenSourceCMS[27] pour avoir un aperçu des solutions logicielles existantes.

1.4.4.4 Communications indirectes

Au lieu de considérer un environnement spécifique pour l'entraide comme dans le cas des bases de documentation, il peut être intéressant de se baser sur l'environnement constitué par les programmes dont se servent les utilisateurs. C'est dans cette optique que s'intègre les travaux sur la collaboration indirecte.

Les premiers travaux sur la collaboration indirecte ont été réalisés par Payton [Payton 98]. Dans les formes classiques de collaboration, les participants s'entraident par le biais de moyens de communication directs (courriels, conversation) selon un but prédéfini (informer, questionner). A l'opposé, la collaboration indirecte est axée sur la capture d'informations provenant des activités courantes des participants. Cette capture passive permet d'éviter une surcharge de travail à l'utilisateur. Sur un principe similaire à ceux de Webxelblat et Footprints, le système présenté enregistre le parcours effectué par chaque utilisateur dans un site Internet. La principale différence par rapport à ces outils se situe au niveau de l'utilisation de chemins enregistrés : plutôt que d'exploiter l'information que fournit ce chemin sur l'espace parcouru, Payton se concentre sur l'information concernant celui faisant le parcours. Dans un espace de données, il est supposé que le parcours effectué par un utilisateur est caractéristique de ses centres d'intérêt. Deux utilisateurs effectuant un parcours similaire, sont donc mis en relation.

Une autre forme de collaboration indirecte bien connue est celle qui consiste à « écouter aux portes ». Prêter une oreille attentive à une discussion auquel il ne participe pas peut permettre à un utilisateur (mal élevé, certes) d'acquérir un certain nombre d'informations. L'efficacité de la manœuvre est variable. Il peut tout aussi bien s'agir d'informations qu'il n'aurait pas dû obtenir, d'autres qu'il aurait dû savoir mais qui ne lui ont pas été communiquées pour cause d'oubli ou encore d'autres dont il disposait déjà. Une des premières applications de ce principe se trouve dans les travaux de Busetta *et al.* [Busetta 01] : en considérant un réseau où des services échangent des messages, ils proposent l'ajout d'un agent scrutant les canaux de communication (voir figure 1.14). Ces informations sont utilisées pour fournir des recommandations aux services concernés. Par exemple, choisir une d'effectuer une certaine action parmi un ensemble d'actions possibles. Mais également changer l'état définit par le modèle comportemental pour le faire coïncider avec les changement dans les d'opinions ou d'intentions perçus dans les messages écoutés.

L'un des principaux problèmes des agents d'écoute est qu'ils se placent au milieu d'un dialogue dont ils ignorent le contexte ; l'identification de la conversation est donc une tâche complexe. De plus, le moyen d'écoute n'étant pas optimal, le message peut être bruité et mal interprété. Des travaux récents [Gutnik 04] ont conduit à une définition plus formelle des systèmes d'écoute quantifiant ces problèmes. Si l'on considère une séquence correspondant à une discussion et son observation, il est possible d'avoir : de la perte (la discussion n'a pu être entendue dans son intégralité), de l'insertion (des fragments d'une autre discussion sont venus s'intercaler dans celle

[27] `www.opensourcecms.com`, dernier accès le 7 octobre 2006

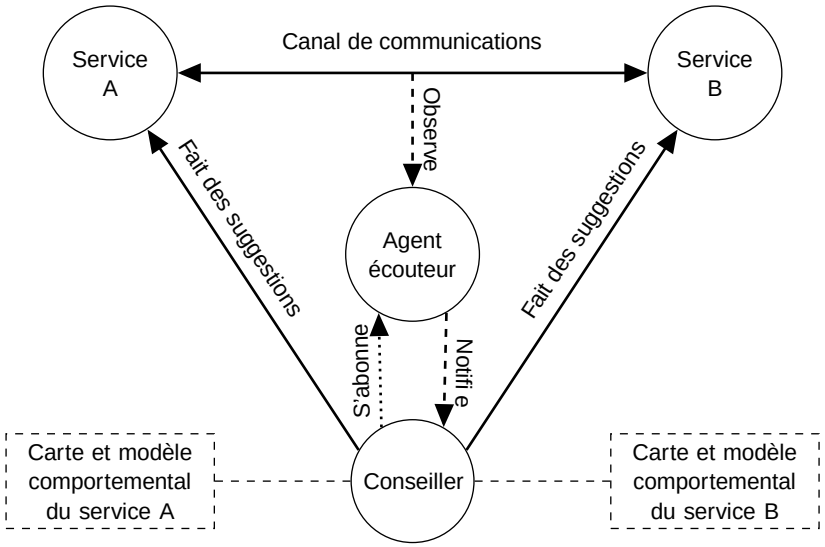


FIG. 1.14 – Système d'écoute d'un dialogue entre deux services. Le conseiller s'abonne auprès de l'agent d'écoute chargé d'observer les messages echangés. Grâce aux informations qu'il récupère, le conseiller peut ainsi suggérer des actions aux services. D'après [Busetta 01]

écoutée), un souci d'ordre (difficultés pour retrouver le « fil » de la discussion). Cette modélisation leur a permis de conclure que dans le cas où ces pertes sont observées, un algorithme de communication naïf ignorant les possibles altérations dans les messages perçus se révèle être le plus efficace. Il est donc préférable d'ignorer les pertes dans les messages plutôt que de chercher à les prendre en compte.

L'écoute peut aussi directement être faite au niveau des utilisateurs et non sur le canal d'écoute. Il s'agit alors d'interfaces à saisie nulle (Zero input) composées de capteurs enregistrant les activités des utilisateurs. Des programmes utilisant de tels capteurs peuvent ainsi communiquer entre eux sans nécessiter l'intervention de l'utilisateur. Le système **Collaborator** est un exemple d'application de ce principe pour la gestion d'espace de travail partagé. Dans ce cadre, un certain nombre de ressources communes sont partagées par les utilisateurs. Parmi celles-ci peuvent par exemple se trouver un emploi du temps indiquant les occupations de chacun ou une liste de projets en cours. Le but de l'outil d'aide va être de favoriser la collaboration entre membres d'une entreprise et faciliter la gestion des tâches de chacun. Collaborator prend en compte plusieurs contraintes du travail collaboratif :

1. indépendance de la plate-forme et Intégration dans le Web : Collaborator est basé sur les standards du Web (Java, HTML, TCP/IP, ...) et est indépendant du système ainsi que du réseau ;
2. ubiquité de l'accès : Collaborator peut être utilisé aussi bien depuis les ordinateurs de bureau que depuis un terminal mobile ;
3. capacité d'adaptation à la bande passante du réseau et aux capacités du terminal : Collaborator peut s'adapter selon le terminal depuis lequel il est utilisé ;
4. intéraction flexible avec l'utilisateur grâce à des agents : un agent personnel est associé aux utilisateurs et leur propose des informations personnalisées selon un profil.

Ces contraintes sont intégrées *via* un système à base d'agents personnels chargé de surveiller les informations circulant dans le réseau. Ce sont ces agents qui tiennent les conversations en lieu

et place des utilisateurs.

Il est important de noter qu'hormis les aspects techniques de l'acquisition passive de données, les systèmes de collaboration indirecte présentent des problèmes concernant la confidentialité. Il convient de trouver un juste milieu entre la totale sécurité des informations personnelles et le partage nécessaire au fonctionnement collaboratif du système.

1.5 Discussion

1.5.1 Critères de comparaison

Nous proposons de comparer ces différentes aides selon les cinq critères suivants :

1. le niveau **d'intégration** situe l'outil d'aide par rapport au navigateur. La question est de savoir si l'aide est intégrée au navigateur ou est externe. Il s'agit d'un critère plus lié à la technique qu'à l'aspect fonctionnel de l'outil ;
2. le niveau **d'autonomie** d'un assistant indique s'il s'agit d'une aide fournie spontanément ou non ;
3. le **délai de réaction** définit si l'aide est synchrone ou asynchrone ;
4. **l'adaptabilité** définit la capacité d'un outil à prendre en compte les informations en retour (« *feed-back* ») de l'utilisateur. Les moteurs de recherche proposant un système d'aide à l'affinement de requête sont un exemple de système adaptatif ;
5. **la personnalisation** de l'aide fournie permet d'adapter chaque résultat en fonction de l'utilisateur.

Le choix de ces critères est discutable. En effet, les travaux menés par B. Trousse sur l'étude des assistants à la navigation offre un autre point de vue, plus global [Trousse 99]. Selon son modèle, les systèmes d'aide à la navigation peuvent être classés d'un point de vue fonctionnel selon 3 critères : la nature de l'assistance fournie, la personnalisation et l'adaptabilité. L'aide fournie pouvant aller du masquage de liens à l'annotation des pages consultées. Cependant nous pensons que dans l'optique d'une définition de l'intelligence de l'aide fournie il est nécessaire d'extraire de l'analyse fonctionnelle les aspects de personnalisation et d'adaptabilité. Ainsi, le remplacement du critère concernant la nature de l'assistance fournie par les critères de réactivité et d'intégration nous permet, non plus de considérer *quelle* aide est proposée mais *comment* elle est apportée.

1.5.1.1 Niveau d'intégration

Typiquement, une interaction basique se fait entre le navigateur Internet et le serveur de pages Internet chargé d'émettre le contenu des pages à afficher. Sur requête de l'utilisateur, le navigateur demande au serveur le code HTML de la page à afficher puis reçoit et interprète la réponse pour dessiner le contenu de la page. L'arrivée d'un programme d'aide fait entrer un troisième acteur : l'assistant. Selon la relation qui existe entre ce dernier, le navigateur et les serveurs de pages, trois niveaux d'intégration peuvent être considérés :

- **Externe**

  L'assistant est associé avec le serveur de pages Web. Dans ce cas, le navigateur Internet n'est pas modifié et continue de jouer son rôle d'afficheur de pages. L'aide est mise en place au niveau du serveur de page qui avant d'envoyer sa réponse ajoute les informations fournies par l'assistant. Le site de vente en ligne Amazon.com implémente ce type d'aide dans leur outil de suggestion d'articles à acheter.

- **Interne**

  L'assistant est « dans » le navigateur Internet. Il peut aussi bien s'agir d'un outil présent dans le navigateur tel que la gestion de signets ou de greffons qui viennent s'ajouter par la suite. Les aides Internes agissent sur le contenu de la page une fois celle-ci fournie par le serveur de pages.

- **Serveur de proximité (Proxy)**

  L'assistant se place entre le navigateur et le serveur de pages. Sans changer le fonctionnement du navigateur, l'assistant va le plus souvent se charger d'intercepter les demandes de pages effectuées par ce dernier. Une fois une demande interceptée, celui-ci récupère le code de la page à afficher, y intègre les informations nécessaires et renvoie le nouveau code au navigateur Internet.

La différence d'intégration influe le plus souvent sur les capacités de l'outil. En étant intégré au serveur, ce dernier à la possibilité de modifier le contenu des pages avant leur envoi et a à sa disposition des informations relatives aux consultations de tous les utilisateurs visitant un site Web donné sur ce serveur. A l'opposé, un outil intégré dans le navigateur pourra plus facilement prendre en compte le comportement de l'utilisateur. La solution intermédiaire (*proxy*) permet quant à elle de profiter de l'avantage de l'une et de l'autre des solutions mais peut présenter des risques de ralentissement du réseau du fait de sa position (goulot d'étranglement).

1.5.1.2 Niveau d'autonomie

Le niveau d'autonomie d'un assistant peut se définir par sa capacité à effectuer sa tâche sans nécessiter l'intervention de l'utilisateur. Il est possible de distinguer deux comportements : actifs et réactifs, qui donnent lieu à deux types d'aides différents :

- les outils fournissant une **aide passive** vont aider l'utilisateur en jouant le rôle de support technique. En sortant du cadre informatique pour se placer dans celui de la vie courante, les indispensables bloc-notes, agenda (papier !) et autres « post-it » constituent un ensemble d'outils passifs aidant à la gestion d'informations (notes, rendez-vous, ...). La passivité de ces outils tient au manque d'intéraction et d'autonomie de décision. Un agenda papier ne se chargera pas de rappeler les rendez-vous pas plus que le post-it n'ira tout seul décider de se coller sur un endroit bien en vue. C'est à l'utilisateur de savoir quelles informations sont enregistrées via ces outils ainsi que de s'en assurer les accès ;
- les outils d'**aide active** sont quant à eux dotés de facultés d'interaction directe avec l'utilisateur. Les logiciels d'agenda éléctronique capables de rappeler par une alarme un rendez vous important sont un exemple d'aide active. Cette aide active requiert un certain niveau d'autonomie et de réflexion de la part du programme. Un des principes de développement répondant à ces attentes est celui des agents intelligents. Des études ont permis de mettre en évidence leur avantage dans ce domaine [Bergenti 04].

Conformément à cette description et selon la définition de Bouvin et al [Bouvin 99], les solutions fournies par les outils d'aide passive peuvent être qualifiés de Web « augmenté » :

> *" A tool shall be considered a Web hypermédia augmentation tool, if it through integration with a Web browser, a HTTP proxy or a Web server adds content or controls not contained within the Web pages themselves to the effect of allowing structure to be added to the Web page directly or indirectly, or to navigate such structure "*

Grâce à des interfaces de saisie, ces outils permettent à l'utilisateur d'ajouter de l'information aux pages consultées. Cette information constitue une structure de données venant se superposer à celle du World Wide Web. L'exemple le plus courant, et le plus simple, est celui des signets proposés par les navigateurs Internet. En offrant à l'utilisateur la possibilité d'enregistrer des adresses de sites, cet outil met en place une structure de portail composée de divers liens. Plus rarement, d'autres structures permettent la personnalisation des documents par l'ajout de notes ou la construction de cartes de navigation.

1.5.1.3 Délai de réaction

En se plaçant d'un point de vue communication, l'aide délivrée constitue une réponse à un message de demande formulé par l'utilisateur. En cas de réponse immédiate, le système d'aide sera dit synchrone. Dans le cas contraire, on parle de système asynchrone. Parmi les exemples de systèmes d'aide synchrone on peut citer les moteurs de recherche qui, si on le souhaite, fournissent une réponse immédiate à une demande formulée sous forme de mots clés. Le courrier électronique constitue lui un support d'aide asynchrone car au moment où est envoyé une demande (un message contenant une question), rien ne permet de présager de la date à laquelle la réponse sera reçue. Dans le cadre de notre étude, les systèmes d'aide à la recherche ainsi qu'à la gestion d'informations seront essentiellement de nature synchrone car l'objectif est de répondre à un besoin précis et ponctuel. Des réponses asynchrones seront, par exemple, plus facilement acceptées dans le cadre de la navigation.

1.5.1.4 L'adaptabilité

L'adaptabilité est un critère relatif à l'interaction entre le système fournissant l'aide et l'utilisateur. Un outil adaptatif va être en mesure de prendre en compte la satisfaction de l'utilisateur et ajuster son aide en fonction de ce paramètre. Pour un état de l'art des systèmes adaptatifs en 2004, le lecteur pourra se référer à [Sadat 04].

1.5.1.5 La personnalisation

Omero et Tasso [Mizzaro 02] définissent **la personnalisation** du Web comme le procédé de sélection, de préparation et de délivrance d'un contenu pour un utilisateur donné en prenant en compte ses besoins spécifiques et ses préférences. Ce qui signifie fournir le contenu le plus approprié de la meilleure façon et au meilleur moment (du point de vue de la durée des besoins). Selon eux, deux types de personnalisation peuvent être mises en évidence :

- **persistante** (long terme) basée sur un profil d'utilisateur, c'est-à-dire sur une description de ses intérêts à long terme ;
- **éphémère** (court terme) qui est basée sur des informations qui sont instantanément collectées. En étant de fait plus adapté aux besoins ponctuels, ce type de personnalisation permet de plus efficacement aider l'utilisateur dans le cadre de ses recherches.

Ce second type de personnalisation basé sur un retour d'informations ponctuel peut être assimilé au processus d'adaptabilité présenté précédemment. Une définition plus large est proposée par Vlachakis *et al.* [Vlachakis 03] qui décrit la personnalisation comme faisant partie de l'espace des « serveurs de contextualisation de contenu ». Un tel serveur est un « logiciel qui délivre du contenu supposé intéressant pour l'utilisateur selon un contexte donné en prenant en compte le comportement de l'utilisateur ainsi que ses préférences sémantiques telles que définies par ses accès au contenu passés et courants ».

1.5.2 Appréciation de l'intelligence d'un assistant

Ce sont donc cinq critères que nous proposons afin de caractériser les différents assistants que nous présenterons plus en détail dans les chapitres suivants. Ces critères permettent de décrire les facultés des outils, mais afin d'en apprécier l'intelligence, certaines simplifications peuvent être apportées :

- le critère portant sur le niveau d'intégration de l'assistant permet de mettre en évidence les spécifités techniques de l'outil. Cependant, les différences entre Externe/Interne et Proxy, bien qu'existantes, n'auront que peu d'influence sur l'expérience ressentie par l'utilisateur. En se focalisant sur un aspect d'efficacité de l'aide vis à vis de l'utilisateur, ce critère peut donc être ignoré ;
- le niveau d'autonomie et le délai de réaction peuvent tous deux être regroupés sous un critère global d'« Activité ». Pour cela nous considérerons qu'un outil passif fournit essentiellement des réponses asynchrones alors que les réponses synchrones seront données par un outil actif ;
- les deux autres critères d'adaptabilité et de personnalisation peuvent également être regroupés dans un critère global de « Sociabilité ». Ainsi, un outil adaptatif et fournissant des résultats personnalisés sera considéré comme social. A l'opposé, un outil neutre ne sera ni adaptatif ni personnalisé, comme l'est, par exemple, le mécanisme de « favoris » des navigateurs Internet.

Nous proposons donc de définir l'intelligence d'un assistant selon deux critères généraux : sa sociabilité et sa réactivité. Ces deux critères sont focalisés sur la façon dont est apportée l'aide et non sur son contenu véritable. Selon ces critères, un outil intelligent se doit d'être social et actif alors qu'à l'opposé les outils « non-intelligents » seront neutres et passifs (voir figure 1.15). Il est possible de mettre ces conclusions en parallèle d'exemples concrets : un bloc-notes qui est, par nature, un accessoire de bureau neutre et réactif ne sera habituellement pas considéré comme « intelligent » alors qu'à l'opposé la sociabilité et la nature active sont caractéristiques d'une aide fournie par un collègue à qui l'on pose une question.

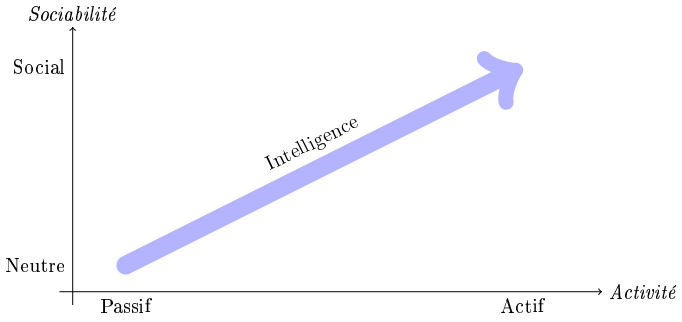


FIG. 1.15 – Graduation de l'intelligence d'un assistant

	Système passif	Système actif
Système neutre	Cartes de navigation Tours guidés Moteurs de recherche Messagerie éléctronique Forums de discussion Liste de sites favoris Annotation de pages	Guides pour site Internet Suggestion d'adresses
Système social	Rappel d'informations Exploration de site	?

TAB. 1.2 – Synthèse comparative de différents outils d'aide

1.6 Conclusion

Nous avons vu dans ce chapitre quels étaient les besoins d'un utilisateur ainsi qu'un panel des solutions destinées à lui fournir une aide. Afin de pouvoir évaluer l'intelligence de ces outils, nous avons introduit un critère d'activité relatif aux actions entreprises par l'assitant et un second de sociabilité se rapportant à sa relation avec l'utilisateur.

Le tableau 1.2 présente une synthèse comparative des différentes catégories d'outils présentés.

La première conclusion que nous pouvons tirer de ce tableau est que le nombre de systèmes neutres et passifs est important au regard des autres catégories. Les outils dont disposent les utilisateurs afin de faciliter leurs tâches d'acquisition, de gestion et de partage de l'information sont majoritairement des programmes servant juste de supports technique à une activité de l'utilisateur. La plupart de ces outils pourraient d'ailleurs être remplacés par d'autres moyens techniques non informatisés : noter les adresses de site sur un carnet, entretenir une conversation avec d'autres utilisateurs de vive voix, tracer une carte manuscrite du parcours effectué sur le Web, *etc.*

La seconde conclusion est que la case des systèmes actifs et sociaux (et par là même intelligents, selon notre définition) est vide. En effet aucun des systèmes présentés ne témoigne de tous les facteurs qui auraient permis de l'identifier comme étant intelligent. L'étude des facteurs manquants à ces outils afin de les rendre intelligents fera l'objet du chapitre suivant.

Chapitre 2

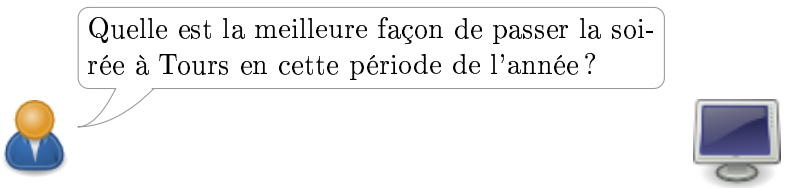
Du navigateur intelligent au Web intelligent

Ce chapitre s'intéresse aux caractéristiques que devrait avoir un assistant social et actif (et donc intelligent). Cette analyse passe par l'étude des dernières évolutions des technologies liées au Web, en particulier la mise en place d'une stratégie d'échanges axés sur les services en remplacement de celle habituellement axée sur les contenus.

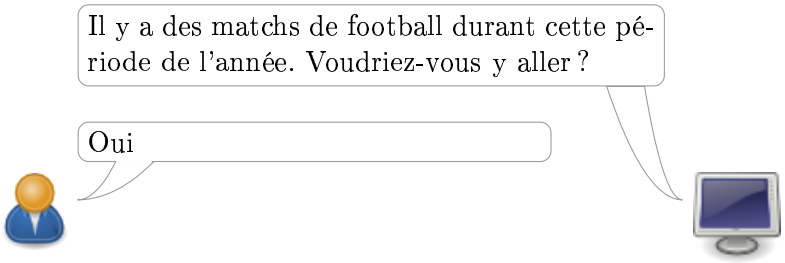
2.1 Une seconde histoire

Au premier chapitre de cette thèse, nous avons présenté l'exemple d'un utilisateur - touriste de son état - se demandant comment occuper son temps libre. Son problème, tenant dans la simple interrogation « quelle est la meilleure façon de passer la soirée à Tours en cette période de l'année ? », peut se résoudre en consultant diverses sources d'informations. Cette consultation peut, comme nous l'avons vu, être facilitée par l'utilisation de divers catégories de programmes assistants mais est néanmoins susceptible de rester fastidieuse pour l'utilisateur. N'étant pas suffisamment intelligentes, les aides fournies ne sont finalement que des moyens supplémentaires venant s'ajouter à ceux dont notre utilisateur dispose dès le départ.

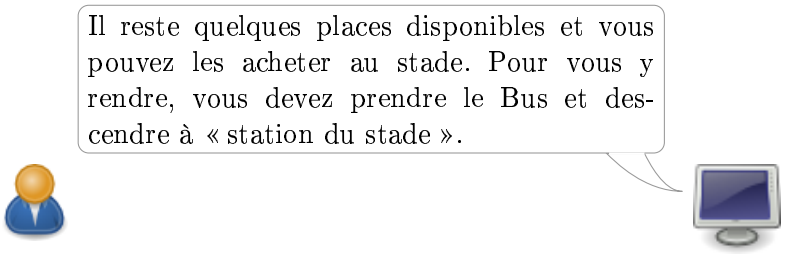
Supposons maintenant que ce même utilisateur ait à sa disposition une machine intelligente capable de comprendre son objectif et de l'aider à y arriver. Nous pourrions alors imaginer assister à la scène suivante : l'utilisateur rentre dans un cybercafé et se connecte au réseau Internet en s'identifiant sous son pseudonyme « Tolgam ». Un fois connecté, il lui suffit d'entamer la discussion avec la machine en lui exprimant son problème :



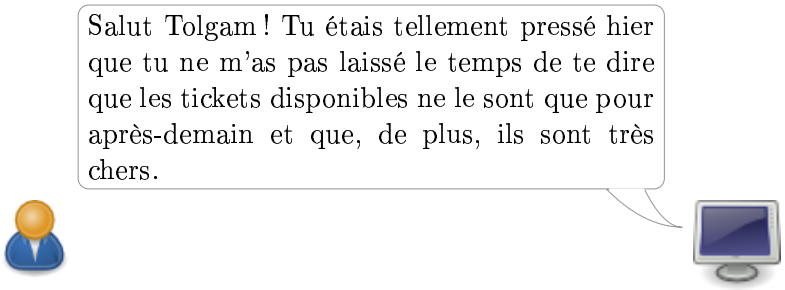
La machine analyse alors cette demande, notamment en situant dans le temps ce que l'utilisateur entend par « cette période de l'année » et « meilleure façon ». Après avoir interrogé diverses sources d'informations, la réponse est retournée et le dialogue continu :



La réponse de l'utilisateur est la plus simple qu'il puisse fournir. Implicitement, ce dernier compte alors sur la capacité de son interlocuteur virtuel à suivre une conversation. L'utilisateur ayant accepté sa proposition la machine poursuit ses recherches et fournit toutes les informations techniques nécessaires :



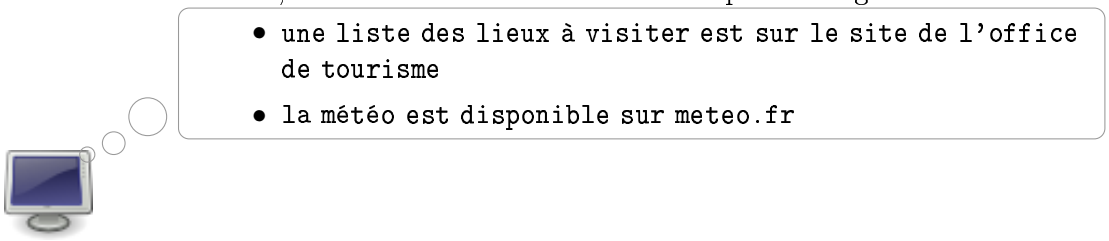
Fort de cette information, notre utilisateur décide que ce serait une bonne idée. D'autant plus qu'un ami lui avait déjà recommandé ces matchs avant son départ. Cependant, une fois rendu sur place, ce dernier se rend compte que les places disponibles ne le sont que pour le match qui aura lieu deux jours plus tard. Le lendemain, « Tolgam » retourne se connecter au cybercafé. La machine, se souvenant de la discussion de la veille, entame la conversation :



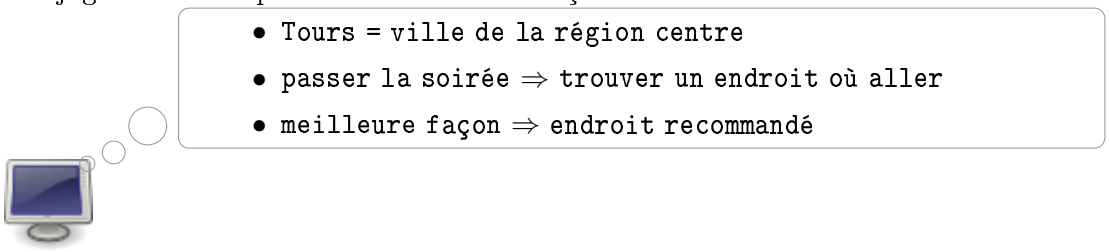
Contrairement à notre première histoire, l'utilisateur n'a pas eu besoin de chercher les horaires des matchs ni de réfléchir au moyen de s'y rendre ou à l'endroit où il pourrait se procurer des tickets. Sa seule activité a été de poser sa question à la machine et de poursuivre la conversation.

Cet exemple est inspiré de celui utilisé par Zhong *et al.*[Zhong 03] afin d'introduire le concept du « Wisdom Web », ou « Web de la sagesse » dans sa traduction littérale. Sur la base de ce scénario fictif, il est possible de dénombrer 6 points caractéristiques que cet outil doit posséder :

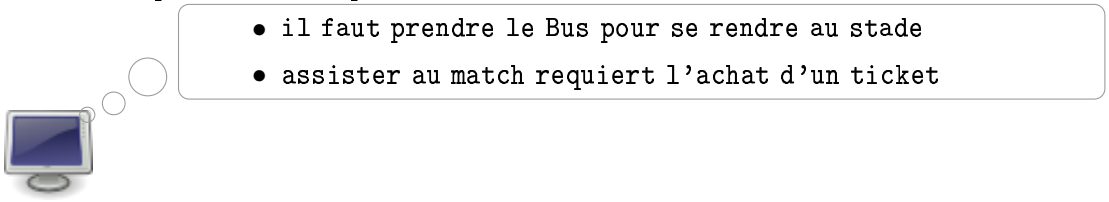
1. **Le support d'un « Autonomic Web »** : Être capable d'utiliser les différents sites du Web et de leur attribuer un rôle fonctionnel. Être capable également de s'auto-réguler en fonction de son activité, des diverses contraintes et de son paramétrage.



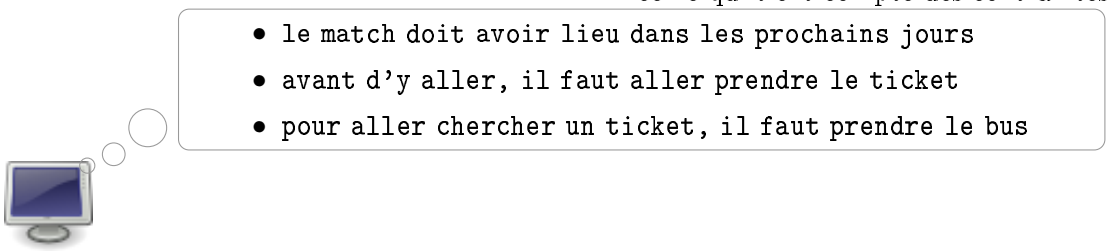
2. Pouvoir prendre en compte la **sémantique** afin d'attribuer un sens à l'expression « passer la soirée », trouver ce que veut dire « Tours » dans le contexte de la phrase et définir quel est le jugement correspondant à « meilleure façon ».



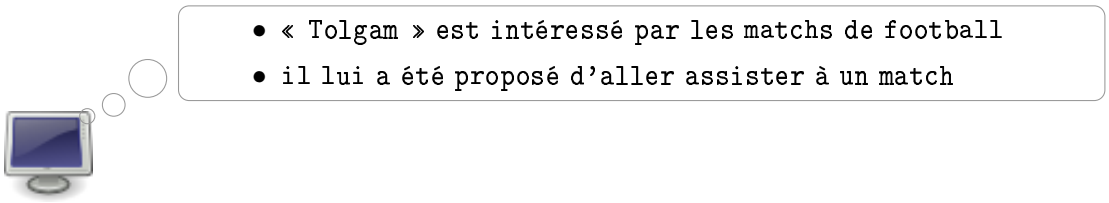
3. L'intégration de **méta-connaissances** lui permettant de lier entre eux les différents concepts rencontrés et prendre en compte les contraintes existantes.



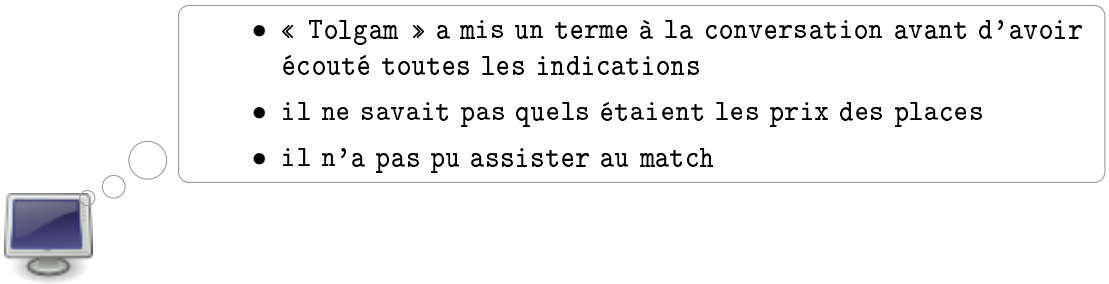
4. Avoir des notions de **gestion de temps (« Planning »)**. Savoir quelles sont les contraintes ne suffit pas, il est également nécessaire de les situer dans le temps et de les coordonner. La définition de cet enchaînement est une action réfléchie qui tient compte des contraintes.



5. Fournir des informations **personnalisées** afin de savoir qui est « Tolgam », y associer la conversation passée et mémoriser ses centres d'intérêts.



6. Un certain **sens de l'humour** pour pouvoir imaginer dans quel état d'esprit est « Tolgam » quand il se connecte à nouveau sur le réseau et s'en amuser.



La nature active du comportement de cet assistant ainsi que sa sociabilité le classe, selon les critères introduits au chapitre précédent, dans la catégorie des assistants intelligents. Selon leurs auteurs, cette intelligence tient dans la capacité à savoir utiliser avec discernement les connaissances dont dispose l'assistant.

Nous pouvons également remarquer que, finalement, la démarche de l'utilisateur dans notre premier exemple est celle suivie par l'assistant intelligent dans le second. Cette « délocalisation » du processus cognitif relatif à la combinaison des informations, l'établissement des liens logiques et la gestion de planning permet de passer d'une démarche orientée sur les moyens à celle axée sur les objectifs. L'utilisateur n'a plus à se soucier de savoir *comment* il va résoudre son problème, il lui suffit de le formuler et de laisser l'assistant faire son travail.

2.2 L'apparition de nouvelles technologies

Le modèle le plus élémentaire du WWW est le suivant : un ensemble d'utilisateurs met en place des machines serveurs sur lesquelles ils publient du contenu. En tant que responsables de publication de contenu, ces utilisateurs sont des Webmestres (« Webmasters »). Un autre groupe d'utilisateurs consulte ce qu'ont publié les Webmestres par l'intermédiaire de machines clientes. Bien que le réseau Internet sur lequel s'appuie le WWW autorise des communications entre toutes les machines, les seuls échanges observés se font entre des clients et des serveurs distincts. Les liens hypertexte étant le seul moyen (unidirectionnel) de lier les serveurs entre eux (voir figure 2.1).

De nouvelles technologies modifient progressivement ce modèle :

- Grâce aux **langages sémantiques**, les Webmestres peuvent désormais indiquer le contenu de leur site à l'aide d'une structure logique facilement utilisable par des programmes automatisés ;
- Les **services Web** permettent de remplacer les classiques demandes de pages et autorisent la formulation de n'importe quelle requête sémantique ;

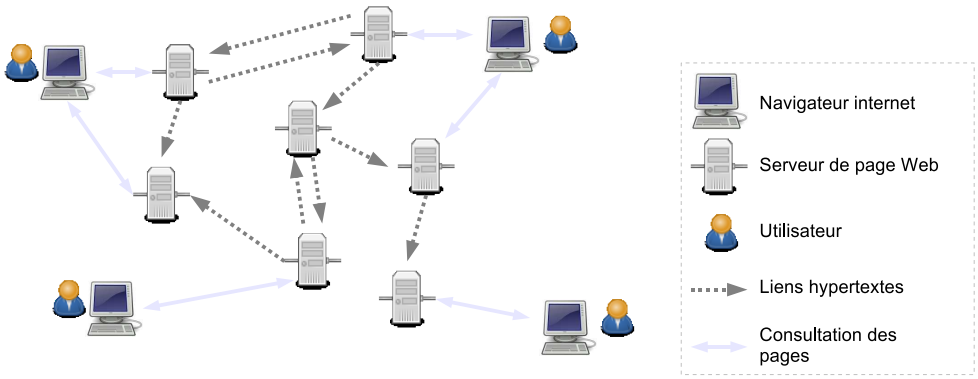


FIG. 2.1 – Modèle simplifié des échanges liés au Web sur Internet

- L'architecture **pair à pair** (P2P) permet à chaque machine d'être utilisée en tant que serveur ;
- Enfin, en prenant en compte les relations entre les quant à lui et leurs activités, l'étude des **réseaux sociaux** (« *social networking* ») permet de développer de nouveaux schémas d'interactions.

2.2.1 Nouveaux langages liés au Web sémantique

Depuis leur invention, les sites Web sont constitués de pages au format HTML (Hyper Text Markup Language) échangées entre le client et le serveur *via* l'utilisation du protocole HTTP (Hyper Text Transfer Protocol). Ces deux systèmes remplissent leur rôle mais ne permettent de prendre en compte que le contenu des pages fournies et non leur sens.

Prenons l'exemple d'une page d'un site de bibliothèque virtuelle contenant un livre sur les recettes de cuisine. Le client voulant consulter cette page va émettre une requête HTTP de type GET indiquant l'adresse de la page (disons, `http://www.bibliotheque.com/livre_cuisine.html`). En réponse, le serveur va retourner le contenu de la page HTML du tableau 2.2.

```
1 <body>
2    <h1>Cuisine facile</h1>
3    <h2>Introduction</h2>
4    Texte du chapitre introductif
5    <h2>Conclusion</h2>
6    Texte de la conclusion
7 </body>
```

FIG. 2.2 – Exemple de fichier HTML (sans indication d'en-tête). `body` indique le corp de la page, `h1` et `h2` sont deux formats de mise en page de section.

Le principal problème du HTML est qu'il ne permet pas de séparer l'information, son encodage et sa représentation. Un client cherchant à connaître la liste des chapitres devra savoir que, dans

la page, la balise `h1` est suivie de balises `h2` décrivant toutes un unique chapitre. Ceci car, dans l'exemple 2.2, la balise `h2` représente : un titre (information), le fait que ce titre soit celui d'un chapitre de l'ouvrage (encodage) et une mise en forme avec (par défaut) une police de caractères plus petite que celle utilisée par `h1` (représentation). Un autre langage, le CSS (Cascading Style Sheet) permet, en accompagnement du HTML, de séparer le contenu de la page de sa représentation (tableau 2.3).

```
1 H1 {
2    font-style: italic;
3    color: green
4 }
5 H2 {
6    color: red
7 }
```

FIG. 2.3 – Exemple de fichier CSS

Afin de séparer l'information de son encodage, le XML (e**X**tensible **M**arkup **L**anguage) permet aux Webmestres de décrire dans un premier document le contenu de la page en inventant toutes les balises nécessaires et dans une seconde page (la DTD - Document Type Definition) la structure de ces balises. Le tableau 2.4 contient un exemple de fichier XML décrivant la page de la bibliothèque. On y remarque que les balises `bibliotheque`, `livre` et `chapitre` ont été créées afin de remplacer les `h1` et `h2` précédemment utilisées. Afin de clarifier la lecture du document, un attribut `titre` permet de facilement associer un titre à chaque niveau structurel du document. Le tableau 2.5 contient quant à lui le fichier DTD qui décrit les règles qui régissent l'utilisation de ces balises nouvellement créées. Le client ayant récupéré le fichier XML ainsi que la DTD auprès du serveur sait qu'une `bibliotheque` est composée de zéro ou plusieurs `livre`, eux-mêmes contenant zéro ou plusieurs `chapitre` dont le contenu est libre. La récupération de la liste des chapitres est ainsi facilitée car il suffit de rechercher l'ensemble des balises `chapitre`.

```
1  <bibliotheque>
2     <livre titre="Cuisine facile">
3        <chapitre titre="Introduction">
4           Texte du chapitre introductif
5        </chapitre>
6        <chapitre titre="Conclusion">
7           Texte de la conclusion
8        </chapitre>
9     </livre>
10 </bibliotheque>
```

FIG. 2.4 – Exemple de fichier xml

Nous présenterons également un dernier langage, le `RDF` (**R**esource **D**escription **F**ramework), qui est une boîte à outils permettant de décrire les ressources disponibles sur le Web. Les fichiers RDF sont écrits en XML. Ils sont destinés à être lus et interprétés par les machines et non par les quant à lui [1] et se composent de triplets associant une ressource, une propriété et une valeur

[1] `http://www.w3schools.com/rdf/rdf_intro.asp`, dernier accès le 23 octobre 2006

```
1 <!ELEMENT  bibliotheque (livre*)>
2 <!ELEMENT  livre (chapitre*)>
3 <!ELEMENT  chapitre ANY>
4 <!ATTRLIST livre titre CDATA #REQUIRED>
5 <!ATTRLIST chapitre titre CDATA #REQUIRED>
```

FIG. 2.5 – Exemple de fichier DTD

à cette propriété. Par exemple, les valeurs de « traite le même sujet que » ou « est du même auteur que » peuvent être appliquées à une propriété de relation entre deux pages Web.

```
1 <RDF>
2   <Description about="http://wwww.bibliotheque.com/
       livre_cuisine.html">
3     <titre>Cuisine facile</titre>
4     <author>Illustre Inconnu</author>
5     <homepage>http://www.bibliotheque.com/</homepage>
6   </Description>
7 </RDF>
```

FIG. 2.6 – Exemple de fichier RDF

Les RDF sont utilisés afin d'écrire des fichiers RSS 1.0 (**R**essource **S**ite **S**ummary) résumant le contenu global d'un site. Avec l'ensemble de ces langages XML, DTD et RDF il est donc possible de définir un résumé du site ainsi qu'une spécification de son contenu. Afin de prendre en compte la sémantique d'une page, il ne reste plus alors qu'à définir ce qu'est un `chapitre`, un `livre` et une `bibliotheque` selon l'enchaînement de ces balises ainsi que le contexte de leur utilisation. Ceci passe par la formalisation d'ontologies.

Definition 1 (Ontologie) *Une ontologie est une représentation de l'existant qui*[2] *:*

1. *reflète les propriétés d'un objet dans son domaine de façon à obtenir une corrélation entre la réalité et cette représentation.*
2. *est intelligible pour un expert du domaine.*
3. *est formatée de façon à autoriser des traitements automatiques de l'information.*

Pour décrire et échanger ces ontologies, le W3C a défini le langage OWL (Web Ontology Language). Ce dernier, en utilisant RDF comme une base pour la définition de triplets autorise la définition de relations entre les différents éléments.

Du fait de la perte de la sémantique et du contexte des données qui y sont publiées, le Web était devenu une ressource de documents plus qu'une ressource d'informations indexables. Ces nouveaux langages définis dans le cadre du « Web sémantique » permettent de mettre en place une extension du Web actuel et d'apporter une solution à cette perte de la sémantique [Berners-Lee 01].

[2] `http://www.ontologyworks.com/what_is_ontology.php/`

2.2.2 Nouveaux langages de service

Le « Web sémantique » permet d'ajouter de la sémantique aux données du Web mais ne change pas le mode d'interrogation. Dans notre exemple, si le client veut accéder au contenu du livre il lui faut nécessairement demander la page le contenant. Se pose alors le problème de connaître le numéro de la page en question. De plus, si ce même client veut savoir le nombre de chapitres de l'ouvrage, il lui est nécessaire d'analyser le contenu de la page lui-même.

Une autre approche est rendue possible grâce aux langages liés à la définition des services Web (« Web services »). Ces langages permettent de donner à un client directement accès à des fonctionnalités d'une application. Jusqu'à présent, avec le HTTP, la seule fonctionnalité était celle de l'envoi de page : à la requête « Donne moi le contenu de la page `livre_cuisine.html` », le serveur répondait avec le contenu *ad-hoc*. Les Web services permettent quant-à-eux de rendre disponibles des requêtes du type « Envoi-moi le contenu du livre sur la cuisine » ou « Quel est le nombre de page du livre de cuisine ? ».

Le langage SOAP (Simple Object Access Protocol) est un exemple de cette technologie. La figure 2.7 montre un exemple de requête et de réponse SOAP[3] portant sur le nombre de chapitres d'un ouvrage. Le client appelle la fonction exportée `GetChapterCount` en indiquant le ou les titres des livres concernés.

```
1 <Body>
2    <GetChapterCount>
3       <Book>XML</Book>
4    </GetChapterCount>
5 </Body>
```

```
1 <Body>
2    <GetChapterCountResponse>
3       <Count>2</Count>
4    </GetChapterCountResponse>
5 </Body>
```

FIG. 2.7 – Exemple (simplifié) de requête et de réponse SOAP

Comme le montre cet exemple, les messages SOAP sont écrits en XML. Ils sont ensuite échangés entre le client et le serveur avec le protocole HTTP. Cette utilisation d'un protocole simple et stabilisé facilite la mise en place des services. Le client n'a besoin que de connaître l'adresse de la page contenant le gestionnaire de services ainsi que le nom des fonctions à appeler.

2.2.3 Architecture réseau Pair à Pair

La figure **??** montrait le Web composé de l'ensemble des quant à lui publiant du contenu (les Webmestres) et de l'autre ensemble le consultant (les Internautes). Dans ce cas de figure, n'importe quel internaute désirant à son tour publier une information doit se comporter en tant que Webmestre. Il lui faut alors disposer d'un serveur et des compétences lui permettant de mettre en place un site Web : une contrainte dissuader un utilisateur de diffuser des informations.

Les réseaux Pair à Pair gardent cette idée de client et serveur mais en modifient l'utilisation. Dans un réseau P2P, tous les quant à lui sont à la fois susceptibles d'être clients et serveurs pour un ensemble de tâches données. Ce modèle a été popularisé avec l'apparition du logiciel d'échange de fichiers musicaux Napster[4]. Grâce à ce logiciel assurant à la fois la recherche et la

[3] Ces exemples ont été simplifiés afin d'en faciliter la lecture

[4] `www.napster.com`, dernier accès le 3 mars 2007

diffusion de contenu, les Internautes ont pu s'échanger des fichiers sans avoir besoin de mettre en place un serveur Web.

Bien que le principe général (tous les postes sont équivalents) soit simple, il existe en réalité différents types de P2P. Les différences se font au niveau de la topologie du réseau et de l'utilisation pratique de ces derniers.

- **Centralisé / Décentralisé**
  Un exemple de P2P centralisé est la solution proposée par le logiciel Napster. Ce dernier se base sur un serveur d'indexation pour la recherche de contenu et des échanges en P2P (figure 2.8). Quand un pair arrive sur le réseau, il contacte le serveur central pour lui signaler sa présence ainsi que les fichiers qu'il propose. Un pair quittant le réseau signal également son départ. Le serveur d'indexation est alors en mesure d'indiquer précisément dans le réseau quels sont les fichiers proposés par chaque pair. Les requêtes lui sont donc directement envoyées.

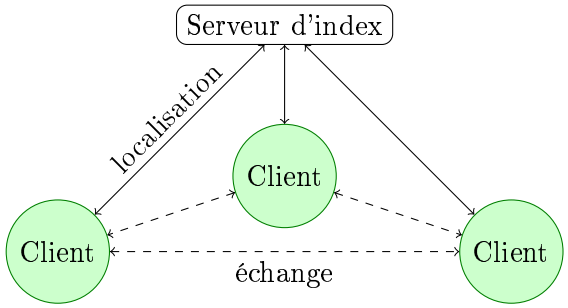


FIG. 2.8 – Localisations et échanges de fichiers avec un serveur centralisé

- **Hybride / Pur**
  Dans un réseau P2P pur, tous les pairs ont des rôles exactement égaux. Un réseau hybride met en œuvre une différenciation donnant généralement lieu à la mise en place de « super pairs » tels qu'utilisés dans Gnutella. Ces super pairs ne sont pas fonctionnellement différents des autres (à la différence d'un serveur d'index dans un P2P centralisé) mais ont une place plus importante. Ils peuvent servir de « proxy » pour les transferts ou de routeur longue distances.
- **Structuré / Non structuré**
  La dernière différenciation se trouve au niveau de la topologie. Cette notion, sur laquelle nous reviendrons par la suite, définit l'agencement global du réseau. Un réseau dit structuré montre une certaine organisation concernant le placement des pairs et des données. Ceci contrairement à son opposé non structuré qui est de nature plus imprévisible. Les systèmes hybrides sont assimilés à une structure semi-structurés.

L'inconvénient des systèmes centralisés réside dans l'utilisation du serveur central (par exemple, un serveur d'index). En son absence, l'ensemble du système est mis hors service. Il s'agit donc d'un élément critique demandans une attention particulière pour prévenir tout risque de panne. De plus, la puissance de cette machine doit être choisie en fonction du nombre de pairs s'y connectant. Un nombre conséquent de ces connexions impliquant des charges conséquentes en terme de temps processeur et de ressources réseau.

L'avantage de ces systèmes est que seul le serveur central a besoin de traiter les demandes des machines dans le réseau. Dans le cadre du partage de fichiers, la centralisation du service d'indexe facilite la recherche de fichiers. Les systèmes d'échanges de fichiers n'utilisant pas de serveur d'index sont, pour la majorité, des systèmes hybrides. Dans ces systèmes, la localisation des contenus se base sur un réseau P2P pur alors que le transfert effectif du contenu convoité se fait selon un mode P2P. Pour que le système soit en P2P pur, il faudrait que ce transfert soit, comme les requêtes, effectué par relais de pair en pair. La charge réseau que cela impliquerait n'est en général pas justifiée. Seuls les outils de diffusion distribués de flux ainsi que ceux destinés à combattre les phénomènes de « flashcrowd » tirent partie de l'utilisation de transferts en P2P.

Au final, les systèmes P2P peuvent être classés selon l'arbre 2.9

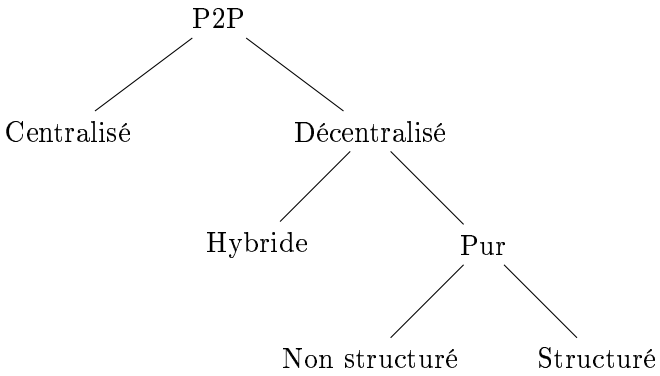


FIG. 2.9 – Classification des systèmes P2P

2.2.4 Réseaux sociaux

Un réseau social se définit comme un ensemble d'éléments (les « acteurs ») connectés entre eux conformément à un ensemble de relations [Newman 01]. Ces relations peuvent, par exemple dans le cas d'individus, aller de la notion de subordination à celle de parenté, les graphes résultants traduisant alors un organigramme ou arbre généalogique. Une des particularités de ces réseaux est que tous les quant à lui ne s'y trouvent pas au même niveau de relation. Ce niveau relationnel se définit par les degrés entrants et sortants d'un nœud du graphe. C'est-à-dire le nombre de liens qui pointent vers ce nœud et le nombre de liens qui en partent. Des nœuds disposant d'un degré entrant élevé sont des « autorités » (« Authority ») alors que ceux disposant d'un degré sortant important sont des « concentrateurs » (« Hub ») ou « portails » (voir figure 2.10).

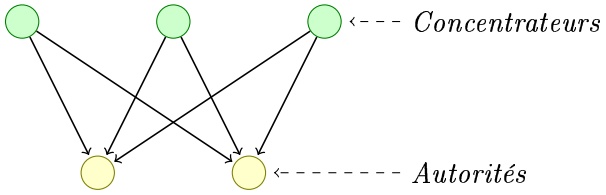


FIG. 2.10 – Relations entre « concentrateurs » et « autorités » dans un réseau social

Dans le cas du Web, où les nœuds sont des pages Web et les connexions des liens hypertextes entre elles, cette différenciation se manifeste par la présence de deux types de sites. Ceux ne

contenant que peu de contenu propre mais un grand nombre de liens tels que les sites portails et ceux qui ont un grand nombre de liens entrants du fait de leur popularité. Un exemple d'exploitation de ce principe est contenu dans l'algorithme « PageRank » utilisé par le moteur de recherche Google [Brin 98]. Cet algorithme est orienté autour de l'équation 2.1 qui permet de définir le niveau d'autorité d'une page en considérant les liens entrants comme autant de votes en sa faveur. Une page ayant un nombre de votes élevé sera supposée plus pertinente que les autres et aura un meilleur score. Le rang PR d'une page i est calculé en fonction des PR de l'ensemble T_i des pages pointant vers elle (les liens entrants). Est également pris en compte dans ce calcul le nombre de liens sortants $C(i)$. Le paramètre d est un facteur de normalisation fixé à 0.85 par les auteurs.

$$PR(i) = (1-d) + d \sum_{j \in T_i} \frac{PR(j)}{C(j)} \tag{2.1}$$

Si, à la place des pages Web, ce sont des individus et une relation de connaissance qui sont représentés, les « Hubs » sont des personnes ayant un carnet d'adresses bien fourni alors qu'une autorité apparaît dans bon nombre de ces carnets. Dans un réseau social où les autorités sont des experts d'un domaine particulier, le graphe est représentatif d'un système de référents où des chaînes de référents se mettent en place pour relier demandeurs d'informations et experts du domaine. Les experts d'un domaine sont les autorités (les personnes qui savent) alors que les référents sont les concentrateurs (les personnes qui connaissent la personne qui sait). Les premières études sur ce type de réseau sont attribués à Kautz *et al.* en 1996 [Kautz 96]. La confrontation de deux stratégies d'accès à l'information, interroger une personne ou un programme, dans une grande entreprise a permis la mise en évidence d'une relation entre la longueur et la qualité des chaînes de référents ainsi que le nombre et la qualité des réponses de ces référents. Plus précisément, le lien qu'entretient un utilisateur avec un référent peut être caractérisé par les deux attributs d'expertise et de sociabilité [Yolum 03, Schmitz 04]. Le premier se rapporte aux domaines de connaissances du référent, il peut être utilisé afin de qualifier la qualité de ses recommandations ou simplement décrire quels sont ces domaines. La sociabilité quant-à-elle représente la force du lien qui lie deux quant à lui, l'appréciation d'un lien peut se faire, par exemple, selon la qualité des référents proposés par l'utilisateur lié ou par une estimation des centres d'intérêt communs.

La figure 2.11 est un exemple de chaîne de référent dans un graphe composé de 10 individus $A, B, C, D, E, F, G, H, I$ et J. Cette figure représente la progression d'une requête émise par A puis transférée de référent en référent jusqu'à J et F. Le parcours menant à F se solde par un échec, ce dernier n'étant en mesure ni de fournir une réponse ni d'indiquer un référent. En revanche, en passant par l'intermédiaire de B et G, il a été possible de trouver une réponse donnée par l'individu J.

Sous certaines conditions, il est possible de voir apparaître des structures de communautés. Dans un réseau social, ces communautés se traduisent par l'apparition de groupements d'individus représentatifs d'une relation les liant les uns aux autres, telle que, par exemple, des centres d'intérêt communs. Les mesures d'expertise et de sociabilité peuvent être utilisées comme une métrique pour la détection de communautés [Yolum 03]. Pour ce faire, la sociabilité $S(i)$ d'un agent i est définie comme étant liée à la sociabilité moyenne de l'entourage de i (voir équation 2.2). Dans cette formulation, I_i désigne l'ensemble des voisins de i et $\delta_{i,j}$ correspond à la sociabilité de i envers j (qui n'est pas obligatoirement égale à son symétrique). Le coefficient d est

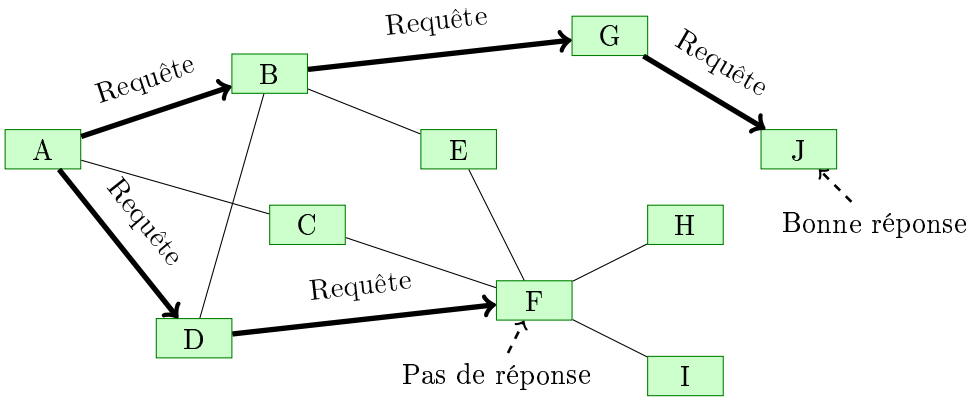


FIG. 2.11 – Circulation d'une requête dans un réseau de référents (d'après [Yu 03])

utilisé à des fins de normalisation du calcul.

$$S(i) = (1-d) + d \sum_{j \in I_i} (S(j) \times \frac{\delta_{j,i}}{\sum_{k \in N_j} \delta_{j,k}}) \tag{2.2}$$

On remarquera ici une certaine similarité avec la formulation du score de page (PageRank) proposée par Google.

Les réseaux dans lesquels des communautés émergent peuvent avoir une topologie de « petit monde » (« small world »). Il s'agit de graphes composés de plusieurs groupements denses de connexions interconnectés entre eux. Cette structure a été observée dans de nombreux cas tels que les connexions entre réalisateurs de film et acteurs et les publications en coauteurs. Elle se caractérise par deux métriques [Watts 98] : le cœfficient de plus court chemin $\mathcal{L}$ et celui de taux de groupement $\mathcal{C}$. Le premier permet d'estimer le taux de groupement des nœuds et le second la densité de leur interconnexions. Un réseau « petit monde » sera identifiable par une faible valeur de $\mathcal{L}$ et un $\mathcal{C}$ élevé. Il est d'ailleurs intéressant de remarquer qu'une valeur faible de plus court chemin est une caractéristique habituelle des réseaux aléatoires alors qu'un coefficient de groupement élevé est facilement associé aux réseaux structurés. Les graphes « petit monde » sont donc à la fois aléatoires et structurés.

Sur la base de ces deux cœfficients, Watts et Strogatz ont montré [Watts 98] que dans un réseau composé d'individus, une topologie de « petit monde » favorise la propagation d'une maladie. De plus, ils ont montré qu'un faible nombre d'inter-liens était suffisant pour obtenir ce résultat. Transposés à notre problème, ces résultats laissent supposer qu'à l'instar d'un virus, la propagation d'un fichier intéressant sera facilitée par cette topologie. Au delà du domaine viral, il a été démontré que des topologies de « petit monde » étaient observables dans les réseaux sociaux. La recherche active dans le domaine de la localisation d'informations a fait apparaître différentes relations. Parmi celles-ci on peut remarquer la confiance, la proximité géographique, la réputation et la notion d'intérêt commun.

Les travaux de Iaminitchi *et al.* [Iaminitchi 02, Iaminitchi 04] sur le « graphe de partage de données » de la communauté scientifique ont prouvé que, dans ce graphe où les liens entre les nœuds traduisent des centres d'intérêt commun entre chercheurs, une topologie de « petit monde » apparaissait. Pour assurer la diffusion des messages, les auteurs proposent l'utilisation d'un algorithme de type « bavardage » (« Gossip »). Considérant un « petit monde » de C groupements

chacun composé de G noeuds. Chaque noeud d'un groupe connaît une partie des noeuds composant le groupe. Quand un noeud veut émettre un message il l'envoit à une partie du groupe. Les noeuds recevant un message le traitent (ajoute des informations, ...) puis le réémettent sur le même principe.

2.3 Web intelligence : les prémices d'un nouveau Web

Les recherches menées sur la « Web Intelligence » déciderons très certainement de la prochaine évolution du WWW, quelle qu'elle soit. Ce nouveau domaine de recherche, identifié il y a peu comme consistant en « l'étude des rôles fondamentaux ainsi que des applications pratiques de l'intelligence artificielle (IA) et des technologies avancées de l'information (TIC) sur la prochaine génération d'outils, produits, services et activités utilisant le Web »[5]. Cette étude peut être menée sur, au moins, quatre niveaux allant d'un point de vue centré sur le matériel à des considérations liées aux applications de plus haut niveau [Zhong 03] :

- **Niveau Internet**
  Ce niveau est axé sur l'aspect d'infrastructure, de sécurité et de protocoles du Web. N'ayant pour unique informations que les communications entre machines, les outils intelligents développés à ce niveau sont axés sur des techniques permettant l'optimisation des échanges. Par exemple, en effectuant de la mise en cache préventive basée sur l'étude de transferts précédents ;
- **Niveau Interface**
  En étant lié aux standards de représentation multimédia, ce niveau tient lieu d'interface entre l'homme et Internet. Les techniques qui y sont développées se rapportent à la personnalisation de l'information ainsi qu'au traitement du langage et de données ;
- **Niveau Connaissances**
  A ce niveau, les outils pour le traitement et la gestion de l'information perçoivent le Web comme une source de données/connaissances distribuée. Les tâches affairantes sont celles de recherche, d'aggrégation, de classification, de filtrage, d'exploration et de fouille de données dans le réseau. Cela concerne, par exemple, l'aspect d'« autonomic Web » développé en introduction de ce chapitre ;
- **Niveau Applicatif**
  Les techniques mises au point à ce dernier niveau, le plus élevé, se focalisent sur la construction de modèles des centres d'intérêt des quant à lui et l'identification des relations entre eux. En considérant le Web comme un environnement pour l'intelligence sociale et l'informatique ubiquiste, ces techniques permettent la mise au point d'outils de personnalisation du Web et de support de communautés.

La figure 2.12, adaptée de [Zhong 03], représente de façon schématique le positionnement de ces différents niveaux d'étude. Les fonctions de support, c'est-à-dire la quantité d'aide fournie à l'utilisateur, vont croissantes avec la montée dans les étages de la WI.

[5] `http://wi-consortium.org/`

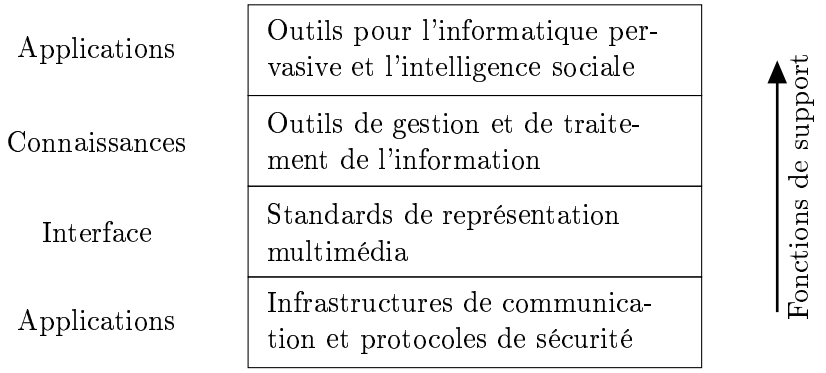


FIG. 2.12 – Les quatre niveaux d'approche de la Web Intelligence (WI), adapté de [Zhong 03]

2.3.1 Modèle proposé par le « Wisdom Web »

Le « Wisdom Web » peut être vu comme une « écologie des ressources de connaissances mondiales »[6][Liu 05]. Ces ressources, qui peuvent provenir de différents domaines et partager différentes ontologies, sont réparties entre différentes entités intelligentes. L'idée du « Wisdom Web » est d'englober les notions de systèmes, d'environnements et d'activités utilisées soit dans la distribution de l'information et des ressources matérielles, soit dans le dialogue et l'échange de connaissances entre quant à lui. Dans cette optique, le « Wisdom Web » doit assurer trois fonctions :

- **Découvrir les meilleurs moyens et les meilleures finalités**
  Le « Wisdom Web » doit découvrir : quels sont les objectifs et sous-objectifs que l'utilisateur essaye d'atteindre ? Quelle sera la meilleure stratégie ? Quelle sera la liste d'actions pour implémentation ?
- **Mobiliser les ressources distribuées**
  Le « Wisdom Web » doit déterminer : quelles ressources sont significatives ? Comment les ressources distribuées peuvent être coordonnées et filtrées ? Quels sont les moyens de les utiliser tout en ayant un bon rapport coût/performances ? Quelles sont les dynamiques qui régissent l'utilisation des ressources ?
- **Enrichir l'intéraction sociale**
  Le « Wisdom Web » doit comprendre : quelles sont les nouvelles formes d'interactions sociales émergeant dans le travail, la vie et le jeu ? Comment certaines formes de normes sociales, de valeurs, de croyances et de sens commun peuvent être promues et partagées ? Comment une communauté sociale peut-elle être soutenue ?

La mise en place de ce « Wisdom Web » se fera progressivement, sur la base d'un ensemble de « Wisdom agents » [Liu 05]. Les différentes étapes de cette mise en place sont résumées dans la figure 2.13. Chaque case de cette représentation indique le problème technique à résoudre ainsi que, entre parenthèses, l'objectif global à atteindre. Au cours de leur évolution, ces agents vont se spécialiser et acquérir de l'information résultant de l'interactions avec d'autres agents (problème 10). D'autres agents seront également créés ou détruits afin de s'adapter aux changements dans leur environnement (problème 9). Grâce à la spécialisation des agents et de leurs communications, chaque agent sera alors en mesure d'identifier des associations entre son rôle et le rôle des autres

[6] « ecology of world knowledge ressources »

agents (problème 8). Associations qui permettront alors la mise en place de relations sémantiques entre agents (problème 7), la création de méta-données pour la gestion de planning (problème 6), ainsi que la coordination permettant de gérer des comportements de population (problème 5). Ce faisant, un langage de spécification de problème (Problem Solver Markup Language - PSML) sera utilisé afin de représenter et manipuler la sémantique ainsi que les méta-données (problème 4). La prise en compte d'aspects psychologiques et contextuels doit permettre aux agents d'avoir un comportement similaire à celui observé entre individus lors d'une conversation (problème 3). Le développement d'agents personnalisés étant rendu possible, une identité propre pourra leur être attribuée (problème 2). La présence d'un modèle ontologique des relations entre services, la connaissance de méthodes de résolution de problèmes et la disponibilité d'une population coordonnée d'agents ayant chacun une identité propre sociale et/ou psychologique permettront alors au « Wisdom Web » de répondre à la demande par des requêtes orientées objectives. C'est-à-dire, de trouver les meilleurs moyens pour arriver à la meilleur finalité (problème 1).

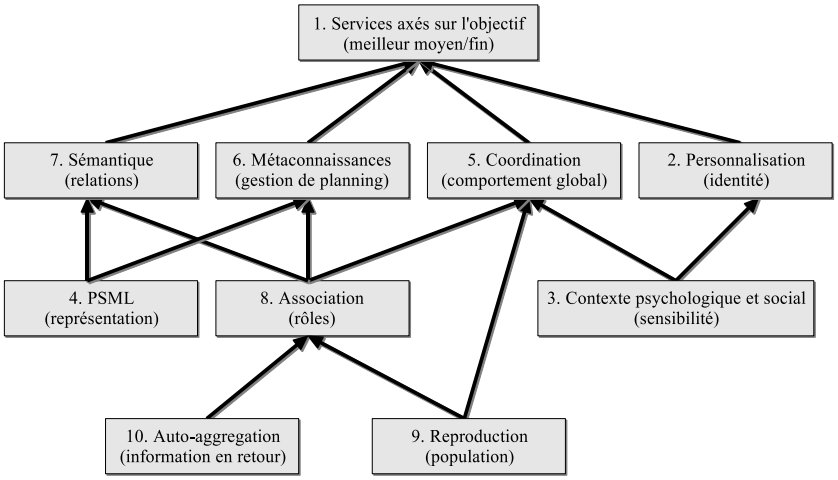


FIG. 2.13 – Le développement du wisdom Web (adapté de [Liu 05])

2.3.2 Le bureau sémantique social

Dans [Decker 04], Decker et Frank partent du constat qu'actuellement, et bien qu'elles soient utiles, les plates-formes de collaboration présentent des faiblesses au niveau de la gestion des informations. En prenant l'exemple des courriers électroniques, on constate que (1) les systèmes actuels (*ie* les arborescences de dossiers) ne permettent pas réellement de classer le courrier selon l'information contenue (2). Le contexte de création du courrier étant perdu lors de son envoi, c'est au récepteur de trouver comment classer l'information et établir le lien avec d'autres données qu'il possède déjà (3). Même si les méta-données sont incluses dans le document, il se peut qu'elles ne correspondent pas aux méthodes de classement de l'utilisateur.

Pour faciliter l'échange de méta-données ainsi que leur utilisation, le bureau sémantique social met en œuvre les résultats des recherches de trois domaines différents :

- le Web sémantique qui propose des structures de méta-données et d'ontologies standardisées telles que le RDF et OWL ;
- les logiciels sociaux permettant la découverte de nouvelles relations ainsi que la formation de communautés ;

- les réseaux P2P qui fournissent un support réseau autorisant la création de systèmes totalement décentralisés.

A l'instar du « Wisdom Web », la mise en place de cette architecture est envisagée en plusieurs étapes (figure 2.14). La première phase de cette mise en place concerne l'ensemble des technologies introduites dans les sections précédentes. La seconde phase comprend : l'ajout du support du RDF aux gestionnaires de fichiers permettant de passer d'un bureau classique à un bureau sémantique et la prise en compte de la sémantique dans les réseaux P2P ainsi que dans les réseaux sociaux. Le P2P sémantique et les réseaux sociaux sémantiques permettraient alors d'envisager un modèle de réseaux sociaux piloté par les ontologies. La troisième phase consiste alors à réunir le bureau sémantique ainsi que les réseaux sociaux ontologiques afin de produire un bureau sémantique social permettant à chaque utilisateur d'échanger des informations de manière efficace avec ses semblables.

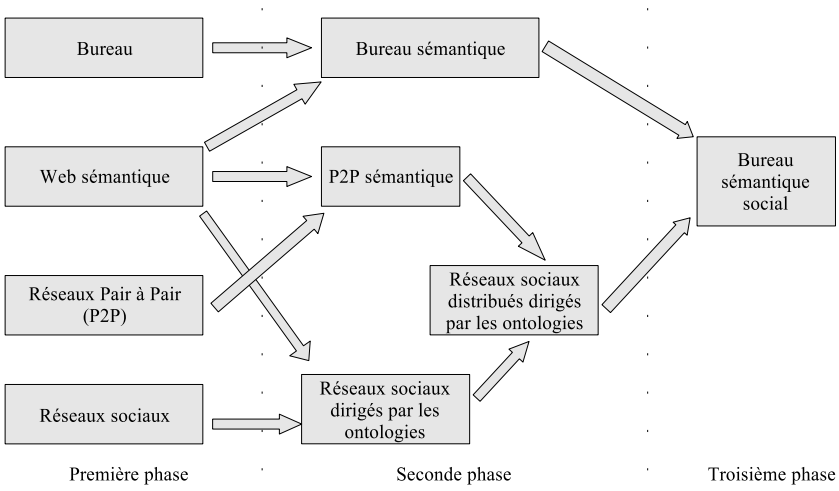


FIG. 2.14 – La mise en place d'un bureau sémantique social (adapté de [Decker 04])

La première application visible de cette idée est le projet **NEPOMUK**[7] démarré en 2006 avec le support notamment de la société française Mandriva [Writer 06]. Il a pour but l'extension du bureau personnel de chaque utilisateur en un environnement collaboratif capable à la fois de gérer les informations personnelles ainsi que leur partage dans des structures sociales ou organisationnelles. Ce projet apportera trois évolutions majeures aux bureaux classiques : la disponibilité d'informations contextuelles, le passage d'un stockage de données hiérarchique à un stockage sémantique et des outils avancés aidant les quant à lui à collaborer et à effectuer des recherches d'informations efficaces.

2.3.3 Le Web communautaire (Web 2.0)

Le terme de Web 2.0 a été initialement proposé en 2003 par Tim O'Reilly, président général et fondateur de la société O'Reilly Media[8]. Dans un texte publié en 2005[9] et qui, un an après,

[7] `nepomuk.semanticdesktop.org/xwiki/bin/Main1/`, dernier accès le 10 septembre 2006

[8] `oreilly.com`, dernier accès le 23 octobre 2006

[9] `www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-Web-20.html`, dernier accès le 24 Septembre 2006

fait office de référence dans ce domaine, Tim O'Reilly définit les caractéristiques du Web2 en sept points sous la forme d'une liste de conseils à suivre :

- Proposer des services en ligne au lieu d'applications capables de tirer partie du Web. Par exemple, privilégier des moteurs de recherche sur le Web (comme Google) plutôt que des applications externes (comme Copernic).
- Faciliter les contributions des quant à lui : dans le Web 2.0, « la valeur du logiciel est proportionnelle à l'ampleur et le dynamisme des données qu'il permet de gérer ». Autrement dit, la valeur du site se fait grâce aux quant à lui. Il suffit de prendre pour exemple des sites tels que **Slashdot**[10] ou **Amazon**[11] pour se rendre compte du volume de données que peuvent générer ces quant à lui.
- Suivre le modèle des logiciels libres et faire intervernir les quant à lui dans le développement de l'outil. En les considérant comme des co-développeurs, les quant à lui peuvent être impliqués dans l'amélioration du service. Le site **Netvibes** [12] propose ainsi à ses quant à lui d'écrire des modules et de les publier. En combinant l'utilisation de modules édités par Netvibes et d'autres provenant des quant à lui (l'« écosystème netvibes ») les abonnés à ce site peuvent se constituer un portail personnalisé. Une capture d'écran de ce site est visible sur la figure 2.15.
- Exploiter l'intelligence collective créée par l'activité de l'ensemble des quant à lui. L'encyclopédie libre Wikipedia ainsi que le site de dépôt de photographies **Flickr**[13] sont deux exemples d'utilisation de cette intelligence collective. La popularité du premier tendant notamment à montrer qu'un ensemble d'quant à lui peut remplacer une poignée de rédacteurs spécialisés.
- Toucher l'ensemble du public avec des outils simples d'accès.
- Prendre en compte le développement de l'informatique ubiquiste et mobile en proposant des applications qui ne se limite plus aux PCs de bureau.
- Mettre au point des modèles de gestion, de développement et d'interface utilisateur qui soient souples.

2.4 Conclusion

L'intelligence d'un navigateur ne serait donc finalement non pas liée aux programmes qui assistent l'utilisateur dans ses activités mais aux services qui lui sont proposés. Un navigateur intelligent ne va pas aider notre utilisateur fictif à chercher une occupation pour la soirée mais va en fait effectuer les recherches à sa place, en tant que service.

L'autre point à noter est la notion d'intelligence collective qui prévaut dans la mise en place des technologies liées au Web 2.0. Faire participer les quant à lui de manière active dans le développement de ces outils et de leur alimentation en informations permet de tirer partie des connaissances de chacun.

Cependant, les outils de collaboration actuels, Web 2.0 ou pas, souffrent les mêmes limitations que ceux dédiés au CSCW. A savoir, le possible manque de motivation des quant à lui, le besoin

[10] `www.slashdot.org`, dernier accès le 26 novembre 2006
[11] `www.amazon.com`, dernier accès le 26 novembre 2006
[12] `www.netvibes.com`, dernier accès le 24 septembre 2006
[13] `www.flickr.com`, dernier accès le 7 octobre 2006

FIG. 2.15 – Exemple de portail netvibes

d'apprendre à utiliser un nouvel outil/programme/service et le lien direct entre la volonté d'un utilisateur à utiliser le système et le profit qu'il en retire. Les travaux menés durant cette thèse s'articulent autour de la conception d'un service de partage de contenu qui soit transparent et non-intrusif pour l'utilisateur.

Chapitre 3

Circulation d'informations dans un réseau P2P

3.1 Introduction

Un échange de contenu, quel-qu'il soit, se fait entre deux acteurs : un serveur et un client. Le serveur fournit un contenu dont le client est demandeur. Selon la classification proposée par Franklin et Zdonik en 1998 [Franklin 98], il est possible de faire la distinction entre deux modes de dialogues :

- le scénario « pull » où le client démarche le serveur afin de lui demander une information ;
- sa formulation duale « push » où le serveur émet directement l'information vers les clients potentiels.

Également, selon la fréquence des échanges , il est possible de distinguer deux stratégies périodique et apériodique. La première s'appliquant aux échanges s'effectuant de façon préprogrammée dans le temps tels que l'envoi de résumés de contenu de liste de diffusion (« mailing list ») et la seconde se réfèrent aux échanges impromptus tels que l'envoi de courriels. Enfin, une dernière différenciation est faite selon le nombre de personnes contactées lors des échanges. L'ensemble de cette classification est représentée sur la figure 3.1.

Cette formalisation ne peut s'appliquer directement à notre problème car elle ne concerne pas les réseaux P2P. En effet, Franklin et Zdonik [Franklin 98] introduisent conjointement à ce modèle, trois acteurs : (1) les sources de données proposant les données à diffuser, (2) les clients qui sont consommateurs de ces informations et (3) les diffuseurs d'informations qui, jouant un rôle d'intermédiaire, s'assurent de la diffusion des informations. Or, dans un système P2P les rôles 1 et 2 correspondent à un seul et même élément tandis que le 3 disparaît totalement dans le cas d'un P2P pur. Nous retiendrons cependant de ce modèle la distinction entre les méthodologies de type « push » et « pull » qui reste toujours d'actualité. Nous verrons dans ce chapitre que ces deux méthodes correspondent à deux stratégies d'échanges dans les réseaux :

1. **Les mécanismes de recherche :** l'utilisateur détenteur d'une information la publie à un endroit où elle est susceptible d'être consultée publiquement (un site Web, par exemple).

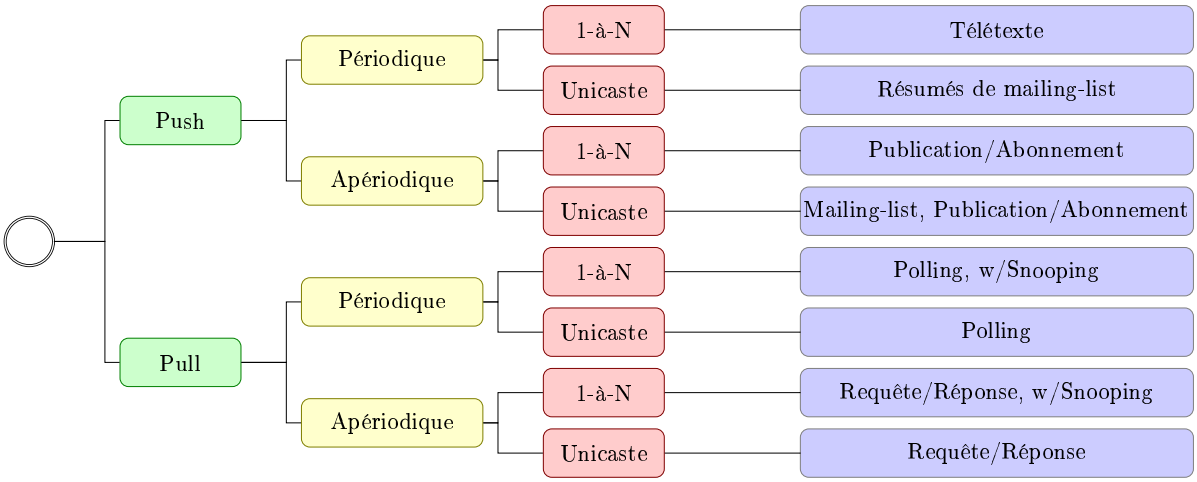


FIG. 3.1 – Mode de diffusion de données (d'après [Franklin 98])

Les personnes à la recherche de cette informations peuvent alors la trouver. C'est une stratégie de type « pull » où le transfert d'informations est à l'initiative du serveur. Dans le cas qui nous concerne, les réseaux P2P, toute entité connectée est susceptible de jouer à la fois le rôle de serveur et celui de client. Cette dualité est la cause des principaux problèmes liés à la circulation de ressources dans un réseau P2P. En effet, si tous les utilisateurs sont susceptibles d'être intéressés par une information précise, à qui l'envoyer ? De même, qui interroger lorsque tous vos interlocuteurs potentiels sont supposés capables de fournir une réponse ?

2. **Les mécanismes de diffusion :** qui correspondent à la démarche inverse. Cette fois-ci, on parle couramment de stratégie « push » : le serveur est à l'initiative du transfert. Un utilisateur détenteur d'une information va l'envoyer auprès de plusieurs autres utilisateurs. Parmi ces systèmes se trouvent notamment les mécanismes d'abonnement/publication (« Publish-subscribe ») ce qui signifie qu'un événement est publié puis récupéré par les consommateurs ayant souscrit au service de publication. En partant du principe qui veut qu'un consommateur ne va pas être intéressé par tous les événements du réseau, l'abonnement permet de faire un tri sélectif. Le problème principal posé par ces mécanismes est le niveau de « spam ». C'est-à-dire le nombre d'informations qui sont envoyées à des individus non intéressés.

Le traitement automatique de l'information nécessite une phase de modélisation. Dans ce cas précis, ce sont trois modèles qu'il est nécessaire d'établir : le réseau, les contenus échangés et les utilisateurs. Nous allons, dans les sections qui suivent, présenter différentes solutions existantes pour chacun de ces modèles. Puis, dans une seconde partie nous aborderons le sujet de la gestion dynamique de la topologie du réseau. La formation de groupes sémantiques permet en effet d'améliorer les performances du réseau.

3.2 Modélisation du système

3.2.1 Modélisation des contenus

Intéressons-nous premièrement à la modélisation des contenus échangés. Ces contenus sont représentés par un ensemble de méta-données les caractérisants. Selon les cas, il pourra par exemple s'agir d'un identifiant alphanumérique, d'une liste de mots clefs ou d'une valeur numérique. L'important est que ces méta-données fournissent une information sur le contenu représenté. Il est possible de faire la distinction entre quatre types de méta-données [Joseph 03] :

- **Valeur de hachage** : un identifiant numérique calculé via une fonction de hachage. Idéalement, la probabilité pour que deux documents obtiennent le même identifiant doit être faible ;
- **Identifiant arbitraire** : un identifiant choisi par une autorité quelconque. Pour éviter les conflits, cette autorité doit avoir une portée globale sur l'ensemble des documents ;
- **Représentation statistique** : représentation obtenue par une analyse du document. Cette analyse peut impliquer la considération d'un plus grand nombre de documents ;
- **Fixé par l'utilisateur** : méta-données définies par l'utilisateur. Par exemple, une liste de mots clefs ou une description RDF.

La mise au point de ces méta-données fait partie des trois étapes nécessaires au traitement informatisé de l'information : l'extraction d'information, l'encodage de cette informations et la comparaison entre les modèles obtenus (voir figure 3.2).

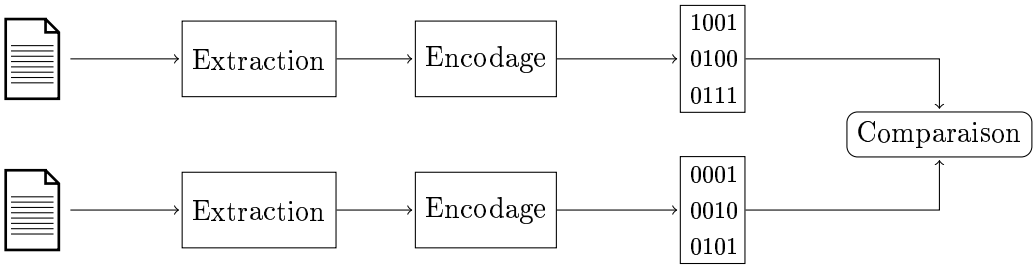


FIG. 3.2 – Étapes du processus de comparaison de deux documents

Voyons maintenant plus en détails en quoi consiste chaque étape. Nous supposerons ici que tous les contenus à encoder peuvent être modélisés sous la forme d'un ensemble de mots. Dans le cas d'une page Web, cet ensemble correspond au texte de la page duquel ont été enlevées les balises HTML. Pour d'autres types de contenus, ce texte sera supposé extrait de méta-données ou fournit par l'utilisateur.

3.2.1.1 Etape d'extraction du contenu

La première étape pour la modélisation du document est d'extraire un ensemble de mots clé. L'identification de ces mots peut mettre à contribution plusieurs algorithmes ou méthodes relatives au traitement du language naturel. Nous en présenterons ci-après quelques unes de façon succinte.

Suppression des mots vides de sens

Quelque soit la langue considérée, les documents analysés sont composés des mots formant le contenu essentiel et d'un ensemble d'autres mots servant à la formulation des phrases. En théorie, ces derniers mots, mots vides de sens (« stop words »), ne présentent que peu d'intérêt et peuvent donc être éliminés du texte. Ainsi la phrase : « Les dirigeants ont en revanche convenu de renoncer à la date butoir initialement fixée au 1er novembre 2006 pour achever les ratifications » extraite d'un site d'informations sera t'elle changée en « dirigeants revanche convenu renoncer date butoir initialement fixée 1er novembre 2006 achever ratifications ». Cet exemple met en relief un des problèmes posés par l'utilisation de listes de mots vides de sens : en retirant les mots « ont en » et en laissant uniquement « revanche » dans la liste des mots clef, le sens de la phrase est susceptible d'être changé. Le second problème est posé par la construction de la liste elle même car le choix des mots qui la compose peut prêter à discussion. Par exemple, le mot « avant », considéré comme étant vide de sens par le projet **Lucene**[1], est important lors de l'indexation de documents traitant, par exemple, de l'apparence accordée à la partie avant d'automobiles.

Lemmatisation

La lemmatisation est une opération permettant d'obtenir le radical d'un mot. Par exemple, les mots « petit », « petite », « petits » et « petites » seront tous les quatre ramenés au radical « petit ». L'objectif est de réduire la taille de l'espace de mots clé en se basant sur l'idée que les mots peuvent être remplacés par leur racine sans perdre de sens. L'algorithme de lemmatisation le plus connu est très certainement celui de Porter [Porter 80]. Ce dernier permet de lemmatiser des mots anglais sur la base d'un ensemble de règles de substitutions[2]. Le consensus général veut qu'on attribue une certaine efficacité à la lemmatisation. Cependant, étant donné que le manque de prise en compte de la sémantique des mots, des erreurs de sens peuvent apparaître. Prenons par exemple le cas des mots anglais « generic », « generous », « general » et « generation » donc la racine lexicale est commune bien que la sémantique diffère. Le lecteur intéressé pourra se référer aux travaux de Hull [Hull 96] pour consulter une analyse critique des différentes solutions de lemmatisation.

Composition de N-grams

Une fois les mots vides de sens supprimés et ceux restants lemmatisés, le document se retrouve sous la forme d'une liste de mots. La structure logique les liant entre eux n'étant plus disponible (la ponctuation a été supprimée), cette liste est en « vrac » et ne reflète pas la structure du document. Ni, dans certains cas, sa sémantique. L'utilisation de N-grams permet de garder cette informations en codant le document sous une autre forme. Une fenêtre d'une taille n est déplacée dans le texte pour extraire ces ensembles. Par exemple, la chaîne `« bonjour le monde »` donnera lieu à la suite de 3-grams (trigrams) suivants : `« bon »`, `« onj »`, `« njo »`,`« jou »`, `« our »`, `« ur »`, `« r l »`...

[1] `http://lucene.apache.org/java/docs/`, dernier accès le 7 octobre 2006

[2] La page `http://www.tartarus.org/martin/PorterStemmer/` contient une implémentation de cet algorithme pour 14 langages de programmation différents.

En composant une liste de mots clé de la sorte, il est possible de conserver l'information relative à l'ordre des mots dans les phrases. Information qui est perdue si ces mots sont considérés individuellement.

3.2.1.2 Encodage des éléments extraits

L'encodage, la seconde étape, permet de passer de l'ensemble de termes T_i à un modèle mathématique. Nous présenterons ici deux méthodes d'encodage parmi les nombreuses solutions existantes. Celle-ci produisent un vecteur dont le contenu est booléen ou composé de réels.

Encodage sous forme d'espace vectoriel (Vector Space Model - VSM)

Dans ce modèle, les documents sont représentés par un vecteur V de mots pondérés. Soit $D = \{d_1 \cdots d_n\}$ un ensemble de documents constitués chacun d'un ensemble de termes T_n. L'espace de représentation des vecteurs est défini par l'intersection de tous les termes $\bigcap_{i=1}^{n} T_n$. L'informations est modélisée sous la forme d'un sac de mots (« bag of word »).

Les vecteurs V sont utilisés pour représenter un document. Ses coordonnées v sont fixées dans le but de refléter l'importance de chaque mot. Il est, par exemple, possible d'utiliser directement la fréquence du mot dans le document considéré (table 3.1) ou une représentation binaire indiquant si le mot est présent ou pas dans le document (table 3.2).

	mot_1	mot_2	mot_3
V_1	4	3	0
V_2	2	3	2

TAB. 3.1 – Pondération en fréquence

	mot_1	mot_2	mot_3
V_1	1	1	0
V_2	1	1	1

TAB. 3.2 – Pondération binaire

On peut supposer que si un mot est très présent dans un document alors qu'il est quasi-absent des autres, celui ci sera le plus représentatif du contenu du document (discriminant). C'est en se basant sur cette idée que la pondération TFIDF est calculée. Le principe est de mettre en concurrence la fréquence du terme dans le document (« Term frequency » $tf()$) contre le nombre total de documents où ce mot apparaît au moins une fois (« Document frequency » $df()$).

$$v_{mot} = tf(mot) \log(\frac{|D|}{df(mot)}) \tag{3.1}$$

Filtres de Bloom

Les filtres de Bloom sont un outil conçu par Burton H. Bloom en 1970 [Bloom 70]. Un filtre de Bloom est un tableau binaire de m bits. Pour accompagner ce tableau, k fonctions de hachages différentes $h_1, \ldots, h_k$ sont définies. Elles permettent de mapper une clef en une des cases du tableau de bits. Lors de l'ajout d'un nouvel élément T_i, tous les bits retournés par $h_i(T_i)$ sont fixés à la valeur 1 alors que les autres sont laissés intacts (voir figure 3.3).

Ces filtres permettent de définir une notion d'appartenance d'un élément à un groupe. Lorsqu'ils sont interrogés, ils présentent l'avantage de ne jamais produire de faux négatifs bien que des

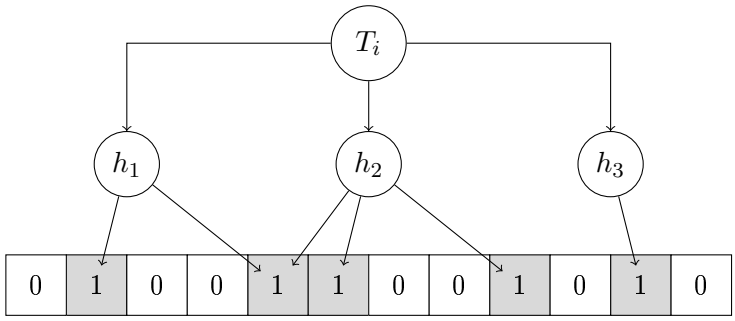


FIG. 3.3 – Exemple d'application d'un filtre de Bloom

faux positifs restent possibles. Si l'appartenance d'un élément x à un ensemble X est testée, il est impossible d'obtenir de réponse positive si $x \notin X$. Par contre, le fait d'avoir $x \notin X$, n'empêche pas d'avoir une réponse positive pour le test $x \notin X$. Pour tester la présence dans le tableau d'un élément x, il suffit de vérifier si tous les bits retournés par le calcul des $h_i(x)$ sont à 1. Si l'un d'eux est à 0 alors $x \notin X$. L'apparition de faux positifs est due aux collisions pouvant arriver lors de l'utilisation des fonctions de hachage. Dans certains cas, l'information d'appartenance est suffisante et le dérangement occasionné par les faux positifs faible. C'est, par exemple, le cas pour un système orthographique vérifiant l'appartenance d'un mot x à un dictionnaire X. Au pire, quelques erreurs ne seront pas relevées (cas de faux positifs).

3.2.1.3 Comparaison entre deux représentations de documents

Enfin, pour comparer deux éléments encodés, une mesure de (dis)similarité peut être employée. Selon le type d'encodage utilisé, le choix de la méthode sera différent.

Modèle vectoriel

En application à un modèle vectoriel tel le VSM, la mesure de similarité la plus couramment employée est celle du cosinus définie comme étant le résultat du produit scalaire entre deux vecteurs divisé par le produit de leurs normes euclidiennes.

$$cos(X,Y) = \frac{\langle X \cdot Y \rangle}{|X||Y|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \tag{3.2}$$

Une autre mesure courante est celle de la distance euclidienne

$$dist(X,Y) = |X - Y| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3.3}$$

Modèle sémantique

Si le contenu des éléments à comparer est de nature sémantique, il est possible d'utiliser des mesures se basant directement sur la signification de ce contenu. Il est ainsi possible de dénombrer le nombre de mots communs aux deux éléments. Dans le cas d'ontologies, la distance séparant les deux éléments dans l'arbre est une mesure valable [Schmitz 04]. Elle impose cependant l'utilisation d'un arbre dont l'interprétation du contenu reste constant.

Modèle booléen

La distance de Hamming peut être utilisée afin de comparer deux vecteurs de ce type. Il suffit alors de comparer une à une les valeurs de chaque vecteur et de compter le nombre de différences. Ce type de distance a par exemple été utilisé par Ganguly *et al.* pour le mécanisme de « matchmaking » de **ImmuneSearch** [Ganguly 04].

3.2.2 Modélisation du réseau

De manière générale, le terme réseau caractérise un ensemble de connexions (arcs) reliant deux à deux un ensemble d'entités (nœuds). Du point de vue matériel, l'ensemble des câbles et matériels de connexions reliant un ensemble de machines entre elles constitue un réseau informatique. Ce réseau permet à chaque machine d'échanger des informations deux à deux. Couramment, ce réseau est désigné par le terme de « réseau physique ». Les réseaux logiques viennent en complément de ce réseau physique en ajoutant une couche supplémentaire de liaisons.

Dans le cas du trafic routier, par exemple, l'ensemble du réseau permettant d'aller d'un point à un autre de la ville constitue le réseau physique alors que les parcours effectués sont représentatifs d'une structure logique. Et dans le réseau physique formé par Internet, les logiciels d'échange de fichiers tel que Napster forment un réseau logique entre les utilisateurs. Il est intéressant de noter que le réseau logique ne peut être considéré sans le réseau physique et qu'il est possible de mettre en place plusieurs réseaux logiques sur un même support physique.

3.2.2.1 Réseaux physiques et overlays

Lors de l'analyse d'un réseau, une distinction est faite entre son aspect physique et logique. Les réseaux physiques sont le résultat de l'interconnexion entre les machines. Les réseaux logiques (« overlays ») traduisent les échanges effectués entre les entités de ce réseau. Typiquement, ceux-ci mettent en œuvre des topologies qui sont très différentes de celles observées au niveau du réseau physique. Quelques réseaux physiques :

- Réseaux câblés (infrastructure)
  Un réseau câblé (« wired network ») définit une architecture fixe. Les différents éléments du réseau sont reliés entre eux de manière fixe et selon une logique prédéfinie.
- Réseaux ad-hoc
  Dans le cadre des réseaux *ad-hoc*, on suppose que les machines sont amenées à apparaître et disparaître du réseau de façon aléatoire. Les connexions entre éléments ne suivent pas

non plus de logique précise. Parmi ces réseaux, il est possible de faire une distinction selon les terminaux considérés.

- Réseaux de capteurs

  Dans un réseau de capteurs, les machines du réseau sont placées aléatoirement dans l'espace et sont relativement statiques. Des capteurs sont des éléments chargés de percevoir des informations dans l'environnement dans lequel ils se situent. Un réseau de capteurs « sensors network » se caractérise par la mise en relation d'un certain nombre de ces capteurs. La caractéristique principale de ces réseaux est que les entités sont supposées devoir diffuser un nombre conséquent d'informations et disposent de faibles capacités de calculs. De tels réseaux sont, par exemple, utilisés dans le domaine de la météorologie.

- Réseaux mobiles

  Dans un réseau mobile, ou « MaNETS », les appareils présents sont des téléphones cellulaires, des PDAs ou tout autre appareil communicant. Des ondes radio font office de support physique pour les transmissions. Celle-ci sont utilisées entre l'appareil mobile et une base radio, la suite du transit des messages pouvant être assuré par un réseau filaire. Le problème posé par le support radio est qu'un appareil hors de portée radio ne sera plus connecté au réseau. De plus, l'énérgie éléctrique consommée pour la communication sera d'autant plus grande que l'appareil sera éloigné de la base radio avec laquelle il communique. L'autonomie étant un facteur important des outils mobiles, l'optimisation de la dépense d'énérgie pour maintenir le signal radio est une contrainte supplémentaire à prendre en compte dans le cadre de la circulation d'informations.

Dans le cadre de l'étude de ces échanges, peu de travaux prennent directement en compte l'architecture physique car elle n'est pas forcément facile à connaître et surtout difficile à changer. Il est généralement préférable de ne considérer qu'un réseau logique.

3.2.2.2 Topologies des réseaux

La topologie d'un réseau désigne la façon dont sont interconnectés ses différents composants. Les principales topologies sont au nombre de 3 :

- **Non structuré**

  La topologie ainsi que le placement des données dans le réseau ne suivent aucun ordre particulier. Cette topologie est caractéristique des réseaux ad-hoc où l'arrivée et le départ stochastique des machines ne permettent pas de maintenir une structure fixe.

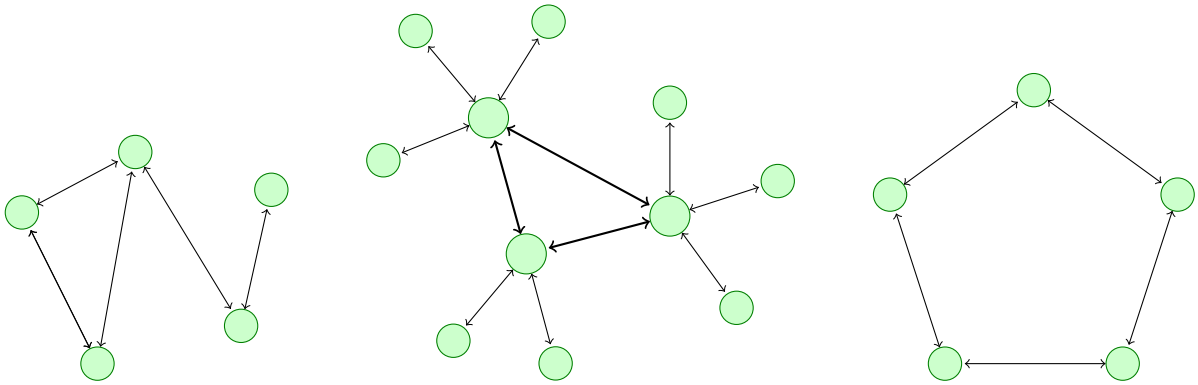
- **Faiblement structuré**

  Le réseau est constitué de plusieurs groupes non structurés reliés entre eux selon une topologie structurée. Un exemple d'utilisation de cette solution est le réseau mis en place par Gnutella quand des « superpairs » y sont ajoutés, il s'agit d'une structure hiérarchique.

- **Fortement structuré**

  Tous les pairs sont reliés entre eux selon une logique d'agencement prédéfinie. Celle-ci s'applique globalement au réseau. Tout y est parfaitement ordonné et rangé, chaque pair sait précisément où se trouve chaque donnée recherchée. C'est le cas par exemple du système CAN [Ratnasamy 01, Ratnasamy 02a].

Ces trois catégories de topologies sont schématisées sur la figure 3.4.



(a) Non Structuré (b) Faiblement Structuré (c) Structuré

FIG. 3.4 – Différentes topologies

D'un point de vue analytique, trois critères permettent de différencier ces topologies. Il s'agit de la valeur moyenne du coefficient de plus court chemin (ce qui fournit une approximation du diamètre du réseau), de la distribution des degrés et du cœfficient de groupement. Pour les définir, considérons un graphe $\mathcal{G}(V, E)$ avec $V = v_1, v_2, \ldots, v_n$ l'ensemble des nœuds et $E \subseteq V \times V$ celui des arcs. Une arc entre un nœud v_i et v_j sera noté e_{ij}, de sorte que $E(V) = \{e_{ij} | v_i, v_j \in V\}$. Soit un nœud v_i, son voisinage N_i est composé de l'ensemble des nœuds auquel v_i est directement connecté.

Definition 2 *Soit un nœud v_i. L'ensemble des nœuds constituant son voisinage est défini par :*

$$N_i = \{v_j | e_{ij} \in E\} \tag{3.4}$$

dans le cas d'un graphe dirigé, il peut également être défini comme étant l'ensemble des nœuds situés à une distance unitaire de v_i soit l'ensemble $\{v_j | D(i,j) = 1\}$. $D(i,j)$ étant la distance minimum entre deux points i et j de V.

Definition 3 *On appelle degré k_i d'un nœud v_i le nombre d'éléments que comporte son voisinage.*

$$k_i = |N_i| \tag{3.5}$$

- Distribution des degrés dans le graphe
  Les différents modèles sont comparables selon la distribution $P(k)$ des degrés k des nœuds du réseau. Le degré d'un nœud correspond à son nombre de connexions. Dans le cas où il est fait une différence entre les connexions entrantes et sortantes, deux probabilités $P_{in}(k)$ et $P_{out}(k)$ pourront respectivement être prises en compte. Afin d'illustrer ce concept de degré, prenons l'exemple d'une page Web comportant des liens hypertexte. Le nombre de liens sortants, c'est à dire pointant vers d'autres pages, correspond au degré sortant. Le degré entrant est quant à lui indiqué par le nombre de liens permettant, depuis une autre page, d'arrivée sur la page concernée.
- Calcul du coefficient de plus court chemin
  Ce coefficient mesure la plus courte distance séparant deux nœuds. Il est moyenné sur l'ensemble des résultats obtenus pour chaque paire possible. L'équation (3.6) permet de

calculer cette valeur. La valeur de C_n^2, soit le nombre de combinaisons de 2 éléments parmi n, indique le nombre total de connexions.

$$\mathcal{L} = \frac{1}{C_n^2} \sum_{i,j \in V} D(i,j) \tag{3.6}$$

- Calcul du coefficient de groupement
  Ce second cœfficient permet de mesurer le taux de groupement du graphe. Pour un nœud v_i, il correspond au rapport entre le nombre total de liens existants entre les membres de son voisinage et le nombre maximum de liens possibles. Un voisinage composé de k_i nœuds, a un nombre maximal de connexions égal à $k_i(k_i - 1)$.

$$C_i = \frac{|E(N_i)|}{k_i(k_i - 1)} \tag{3.7}$$

  Dans le cas d'un graphe non-dirigé, $e_{ij} = e_{ji}$ le nombre maximum de liens est donc égal à $\frac{k_i(k_i-1)}{2}$. Le coefficient de groupement du réseau $\mathcal{C}$ est obtenu en moyennant les valeurs des C_i obtenues pour chaque nœud du graphe (equation 3.8).

$$\mathcal{C} = \frac{1}{|V|} \sum_{v_i \in V} C_i \tag{3.8}$$

  Cette définition peut être étendue de façon à prendre en compte des arcs valués. Schmitz [Schmitz 04] propose une formulation basée sur la mesure de similarité entre deux nœuds.

$$\gamma_v^w = \frac{1}{k_v(k_v - 1)} \sum_{w \in \Gamma(v)} sim(v,w) |\{u \in \Gamma(v) : (w,v) \in E\}| \tag{3.9}$$

  avec
  – $\Gamma(v) = \{u \in V \backslash \{v\} : (v,u) \in E\}$ les nœuds auxquels v est connecté.
  – $k_v = |\Gamma(v)|$ la taille du voisinage.
  Le nombre maximum de liens possibles dans le voisinage de v est de $k_v(k_v - 1)$ dans le cadre d'un graphe dirigé où un maximum de deux connexions peut être établi entre deux pairs. Si $\Gamma_v = 0$, aucun des voisins de v n'a de connexion avec un autre des voisins. Si $\Gamma_v = 1$, tous les voisins sont interconnectés entre eux.

L'exemple numérique ci-après permet d'illustrer le calcul de ces cœfficients.

$$V = \{v_1, v_2, v_3, v_4\} \qquad E = \left\{ \begin{array}{l} (v_1,v_2),(v_2,v_3),(v_2,v_4),(v_3,v_5),(v_4,v_1) \\ (v_4,v_2),(v_4,v_3),(v_4,v_5),(v_5,v_3),(v_5,v_4) \end{array} \right\}$$

En considérant le nœud v_4 on obtient le voisinage $\Gamma(v_4) = \{v_1, v_2, v_3, v_5\}$ qui a une taille $k_{v_4} = |\Gamma(v_4)| = 4$.

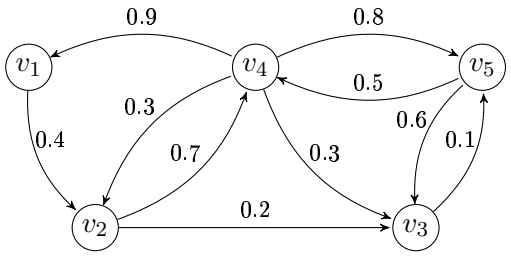


FIG. 3.5 – Exemple de graphe pour le calcul de coefficients

$$\begin{aligned}
\gamma_{v_4} &= \frac{1}{4(4-1)} \sum_{w \in \{v_1,v_2,v_3,v_5\}} |\{u \in \{v_1,v_2,v_3,v_5\} : (w,u) \in E\}| \\
&= \frac{1}{12}(|\{(v_1,v_2)\}| + |\{(v_2,v_3)\}| + |\{(v_3,v_5)\}| + |\{(v_5,v_3)\}|) \\
&= \frac{1}{3} = \mathbf{0.33}
\end{aligned}$$

$$\begin{aligned}
\gamma_{v_4}^{w} &= \frac{1}{4(4-1)} \sum_{w \in \{v_1,v_2,v_3,v_5\}} sim(v,w)|\{u \in \{v_1,v_2,v_3,v_5\} : (w,u) \in E\}| \\
&= \frac{1}{12}(0.9 \times |\{(v_1,v_2)\}| + 0.3 \times |\{(v_2,v_3)\}| + 0.3 \times |\{(v_3,v_5)\}| + 0.8 \times |\{(v_5,v_3)\}|) \\
&= \frac{2.3}{12} = \mathbf{0.19}
\end{aligned}$$

Nous présentons ci-dessous une rapide étude de trois modèles de distribution : aléatoire, sans échelle (« scale free ») et petit monde (« small world »). Le lecteur intéressé pourra consulter les travaux de Barabási [Albert 02] pour une étude plus complète de toutes les propriétés de ces modèles.

3.2.2.3 Réseaux aléatoires (« Random Network »)

Les réseaux aléatoires ont été le premier modèle mis au point. Il est dû à Erdös et Renyi qui en 1959 ont été les premiers à étudier la distribution des degrés dans un graphe aléatoire. La recette d'un graphe aléatoire est simple : il suffit de prendre un ensemble de n nœuds et d'ajouter des connexions entre ces nœuds de manière aléatoire. Selon la loi utilisée pour gouverner cet aléatoire, différents modèles de graphes, avec différentes distributions pour les degrés, peuvent être produits.

Nous reprendrons ici l'exemple numérique de [Albert 02] pour le calcul de la loi de distribution des degrés. Soit un graphe aléatoire non orienté dont la probabilité de connexion est p. Soit également un nœud i de degré k_i et un degré de distribution k. La probabilité pour avoir $k_i = k$ peut être formulée comme étant le produit entre la probabilité pour que i ait exactement k connexions et le nombre de choix possibles pour les k voisins. Etant donné qu'il y a N pairs, ce choix est égal au nombre de combinaisons de k éléments parmi $N-1$ (on ne compte pas i),

soit C_{N-1}^{k}. La probabilité pour que i ait exactement k connexions est égal au produit entre la probabilité d'avoir k connexions, soit p^k, et la probabilité pour que les $N-k-1$ autres connexions ne soient pas établies : $(1-p)^{N-1-k}$. Au final, nous obtenons la formulation suivante :

$$P(k_i = k) = C_{N-1}^{k} p^k (1-p)^{N-1-k}$$

Soit X_k le nombre de pairs de degré k, l'espérance de cette variable est de $E(X_k) = NP(k_i = k) = \lambda_k$ avec $\lambda_k = NC_{N-1}^{k} p^k (1-p)^{N-1-k}$. Sur la base de la condition d'existence de sous-graphes dans les graphes aléatoires, il est alors possible de démontrer que la distribution des X_k approche une loi de poisson de paramètre λ_k :

$$P(X_k = r) = e^{-\lambda_k} \frac{{\lambda_k}^r}{r!}$$

La probabilité de distribution des degrés obtenue indique que, dans ce type de graphe, peu de pairs sont connectés à très peu de voisins ou un nombre important. Pour la majorité des nœuds, le degré est le même, donc le graphe est homogène.

A l'exact opposé du réseau aléatoire se trouve le réseau régulier. Tous les nœuds ont le même degré et les connexions sont établies selon un schéma particulier. Un exemple de tel réseau est celui de l'anneau où les nœuds sont disposés en cercle, chacun relié à son successeur et à son prédécesseur.

3.2.2.4 Réseaux sans échelle (« Scale-Free ») (SFN)

La distribution homogène des degrés dans un graphe aléatoire ne correspond pas à la réalité d'un graphe tel que celui des connexions entre les pages Web sur Internet. Il a été prouvé que pour de tels graphes la répartition des degrés suit une loi puissance. Cette répartition conduit à la création de réseaux dits « sans échelle » (« scale-free »). Dans ces réseaux, plusieurs nœuds agissent comme des « hub » ayant un nombre important de connexions alors que la majorité des autres nœuds ont un degré plus faible.

Pour produire de tel réseaux, Barabási et Albert ont proposé un modèle de construction simple [Barabasi 99]. Celui se base sur une idée de « croissance » et « d'attachement préférentiel » traduisant respectivement le fait que, dans un cas réel, un graphe est construit à partir d'un autre graphe et que la probabilité d'établir une nouvelle connexion est en fait variable selon le degré actuel du nœud. L'algorithme est donc le suivant :

1. *Croissance :* Partir d'un graphe contenant un petit nombre de nœuds m_0. A chaque itération, ajouter un nouveau nœud et $m \leq m_0$ connexions pour le relier aux autres nœuds déjà présents.
2. *Attachement préférentiel :* Lors de l'établissement de ces m connexions, la probabilité de connexion p est définie en fonction du degré k_i du nœud i.

$$p(k_i) = \frac{k_i}{\sum_j k_j}$$

Au bout de t itérations, cet algorithme produit un graphe contenant $N = t + m_0$ nœuds et mt connexions. La distribution des degrés qui en résulte suit alors une probabilité correspondant à une loi puissance ayant pour paramètres $C = 2m^2$ et $\gamma = 3$.

$$P(k_i = k) = Ck^{-\gamma}$$

Ce qui revient à dire que si tous les nœuds étaient classés selon leur k, le plus connecté aurait $k^{-\gamma}$ voisins.

3.2.2.5 Réseaux petit monde (SWN)

Les premiers travaux sur ces structures ont été menées par Watts et Strogatz [Watts 98]. La topologie des réseaux petits mondes se caractérise par deux particularités :

- la plupart des nœuds sont voisins d'autres nœuds ;
- deux nœuds quelconques ne sont distants que de quelques sauts. Depuis un nœud de départ, il est possible de se rendre vers n'importe quel autre nœud en un minimum de déplacements.

Ces deux propriétés confèrent aux réseaux petits mondes un graphe composé de plusieurs groupements denses de connexions interconnectés entre eux (figure 3.6). Ceci les situe entre les graphes aléatoires et réguliers. La recette proposée par Watts et Strogatz pour construire un réseau petit monde s'inspire d'ailleurs de cette observation. A l'instar de la création des graphes invariant d'échelle, cette recette comporte deux étapes :

1. *Démarrage ordonné :* Partir d'un graphe régulier en anneau composé de N nœuds chacun connecté à ses K plus proches voisins ($K/2$ de chaque côté).
2. *Brassage des connexions :* Modifier chaque connexion du graphe avec une probabilité p tout en excluant les bouclages (un nœud ne peut se connecter à lui même) et les duplications de liens (deux nœuds ne peuvent pas être reliés par plus d'une connexion).

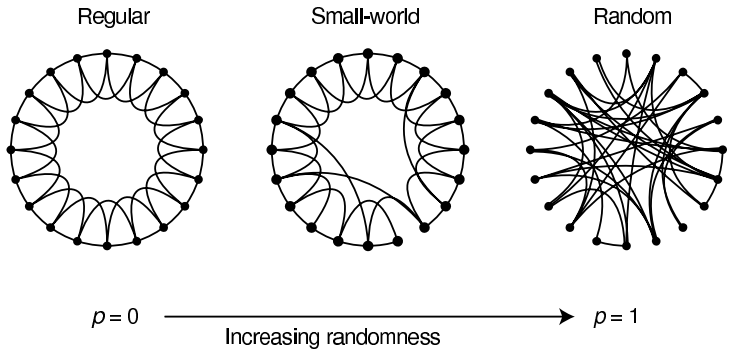


FIG. 3.6 – Les « Small World » : entre ordre et désordre [Watts 98]

3.2.3 Modélisation des utilisateurs

Outre le réseau et les informations à y échanger, il est nécessaire de transcrire sous une forme interprétable par la machine les besoins de l'utilisateur. Ceci donne lieu à la création de profils. Selon l'activité considérée, il est possible de définir des profils portant sur les centres d'intérêt, des profils de recherche, des profils d'utilisation du Web...

A notre modélisation basée sur les documents et le réseau vient s'ajouter un modèle de l'utilisateur. Ce modèle tient en la définition de deux éléments :

- un profil e représentatif des centres d'intérêt d'une personne ;
- une similarité $sim(e, e)$ permettant d'avoir une estimation de la corrélation entre les centres d'intérêt de deux personnes différentes.

Le profil d'un utilisateur est utilisé par les outils de recherche d'informations pour modéliser ses centres d'intérêt. Ce profil doit contenir des données pouvant être utilisées pour effectuer des requêtes dans les index de documents. Les requêtes étant faite en effectuant des mesures de similarité entre les vecteurs des documents et celui de la requête, la méthodologie usuelle est de modéliser un profil sous la forme d'un vecteur de mots au même titre que les informations stockées dans les entrepôts de données.

Les profils peuvent être assimilés à des méta-données concernant l'utilisateur. En ce sens, leur contenu est similaire à ceux observés pour les documents.

La construction du profil d'un utilisateur peut être envisagée d'un point de vue manuel, semi-automatique ou automatique.

- **Manuelle** La première catégorie fait référence aux systèmes interrogeant l'utilisateur sur ses centres d'intérêt. Lors de l'étape de configuration de l'outil d'aide, l'utilisateur est invité à saisir un ensemble de mots clef. Cette solution est utilisée par le système collaboratif **Yenta**. L'avantage de cette pratique est que l'utilisateur saisi directement les mots clé qu'il désire. Cette saisie de mots clés constitue aussi une source de faiblesse de cette technique car si un mot est indiqué, son synonyme pourrait être omis.
- **Semi automatique** Pour simplifier la tâche de l'utilisateur, une solution semi automatique consiste à demander une liste de pages dites de référence à la place d'une liste de mots. Cette liste de pages devra être choisie par l'utilisateur comme étant particulièrement représentative de ses centres d'intérêt. Une analyse lexicale des documents permet d'en extraire les différents mots clés à utiliser.
- **Automatique**
  Cette dernière option se place dans le cadre des interfaces homme-machine à entrée nulle de données (Zero Input Interfaces). Les systèmes basés autour de ce principe n'utilisent pas d'interface de dialogue avec l'utilisateur pour l'acquisition de données. Ces boîtes de dialogues sont remplacées par un ensemble de capteurs chargés de s'insérer dans l'environnement de travail de l'utilisateur sans changer ses habitudes. Selon les capteurs utilisés, la liste des pages Internet consultées, les fichiers ouverts ou d'autres informations de nature à indiquer les centres d'intérêt de l'internaute peuvent être captés et exploités. Les projets ReferralWeb [Kautz 97] et, plus récemment, les travaux menés par Eagle *et al.* [Eagle 02] illustrent ce principe de fonctionnement. Le profil de l'utilisateur est respectivement construit en observant les pages consultées sur Internet ou en analysant le contenu des conversations téléphoniques.

Une même personne ne peut avoir qu'un seul et unique centre d'intérêt. Ne construire qu'un seul profil revient donc soit à considérer que l'ensemble de ces mots clé peut être représenté par une unique liste de mots clé, soit à ne prendre en compte qu'un unique centre d'intérêt « principal ». Pour palier ce problème, il est possible d'avoir recours à différentes stratégies de gestion de profils multiples :

- **Profils à court et long terme** : La définition des profils à court et long terme permet de différencier les besoins ponctuels de l'utilisateur de ses centres d'intérêts ;
- **Profils dupliqués** : Dans ce cas, il est fait utilisation de plusieurs vecteurs pour traduire l'ensemble des centres d'intérêt de l'utilisateur [Chen 98] ;
- **Profils hiérarchiques** : Il est également possible de considérer que les centres d'intérêts d'un utilisateur peuvent être classés par spécialisations successives. Par exemple, un utilisateur interessé par le modélisme en général pourra être plus particulièrement attiré par la création d'avions miniaturisés. Ceci décrit alors deux centres d'intérêts « modélisme » et « aéromodélisme », le second étant une sous-partie du premier ;
- **Profil de recherche et profil informatif** : Le profil de l'utilisateur peut être différent de ce qu'il cherche. Afin de prendre cela en compte, il est possible de définir un profil de recherche et un autre destiné à informer les autres utilisateurs sur les centres d'intérêts. C'est, par exemple, la solution adoptée par le système ImmuneSearch.

3.3 Circulation d'informations

Après avoir présenté les notions relatives à la spécification des utilisateurs, des données à échanger et du réseau, passons à l'étude de la dynamique de ce système. Il s'agit de décrire les outils et méthode permettant de mettre en place la circulation d'informations entre utilisateurs *via* le réseau.

Plusieurs solutions seront ci-après présentées. Nous en proposons une classification selon le schéma de la figure 3.7.

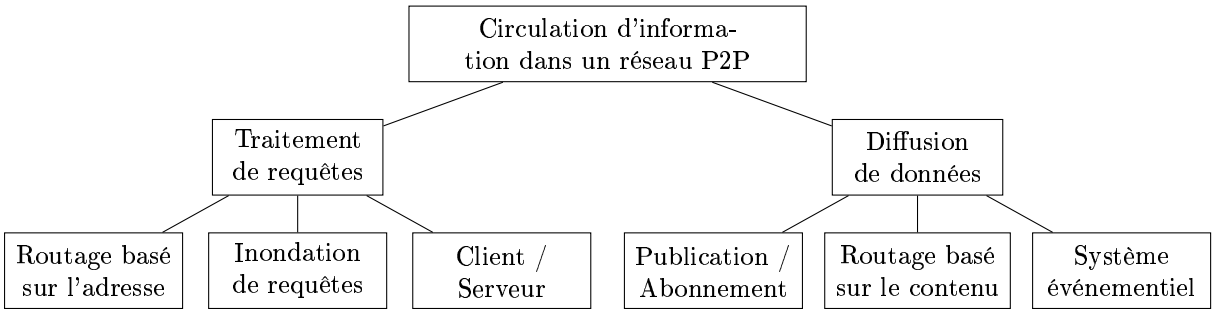


FIG. 3.7 – Classification des systèmes de circulation de l'information. Le premier niveau de distinction correspond à l'initiative de l'échange (client / serveur). Le second niveau indique la nature du réseau (structuré / non structuré / hybride).

3.3.1 Recherche d'informations (« pull »)

Nous commencerons par nous intéresser au problème de la recherche de contenu. Cette démarche peut être modélisée sous la forme d'un processus composé de trois sous-processus répondant chacun à une de ces questions « Quoi ? », « Où ? » et « Comment ? » [Joseph 03] :

1. **« Quoi ? » : rechercher les méta-données associées au document**
   Cette première étape du processus de recherche consiste, en partant d'une requête de l'utilisateur, à savoir quelles sont les méta-données des documents recherchés.

2. **« Où ? » : localiser le document**
   Maintenant que le document voulu est identifié, il s'agit de le localiser dans le réseau. Dans les architectures réseau offrant une gestion décentralisée de l'information, cette étape est la plus problématique. La nature changeante de la topologie du réseau rend difficile cette localisation.
3. **« Comment ? » : récupérer le document**
   Une fois le document identifié et localisé, cette dernière étape consiste à en télécharger le contenu.

Dans ce modèle, l'étape 1) peut éventuellement être supprimée ou fusionnée avec la 2) en présence de solutions où la requête de l'utilisateur est directement représentée sous la forme de méta-données. Si les méta-données sont une liste de mots clef associées aux documents et que la requête de l'utilisateur est également une liste de mots clef, seule la localisation est nécessaire.

La dernière étape, celle de transfert du document ne sera pas développée ici car peu de systèmes différents existent. La solution communément adoptée est d'opter pour un échange de type client/serveur entre la machine détentrice du document et celle émétrice de la requête. Cette étape est pour beaucoup d'algorithmes l'occasion de récupérer un « feedback » de l'utilisateur : en récupérant des informations de satisfaction de ce dernier, ils sont capables d'améliorer leurs paramètres de recherches.

Le point le plus délicat concerne la réponse à la question « Où ? » et qui généralement se pose dans les termes suivants : « Quels sont les pairs proposant un document X ? ». Il s'agit d'un opération dite de « lookup » pour laquelle plusieurs solutions utilisant, par exemple, de schémas et des indexes et qui s'apparentent pour la plupart à une question de routage, existent. Le lecteur intéressé est invité à consulter l'étude très complète de Risson & Moors sur la recherche d'informations dans les réseaux P2P [Risson 04]

3.3.1.1 Dans un réseau P2P structuré

La particularité d'un réseau P2P (fortement) structuré est que les connexions entre les pairs répondent à une logique très précise. Ces relations étant établies selon un algorithme fixe que tous les pairs connaissent, il est possible pour un pair donné de fournir des informations sur le contenu de ses voisins.

Les stratégies de routage s'appliquent particulièrement bien dans les réseaux structurés où il existe des liens logiques entre les pairs. La recherche d'informations revient à effectuer une opération de routage. Tout message circulant dans le réseau a un destinataire prédéfini. Les réseaux P2P structurés sont la solution la plus efficace à l'heure actuelle pour la localisation d'informations. Il s'agit d'un réseau pur tel que présenté sur lequel est ajouté une structure logique. Cette structure peut ne prendre en compte que les pairs ou également les fichiers qu'ils proposent. Ci-après, nous en présentons deux exemples.

Table de hachage distribuées (DHT)

Une première structure possible est celle de la table de hachage distribuée illustrée par les projets Chord, Can et Pastry. Une table de hachage est une structure basée sur un outil mathématique (la fonction de hachage) permettant d'assigner des valeurs v à des clefs k. La table elle

même est constituée de l'ensemble de ces associations clef,valeur. Dans les systèmes distribués, chaque nœud est chargé de stocker une partie de cette table. La différence entre les projets se situe sur la manière dont cette répartition est faite.

Chord : La structure proposée par Stoica *et al.* dans Chord [IonStoica 01] est un espace à une dimension représentant les pairs ainsi que les fichiers qu'ils proposent. Pour ce faire une fonction de hachage est utilisée pour associer à chaque nœud et fichier un identifiant codé sur m bits. La valeur de m étant choisie suffisament grande pour que la probabilité que deux fichiers ou pairs aient le même identifiant soit négligeable. Afin d'avoir un espace de représentation circulaire, le resultat de la fonction de hachage est calculé modulo 2^m. Chaque clef k est assignée au premier nœud qui le suit dans l'espace des identifiants. Ce nœud est désigné par $successeur(k)$. En représentant la position des pairs sur le cercle des valeurs d'indentifiant, $successeur(k)$ est le premier nœud rencontré en parcourant le cercle dans le sens des aiguilles d'une montre. Chaque nœud ne connaît du réseau que son successeur.

La figure 3.8 représente le cercle d'identifiant obtenu pour une valeur de m égale à 3, soit un espace d'identifiants contenant 8 valeurs. 3 nœuds sont présents dans le réseau avec les identifiants 0, 1 et 3, il s'agit des cercles verts. Les valeurs encadrées indiquent pour chacun les clef qu'ils contiennent. Il s'agit des clefs associées à 3 ressources présentes dans le réseau.

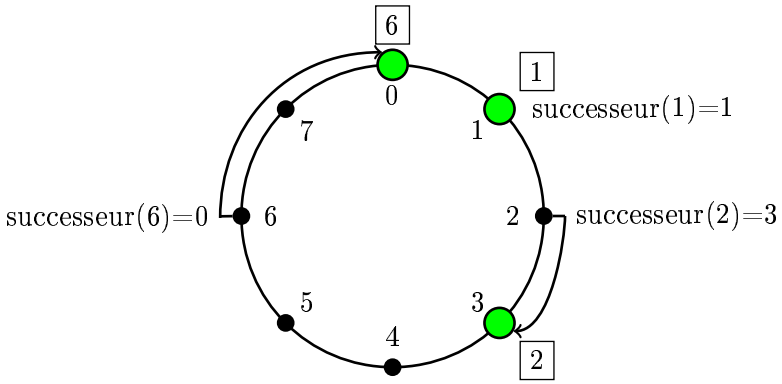


FIG. 3.8 – Distributions des identifiants dans Chord (d'après [IonStoica 01])

La recherche d'une ressource X depuis un nœud d'identifiant i_1 s'effectue en calculant l'identifiant i_2 qui lui est associé. Si $i_2 > i_1$, la clef est stockée sur un autre nœud, la requête est donc envoyée au nœud suivant.

Can : A la différence de Chord, le système CAN (Content Addressable Network) [Ratnasamy 01, Ratnasamy 02a] se base sur un espace cartésien de dimension d et une fonction de translation associant à une valeur de clef k un point P de cet espace.

Chaque pair a la charge d'une partie de l'ensemble de l'espace. La répartition de ces zones est faite par découpage successifs : un point P choisi au hasard par un nœud arrivant dans le réseau indique quelle sera la zone à partager. Une fois ce découpage effectué, le nœud responsable de la zone et le nouveau s'occupent chacun d'une des sous parties. La figure 3.9 montre un exemple de découpage effectué entre 5 nœuds dans un espace de dimension 2. Un point P de coordonnées (0.75,0.6) est également représenté : celui-ci se trouve dans la zone contrôlée par le nœud « E ».

Lors de la recherche d'une valeur v associée à un k donné, la requête est passée de voisins en

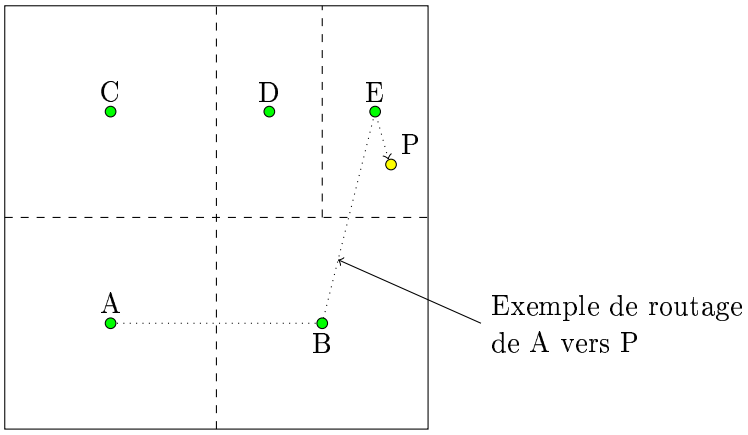


FIG. 3.9 – Distribution des identifiants dans CAN

voisins depuis le nœud source. Le choix du chemin est fait de façon à minimiser la distance entre le point courant et la destination.

Routage de Plaxton et al.

Plaxton *et al.* [Plaxton 97] proposent une solution de routage basée sur l'utilisation de préfixes/suffixes. Les valeurs retournées par la fonction de hachage sont un code composé de n éléments pouvant chacun prendre une valeur allant de 0 à F. Chaque nœud N maintient une table de voisinage indiquant le nom de ses proches voisins. La table est découpée en différents niveaux de 1 à n contenant différentes entrées. L'entrée i du niveau j indique l'emplacement du nœud le plus proche dont l'identifiant se termine par $i + suffixe(N, j-1)$. Par exemple, la 9^{eme} entre du 4^{eme} niveau du nœud `325AE` est le nœud le plus proche dont l'identifiant se termine par `95AE`.

La figure 3.10 est un exemple de réseau obtenu pour $n = 4$. Les flèches plus épaissent indiquent le chemin parcouru pour router un paquet partant du nœud `0325` et à destination du nœud `4598`. En sortie du nœud `0325`, deux choix son possible. On suppose que la connexion avec `B4F8` est celle offrant une latence moindre, elle est donc préféré à celle avec `2BB8`.

Un nœud S voulant indiquer qu'il possède un objet O envoie un message à destination d'un nœud racine de O. Il s'agit de celui contenant le sous arbre complet permettant d'accéder à O. Chaque pair relayant le message va stocker d'informations une information de type `<Object-ID(O), Server-ID(O)>`. Dans le cas où cette information serait déjà présente, seule l'adresse de la copie la plus proche est gardée. Ces paires de valeurs peuvent être assimilées à des pointeurs indiquant pour chaque objet O le serveur S le plus proche où il est stocké. Concrètement, à l'insertion, le routage est effectué de nœud jusqu'à ce qu'il n'y ait plus de voisin disponible.

Tapestry [Zhao 01] est un système basé sur [Plaxton 97] et qui se propose d'en corriger plusieurs inconvénients : le besoin d'une connaissance globale (1), la vulnérabilité du nœud racine (2) et le manque d'adaptabilité de l'algorithme de routage (3).

1. L'arrivée et le départ fréquents de nœuds dans le réseau rendent aléatoire le placement des nœuds racine dans la topologie. Une alternative incrémentale est proposée pour choisir ce

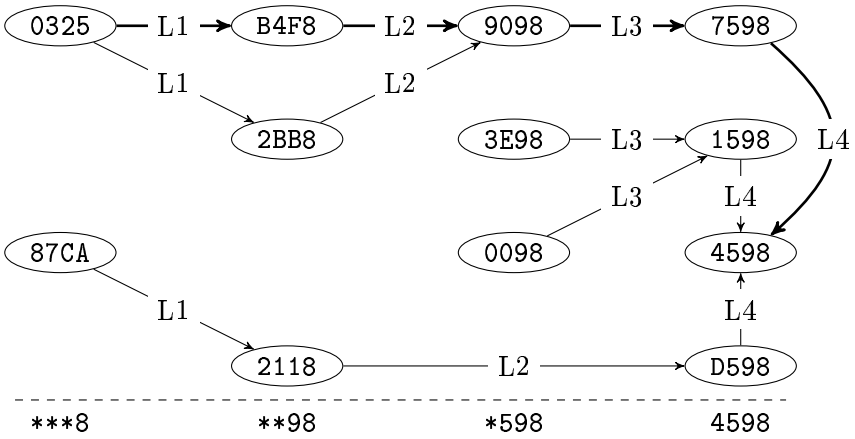


FIG. 3.10 – Exemple de routage selon l'algorithme de Plaxton *et al.* (adapté de [Plaxton 97])

nœud. Un objet O devant être inseré dans le réseau va envoyer un message à destination du nœud ayant le même identifiant que lui, c'est à dire tel que Object-ID(O) = Server-ID(O). Il se peut que ce nœud n'existe pas, ce qui implique que des nœuds ne pourront fournir de route pour le contacter. Dans tous ces cas, le message est transferé en choisissant une route parmi celles possibles. Le routage s'arrête au premier nœud dont la table de routage est non vide.

2. Par sa nature unique, le nœud racine présentant un point faible, Tapestry étend l'ID de l'objet en lui concaténant une série de valeurs fixes. Chaque nouvel identifiant obtenu est alors utilisé selon le fonctionnement prévu au départ. Un même objet dispose alors d'alias répartis dans le réseau. Au cas où un des nœud racine tomberait en panne, un de ces alias peut être utilisé pour localiser l'objet.
3. Dans les tables de routages proposées par Plaxton, seule l'adresse la plus proche de chaque objet est mémorisée. Tapestry étend cette table en gardant l'adresse de toute les copies localisées. Lors d'une requête, la réponse renvoyée par le nœud n'est donc plus unique mais multiple. Le nœud demandeur peut alors faire un choix parmi les différentes adresses proposées.

Le système de routage **Pastry** [Rowstron 01] vise également les mêmes objectifs que Tapestry mais en apportant des réponses différentes. Pour palier le problème de connaissance globale, la publication d'un objet O est accompagnée par la publication d'un certain nombre de duplicatas. Ceux-ci sont placés sur les nœuds dont l'identifiant est le plus proche de celui de l'objet. De plus, une notion de localité basée sur la prise en compte de la distance géographique ou du nombre de saut IP séparant les pairs est prise en compte.

3.3.1.2 Dans un réseau P2P non structuré

Dans un réseau P2P non structuré, le placement des données ainsi que celui des pairs ne suit aucun ordre particulier. Dans ces conditions, il est impossible de prévoir une destination à un message et donc d'effectuer du routage. Pour qu'un message arrive à faire ce qu'il faut, il est nécessaire de le dupliquer et de l'envoyer à plusieurs endroits dans le réseau. Chaque pair recevant un message le duplique puis choisit un ensemble de pairs à qui le transférer. On parle alors de transfert de requêtes. Les messages sont envoyés de pairs en pairs durant un certain temps limité

(TTL). A défaut de destination il est possible de définir un but aux messages. L'évaluation de la distance séparant le message de son but pouvant être, par exemple, faite avec une mesure de similarité telle que le cosinus présenté plus haut. La technique employé pour faire transiter les messages est dite de transfert (« forward ») : chaque pair choisi un ou plusieurs pairs auquel il transfert les messages. A chaque transfert, une valeur de durée de vie (TTL) du message est décrémentée. Quand cette durée de vie devient nulle, le message est supprimé.

Nous pouvons ici distinguer 2 stratégies pour le routage de la requête dans le réseau : la marche aléatoire (« Random Walk ») et la diffusion (« Flooding »).

Routage par marche aléatoire

Une des méthodologies les plus courantes est celle du « random walk », ou « marche aléatoire ». Un message est transféré de pair en pair, aléatoirement, jusqu'à ce son objectif soit atteint. Il peut par exemple s'agir d'une requête visant à trouver un fichier, auquel cas l'objectif sera la découverte d'une ressource adéquate, ou ce peut être un message d'avertissement se concluant sur une approbation de la machine l'ayant reçu.

Un premier exemple de système basé sur ce principe est **Anthill** proposé par Babaoglu *et al.* [Babaoglu 01, Babaoglu 02, Montresor 01]. Ce modèle se base sur une analogie avec les fourmis : chaque nœud dans le réseau devient un nid et les messages échangés sont les fourmis. Trois types de fourmis « insertAnt », « searchAnt » et « replyAnt » sont respectivement associées à la publication de document, la recherche et le retour de réponse. Les documents échangés sont représentés par un ensemble de mot clés mais, pour réduire la taille de l'espace, c'est le résultat d'une fonction de hachage qui est utilisée à la place du mot lui même. Chaque nœud contient une table associant valeurs de hachage de mot et liste de nœuds.

A la publication d'un document, une fourmi « insertAnt » est générée pour chaque mot clef associé au nouveau document. Celles-ci se promènent de pair en pair suivant les routes indiquées par les tables de hachage de chaque nœud. A son passage dans un nœud, la table de hachage est actualisée et l'URL du document ainsi que sa valeur de hachage est enregistrée. La recherche de document entraîne la création de « searchAnt », une par mot clef contenu dans la requête. A la manière des « insertAnt », ces fourmis vont de pair en pair. Chaque réponse positive d'un pair entraîne la création d'une « replyAnt » chargée de rapporter l'URL du document à l'émetteur de la requête.

Dans un autre registre, il existe le système **Freenet** qui est lui destiné à fournir un environnement de publication libre de toute censure. Ses algorithmes sont axés sur l'anonymat et le chiffrage des données. Son principe de fonctionnement est similaire à Anthill si ce n'est au niveau de représentation des données. Alors que Anthill considère la valeur de hachage de chaque mot associé aux fichiers, Freenet leur associe une clef indépendante de l'emplacement. Chaque pair du réseau contient un dépôt de clef ainsi qu'une table de routage indiquant l'adresse d'autres pairs ainsi que les clef qu'ils sont supposés détenir. Un système de cache permet d'assurer la disponibilité des fichiers ainsi que l'optimisation des requêtes.

Les déplacements sont dictés, aussi bien dans Anthill que dans Freenet, en fonction d'une valeur de clé indépendante de la sémantique. Certains auteurs ont cherché à prendre en compte cette sémantique. Un exemple de résultat est **REMINDIN'** [Tempich 04]. Celui-ci se présente

comme une solution de routage pour les requêtes sémantiques. Les auteurs font état de 5 aspects sociaux de la recherche d'informations par un utilisateur : (1) Une question est posée à une personne susceptible d'y répondre[3] (2) En répondant à cette question, une personne est supposée avoir des connaissances dans le domaine concerné (3) Une personne est bien informée sur un domaine particulier le sera également sur un autre connexe ou plus général (4) Les connaissances dans des domaines particuliers varient d'une personne à une autre (5) L'estimation du savoir d'une personne est une mesure subjective relative aux connaissances personnelles. La prise en compte de ces aspects passe par deux aspects essentiels de l'algorithme : la mémorisation de la provenance des informations circulant et une notion de confiance entre les pairs. En particulier, chaque requête traitée par un pair est stockée en y associant le nom de l'émetteur. Lors du transfert d'une requête, le choix des pairs de destination est fait en fonction de ces informations collectées. Des tests effectuées sur l'annuaire DMOZ témoignent de l'efficacité de cette méthode comparativement à une méthode de choix aléatoire. Paradoxalement, l'ajout d'une certaine part d'aléatoire permet d'apporter de nettes améliorations à l'algorithme.

Diffusion de requête (Flooding)

L'exemple le plus représentatif de cette classe d'algorithmes est le logiciel **Gnutella**. Il s'agit d'un logiciel d'échange de fichiers basé sur cette architecture réseau. Il a été développé par la société Nullsoft (filière d'America OnLine -AOL) et demeurait en 2006 parmi les 4 solutions de partage les plus utilisées. La diffusion des requêtes y est assurée selon une méthode de Flooding. Chaque paquet émit sur le réseau contient un paramètre de durée de vie (TTL) qui diminue à chaque envoi, il est transmis de pair en pair jusqu'à expiration de cette valeur. En fait, le Flooding est équivalent à un mécanisme de Gossip où la probabilité de retransmission est fixée à 1.

Quand un pair arrive, celui ci se connecte à un certain nombre de machines et leur envoi un message de présence (Ping) indiquant le nombre de fichiers partagés ainsi que le volume de données que cela représente. Les machines contactées répondent par un message similaire (Pong). Une fois connecté, l'utilisateur peut donc avoir une estimation du volume de données disponible. La gestion des requêtes suit un principe similaire à la différence qu'un pair n'est pas tenu d'envoyer un message de réponse négative.

Pour diminuer la charge du réseau, Gnutella propose l'utilisation de super pairs. Il s'agit de machines destinées uniquement à faire du routage de paquets et à maintenir un certain niveau de cache. Les réseaux Gnutella+superpairs sont assimilables à des réseaux P2P faiblement structurés. Les algorithmes de Flooding ont pour principal inconvénient de mal gérer le passage à l'échelle et de générer du flux réseau inutile.

3.3.2 Diffusion active d'informations (« push »)

Intéressons nous maintenant au problème dual de la résolution de requête, celui de la diffusion. Contrairement aux mécanismes « pull », il n'y a pas ici de notion d'interrogation (sous la forme de requête ponctuelle[4]) et la circulation n'est pas la conséquence d'une action directe de l'utili-

[3] la plus cultivée par exemple mais ce peut également être la plus proche dans le réseau

[4] Cette précision est importante, comme nous le verrons par la suite

sateur. Le problème se pose sous la forme suivante : un ensemble de pairs connectés ensembles via un réseau et disposant chacun de données d'informations stockées. Il porte généralement la dénomination de « Selective informations dissemination (SDI) » ou « selective informations push » [Koubarakis 03, Franklin 98].

Le but est d'optimiser la distribution de ces données de manière à maximiser un critère. Exemples de critères : la satisfaction (informelle) des utilisateurs pour les systèmes de diffusion de nouvelles (voir Newscast) et la charge processeur pour les systèmes d'équilibrage de charge dans les réseaux. Typiquement, chaque pair a une vue partielle du réseau ainsi que des informations à dispositions. Dès que le nombre de pairs dans le système devient important, cet objectif fait apparaître deux problèmes distincts :

- la diffusion des informations
- la gestion de la liste des membres : choisir au mieux la liste des pairs constituant la vue partielle du réseau

Ces mécanismes sont distingués selon deux caractéristiques :

- **pro-actif** ou **réactif** selon que la diffusion est faite de manière autonome (pro-active) ou en réaction à un stimulus (réactive).
- **épidémique ou non épidémique** selon qu'il y ait ou non duplication des informations échangées dans le réseau. Le « flooding » est un exemple de diffusion épidémique alors qu'à l'opposé l'équilibrage de charge dans un réseau n'a pas pour objectif la duplication des tâches.

Ces deux critères nous permettent d'élargir le champ d'application des algorithmes de diffusion. Contrairement à Koubarakis *et al.*, nous ne pensons pas que le SDI se résume à un mécanisme de recherche basé sur l'utilisation de profils et d'un système événementiel [Koubarakis 03, Koubarakis 04]. D'après nos critères, il s'agit d'un mécanisme épidémique et réactif que nous inclurons dans la catégorie des algorithmes à base de publication/abonnement (« publish/subscribe »).

3.3.2.1 Diffusion à tous les pairs avec une forte probabilité (diffusion fiable)

Newscast [Jelasity 02, Voulgaris 03] est une architecture de diffusion fiable de l'information. Les auteurs définissent cette architecture comme étant composée de deux éléments principaux : Un agent et un correspondant dont une instance est située sur chaque machine du réseau. L'agent échange des informations avec son correspondant qui, à son tour, les échange avec ses semblables dans le réseau. Cet ensemble de correspondants est désigné comme la « news agency ». Pour qu'un nouvel arrivant la rejoigne, il lui suffit de connaître l'adresse d'un des membres et de commencer à dialoguer avec lui.

Chaque correspondant maintient un cache d'informations pouvant contenir c éléments. A chaque élément est associée l'adresse et l'identifiant unique du pair en ayant été l'émetteur. Tous les t_r unités de temps (la fréquence est identique mais ils ne sont pas synchrones), les correspondants exécutent l'algorithme suivant :

1. Demander à l'agent un nouvel item et l'ajouter dans le cache
2. Sélectionner aléatoirement l'adresse d'un des pairs parmi ceux listés dans le cache
3. Envoyer à ce pair la totalité du contenu du cache et fusionner le contenu de son cache reçu en retour

4. Envoyer le nouveau contenu du cache à l'agent.
5. Faire le tri dans le cache en enlevant les entrées les plus vieilles. L'objectif est de repasser d'une taille $2c$ à une taille de c

Les informations en cache servent donc à la fois de données échangées mais également de liste pour le voisinage. L'autre particularité de cette architecture est qu'elle autorise l'ajout et la suppression de pairs à coûts nuls. De ce fait, les pannes de certains éléments sont également sans incidence sur l'efficacité de la diffusion de l'information. Les expérimentations menées par les auteurs ont montré la fiabilité et la robustesse de cet algorithme. Ils ont également mis en évidence une topologie de réseau petit monde émergente.

Les algorithmes de bavardage (« Gossiping ») sont une méthode simple et efficace pour propager des messages dans un réseau [Kermarrec 00]. Le service de publication PlanetP [Cuenca-Acuna 02, Cuenca-Acuna 03] utilise ce genre d'algorithme pour diffuser une structure de données. Cette structure est un filtre de Bloom représentant un ensemble de fichiers partagés par un nœud donné. Les filtres sont les messages diffusés dans le réseau. Un pair x apprenant qu'un filtre a changé (le message a été mis à jour) va initier une procedure de gossiping. Toutes les T_g secondes, x va informer un de ses voisins y du changement (il déclenche une rumeur). Si y n'était pas déjà au courant, il va enregistrer cette informations et à son tour diffuser la rumeur. Le pair x stoppe sa diffusion après avoir contacté n pairs déjà au courant du changement. Il se peut qu'un pair ne soit pas mis au courant d'une rumeur avant que celle-ci ne s'éteigne. Pour contrer ce problème, les auteurs proposent également un mécanisme d'anti-entropie. Périodiquement, chaque pair x éxecute une opération d'anti-entropie à la place d'une opération de diffusion de rumeur : x demande à y de lui envoyer m des rumeurs les plus récentes qu'il a reçu. Si, parmi celles-ci, x trouve des messages qu'il n'a pas reçu, il demande à y de les lui envoyer.

Une approche similaire peut être trouvée dans les travaux de Iaminitchi *et al.* [Iaminitchi 02]. Ils utilisent un algorithme de ce type pour maintenir une information relative à une structure de groupe et faire circuler des informations entre les membres de ce groupe.

3.3.2.2 Diffusion selective à un groupe

Dans le cas ou l'objectif n'est pas de diffuser les informations à tous les membres du réseau, il est possible d'effectuer des diffusions sélectives. Le critère utilisé pour la sélection peut concerner le contenu même des informations à diffuser (« Content Based »), leur sujet (« Topic Based ») ou la conséquence de leur publications sous la forme d'un ensemble de règles (« Rules Based »).

Cette diffusion est effectuée en direction d'un groupe d'utilisateurs clairement identifié. Selon la façon dont est constitué ce groupe, on peut distinguer deux classes d'outils.

3.3.2.3 Systèmes événementiels

Un système événementiel se base sur l'utilisation de 3 éléments : un émetteur, des écouteurs et un système de diffusion d'événements. Ce système de diffusion est mis en œuvre par l'utilisation d'un ou plusieurs diffuseurs d'événements (« event brokers »). Ces deux possibilités sont illustrées en figure 3.11.

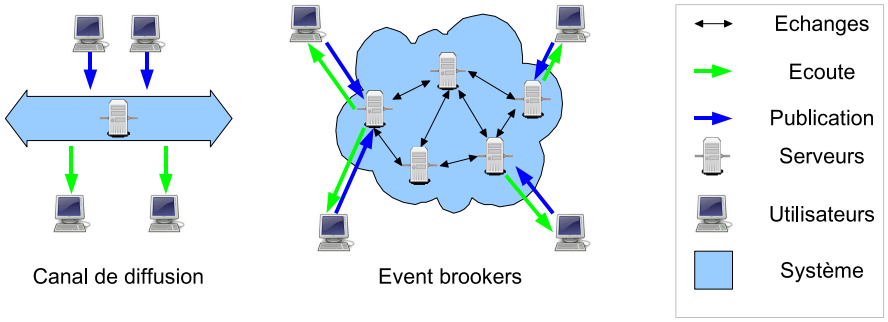


FIG. 3.11 – Comparatif des méthodes de mise en place d'une infrastructure de diffusion d'événement

Les systèmes événementiels peuvent être gérés de deux façons principales : soit par l'utilisation d'un canal de diffusion, soit par l'utilisation d'un mécanisme de publication/abonnement. La différentiation est faite au niveau de l'activité de publication d'un événement. Dans le premier cas, cette publication sera effectuée sur un canal dont l'émetteur ne connaît pas les clients qui écoutent. Dans le second, l'événement est envoyé à ceux qui sont abonnés. Pour implémenter le système événementiel, on peut soit utiliser un Bus de diffusion (auquel cas ce sont les clients qui filtrent, comme dans CORBA ou Dbus) ou mettre en œuvre un mécanisme de pub/sub. Ces systèmes sont basés sur un mécanisme de abonnement/publication (Publish-subscribe) ce qui signifie qu'un événement est publié puis récupéré par les consommateur y ayant souscrit. En partant du principe qui veut qu'un consommateur ne va pas être intéressé par tous les événements du réseau, la souscription permet de faire un tri sélectif. L'avantage du premier est la visibilité mais celui du second est la montée à l'échelle.

La réception d'un événement par une des machine peut donner lieu à la publication d'autres événements. Une gestion habituelle des événements est la règle ECA (« Event - Condition - Action ») qui se définit ainsi "Si un événement intervient, vérifier la condition et si elle est vérifiée exécuter l'action". Ceci permet d'associer à l'apparition d'un événement une action à effectuer [de Ipiña 01]. D'autres modèles tels que le ETR (Event - Trigger - Rules) [Leeb 04] ont été récemment proposés afin d'étendre ECA. ETR différencie les événements (Event) qui sont produits par les producteurs et envoyés aux déclencheurs, les déclencheurs (Trigger) qui sur la base de conditions appliquées à un ou plusieurs événements déclenchent l'exécution de règles et les règles (Rules) qui enfin déterminent la réaction à avoir vis à vis du ou des événements reçus. Cette architecture fait une meilleure séparation entre les événements et les réactions engendrées.

Les systèmes de publication/abonnement se basent sur l'utilisation deux 2 types de messages :

- Les abonnements qui permettent à un client se souscrire à la publications d'événements auprès d'un publieur ;
- Les notifications correspondant à la publication d'un événement ;

Le service de notification **Siena** [Carzaniga 00, Carzaniga 01] ajoute à ces deux éléments un événement d'avertissement permettant à un des composants d'indiquer quels sont les événements qu'il est susceptible de publier. Le système **P2P-DIET** [Koubarakis 03, Koubarakis 04] reprend cette même idée. Celui-ci se base sur l'utilisation d'une architecture hybride composée de clients et de super-pairs qui jouent le rôle de *brookers*. Les clients, dont chaque utilisateur possède une copie sur sa machine, se voient assigné un identifiant unique lors de leur enregistrement auprès

d'un des super-pairs. Cet identifiant permet de connaître l'origine de chaque notification faite dans le réseau. Ces notifications sont un des 3 messages qu'un client peut envoyer au super pair. Elles indiquent une ressource partagée. Les deux autres messages contiennent les profils et un résumé indiquant ce que le client est susceptible de publier. L'ensemble de ces messages circule entre les super-pairs. La principale faiblesse des systèmes à base d'abonnement/publication reste la gestion des abonnements.

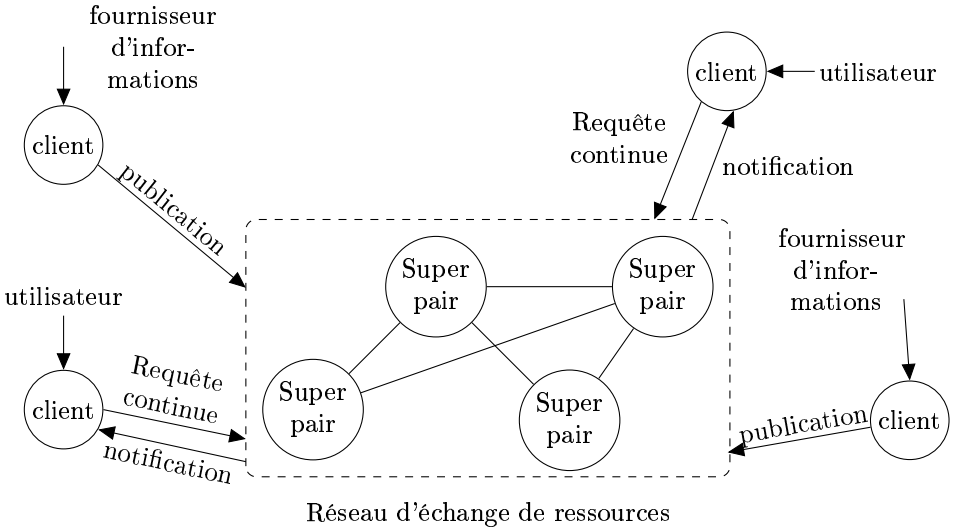


FIG. 3.12 – Architecture pour le partage de ressources proposé par Koubarakis *et al.* (d'après [Koubarakis 04])

3.3.3 Diffusion passive d'informations (non dirigée)

Aussi bien dans le cas de la recherche que de la diffusion, nous avons pu constater que l'utilisateur se doit de soumettre une requête soit explicite (« pull ») soit implicite (« push/pull »). L'idée générale de cette diffusion passive est que l'information circule dans le réseau sans qu'il n'y ait eu d'intervention explicite du client et/ou du serveur.

Les systèmes présentés jusqu'à présent ont tous en commun le fait d'être des modes de communication dirigés. Par un moyen ou un autre, l'émetteur d'un message maintient une liste de destinataires qui lui permet de savoir à qui envoyer un message ou un événement. Cette liste de destinataire peut être difficile à maintenir (par exemple dans le cas de réseaux de grandes tailles ou très dynamiques). Elle peut également être absente si l'émission d'un message ne résulte pas d'une activité directe de l'utilisateur (messages informant les utilisateurs de l'activité d'un autre utilisateur).

La diffusion passive s'illustre par l'utilisation de système à base uniquement de « pull ». Un des éléments du réseau émet un message d'informations sans se soucier de la destination.

3.3.3.1 Routage basé sur le contenu

Alors que le routage axé sur les adresses du ou des destinataires dépend du système employé pour assurer la concordance de ces adresses dans le réseau, le routage basé sur le contenu qui permet aux émetteurs d'informations de s'affranchir de toute informations concernant les destinataires des messages envoyés. Ce routage basé sur le contenu présente l'avantage de pouvoir s'adapter selon le contexte. Les possibilités de communications des composants sont limitées par l'API qui est utilisée.

Il en demeure néanmoins que les clients doivent pouvoir indiquer quels sont les messages qui les intéressent. Le routage axé sur le contenu (« Content based routing »), également appelé diffusion dirigée (« Directed Diffusion ») met en œuvre deux mécanismes : la diffusion des centres d'intérêt et la propagation de données [Intanagonwiwat 00]. L'intérêt d'un client pour un message donné est exprimé selon un ensemble de règle portant sur le contenu du dit message. Cet ensemble de règles est diffusé périodiquement afin de maintenir l'information à jour dans le réseau. Chaque nœud recevant un message exprimant des centres d'intérêt le transfert à ses voisins (figure 3.13(a)) et met à jour un gradient vers la source (figure 3.13(b)). Ces gradients sont par la suite utilisés pour définir un chemin à suivre lors de la propagation de données (figure 3.13(c))

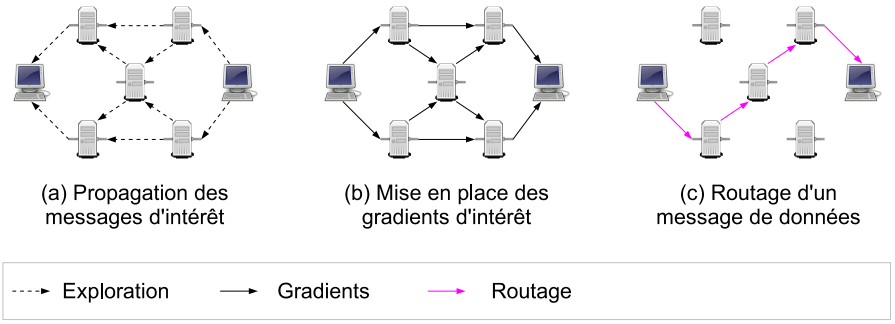


FIG. 3.13 – Mise en place d'une diffusion dirigée. Exemple inspiré de celui de [Dressler 06]

Le système **Elvin3** [Arnold 99] propose un mécanisme de routage par le contenu utilisant un processus d'abonnement/publication. Les abonnements se font en spécifiant un certain nombre de règles concernant les attributs des messages. Lorsqu'un de ces messages est reçu par Elvin de la part d'un des producteurs, il est comparé à l'ensemble des abonnements en cours et transféré vers ceux dont les règles sont satisfaites. Un mécanisme de « Quenching » permet également d'avertir les producteurs des abonnements qui sont fait. De façon périodique, où à chaque changement le justifiant, une copie intégrale de la liste des abonnements est envoyée à chaque client du système. Ainsi, un producteur sur le point de publier un message peut savoir s'il y a au moins un autre client qui va en être le destinataire. Si ce n'est pas le cas, l'envoi du message peut être annulé et la bande passante économisée.

En succédant à Elvin3, **Elvin4** [Segall 00] propose certaines améliorations au niveau de la sécurité et un mécanisme de « Quenching » repensé. Celui de Elvin3 présentait comme principal inconvénient une taille de message élevée due à l'envoie de la liste complète des abonnements. Le nouveau mécanisme autorise les client à indiquer des préférences sur le contenu de cette liste sous la forme de filtres gérés de façon similaire aux abonnements. Un client susceptible

de poster des messages et désirant utiliser le « quenching » va ainsi indiquer au serveur quelles sont les informations dont il a besoin concernant les abonnements. Ce mécanisme de filtrage peut être comparé à l'utilisation de message d'avertissement introduits par Siena, car envoyer un avertissement ou une liste de filtre permet dans les deux cas au client d'informer le serveur sur les messages à venir.

3.3.3.2 Déplacement autonome des informations

Automonous Gossiping [Datta 04] est un algorithme de diffusion sélective de l'information qui se base sur le principe d'une diffusion automatique de messages sur la base de profils. Sa principale caractéristique et de considérer les nœuds comme étant plus ou moins vulnérables au lieu de les traiter de façon homogène. Il est présenté par ses auteurs, Datta *et al.*, comme un concept pouvant être implémenté de différentes manières. La solution présentée dans [Datta 04] se base sur des paradigmes écologiques et économiques et est appliquée aux réseaux MANETS.

Les nœuds mobiles sont des habitats dont les préférences sont indiquées par un profil. Les items sont mis en compétition pour l'accès aux ressources qu'ils proposent. Afin de réduire les risques de « spams », un mécanisme permet de récompenser les items intéressants tout en pénalisant les autres.

Chacun des pairs m_k a une mémoire de taille M et un profil individuel m_k^p. Le profil est composé d'une liste de catégories C_j pouvant correspondre à des mots clé ou des contenus plus élaborés. Ces catégories sont pondérées selon le degré d'intérêt (affinité) du pair. Pour une catégorie C_i du profil m_k^p, cette affinité est notée MW_k^i. Les pairs sont initialisés avec un nombre aléatoire ($\leq M$) d'éléments de donnée (data item) d_j. Chaque d_j a son propre profil d_j^p, un réel indiquant son utilité d_j^u et un dernier attribut d_j^l définissant à quelle zone il est supposé appartenir. Au final, $d_j = \{d_j^p, d_j^u, d_j^l\}$. Un profil d_j^p a une affinité DW_j^i avec le sujet C_i. De plus, un profil spécial « any » permet de modéliser un pair interessé par tous les sujets ou une information correspondant à tous les C_j.

L'utilité d'un élément est actualisée à la hausse ($d_j^u(t) = d_j^u(t-1)+1$) si celui ci se trouve sur un pair ayant un profil proche ($similarity(d_j^p, m_i^p) \geq threshold$) ou si un autre pair ayant un profil proche est à portée. Les éléments les plus utiles sont également susceptibles d'être dupliqués. Les inutiles sont purgés du système. Les communications sont supposées symétriques et limitées à un rayon r_{comm}. Chaque pair diffuse son profil ainsi que sa destination. Cette destination change à chaque fois qu'elle est atteinte et sert à simuler le comportement d'un utilisateur qui se déplace.

Messor est un outil d'équilibrage de charge dans les réseaux P2P basé sur **AntHill** [Babaoglu 01, Babaoglu 02]. Ce dernier se place comme une architecture générique pour le développement d'application P2P. Cette architecture se base sur une analogie avec un ensemble de colonies de fourmis. Des fourmis artificielles sont chargées de l'échange de messages entre les machines du réseau. Machines qui correspondent alors à autant de nids. Ces nids sont composés de trois éléments (voir figure 3.14) :

1. Un « Ant scheduler » qui permet de coordonner l'action des fourmis mais également de servir de bac à sable. Ce « scheduler » assure la sécurité du système en limitant le champ d'actions des fourmis ;
2. Un « Communication Layer » chargé de la découverte de nouveaux nœuds, de la gestion des connexions et du déplacement des fourmis entre les nids ;

3. Un ou plusieurs « resource managers » permettant de gérer les ressources mises à la disposition des autres utilisateurs du réseau.

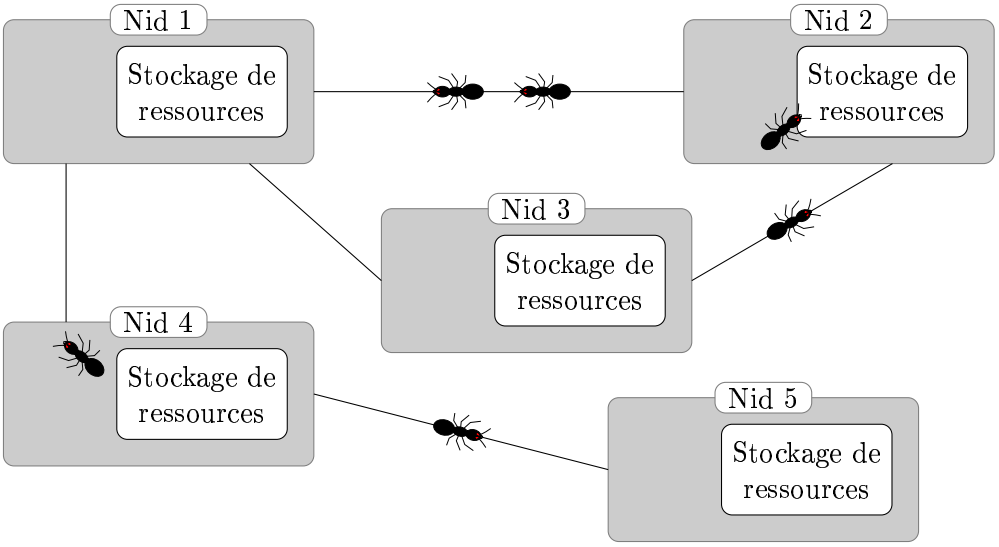


FIG. 3.14 – Fourmis assurant la circulation de ressources entre différentes zones de stockage dans Anthill [Babaoglu 01, Babaoglu 02]

Les fourmis sont chargées d'accomplir des tâches particulières. Celles-ci sont crées en réponse à une requête de la part de l'utilisateur. Une fourmi aura pour activité principale de parcourir le réseau de nid en nid jusqu'à ce qu'elle puisse satisfaire la requête où qu'elle décède. Sa durée de vie maximum est fixée par une TTL décrémentée à chaque nid visité. Durant leurs parcours, les fourmis emportent avec elles une variable d'état ainsi que leur code exécutable.

Pour le fonctionnement pratique de Anthill, Babaoglu *et al.* présentent l'exemple d'une application de partage de fichiers Gnutant. Dans ce cadre, la particularité de Anthill se situe au niveau de sa façon de gérer les index des fichiers partagés. En effet, chaque pair contient une table de routage associant mots clef et pairs. 3 algorithmes différents sont utilisés pour les fourmis, donnant lieu à l'utilisation de 3 types de fourmi :

- Les fourmis « InsertAnt » sont crées lors de la publication par un pair d'un nouveau document partagé. Une fourmi de ce type est générée pour chaque mot clé associé au document ;
- Des « SearchAnt » sont utilisées afin de traiter les requêtes de l'utilisateur. Chaque mot clé de la requête donne lieu à la création d'une de ces fourmis ;
- Enfin, dès qu'une « SearchAnt » a pu satisfaire sa requête celle ci se change en « ReplyAnt ». Elle retourne alors directement vers la machine depuis laquelle elle est partie afin d'indiquer l'URL où le contenu recherché à été trouvé.

Bien qu'associées à un mot clé particulier, les « InsertAnt » et « SearchAnt » emportent toutes deux l'ensemble de la requête. Les fourmis suivent cependant des chemins différents dans le réseau. Ces choix dépendent du mot clé qu'elle ont : lorsqu'elle est sur une machine, la fourmi choisi sa destination suivante en fonction de la similarité entre son mot clé et la connexion menant au nid concerné.

Dans le cadre du projet « Messor » [Montresor 01, Montresor 02a, Montresor 02b], les fourmis peuvent être dans deux états différents :

- « SearchMax » : Une fourmi dans cet état va être à la recherche d'un pair en sous-charge. Une fois un tel pair trouvé, son adresse est mémorisée et la fourmi passe en « SearchMin »

– « SearchMin » : Dans cet état, le but pour la fourmi est de trouver un pair sous-chargé. Une fois qu'un tel pair est trouvé, la fourmi demande au pair surchargé de transférer son surplus de tâche vers ce pair ci. Une fois ceci effectué, la fourmi se remet en « SearchMax »

Le choix de la destination suivante est fait aléatoirement avec la possibilité pour la fourmi de faire de l'exploration ou non.

3.4 Gestion de la topologie du réseau

Les algorithmes que nous venons de voir partagent un certain nombre de faiblesses. En particulier nous pouvons citer pour les algorithmes de type « push » le manque de prise en compte de la sémantique du contenu et des relations entre les nœuds qui en découle. La limitation commune à ces deux systèmes est qu'aucune des solutions présentées actuellement ne permet de prendre en compte l'utilisateur et ses centres d'intérêt. L'idée de pouvoir bénéficier du dynamisme de l'architecture non structurée tout en ayant l'efficacité du structuré à donc été explorée. En partant d'un réseau non structuré, une nouvelle couche de réseau logique est appliquée. Cette couche supplémentaire peut éventuellement être utilisée afin de reprogrammer la topologie du réseau physique.

Afin d'améliorer les performances des algorithmes précités, deux stratégies peuvent être considérées. Les systèmes de recherche, qu'ils s'appliquent aux réseaux P2P structurés ou non, souffrent de plusieurs limitations d'ordre général ou plus particulièrement liées à notre problématique :

– Bien que les solutions de recherche dans un réseau fortement structuré soit efficaces, deux reproches peuvent leur être fait. Le premier est de ne pouvoir prendre en compte que des requêtes basées sur une recherche d'identifiant. Ces structures ne sont pas adaptées pour effectuer des recherches par mot clef.
– Un second reproche vient de leur nature même : la structuration du réseau est un mécanisme qui s'adapte mal aux conditions dynamiques inhérentes aux réseaux P2P telles qu'observées dans le cadre de l'échange de fichiers [Ratnasamy 02b]. Le maintient de la structure étant une opération tout aussi coûteuse qu'essentielle pour l'environnement. De plus, au delà de leurs fonctionnalités, tous les pairs sont considérés comme strictement équivalents. C'est à dire qu'une machine de faible puissance et/ou connectée avec une liaison réseau de bas-débit aura la même importance (et donc la même charge !) que tout autre machine du réseau.
– La prise en compte d'un réseau faiblement structuré permet d'obtenir des algorithmes moins couteux mais également moins efficaces.

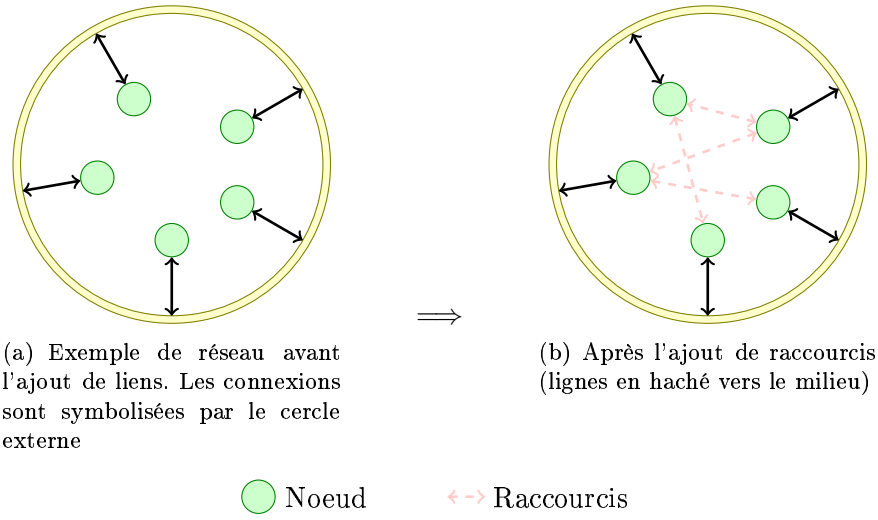
Les sciences sociales ont permit de démontrer que ces communications interpersonnelles sont un moyen de diffusion de l'information important et essentiel. Ce sont elles qui structurent les relations sociales observées entres utilisateurs. Relations qui donnent d'ailleurs lieu à l'apparition de structures de réseaux de type petit monde. Ainsi, la théorie des réseaux petits mondes (« Small world ») quitte le domaine viral pour s'appliquer à celui des réseaux sociaux dans lesquels les nœuds sont des individus et les arcs des relations entre eux. La recherche active dans le domaine de la localisation d'informations a fait apparaître différentes relations. Parmis celles-ci on peut remarquer la confiance, la proximité géographique, la réputation et la notion d'intérêt commun. Iaminitchi *et al.* ont mené des études concernant la communauté scientifique. Une première partie de leur travaux a permis de démontrer que dans un réseau composé de chercheurs où les liens entre les nœuds traduisent des centres d'intérêt commun, une topologie de « small world » apparaissait [Iaminitchi 02]. Pour assurer la diffusion des messages, les auteurs proposent

l'utilisation d'un algorithme de bavardage. Considérant un petit monde de C groupements chacun composés de G nœuds. Chaque nœud d'un groupe connait une partie des nœuds composant le groupe. Quand un nœud veut emettre un message il l'envoit à une partie du groupe. Les nœuds recevant un message le traitent (ajoutent des informations, ...) puis le réémettent sur le même principe. Plus récemment cette même équipe a introduit le terme de « graphe de partage de données » (data sharing graph) [Iaminitchi 04] pour designer les graphes dont les nœuds sont des utilisateurs et dans lesquels une connexion entre deux nœuds indique des centres d'intérêt similaires. Une récente étude [Palau 04] a mis en évidence l'apport de l'étude des topologies petit monde sur l'efficacité des systèmes de recommandation.

Ceci dit, ce n'est pas cette informations qui va directement nous permettre d'optimiser les accès en P2P. En effet, même si l'on sait que certaines personnes seront plus à même de répondre que d'autre, la question principale demeure : « qui interroger ? ». Ce sont donc des solutions hybrides qui sont mises en place et qui permettent soit de créer des raccourcis dans les réseaux P2P soit de les structurer dynamiquement.

3.4.1 Ajout de raccourcis

Ajouter une sur-couche sémantique permet d'améliorer les performances des mécanismes de recherche. L'idée est d'ajouter une structure supplémentaire de liens entre les nœuds. Cette structure identifie les pairs à interroger en priorité avant de se rabattre, le cas échéant, au mécanisme d'interrogation proposé par le réseau. La sémantique associée dépend du mécanisme de raccourcis mis en place, nous en verrons ici deux : l'identification de petit mondes et la mise en correspondance de pairs ayant des intérêts similaires.



(a) Exemple de réseau avant l'ajout de liens. Les connexions sont symbolisées par le cercle externe

(b) Après l'ajout de raccourcis (lignes en haché vers le milieu)

FIG. 3.15 – Ajout d'une structure de liens sémantiques supplémentaires

Ces *overlay* prennent généralement la forme d'une liste sémantique contenant les adresses des pairs à contacter en priorité. Nous ferons une distinction entre les « overlays » destinés à s'appliquer aux réseaux structurés et ceux venant en renfort des réseaux non structurés.

3.4.1.1 Dans un P2P structuré

Les réseaux structurés sont, comme nous l'avons vu, régi par une structure précise allant de paire avec une stratégie de recherche par identifiant.

Hui *et al.* [Hui 04] proposent **SWOP** (« Small-World Overlay Protocol »). Il s'agit d'une structure sémantique venant compléter celle déjà mise en place dans un réseau Chord. Deux types de nœuds sont définis : « tête » (« head node ») et « internes » (« inner node ») ainsi que deux types de liens « long » (« long links ») et « groupe » (« cluster links »). Les liens longs sont ceux existant entre deux groupements alors que les liens courts sont ceux internes à ces groupes. En ce qui concerne les nœuds, chaque groupe est composé d'un nœud tête ayant au maximum k liens longs avec d'autres nœuds ainsi que des nœuds internes. Les nœuds internes d'un groupe donné sont connectés à la racine ainsi qu'à un certain nombre d'autres nœuds internes du groupe. Ces groupements sont re-définis au moment d'arrivées ou de départs de nœuds. Selon sa position dans le cercle, un nœud nouvel arrivant va tenter de joindre à un des groupement existant avant où après lui. Si il n'existe pas de groupements où que ceux existants ont atteint une taille limite, le nœud crée son propre groupe et devient nœud tête. Ces groupements permettent d'accélérer le processus de recherche. La figure 3.16 présente l'exemple d'un tracé obtenu pour la recherche d'un élément d'identifiant 16 depuis le nœud 1. On remarque que $successeur(16) = 17$, c'est donc le nœud 17 qui contient la valeur de hash de l'objet convoité. En temps normal, cette recherche aurait suivi le chemin suivant $1 \rightarrow 4 \rightarrow 6 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 14 \rightarrow 17$. Avec les raccourcis, ce chemin est réduit à $1 \rightarrow 26 \rightarrow 14 \rightarrow 17$: le nœud 1 commence par interroger le nœud tête de son groupe, le 26. Ce dernier n'aillant pas l'objet il transfere la demande au nœud tête 14 lequel dirige finalement le message vers le nœud 17.

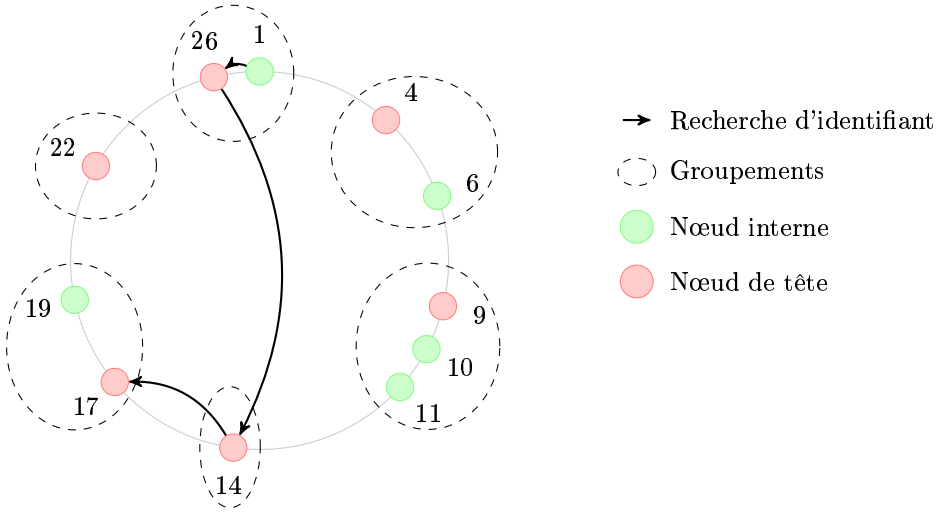


FIG. 3.16 – Exemple de raccourcis créés par SWOP (d'après [Hui 04])

SWAN [Bonsma 02] propose une approche similaire qui se base sur la création de nouveaux liens dans un réseau CAN. Ces liens longs sont ajoutés de façon à produire une topologie de réseau petit monde.

3.4.1.2 Dans un P2P non structuré

Le système proposé par Sripanidkulchai [Sripanidkulchai 03] crée des raccourcis dans le réseau de Gnutella en partant de l'idée suivante : si un nœud à répondu favorablement à une requête, il est probable qu'il ait d'autres documents intéressants. La relation d'intérêt est donc unidirectionnelle, à la différence d'autres systèmes créant des liens selon une relation de similarité entre les centres d'intérêt de deux utilisateurs.

Un pair arrivant dans le réseau va commencer par effectuer une requête sur le mode habituel de diffusion de Gnutella. Un ensemble de nœuds est alors choisis au hasard parmi ceux ayant fournit une réponse. Cet ensemble constitue une liste de raccourcis composés de pairs susceptibles de fournir des réponses positives pour d'autres requêtes. Par la suite, le pair interrogera les nœuds de sa liste de raccourcis avant d'utiliser le réseau Gnutella directement. Les pairs de la liste sont classés selon un rang particulier calculé en fonction de paramètres tels que la bande passante disponible, le temps de latence ou le taux de réponse satisfaisantes fournies. Ce rang sert à la fois à choisir quel raccourci utiliser en priorité mais également à décider lequel supprimer le cas échéant. La liste étant de taille fixe (10 entrées), ces suppressions sont nécessaire à l'ajout de nouvelles adresses.

Dans [Voulgaris 04], ces travaux ont été repris et étendus. L'étude de 3 stratégies différentes pour ordonner la liste des pairs interrogés en priorité :

1. La première stratégie est de classer les pairs en fonction de leur utilité (Last Recently Used - LRU). LRU prend à la fois en compte les pairs ayant répondu et ceux ayant contribué à répondre ce qui revient à la méthode employée dans [Sripanidkulchai 03].
2. La seconde stratégie « History » ne prend en compte que les réponses fournies par les pairs. Le rang d'un pair n'est augmenté que s'il a directement répondu.
3. En prenant également en compte le nombre de pairs ayant répondu, il est possible de définir la troisième stratégie : « Popularity »

Leur conclusions sont que le mécanisme basé sur la popularité permet d'obtenir d'aussi bon résultats que l'historique tout en étant plus léger. Ceci dit, cette méthode requiert d'avoir une information concernant la popularité du document recherché. Information qui peut ne pas être facilement disponible.

Contrairement aux systèmes qui se base sur les requêtes effectuées ainsi que leur réponses pour détecter les relations, Voulgaris et Steen proposent dans [Voulgaris 05] un algorithme pro-actif de mise en place d'*overlay*. Cet algorithme se base sur l'utilisation d'une architecture à deux niveaux. Le niveau le plus bas, baptisé « CYCLON » a pour tâche d'assurer la connectivité du réseau tout en envoyant des informations sur les pairs présents au niveau supérieur. Ce niveau supérieur, « VICINITY » fait le tri dans les liste des pairs envoyés afin de sélectionner les plus intéressants à garder dans la liste de raccourcis sémantiques. Ces niveaux communiquent entre eux dans le réseau par le biais d'algorithmes de Gossip et ont pour particularité de ne s'échanger qu'une partie de leur vue. Plus précisément, CYCLON maintient en permanence un *overlay* de connexions totalement aléatoires entre les pairs avec pour seul objectif de pouvoir proposer à la demande des pairs à VICINITY. Les deux objectifs visés lors de la construction de l'*overlay* sont :

- en supposant l'existance de la transitivité, explorer les voisins les plus sémantiquement proches des voisins présents dans la liste de raccourcis ;

– examiner tous les nœuds afin de pouvoir détecter les liens longs présents dans les structures de type small world. Ceci implique une selection stochastique des nœuds à placer dans la liste de raccourcis.

Les auteurs supposent l'existence d'une fonction de proximité (éventuellement transitive) s'appliquant entre le profil de deux pairs. Les profils eux mêmes correspondent à la liste des fichiers partagés. En utilisant comme fonction de similarité le nombre de fichiers communs partagés et un jeu de données issu du réseau Gnutella, montrer l'efficacité de leur système. D'un point de vue dynamique, seul des changements radicaux ont été testés. La construction progressive de la liste de fichiers n'a pas été envisagée.

3.4.2 Ajustement dynamique de la topologie

Au lieu de rajouter une structure supplémentaire (et donc devoir en assurer la maintenance), il peut être envisagé d'adapter dynamiquement les connexions. L'objectif est alors de calquer les connexions sur les échanges de données et/ou les affinités observés entre les pairs. Les avantages de cette solution sont doubles : le premier est qu'il n'y a pas de structure supplémentaire à maintenir et le second est que cela permet de prendre directement en compte les capacités des machines.

La répartition des identifiant dans les algorithmes à base de DHT est une illustration de ce principe. Dès qu'un pair quitte ou entre dans le réseau, les connections sont ajustées et les identifiant redistribués afin d'obtenir, par exemple, une topologie en arbre ou en cercle. Ce principe s'applique à des réseaux structurés dont le fonctionnement du mécanisme de propagation des messages est directement lié à la topologie. Dans le cas de réseaux non structurés, il est néanmoins possible de mettre en place des outils permettant de modifier la topologie selon un critère. Parmis ces critères peuvent se trouver des centres d'intérêt partagés, la diffusion de message aux contenus similaires et la proximité géographique. Nous présenterons ci-après quelques exemples d'utilisation.

Dans le modèle proposé par Schmitz [Schmitz 04], un pair P_k va initier une marche aléatoire de façon périodique afin de rechercher les pairs du réseau ayant des profils similaire aux siens. Le critère consideré est alors celui de centres d'intérêts communs. Ce processus de marche se compose des étapes suivantes :

1. P_k calcule un coefficient de satisfaction de son voisinage (équation 3.10). Si la valeur obtenue est inférieure à un certian seuil, le voisinage peut être amelioré.

$$\frac{1}{k_{P_k}} \sum_{P_j \in \{P_j | connait(P_k, P_j)\}} sim(P_k, P_j) < minSimilarity \tag{3.10}$$

2. Si on est dans ce cas, P_k compose un message M de type $WalkMessage$ contenant son expertise ainsi qu'une durée de vie maximum TTL.
3. Le message est tranféré de pair en pair, lors de chaque transfert la valeur de TTL est decrémentée. Chaque pair transférant le message y enregistre sa propre expertise. Le processus de transfert s'interomp quand $TTL = 0$.
4. Si $TTL = 0$, le message M est retourné à son expéditeur P_k

5. P_k récupère le message M et, de part son contenu, les expertises de l'ensemble des pairs en ayant assuré le cheminement. Il est alors susceptible d'ajouter dans sa table de routage un ou plusieurs pairs de cet ensemble. Si la table de routage est pleine, les connexions les moins similaires seront abandonnées au profit des nouvelles plus prometeuses.

L'algorithme **T-Man** [Jelasity 04, Jelasity 05] se veut plus généraliste en proposant une solution basée sur la définition d'une fonction de rang et une diffusion probabiliste des messages. La fonction de rang décrit la topologie visée sous la forme d'une fonction d'ordre partielle appliquée sur les profils des nœuds. Ces profils indiquent l'emplacement du nœud dans l'espace. Pour un ensemble de N nœuds ayant chacun c connections et une fonction d'ordre $R(x, \{y_1, \ldots, y_m\})$ classant les nœuds $\{y_1, \ldots, y_m\}$ par rapport à x, la topologie optimale du réseau est atteinte quand pour tout nœud x, les c premiers éléments de $R(x, \{Tous\ les\ autres\ nuds\})$ correspondent aux c nœuds auxquels x est connecté. Le fonctionnement de T-man se décompose en deux parties, une active et une passive : la première prend contact avec les autres nœuds et effectue des échanges de profils, la seconde attend l'arrivée de messages. Périodiquement, la partie active sélectionne un des pairs du voisinage et lui envoie le contenu de son voisinage. En réponse, la partie passive de ce nœud retourne également son voisinage. Après avoir appliqué la fonction de rang sur ces nouvelles informations, les deux nœuds ne gardent chacun que les c premières connections selon R. Ce mécanisme autocatalyseur permet de rapidement identifier et regrouper les pairs ayant un voisinage idéal similaire. Afin de l'utiliser il est cependant nécessaire d'être en mesure d'assigner un emplacement à chaque nœud, de définir une fonction d'évaluation relative à ces emplacements et de fixer le degré de façon globale dans le réseau.

3.5 Conclusion

Notre problème se compose de 3 éléments : du contenu à échanger, des utilisateurs et un réseau d'échange pour lesquels nous venons d'apporter quelques précisions. De ce chapitre introductif nous pouvons tirer des premières conclusions. En particulier concernant la modélisation des contenus. En effet, quelque soit l'encodage utilisé, il est possible de mettre en œuvre soit une mesure de similarité soit une mesure de distance. Sous certaines conditions, il est donc possible de s'affranchir de cet aspect en ne considérant que les éléments produits et la métrique servant à les comparer. Une des conditions est que l'algorithme étudié puisse se contenter de cette informations diminuée.

Pour que notre définition soit complète, et avant de s'intéresser aux différentes méthodes de résolutions pouvant être utilisées, il est nécessaire de définir la nature du problème. Dans la littérature, il est possible de distinguer deux approches différentes pour la diffusion d'informations dans un réseau. La première consiste à considérer les préférences de chaque utilisateurs comme étant un ensemble de contraintes et d'appliquer ainsi des techniques de SAT (satisfaction de contraintes). Cependant, cette modélisation requiert une connaissance globale du réseau ce qui n'est pas toujours applicable. La seconde consiste à considérer le réseau, ses utilisateurs et les données comme formant un système complexe adaptatif comme l'a suggéré [Montresor 02a]. Notre problème se présente, sous certains aspects, comme un système de capteurs. Des études de ces systèmes basées sur l'utilisation de systèmes à base de colonies de fourmis ou d'automates cellulaires ont prouvé qu'un modèle biologique pouvait s'appliquer à l'échange d'informations entre sondes [Wokoma 02b].

Ce chapitre présente un tour d'horizon des différents moyens de diffusion de l'information dans un réseau. Deux stratégies peuvent être observées selon que l'utilisateur est à l'origine de l'échange (« push ») ou que celle-ci est à l'initiative du serveur. Après avoir fait ce tour d'horizon, la question qui se pose alors est celle du choix de la méthode la mieux adaptée à notre problématique. Cette problématique, qui est celle de la diffusion d'informations sans notion de profil, nous situe dans le cas d'un utilisateur de passage dans un cybercafé qui demande des renseignements à un programme installé sur la machine. Notre objectif est que ce programme puisse lui fournir un maximum d'informations pertinentes.

Les méthodes à base de requêtes ne permettent pas d'atteindre notre objectif. Leur but étant généralement d'interroger un réseau d'informations puis de laisser l'utilisateur faire le tri. Ainsi notre internaute devrait spécifier clairement ce qu'il cherche et faire le tri dans les réponses. A la place, nous considérons un dialogue entre ce dernier et la machine qui, sur la base des informations qu'elle a accumulées, tente d'apporter une réponse. L'interrogation est donc locale. Ceci dit, de telles solutions pourraient être envisagées directement sous la forme d'une application utilisant PIAF qui, pour répondre à l'utilisateur, pourrait formaliser une requête et interroger des réseaux d'informations. Un peu dans le même ordre d'idée que les moteurs de recherche intelligents qui interrogent plusieurs moteurs avant de proposer une réponse synthétique à l'utilisateur. Parmi les méthodes de diffusion, le abonnement/publication nécessite la définition d'un profil contenant les centres d'intérêt de l'utilisateur. Celle-ci ne correspond donc pas à nos besoins.

Chapitre 4

Quelques modèles biologiques pour les systèmes complexes

Ce chapitre est consacré à la présentation de quelques modèles inspirés du vivants et pouvant nous aider à résoudre notre problématique. Les systèmes utilisant des colonies de fourmis artificielles et les systèmes immunitaires artificiels seront plus particulièrement détaillés.

4.1 Introduction

L'environnement dans lequel se pose la problématique de la circulation d'informations dans un réseau P2P peut être qualifié de système complexe. Selon [1]

> *" Un système complexe peut être défini comme un système composé de nombreux éléments différenciés interagissant entre eux de manière non triviale (interactions non-linéaires, boucles de rétroaction, etc.). "*

Les algorithmes inspirés du vivant (fourmis, systèmes immunitaires, ...) sont, comme nous le verrons par la suite, particulièrement adaptés à la résolution de ce genre de problèmes. Beaucoup de solutions pour la recherche d'informations dans les réseaux P2P non structurés sont d'ailleurs basées sur ces théories. Parmis les nombreuses méta-heuristiques de résolution existantes, nous présenterons ici les algorithmes de colonies de fourmis, l'optimisation par essaim particulaire ainsi que les systèmes épidémiques et immunitaires. L'observation du vivant a été la source d'inspiration pour la conception d'algorithmes de résolution appliqués aux systèmes complexes. En particulier, ce sont les phénomènes d'auto-organisation et d'émergence qui ont suscité le plus d'attention. Le lecteur est invité à consulter [Bonabeau 99] pour un panorama des approches concernant l'intelligence collective.

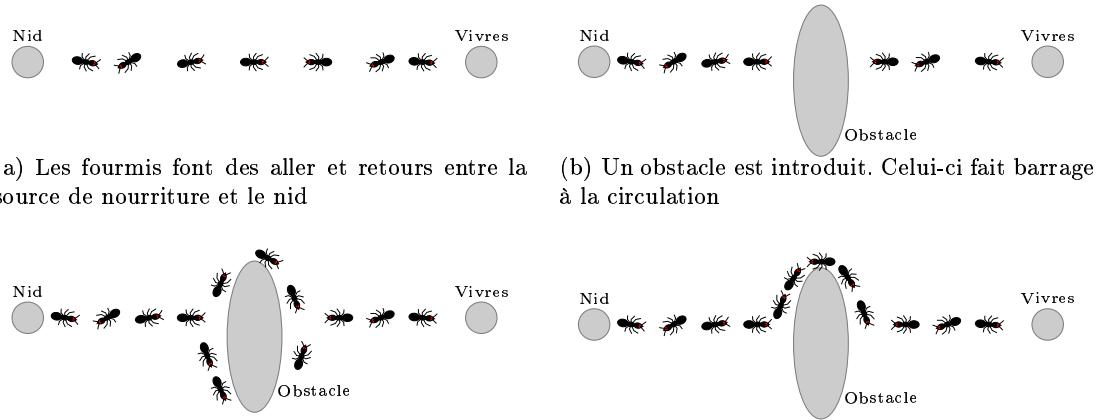
[1] `www.univ-lille3.fr/ureca/ureca/conferences-complexes/conf-complexe.htm`, dernier accès le 22 octobre 2006

Certains des modèles présentés ici ont directement trouvé une application dans les systèmes de diffusion d'information (cf chapitre 3. D'autres ne trouvent pas pour l'instant d'application mais sont néanmoins cités ici car ils ont inspiré notre système PIAF (Personnal Intelligent Agent Framework) ou pouvant être mis en perspective pour des développements futurs de l'algorithme.

4.2 Activité de fourragement des fourmis

4.2.1 Observations biologiques et modèle mathématique

Les méta-heuristiques à base de colonies de fourmis ont été introduites à partir des années 1990. L'idée d'utiliser des fourmis artificielles pour résoudre des problèmes d'optimisation prend racine dans l'observation et l'étude du comportement des fourmis réelles. Notamment, pour leurs activités de fourragement, ces insectes se sont montrés capables de trouver le plus court chemin reliant le nid à un point de nourriture (voir figure 4.1).



(a) Les fourmis font des aller et retours entre la source de nourriture et le nid

(b) Un obstacle est introduit. Celui-ci fait barrage à la circulation

(c) Durant une phase exploratoire, les fourmis essayent les différents chemins possibles. Ce faisant, elles déposent des phéromones qui vont par la suite progressivement s'évaporer

(d) Le chemin le plus court est finalement detecté : il s'agit de celui sur lequel les phéromones ont le moins le temps de s'évaporer avant le passage d'une autre fourmi.

FIG. 4.1 – Contournement d'un obstacle par une population de fourmis (extrait de [Dorigo 96a])

Cette faculté suscite d'autant plus l'intérêt quand on remarque que les fourmis ne peuvent agir que sur la base d'informations locales. C'est un phénomène de stigmergie : par l'action autonome de chacune d'entre elle sur l'environnement, un comportement global émergent apparaît.

La méthodologie ACO (« Ant Colony Optimisation ») a dans un premier temps été appliquée à la résolution du problème du voyageur de commerce via l'algorithme « Ant system » [Dorigo 96a, Dorigo 96b]. Ce problème consiste à trouver dans un graphe pondéré le plus court chemin permettant de passer par tous les nœuds. L'analogie est faite avec le cas d'un voyageur de commerce qui pour son travail aurait à visiter un ensemble de villes avec deux contraintes : passer une seule fois par chaque ville et minimiser les coûts des déplacements.

Soit l'ensemble des villes N. A chaque couple de ville (i, j) est associée une distance x_{ij}. Une valeur numérique représentant une quantité de phéromones artificielles est également associé au chemin ij reliant ces deux villes.

Un ensemble de fourmis m est initialement placé aléatoirement sur différents sommets du graphe. Le but d'une fourmi est alors de compléter un tour en passant de ville en ville. Leur comportement est régit par trois principes :

1. la fourmi choisi la prochaine ville où elle va se rendre en fonction de la distance à parcourir pour s'y rendre et de la quantité de phéromone présente sur la route ;
2. pour n'obtenir que des circuits valides (*i.e.* respectant l'unicité de passage), une fourmi k compose une liste tabou $T(k)$ des villes déjà visitées ;
3. une fois qu'elle a fini son tour, la fourmi laisse une trace de phéromones sur le chemin total parcouru.

Soit $\tau_{ij}(t)$ la quantitée de phéromones présente sur une connexion de i à j à l'instant t. Les fourmis vont au cours de leur déplacement faire évoluer cette valeur. Le déplacement d'une ville à une autre s'effectue entre les instants t et $t+1$. L'algorithme est itératif donc le nombre de fois où ces phéromones seront réajustée est égal au nombre de villes n. Ainsi

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}$$

Dans cette formulation, ρ est un facteur de persistance des phéromones. $(1-\rho)$ représente la proportion de phéromones qui se seront évaporées entre les instants t et $t+n$. La quantité de phéromones apportées, $\Delta\tau_{ij}$, correspond à la somme de l'apport fait par chaque fourmi ayant emprunté le chemin concerné (toujours entre les instants t et $t+n$).

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

Où $\Delta\tau_{ij}^{k}$ représente l'apport de la k-ème fourmi ayant emprunté le chemin ij. Cet apport est fonction de la qualité du chemin trouvé. Etant donné que l'objectif visé est la minimisation de la longeur d'un tour, cette qualité est inversement proportionnelle à la longeur L_k du tour effectué par cette fourmi entre t et $t+n$. Dans sa formulation initiale, cette quantité est définie ainsi :

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \text{si la fourmi } k \text{ est passée par l'arc } (i,j) \\ 0 & \text{sinon} \end{cases}$$

Deux variantes baptisées « *ant-density* » et « *ant-quantity* » peuvent être considérées :

$$\Delta\tau_{ij}^{k} = \begin{cases} Q & \text{si la fourmi } k \text{ est} \\ & \text{passée par l'arc } (i,j) \\ 0 & \text{sinon} \end{cases} \quad \text{et} \quad \Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{d_{ij}} & \text{si la fourmi } k \text{ est} \\ & \text{passée par l'arc } (i,j) \\ 0 & \text{sinon} \end{cases}$$

Celles-ci permettent respectivement d'attribuer un score équivalent indépendant de la qualité du chemin parcouru (« *ant-density* ») ou de privilégier les chemins les plus courts en prenant en compte la distance séparant les villes (« *ant-quantity* »). Dans les trois formulations, Q est un paramètre constant utilisé pour la régulation de la quantité déposée.

Les déplacements des fourmis sont régis par des probabilités de transition. La probabilité pour que la fourmi k, à un instant t, se rende de la ville i à la ville j est notée $p_{ij}^{k}(t)$.

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N \setminus T(k)} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} & \text{si } j \in N \setminus T(k) \\ \\ 0 & \text{sinon} \end{cases} \tag{4.1}$$

Cette formulation pose un compromis entre les phéromones $\tau_{ij}(t)$ déposées sur un chemin et sa visibilité η_{ij}. La visibilité du chemin est définie en fonction de sa distance, les chemins les plus courts étant les plus visibles.

$$\eta_{ij} = \frac{1}{d_{ij}}$$

Les cœfficients α et β permettent de pondérer l'influence de ces deux éléments dans le calcul de la probabilité de choix.

Sur un aspect plus général, la méthodologie ACO se présente de la façon suivante : le problème à résoudre est modélisé par un graphe. Chaque solution au problème est alors définie comme étant un parcours dans ce graphe. Une itération de l'algorithme consiste à faire se déplacer l'ensemble des fourmis puis à mettre à jour les quantités de phéromones. Les meilleures solutions sont alors favorisées avant de passer à l'itération suivante.

4.2.2 Adaptation des fourmis artificielles au problème de routage de paquets dans les réseaux

Si l'on s'intéresse à l'application de ACO au problème du routage de message dans un réseau, on remarque rapidement que la méthodologie n'est pas directement applicable. Contrairement au voyageur de commerce, un message ne doit pas nécessairement passer par *tous* les nœuds alors qu'il se rend d'un point à un autre du réseau.

Deux versions adaptées d'ACO baptisées « AntNet » et « AntHocNet » ont ainsi été proposées afin de respectivement aborder le problème du routage dans le cas de réseaux *ad-hoc* fixes et mobiles [Caro 98]. La principale modification apportée à ACO concerne la mise à jour des phéromones. Celle-ci n'est plus effectuée globalement après le déplacement de toutes les fourmis mais progressivement selon l'activité de chacune d'entre elles. Le problème consiste à faire circuler un message d'une machine à une autre en utilisant un ensemble d'autres machines en tant que relais.

Le réseau est modélisé par un graphe dont chaque nœud est une machine et les arcs les connexions réseau. Les fourmis y assurent la circulation des paquets et mémorisent dans une liste $\mathcal{P}$ l'ensemble des nœuds visités. Chaque nœud i contient une table de routage τ^i. Ces tables permettent de décider de la direction vers laquelle envoyer un paquet reçu par le pair : les valeurs $\tau_{nd}^i \in \mathbb{R}$ qui y sont indiquées correspondent au taux de confiance accordé au nœud n dans l'optique de rejoindre d en partant du nœud i. Selon une probabilité P_{nd}, les fourmis porteuses de paquets choisissent leur destination suivante :

$$P_{nd} = \frac{[\tau_{nd}^i]^\beta}{\sum_{j \in N_d^i} [\tau_{jd}^i]^\beta}$$

Dans cette équation, N_d^i est l'ensemble des voisins de i connus comme pouvant mener vers d et $\beta \geq 1$ est un paramètre controlant le taux d'exploration des fourmis. Si N_d^i est vide, c'est à dire que le pair ne dispose d'aucune information concernant le meilleur chemin à suivre, la fourmi est diffusée : une copie en est envoyée à chaque machine connectée.

Lorsqu'un nœud s initie une session avec une destination d pour lequel il ne connait pas de route, il envoie à chacun de ses voisins une fourmi de type « forward ». Une fois arrivée à destination, la fourmi « forward » se change en « backward » et retourne vers s en suivant le même chemin qu'à l'aller. Si un des nœuds de P n'est plus présent dans le réseau au moment du retour, la fourmi est supprimée. Alors qu'elle retourne vers s, la fourmi calcule de façon itérative le temps $\hat{T}_{\mathcal{P}}$ qu'elle estime devoir mettre pour effectuer le trajet $\mathcal{P}$. Pour un tronçon entre un pair i et son suivant $i+1$, l'estimation dépend du nombre de paquets Q_{mac}^i en attente dans la carte réseau de i ainsi que le delai de traitement de ces paquets $\hat{T}_{mac}^i$. Cette formulation a l'avantage de prendre en compte l'éventuelle congestion de la carte réseau :

$$\hat{T}_{\mathcal{P}} = \sum_{i=1}^{k-1} \hat{T}_{i+1}^i = \sum_{i=1}^{k-1} (Q_{mac}^i + 1)\hat{T}_{mac}^i$$

Cette information est utilisée pour mettre à jour les phéromones.

$$\tau_{nd}^i = \gamma \tau_{nd}^i + (1-\gamma)\left(\frac{\hat{T}_d^i + h\hat{T}_{hop}}{2}\right)^{-1}$$

Dans cette équation, h représente le nombre de nœuds à traverser (« hop ») et $\hat{T}_{hop}$ est un paramètre indiquant le temps nécessaire à une transition entre deux nœuds dans des conditions normales. Le paramètre $\gamma \in [0,1]$ fixe le taux d'évaporation des phéromones.

AntNet et AnHocNet sont des algorithmes de routage pour réseau à commutation de paquets. L'algorithme ABC (Ant-based Control) [Schoonderwoerd 96] permet de se placer à un autre niveau d'analyse en traitant le problème de la commutation de circuit.

4.2.3 Autre modélisation à base de fourmis artificielles

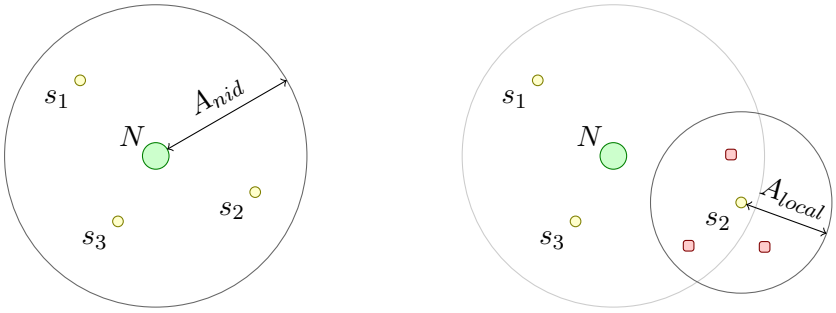
La méta-heuristique API, proposée par N. Monmarché [Monmarché 00], se base sur l'observation du comportement de prédation d'une espèce de fourmis particulière : *Pachycondyla apicalis*. Celles ci sont considérées comme relativement primitives car elles n'utilisent pas de système de communication à base de phéromones et ont un comportement de chasse assez basique. Plus précisément, ces fourmis se caractérisent par les points suivants :

- les fourmis vivent en petites colonies ;
- comme elles ne savent pas construire de nids, les fourmis s'installent dans des souches de bois en décomposition, ce qui les oblige à déménager régulièrement ;
- ces fourmis n'utilisent pas de phéromones mais sont capable de repérer visuellement un chemin et donc peuvent retourner sur un emplacement précis ultérieurement ;
- chaque fourmi possède son(ses) site(s) de chasse qu'elle visite régulièrement ;
- une visite fructueuse dans un site de chasse encourage la fourmi à revenir sur ce même site. A l'inverse, un nombre trop grand d'échecs conduit à l'abandon du site. La recherche initiale de sites de chasses est faite aléatoirement.

D'un point de vue algorithmique, la chasse se traduit par l'optimisation d'une fonction objectif f définie sur un espace de recherche S et se base sur l'utilisation de deux opérateurs :

- $O_{Rand}(S)$: génère un point aléatoirement dans l'espace S.
- $O_{Explo}(s)$: génère un point dans le voisinage d'une solution s.

Ce dernier opérateur peut être utilisé à la fois pour la création de nouveaux sites de chasses (si il est appliqué au nid) et l'exploration locale d'un site de chasse (si il est appliqué au site en question). Un paramètre d'amplitude permet de définir la portée de la recherche. Ce paramètre est respectivement nommé A_{nid} pour la recherche de nouveaux sites de chasses et A_{local} pour l'exploration locale d'un site.



(a) La fourmi sort du nid et établi 3 sites de chasse autour du nid N

(b) La fourmi explore 3 solutions autour du site de chasse s_2

FIG. 4.2 – Utilisation des deux opérateurs d'API

L'avantage principal de ce modèle est que, contrairement à ACO, il ne nécessite pas la mise en place d'une structure de données modélisant des phéromones artificielles. Cette structure pouvant éventuellement être complexe à définir ou à implémenter, l'utilisation d'API au lieu d'ACO peut simplifier la résolutions de certains problèmes. API a notamment été utilisé pour la résolution du problème du voyageur de commerce, l'optimisation de fonctions à variables continues ainsi que l'apprentissage de modèles de Markov cachés [Aupetit 05].

4.3 Systèmes immunitaires

Les systèmes immunitaires sont un mécanisme de défense complexe qui a été mis au point par les organisme vivants au cours de leur évolution [Aickelin 04]. Ce dernier est animé par la rivalité de deux éléments : les antigènes et les anticorps. Les anticorps traquent les antigènes selon une stratégie de prolifération et de mutation dans le but de les éliminer. Plus particulièrement, parmi les globules blancs, les lymphocytes sont les cellules les plus intéressantes. Celle-ci se différencient selon deux types B et T :

- Les cellules de type B sont chargées d'identifier les cellules à détruire. Elles sont responsable de la production d'anticorps. Leur tâche est spécialisée selon un antigène donné.
- Les cellules de type T sont spécialisées et se divisent en trois types selon leur tâche. Les « auxiliaires (helper) » contrôlent l'activité des cellules B. Les « suppresseurs (suppressor) » régularisent l'activité des autres cellules en mettant un terme à la réponse immunitaire dès qu'un antigène a été éradiqué. Enfin, les cellules « Cytotoxiques », ou « tueuses (killer) » s'attaquent aux cellules infectées et les détruisent.

Le système **ImmuneSearch** [Ganguly 04, Di Caro 05], présenté au chapitre 1 utilise ce principe pour la recherche d'informations. Un parallèle y est fait entre modèle biologique et système d'information décentralisé où un antigène est une information et les anticorps des requêtes émises par les utilisateurs. La prolifération des anticorps permet d'explorer l'ensemble du réseau alors que leur mutation leur permet de s'adapter aux différents antigènes rencontrés. Un anticorps capable de détruire un antigène est synonyme d'une requête satisfaite.

4.4 Systèmes épidémiques

Les systèmes épidémiques s'inspirent du mode de propagation des maladies contagieuses. Ces modèles se basent sur l'utilisation d'un graphe dont les sommet sont les différents individus et les arcs les relations à travers lesquelles la maladie peut être transmise. Selon leur état vis-à-vis de la maladie, les individus changent de statut au court du temps et passent ainsi, par exemple, du statut de contagieux à celui de contaminant. La fréquence de ces changements est définie par des taux d'infections, de guérison, *etc.* Selon le nombre d'états ainsi que leur enchaînement, il est possible de distinguer plusieurs modes de propagation épidémique. Nous présenterons ci-après quelques modèles épidémiques simples avec pour chacun, le modèle continu et discret dans le temps. Les travaux de [Allen 94] ont permit de démontrer que ces deux modélisations étaient équivalentes à condition de disposer d'une population constante dans le temps.

4.4.1 Susceptible-Infecté (Susceptible-Infected - SI)

Le modèle de propagation le plus simple baptisé « SI », pour « Susceptible-Infected », est composé de deux états. Le premier « Susceptible » où l'individu est en bonne santé et peut être contaminé. Le second « Infected » où l'individu est malade et peut transmettre la maladie. Les deux suppositions faites dans ce modèle sont que tout individu contaminé est un vecteur de transmission et tout individu sain peut être contaminé. L'enchaînement de ces états se fait comme suit :

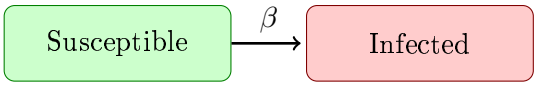


FIG. 4.3 – Chaîne d'états du modèle épidémique SI

Il est possible de définir une description continue de ce modèle afin de déterminer la taille de chaque population [Allen 94]. Soit S_n et I_n le nombre d'invididus respectivements dans l'état **S** ou **I**. La population est constante égale à N individus. Les conditions initiales sont $S_0 > 0$ et $I_0 > 0$ de telle sorte que $N = S_0 + I_0$. Entre deux itérations, le nombre individus contaminés dépend de la virulance de la maladie. C'est à dire de sa force d'infection λ qui se définie ainsi[2] :

$$\lambda = \frac{\text{nombre de nouvelles infections}}{\text{nombre d'invidus susceptibles exposés} \times \text{durée moyenne d'exposition}}$$

On suppose que chaque individu contagieux ne contamine qu'une personne par itération. La force d'infection est alors égale au produit de deux probabilités : celles pour qu'un individu

[2] source wikipedia

infectieu contamine un susceptible β et celle définissant la fraction de la population en contact avec des individus infectueux I_n/N :

$$\lambda = \beta\frac{I_n}{N} = \frac{I_{n+1} - I_n}{S_n \Delta t}$$

Nous prendrons ici l'hypothèse que la population restant constante, il est alors possible de déduire S_{n+1} de I_{n+1}. Ainsi, le modèle continu de SI se propose sous cette forme [Allen 94] :

$$\begin{cases} S_{n+1} = S_n - rS_nI_n \\ I_{n+1} = I_n + rS_nI_n \end{cases}$$

Avec $r = \frac{\beta\Delta t}{N}$ le taux d'infection de la maladie.

En prenant l'hypothèse d'un Δt court, la différence entre deux états permet d'établir une dérivée partielle :

$$\frac{S_{n+1} - S_n}{\Delta t} \simeq \frac{\partial S}{\partial t}$$

Il est possible de retrouver la formulation continue de SI :

$$\begin{cases} \frac{\partial S}{\partial t} = -\beta S(t)\frac{I(t)}{N} \\ \frac{\partial I}{\partial t} = \beta S(t)\frac{I(t)}{N} \end{cases}$$

Le facteur $\beta S(t)I(t)$ porte souvent le nom de facteur d'incidence et est caractéristique de la virulance de la maladie.

4.4.2 Susceptible-Infected-Susceptible (SIS)

Le modèle SIS étant SI en ajoutant, pour un individu, la possibilité de guérir. Cette guérison correspond à un retour dans l'état susceptible et se fait selon un taux γ qui correspond à la probabilité pour qu'un individu infecté soit retiré pendant une unitée de l'intervale de temps.

FIG. 4.4 – Chaîne d'états du modèle épidémique SIS

A population constante, l'évolution discrète de la population de ce modèle est définie ainsi [Allen 94] :

$$\begin{cases} S_{n+1} = S_n - rS_nI_n + bI_n \\ I_{n+1} = I_n + rS_nI_n - bI_n \end{cases}$$

avec $b = \gamma\Delta t$ le taux de guerison entre deux itérations. En prenant une hyptothèse similaire à la précédente, il est également possible de retrouver le modèle continu :

$$\begin{cases} \frac{\partial S}{\partial t} = -\beta S(t)\frac{I(t)}{N} + \gamma I(t) \\ \frac{\partial I}{\partial t} = \beta S(t)\frac{I(t)}{N} - \gamma I(t) \end{cases}$$

Tout comme pour SI, nous nous sommes ici placé dans l'hypothèse d'une population constante. Ce qui revient à dire qu'aucune mort n'est générée par la maladie ou que le nombre de naissance permet de stabiliser le nombre d'individus au cour du temps. D'autres modèles permettent notamment de prendre en compte une période d'infection et le décès éventuel des individus [Hethcote 95] mais leur présentation dépasserait ici le cadre de notre propos. On remarque que dans ce modèle, les individus oscillent perpétuellement entre les deux états. D'autres modèles plus complexes, tels que Susceptible-Infected-Removed-Recovered (SIR) et Susceptible-Exposed-Infected-Removed (SEIR) permettent d'éviter de telles oscillations en prenant en compte des phénomènes naturels.

4.4.3 Susceptible-Infected-Removed (SIR)

SIR ajoute à SIS la notion de vaccination ou de mort de l'individu. En effet, un individu contaminé ne vas pas nécessairement redevenir malade après sa guérison et peut être considéré comme immunisé après avoir été contaminé une première fois. Le modèle SIR contient donc un état supplémentaire pour prendre en compte cet état final.

FIG. 4.5 – Chaîne d'états du modèle épidémique SIR

La version discrète de ce modèle est la suivante [Allen 94] :

$$\begin{cases} S_{n+1} = S_n - rS_nI_n \\ I_{n+1} = I_n + rS_nI_n - bI_n \\ R_{n+1} = R_n + bI_n \end{cases}$$

Alors que la version continue prend cette apparence :

$$\begin{cases} \dfrac{\partial S}{\partial t} = -\beta S(t)\dfrac{I(t)}{N} \\ \dfrac{\partial I}{\partial t} = \beta S(t)\dfrac{I(t)}{N} - \gamma I(t) \\ \dfrac{\partial R}{\partial t} = \gamma I(t) \end{cases}$$

Ce modèle correspond à ce qui est observé dans un réseau lorsqu'une diffusion de type « broadcast » est utilisée.

4.4.4 Susceptible-Exposed-Infected-Removed (SEIR)

Le modèle SEIR ajoute au SIR un état « exposé » dans lequel l'individu est infecté mais non infectieux Ce modèle s'est montré particulièrement efficace dans le cadre de l'étude de la propagation de maladies infantiles.

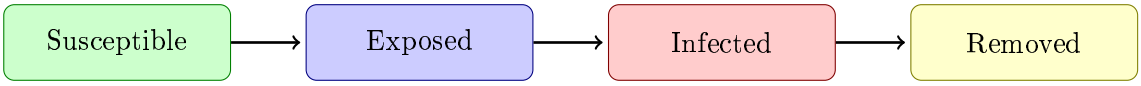


FIG. 4.6 – Chaîne d'états du modèle épidémique SEIR

$$\begin{cases} S_{n+1} = S_n - rS_nI_n \\ E_{n+1} = E_n + rS_nI_n - aE_n \\ I_{n+1} = I_n + aE_n - bI_n \\ R_{n+1} = R_n + bI_n \end{cases}$$

4.4.5 D'autres modèles...

A l'instar de SIS qui permet de boucler SI, il existe des variantes SIRS et SEIRS de SIR et SEIR [Hethcote 89b]. D'autre part, il existe une multitude de modèles rajoutant chacun leur nouveaux lots d'états. Le Progressive Susceptible-Infected-Detected-Removed (PSIDR) [Leveille 02] permet de prendre en compte deux périodes temporelles différentes dans l'évolution d'un réseau : la « Pré-réponse » et la phase « Réponse ». Ce modèle décrit ainsi la propagation d'un virus qui après une phase de prolifération (de type SI) suit une phase de déclin (de type SIDR) engendré par la présence d'anti-virus qui une fois mis à jour détectent et éradiquent l'infection.

Les modèles de SIR et SIS à population non constante sont présentés dans [Hethcote 89a]. SEIRS avec incidence non linéaire, c'est à dire avec une force d'infection variable, est présenté dans [Hethcote 91]

4.4.6 Adaptation des modèles épidémiques à la diffusion de messages dans les réseaux

Une des premières applications des systèmes épidémiques aux problèmes informatiques est la maintenance des base de données distribuées. Le problème est le suivant : s'assurer de synchronisation du contenu d'une structure de données répartie sur un ensemble de pairs en prenant en compte les modifications effectuées par chacun d'entre eux. Ces travaux sont attribués à Demers et datent de 1988 [Demers 88]. Ils tiennent en la proposition d'un algorithme de bavardage « gossip » donc le principe de fonctionnement est simple : sur réception d'un nouveau message, le pair choisit, parmi le sous ensemble des nœuds qu'il connaît, certains destinataires auxquels il envoie le message avec une probabilité donnée. Les destinataires ayant reçu le message, l'exploitent (par exemple en y ajoutant des modifications personnelles) et répètent le processus. Dans un environnement dynamique, où des nœuds arrivent et partent continuellement, il se peut que certains messages se diffusent mal. Pour éviter qu'une rumeur (résultant d'un bavardage) ne meurt trop vite, un processus d'anti-entropie est couramment employé. L'idée principale est qu'un des nœuds peut, périodiquement, s'enquérir auprès de son voisinage de messages qu'il n'aurait pas reçu.

D'autres travaux ont été menés l'utilisation de modèles de type SI pour la diffusion d'information dans un réseau de type MANET. La particularité de ces réseaux est que les communications ne peuvent se faire qu'avec certains voisins à portée de communication. Il est donc nécessaire

d'adapter la diffusion des message à la densité locale du voisinage. Khelil *et al.* [Khelil 02] ont abordé ce problème en cherchant une réponse à la question suivante : « comment régler le taux de diffusion en fonction de la densité du voisinage de façon à toucher un certain pourcentage de nœuds dans un intervalle de temps donné ? » et ont pu mettre en évidence un lien entre le taux d'infection et la taille du voisinage. Il existe une densité pour laquelle le taux de diffusion est optimal.

L'utilisation de l'anti-entropie est quasi-inévitable afin d'obtenir un algorithme de bavardage fiable pour un réseau. C'est à dire assurant qu'un message est reçu par tous les nœuds. Cependant cela n'est pas une condition suffisante et certaines topologies mettent ces algorithmes en difficulté. Cela se produit essentiellement quand l'ensemble des connexions du graphe forme une clique permettant une communication directe entre n'importe quel couple de pairs (c'est à dire, quand le graphe est complet).Ce problème est illustré par la figure 4.7 inspirée de [Lin 00].

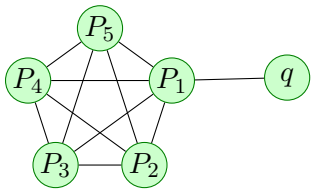


FIG. 4.7 – Exemple de graphe mettant l'algorithme de bavardage en difficulté

Considérons un réseau constituée d'une clique formée par les pairs P_1, P_2, P_3, P_4, P_5. Un sixième pair, q n'a pour unique connexion P_1. Le problème consiste à détecter ce type de topologie afin d'ajuster les probabilités de retransmission en conséquence. La solution **Directional Gossip** [Lin 00] est un protocole de bavardage à deux niveaux. Un protocole « classique » est utilisé entre les pairs formant une clique et une inondation (« flooding ») est utilisé sur les connexions reliant les cliques entre elles. Les pairs mettent à jour une estimation du plus court chemin les séparant de leur voisins. Ceux étant les plus isolés sont destinataires prioritaires pour le bavardage.

On suppose que, pour le graphe de la figure 4.7, P_2 envoie un message m à P_1. Ce dernier va à son tour retransmettre à B pairs de son choix parmi ses $n-1$ voisins (-1 car on ne tient pas compte de P_2). La probabilité pour que q fasse partie de ce choix s'exprime ainsi :

$$1 - \frac{\text{nombre de choix n'incluant pas } q}{\text{nombre de choix possibles}} = 1 - \frac{C_B^{n-2}}{C_B^{n-1}} = 1 - \frac{(n-2)!}{B!(n-2-B)!}\frac{B!(n-1-B)!}{(n-1)!} = \frac{B}{n-1}$$

Plus B est proche de $n-1$, meilleures sont les chances pour q de recevoir le message. Le problème étant que pour $B = n-1$ la stratégie de bavardage est alors équivalente à de l'inondation, accompagné de ses problème de surchage du réseau. Pour éviter ce cas extrême tout en ayant une diffusion la plus fiable possible, un protocole de bavargade doit peut non seulement prendre en compte la topologie de son entourage mais également un retour d'informations du fait de ses précédents envois.

La probabilité de retransmission peut donc être calculée de différentes manières selon les objectifs visés par l'application et doit dans certains cas être dynamique afin d'optimiser les performances. Selon [Burmester 06], ce sont trois classes d'algorithmes de bavardage qui existent selon que leur méthode de calcul est fixe, dynamique ou adaptative :

- fixe : la probabilité de retransmission a une valeur fixée. Celle-ci est paramétrée et ne peut s'adapter aux possibles conditions changeantes de la topologie du réseau ou des besoin des applications. C'est le cas du modèle proposé par Demers [Demers 88].
- dynamique : une information locale est utilisée afin de déterminer la probabilité de retransmission. L'algorithme utilisé dans PlanetP [Cuenca-Acuna 02, Cuenca-Acuna 03] en est un exemple.
- adaptative : la probabilité de retransmission est calculée en fonction d'une variable aléatoire, de la stucture du réseau et des réponses reçues suite aux précédents envois. Le protocole **Smart Gossip** [Roy 06] en est un exemple.

4.5 Conclusion

Ce rapide panorama nous a permit d'introduire quelques modèles dont nous nous sommes servis ou inspirés afin de concevoir les outils présentés dans cette thèse. Mais les fourmis, les virus et les anticorps ne sont pas les seuls représentants du « vivant » à avoir suscité la curiosité de la communauté informatique.

Wokoma *et al.* [Wokoma 02a] se sont, par exemple, intéressés au comportement des **lucioles** pour la mise au point d'un algorithme de bavardage. Isolés, ces insectes ont la particularité d'émettre un signal lumineux à un intervalle régulier qui leur est propre. Une fois en groupe, leur réaction est de modifier cet intervalle afin de synchroniser les émissions lumineuses de chaque individu. Ramené au domaine informatique, la fréquence d'allumage des lucioles devient une fréquence de mise à jour pour les pairs d'un réseau. Considérant un ensemble de données réparti sur tout le réseau, un pair désirant communiquer une mise à jour n'aura qu'à clignoter plus rapidement pour qu'à leur tour les voisins cherchent à se mettre à jour plus rapidement et ainsi de suite.

Les écureuils ont aussi une place à prendre dans cette jungle de modèles bio-inspirés.Le comportement de ces rongeurs trouve en effet une application dans le cadre de la gestion de système de fichier distribué dans un réseau P2P [Camorlinga 03, Camorlinga 04]. Les écureuils constituent des réserves de noisettes : quand des fruits sont trouvés, ceux-ci sont acheminés en direction d'une ou plusieurs cachettes aménagées dans les arbres. L'objectif est le remplissage égal de chacune de ces réserves et si une cachette est pleine ou trop peu remplie, les stocks sont équilibrés. Le parallèle informatique attribue à chaque pair d'un réseau P2P un ou plusieurs emplacements où stocker des paquets de données (le pendant des noisettes). Quand des fruits artificiels sont produits d'informations par un des pairs, un des animaux en prend une partie et cherche des cachettes disponibles dans le réseau.

D'autres approches telles que l'optimisation par essaim particulaire (OEP), développée par Eberhart et Kennedy en 1995 [Eberhart 95], ont également trouvé leur place dans la gestion des réseaux de capteurs. Son principe de fonctionnement s'inspire du comportement qu'adoptent les nuées d'oiseaux ainsi que les bancs de poissons lors de leur déplacements. L'OEP se base sur l'utilisation d'un certain nombre de particules. Ces particules, qui représentent autant de solutions potentielles pour le problème, se déplacent dans l'espace des solutions. Chacune d'elle contient une mémoire contenant, à un instant t, sa vitesse, sa position ainsi que la meilleure position trouvée jusqu'à présent. Une autre valeur mémorisée est celle trouvée globalement par les particules présentes dans son voisinage. L'interprétation de ce voisinage est liée au problème

consideré, il peut par exemple, s'agir de l'ensemble des machines connectées (dans le cadre d'un réseau) ou de l'ensemble des particules dont la distance est inférieure à un certain seuil. A chaque itération, de la méthode, les paramètres des particules sont mis à jour de la façon suivante en prenant en compte la vitesse de déplacement, les meilleures solutions trouvées (globalement et d'informations) et un paramètre d'intertie. Le résultat est un déplacement groupé de l'ensemble des particules.

Les deux modèles qui ont retenu notre attention pour le problème de la circulation d'information sont les fourmis artificielles et les systèmes épidémiques. La première pour l'aspect de mémoire collective et l'environnement stigmergique que l'utilisation des phéromones artificielles permet de mettre en place. La seconde car les algorithmes de bavardage permettent de considérer une diffusion de contenu indépendamment d'un processus de question/réponses tel que la recherche ou la diffusion par abonnement/publication. La contamination peut, en effet, se faire sans nécessiter un abonnement à un service de livraisons de virus ni avoir besoin d'effectuer une recherche explicite.

Nous avons choisi d'étudier l'hybridation de ces deux approches en créant dans PIAF un algorithme de distribution épidémique d'information utilisant une population de fourmis artificielles. Ce choix nous a mené, nous le verrons au chapitre suivant, à la création d'un système multi-agent réactif. Une instance de PIAF peut être vue comme un système pervasifs s'intégrant dans l'environnement de travail de l'utilisateur et capable de communiquer avec ses semblables. Notre objectif d'organisation des communications entre les PIAFs selon un modèle inspiré de la biologie permet de rapprocher la conception de PIAF de la recherche sur l'intelligence pervasive [Drogoul 02] et par la même de l'idée de rapprochement entre SMA et modèles d'informatique amorphe telle que suggérée par [Servat 02].

Le résultat de ces travaux, sous la forme d'une présentation de la plate-forme dans son ensemble et d'un ensemble d'expérimentations, font l'objet des chapitres suivants.

Chapitre 5

Description de la plateforme PIAF

Ce chapitre est consacré à la présentation de la plateforme PIAF que nous avons développé. Cette présentation commence par une définition générale du système et des algorithmes qui le composent.

5.1 Introduction

La recherche sur les navigateurs internet intelligents à évoluée pour passer de la mise au point d'outils d'assistance à celle consistant à rendre le Web lui même plus intelligent. Ce changement est récent et la définition de ce Web intelligent ainsi que ses caractéristiques restent à écrire mais il apparaît que ce dernier sera social et pro-actif. Les deux concepts présentés au chapitre 3, le Web de la sagesse et le bureau sémantique social mettent en avant ces deux aspects en facilitant le dialogue entre l'homme et la machine ou entre des utilisateurs désireux de partager des documents (tout en les gérant plus efficacement).

Dans ce contexte, l'objectif de la plate-forme PIAF (Personal Intelligent Agent Framework) est la mise en place d'un service d'échange de ressources pro-actif et transparent. Ce service permet aux utilisateurs de s'affranchir des trois principaux problèmes ressentis lors de l'utilisation d'outils de travail collaboratif (CSCW) ; à savoir, le besoin de connaissance mutuelle, le manque de motivation et le besoin d'expressivité, en ayant trois objectifs :

1. Être capable de détecter quels sont les utilisateurs ayant des centres d'intérêt en commun et faciliter leur mise en relation ;
2. Avoir un fonctionnement transparent et non intrusif pour l'utilisateur ;
3. Ne se baser que sur des données acquises par l'observation du comportement de l'utilisateur.

Le positionnement global de PIAF dans l'environnement de travail de l'utilisateur est représenté sur la figure 5.1. Ce dessin est composé de différents blocs fonctionnels dans lequel sont précisés un ou deux éléments caractéristiques. Ces blocs sont en interface entre l'utilisateur et un réseau autorisant la communication entre différents utilisateurs.

Dans le cadre de ses activités, l'utilisateur va utiliser différentes applications. Parmi celles-ci, nous pouvons citer le Navigateur Internet lui permettant d'aller consulter le Web ou le traitement

de texte utilisé pour la rédaction de documents divers. Comme c'est déjà le cas pour d'autres systèmes, PIAF part de la supposition que ces activités traduisent les centres d'intérêt de l'utilisateur. Ainsi, par exemple, l'enregistrement d'une adresse de site Web traitant de l'élevage des chevaux dans le marque pages du navigateur ainsi que la rédaction d'un document abordant le même sujet seront un indice de l'intérêt que porte l'utilisateur pour les équidés. La fonction identifiée comme la « Création d'informations » permet de partager le fruit de ces activités avec les autres utilisateurs. Les informations dont il est question seront, pour cet exemple, formulé ainsi : « Il existe une page intéressante sur les chevaux à cette adresse » et « Un document parlant de chevaux est disponible sur la machine d'un utilisateur ». Celles-ci se permettent à un utilisateur qui les consulte de prendre connaissance de l'existence de ressources sur le Web (pour le premier cas) ou dans le réseau d'utilisateur (pour le second exemple). Cette génération d'information est effectuée de façon transparente pour l'utilisateur en étant intégrée dans les outils qu'il utilise habituellement ; ce qui correspond à l'objectif 3. Nous ne nous attarderons pas sur les problèmes de confidentialité et de légalité de contenu qui se posent dès qu'il est question de partager du contenu. Ceux-ci ne concernent pas la circulation d'information proprement dite et sont donc hors de notre propos.

Les descriptions qui suivent s'appuient sur la figure 5.1. Une fois créées, les informations sont mises en circulation dans le réseau des utilisateurs. Chaque utilisateur dispose d'un PIAF installé sur sa machine et chaque PIAF est capable à la fois de produire ou de consommer des informations, il s'agit donc d'un réseau P2P. Les flux d'informations dans ce réseau sont gérés par la fonction de « circulation des informations » située dans le bloc numéro 3. Dans un premier sens, cette fonction émet sur le réseau les informations résultantes de l'activité de l'utilisateur. Le second sens de fonctionnement permet de faire remonter à l'utilisateur les informations récupérées suite aux différents échanges. Afin d'en faciliter l'exploitation, une fonction permet la « consultation des informations et de l'état du réseau » en présentant ces données sous une forme accessible pour l'utilisateur. A sa guise, l'utilisateur peut interroger ces outil pour consulter les informations récupérées (objectif numéro 2 de notre liste de départ). Des outils issus de la recherche sur les bureaux sémantiques sociaux ou le « WisdomWeb » peuvent également tirer partie de ces données (*cf* bloc numéro 4 de la figure 5.1). L'ensemble données récupérée constitue pour le « WisdomWeb » une source d'informations supplémentaire venant s'ajouter à l'utilisation de moteurs de recherche ou la consultation d'autres services Web. Alors que, dans sa définition générale, le bureau sémantique social permet aux utilisateurs de partager du contenu, l'exploitation des résultats de la circulation d'information facilite également l'échange de connaissance. Prenons, pour exemple, la publication d'une information concernant un événement à venir et produite suite à l'ajout d'un rendez-vous dans un logiciel de gestion d'emploi du temps. Aucun contenu n'est à partager suite à cet ajout[1] mais l'information concernant l'existence de cet événement peut être échangée entre les pairs.

Après avoir répondu à nos deux premiers objectifs, il ne reste plus qu'à traiter la question de la détection d'utilisateurs ayant des centres d'intérêts partagés. Cette tâche est confiée à la brique intitulée « Création du réseau social », située dans le bloc 3, figure 5.1. Les algorithmes situés dans cette brique observent les flux d'informations et émettent des hypothèses concernant les centres d'intérêt des membres du réseau. Celles-ci sont utilisées pour mettre en place un réseau social défini par la relation « a les mêmes centres d'intérêt que ».

[1] nous supposerons ici que le fichier dans lequel sont enregistrés les données de l'application n'est pas consideré comme étant *facilement* partageable

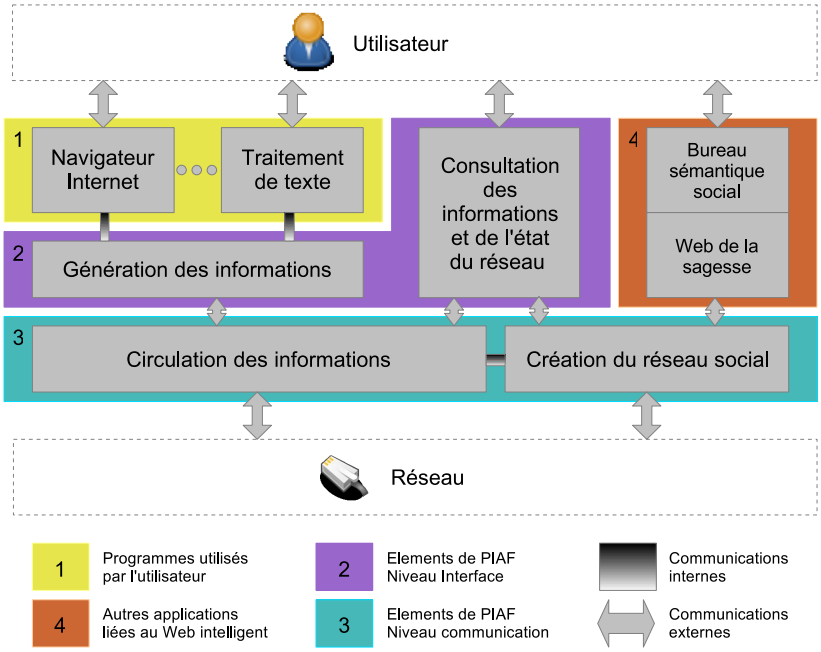


FIG. 5.1 – Vision globale de l'implication de PIAF dans l'environnement de travail de l'utilisateur

La plate-forme PIAF se compose donc de quatre éléments répartis dans deux blocs fonctionnels :

- un niveau « communication » (bloc numéro 3) qui aborde la problématique suivante : diffuser de l'information automatiquement entre les utilisateurs et les connecter entre eux selon leurs centres d'intérêts communs. Il s'agit d'un niveau servant de base au développement des autres blocs. C'est pourquoi nous avons concentré notre travail sur la création des algorithmes permettant sa mise en place ;
- un niveau « interface » (bloc numéro 2) qui a une utilité double du point de vue de l'utilisateur, il s'agit de présenter de façon utilisable l'ensemble des informations glanées par le système de diffusion du premier niveau. Du point de vue inverse, il s'agit de détecter les activités de l'utilisateur afin de générer puis mettre en circulation les informations adéquates. Nous avons déjà eu l'occasion lors des, précédents chapitres, de mettre en évidence le niveau de complexité du dialogue homme<->machine. Durant notre travail, afin de nous focaliser sur le niveau « communication », nous avons decidé dans un premier temps de faire abstraction de ce niveau en supposant que des outils tels que les interfaces à saisie nulle de données [Sharon 03] (ou « *Zero-Input Interfaces* ») ou des cartes de visualisation avec parcours à profondeur multiple tel que celui utilisé dans **Margin Notes** [Rhodes 00] pourraient être mise en œuvre pour ces tâches. Du point de vue expérimental, cette supposition se traduit par l'utilisation de vecteurs de données artificiels.

Au niveau communication, les données dont dispose PIAF aussi bien en entrée qu'en sortie sont un flux d'informations provenant des pairs auxquels il est connecté. Localement, un PIAF a la possibilité d'agir sur la part de flux qu'il génère ainsi que sur le contenu de son entourage. Il lui suffit de modifier le contenu de ses émissions d'informations ou de se déplacer dans le réseau (*ie*, de supprimer ou créer des connexions avec d'autres pairs). En dehors du domaine informatique, ces deux postulats se retrouvent dans une situation sociale de la vie courante : celle de la discussion

de groupe. Considérons un groupe d'individus en train de tenir une discussion libre où chacun communique un certain nombre d'informations dont il dispose et écoute ce que les autres ont à lui dire. Selon l'adéquation entre les centre d'intérêts de ces individus et le nombre d'informations à échanger, la discussion sera plus ou moins fournie (régulation du flux d'informations). Egalement, dans l'hypothèse ou le groupe est suffisamment grand, il se peut qu'un individu décide de prendre congé de ses compères pour aller s'entretenir avec d'autres personnes (modification du voisinage).

Les algorithmes situés dans le niveau « communications » de PIAF sont construits autour de la métaphore suivante : les pairs sont des individus se trouvant dans une salle commune où chacun est libre de s'exprimer et d'écouter (réseau P2P), aucun d'entre eux ne dispose de signe visible permettant d'annoncer quels sont ses centres d'intérêts (aucun profil n'est publié) et les discussions se résument à une libre expression (aucune demande explicite d'information n'est formulée). Comme nous venons de le voir, dans cette situation les individus auront tendance à se déplacer jusqu'à former des groupes autour de centres d'intérêt communs. Un autre phénomène est intéressant à modéliser : alors qu'un individu s'exprime, les personnes qui lui sont les plus proches entendront parfaitement le message alors que les plus éloignées, jusqu'à une certaine limite, ne l'entendront que partiellement et vraisemblablement avec un contenu altéré. Nous considérerons également qu'un individu répétera systématiquement à l'un de ses voisins le contenu d'une information entendue. A supposer qu'il trouve dans son voisinage un voisin susceptible d'être intéressé par ladite information. Cette estimation se base sur les précédents envois : pour reprendre notre exemple précédent, un pair émettant fréquemment des informations relatives aux chevaux sera vraisemblablement intéressé pour en recevoir d'autres.

Nous proposons d'utiliser un algorithme à base de fourmis artificielles pour la circulation des informations dans le réseau. Ces informations sont assimilées à des denrées périssables que les fourmis déplacent de nid en nid. Des phéromones artificielles sont associées à chaque chemin menant d'un nid à un autre (soit, une connexion entre deux pairs). Celles-ci jouent le rôle d'une mémoire globale des informations échangées dans le réseau et permet d'estimer les centres d'intérêts de chacun. La modification de la topologie est faite en évaluant périodiquement la qualité du voisinage. Les pairs avec lesquels peu d'informations sont échangées sont détectés et déconnectés.

La suite de ce chapitre est consacrée à une présentation détaillée de PIAF ainsi que de ses applications.

5.2 Description du système de gestion de communications de PIAF

Notre description suivra une logique englobante en partant de l'élément échangé entre les pairs, les informations, pour terminer par une présentation des flux observés à l'échelle du réseau.

5.2.1 Contenu des informations en circulation dans le réseau

Une information décrit une ressource partagée. Elle est composée de différents éléments que nous pouvons classer en deux groupes. Le premier groupe concerne les éléments « utiles » qui

sont exploités par les outils situés dans le niveau « interface » de PIAF. Ceux-ci ne sont pas pris en compte lors du processus de diffusion des messages. Ces éléments sont :

❶ **Un type**
Le type permet de savoir quel est la nature de la resource partagée. Il s'agit d'un champ texte librement paramétrable par l'application qui créée l'information.

❷ **Une URI** (Uniform Ressource Indicator)
L'URI indique où se trouve la ressource partagée. Elle peut prendre la forme d'une adresse de site Internet ou utiliser un espace de nommage interne pour indiquer que la ressource ce trouve sur l'un des pairs utilisant PIAF.

❸ **Une limite de validité**
Date limite à laquelle l'information ne sera plus valable et pourra être automatiquement effacée par le système.

L'autre catégorie d'éléments, regroupe les variables utiles pour la circulation des messages :

❹ **Un vecteur de phéromones** τ défini dans $\mathbb{R}^n$
Le vecteur de phéromones définit la trace que laissera l'information sur son passage. Il s'agit des métadonnées en décrivant le contenu ou la nature telles que définies aux chapitres précédents.

❺ **Une liste de pairs déjà visités** $vus(I)$
Il s'agit d'une zone de mémoire destinée à éviter les risques de cycles.

❻ **Un compte-tours** $round(I)$
Défini quel est le nombre de pairs par lesquels cette information est passée. Ce compteur fourni une indication sur la distance parcourue dans le réseau depuis son origine.

❼ **Une date de péremption** $TTL(I)$
Date limite de vie d'une information en terme de déplacements dans le réseau. Cet indicateur n'est pas à confondre avec la date de validité explicitée plus haut. Il s'agit d'une commune aux algorithmes de diffusion de message dans un réseau, destinée à limiter le nombre d'envois et donc le risque de surcharge.

Aucune supposition n'est faite sur la nature de τ. Par contre, on suppose l'existance d'une fonction de similarité permettant de définir la ressemblance entre deux traces de phéromones.

$$s : \mathbb{R}^n \times \mathbb{R}^n \mapsto [0, 1]$$

Dans cette définition n correspond à la taille de l'espace de représentation des phéromones. Par exemple, si on considère que ces vecteurs sont issus d'une vectorisation de type TFIDF, cet espace correspond à l'ensemble des mots clé du dictionnaire.

5.2.2 Architecture interne d'un PIAF

La structure interne de PIAF, en se résumant aux éléments du niveau communication, est représentée figure 5.2. Trois groupes fonctionnels y sont identifiés : la gestion des stocks de nourriture du nid, la gestion des connexions avec les nids aux alentours et enfin les agents qui s'activent pour faire fonctionner le système. L'ensemble des activités est contrôlée par un module noyau qui s'occupe principalement de la circulation des messages entre les différents blocs fonctionnels.

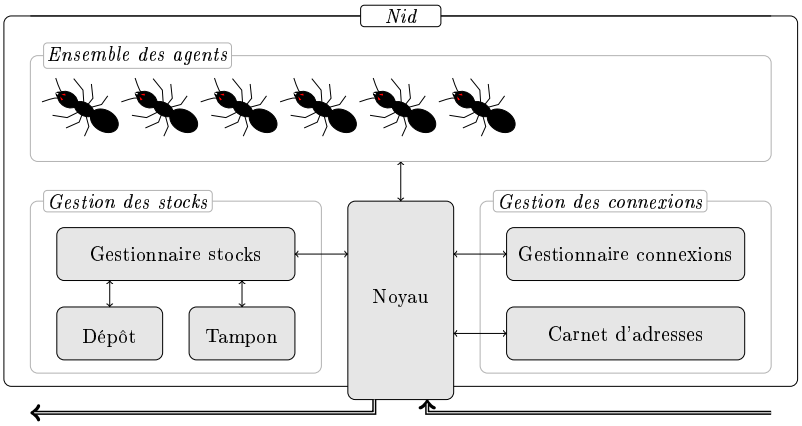


FIG. 5.2 – Description des éléments du niveau « communication » de PIAF

- Gestion des stocks

  Chaque information est assimilée à un bloc de nourriture. Le gestionnaire de stock est utilisé afin d'en gérer le stockage et les accès. Ce gestionnaire utilise deux entrepôts :
  - le **« dépôt »** contient des informations qui seront utilisées par le niveau supérieur de PIAF. C'est à dire la partie « Interface » ;
  - le **« tampon »** est une zone d'attente. C'est de cet entrepôt que les fourmis extraient les informations à faire circuler dans le réseau.

  Quand une information arrive dans le système, elle est dupliquée et un exemplaire est stocké dans chaque entrepôt. Cette information peut arriver dans le système soit à la suite d'une création par l'utilisateur, soit par transfert via le réseau. Dans le cas où elle est déjà présente dans la zone de stockage, seules les quantités sont actualisées. Pour qu'une information de la zone de dépôt puisse passer dans la zone tampon (et donc être (re)mise en circulation), une intervention de l'utilisateur est nécessaire. Les informations qui ne sont consultées par aucun utilisateur sont ainsi progressivement éliminées du système car non ravivées.

- Gestion des connexions

  La gestion des connexions est confiée à un processus lancé périodiquement. Son objectif est de s'assurer que le voisinage est correct et, si besoin, de l'optimiser en changeant les connexions établies.

- Ensemble des agents

  Dans cet ensemble se trouvent les fourmis ainsi que les autres agents du système. Le nombre de fourmis utilisé est un des paramètres du système.

5.2.3 Mise en réseau de plusieurs PIAFs

Une métaphore est faite entre un réseau où des données sont échangées et des nids de fourmis entre lesquels sont échangés des paquets de nourriture. Ces nids sont reliés entre eux par un ensemble de connexions dirigées correspondant à des différents chemins. Chaque fourmi est associée à un unique nœud, ses déplacements se résument à porter une information vers un nid par un chemin puis à en revenir en suivant le chemin inverse une fois sa cargaison déposée. Cet ensemble de nids ainsi que leurs connexions est représenté sur la figure 5.3. Les couleurs utilisées

pour les traces de phéromones indiquent des sujets prépondérants. Chaque couleur peut être vue comme définissant un sujet particulier associé à la ressource représentée.

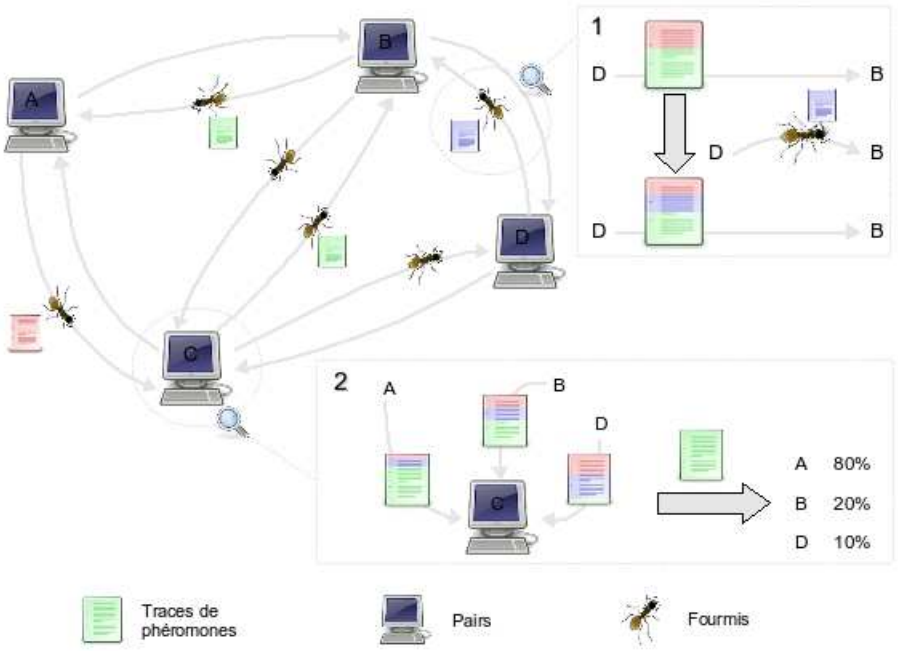


FIG. 5.3 – Vision globale

Les deux sous parties agrandies de la figure 5.3 apportent des précisions relatives à la mise à jour des phéromones et le mode de déplacement des fourmis.

Le premier représente l'exemple de la connexion qu'il existe entre les pairs D et B. Initialement, la phéromones associée à cette connexion, représentée par le document, se compose de deux couleurs. Ces deux couleurs correspondent à la trace qu'auront laissés deux informations ayant circulé entre D et B lors de précédents transferts. Une fourmi transporte une nouvelle information d'une troisième couleur, le vecteur de phéromones est alors mis à jour afin de prendre en compte ce nouveau transfert. Le passage d'une nouvelle information ayant une couleur déjà présente dans la trace résulterait quand à elle à une actualisation des proportions représentées.

La seconde zone détaillée illustre le processus de sélection de destination mis en œuvre par les fourmis. Nous reviendrons plus en détail sur celui-ci par la suite mais l'on peut remarquer sur cette représentation que ce choix se base sur la similarité qu'il existe entre la trace de phéromones associée à l'information à envoyer et celles associées aux connexions avec les autres pairs. Il est intéressant de remarquer que ce sont les connexions entrantes qui sont considérées. Le choix de la fourmi se base sur ce que les pairs auront précédemment envoyé afin de deviner ce qui devrait les intéresser. Du point de vue du pair depuis lequel la fourmi part, ces données sont associées aux connexions dirigées vers le nid.

D'un point de vue plus formel, nous définissons le réseau par un ensemble de nids n_i reliés par un ensemble de chemins dirigés (i, j). Une trace de phéromones $\tau_{i \to j}(t)$ est associée à un chemin allant du nid i au nid j (voir figure 5.4).

Contrairement aux traces de phéromones associées aux informations, les traces associées aux

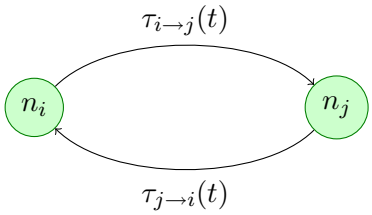


FIG. 5.4 – Modélisation des traces de phéromones

chemins évoluent au cours du temps alors que des données sont échangées. Lorsqu'une information ayant un vecteur de phéromones $\tau(I)$ est déplacée du pair n_i vers le pair n_j entre les instants t et $t^{'}$, les phéromones $\tau_{j\to i}$ sont mises à jour de la façon suivante :

$$\tau_{j\to i}(t^{'}) = (1-\rho)\cdot\tau_{j\to i}(t) + \rho\cdot\tau_I$$

Le facteur ρ est utilisé de façon à effectuer une translation dans l'espace de représentation des phéromones. Il est borné par une valeur ρ_{max} et dépend de $round(I,t)$ et $TTL(I)$.

$$\rho = \rho_{max}\cdot \mathrm{e}^{-\alpha\left(\frac{round(I,t)}{TTL(I)-1}\right)}$$

La figure 5.5 présente l'évolution du cœfficient ρ pour différentes valeurs de α. On remarque que ce coefficient joue un rôle de régulation. Plus sa valeur est élévée, plus l'impact de l'information sur la trace de phéromones associée à la connexion sera faible.

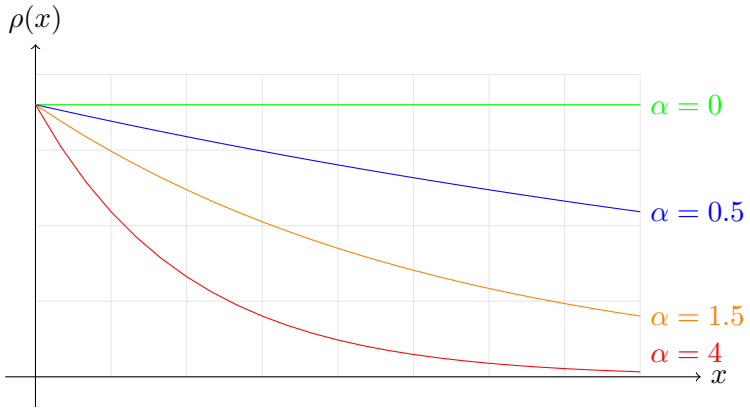


FIG. 5.5 – Evolution de $\rho(x) = 0.9\mathrm{e}^{-\alpha x}$

Cette section présente successivement les notations employées puis le modèle établi pour chaque élément.

Les notations générales sont résumées dans le tableau 5.3.

D'une manière générale, les choix probabilistes suivent la notation suivante : $P^{action}_{connexion}(params)$. Avec *connexion*, la connexion concernée, *action* l'action relative à cette probabilité et *params* les paramètres utilisés. Les différentes probabilités ainsi définies sont résumées dans le tableau 5.4.

5.3 Dynamique de fonctionnement

Nous décrivons ci-après l'activité de chaque élément du système. D'une manière générale, ces activités correspondent à l'automate représenté figure 5.6

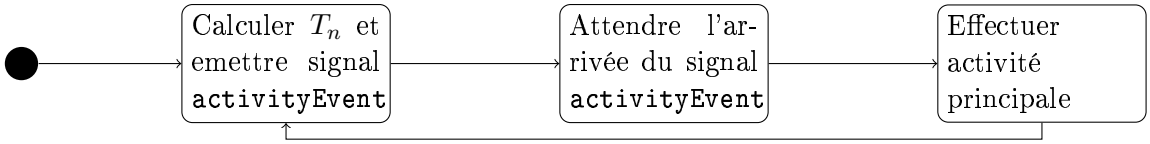


FIG. 5.6 – Activités des composants

Le paramètre T_n est une variable réglant la fréquence d'activité de l'élément concerné. Celle-ci est définie en prenant en compte une échelle de temps. Le contenu de l'activité principale dépend ensuite du composant selon qu'il s'agisse d'une fourmi, du gestionnaire de carnet d'adresses ou du gestionnaire de connexions.

5.3.1 Echelles de temps

Comme indiqué précédemment, l'ensemble du système est animé selon un système événementiel. Outre les événements liés à la circulation des informations, PIAF utilise un type d'événement particulier chargé de déclencher l'activité des modules. Cet événement, nommé « activityEvent », est comparable à l'horloge qui dicte la cadence de fonctionnement d'un processeur (si tant est qu'on fasse abstraction des considérations d'atomicité des opérations effectuées).

Soit une échelle de temps globale T. Nous introduisons 4 coefficients T^f, T^b, T^c et T^a correspondant respectivement aux périodes d'activité des fourmis, du bot, du gestionnaire de connexion et du carnet d'adresse. Leur valeur s'exprime en unités de temps comme étant un multiple de T. Concrètement, si $T = 2$ minutes et $T^c = 30$ le gestionnaire de connexion sera actif toutes les heures.

A la fin de chaque activité, les module calculent une nouvelle valeur pour leur cœfficient d'activité. Ce changement très fréquent du délai entre deux activités d'un même éléments nous permet de limiter les risques de résonance dans le réseau. C'est à dire la probabilité pour que deux éléments de deux pairs différents effectuent leurs actions de manière synchrone. Les valeurs obtenues sont indexées selon la variable n d'indice de temps. Ainsi, T^c_5 dénote la 5ème valeur de T^c calculée par le gestionnaire de connexions et, d'une manière générale, T^c_n indique la n-ème valeur calculée.

Ces cœfficients sont calculés en fonction d'une loi de probabilité triangle dont la fonction de densité de probabilité est représentée figure 5.7. Nous avons choisi cette loi car elle permet de prendre en compte une valeur minimale, une autre maximale ainsi qu'une estimation de la valeur « idéale » selon l'avis de l'expert qui paramètre le système. Toutes les précisions mathématiques concernant cette loi sont consultables sur le site de **Wikipedia**[2].

[2] `http://en.wikipedia.org/wiki/Triangular_distribution`, dernier accès le 25 novembre 2006

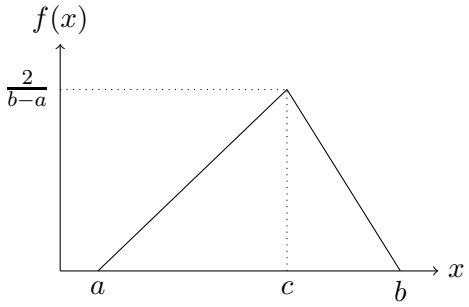


FIG. 5.7 – Fonction de densité de probabilité de la loi triangle

En introduisant trois nouvelles variables T_{min}, T_{max} et T_{mod} pour chaque coefficient, nous obtenons les lois suivantes :

$$\begin{cases} T_n^f \sim Triang(T_{min}^f, T_{mod}^f, T_{max}^f) \\ T_n^b \sim Triang(T_{min}^b, T_{mod}^b, T_{max}^b) \\ T_n^c \sim Triang(T_{min}^c, T_{mod}^c, T_{max}^c) \\ T_n^a \sim Triang(T_{min}^a, T_{mod}^a, T_{max}^a) \end{cases}$$

Afin de pouvoir s'affranchir de la contrainte temporelle durant les simulations, ces cœfficient sont exprimés les uns en fonction des autres. Nous avons pour cela fait un certain nombre de choix préalables :

- la loi de distribution est considerée dans le cas particulier où elle est symétrique, donc $T_{max} + T_{min} = 2T_{mod}$
- les agents obéissent tous à la même loi de probabilité, donc $T^b \simeq T^f$
- la valeur modale de l'activité des agents T_{mod}^f sert de temps de référence

En premier lieu, nous introduisons deux cœfficients k^c et k^a permettant respectivement d'exprimer T_{mod}^c et T_{mod}^a en fonction de T_{mod}^f. Un autre coefficient γ permet de définir l'étalement de la loi et donc l'écart entre le mode et une des bornes, minimum ou maximum.

$$\begin{cases} T_n^f \sim Triang((1-\gamma)T_{mode}^f, T_{mode}^f, (1+\gamma)T_{mode}^f) \\ T_n^b \sim Triang((1-\gamma)T_{mode}^f, T_{mode}^f, (1+\gamma)T_{mode}^f) \\ T_n^c \sim Triang((k^c-\gamma)T_{mode}^f, k^cT_{mode}^f, (k^c+\gamma)T_{mode}^f) \\ T_n^a \sim Triang((k^a-\gamma)T_{mode}^f, k^aT_{mode}^f, (k^a+\gamma)T_{mode}^f) \end{cases}$$

La dynamique générale du système est donc globalement définie par les paramètres suivants :

- T_{mod}^f : Mode de la période d'activité des agents. Jour le rôle de facteur d'échelle de temps
- k^c : Ratio entre la période d'activité du gestionnaire de connexion et celle des agents
- k^a : Ratio entre la période d'activité du carnet d'adresses et celle des agents
- γ : Facteur d'étalement de la loi dans l'échelle de temps

Une contrainte supplémentaire est imposée en considérant qu'il est nécessaire de mettre à jour le carnet d'adresses avant de regarder si les connexions avec le voisinage peuvent être optimisées. Cette contrainte prend la forme de l'inégalité suivante : $1 \leq k^a \leq k^c$.

5.3.2 Fourmi : Déplacement d'une information

L'activité des fourmis se résume au déplacement d'une information. Chaque fois qu'elle est active, la fourmi peut soit essayer de trouver un pair vers lesquels apporter l'information quelle tient soit en prendre une de la zone d'attente si elle est libre. Le statut de la fourmi est réglé par une variable booléenne *busy* qui est a vrai quand elle a en charge une information.

Algorithme 1 : Activité d'une `Fourmi`

```
1  if busy then
2  |  /* Choisir un pair de destination */
3  |  destination ← selectDestination()
4  |  if dest ≠ −1 then
5  |  |  Envoyer I à destination
6  |  |  busy ← false
7  |  end
8  |  else
9  |  |  round(I) ← round(I) + 1
10 |  |  if round(I) = TTL(I) then
11 |  |  |  effacer I
12 |  |  |  busy ← false
13 |  |  end
14 |  end
15 end
16 else
17 |  /* Essayer de prendre une information en attente */
18 |  busy ← grabInformation()
19 end
```

Un algorithme probabiliste permet de choisir quelle sera la destination n_j parmi les pairs du voisinage $V_i(t)$ qu'empruntera la fourmi. Cet algorithme se compose de plusieurs étapes qui commencent par le tri de l'ensemble des voisins en deux groupes selon s'ils sont où non (a priori) interessés par l'information à diffuser. Le résultat de ce classement est la création d'un groupe des pairs interessés $V_i(I,t)$ et d'un autre groupe de pairs non intéressés $\overline{V_i(I,t)}$.

$$V_i(I,t) = \{n_j \in V_i(t) \mid s(\tau_{i \leftarrow j}(t), \tau(I)) \geq s_{min}\} \quad (5.1)$$

$$\overline{V_i(I,t)} = \{n_j \in V_i(t) \mid n_j \notin V_i(I,t)\} \quad (5.2)$$

Lors de cette estimation, la fourmi met à jour une mémoire circulaire contenant les résultats d'évaluations des connexions. Pour la connexion de n_i vers n_j cette mémoire porte le nom de $E_{i \leftarrow j}(t)$. Si à la suite de la classification de la fourmi, le pair n_j est dans le groupe $V_i(I,t)$, alors la valeur 1 sera enregistrée dans cette mémoire. Dans le cas contraire, la valeur 0 est inscrite. Une fois le résultat enregistré, le curseur de la mémoire est déplacé vers la case suivante (voir figure 5.8).

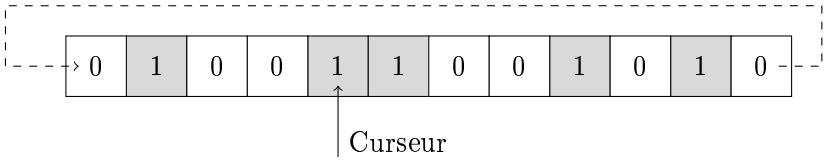
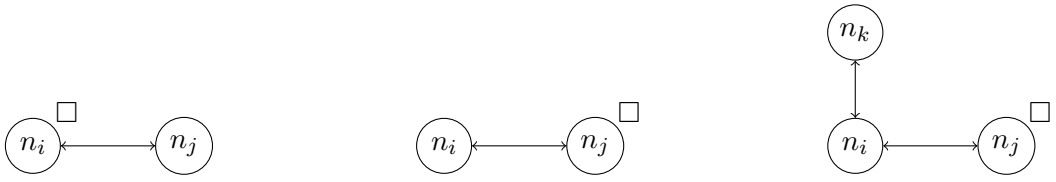


FIG. 5.8 – Exemple d'une mémoire $E_{i \leftarrow j}(t)$ contenant 5 évaluations positives sur 12

Cette information par la suite utilisés afin d'évaluer l'utilité d'une connexion lors du processus de modification du voisinage.

Une stratégie visant à optimiser les flux pourrait alors constituer en la sélection d'un des pairs parmi ceux du groupe $V_i(I,t)$. Ce sont en effet eux qui sont *a priori* les plus intéressés. Or, ce choix ne permet pas d'envisager l'exploration de l'espace du voisinage qui s'avère pourtant nécessaire dans l'optique de trouver de nouveaux pairs à qui se connecter. Un compromis doit donc être trouvé entre l'envoi vers des pairs dont on est confiant dans l'intérêt porté pour l'information et les prises de contact avec d'autres qui semblent moins interessés. Nous avons choisi de doter les fourmis d'une notion de libre arbitre. Une fois les différentes destinations réparties en deux groupes, la fourmi tire à pile ou face pour décider si une connexion du groupe $V_i(I,t)$ ou une de son opposé $\overline{V_i(I,t)}$ doit être choisie. Cette pièce est biaisée : la probabilité de choisir un pair de $V_i(I,t)$ est parametrée à une valeur η. Une fourmi peut également décider de ne pas quitter le nid en optant pour la direction n_i. Ceci permet de garder une information au nid et attendre que celle-ci se périme plutôt que de la diffuser quand le voisinage ne s'y intéresse pas.

Il est également préférable, pour optimiser le flux, d'éviter que les message n'effectuent de parcours cyclique entre les pairs. C'est la raison de la présence de la mémoire $vus(I,t)$ associée à chaque information I. Un pair recevant une information va s'ajouter à cette liste qui est en fait, comme le sont les compteurs d'évaluation, une mémoire circulaire. La taille de cette mémoire permet de définir un nombre minimal de noeds à consulter avant de pouvoir retourner sur une seconde fois sur un pair déjà visité. Tel qu'illustré par le scénarion de la figure 5.9, il peut être en effet nécessaire qu'une information soit renvoyée à un pair qui l'a déjà reçue afin que celle ci puisse atteindre de nouveaux arrivants. Etant donné que les informations ne sont pas dupliquées mais transferées d'un pair à l'autre, des situations de blocage peuvent se produire.



(a) n_i détient une information (représentée par le carré □). Il décide de l'envoyer à n_j

(b) n_j reçoit l'information et enregistre que n_i en est l'émetteur

(c) Un nouveau pair n_k se connecte à n_i. Etant donné que l'information est maintenant située sur n_j et qu'elle ne peut revenir sur n_i, n_k ne pourra la recevoir

FIG. 5.9 – Scénario conduisant à une situation de blocage pour une information

Nous aurions pu envisager d'autres stratégies pour pallier à ce problème. En particulier, les algorithmes d'anti-entropie et de génération de duplicatas sont destinés à autoriser de tels retours ou à générer des copies pour limiter les risques de se trouver dans ce genre de situation. Nous avons

préféré cette solution de mémoire limitée pour deux raisons : la première est qu'elle n'implique pas la mise en place d'un algorithme particulier de gestion de renvois ou de gestion de duplicata. La seconde est que le scénario considéré ne se place pas tout à fait dans des conditions exactes de l'utilisation de PIAF. En effet, il se peut que l'information, bien que restant sur le pair n_j puisse à nouveau réaparaître sous la forme d'un double et être émise vers n_i. Il suffit pour cela que l'utilisateur situé sur la machine n_k consulte l'information en question. Cette consultation entraine alors la génération de cette nouvelle information.

La liste des pairs déjà visitées est utilisée pour définir les ensembles $V_i^{good}(I,t)$ et $V_i^{bad}(I,t)$ des distinations pouvant être sélectionnées à un instant t, pour une information I.

$$V_i^{good}(I,t) = V_i(I,t) \setminus vus(I,t) \tag{5.3}$$

$$V_i^{bad}(I,t) = \overline{V_i(I,t)} \setminus vus(I,t) \tag{5.4}$$

Chaque destination potentielle se voit attribué une probabilité $P_{i\to j}^{send}(I,t)$ d'être choisie. Selon le classement de cette destination, trois cas différents peuvent se présenter :

1. $n_j \in V_i^{good}(I,t)$: Le pair est potentiellement interessé, ses chances d'être choisi seront d'autant plus forte que sa similarité avec l'information sera élevée ;
2. $n_j \in V_i^{bad}(I,t)$: Il s'agit d'un pair *à priori* non interessé qui aura autant de chances d'être choisi que les autres pairs de $V_i^{bad}(I,t)$.
3. Pour $n_j = n_i$, cas où la fourmi décide de rester sur place, nous considérerons que les fourmi ont une certaine prédisposition n^+ à rester sur place. Par exemple, pour $n^+ = 5$ une fourmi aura 5 fois plus de chances de rester à son nid plutôt que de se rendre vers un pair classé dans $V_i^{bad}(I,t)$.

La probabilité $P_{i\to j}^{send}(I,t)$ prend donc trois valeurs différentes selon j. Rester au nid etant consideré comme équivalent à sélectionner un pair non intéressé pour destination, le paramètre n^+ est utilisé dans la formulation des cas $n_j \in V_i^{bad}(I,t)$ et $n_j = n_i$:

$$P_{i\to j}^{send}(I,t) = \begin{cases} \eta \dfrac{s(\tau_{i\leftarrow j}(t), \tau(I))}{\sum\limits_{z \in V_i^{good}(I,t)} s(\tau_{i\leftarrow z}(t), \tau(I))} & \text{si} \quad n_j \in V_i^{good}(I,t), \\ \left(1 - \eta\delta\left(|V_i^{good}(I,t)| > 0\right)\right) \dfrac{1}{|V_i^{bad}(I,t)| + 1 + n^+} & \text{si} \quad n_j \in V_i^{bad}(I,t) \\ \left(1 - \eta\delta\left(|V_i^{good}(I,t)| > 0\right)\right) \dfrac{1+n^+}{|V_i^{bad}(I,t)| + 1 + n^+} & \text{si} \quad i = j \end{cases} \tag{5.5}$$

Dans le cas où $V_i^{good}(I,t)$ est vide, aucun choix n'est nécessaire car il n'existe qu'un seul groupe. Ceci explique la présence du symbole de Kronecker $\delta(\cdot)$ qui vaut 1 quand l'expression qu'il contient est vérifiée et 0 dans les autres cas. Le cas inverse, où $V_i^{bad}(I,t)$ ne pose pas de problème puisque la destination de bouclage n_i est toujours présente. Dans le cas où $|V_i^{bad}(I,t)| = 0$, la direction n_i a une probabilité égale à $1-\eta$ d'être choisie.

Une fois ces probabilités calculées, une variable aléatoire est utilisée et l'une des destination est sélectionnée pour transmettre le paquet.

5.3.3 Carnet d'adresse : Mise à jour des informations

L'activité du carnet d'adresse consiste en une mise à jour des informations concernant les pairs connus. Cette mise à jour a un double objectif : actualiser les valeurs de recommandation concernant les pairs connus et découvrir de nouveaux pairs à qui se connecter. Un message va pour cela être envoyé à destination de tous les pairs connectés. Ce message ne contient aucune information particulière si ce n'est le nom de son émetteur. Les réponses à ces messages arriveront par la suite, de manière asynchrone.

Algorithme 2 : Activité du `AddressBook`

```
1 for Tout pair n_j ∈ V_j(t) do
2 |   /* Envoyer une demande de mise à jour du carnet */
3 |   sendABRequest(j)
4 end
```

Chaque pair recevant un message de mise à jour de carnet d'adresse répond en recommandant à son émetteur un des pairs de son voisinage. Plus précisément, considérons le cas d'un pair n_j reçevant à un instant t une demande émanant de n_i. Ce pair dispose d'un certain voisinage $V_j(t)$ parmi lequel il va choisir un pair à recommander à n_i. La figure 5.10 représente une vue partielle du réseau consideré.

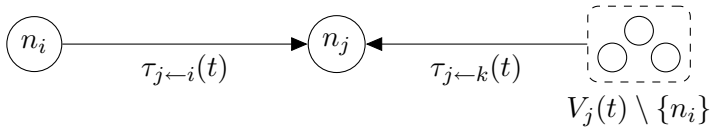


FIG. 5.10 – Variables entrant en jeu lors de la recommandation

Le choix est aléatoire et dépend d'un score de recommandation. Pour chacun des voisin, ce score est calculé comme étant la similarité entre la connexion. A un instant t, le score de la recommandation du pair n_j pour la connexion i,k est noté $R^j_{i\to k}(t)$. Le pair n_i ne pouvant évidemment pas être recommandé à lui même, le score de ce dernier n'est pas calculé.

$$R^j_{i\to k}(t) = sim(\tau_{j\leftarrow i}(t), \tau_{j\leftarrow k}(t)), \quad n_k \in V_j(t) \setminus \{n_i\} \tag{5.6}$$

Selon le score, chaque voisin à une probabilité plus ou moins importante d'être recommandé. Pour que le pair n_j recommande à n_i de se connecter au pair n_k, cette probabilité est notée $P^{recom}_{j\to i}(k,t)$. Elle est définie ainsi :

$$P^{recom}_{j\to i}(k,t) = \frac{R^j_{i\to k}(t)}{\sum_{n_l \in V_j(t)\setminus\{n_i\}} R^j_{i\to l}(t)}, \quad n_k \in V_j(t) \setminus \{n_i\} \tag{5.7}$$

Une fois que le pair n_j a effectué son choix, il compose un message de réponse qu'il envoie à n_i. Ce message contient les coordonnées du pair recommandé ainsi que le score associé à cette recommandation. Sur réception de ce message, n_i met à jour son carnet d'adresses. Si le pair indiqué n'était pas connu, il est préalablement ajouté au carnet. Puis le nouveau score

est enregistré. Il se peut que la nouvelle valeur soit inférieure à la précedente, auquel cas cela indiquera un changement des centres d'intêret soit de ce pair, soit de n_i tel que perçu par n_i.

5.3.4 Gestionnaire de connexions : Vérification des connexions

Les connexions du nid peuvent être ajustées afin de le déplacer vers un meilleur emplacement dans le réseau. Pour savoir si une connexion doit être coupée ou non, le pair évalue son utilité en se basant sur le contenu du compteur d'évaluation rempli par les fourmis. Le choix d'un nouveau pair parmi l'ensemble de ceux connus se base quand à lui sur le taux de recommandation qui en a été fait. Cette stratégie de choix se rapproche de ce qui est envisagé dans les réseaux de référents avec la notion de sociabilité [Yu 03]. La principale différence se situe au niveau de la signification de ce score. Alors que dans les réseaux de référents la sociabilité indique la capacité qu'a un pair à fournir de bonnes expertises, cet indicateur traduit ici la recommandation qui a été faite de ce pair par un autre membre du réseau.

Chaque pair a un maximum de V_{max} connexion autorisées. De plus, le nombre de ses connexion est également borné par un minimum V_{min}. Pour s'assurer que la valeur maximum est respectée, une nouvelle connexion peut être précedée de la suppression de la moins utile parmi celles déjà établies. Afin de garantir le respect du minimum V_{min}, toute déconnexion doit faire l'objet d'une demande au pair distant. Ce dernier accepte de couper la connexion si la taille de son voisinage est strictement supérieure au minimum autorisé. Nous nous plaçons ici dans un cadre idéal où les pairs ne tombent jamais en panne. Une déconnexion est donc nécessairement précédée de cette demande d'autorisation. Dans un environnement instable, une procédure de vérification périodique des connexions accompagnée d'une autre de réparation seraient nécessaire afin de garantir le respect des bornes concernant la taille du voisinage.

Algorithme 3 : Activité du `ConnectionsManager`

```
1  /* Calculer le scores des connexions */
2  calculerScores
3  pair ← GrabPeerFromAB
4  if pair ≠ −1 then
5  |  if |V_i(t)| = V_max then
6  |  |  /* Se déconnecter d'un des pairs */
7  |  end
8  |  if |V_i(t)| < V_max then
9  |  |  /* Essayer de se connecter à pair */
10 |  end
11 end
```

La fonction `ComputeScore` calcule le score d'utilité des connexions. Cette utilité est définie par le nombre de dernières bonnes évaluations que la connexion a reçu comparativement au nombre total d'évaluations prises en compte. Pour la connexion i, j, ce score est noté $U_{i\to j}(t)$.

$$U_{i\to j}(t) = \frac{nombre\ de\ 1\ dans\ E_{i\to j}(t)}{taille\ de\ E_{i\to j}(t)} \tag{5.8}$$

En partant de cette utilité, un score S^{drop} est attribué à chaque connexion.

$$S^{drop}_{i\to j}(t) = -\log\left(\frac{U_{i\to j}(t)+\varepsilon}{1+\varepsilon}\right), \quad \forall n_j \in V_i(t) \tag{5.9}$$

Avec ε un facteur non nul proche de 0. Etant donné que $S^{drop} \in [0, -\log\left(\frac{\varepsilon}{\varepsilon+1}\right)]$, ce paramètre ε définit la valeur maximum des scores obtenus. Pour l'ensemble des simulations, nous avons fixé $\varepsilon = 0.01$. Des tests avec différentes valeurs ont montré que l'influence de ce paramètre n'était que très limitée.

Un autre paramètre β permet de fixer le niveau d'utilité minimum d'une connexion. Par exemple, en fixant $\beta = 0.1$, toutes les connexions telles que $S^{drop}_{i\to j}(t) \leq 0.1$ seront susceptibles d'être supprimées. Si l'on interprete cette valeur, cela revient à dire qu'une connexion n'ayant pas reçu plus de 90 % d'évaluations positives est considerée comme étant inutile.

Pour chaque connexion i, j du voisinage, une probabilité de déconnexion $P^{drop}_{i\to j}(t)$ est calculée en fonction du score et de l'utilité attribuée au voisin :

$$P^{drop}_{i\to j}(t) = \begin{cases} \dfrac{S^{drop}_{i\to j}(t)}{\sum\limits_{n_k \in V'_i(t)} S^{drop}_{i\to k}(t)} & \text{if } U_{i\to j}(t) \leq \beta \\ 0 & \text{if } U_{i\to j}(t) > \beta \end{cases} \tag{5.10}$$

Avec $V'_i(t) = \{n_j \in V_i(t) \mid U_{i\to j}(t) \leq \beta\}$ l'ensemble des pairs du voisinages idenditifiés comme étant inutilement connectés.

5.4 Etude des paramètres des algorithmes

Chaque élément dispose de son jeu de paramètres, nous en dressons, ci-dessous, un bref rappel en les groupant selon la partie du système concerné :

- Périodes d'activités
  - T_{mode} : Echelle de temps
  - γ : Facteur d'étalement de la loi triangle
  - k^a : Période relative de l'activité du carnet d'adresse
  - k^c : Période relative de l'activité du gestionnaire de connexions
- Gestionnaire de connexions
  - V_{min} : Taille minimum du voisinage
  - V_{max} : Taille maximum du voisinage
  - ε : Régulation du calcul des scores
  - β : Utilité minimum d'une connexion
  - $|E|$: Taille de la mémoire pour les évaluations
- Dépôt des phéromones
  - α : Réglage de la rapidité de décroissement
  - ρ_{max} : Taux de dépôt maximum

  - TTL : Durée de vie d'une information
- Fourmis artificielles
  - Le nombre de fourmis
  - η : Coefficient de libre arbitre
  - s_{min} : Intérêt minimum d'une connexion
  - n^+ : Tendance casanière
- Bot
  - Nombre de copies crées pour chaque information
- Informations
  - Taille de la mémoire des pairs visités

Cela représente un total de 18 paramètres. Certains d'entre eux, tels que le ε, ont une influence négligeable. D'autre sont imposés par des contraintes extérieures et ne peuvent donc pas être ajustées. Afin de réduire le nombre de paramètres restants à régler, nous proposons donc une étude portant sur l'encadrement d'un certain nombre de variables aléatoires afin de mettre en évidence des relations entre les différents paramètres.

Pour l'ensemble des calculs qui suivent, et afin de ne pas nous placer dans un cas particulier, nous considérerons que

$$\begin{cases} |V_i^{good}(I,t)| > 0 \\ |V_i^{bad}(I,t)| > 0 \end{cases}$$

Etant donné que $V_{min} \leq |V_i^{good}(I,t)| + |V_i^{bad}(I,t)| \leq V_{max}$, il est possible de déduire un encadrement de $|V_i^{bad}(I,t)|$ et $|V_i^{good}(I,t)|$:

$$\begin{cases} 1 \leq |V_i^{good}(I,t)| \leq V_{max} - 1 \\ 1 \leq |V_i^{bad}(I,t)| \leq V_{max} - 1 \end{cases} \tag{5.11}$$

Ces équations imposent logiquement que $V_{min} \geq 2$.

La probabilité pour une fourmi f de décider de rester sur place k fois consécutives se définit comme les produit des probabilités de faire ce choix à chacune des k actions. Le moment où la fourmi f effectue sa première action est noté $t_0(f)$. Les autres instants où la fourmi agit peuvent être calculés en ajoutant à ce temps initial le délai entre deux actions $T(f,n)$: la fourmi fera son second choix au temps $t_0(f) + T(f,1)$, puis le troisième à $t_0(f) + T(f,1) + T(f,2)$ et ainsi que suite jusqu'au choix n^* au temps $t_0(f) + \sum_{n=1}^{n^*(f)} T(f,n)$. En supposant que ceci se produit entre les actions a et b ($b - a + 1 = k$), on obtient ξ_k

$$\xi_k = \prod_{n=a}^{b} (1 - \eta\delta(|V_i^{good}(I,t)| > 0)) \frac{1 + n^+}{|V_i^{bad}(I, t_0(f) + \sum_{l=1}^{n} T(f,l))| + 1 + n^+} \tag{5.12}$$

Chaque pair maintient le nombre de ses voisins entre deux valeurs V_{min} et V_{max}. Ceci nous permet de définir un intervalle de valeurs pour l'ensemble des voisins non interessés, l'ajout de la valeur 1 permettant de prendre en compte la connexion de bouclage :

$$1 \leq |V_i^{bad}(I, t_0(f) + \sum_{l=1}^{n} T(f,l))| \leq V_{max} - 1$$

soit également

$$2 + n^+ \leq |V_i^{bad}(I, t_0(f) + \sum_{l=1}^{n} T(f,l))| + 1 + n^+ \leq V_{max} + n^+$$

et donc

$$\frac{1+n^+}{V_{max}+n^+} \leq \frac{1+n^+}{|V_i^{bad}(I, t_0(f) + \sum_{l=1}^{n} T(f,l))| + 1 + n^+} \leq \frac{1+n^+}{2+n^+}$$

et, étant donné que $1 - \eta\delta(|V_i^{good}(I,t)| > 0) \in \{1-\eta, 1\}$, nous obtenons finalement l'inégalité suivante :

$$\left(\frac{(1-\eta)(1+n^+)}{V_{max}+n^+}\right)^k \leq \xi_k \leq \left(\frac{1+n^+}{2+n^+}\right)^k \tag{5.13}$$

L'efficacité de la diffusion des informations est dictée principalement par la loi de transmission $P_{i\to j}^{send}(I,t)$. Si celle-ci est proche de 0, aucun transfert ne sera effectué alors qu'à l'inverse, pour une valeur de 1, tous les pairs seront systématiquement inondés (« *floodés* »). Les paramètres entrant en jeu dans le calcul de cette probabilité sont $s_{min}, n^+, V_{min}, V_{max}$ and η. Les quelques équations qui suivent permettent la mise en évidence d'un ensemble de relations existant entre elles. Trois cas existent pour le calcul de $P_{i\to j}^{send}(I,t)$:

- Si $n_j \in V_i^{good}(I,t)$ nous avons

$$P_{i\to j}^{send}(I,t)\Big|_{n_j \in V_i^{good}(I,t)} = \eta \frac{s(\tau_{i\leftarrow j}(t), \tau(I))}{\sum\limits_{z \in V_i^{good}(I,t)} s(\tau_{i\leftarrow z}(t), \tau(I))}$$

  or, étant donné que $n_j \in V_i^{good}(I,t)$, nous savons que $s_{min} \leq s(\tau_{i\leftarrow j}(t), \tau(I)) \leq 1$. Donc :

$$\eta \frac{s_{min}}{|V_i^{good}(I,t)|} \leq P_{i\to j}^{send}(I,t)\Big|_{n_j \in V_i^{good}(I,t)} \leq \eta \frac{1}{|V_i^{good}(I,t)| s_{min}}$$

  enfin, puisque $2 \leq |V_i^{good}(I,t)| + |V_i^{bad}(I,t)| \leq V_{max}$ nous déduisons que $1 \leq |V_i^{good}(I,t)| \leq V_{max} - 1$ et aboutissons à l'encadrement suivant :

$$\eta \frac{s_{min}}{V_{max}-1} \leq P_{i\to j}^{send}(I,t)\Big|_{n_j \in V_i^{good}(I,t)} \leq \eta \frac{1}{s_{min}} \tag{5.14}$$

- Si $n_j \in V_i^{bad}(I,t)$ nous avons

$$P_{i\to j}^{send}(I,t)\Big|_{n_j \in V_i^{bad}(I,t)} = (1-\eta)\frac{1}{|V_i^{bad}(I,t)| + 1 + n^+}$$

  du fait que $1 \leq |V_i^{bad}(I,t)| \leq V_{max} - 1$, nous établissons que :

$$(1-\eta)\frac{1}{V_{max}+n^+} \leq P_{i\to j}^{send}(I,t)\Big|_{n_j \in V_i^{bad}(I,t)} \leq (1-\eta)\frac{1}{2+n^+} \tag{5.15}$$

  ce qui nous donne l'encadrement pour le second cas de figure.

- Finalement, pour le cas où $n_i = n_j$, un raisonnement similaire permet d'établir que :

$$(1-\eta)\,\frac{1+n^+}{V_{max}+n^+} \leq P^{send}_{i\to i}(I,t) \leq (1-\eta)\,\frac{1+n^+}{2+n^+} \tag{5.16}$$

L'inégalité 5.14 est définie par l'ensemble de valeurs $\{s_{min}, \eta, V_{max}\}$. Celles d'indices 5.15 et 5.16 peuvent être calculées avec l'ensemble de paramètres $\{, n^+, \eta, V_{max}\}$. L'équation 5.13 peut elle aussi être définie en utilisant ce même ensemble de paramètres et le nombre de bouclages k que l'on désire évalué. Les valeurs des paramètres V_{min} et V_{max} dépendent de contraintes extérieures (limitation de bande passante, préférence de l'utilisateur, ...). Nous avons vu qu'il fallait que V_{min} soit supérieur à 2 pour que les équations puissent être définies. V_{max} sera supposé ayant une valeur constante, fixée au démarrage de PIAF et supérieure à $V_{min} = 2$.

En retirant V_{max} des ensembles de paramètres de PIAF à ajuster, il nous apparaît que les équations 5.13, 5.14, 5.15 et 5.16 peuvent être totalement définies en choissant les valeurs de s_{min}, n^+ et η et ceux soit en considérant la paire $\{n^+, \eta\}$ ou $\{s_{min}, \eta\}$. Ces paramètres ont une influence directe sur le taux de diffusion des informations qui sera pratiqué. Les inégalitées établies nous permettent de conclure sur le fait que η sera une variable ayant une influence globale aussi bien dans le cas de la diffusion vers des pairs potentiellement intéressés que dans le cas contraire. Une fois celui-ci fixé, le choix d'une valeur de s_{min} et de n^+ permettra respectivement d'ajuster les envois vers l'une ou l'autre de ces catégories de pairs.

5.5 Conclusion

Nous avons ici présenté la version « finale » de l'algorithme de diffusion d'information et de modification du réseau utilisé dans PIAF. Cette présentation fût suivie de tests vérifiant les performances de l'outil dans le cadre d'application pour lequel il était conçu, celui de la circulation d'informations entre pairs. Durant la conception de PIAF, d'autres modélisation du système ont vu le jour et, en particulier, les deux variantes suivantes ont été étudiées :

- **Fourmis mémorisant leurs choix**

  Cette première variante supprime la mémoire associée aux informations qui circulent dans le réseau. En remplacement, et afin d'éviter les envois multiples, chaque fourmi mémorise les derniers pairs auquels elle a envoyé une informations. Egalement, dans ce modèle, les fourmis dupliquent l'information envoyée. Au lieu de transferer la version qu'elles ont en charge, elles en font des duplicatas qui sont émis vers les destinataires choisis.

  Chaque fourmi dispose d'une zone de mémoire. Lorsque l'une d'elle prend en charge une information à diffuser, cette mémoire est vidée des données qu'elle pouvait précédement contenir. Elle est par la suite utilisée d'un façon similaire à la mémoire associée aux informations telle que présentée aux sections précédentes.

- **Evaluation personnelle des connexions**

  Contrairement au modèle actuel où le gestionnaire de connexion ne se base que sur les scores de recommandations formulés par les voisins afin de faire son choix parmi les connexions possible à établir.

Une autre version de l'algorithme se propose de donner au gestionnaire de connexions le choix d'évaluer lui-même les connexions. Pour ce faire, les évaluations sont utilisée à la place du score de recommandation.

Nous terminerons ce chapitre sur un tableau récapitulatif des variables les plus importantes pour le fonctionnement de PIAF.

Notation	**Définition**
s_{min}	Seuil minimum de similarité entre les traces de phéromones pour considérer qu'un pair est intéressé
n^+	Importance de la possibilité de rester au nid dans le choix d'une destination
η	Coefficient de libre arbitre

TAB. 5.1 – Paramètres régissant le comportement des fourmis

Notation	**Définition**
I	Une information
$\tau(I)$	Trace de phéromones associée à une information I
$round(I)$	Nombre de déplacements (round) depuis la création de l'information I
$vus(I,t)$	Liste des derniers pairs visités par I jusqu'à l'instant t

TAB. 5.2 – Notations relatives à la définition des informations

Notation	**Définition**
$\tau_{i\to j}(t)$	Trace de phéromones associée à une connexion de n_i vers n_j. On remarquera que les deux notations $\tau_{i\to j}(t)$ et $\tau_{j\leftarrow i}(t)$ sont équivalentes
I	Une information
$V_i(t)$	L'ensemble des pairs auquels n_i est connecté à l'instant t (*i.e.* son voisinage)
$\underline{V_i(I,t)}$	Sous ensemble de $V_i(t)$ interessé par une information I
$\overline{V_i(I,t)}$	Sous ensemble de $V_i(t)$ non-interessé par une information I
$\tau(I)$	Trace de phéromones associée à une information I
$round(I)$	Nombre de déplacements (round) depuis la création de l'information I

TAB. 5.3 – Notations relatives à la définition du réseau

Probabilité	**Description**
$P^{send}_{i\to j}(I,t)$	Probabilité de transferer I du pair n_i au pair n_j à l'instant t
$P^{drop}_{i\to j}(t)$	Probabilité de couper la connexion du pair n_i au pair n_j à l'instant t
$P^{recom}_{i\to j}(k,t)$	Probabilité pour qu'à l'instant t, n_i recommande à n_j la connexion à n_k

TAB. 5.4 – Définitions des probabilités relatives aux connexions

Le chapitre qui suit est consacré à la validation de cette architecture par l'intermédiaire d'une série de tests expérimentaux.

Chapitre 6

Etude expérimentale de la plateforme PIAF

Ce chapitre est consacré à l'étude des performances de la plateforme PIAF. Les sujets qui y sont successivement abordés sont l'implémentation des algorithmes présentés au chapitre précédent, la création des jeux de données ainsi que les simulations effectuées.

6.1 Introduction

L'étude expérimentale présentée ici composée de deux parties. La première concerne la recherche de valeurs optimales pour les paramètres de PIAF. Utilisant ces valeurs, la seconde partie est consacrée à une comparaison de performances par rapport à une diffusion aléatoire.

La notion d'optimalités est ici à relativiser par rapport au conditions de tests. En effet, les nombreux paramètres de l'algorithme ainsi que l'influence forte de l'aléatoire (fréquence d'activation des différents composants) ne permettent pas de statuer sur des valeurs optimales dans tous les cas de figure.

6.2 Implémentation de PIAF

Nous commencerons ce chapitre par quelques mots concernant l'implémentation de PIAF. Durant les deux premières années de travaux, les efforts ont eu pour objectif la réalisation d'un « produit fini » avec l'idée de pouvoir effectuer des tests d'utilité de la plateforme auprès d'utilisateurs. En partie à cause des choix techniques qui ont été faits à ce moment là, cette première implémentation a été mise de côté au profit d'un autre utilisant un simulateur d'événements discrets.

6.2.1 Réalisation d'un environnement destiné à la « production »

Cette première implémentation a été faite avec le « Practical Extraction and Report Language » (PERL)[1]. Il s'agit d'un langage de programmation semi-interprété bien connu des administrateurs de systèmes car habituellement rencontré dans le cadre de la réalisation de scripts pour l'administration de serveurs. Les atouts de PERL sont sa facultée à traiter des chaînes de caractères (ce fût d'ailleurs pour cela qu'il a été inventé) ainsi que sa légereté et sa portabilité puisqu'il suffit de disposer d'un intérpréteur pour pouvoir exécuter les scripts.

L'objectif de la réalisation était la mise en place d'un environnement dit de « production », pouvant être déployé et testé auprès des utilisateurs potentiels. Ce choix a été fait contre la logique habituelle de développement qui veut que les étapes de la réalisation soient, dans cet ordre, la conception, la simulation et les tests puis l'implémentation finale et la distribution auprès des utilisateurs. La raison en est la nature même du produit développé et de son critère de qualité associé. L'objectif de PIAF est de fournir un support facilitant le travail des utilisateurs en s'occupant à leur place des tâches de partage des informations et de la mise en place d'un réseau social basé sur leurs centres d'intérêts. Le critère de qualité d'un tel outil est la satisfaction que l'utilisateur vis à vis des résultats obtenus, au regard de l'effort qu'il devra fournir pour utiliser l'outil et des ressources machine qui auront été utilisées. L'évaluation d'un tel critère ne peut se faire que par des tests en situation, en utilisant un produit finalisé.

Nous avons ainsi opté pour un développement enchaînant conception et implémentation finale puis tests et validations. Mais cette stratégie ne s'est pas averée satisfaisante, nous reviendrons sur ce point dans la partie bilan de cette section.

Quelques points caractérisant cette implémentation sont ci-après détaillés

- **Implémentation des éléments constitutifs de PIAF**
  Les éléments constitutifs de PIAF ont été implémentés sous la forme de composants en tirant profit des outils de conception fournis par POE (Perl Object Environment). POE est une plateforme permettant la création en PERL de programmes multi-tâches pilotés par un système événementiel.
  Deux composants supplémentaires ont été ajoutés à ceux décrit au chapitre précédent. Le premier est un serveur écoutant les requêtes provenant de l'interface graphique de contrôle. Il permet à cette interface de commander diverses opérations telles que le redémarrage ou l'arrêt du programme. Le second est un client se connectant à l'interface de contrôle de l'état du réseau. Celui-ci fournit à l'interface les données dont elle a besoin selon une stratégie de communication de type « push » : périodiquement, le pair génère un paquet contenant toutes les données utiles et l'envoie en direction du serveur.
- **Création d'une interface graphique de contrôle du système**
  Une première interface a été réalisée afin de contrôler l'activité des PIAFs (voir figure 6.1). Celle-ci est capable de se connecter à un certain nombre d'instances de la plateforme et de récupérer des données concernant l'état des pairs. Les fonctionnalités fournies par l'application sont la visualisation de la topologie du réseau, la consultation des stocks d'informations dont disposent les pairs ainsi que leur voisinage. Le sourire qu'arborent les visages situés à la place des nœuds traduisent visuellement la valeur du critère de satisfaction du pair. Plus cette satisfaction est grande (proche de 1), plus le visage représenté est souriant.

[1] `www.perl.org`, dernier accès le 10 octobre 2006

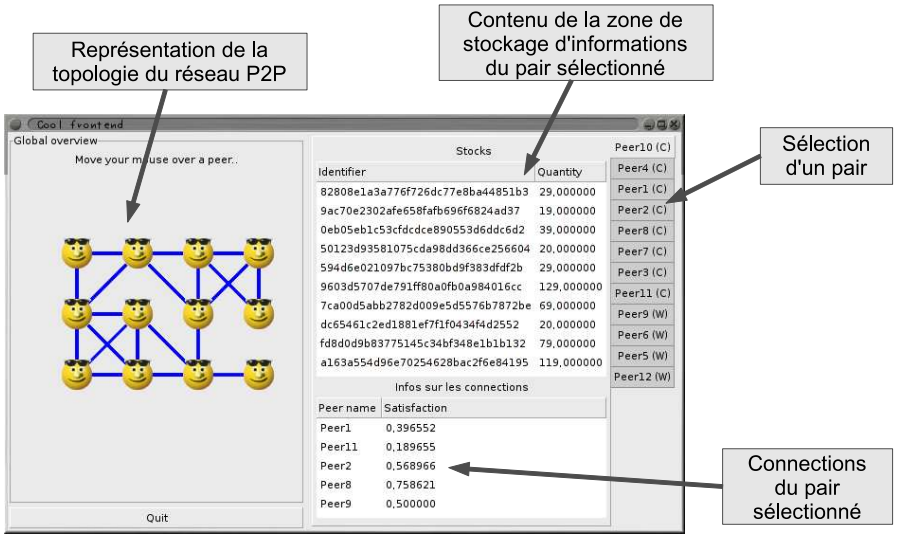


FIG. 6.1 – Interface graphique contrôlant l'activité de PIAF dans sa version en PERL

La seconde interface est celle jouant le rôle de télécommande à distance des PIAFs (voir figure 6.2). Ses fonctionnalités se résument à l'exécution de commandes et l'affichage de messages envoyés par les pairs. C'est cette même interface qui est également utilisée pour la création d'un réseau virtuel composé de plusieurs nœuds. Ces derniers sont alors créés sur une même machine et utilisent différents ports de communication de la carte réseau afin de pouvoir fonctionner tous indépendamment sans se gêner les uns les autres.

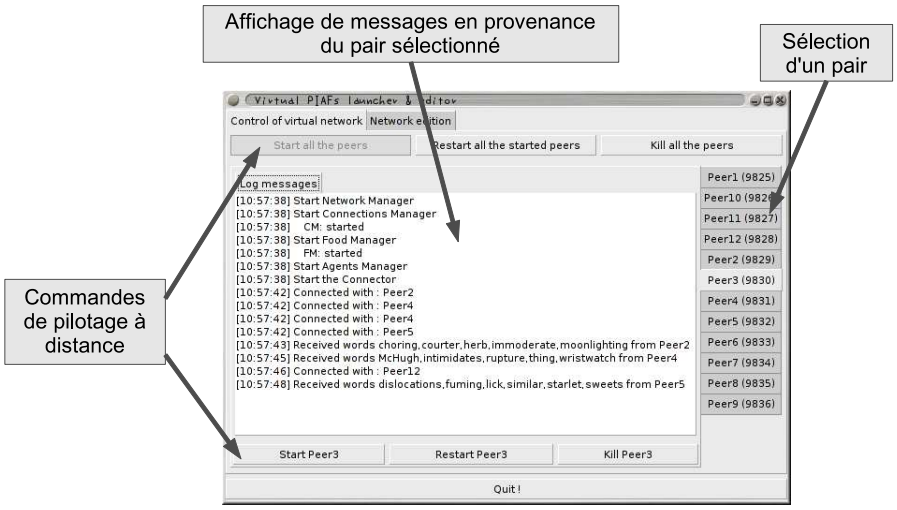


FIG. 6.2 – Interface graphique permettant de piloter à distance un ou plusieurs PIAFs

Ces deux programmes sont eux aussi écrits en PERL et utilisent l'ensemble de composants graphiques Gtk+ par l'intermédiaire du module `Gtk2`[2].

- **Création d'un greffon pour le navigateur internet « Firefox »** Firefox est un projet

[2] `gtk2-perl.sourceforge.net`, dernier accès le 23 octobre 2006

de navigateur libre de la fondation Mozilla[3]. Nous avons développé pour ce navigateur un greffon effectuant la récupération d'information et assurant le dialogue avec les éléments de circulation de l'information de PIAF. Ce greffon est un exemple des applications se situant au « niveau interface » de PIAF.

Cette implémentation, bien que fonctionnelle, ne pu être utilisée efficacement afin d'effectuer des séries de tests des algorithmes. Le problème se situait essentiellement au niveau de la conception du programme. Etant initialement prévu pour un cadre de production, tous les pairs étaient exécutés par des processus séparés utilisant des ressources réseau (locales) pour communiquer entre eux. Les capacités réseau ont été un facteur limitant mais surtout la remontée d'informations stastiques sur l'efficacité du système s'est averée particulièrement fastidieuse et coûteuse. Périodiquement, chaque pair devait constituer un message contenant les informations nécessaires aux calculs et l'envoyer à un serveur effectuant l'agrégation des résultats. La création et la circulation de ces messages été une activité consommatrice de ressources CPU et réseau. A mi-chemin de nos travaux, celle-ci a donc été mise de côté afin de passer à une seconde implémentation présentée ci-après.

Entre la seconde année marquant l'arrêt du développement de cette implémentation et la quatrième marquant la fin de cette thèse, les algorithmes au cœur de PIAF ont évolués et se sont améliorés. Afin de mettre en production PIAF, une actualisation du programme serait donc nécessaire mais la plus grosse partie du développement est déjà réalisée.

6.2.2 Nouvelle implémentation à l'aide d'un outil de simulation

L'ensemble de la modélisation a été implémentée sur un simulateur d'événements discrets. Le simulateur est OmnetPP [Varga 02]. L'interface graphique de ce simulateur est visible sur la figure 6.3.

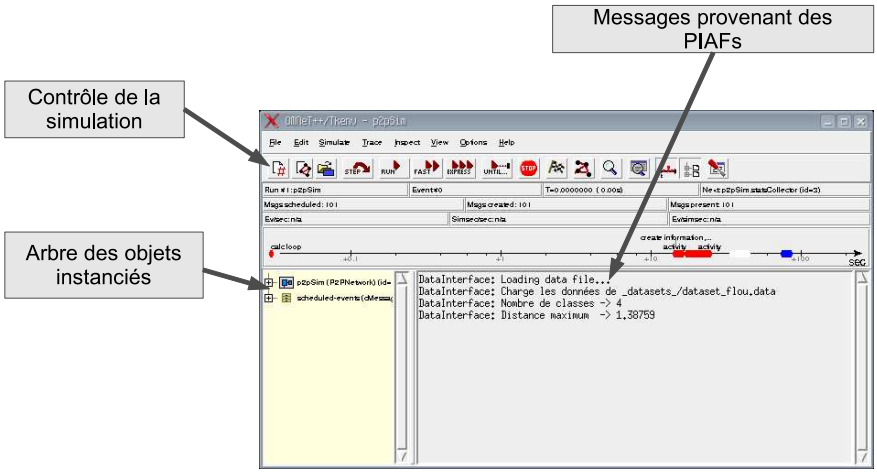


FIG. 6.3 – Interface graphique du simulateur

[3] `www.mozilla.org`, dernier accès le 7 octobre 2006

Les quelques points suivants fournissent des informations sur cette implémentation selon une approche similaire à la précédente :

- **Implémentation des éléments constitutifs de PIAF** Une représentation graphique des éléments implémentés dans le simulateur est visible sur la figure 6.4. On y remarque la présence d'un robot sur lequel nous reviendrons par la suite. Par soucis de rapidité dans la circulation interne des messages, quelques aménagements ont été faits par rapport à la définition initiale de PIAF au chapitre précédent. En particulier, la connexion de tous les éléments au noyau permet de réduire la distance séparant chaque élément fonctionnel (et donc le temps de communication lorsqu'il leur s'agit d'échanger des messages).

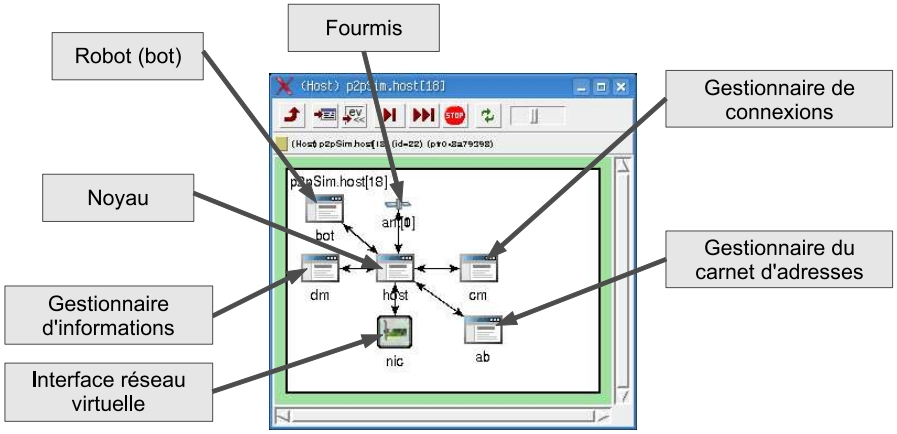


FIG. 6.4 – Représentation des éléments de PIAF implémentés dans le simulateur

- **Création d'un agent robot (« bot ») pour l'injection d'informations** La première implémentation effectuée de PIAF avait pour objectif la mise en test avec des utilisateurs. Dans le cadre d'une simulation, ces utilisateurs ne sont plus disponibles et doivent être remplacés. Ce sont donc des robot (ou « bot ») qui effectuent de l'injection d'informations dans le système afin de simuler la présence d'un utilisateur qui générerait des informations de part son activité. Chaque pair contient un bot dont la fréquence de fonctionnement est identique à celle des fourmis.
- **Création d'éléments de contrôle**

  Les informations publiées sont extraites d'un jeu de données chargé globalement par le simulateur. C'est un agent externe aux PIAFs qui gère ce jeu de données.Celui-ci implémente deux fonctions `PickRandomInformation` et `PickCloseInformation` que les bot appellent afin de récupérer une information à publier. La première de ces fonctions permet de choisir un élément au hasard parmi l'ensemble de ceux disponibles alors que la seconde retourne l'élément le plus proche de celui passé en paramètre. Une fois qu'un élément du jeu de donnée à été donné à un bot, celui-ci est retiré. Une fois que le jeu de données est vide, celui-ci est re-rempli avec l'ensemble des données de départ.

  Un second élément de contrôle effectue l'ensemble des mesures statistiques nécessaires à l'évaluation des performances du système.

  Le réseau des PIAFs ainsi que ces deux éléments externes sont visibles sur la figure 6.5

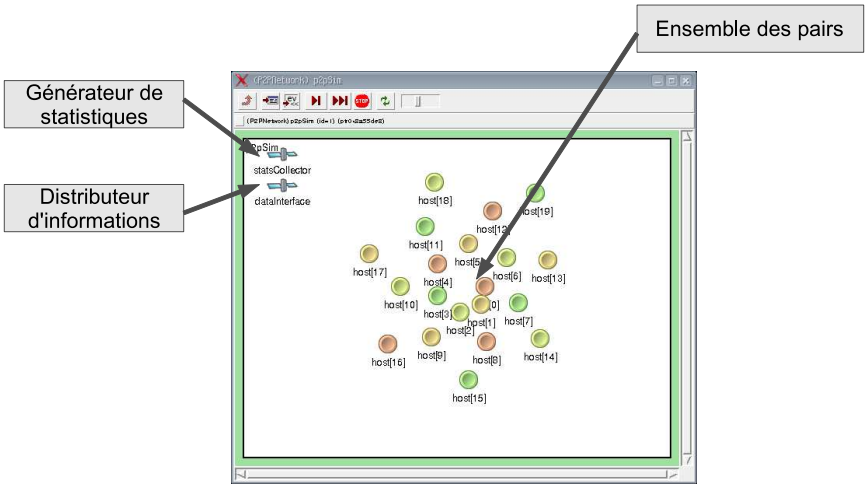


FIG. 6.5 – Représentation d'un des réseaux de test de PIAF ainsi que des éléments externes de contrôle

C'est cette seconde implémentation qui nous a permi d'effectuer l'ensemble des tests présentés ci-après.

6.3 Jeu de données mis en œuvre

Afin d'effectuer les tests, deux jeux de données sont nécessaires. Il est nécessaire de disposer d'une base de données d'informations à échanger ainsi que d'une topologie initiale du réseau permettant de remplir le carnet d'adresse des pairs au démarrage du système.

6.3.1 Base de données d'informations

Le *Bot* est configuré de façon à simuler la présence d'un utilisateur. Son activité se résume à produire une série d'informations correspondant à un thème particulier. Selon la modélisation de PIAF, ces informations doivent être disponibles sous la forme d'un vecteur de valeurs réelles. Celles-ci seront ici supposées être le fruit d'une modélisation VSM de documents, chaque valeur représentant la pondération d'un mot clé extrait du contenu associé à l'information (*cf* chapitre 3.2 sur la modélisation des informations). Nous supposerons également la définition d'un dictionnaire commun à l'ensemble du réseau permettant de fixer la taille des vecteurs. Les jeux de données sont générés artificiellement selon l'algorithme indiqué en annexes.

En utilisant un ensemble de valeurs différentes pour chaque paramètre de cet algorithme, nous produisons 3 jeux de données : `FuzzyDataset`, `NoisyDataset`, `SeparatedDataset`. Ceux-ci correspondent respectivement à un niveau de bruit égal à l'information utile, la moitié de l'information utile et un tiers de l'information utile. Les valeurs de ces paramètres sont reportées dans le tableau 6.1. Les jeux de données artificiels utilisés sont composés, pour chaque domaine, d'une centaine d'informations de taille 50. Cette taille représente la dimension de l'espace vectoriel

de représentation des phéromones $\mathbb{R}^n$. Elle pourrait par exemple correspondre à un dictionnaire de 100 mots utilisés pour une modélisation de type VSM de contenu textuels en supposant que les différentes valeurs de $\tau(I)$ correspondent à des fréquences d'apparition de mots.

Jeu de données	Themes	Documents	TailleDico	Mots	FreqMot	Bruit	FreqBruit
`FuzzyDataset`	4	100	50	10	15	10	15
`NoisyDataset`	4	100	50	10	15	5	6
`SeparatedDataset`	4	100	50	10	15	3	5

TAB. 6.1 – Paramétrage de l'algorithme de génération de données

Les propriétés concernant la similarité entre les différentes informations sont présentées dans la table 6.6, ces résultats sont obtenus en appliquant la formule 6.1.

$$Sim(D_x, D_y) = \frac{1}{|D_x|} \sum_{I \in D_x} \left(\frac{\sum_{I' \in D_y \setminus \{I\}} s(\tau(I), \tau(I'))}{|D_y \setminus \{I\}|} \right) \tag{6.1}$$

Pour chaque case, les valeurs indiquent la similarité moyenne entre un membre de la classe en abscisse et tous les autres membres de la classe en ordonnées. Le jeu de données est volontairement simpliste afin de pouvoir plus facilement étudier le comportement de l'algorithme et établir des conclusions sur les résultats obtenus.

	D_1	D_2	D_3	D_4
D_1	0.77	0.37	0.22	0.22
D_2	0.37	0.78	0.28	0.22
D_3	0.22	0.28	0.78	0.23
D_4	0.22	0.22	0.23	0.78

(a) `FuzzyDataset`

	D_1	D_2	D_3	D_4
D_1	0.77	0.16	0.03	0.15
D_2	0.16	0.78	0.10	0.07
D_3	0.03	0.10	0.77	0.05
D_4	0.15	0.07	0.05	0.79

(b) `NoisyDataset`

	D_1	D_2	D_3	D_4
D_1	0.78	0.03	0.03	0.06
D_2	0.03	0.79	0.04	0.06
D_3	0.03	0.04	0.79	0.00
D_4	0.06	0.06	0.00	0.78

(c) `SeparatedDataset`

FIG. 6.6 – Jeu de données

6.3.2 Topologie initiale

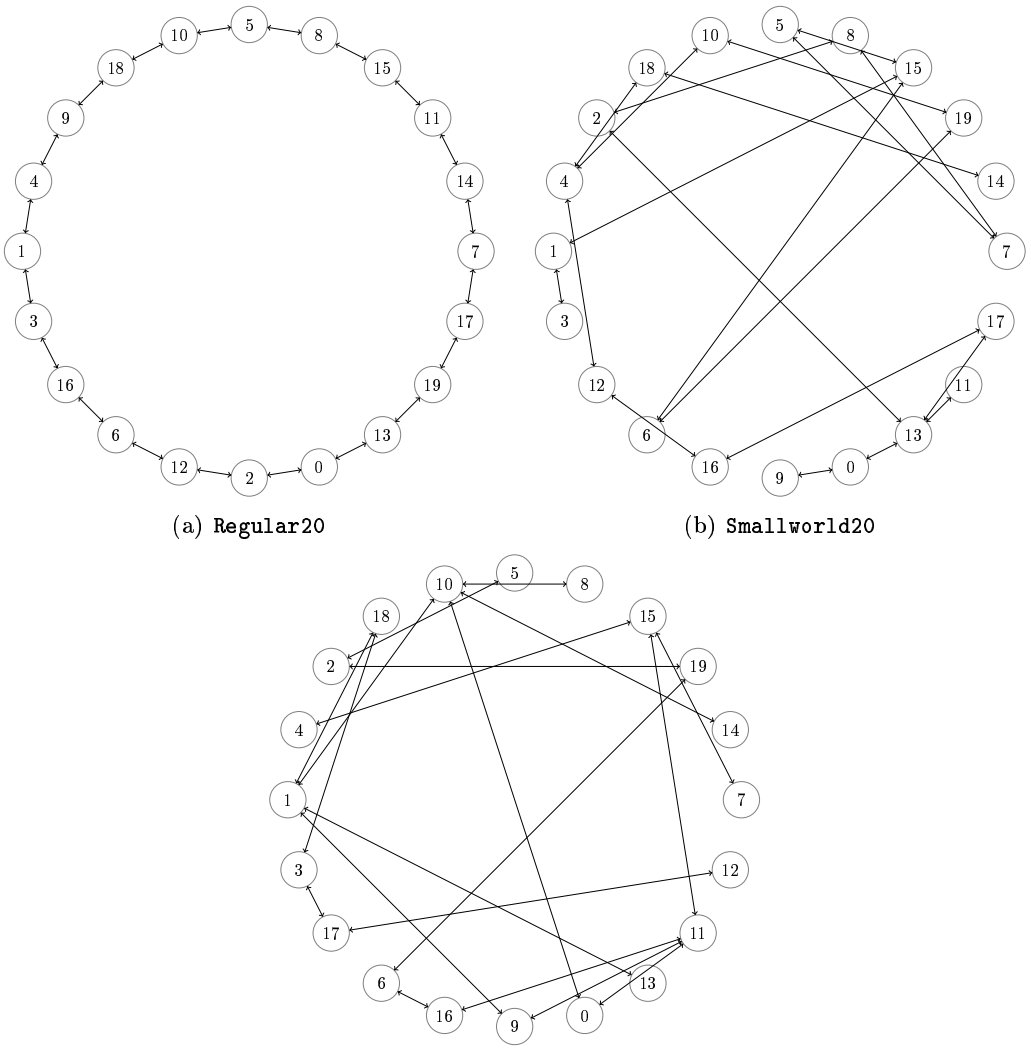
La topologie initiale du réseau est utilisée afin d'initialiser les carnets d'adresses des pairs. Au démarrage de PIAF, tous les pairs sont déconnectés. Cette topologie ne définie donc pas un schéma de connexion initial mais un schéma probable. L'algorithme de remplissage du carnet d'adresse est disponible en annexes de ce document. Il s'agit de l'algorithme β de construction de réseaux petit monde [Watts 98].

Par niveau de structuration croissant, nous considérons 3 topologies initiales : aléatoire, petit monde et circulaire. Celles-ci ont obtenues en ajustant le paramètre p de l'algorithme de génération (voir tableau 6.2).

Réseaux	Pairs	Addresses	Interets	p
`Regular20`	20	2	4	0
`Smallworld20`	20	2	4	0.3
`Random20`	20	2	4	1
`Regular40`	40	2	4	0
`Smallworld40`	40	2	4	0.3
`Random40`	40	2	4	1

TAB. 6.2 – Paramétrage de l'algorithme de génération de réseaux

Dans les réseaux de 20 machines, ce sont 5 pairs qui seront intéressés par un même sujet.

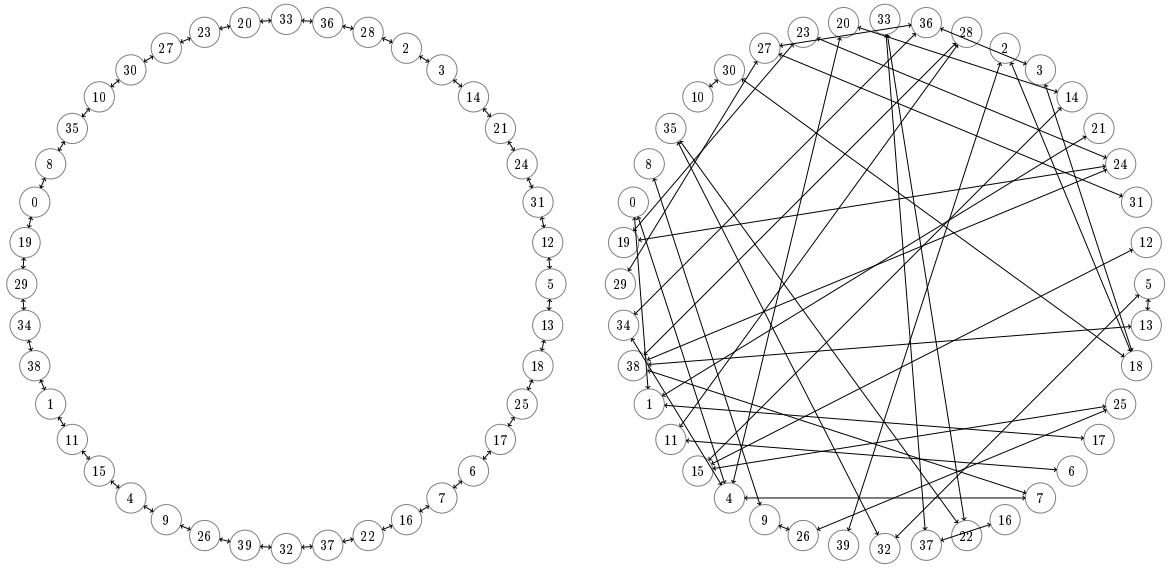


(a) `Regular20` (b) `Smallworld20`

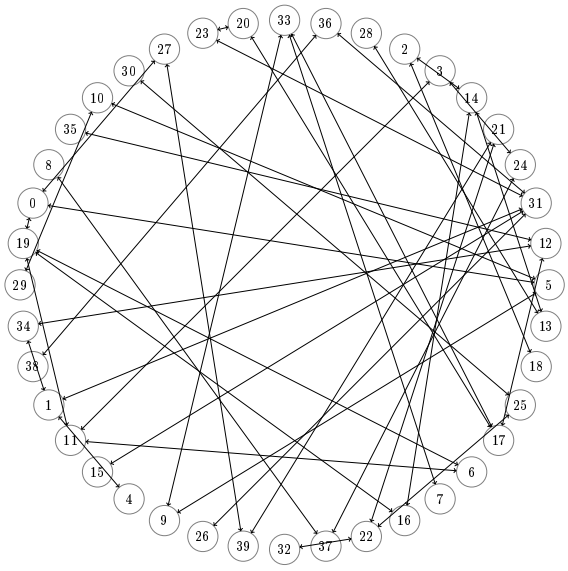
(c) `Random20`

FIG. 6.7 – Représentation du contenu initial du carnet d'adresses - réseau 20

Pour les réseaux à 40 pairs, ce nombre sera de 10.



(a) `Regular40` (b) `Smallworld40`



(c) `Random40`

FIG. 6.8 – Représentation du contenu initial du carnet d'adresses - réseau 40

6.4 Etude des paramètres de l'algorithme

PIAF comporte un bon nombre de paramètres. Un certain nombre d'entre eux ont été fixés arbitrairement : nous avons fixé $\rho_{min} = 0.05$, $\rho_{max} = 0.8$ pour respectivement s'assurer que chaque information laisse une trace minimale après avoir été transferée et pour éviter de risquer de perdre les données accumulées associée aux connexions. Nous avons également fixé $TTL(I) = 5$ ce qui correspond à la durée de vie minimum pour qu'une information soit consultée par tous les pairs d'un groupement et $seen(I,t) = 1$ de sorte que seul le retour vers l'envoyeur soit interdit lors du choix des destinations d'envoi. Chaque série de résultats a été moyennée sur 20 exécutions de l'algorithme afin de réduire l'influence des paramètres aléatoires sur les performances observées.

Nous avons vu qu'une fois V_{max} fixé, les autres limites pour les paramètres sont définies par l'un des couples $\{\eta, n^+\}$ et $\{\eta, s_{min}\}$. Il est possible de chercher des valeurs optimales de ces paramètres en suivant une procédure progressive où η est celui qui sera fixé en dernier étant donné sa présence dans les deux paires considerées :

1. partant d'une valeur fixe pour η et n^+ , chercher une valeur optimale pour s_{min} ;
2. partant d'une valeur fixe pour η et s_{min}, chercher une valeur optimale pour n^+ ;
3. partant d'une valeur fixe pour n^+ et s_{min}, chercher une valeur optimale pour η.

Le critère étudié est celui des valeurs atteintes par les estimateurs quand plus aucune information ne peut être échangée entre les pairs. C'est à dire dès que les *bots* ont arrêté leur production et que les stocks d'informations à diffuser sont vides. Ces valeurs sont reportées dans divers tableaux où un code couleur permet d'identifier quel le meilleur résultat obtenu ainsi que celui est son premier suivant. Ces tableaux indiquent également quel est le paramètre considéré ainsi que la valeur d'autres paramètres si il y a lieu. La figure 6.9 contient un guide de lecture.

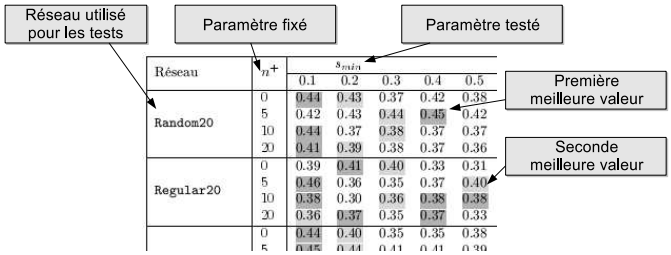


FIG. 6.9 – Indications de lecture pour les tableaux de résultats

6.4.1 Paramétrage de s_{min}

Afin d'établir une valeur optimale de s_{min}, cette première série de tests prend en considération $\eta = 0.8$ et un ensemble de valeurs pour n^+. Le premier de ces paramètre définit la préférence qu'à une fourmi à choisir entre partir en direction d'un pair potentiellement intéressé ou non. Le second permet d'augmenter les chances d'opter pour la solution revenant à ne pas quitter le nid dans le cas où le choix a été fait de se rendre à un nid non intéressé. La possibilité de choisir de rester dans le nid étant dépendante de celle du choix du groupe de pairs à considerer, ces deux paramètres sont liés. Avec un n^+ fixe, diminuer η permet d'augmenter la probabilité de faire un

tel choix et inversement. C'est pourquoi nous avons opté pour fixer l'un et effectuer des tests d'influence pour quelques valeurs de l'autre.

Par définition, $s_{min} \in [0, 1]$. Une discretisation de cet intervalle avec un pas de 0.1 ainsi que la suppression des valeurs extrêmes conduit à tester les valeurs $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$

Critère de groupement

Le taux de groupement obtenu est indiqué par la série de tableaux 6.3(a,b,c) pour le réseau de 20 pairs et par les tableaux 6.4(a,b,c) pour le réseau de 40 machines. Afin de mettre en valeur les résultats, un code composé de deux couleurs permet de repérer la plus forte valeur et sa première suivante dans chaque ligne des tableaux.

La comparaison des résultats entre les deux tailles de réseau puis entre chacune des trois topologies, amene à la formulation de deux remarques générales :

- Le passage d'un réseau de 20 à 40 machines n'affecte que peu les résultats obtenus. On observe cependant un certain « tassement » des valeurs optimales relevées. Ceci se remarque tout particulièrement en comparant les figures 6.2(a) et 6.3(a).
- Les valeurs finales obtenues vont croissantes avec l'augmentation du brassage initial des connexion. Ce sont en effet les réseaux `Random20` et `Random40` qui permettent généralement d'atteindre les meilleurs résultats alors que `Regular20` et `Regular40` en fournissent de moins bons.

En se focalisant sur ce premier aspect il est donc possible de conclure qu'en présence d'un réseau de taille importante, il est donc préférable de choisir une valeur faire de s_{min} afin de faciliter la diffusion des informations. Si l'on compare les résultats selon les différents jeux de données utilisés, il apparaît également que plus ces données sont floues, plus les valeurs optimales se rapprochent de $s_{min} = 0.1$.

Ces premiers tableaux montrent donc qu'en présence d'un réseau de taille élévée dans lequel les utilisateurs font circuler des informations ayant peu de différences, une valeur faible de s_{min} permettra d'obtenir les meilleurs résultats. Plus les pairs recevront d'informations, plus il leur sera facile de constituer des groupes.

6.4.1.1 Critère de rappel

Le taux de rappel définit le ratio entre le nombre d'informations intéressantes obtenues par un pair et le nombre d'informations disponibles dans le réseau. Les valeurs obtenues à la convergence de la simulation sont représentées dans les tableaux 6.5(a,b,c) et 6.6(a,b,c).

Comme pour le critère précédent, on remarque que le changement d'échelle du réseau n'affecte pas le profil des résultats obtenus. Egalement, l'augmentation du niveau de flou dans le jeu de données permet de restreindre la plage de valeur correspondant aux optima.

Globalement, le taux de rappel obtenu avoisine les 30% indépendamment de la taille du réseau, des données qui y circulent ou du contenu initial du carnet d'adresses. Un réglage à $s_{min} = 0.7$ permet néanmoins d'obtenir le meilleur résultat dans 86% des cas et l'un des deux meilleurs dans

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.44	0.43	0.37	0.42	0.38	0.34	0.32	0.21	0.18
	5	0.42	0.43	0.44	0.45	0.42	0.41	0.39	0.34	0.21
	10	0.44	0.37	0.38	0.37	0.37	0.38	0.37	0.35	0.20
	20	0.41	0.39	0.38	0.37	0.36	0.32	0.26	0.29	0.24
`Regular20`	0	0.39	0.41	0.40	0.33	0.31	0.26	0.29	0.25	0.14
	5	0.46	0.36	0.35	0.37	0.40	0.36	0.29	0.30	0.17
	10	0.38	0.30	0.36	0.38	0.38	0.34	0.27	0.28	0.21
	20	0.36	0.37	0.35	0.37	0.33	0.33	0.29	0.28	0.18
`Smallworld20`	0	0.44	0.40	0.35	0.35	0.38	0.39	0.36	0.26	0.14
	5	0.45	0.44	0.41	0.41	0.39	0.38	0.37	0.34	0.23
	10	0.44	0.40	0.34	0.37	0.34	0.34	0.39	0.40	0.23
	20	0.38	0.33	0.34	0.34	0.32	0.33	0.34	0.33	0.19

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.52	0.45	0.38	0.36	0.37	0.32	0.30	0.22	0.14
	5	0.54	0.46	0.41	0.41	0.38	0.40	0.40	0.33	0.16
	10	0.51	0.49	0.40	0.29	0.37	0.41	0.38	0.30	0.18
	20	0.50	0.37	0.36	0.38	0.37	0.38	0.37	0.35	0.22
`Regular20`	0	0.45	0.42	0.37	0.32	0.31	0.26	0.27	0.15	0.13
	5	0.50	0.39	0.39	0.33	0.32	0.32	0.29	0.25	0.13
	10	0.48	0.39	0.38	0.35	0.38	0.30	0.28	0.32	0.17
	20	0.46	0.37	0.32	0.31	0.35	0.39	0.34	0.25	0.18
`Smallworld20`	0	0.45	0.41	0.38	0.41	0.33	0.29	0.31	0.24	0.12
	5	0.48	0.45	0.42	0.40	0.40	0.37	0.35	0.30	0.20
	10	0.47	0.41	0.38	0.35	0.39	0.36	0.38	0.31	0.22
	20	0.40	0.35	0.36	0.33	0.32	0.28	0.35	0.30	0.21

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.54	0.57	0.51	0.38	0.36	0.28	0.26	0.24	0.13
	5	0.54	0.58	0.49	0.47	0.38	0.38	0.35	0.28	0.15
	10	0.56	0.54	0.50	0.40	0.38	0.36	0.36	0.25	0.16
	20	0.50	0.52	0.45	0.40	0.39	0.33	0.36	0.30	0.18
`Regular20`	0	0.51	0.49	0.48	0.40	0.30	0.24	0.20	0.15	0.13
	5	0.50	0.54	0.49	0.42	0.39	0.31	0.31	0.23	0.11
	10	0.51	0.52	0.45	0.35	0.29	0.28	0.30	0.21	0.14
	20	0.50	0.47	0.45	0.37	0.27	0.32	0.28	0.26	0.18
`Smallworld20`	0	0.50	0.50	0.46	0.40	0.39	0.25	0.28	0.20	0.10
	5	0.50	0.51	0.48	0.41	0.37	0.35	0.35	0.32	0.16
	10	0.52	0.48	0.47	0.38	0.34	0.38	0.33	0.33	0.14
	20	0.46	0.46	0.42	0.36	0.35	0.35	0.27	0.25	0.18

TAB. 6.3 – Influence de s_{min} sur la valeur du coefficient de groupement pour un réseau de 20 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.47	0.42	0.39	0.39	0.35	0.35	0.33	0.26	0.14
	5	0.43	0.41	0.41	0.37	0.36	0.36	0.36	0.34	0.22
	10	0.40	0.39	0.35	0.34	0.37	0.34	0.33	0.32	0.20
	20	0.35	0.33	0.30	0.32	0.31	0.31	0.33	0.31	0.20
`Regular40`	0	0.37	0.36	0.32	0.28	0.28	0.25	0.23	0.14	0.11
	5	0.38	0.33	0.32	0.31	0.29	0.27	0.28	0.25	0.15
	10	0.38	0.33	0.33	0.33	0.31	0.32	0.31	0.27	0.18
	20	0.29	0.31	0.29	0.28	0.26	0.29	0.27	0.25	0.17
`Smallworld40`	0	0.47	0.43	0.40	0.37	0.36	0.37	0.36	0.26	0.16
	5	0.46	0.41	0.39	0.40	0.40	0.40	0.35	0.34	0.25
	10	0.42	0.41	0.38	0.35	0.37	0.36	0.38	0.35	0.24
	20	0.35	0.38	0.35	0.32	0.34	0.32	0.29	0.34	0.22

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.55	0.44	0.45	0.42	0.36	0.31	0.32	0.21	0.15
	5	0.53	0.43	0.41	0.41	0.35	0.37	0.34	0.31	0.19
	10	0.51	0.44	0.40	0.36	0.35	0.37	0.37	0.30	0.24
	20	0.43	0.35	0.29	0.34	0.31	0.34	0.32	0.29	0.21
`Regular40`	0	0.44	0.39	0.32	0.29	0.30	0.26	0.21	0.14	0.08
	5	0.48	0.39	0.31	0.35	0.31	0.29	0.25	0.22	0.16
	10	0.46	0.40	0.33	0.33	0.33	0.29	0.28	0.27	0.16
	20	0.41	0.30	0.29	0.31	0.28	0.31	0.28	0.23	0.16
`Smallworld40`	0	0.51	0.46	0.44	0.38	0.36	0.34	0.32	0.25	0.15
	5	0.49	0.44	0.39	0.41	0.38	0.38	0.35	0.35	0.20
	10	0.49	0.44	0.40	0.40	0.36	0.34	0.34	0.32	0.23
	20	0.48	0.37	0.34	0.33	0.30	0.32	0.36	0.33	0.19

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.54	0.53	0.50	0.45	0.37	0.36	0.30	0.22	0.12
	5	0.52	0.54	0.48	0.43	0.38	0.36	0.35	0.29	0.16
	10	0.52	0.52	0.46	0.42	0.36	0.35	0.35	0.28	0.17
	20	0.48	0.45	0.40	0.36	0.31	0.30	0.32	0.28	0.18
`Regular40`	0	0.53	0.52	0.44	0.35	0.29	0.25	0.21	0.13	0.09
	5	0.49	0.50	0.44	0.34	0.34	0.28	0.25	0.21	0.12
	10	0.51	0.51	0.43	0.34	0.29	0.28	0.27	0.25	0.13
	20	0.47	0.49	0.43	0.34	0.27	0.26	0.23	0.23	0.13
`Smallworld40`	0	0.55	0.56	0.50	0.44	0.36	0.31	0.31	0.24	0.13
	5	0.53	0.53	0.51	0.41	0.40	0.36	0.35	0.30	0.17
	10	0.52	0.53	0.47	0.38	0.36	0.37	0.34	0.30	0.19
	20	0.50	0.46	0.41	0.40	0.35	0.33	0.33	0.30	0.21

TAB. 6.4 – Influence de s_{min} sur la valeur du coefficient de groupement pour un réseau de 40 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.34	0.34	0.34	0.34	0.35	0.35	0.35	0.34	0.30
	5	0.32	0.33	0.33	0.33	0.34	0.34	0.34	0.32	0.26
	10	0.32	0.32	0.32	0.32	0.32	0.33	0.33	0.31	0.24
	20	0.31	0.31	0.31	0.31	0.31	0.32	0.31	0.29	0.23
`Regular20`	0	0.33	0.33	0.33	0.34	0.34	0.35	0.35	0.34	0.29
	5	0.31	0.32	0.33	0.32	0.33	0.33	0.33	0.31	0.25
	10	0.30	0.31	0.31	0.31	0.32	0.32	0.31	0.30	0.24
	20	0.29	0.29	0.29	0.29	0.29	0.30	0.29	0.27	0.22
`Smallworld20`	0	0.35	0.36	0.37	0.37	0.37	0.38	0.37	0.36	0.31
	5	0.34	0.35	0.34	0.35	0.35	0.35	0.36	0.34	0.27
	10	0.32	0.33	0.32	0.33	0.33	0.34	0.34	0.32	0.24
	20	0.31	0.31	0.31	0.31	0.32	0.31	0.32	0.29	0.23

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.32	0.34	0.34	0.35	0.35	0.35	0.35	0.33	0.30
	5	0.32	0.33	0.33	0.33	0.33	0.33	0.33	0.32	0.26
	10	0.30	0.32	0.32	0.33	0.33	0.33	0.33	0.30	0.24
	20	0.30	0.30	0.31	0.31	0.32	0.31	0.30	0.29	0.22
`Regular20`	0	0.31	0.32	0.33	0.33	0.34	0.35	0.35	0.32	0.29
	5	0.30	0.31	0.32	0.32	0.33	0.33	0.33	0.31	0.25
	10	0.28	0.30	0.31	0.31	0.31	0.31	0.31	0.29	0.23
	20	0.27	0.29	0.30	0.29	0.30	0.30	0.30	0.27	0.22
`Smallworld20`	0	0.34	0.35	0.36	0.37	0.36	0.37	0.37	0.36	0.31
	5	0.32	0.34	0.35	0.35	0.35	0.35	0.35	0.33	0.26
	10	0.31	0.33	0.33	0.33	0.33	0.33	0.33	0.31	0.24
	20	0.30	0.31	0.32	0.31	0.32	0.31	0.32	0.28	0.23

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.30	0.30	0.32	0.33	0.34	0.34	0.35	0.33	0.29
	5	0.29	0.30	0.32	0.32	0.33	0.33	0.34	0.32	0.25
	10	0.28	0.29	0.30	0.32	0.32	0.33	0.32	0.30	0.23
	20	0.27	0.27	0.29	0.30	0.31	0.31	0.31	0.28	0.22
`Regular20`	0	0.29	0.29	0.30	0.32	0.32	0.33	0.34	0.32	0.28
	5	0.27	0.28	0.29	0.31	0.32	0.33	0.33	0.30	0.24
	10	0.26	0.27	0.28	0.30	0.31	0.31	0.31	0.29	0.23
	20	0.25	0.25	0.27	0.29	0.28	0.29	0.29	0.27	0.21
`Smallworld20`	0	0.31	0.32	0.33	0.34	0.36	0.37	0.37	0.35	0.30
	5	0.29	0.30	0.32	0.34	0.34	0.34	0.35	0.33	0.25
	10	0.29	0.29	0.31	0.32	0.32	0.33	0.33	0.31	0.23
	20	0.27	0.28	0.30	0.31	0.31	0.31	0.32	0.29	0.22

TAB. 6.5 – Influence de s_{min} sur la valeur du coefficient de rappel pour un réseau de 20 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.31	0.31	0.31	0.32	0.32	0.32	0.32	0.31	0.28
	5	0.30	0.31	0.31	0.31	0.31	0.31	0.31	0.29	0.24
	10	0.29	0.30	0.30	0.30	0.30	0.30	0.29	0.28	0.23
	20	0.28	0.29	0.29	0.29	0.29	0.29	0.28	0.27	0.21
`Regular40`	0	0.30	0.30	0.30	0.30	0.31	0.32	0.32	0.30	0.27
	5	0.28	0.29	0.29	0.30	0.30	0.30	0.30	0.29	0.24
	10	0.28	0.29	0.29	0.29	0.29	0.29	0.29	0.28	0.22
	20	0.27	0.28	0.28	0.28	0.28	0.28	0.28	0.26	0.21
`Smallworld40`	0	0.32	0.32	0.32	0.32	0.33	0.33	0.33	0.32	0.29
	5	0.30	0.31	0.31	0.31	0.31	0.31	0.31	0.30	0.25
	10	0.30	0.30	0.30	0.30	0.30	0.30	0.31	0.29	0.24
	20	0.29	0.29	0.29	0.29	0.29	0.30	0.30	0.27	0.22

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.29	0.31	0.31	0.31	0.32	0.32	0.32	0.31	0.27
	5	0.29	0.30	0.30	0.31	0.31	0.31	0.31	0.29	0.24
	10	0.27	0.29	0.30	0.30	0.30	0.30	0.30	0.28	0.22
	20	0.27	0.28	0.28	0.28	0.28	0.29	0.28	0.26	0.21
`Regular40`	0	0.28	0.30	0.30	0.30	0.31	0.31	0.31	0.30	0.27
	5	0.27	0.29	0.29	0.29	0.30	0.29	0.29	0.28	0.23
	10	0.26	0.28	0.29	0.29	0.28	0.29	0.29	0.27	0.22
	20	0.26	0.27	0.27	0.27	0.27	0.28	0.28	0.25	0.21
`Smallworld40`	0	0.30	0.31	0.32	0.32	0.32	0.32	0.33	0.32	0.28
	5	0.29	0.30	0.31	0.31	0.31	0.31	0.31	0.30	0.24
	10	0.28	0.29	0.30	0.30	0.30	0.30	0.30	0.29	0.23
	20	0.28	0.28	0.29	0.29	0.29	0.29	0.29	0.27	0.22

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.28	0.28	0.29	0.30	0.31	0.31	0.32	0.30	0.27
	5	0.27	0.27	0.28	0.30	0.30	0.30	0.31	0.29	0.23
	10	0.26	0.27	0.28	0.29	0.30	0.30	0.30	0.27	0.22
	20	0.25	0.26	0.27	0.28	0.29	0.29	0.29	0.26	0.21
`Regular40`	0	0.26	0.27	0.28	0.29	0.30	0.31	0.31	0.30	0.26
	5	0.26	0.26	0.27	0.28	0.29	0.29	0.29	0.27	0.23
	10	0.25	0.26	0.27	0.28	0.28	0.28	0.29	0.26	0.21
	20	0.24	0.25	0.26	0.27	0.27	0.28	0.28	0.25	0.21
`Smallworld40`	0	0.29	0.29	0.30	0.31	0.31	0.32	0.32	0.32	0.28
	5	0.27	0.28	0.29	0.30	0.31	0.31	0.31	0.29	0.24
	10	0.27	0.27	0.28	0.29	0.30	0.30	0.30	0.28	0.22
	20	0.25	0.26	0.28	0.28	0.29	0.29	0.29	0.27	0.21

TAB. 6.6 – Influence de s_{min} sur la valeur du coefficient de rappel pour un réseau de 40 pairs

99% des cas. La valeur inférieure de 0.6 permet quant à elle d'atteindre une des deux meilleurs valeurs dans 100% des cas et l'optimum dans 79% des cas

Critère de précision

Les résultats pour le critère de précision, quatifiant le taux d'informations intéressantes reçu, sont visibles dans les tableaux 6.7(a,b,c) et 6.8(a,b,c) respectivement pour les réseaux composés de 20 et 40 pairs.

C'est une valeur entre 0.7 et 0.8 qui permet d'obtenir les meilleurs résultats avec un taux de précision allant jusqu'à 85%.

Résultat

Chacun de ces trois critères permet donc de conclure sur une valeur idéale pour s_{min} :

- taux de groupement : $s_{min} \in \{0.1, 0.2\}$
- taux de rappel : $s_{min} \in \{0.6, 0.7\}$
- taux de précision : $s_{min} \in \{0.7, 0.8\}$

La valeur 0.7 était proposée deux fois, c'est celle-ci qui nous retiendrons comme solution optimale pour s_{min}. Ce résultat est à rapprocher le la similarité moyenne entre les informations d'une même classe. Cette similarité étant d'environ 0.78, s_{min} trouve son réglage optimal dans la valeur la plus proche strictement inférieure

6.4.2 Paramétrage de n^+

En gardant $\eta = 0.8$ et en fixant $s_{min} = 0.7$ nous testons à présent l'inpact de n^+ sur les différents critères. Les valeurs testées sont celle introduites précédement, $\{0, 5, 10, 20\}$.

Critère de groupement

Dans le cas des réseaux composés de 20 pairs, la meilleure valeur de n^+ dépend de la base de données d'informations utilisée. Toutes les valeurs possibles sont estimées au moins une fois comme étant les plus efficaces. Cependants, certaines d'entre elles peuvent être exclues. Nottament le cas $n^+ = 0$ qui permet soit d'obtenir un résultat proche de ceux obtenus pour les autres valeurs de n^+ (voir, par exemple, le tableau 6.8(a)), soit un résultat nettement inférieur (tableaux 6.8(b) et 6.8(c)). Son opposé, $n^+ = 20$ conduit également à des résultats moins bons ou similaires.

La signification de ces deux valeurs permet d'expliquer ce résultat :

- dans le cas où $n^+ = 0$ les fourmis auront autant de chance d'envoyer une information à un pair non intéressé que de rester au nid. Les pairs auront donc plus de chances d'être « spammés ». Cette circulation d'informations non intéressante introduit un bruit général dans le réseau qui rend plus difficile l'indentification des groupes. L'impact de ce dernier

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.64	0.64	0.63	0.63	0.64	0.65	0.66	0.62	0.54
	5	0.70	0.72	0.73	0.74	0.75	0.75	0.76	0.76	0.69
	10	0.75	0.77	0.77	0.78	0.78	0.79	0.81	0.81	0.75
	20	0.80	0.81	0.82	0.83	0.83	0.83	0.85	0.85	0.82
`Regular20`	0	0.61	0.61	0.61	0.62	0.63	0.64	0.65	0.61	0.52
	5	0.68	0.71	0.72	0.71	0.73	0.74	0.75	0.75	0.67
	10	0.72	0.74	0.76	0.76	0.77	0.78	0.78	0.78	0.74
	20	0.77	0.78	0.78	0.78	0.79	0.80	0.81	0.82	0.81
`Smallworld20`	0	0.64	0.66	0.67	0.68	0.68	0.69	0.69	0.64	0.56
	5	0.72	0.74	0.75	0.75	0.76	0.77	0.78	0.78	0.71
	10	0.75	0.77	0.77	0.78	0.80	0.80	0.83	0.82	0.76
	20	0.80	0.81	0.82	0.82	0.83	0.82	0.85	0.85	0.82

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.60	0.62	0.63	0.64	0.64	0.64	0.65	0.61	0.53
	5	0.66	0.71	0.72	0.73	0.74	0.75	0.76	0.75	0.69
	10	0.68	0.74	0.77	0.78	0.78	0.80	0.81	0.80	0.74
	20	0.71	0.78	0.81	0.83	0.83	0.84	0.84	0.86	0.82
`Regular20`	0	0.58	0.59	0.60	0.61	0.62	0.63	0.64	0.58	0.51
	5	0.62	0.67	0.70	0.71	0.72	0.74	0.75	0.74	0.65
	10	0.63	0.70	0.74	0.75	0.76	0.77	0.79	0.79	0.73
	20	0.66	0.75	0.78	0.79	0.80	0.81	0.82	0.82	0.81
`Smallworld20`	0	0.63	0.65	0.66	0.67	0.67	0.67	0.68	0.64	0.55
	5	0.67	0.72	0.74	0.75	0.76	0.76	0.77	0.76	0.70
	10	0.69	0.75	0.78	0.78	0.79	0.80	0.81	0.81	0.76
	20	0.70	0.78	0.82	0.82	0.83	0.83	0.85	0.85	0.82

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0	0.56	0.56	0.58	0.60	0.63	0.63	0.64	0.61	0.51
	5	0.58	0.59	0.65	0.69	0.73	0.74	0.75	0.75	0.66
	10	0.59	0.61	0.66	0.74	0.77	0.79	0.79	0.79	0.73
	20	0.61	0.61	0.70	0.78	0.81	0.83	0.84	0.85	0.81
`Regular20`	0	0.54	0.53	0.55	0.59	0.59	0.61	0.62	0.57	0.49
	5	0.54	0.55	0.60	0.67	0.69	0.72	0.75	0.72	0.64
	10	0.54	0.56	0.63	0.69	0.73	0.76	0.77	0.78	0.72
	20	0.56	0.57	0.65	0.74	0.76	0.78	0.81	0.83	0.79
`Smallworld20`	0	0.57	0.58	0.60	0.63	0.64	0.67	0.67	0.64	0.53
	5	0.58	0.59	0.66	0.71	0.73	0.76	0.76	0.77	0.68
	10	0.59	0.60	0.68	0.74	0.76	0.78	0.80	0.81	0.74
	20	0.60	0.62	0.71	0.77	0.81	0.82	0.84	0.85	0.81

TAB. 6.7 – Influence de s_{min} sur la valeur du coefficient de précision pour un réseau de 20 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.62	0.62	0.62	0.63	0.64	0.64	0.64	0.61	0.53
	5	0.70	0.72	0.73	0.73	0.75	0.74	0.76	0.74	0.67
	10	0.74	0.76	0.77	0.77	0.78	0.79	0.79	0.80	0.75
	20	0.79	0.81	0.81	0.82	0.82	0.83	0.83	0.84	0.81
`Regular40`	0	0.59	0.60	0.60	0.60	0.62	0.63	0.63	0.58	0.50
	5	0.67	0.70	0.71	0.71	0.72	0.73	0.73	0.73	0.65
	10	0.72	0.74	0.75	0.76	0.76	0.77	0.77	0.78	0.73
	20	0.77	0.79	0.80	0.80	0.81	0.81	0.83	0.83	0.81
`Smallworld40`	0	0.63	0.63	0.63	0.64	0.65	0.66	0.66	0.62	0.55
	5	0.71	0.73	0.73	0.73	0.75	0.75	0.76	0.76	0.70
	10	0.75	0.77	0.77	0.78	0.78	0.79	0.80	0.80	0.76
	20	0.80	0.82	0.81	0.82	0.82	0.84	0.85	0.85	0.83

(b) Jeu de données `NoisyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.59	0.61	0.62	0.62	0.63	0.64	0.64	0.60	0.52
	5	0.64	0.69	0.72	0.72	0.73	0.75	0.75	0.74	0.67
	10	0.65	0.73	0.77	0.77	0.77	0.79	0.80	0.79	0.74
	20	0.69	0.77	0.80	0.81	0.82	0.83	0.83	0.84	0.81
`Regular40`	0	0.56	0.58	0.60	0.60	0.61	0.62	0.61	0.57	0.50
	5	0.60	0.67	0.69	0.70	0.71	0.72	0.73	0.71	0.65
	10	0.62	0.71	0.74	0.75	0.75	0.76	0.78	0.77	0.72
	20	0.66	0.75	0.79	0.79	0.80	0.82	0.83	0.82	0.80
`Smallworld40`	0	0.60	0.62	0.63	0.64	0.64	0.65	0.65	0.62	0.54
	5	0.64	0.70	0.72	0.72	0.74	0.74	0.76	0.75	0.68
	10	0.67	0.72	0.77	0.78	0.78	0.79	0.80	0.80	0.75
	20	0.70	0.77	0.82	0.82	0.83	0.83	0.84	0.84	0.82

(c) Jeu de données `FuzzyDataset`

Réseau	n^+	s_{min}								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0	0.55	0.56	0.58	0.60	0.61	0.63	0.63	0.59	0.51
	5	0.57	0.58	0.63	0.68	0.71	0.73	0.75	0.73	0.65
	10	0.58	0.59	0.65	0.72	0.76	0.77	0.79	0.79	0.73
	20	0.60	0.62	0.68	0.77	0.81	0.82	0.83	0.84	0.81
`Regular40`	0	0.52	0.52	0.55	0.57	0.59	0.60	0.60	0.57	0.48
	5	0.54	0.55	0.60	0.65	0.68	0.71	0.73	0.71	0.63
	10	0.55	0.57	0.62	0.71	0.74	0.75	0.76	0.76	0.71
	20	0.55	0.58	0.66	0.75	0.79	0.81	0.81	0.82	0.79
`Smallworld40`	0	0.57	0.57	0.58	0.61	0.62	0.64	0.64	0.62	0.53
	5	0.58	0.59	0.63	0.69	0.72	0.73	0.75	0.74	0.67
	10	0.60	0.61	0.66	0.73	0.76	0.77	0.79	0.79	0.74
	20	0.60	0.62	0.70	0.77	0.81	0.83	0.83	0.85	0.82

TAB. 6.8 – Influence de s_{min} sur la valeur du coefficient de précision pour un réseau de 40 pairs

sur la valeur finale obtenu est d'autant plus important que les données en entrée étaient elles-même bruitées (cf 1ère colonne des tableaux 6.8(a), 6.8(b) et 6.8(c)).

- dans le cas où $n^+ = 20$ les fourmis auront une forte tendance à préférer rester au nid. En considérant que le pair est connecté à $V_{max} = 4$ voisins tous non intéressés, l'application numérique des formules décrites au chapitre précédent nous indique que la probabilité de bouclage sur le nid est de 84% alors que celle de choisir un des voisin sera de 4%. La circulation d'information est par conséquent globalement diminuée et, là encore, les groupes sont mis en place plus difficilement.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.32	0.39	0.37	0.26
`Regular20`	0.29	0.29	0.27	0.29
`Smallworld20`	0.36	0.37	0.39	0.34

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.30	0.40	0.38	0.37
`Regular20`	0.27	0.29	0.28	0.34
`Smallworld20`	0.31	0.35	0.38	0.35

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.26	0.35	0.36	0.36
`Regular20`	0.20	0.31	0.30	0.28
`Smallworld20`	0.28	0.35	0.33	0.27

TAB. 6.9 – Influence de n^+ sur la valeur du coefficient de groupement pour un réseau de 20 pairs

Contrairement aux tests effectués sur s_{min}, le passage à un réseau de 40 pairs ne confirme pas résultats obtenus avec le réseau de 20 machines mais propose des valeurs différentes. Cependant les remarques formulées sur les deux valeurs extrèmes restent valides.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.33	0.36	0.33	0.33
`Regular20`	0.23	0.28	0.31	0.27
`Smallworld20`	0.36	0.35	0.38	0.29

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.32	0.34	0.37	0.32
`Regular20`	0.21	0.25	0.28	0.28
`Smallworld20`	0.32	0.35	0.34	0.36

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.30	0.35	0.35	0.32
`Regular20`	0.21	0.25	0.27	0.23
`Smallworld20`	0.31	0.35	0.34	0.33

TAB. 6.10 – Influence de n^+ sur la valeur du coefficient de groupement pour un réseau de 40 pairs

D'après les résultats sur le coefficient de groupement, il apparait finalement qu'une valeur de $n^+ = 5$ ou $n^+ = 10$ soient préférables. Dans le cas d'un réseau à 20 machines, c'est la plus faible des deux valeurs qui est la plus satisfaisante. Pour un réseau de 40 machines, mieux vaut opter pour $n^+ = 10$.

Critère de précision

Les tableaux 6.10(a), 6.10(b) et 6.10(c) montrent une très nette croissance de la valeur de la précision en fonction de la valeur de n^+. Ceci s'explique par le même phénomène décrit précédement : une valeur de n^+ élévée diminue les probabilités d'envoi d'information vers des pairs potentiellement non intéressés. La précision, qui décrit la proportion d'information intéressantes reçues sur le nombre d'informations totales reçues, est d'autant plus élevée que ces envois sont limités.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.66	0.76	0.81	0.85
`Regular20`	0.65	0.75	0.78	0.81
`Smallworld20`	0.69	0.78	0.83	0.85

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.65	0.76	0.81	0.84
`Regular20`	0.64	0.75	0.79	0.82
`Smallworld20`	0.68	0.77	0.81	0.85

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.64	0.75	0.79	0.84
`Regular20`	0.62	0.75	0.77	0.81
`Smallworld20`	0.67	0.76	0.80	0.84

TAB. 6.11 – Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 20 pairs

Les résultats obtenus avec le réseau de 40 pairs sont identiques comme l'attestent les tableaux 6.11(a), 6.11(b) et 6.11(c). Cette constance dans les résultats tend à monter la capacité de monter à l'échelle de PIAF. La qualité du tri effectué lors de la diffusion des informations ne dépend pas de la taille du réseau sous-jacent.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.64	0.76	0.79	0.83
`Regular20`	0.63	0.73	0.77	0.83
`Smallworld20`	0.66	0.76	0.80	0.85

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.64	0.75	0.80	0.83
`Regular20`	0.61	0.73	0.78	0.83
`Smallworld20`	0.65	0.76	0.80	0.84

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.63	0.75	0.79	0.83
`Regular20`	0.60	0.73	0.76	0.81
`Smallworld20`	0.64	0.75	0.79	0.83

TAB. 6.12 – Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 40 pairs

Les deux meilleures valeurs retenues d'après ce second critère sont $n^+ = 10$ et $n^+ = 20$.

Critère de rappel

Le dernier critère à prendre en compte pour choisir la valeur de n^+ est celui du rappel. On constate que, de façon générale, les résultats chutent avec une valeur de n^+ croissante. C'est en favorisant une diffusion plus systématique $n^+ = 0$ que ces pairs ont le plus de chances de

récupérer toutes les informations (cf tableaux 6.12(a), 6.12(b) et 6.12(c)) mais cela se fait alors au détriment de la précision (tableaux 6.10(a), 6.10(b), 6.10(c)). Avec une diffusion d'informations limité, les pairs ont moins de chance de récupérer les informations intéressantes. A l'opposé, une diffusion plus systématique entraine la récupération de la majeure partie des informations en circulation, intéressantes ou pas.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.35	0.34	0.33	0.31
`Regular20`	0.35	0.33	0.31	0.29
`Smallworld20`	0.37	0.36	0.34	0.32

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.35	0.33	0.33	0.30
`Regular20`	0.35	0.33	0.31	0.30
`Smallworld20`	0.37	0.35	0.33	0.32

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.35	0.34	0.32	0.31
`Regular20`	0.34	0.33	0.31	0.29
`Smallworld20`	0.37	0.35	0.33	0.32

TAB. 6.13 – Influence de n^+ sur la valeur du coefficient de rappel pour un réseau de 20 pairs

Dans le cas du réseau de 40 pairs, les constatations sont les mêmes (cf tableaux 6.13(a), 6.13(b) et 6.13(c)). Cependant, les valeurs finales obtenues sont plus faibles avec un taux de 31% en moyenne contre 35% pour le réseau précédent. Ceci est une conséquence directe de la taille du réseau.

(a) Jeu de données `SeparatedDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.32	0.31	0.29	0.28
`Regular20`	0.32	0.30	0.29	0.28
`Smallworld20`	0.33	0.31	0.31	0.30

(b) Jeu de données `NoisyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.32	0.31	0.30	0.28
`Regular20`	0.31	0.29	0.29	0.28
`Smallworld20`	0.33	0.31	0.30	0.29

(c) Jeu de données `FuzzyDataset`

Réseau	n^+			
	0	5	10	20
`Random20`	0.32	0.31	0.30	0.29
`Regular20`	0.31	0.29	0.29	0.28
`Smallworld20`	0.32	0.31	0.30	0.29

TAB. 6.14 – Influence de n^+ sur la valeur du coefficient de rappel pour un réseau de 40 pairs

D'après le coefficient de rappel, les valeurs idéales de n^+ sont donc 0 et 5.

Résultat

Comme nous l'avons fait pour s_{min}, confrontons les résultats des 3 estimateurs afin de décider d'une valeur idéale pour n^+ :

- taux de groupement : $n^+ \in \{5, 10\}$

- taux de rappel : $n^+ \in \{10, 20\}$
- taux de précision : $n^+ \in \{0, 5\}$

En suivant le raisonnement que pour s_{min}, celui de compter le nombre de citations d'une valeur, nous obtenons un « match nul » entre $n^+ = 5$ et $n^+ = 10$. Ces deux valeurs convenant à un taux groupement satisfaisant, le choix doit se faire entre privilégier le taux de rappel ou celui de précision.

Nous avons jusqu'à présent utilisés ces estimateurs comme autant de fonctions mathématiques équivalentes entre elles. Cependant, d'un point de vue de l'utilisateur, les résultats concrets que traduisent ces fonctions permettent de définir un ordre de priorité. En effet, seuls le taux de groupement et de précision seront directement appréciables par cet utilisateur. Le premier sous la forme d'une indication de voisinage qui pourra prendre la forme, par exemple, d'une carte affichée par un agent situé dans le niveau applicatif de PIAF. Le second aura un impact direct sur le contenu des informations qui seront accessibles à l'utilisateur. L'utilisateur n'aura par contre aucun *a priori* sur le nombre d'informations susceptibles de l'intéresser dans le réseau et ne sera donc pas en mesure d'apprécier le taux de rappel du système lorsqu'il consultera la liste des informations reçues.

Dans ce cadre, il est donc préférable de privilégier le taux de précision à celui de rappel. La solution retenue est finalement $n^+ = 5$.

6.4.3 Paramétrage de η

Nous avons à présent estimé $n^+ = 5$ et $s_{min} = 0.7$ comme étant les deux valeurs les plus adaptées pour un choix de $\eta = 0.8$. Cette dernière série de tests permet de vérifier l'impact de η sur les performances du système en faisant varier sa valeur sur $[0.1, 0.9]$. Selon que cette valeur sera faible ou non, le taux d'exploration sera plus ou moins important car η est le paramètre d'ajustement du choix des fourmis entre le fait d'envoyer des informations vers des pairs potentiellement intéressés ou non.

Critère de groupement

Le réseau constitué de 20 machines présente des résultats très variables selon le jeux de données consideré. Le point commun entre les tableaux 6.14(a),6.14(b),6.14(c) est que la valeur retenue doit être supérieure à 0.5. Il faut donc dans tous les cas s'assurer que l'exploitation sera priviligiée par rapport à l'exploration. Accorder trop d'importance à l'envoi de données vers des pairs non intéressé entraîne une baisse des performance du système. La différence se fait ensuite sur le choix de η dans $\{0.5, 0.6, 0.7, 0.8, 0.9\}$ avec pour tendance générale de meilleurs résultats quand le choix du valeur plus importance coincide avec des données bruitées et/ou un réseau dont les connexions sont très mélangées.

Le réseau de 40 machines nuance cette première conclusion. Il reste important de garder η supérieur à 0.5 mais le choix $\eta = 0.7$ se démarque plus nettement des autres en permettant d'atteindre une solution optimale ou proche de l'optimal dans tous les cas de figure.

Le choix de cette valeur dans les réseaux de 20 machines ne fournit pas non plus de mauvais résultats même si l'optimal ou un sous optimal que dans 44% des cas testés.

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.24	0.25	0.36	0.35	0.42	0.36	0.38	0.39	0.39
`Regular20`	0.21	0.26	0.26	0.30	0.34	0.35	0.32	0.29	0.35
`Smallworld20`	0.24	0.30	0.36	0.37	0.41	0.37	0.37	0.37	0.41

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.21	0.30	0.28	0.34	0.32	0.26	0.36	0.40	0.37
`Regular20`	0.20	0.24	0.25	0.26	0.26	0.26	0.33	0.29	0.31
`Smallworld20`	0.24	0.29	0.34	0.32	0.34	0.37	0.33	0.35	0.36

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.18	0.25	0.28	0.30	0.30	0.28	0.31	0.35	0.37
`Regular20`	0.19	0.21	0.23	0.24	0.27	0.29	0.31	0.31	0.31
`Smallworld20`	0.18	0.21	0.22	0.32	0.34	0.34	0.36	0.35	0.35

TAB. 6.15 – Influence de η sur la valeur du coefficient de groupement pour un réseau de 20 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.29	0.31	0.31	0.34	0.35	0.35	0.35	0.36	0.36
`Regular40`	0.18	0.21	0.24	0.26	0.27	0.32	0.29	0.28	0.28
`Smallworld40`	0.27	0.34	0.34	0.37	0.35	0.37	0.38	0.35	0.34

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.27	0.25	0.28	0.32	0.33	0.36	0.36	0.34	0.39
`Regular40`	0.17	0.22	0.22	0.25	0.26	0.26	0.27	0.25	0.27
`Smallworld40`	0.28	0.27	0.27	0.36	0.33	0.34	0.36	0.35	0.32

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.21	0.22	0.29	0.29	0.31	0.32	0.33	0.35	0.35
`Regular40`	0.19	0.19	0.20	0.22	0.23	0.25	0.28	0.25	0.27
`Smallworld40`	0.23	0.25	0.29	0.32	0.34	0.33	0.34	0.35	0.30

TAB. 6.16 – Influence de η sur la valeur du coefficient de groupement pour un réseau de 40 pairs

Critère de precision

Les résultats pour ce critère sont respectivement présentées dans les tableaux 6.17(a,b,c) puis 6.18(a,b,c) respectivement pour les réseaux de 20 et 40 machines. On remarque que l'intervalle de définition $]0.5, 0.9]$ donne les meilleurs résultats, ce qui va confirme les conclusions formulées précédemment. Le choix d'un η permet d'ajuster le taux d'envoi de données vers des pairs non intéressés, c'est donc sans surprise (et en tirant les mêmes conclusions que pour le choix d'un n^+ élevé) que nous retrouvons ici une valeur du coefficient de précision d'autant plus importante que ces envois sont faibles.

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.68	0.70	0.72	0.75	0.75	0.75	0.76	0.76	0.78
`Regular20`	0.65	0.68	0.70	0.72	0.74	0.74	0.75	0.75	0.74
`Smallworld20`	0.69	0.71	0.73	0.75	0.76	0.78	0.78	0.78	0.79

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.66	0.70	0.72	0.74	0.75	0.75	0.76	0.76	0.76
`Regular20`	0.65	0.68	0.70	0.71	0.73	0.73	0.74	0.75	0.75
`Smallworld20`	0.68	0.71	0.73	0.75	0.76	0.77	0.77	0.77	0.78

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.66	0.69	0.72	0.73	0.74	0.74	0.75	0.75	0.75
`Regular20`	0.64	0.67	0.70	0.71	0.72	0.74	0.74	0.75	0.74
`Smallworld20`	0.68	0.71	0.73	0.74	0.76	0.76	0.77	0.76	0.77

TAB. 6.17 – Influence de η sur la valeur du coefficient de precision pour un réseau de 20 pairs

Ces conclusions se retrouvent dans le cas des réseaux à 40 machines. On constate également que la taille du réseau n'a qu'un impact limité sur la valeur de la précision atteinte.

Critère de rappel

Contrairement à ce qui a pu être observé lors des tests concernant n^+ et bien que η contrôle, lui aussi le taux de diffusion qui sera effectué. Nous constatons qu'une variation de η dans l'intervalle $]O.5, 0.9]$ n'a pas de conséquence importante sur les valeurs du coefficient de rappel. Ceci est valable aussi bien dans le cas des réseaux de 20 machines (tableaux 6.19(a,b,c)) que pour ceux à 40 pairs (tableaux 6.20(a,b,c)).

Ceci indique que lors de la diffusion des informations, c'est la tendance qu'à une fourmi à rester au nid une fois le choix de prendre une direction non intéressée qui a le plus fort impact face à la décision de se rendre vers un pair intéressé ou non.

RésultatUne dernière comparaison des résultats nous permet de fixer une valeur pour η :

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.66	0.69	0.71	0.73	0.73	0.75	0.75	0.76	0.76
`Regular40`	0.64	0.67	0.69	0.71	0.72	0.73	0.73	0.73	0.75
`Smallworld40`	0.68	0.70	0.71	0.73	0.75	0.75	0.76	0.76	0.76

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.66	0.68	0.70	0.73	0.73	0.75	0.75	0.75	0.76
`Regular40`	0.64	0.66	0.68	0.70	0.71	0.72	0.72	0.73	0.74
`Smallworld40`	0.67	0.69	0.72	0.73	0.74	0.75	0.75	0.76	0.76

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.65	0.68	0.69	0.71	0.72	0.74	0.74	0.75	0.74
`Regular40`	0.63	0.65	0.68	0.69	0.71	0.71	0.72	0.73	0.73
`Smallworld40`	0.66	0.69	0.71	0.73	0.73	0.74	0.74	0.75	0.75

TAB. 6.18 – Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 40 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.24	0.27	0.28	0.30	0.32	0.32	0.34	0.34	0.35
`Regular20`	0.24	0.26	0.28	0.30	0.31	0.32	0.33	0.33	0.33
`Smallworld20`	0.25	0.27	0.29	0.31	0.32	0.34	0.35	0.36	0.36

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.24	0.27	0.28	0.30	0.31	0.32	0.33	0.33	0.34
`Regular20`	0.24	0.26	0.28	0.29	0.31	0.31	0.33	0.33	0.33
`Smallworld20`	0.25	0.27	0.29	0.31	0.32	0.33	0.34	0.35	0.35

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random20`	0.24	0.26	0.28	0.30	0.31	0.32	0.33	0.34	0.34
`Regular20`	0.24	0.26	0.28	0.29	0.31	0.32	0.32	0.33	0.33
`Smallworld20`	0.24	0.27	0.29	0.31	0.32	0.33	0.34	0.35	0.35

TAB. 6.19 – Influence de η sur la valeur du coefficient de rappel pour un réseau de 20 pairs

(a) Jeu de données `SeparatedDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.23	0.25	0.26	0.28	0.29	0.30	0.31	0.31	0.31
`Regular40`	0.23	0.24	0.26	0.27	0.28	0.29	0.30	0.30	0.31
`Smallworld40`	0.23	0.25	0.27	0.28	0.30	0.30	0.31	0.31	0.32

(b) Jeu de données `NoisyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.23	0.24	0.26	0.28	0.28	0.30	0.30	0.31	0.31
`Regular40`	0.23	0.24	0.26	0.27	0.28	0.29	0.29	0.29	0.30
`Smallworld40`	0.23	0.25	0.27	0.28	0.29	0.30	0.31	0.31	0.31

(c) Jeu de données `FuzzyDataset`

Réseau	*eta*								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
`Random40`	0.23	0.24	0.26	0.27	0.28	0.29	0.31	0.31	0.31
`Regular40`	0.22	0.24	0.26	0.27	0.28	0.28	0.30	0.29	0.30
`Smallworld40`	0.23	0.25	0.26	0.28	0.29	0.30	0.31	0.31	0.31

TAB. 6.20 – Influence de η sur la valeur du coefficient de rappel pour un réseau de 40 pairs

- taux de groupement : $\eta \in \{0.7, 0.9\}$
- taux de rappel : $\eta \in \{0.7, 0.8, 0.9\}$ avec des résultats numériques sensiblement identiques pour les tests effectués.
- taux de précision : $\eta \in \{0.7, 0.8, 0.9\}$ avec des résultats numériques sensiblement identiques pour les tests effectués.

Le choix se fait entre $\eta = 0.9$ et $\eta = 0.7$. Le premier permet d'atteindre de meilleurs résultats pour les réseaux de 20 machines alors que le second réglage convient mieu aux réseaux de 40 pairs. Afin de fixer des paramètres globaux destinés à s'affranchir de suppositions concernant la taille du réseau, nous proposons d'opter pour $\eta = 0.7$.

6.4.4 Discussion

Cette série de tests était destinée à trouver un ensemble de valeurs « idéales » pour les paramètres s_{min}, n^+, η. En faisant varier la taille du réseau, le contenu de l'ensemble des informations utilisées et le contenu initial du carnet d'adresses nous avons pu conclure sur le réglage suivant : $s_{min} = 0.7$, $n^+ = 5$ et $\eta = 0.7$.

Deux critiques peuvent être formulées au regard de cette étude des paramètres : la première concernant le choix des paramètres étudiées et la seconde le choix arbitraire d'un certain nombre de valeurs pour d'autres paramètres.

Sur les 18 paramètres que comporte le système, seuls trois ont été étudiés. Il aurait en effet pu être intéressant d'effectuer une série de tests afin de déterminer, par exemple, le nombre idéal de fourmis, la taille de la mémoire anti-retour associée aux informations ou encore la taille de la mémoire utilisée pour évaluer la qualité du voisinage. Toutes ces variables se retrouvent dans l'un ou l'autre des algorithmes régissant le fonctionnement de PIAF (celui de diffusion et celui

de modification des connexions) et ont un impact direct ou indirect sur les deux. Modifier la taille de la mémoire utilisée pour l'évaluation des connexions influera sur la dynamique de la topologie qui, à son tour, aura des répercutions sur les choix de directions s'offrant aux fourmis de l'algorithme de diffusion. Pour cette étude, nous avons donc décidé de nous focaliser sur un seul des deux algorithmes et ses paramètres principaux. Du point de vue de l'utilisateur, l'impact direct de l'algorithme de diffusion est une (bonne) circulation de l'information dans le réseau. La topologie ne permet quant à elle de produire une carte du voisinage fournissant à l'utilisateur une vision de son réseau social. C'est pourquoi, c'est l'algorithme de diffusion qui a été étudié avec ses 3 paramètres principaux s_{min}, n^+, η.

Les valeurs préfixées l'ont été en cherchant à établir des liens entre elles. Par exemple, $V_{max} = TTL$ afin que la durée de vie maximale d'une information soit égale à la taille maximale du voisinage d'un pair. En guise d'autre exemple, la taille de la mémoire pour l'évaluation des connexions est égale au nombre de fourmis afin de prendre en compte au maximum 1 évaluation de chaque fourmi. Les variables n'étant pas liées à d'autres sont fixées à des valeurs critiques comme c'est le cas pour la mémoire anti-retour des informations qui est fixée à 1 afin de se limiter à l'interdiction d'un retour vers le dernier emetteur. Seul un modèle statistique complet du système permettrait d'apporter une idée plus précise de l'impact de ces paramètres sur les performances globales. L'imbrication des deux algorithmes de diffusion et d'ajustement de la topologie fait cependant de la construction d'un tel modèle une tâche très complexe (en supposant qu'elle soit réalisable). Comme c'est souvent le cas lors des études sur les systèmes P2P, nous avons opté pour une étude axée sur la simulation afin de statuer sur les meilleurs valeurs pour certains paramètres du système sous un certain nombre d'hypothèses concernant les autres.

Néanmoins, ces résultats peuvent être supposés généralisables à d'autres cas de réseaux ou de données en les liant à d'autres variables ou en les supposant facilement paramétrables par l'utilisateur. Ainsi, $s_{min} = 0.7$ revient à choisir ce paramètre comme étant légèrement inférieur à la similarité moyenne intraclasse des informations mises en circulation. Dans le contexte d'une application réelle, cette information ne sera pas connue (pour cela, il faudrait calculer la similarité entre toutes les informations produites par les utilisateurs) mais sa valeur supposée par l'utilisateur selon ses désirs. Cette valeur pourrait aussi être utilisée pour η : plus l'utilisateur désire obtenir d'informations ciblées, moins l'outil n'effectuera de diffusion vers des pairs susceptible d'envoyer des informations peu intéressantes en retour. n^+ peut quant à lui être fixé égal à la taille maximum du voisinage V_{max}.

Pour l'utilisateur, ces réglages pourraient être présentés sous la forme de l'interface de la figure 6.10. Il est important de noter ici qu'il n'est toujours pas question pour l'utilisateur d'indiquer ses centres d'intérêt afin de constituer un profil mais de spécifier des limites pour l'outil de circulation de l'information.

6.5 Comparaison avec une diffusion aléatoire

L'algorithme de circulation d'informations proposé dans PIAF a pour objectif une diffusion autonome et sélective d'informations dans un réseau P2P pur non structuré. D'après ce que nous avons vu au au point 3.3.1.2 du chapitre précédent, la circulation d'informations dans ces réseaux est généralement assurée selon un algorithme aléatoire de marche ou de diffusion. Ces algorithmes peuvent dans certains cas se baser sur des données locales. Pour PIAF, ce sont les

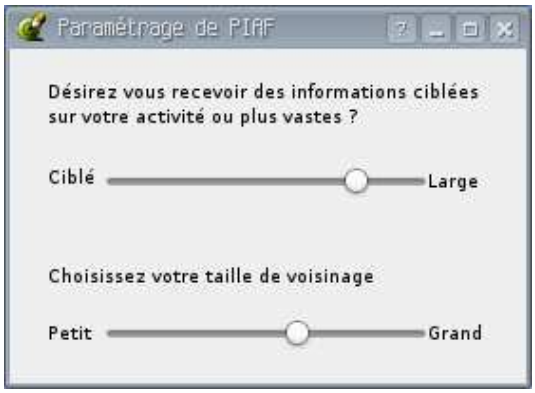


FIG. 6.10 – Fenêtre de réglage de PIAF (esquisse)

traces laissées par les transferts précédents qui sont utilisées pour guider les fourmis dans leurs choix.

Nous proposons ci-après une comparaison avec une diffusion aléatoire. Celle-ci illustre le cas où les fourmis n'auraient pas à disposition ces traces leur permettant d'optimiser leur transferts. Dans ce cas particulier, tous les pairs du voisinage pourront équipabrobablement être choisis comme destination. La probabilité de transmission $P^{rand}_{i\to j}(I,t)$ correspondante utilise la même formulation que $P^{send}_{i\to j}(I,t)$ (equation 5.5) mais en considérant que tous les pairs ont reçu une mauvaise évaluation de la part de la fourmi ($V_i(t) = V_i^{bad}(I,t)$).

$$P^{send}_{i\to j}(I,t)\Big|_{V_i^{bad}(I,t)=V_i(t)} = \begin{cases} \dfrac{1}{|V_i(t)|+1+n^+} & \text{si} \quad n_j \in V_i(t) \\ \\ \dfrac{1+n^+}{|V_i(t)|+1+n^+} & \text{si} \quad i=j \end{cases} \tag{6.2}$$

Afin de ne pas influencer le choix revenant à ne pas quitter le nid, nous considérerons pour la diffusion aléatoire que $n^+ = 0$. Ce qui, finalement, $P^{rand}_{i\to j}(I,t)$ comme étant égale à l'inverse de la taille du voisinage plus 1 pour tous les choix possibles (equation 6.3).

$$\forall n_j \in V_i(t) \cup \{n_i\}, \quad P^{rand}_{i\to j}(I,t) = \frac{1}{|V_i(t)|+1} \tag{6.3}$$

L'utilisation de l'aléatoire ne concerne que la partie diffusion de l'information de PIAF. Ce qui signifie que les fourmis continueront d'évaluer les pairs mais que cet avis n'entrera pas en compte lors du calcul de la probabilité de choix de la destination. Ces avis seront utilisés par l'algorithme de modification dynamique du réseau resté inchangé.

Les résultats de cette comparaison sont rapportés sur les figures 6.11 et 6.12. Les conditions de tests sont les mêmes que pour ceux destinés aux réglages des paramètres : $V_{max} = 4$, $V_{min} = 2$ et les résultats présentés sont une moyenne obtenue sur 20 exécutions du programme. Les graphes représentent chacun l'évolution d'un des critères pour les deux solutions "PIAF" ou "aléatoire" avec en abscisses le temps écoulé pendant la simulation et en ordonnée la valeur du critère.

La condition d'arrêt de la simulation étant l'absence d'informations à faire circuler, les mesures ne se terminent pas nécessairement au même moment. C'est la raison pour laquelle les courbes représentatives de l'une ou l'autre des solutions ne sont parfois pas définies sur le même intervale.

La figure 6.11 contient les deux premières séries de tests concernant les réseaux `Random40` et `SmallWorld40`. Le jeu de données utilisé est $dataset_{bruit}$. Nous nous plaçons ainsi dans des conditions probables pour un scénario d'utilisation de PIAF. Un réseau de 40 personnes qui se connaissent soit par le biais de relations entre eux (`SmallWorld40`) ou sans aucune relation initiale (`Random40`) et qui sont amenées à faire circuler des informations dont les sujets peuvent se chevaucher ($dataset_{bruit}$). En comparant les figures 6.11(a) et 6.11(a) nous remarquons que le contenu initial du carnet d'adresses n'influe pas sur les résultats de rappel. Il apparaît également que PIAF permet de récupérer plus rapidement et en plus grande quantité des informations intéressantes. Les figures 6.11(c) et 6.11(d) attestent que cette récupération plus importante ne s'effectue pas au détriement de la précision globale du stock récuperé. Sur ces graphes également, il apparaît que les résultats ne sont pas sensibles au changement de réseau initial ce qui tend à prouver l'efficacité de l'algorithme de modification du réseau. Dans les deux cas, celui-ci adapte les connexions de façon à obtenir les mêmes performances au niveau du rappel et de la précision. La comparaison des graphes 6.11(e) et 6.11(f) montre que cet ajustement conduit à la mise en place de deux topologies différentes. En ne tenant pas compte de la valeur singulière au alentours du temps 2800 sur la figure 6.11(e), on remarque si la diffusion est faite en utilisant les informations laissées par les phéromones un taux de groupement stable de 35% est rapidement atteint. A l'opposé, lorsque la diffusion est aléatoire les évaluations des fourmis sont faussées et conduisent à la création de topologies de moindre qualité.

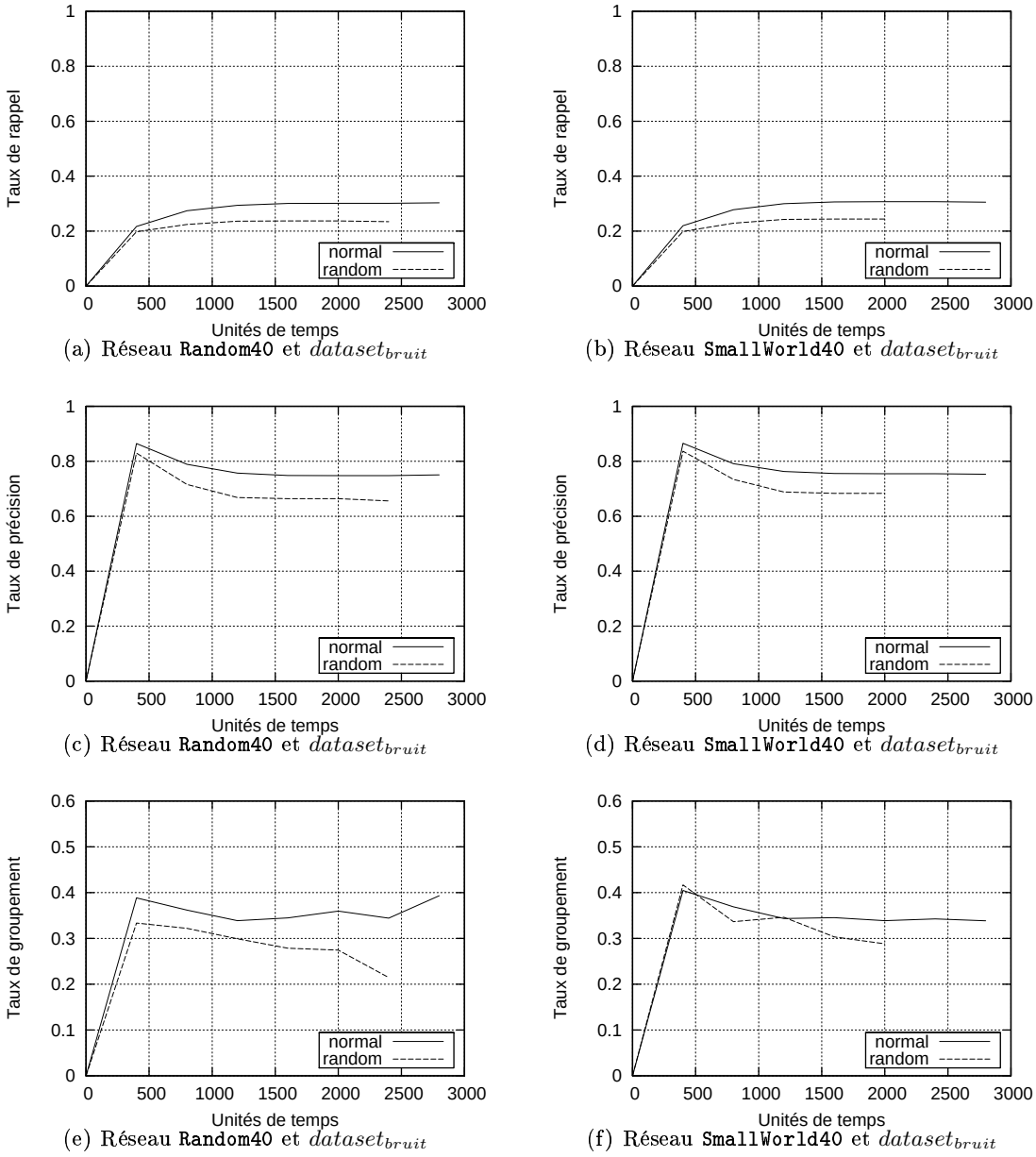
Ces travaux ont fait l'objet d'une publication dans « Multiagent and Grid Systems - An International Journal »[Guéret 07]. Les tests qui y sont présentés ont été effectués avec un réseau initial et une base de données d'informations différentes de ceux présentés ici. Le réseau utilisé est constitué de 20 pairs divisés en 4 groupe de 5 ayant un même centre d'intérêt. Les caractéristiques des informations échangées sont les suivantes :

	D_1	D_2	D_3	D_4
D_1	0.74	0.25	0.08	0.08
D_2	0.25	0.74	0.11	0.17
D_3	0.08	0.11	0.75	0.07
D_4	0.08	0.17	0.07	0.75

Les résultats sont rapportés dans la figure 6.12. Les conclusions que nous pouvons en tirer sont les mêmes que pour les deux cas que nous venons de voir dans la figure 6.11 avec cependant un écart plus marqué entre les deux solutions testées. Ces différences peuvent s'expliquer par la différence de la taille de réseau ainsi que le contenu du jeu de données. En particulier, le nombre plus faible de pairs permet d'atteindre un plus fort taux de groupement.

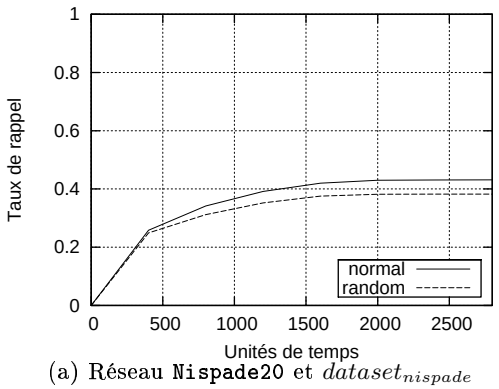
6.6 Autres applications de PIAF

Nous nous sommes également interessé à d'autres applications possibles des algorithmes de diffusion d'information de PIAF.Nous terminerons ce chapitre par quelques mots sur les pro-

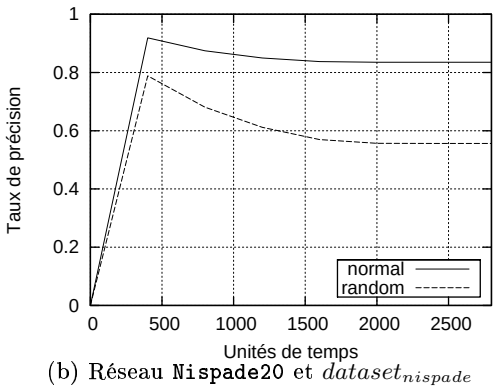


(a) Réseau `Random40` et $dataset_{bruit}$
(b) Réseau `SmallWorld40` et $dataset_{bruit}$
(c) Réseau `Random40` et $dataset_{bruit}$
(d) Réseau `SmallWorld40` et $dataset_{bruit}$
(e) Réseau `Random40` et $dataset_{bruit}$
(f) Réseau `SmallWorld40` et $dataset_{bruit}$

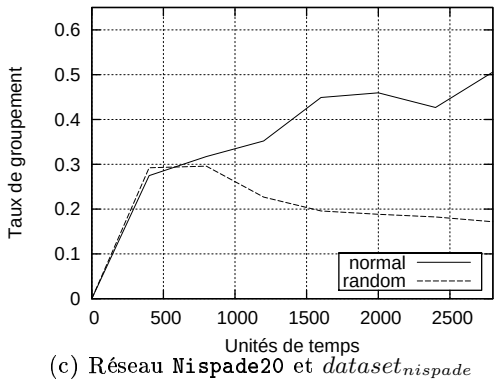
FIG. 6.11 – Comparaison avec une diffusion aléatoire



(a) Réseau `Nispade20` et $dataset_{nispade}$



(b) Réseau `Nispade20` et $dataset_{nispade}$



(c) Réseau `Nispade20` et $dataset_{nispade}$

FIG. 6.12 – Comparaison avec une diffusion aléatoire

blèmes de classification non supervisée et d'équilibrage de charges de calcul dans un réseau de machines.

6.6.1 Classification non supervisée

La création des groupes dans le réseau peut s'apparenter à une action de classification non supervisée. C'est en partant de ce constat que nous est venu l'idée d'étudier l'utilisation de PIAF pour ce genre de problème.

Dans ce contexte, il n'est plus question de faire circuler des informations entre un certain nombre d'utilisateurs mais il s'agit de regrouper un certain nombre d'informations selon des thématiques proches. Certaines adaptations conceptuelles ainsi qu'algorithmiques ont dû être apportées à PIAF afin de prendre en compte cette différence d'approche :

- les *bot* sont des fournisseurs qui alimentent le classifieur d'informations puisées dans la base de données à classifier. Leurs publications ne traduisent plus de centre d'intérêt mais une certaine proximité géographique entre les éléments générés (voir détails ci-après) ;
- les groupements effectués par le gestionnaire de connexions traduisent des liens de proximité entre les publications effectuées par les bot. Une connexion entre deux pairs traduit donc le fait qu'ils publient des informations provenant d'une même zone dans l'espace de données.

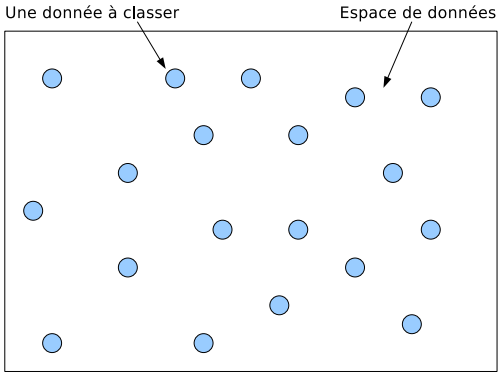
Le fonctionnement des bot dans ces conditions est représenté par la figure 6.13 et indiqué dans l'algorithme 4.

Algorithme 4 : Activité du `Bot`

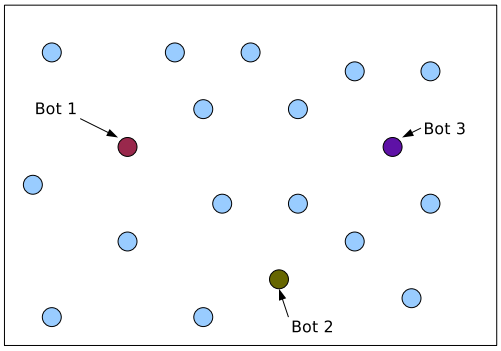
```
1  if I_0 non défini then
2  |   /* Choisir un centre d'intérêt au hasard */
3  |   I ← PickRandomInformation()
4  |   I_0 ← I
5  end
6  else
7  |   I ← PickCloseInformation(I_0)
8  end
9  i ← 0
10 while i ≠ nbDuplicats do
11 |   Envoyer une copie de I au DataManager
12 |   i ← i + 1
13 end
```

Pour être en adéquation avec ce fonctionnement, la mesure de similarité jusqu'à lors utilisée a été remplacée par une mesure de distance euclidienne. Contrairement au cosinus qui permet de mesurer un espace angulaire, cette distance permet d'évaluer la distance séparant deux éléments. Pour deux vecteurs $A = \{a_i\} \in \mathbb{R}^n$ et $B = \{b_i\} \in \mathbb{R}^n$, celle-ci se formule ainsi :

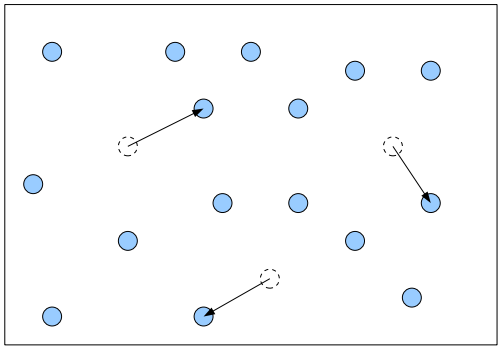
$$Distance(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \tag{6.4}$$



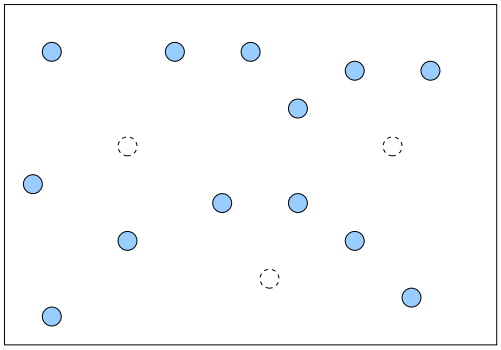
(a) L'espace composé des informations à classer



(b) Chaque bot choisi une information au hasard et la publie



(c) La seconde information publiée et la plus proche du point central du bot



(d) Les données publiées sont enlevées de l'espace. Celui-ci est re-rempli une fois vidé

FIG. 6.13 – Méthode d'attribution des informations pour la classification

Le critère de décision des fourmis a lui aussi dû être adapté afin de pouvoir prendre en compte la nature non bornée de la distance. En effet, alors que $Sim(A,B) \in [0,1]$, $Distance(A,B) \in [0, D_{max}]$ avec D_{max} la plus grande distance observable dans l'espace de données.

$$V_i(I,t) = \{n_j \in V_i(t) \mid s(\tau_{i \leftarrow j}(t), \tau(I)) \leq (1 - s_{min}) * D_{max}\} \quad (6.5)$$

$$\overline{V_i(I,t)} = \{n_j \in V_i(t) \mid n_j \notin V_i(I,t)\} \quad (6.6)$$

Pour qu'un pair soit intéressé il faut alors que la trace de phéromones associée à sa connexion se trouve dans un cercle de rayon $(1 - s_{min}) * D_{max}$ centré sur celle de l'information à déplacer (voir figure 6.14. Si $s_{min} = 0$ la condition sera que la distance soit inférieure ou égale à D_{max} et donc sera toujours vérifiée. Nous préservons ainsila signification de s_{min}.

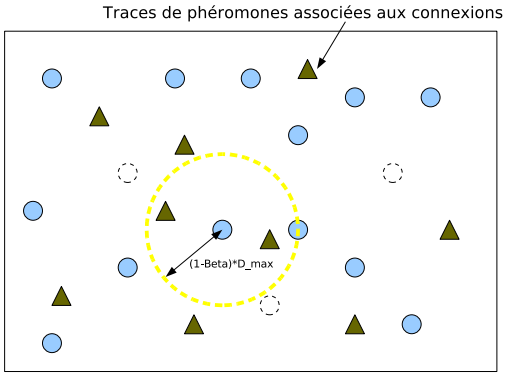


FIG. 6.14 – Exemple de selection d'une destination

Une série initiale de test a été effectuée sur la, très classique, base de donnée « Iris ». Celle-ci se compose de 150 entrées, réparties de façon égale en 3 classes. Les figures 6.15, 6.16 et 6.17 montrent le graphe obtenu après stabilisation du système. Chaque couleur correspond à un centre d'intérêt différent du nœud et donc indique une des trois classes d'iris. La nature préliminaire de ces résultats ne permet pas de conclure sur l'effacité ou non de la méthode mais l'on remarque néanmoins que des regroupements de couleurs similaires apparaissent sur ces 3 graphes. Le dernier, 6.17, montre en particulier un cas extrême ou un sous graphe de nœuds intéressés par le même sujet a pu être isolé du reste du réseau.

La base Iris se caractérise par l'enchevêtrement de deux de ses classes. Alors que l'une d'entre elle est parfaitement séparée, les deux autres ont plusieurs points communs qui peuvent mettre en difficulté les algorithmes de classification. D'après les quelques graphes ici rapportés, PIAFs semble pouvoir être utilisé pour détecter visuellement de tels cas en analysant les liens entre les nœuds dans le graphe produit. De plus amples tests seront nécessaire pour infirmer ou confirmer cette première impression.

Cette utilisation de PIAF présente certaines similitudes avec la méthode de partitionnement de graphe coloré proposée par Dutot [Dutot 05]. La principale différence se situe au niveau des phéromones déposées sur les connexions reliant deux nœuds. Dans notre cas, les phéromones sont une information représentative de l'ensemble des informations ayant été transportées sur le chemin lors de transferts antérieurs. Ce ne sont pas une liste de quantités de présence d'une

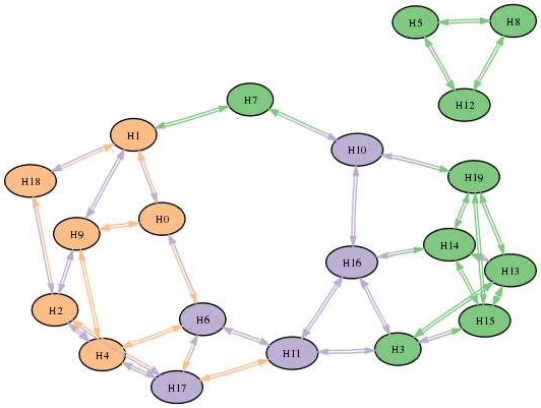


FIG. 6.15 – Exemple de résultat de classification de la base Iris (1)

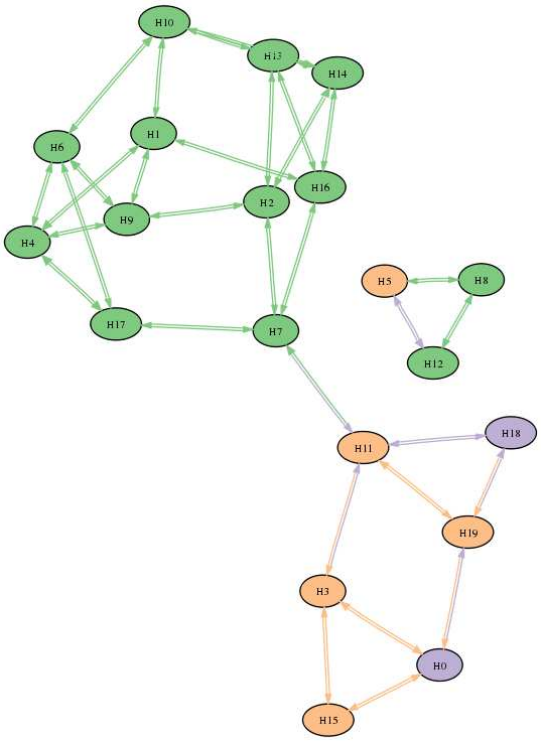


FIG. 6.16 – Exemple de résultat de classification de la base Iris (2)

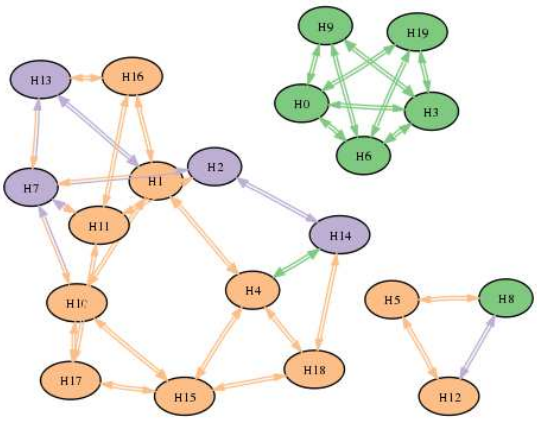


FIG. 6.17 – Exemple de résultat de classification de la base Iris (3)

classe qui y sont indiqué mais une valeur moyenne résultant des multiples translations vectorielles effectuées dans l'espace de représentation des phéromones. Ceci autorise l'utilisation de la trace de phéromone comme ce que nous pourrions appeler un « profil de région » qui, à l'opposé des profils habituels définis pour un nœuds, est représentatif d'un groupement de nœuds du graphe.

6.6.2 Répartition de charge

La répartition de charges est la dernière application que nous avons envisagé pour notre algorithme. Ce problème consiste à répartir un certain nombres de tâches à effectuer parmi un certain nombre de machines. Les critères de répartition généralement pris en compte sont l'équitabilité (toutes les machines ont autant de travail), la proximité des machines (transmettre les tâches peut coûter cher en terme de bande passante réseau) et les capacités calculatoires (plus une machine est puissante, plus elle a de travail).

Bien que PIAF n'ait pas été conçu pour résoudre les problèmes d'affectation de tâches, nous pensons que son utilisation peut apporter une autre vision du problème en introduisant un nouveau critère : les centres d'intérêt des utilisateurs des machines. Pour modéliser ces centres d'intérêt nous n'avons, jusqu'à présent, consideré que les documents consultés par un utilisateur. Or, ormis les tâches relatives au traitement de l'information (en tant que donnée informative), une machine peut être aussi utilisée pour traiter différents problèmes. Prenons par exemple le cas d'un utilisateur qui consulte une fois une page Web traitant des algorithmes de k-moyenne puis implémente un algorithme pour les tester. Sa consultation ne générera qu'une unique information, ce qui peut être insuffisant pour permettre au système de construire les groupements, alors que l'exécution répétée de cette algorithme sur sa machine démontrera l'intéret de l'utilisateur pour cette méthode.

En traitant les paquets d'intructions à traiter de la même manière que le reste des informations échangées. PIAF peut faciliter la mise en relation des personnes qui consultent la même documentation ou utilisent des outils mathématiques similaires. L'efficacité de l'outil du point de vue des performance dans le cadre du calcul disribué seraient très vraisemblablement moindres que des outils dédiés à cette tâche. Afin de garder de bonnes performances dans ce domaine, une des pistes à explorer pourrait être l'intégration de PIAF dans un de ces outils et le décharger

ainsi des tâches relatives au calcul.

Cette idée a fait l'objet de la publication d'un rapport interne de 27 pages intitulé « Grid Computing et fourmis artificielles dans un réseau P2P : Application à l'architecture PIAF » [Guéret 04].

6.7 Conclusion

Nous avons présenté dans ce chapitre les deux implémentations qui ont été faites d'une de PIAF. Après une première production ayant finalement pris des allures de « faux départ », une seconde réalisation nous a permi de tester les algorithmes dans de bonnes conditions.

Ces tests, menés sur diverses bases pour diverses topologies initiales et sur différentes tailles du réseau on prouvé l'efficacité de la méthode pour la diffusion « à l'aveugle » (ie sans définition de profil) d'informations dans un réseau P2P.

D'autres applications pourraient être trouvées pour PIAF. En particulier pour les algorithmes relatifs à la circulation des messages et la modification de la topologie du réseau. Quelques résultats préliminaires concernant un problème de classification non supervisée nous conforte dans cette idée bien que de plus amples tests seraient nécessaires avant de pouvoir conclure.

Les résultats de ces travaux ont fait l'objet de publications dans les congrès :
***ROADEF** [Guéret 05], **ANTS** [Guéret 06a], **I2CS** [Guéret 06b] et **NIDISC** [Guéret 06c]*

Un article de revue est également attendu à paraître pour l'année 2007 [Guéret 07]

Conclusion

Ce qu'il faut retenir

Dans ce travail de thèse, nous présentons des travaux s'intéressant à la possibilité de rendre un navigateur Internet « Intelligent ». La fonctionnalité principale d'un navigateur est d'interpréter le code d'une page écrite dans un language de balisage hypertexte (HTML) afin d'en produire un affichage graphique. Ces pages sont fournies par des serveurs dont le navigateur est client. La plus importante concentration de ces serveurs est sans aucun doute le réseau du World Wide Web, accessible depuis Internet, dont la taille augmente d'années en années.

Un scénario typique d'utilisation du Web nous a permi l'identification des différents acteurs : les utilisateurs, leurs données personnelles, le réseau reliant les machines ainsi que le navigateur Internet. Vis à vis de ces divers éléments, un utilisateur sera amené à avoir quatre activités différentes. 1) En consultant les ressources disponibles sur le Web, il peut conduire un processus de recherche visant à répondre à un besoin précis et ponctuel. 2) Dans une optique à plus long terme visant à trouver des informations correspondant à ses centres d'intérêts, 3) cette recherche se changera en navigation alors qu'il va de page en page sans objectif précis. 4) Les utilisateurs sont également amenés à gérer la masse d'informations récupérées qu'ils pourront, s'ils le souhaitent, être amenés à vouloir partager avec d'autres utilisateurs. Pour toutes ces tâches, le niveau d'assistance fourni par le navigateur témoignera de son intelligence. Dans la littérature, les premiers travaux menés dans ce sens ont conduit à la mise au point de divers outils proposant la construction de cartes de navigation, offrant la possibilité d'annoter les pages, capables de formuler diverses recommandations ou de modifier le contenu des pages affichées (voir chapitre 1).

Le Web est, depuis son invention dans les années 1990, assimilable à une gigantesque source d'informations qu'une partie des utilisateurs publie (les webmestres) et que l'autre partie consulte (les internautes). Après 15 ans d'existence, ce schéma est en train d'évoluer. Le web de l'accès aux documents laisse place à un Web axé sur les services : la démarche de consultation du web n'est plus axée sur la recherche de documents mais sur le problème à résoudre. Les relations client/-serveur qui existaient entre les webmestres et les internautes sont remplacées et complétées par des modèles sociaux plus évoluées autorisant une plus grande lattitude d'intéractions et de communication. L'apparition du P2P, modèle dans lequel n'importe quelle machine peut jouer à la fois le rôle de serveur et de client, remet en question la place des serveurs de contenu centralisés. Bien qu'ils soient plus conceptuels que technologiques, ces changements ont été identifiés comme les signes de l'apparition d'un « Web 2.0 » (également appelé « Web communautaire »). La recherche sur la mise au point d'outils d'assistance doivent évoluer en conséquence et s'identifient désormais sous la thématique de la « Web Intelligence ». Des projets tels que le Web sage

(« Wisdom web ») capable de comprendre les besoins de l'utilisateur afin de l'aider à y répondre ou le bureau sémantique social facilitant le partage d'informations sont les premiers signes des nouvelles façons d'apprécier l'intelligence d'un navigateur (chapitre 2).

Nous nous sommes intéressés à la communication et l'échange de ressources entre utilisateurs. La direction que nous avons exploré concerne la mise en place d'un service polyvalent de circulation d'informations transparent pour l'utilisateur, autonome et non intrusif. L'objectif étant la mise au point d'une architecture favorisant le partage des ressources et la découverte de relations sociales au sein d'un réseau d'utilisateurs tout en ne nécessitant aucune implication active de ces derniers. Notre travail s'est décomposé en deux parties, la première axée sur la circulation de contenu et la mise en place du réseau social, la seconde s'intéressant au dialogue Homme-Machine avec l'utilisateur. L'architecture proposée pour la circulation des données s'inspire des principes issus de l'étude des systèmes immunitaires et se base sur l'utilisation d'un modèle de fourmis artificielles. Les traces de phéromones laissées sur les chemins menant d'un pair à un autre sont utilisées à la place des profils.

Les algorithmes destinés à la circulation d'informations et la détection de réseaux sociaux ont été implémentés sous la forme d'un produit fini utilisable puis dans un simulateur afin d'en vérifier les performances. Une série de tests effectués sur plusieurs jeux de données et plusieurs types de réseaux nous a permi de mettre en évidence l'efficacité de la diffusion et du processus de modification dynamique du réseau.

Perspectives

Les perspectives pour ce travail sont nombreuses. Deux directions principales peuvent être envisagées selon que l'on considère le développement et l'élargissement du cadre d'application de la plateforme ou la création de nouveaux services exploitant le système d'information.

La principale contrainte imposée par l'outil est le nombre de paramètres à régler. Les développement futurs de l'algorithme de circulation de l'informations pourraient se concentrer sur la réduction de ces paramètres à un minimum dont le rôle serait facilement interprétable par l'utilisateur. La recherche de liens entre ces différents paramètres est une première piste qu'il serait intéressant de poursuivre, la mise en place de paramètrages dynamiques s'adaptant à l'évolution de l'environnement en est une autre qu'il reste à explorer.

En faisant abstraction de leur cadre initial de mise au point, on remarque que les algorithmes de circulation de données et de création de liens entre les nœuds s'apparentent à un processus de classification. Partant de ce constat, un travail mené sur l'utilisation de PIAF pour les problèmes de classification non supervisée donne de premiers résultats intéressants. L'application à la circulation des paquets de données dans les réseaux de calcul distribué est également à l'étude. Le soucis de généricité poursuivi lors de la conception du système en autorise diverses adaptations dont plusieurs restent encore à définir.

Le résultat de ce travail de thèse est la proposition d'une plateforme composée de deux niveaux. Alors que le premier niveau a été implémenté et ses performances validées par expérimentations, le second, essentiellement constitués d'outils de communication avec l'utilisateur, a été moins avancé. Une poursuite naturelle de ce travail serait le développement des outils de ce second niveau.

Bibliographie

[Aickelin 04] Uwe Aickelin. *Artificial Immune Systems (AIS) – A New Paradigm for Heuristic Decision Making ?* OR46 (2004) AIS for OR, Keynote Speech, 2004.

[Albert 02] Reka Albert & Albert-Laszlo Barabasi. *Statistical mechanics of complex networks.* REVIEWS OF MODERN PHYSICS, vol. 74, pages 47–97, January 2002.

[Allen 94] Linda J.S. Allen. *Some discrete-time SI, SIR, and SIS epidemic models.* Mathematical Bioscience, vol. 124, no. 1, pages 83–105, 1994.

[Arnold 99] David Arnold, Bill Segall, Julian Boot, Andy Bond, Melfyn Lloyd & Simon Kaplan. *Discourse with Disposable Computers : How and Why You Will Talk to Your Tomatoes.* In Proceedings of the Usenix Workshop on Embedded Systems, pages 9–22, March 1999.

[Aupetit 05] Sébastien Aupetit. *Contributions aux modèles de Markov cachés : métaheuristiques d'apprentissage, nouveaux modèles et visualisation de dissimilarité.* Thèse de doctorat, Université François-Rabelais de Tours, Laboratoire d'Informatique, 30 novembre 2005. 239 pages.

[Babaoglu 01] Ozalp Babaoglu, Hein Meling & Alberto Montresor. *Anthill : A Framework for the Development of Agent-Based Peer-to-Peer Systems.* Rapport technique UBLCS-2001-09, University of Bologna, Italy, November 2001.

[Babaoglu 02] Ozalp Babaoglu, Hein Meling & Alberto Montresor. *Anthill : A Framework for the Development of Agent-Based Peer-to-Peer Systems.* In Proceedings of the 22th International Conference on Distributed Computing Systems (ICDCS '02), Vienna, Austria, July 2002.

[Bagci 03] Faruk Bagci, Jan Petzold, Wolfgang Trumler & Theo Ungerer. *Ubiquitous Mobile Agent System in a P2P-Network.* In UbiSys-Workshop at the Fifth Annual Conference on Ubiquitous Computing, Seattle, October 12-15 2003.

[Balabanovic 97] Marko Balabanovic. *An adaptative web page recommendation service.* In To appear in First International Conference on Autonomous Agents, Marina del Rey, CA, February 1997.

[Barabasi 99] AL Barabasi & R Albert. *Emergence of scaling in random network.* Science, vol. 286, pages 509–512, 1999.

[Barrett 97] R. Barrett, P. P. Maglio & D. C. Kellem. *How to Personalize the Web.* In Proceedings of the Conference on Human-computer Interaction (CHI'97), pages 75–82. Addison-Wesley, 1997.

[Bauer 01] Travis Bauer & David Leake. *A Research Agent Architecture for Real Time Data Collection and Analysis.* In Proceedings of the Workshop on Infrastructure for Agents, MAS, and Scalable MAS, 2001.

[Bergenti 04] Federico Bergenti, Marco Mari & Mercedes Garijo. *Collaborator - Enabling Enterprise Collaboration through Agents.* In Proceedings of the 2nd International Workshop on Agent-based Computing for Enterprise Collaboration (WETICE-2004), pages 41–46, University of Modena, Italy, June 14-16 2004. IEEE.

[Berners-Lee 01] Tim Berners-Lee, James Hendler & Ora Lassila. *The Semantic Web.* Scientific American, vol. 284, no. 5, pages 34–43, May 2001.

[Bloom 70] Burton H. Bloom. *Space/time trade-offs in hash coding with allowable errors.* Commun. ACM, vol. 13, no. 7, pages 422–426, July 1970.

[Bonabeau 99] E. Bonabeau, M. Dorigo & G. Heraulaz. Swarm intellgence : From natural to artificial systems. Oxford University Press, 1999.

[Bonsma 02] E.R. Bonsma. *Fully decentralised, scalable look-up in a network of peers using small world networks.* In Proceedings of Sixth Multi. Conf. On Systemics, Cybernetics and Informatics, Orlando, July 2002.

[Bouras 01] Christos Bouras, Agisilaos Konidaris, Eleni Konidari & Afrodite Sevasti. *Introducing Navigation Graphs as a Technique for Improving WWW User Browsing.* In Advances in Web-Age Information Management (WAIM), pages 249–256, 2001.

[Bouvin 99] Niels Olof Bouvin. *Unifying Strategies for Web Augmentation.* In Hypertext'99, 1999.

[Brin 98] Sergey Brin & Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* Computer Networks and ISDN Systems, vol. 30, no. 1-7, pages 107–117, 1998.

[Burmester 06] Mike Burmester, Tri Van Le & Alec Yasinsac. *Adaptive gossip protocols : Managing security and redundancy in dense ad hoc networks.* Journal of Ad Hoc Networks, vol. 4, no. 3, pages 504–515, 2006.

[Busetta 01] P. Busetta, L. Serafini, D. Singh & F. Zini. *Extending Multi-Agent Cooperation by Overhearing.* Rapport technique 0101-01, Istituo Trentino di Cultura, January 2001.

[Bush 45] Vannevar Bush. *As We May Think.* The atlantic monthly, vol. 176, no. 1, pages 101–108, 1945.

[Bélisle 99a] Claire Bélisle. *La navigation hypermedia : un défi pour la formation à distance.* Journal of Distance Education/Revue de l'enseignement à distance, 1999.

[Bélisle 99b] Claire Bélisle, R Zeiliger & T Cerratto. *S'orienter sur le Web en construisant des cartes interactives : le navigateur NESTOR.* In Balpe, Natkin, Lelu & Saley, editeurs, Hypertextes, hypermedias et internet H2PTM'99, pages 101–117. Hermes Science, 1999.

[Camorlinga 03] Sergio Camorlinga & Ken Barker. *Multiagent Systems Storage Resource Allocation in a Peer-to-Peer Distributed File System.* TechnicalReport 2003-716-19, Department of Computer Science, The University of Calgary, January, 28 2003.

[Camorlinga 04] Sergio Camorlinga, Ken Barker & John Anderson. *Multiagent Systems for Resource Allocation in Peer-to-Peer Systems.* In Proceedings of the winter international synposium on Information and communication technologies, SESSION : Distributed and Internet computing, pages 1–6, Cancun, Mexico, 2004.

[Caro 98] Gianni Di Caro & Marco Dogiro. *AntNet : Distributed Stigmergetic Control for Communications Networks.* Journal of Artificial Intelligence Research, vol. 9, pages 317–365, 1998.

[Carzaniga 00] A. Carzaniga, D.S. Rosenblum & A.L. Wolf. *Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service.* In Nineteenth ACM Symposium on Principles of Distributed Computing (PODC2000), Portland, Oregon, July 2000.

[Carzaniga 01] A. Carzaniga, D. S. Rosenblum & A. L. Wolf. *Design and Evaluation of a Wide-area Event Notification Service.* ACM Trans. on Computer Systems, vol. 19, no. 3, page 332–383, August 2001.

[Chen 98] Liren Chen & Katia Sycara. *WebMate : A Personal Agent for Browsing and Searching.* In Katia P. Sycara & Michael Wooldridge, editeurs, Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98), pages 132–139, New York, 9–13, 1998. ACM Press.

[Consortium 06] Internet Software Consortium. *ISC Internet Domain Survey.* `http://www.isc.org/index.pl?/sw/bind/`, Janvier 2006.

[Cuenca-Acuna 02] F. M. Cuenca-Acuna, R. P. Martin & T. D. Nguyen. *PlanetP : Using Gossiping and Random Replication to Support Reliable Peer-to-Peer Content Search and Retrieval.* Rapport technique DCS-TR-494, Department of Computer Science, Rutgers University, July 2002.

[Cuenca-Acuna 03] F. M. Cuenca-Acuna, C. Peery, R. P. Martin & T. D. Nguyen. *PlanetP : Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities.* In Proceedings of the 12th International Symposium on High Performance Distributed Computing (HPDC), June 2003.

[Datta 04] Anwitaman Datta, Silvia Quarteroni & Karl Aberer. *Autonomous Gossiping : A Self-Organizing Epidemic Algorithm for Selective Information Dissemination in Wireless Mobile Ad-Hoc Networks.* LNCS, vol. 3226, pages 126–143, 2004.

[de Ipiña 01] Diego López de Ipiña & Eleftheria Katsiri. *An ECA Rule-Matching Service for Simpler Development of Reactive Applications.* Published as a supplement to the Proceedings of Middleware 2001 at IEEE Distributed Systems Online, vol. 2, no. 7, November 2001.

[Decker 04] Stefan Decker & Martin Frank. *The Social Semantic Desktop.* Rapport technique 2004-05-02, DERI - Digital Enterprise Research Institute, May 2004.

[Demers 88] Alan J. Demers, Daniel H. Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard E. Sturgis, Daniel C. Swinehart & Douglas B. Terry. *Epidemic Algorithms for Replicated Database Maintenance.* Operating Systems Review, vol. 22, no. 1, pages 8–32, 1988.

[Di Caro 05] Gianni Di Caro, Frederick Ducatelle & Luca M. Gambardella. *Project description : BISON : Biology-Inspired techniques for Self-Organization in dynamic Networks.* Zeitschrift Knstliche Intelligenz, *Special Issue on Swarm Intelligence*, vol. 4, page 4, November 2005.

[Dorigo 96a] Marco Dorigo & Luca Maria Gambardella. *Ant Colonies for the Traveling Salesman Problem.* Biosystems, vol. 43, pages 73–81, 1996.

[Dorigo 96b] Marco Dorigo, Vittorio Maniezzo & Alberto Colorni. *The Ant System : Optimization by a colony of cooperating agents.* IEEE Transactions on Systems, Man, and Cybernetics Part B : Cybernetics, vol. 26, no. 1, pages 29–41, 1996.

[Dressler 06] Falko Dressler. *Self-Organization in Ad Hoc Networks : Overview and Classification.* Technical Report 02/06, University of Erlangen, Departement of computer science 7, 2006.

[Drogoul 02] Alexis Drogoul. *Pervasive Intelligence.* In Proceedings of the 1st Workshop on Radical Agent Concepts (WRAC'02), Orlando, US, 2002. NASA.

[Dutot 05] Antoine Dutot. *Distribution dynamique adaptative à l'aide de mécanismes d'intelligence collective.* Thèse de doctorat, Université du Havre, 2005.

[Eagle 02] Nathan Eagle & Alex Pentland. *Collaboration from Conversation.* In Proceedings of the Conference on Computer Supported Cooperative Work (CSCW'02), New Orleans, Louisiana, November 2002.

[Eberhart 95] R.C. Eberhart & J. Kennedy. *A new optimizer using particle swarm theory.* In Proceedings of the sixth international symposium on micro machine and human science, pages 39–43, IEEE service center, Piscataway, NJ, Nagoya, Japan, 1995.

[Franklin 98] M.J. Franklin & S.B. Zdonik. *"Data in Your Face" : Push Technology in Perspective.* In Proceedings ACM SIGMOD International Conference on Management of Data, pages 516–519, 1998.

[Ganguly 04] Niloy Ganguly, Geoff Canright & Andreas Deutsch. *Design Of An Efficient Search Algorithm For P2P Networks Using Concepts From Natural Immune Systems.* In Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII), Birmingham, UK, 18-22 September 2004.

[Grudin 88] Jonathan Grudin. *Why CSCW applications fail : problems in the design and evaluation of organization of organizational interfaces.* In Proceedings of the 1988 ACM conference on Computer-supported cooperative work, pages 85–93, Portland, Oregon, United States, 1988.

[Gutnik 04] Gery Gutnik & Gal Kaminka. *Towards a formal approach to overhearing : algorithms for conversation identification.* In AAMAS'04, New York, USA, July 19-23 2004.

[Guzdial 97] M. Guzdial. *Technological support for an apprenticeship in object-oriented design and programming.* In Proceedings of the OOPSLA'97 Educators Symposium, 1997.

[Guéret 04] Christophe Guéret, Nicolas Monmarché & Mohamed Slimane. *Grid Computing et fourmis artificielles dans un réseau P2P : Application à l'architecture PIAF.* Rapport technique 277, Laboratoire d'Informatique de l'université de Tours (EA 2101), Tours, 7 Octobre 2004.

[Guéret 05] C. Guéret, N. Monmarché & M. Slimane. *Aide à la navigation sur Internet : utilisation de fourmis artificielles pour l'échange d'informations dans un réseau P2P.* In Actes du congré de la ROADEF, 2005.

[Guéret 06a] Christophe Guéret, Nicolas Monmarché & Mohamed Slimane. *Automonous gossiping of information in a P2P network with artificial ants.* In M. Dorigo, L.M. Gambardella, M. Birattari, A. Martinoli, R. Poli & T. Stützle, editeurs, Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006, volume 4150 of *Lecture Notes in Computer Science*, pages 388–395, Bruxelles, Belgium, September 5-7 2006. Springer-Verlag.

[Guéret 06b] Christophe Guéret, Nicolas Monmarché & Mohamed Slimane. *A self-organizing ant-based information gossiping algorithm for P2P networks.* In 6th International Workshop on Innovative Internet Community Systems (I2CS'2006), Lecture Notes in Computer Science, Neuchâtel, Switzerland, June 26-28 2006. Springer-Verlag. to appear.

[Guéret 06c] Christophe Guéret, Nicolas Monmarché & Mohamed Slimane. *Sharing Resources with Artificial Ants.* In Proceedings of the 9th International Workshop on Nature Inspired Distributed Computing (NIDISC'06), Rhodes Island, Greece, April, 25-29 2006. 8 pages CD-ROM.

[Guéret 07] Christophe Guéret, Nicolas Monmarché & Mohamed Slimane. *A biology-inspired model for the automatic dissemination of informations in P2P networks.* Nature-Inspired Systems for Parallel, Asynchronous and Decentralised Environments, special issue of Multiagent and Grid Systems - An International Journal, vol. 3, no. 1, 2007. To appear.

[Hethcote 89a] Herbert W. Hethcote. *Three Basic Epidemiological Models.* Biomathematics, vol. 18, pages 119–142, 1989.

[Hethcote 89b] H.W. Hethcote & . S.A. Levin. *Periodicity in epidemiological models.* Applied Mathematical Ecology, vol. 1, pages 193–211, 1989.

[Hethcote 91] H.W. Hethcote & P. van den Driessche. *Some epidemiological models with nonlinear incidence.* Journal of Mathematical Biology, vol. 29, pages 271–287, 1991.

[Hethcote 95] Herbert W. Hethcote & P. van den Driessche. *An SIS epidemic model with variable population size and a delay.* Journal of Mathematical Biology, vol. 34, pages 177–194, 1995.

[Hui 04] Ken Y. K. Hui, John C. S. Lui & David K. Y. Yau. *Small World Overlay P2P Networks.* In IEEE International Workshop on Quality of Service (IWQoS), Montreal, Canada, June 2004.

[Hull 96] David Hull. *Stemming algorithms - a case study for detailed evaluation.* Journal of the American Society for Information Science, vol. 47, no. 1, pages 70–84, 1996.

[Iaminitchi 02] Adriana Iaminitchi, Matei Ripeanu & Ian T. Foster. *Locating Data in (Small-worlds ?) Peer-to-Peer Scientific Collaborations.* Lecture Notes In Computer Science, Revised Papers from the First International Workshop on Peer-to-Peer Systems, vol. 2429, pages 232 – 241, 2002.

[Iaminitchi 04] Adriana Iaminitchi, Matei Ripeanu & Ian Foster. *Small-world file-sharing communities.* In Proceedings of the 23rd Conference of the IEEE Communications Society (Infocom 2004), 2004.

[Intanagonwiwat 00] Chalermek Intanagonwiwat, Ramesh Govindan & Deborah Estrin. *Directed diffusion : A scalable and robust communication paradigm for. sensor networks.* In Proceedings of 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCOM'00), pages 56–67, Boston, MA, USA, August 2000.

[IonStoica 01] IonStoica, Robert Morris, David Karger, M. Frans Kaashoe & Hari Balakrishnan. *Chord : A Scalable Peer-to-peer Lookup Service for Internet Applications.* In SIGCOMM'01, August 27-31 2001.

[Jelasity 02] Mark Jelasity & Maarten van Steen. *Large-Scale Newscast Computing on the Internet.* Rapport technique IR-503, Department of Computer Science, Vrije Universiteit, Amsterdam, Netherland, October 2002.

[Jelasity 04] Márk Jelasity & Ozalp Babaoglu. *T-Man : Fast Gossip-based Construction of Large-ScaleOverlay Topologies.* Rapport technique UBLCS-2004-7, University of Bologna, Department of Computer Science, Bologna, Italy, May 2004. `http://www.cs.unibo.it/techreports/2004/2004-07.pdf`.

[Jelasity 05] M. Jelasity & O. Babaoglu. *T-man : Gossip-based overlay topology management.* In S. Brueckner, G. Di Marzo Serugendo, D. Hales & F. Zambonelli, editeurs, Engineering Self-Organising Applications (ESOA'05), Utrecht, The Netherlands, July 2005.

[Joachims 97] Thorsten Joachims, Dayne Freitag & Tom Mitchell. *WebWatcher : A tour guide for the world wide web.* In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'97), pages 770–775. Morgan Kaufmann, 1997.

[Joseph 03] Sam Joseph & Takashige Hoshiai. *Decentralized Meta-Data Strategies : Effective Peer-to-Peer Search.* IEICE Trans. Commun., vol. E86-B, no. 6, pages 1740–1753, June 2003.

[Kautz 96] Henry Kautz, Bart Selman & Al Milewski. *Agent Amplified Communication.* In Proceedings of the Thirteenth National on Artificial Intelligence (AAAI-96), Portland, OR, 1996.

[Kautz 97] Henry Kautz, Bart Selman & Mehul Shah. *Referral Web : combining social networks and collaborative filtering.* Communications of the ACM, vol. 40, no. 3, pages 63 – 65, March 1997. ISSN :0001-0782.

[Kermarrec 00] A.-M. Kermarrec, L. Massoulie & A.J. Ganesh. *Reliable Probabilistic Communication in Large-Scale Information Dissemination Systems.* Rapport technique MSR-TR-2000-105, Microsoft Research, One Microsoft Way, Redmond, WA 98052, October 2000. 20 p.

[Khelil 02] Abdelmajid Khelil, Christian Becker, Jing Tian & Kurt Rothermel. *An epidemic model for information diffusion in manets.* In Proceedings of the 5th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems, page 54–60. ACM Press, 2002.

[Koubarakis 03] Manolis Koubarakis, Christos Tryfonopoulos, Stratos Idreos & Yannis Drougas. *Selective Information Dissemination in P2P Networks : Pro-*

blems and Solutions. SIGMOD Record, Special Issue on Peer-to-Peer Data Management, vol. 32, no. 3, pages 71–76, 2003.

[Koubarakis 04] M. Koubarakis & C. Tryfonopoulos. *Distributed Resource Sharing using Self-Organized Peer-to-Peer Networks and Languages from Information Retrieval.* In Invitational Workshop on Self-* Properties in Complex Information Systems, Bertinoro, Italy, 31 May and 2 June 2004.

[Leeb 04] Minsoo Leeb, Stanley Y.W. Sua & Herman Lama. *Event and rule services for achieving a Web-based knowledge network.* Knowledge-Based Systems, vol. 17, pages 179–188, 2004.

[Leveille 02] Jasmin Leveille. *Epidemic Spreading in Technological Networks.* Rapport technique HPL-2002-287, Information Infrastructure Laboratory, HP Laboratories Bristol, October, 23 2002.

[Lieberman 95] Henry Lieberman. *Letizia : An agent that assists web browsing.* In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), pages 924–929. Morgan Kaufmann, 1995.

[Lieberman 99] Henry Lieberman, Neil W. Van Dyke & Adriana S. Vivacqua. *Let's Browse : A Collaborative Web Browsing Agent.* In Proceedings of the 4th international conference on Intelligent user interfaces (IUT'99), pages 65–68, Los Angeles, California, United States, January 05-08 1999.

[Lieberman 01] Henry Lieberman, Christopher Fry & Louis Weitzman. *Exploring the web with Reconnaissance Agents.* Communications of the ACM, vol. 44, no. 8, pages 69–75, August 2001.

[Lin 00] Meng-Jang Lin & Keith Marzullo. *Directional Gossip : Gossip in a Wide Area Network.* In Proceedings of European Dependable Computing Conference, 2000.

[Liu 05] Jiming Liu. *The Making of the Wisdom Web : Origin, Meaning, and Opportunities.* Rapport technique COMP-05-004, Departement of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong, March, 11 2005.

[Lueg 98] Christopher Lueg. *Considering Collaborative Filtering as Groupware : Experiences and Lessons Learned.* In Proceedings of the 2nd International Conference on Pratical Aspects of Knowledge Management (PAKM), 1998.

[Mizzaro 02] Stefano Mizzaro & Carlo Tasso. *Ephemeral and Persistent Personalization in Adaptive Information Access to Scholarly Publications on the Web.* In Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH), pages 306–316, 2002.

[Monmarché 00] Nicolas Monmarché. *Algorithmes de fourmis artificielles : applications à la classification et à l'optimisation.* Thése de doctorat, Université François Rabelais de Tours, Laboratoire d'Informatique, 2000.

[Montresor 01] Alberto Montresor. *Anthill : a Framework for the Design and Analysis of Peer-to-Peer Systems.* In Proceedings of the 4th European Research Seminar on Advances in Distributed Systems, Bertinoro, Italy, May 2001.

[Montresor 02a] A Montresor, H Meling & O Babaoglu. *Messor : Load-Balancing through a Swarm of Autonomous Agents.* Rapport technique UBLCS-02-08, Departement of Computer Science, University of Bologna, Bologna, Italy, May 2002.

[Montresor 02b] Alberto Montresor, Heing Meling & Ozalp Babaoglu. *Towards Self-Organizing, Self-Repairing and Resilient Peer-to-Peer Systems.* In Proceedings of the International Workshop on Peer-to-Peer Computing, Networking 2002, Italy, 2002. UBLCS-2002-09.

[Moukas 96] Alexandros Moukas & Pattie Maes. *Amalthaea : An Evolving Multi-Agent Information Filtering and Discovery System for the WWW.* In Proceedings of the Conference on Practical Applications of Agents and Multiagent Technology, 1996.

[NCSA 97] NCSA. *NCSA Mosaic.* `http://archive.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html`, 1997.

[Newman 01] M. E. J. Newman, D. J. Watts & S. H. Strogatz. *Random graph models of social networks.* In Proceedings of the National Academy of Science, 2001.

[Palau 04] Jordi Palau, Miquel Montaner & Beatriz López. *Collaboration Analysis in Recommender Systems using Social Networks.* In Eighth International Workshop on Cooperative Information Agents (CIA'04), Erfurt (Germany), September 27-29 2004.

[Parunak 05] H. Van Dyke Parunak. *Expert Assessment of Human-Human Stigmergy.* Rapport technique, Altarum Institute, 3520 Green Court, Suite 300. Ann Arbor, Michigan 48105, May, 16 2005.

[Payton 98] David W. Payton. *Discovering Collaborators by Analyzing Trails Through an Information Space.* In AAAI Fall Symposium on Artificial Intelligence and Link Analysis, October 23-25 1998.

[Pitkow 96] J. Pitkow & C. Kehoe. *Emerging Trends in the WWW User Population.* Communications of the ACM, vol. 39, no. 6, pages 106–108, 1996. GVU Technical Report : GIT-GVU-96-11.

[Plaxton 97] C. Greg Plaxton, Rajmohan Rajaraman & Andrea W. Richa. *Accessing Nearby Copies of Replicated Objects in a Distributed Environment.* In Proceedings of ACM Symposium on Parallel Algorithms and Architectures (SPAA), June 1997.

[Porter 80] M.F. Porter. *An algorithm for suffix stripping.* Program, vol. 14, no. 3, page 130−137, 1980.

[Ratnasamy 01] Sylvia Ratnasamy, Paul Francis, Mark Handley & Richard Karp. *A Scalable Content-Addressable Network.* In SIGCOMM'01, San Diego, California, USA, August 27-31 2001.

[Ratnasamy 02a] Sylvia Ratnasamy. *A Scalable Content-Addressable Network.* Thése de doctorat, ICSI Center for Internet Research, Berkeley, California., San Diego, California, USA, August 27-31 2002.

[Ratnasamy 02b] Sylvia Ratnasamy, Scott Shenker & Ion Stoica. *Routing Algorithms for DHTs : Some Open Questions.* In Proceedings of International Peer-To-Peer Workshop, 2002.

[Rhodes 96] Bradley Rhodes & Thad Starner. *The Remembrance Agent : A continuously running automated information retrieval system.* In The Proceedings of The First International Conference on The Practical Application of Intelligent Agents and Multi Agent Technology (PAAM '96), pages 487–495, London, UK, April 1996.

[Rhodes 00] Bradley J. Rhodes. *Margin notes : Building a Contextually aware associative memory.* In Proceedings of the international conference on intelligent user interface (IUI'00), New Orleans, LA, January 9-12 2000.

[Rhodes 04] Bradley Rhodes. *The Remembrance Agent.* `www.remem.org`, February 2004.

[Risson 04] J. Risson & T. Moors. *Survey of Research towards Robust Peer-to-Peer Networks : Search Methods.* Technical Report UNSW-EE-P2P-1-1, University of New South Wales, Australia, September 2004.

[Rowstron 01] A. Rowstron & P. Druschel. *Pastry : Scalable, distributed object location and routing for largescale peer-to-peer systems.* In Conference on Distributed Systems Platforms (Middleware), pages 329–350, Heidelberg, Germany, November 2001.

[Roy 06] Choudhury Romit Roy, Pradeep Kyasanur & Indranil Gupta. *Smart Gossip : An Adaptive Gossip-based Broadcasting Service for Sensor Networks.* In Proc. IEEE MASS, 2006.

[Röscheisen 94] Martin Röscheisen, Christian Mogensen & Terry Winograd. *Shared Web Annotations as a Platform for Third-Party Value-Added, Information Providers : Architecture, Protocols and Usage Examples.* Rapport technique, Computer Science Department, Stanford University, CA 94305, USA, 1994.

[Sadat 04] Hossein Sadat & Ali A. Ghorbani. *On The Evaluation of Adaptive Web Systems.* In J. T. Yao, V. V. Raghavan & G.Y. Wang, editeurs, Proceedings of the 2nd Workshop on Web-based Support Systems (WSS 04), pages 127–136, Beijing, China, September 20 2004. In Conjunction with IEEE/WIC/ACM WI/IAT'04.

[Schmitz 04] Christoph Schmitz. *Self-Organization of a Small World by Topic.* In First International Workshop on Peer-to-Peer Knowledge Management (P2PKM), August 2004.

[Schoonderwoerd 96] Ruud Schoonderwoerd, Owen E. Holland, Janet L. Bruten & Leon J. M. Rothkrantz. *Ant-Based Load Balancing in Telecommunications Networks.* Adaptive Behavior, vol. 5, no. 2, pages 169–207, 1996.

[Segall 00] B. Segall, D. Arnold, J. Boot, M. Henderson & T. Phelps. *Content Based Routing with Elvin4.* In Proceedings of AUUG2K, 2000.

[Servat 02] D. Servat & A. Drogoul. *Combining amorphous computing and reactive agent-based systems : a paradigm for pervasive intelligence ?* In Proceedings of AAMAS'02 (Autonomous Agents and Multi-Agent Systems), Bologna, Italy, July 2002.

[Sharon 03] Taly Sharon, Henry Lieberman & Ted Selker. *A Zero-Input Interface for Leveraging Group Experience in Web Browsing.* In Proceedings of the 8th international conference on Intelligent user interfaces, pages 290 – 292, Miami, Florida, USA, 2003.

[Sripanidkulchai 03] Kunwadee Sripanidkulchai, Bruce Maggs & Hui Zhang. *Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems.* In IEEE Infocom, page 11, San Francisco, April 2003.

[Taher 04] Razan Taher. *Recherche d'information collaborative.* Thése de doctorat, Laboratoire CLIPS-IMAG, Grenoble, 5 Mars 2004.

[Team 97] GVU's WWW Surveying Team. *GVU 8th WWW User Survey'*. `http://www.cc.gatech.edu/gvu/user_survey/survey-1997-10/`, October-November 1997.

[Tempich 04] Christoph Tempich, Steffen Staab & Adrian Wranik. *REMINDIN' : Semantic Query Routing in Peer-to-Peer Networks Based on Social Metaphors.* In Proceedings of the Thirteenth World Wide Web conference (WWW2004), pages 640–649, New York, USA, May 17-22 2004.

[Trousse 99] Brigitte Trousse, Michel Jaczynski & Rushed Kanawati. *Une approche fondée sur le raisonnement à partir de cas pour l'aide à la navigation sur le web.* In Proceedings of Hypertexte & Hypermedia : Products, Tools and Methods (H2PTM'99), Paris, Septembre 1999.

[Varga 02] András Varga. *OMNeT++ discrete event simulation environment.* `www.omnetpp.org`, 2002.

[Vlachakis 03] Joannis Vlachakis, Magdalini Eirinaki & Sarabjot Singh Anand. *IKUM : An Integrated Web Personalization Platform Based on Content Structures and Usage Behavior.* In Workshop on Intelligent Techniques for Web Personalization (ITWP '03), 2003.

[Voulgaris 03] Spyros Voulgaris, Mark Jelasity & Maarten van Steen. *A Robust and Scalable Peer-to-Peer Gossiping Protocol.* In Proceedings of the 2nd International Workshop on Agents and Peer-to-Peer Computing (AP2PC03), Melbourne, Australia, 2003.

[Voulgaris 04] S. Voulgaris, A.-M. Kermarrec, L. Massoulie & M. van Steen. *Exploiting Semantic Proximity in Peer-to-peer Content Searching.* In Proceedings 10th IEEE Int'l Workshop on Future Trends in Distributed Computing Systems (FTDCS 2004), Suzhou, China, May 2004.

[Voulgaris 05] Spyros Voulgaris & Maarten van Steen. *Epidemic-style management of semantic overlays for content-based searching.* In José C. Cunha & Pedro D. Medeiros, editeurs, Euro-Par 2005 Parallel Processing : 11th International Euro-Par Conference. LNCS 3648, Lisbon, Portugal, August 30 - September 2 2005.

[Watts 98] Duncan J. Watts & H. Strogatz Steve. *Collective dynamics of 'small-world' networks.* Nature, vol. 393, pages 440–442, 4 June 1998.

[Weiser 99] M. Weiser, R. Gold & J. S. Brown. *The origins of ubiquitous computing research at PARC since the late 1980s.* IBM Systems Journal, vol. 38, no. 4, pages 693–696, 1999.

[Wexelblat 99] Alan Wexelblat & Pattie Maes. *Footprints : History-Rich Tools for Information Foraging.* In Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'99), David Lawrence Convention Center, Pittsburgh, Pennsylvania, USA, May 15-20 1999.

[Wokoma 02a] I. Wokoma, I. Liabotis, O. Prnjat, L. Sacks & I. Marshall. *A Weakly Coupled Adaptive Gossip Protocol for Application Level Active Networks.* In IEEE 3rd International Workshop on Policies for Distributed Systems and Networks - Policy, Monterey, CA, USA, June 2002.

[Wokoma 02b] Ibiso Wokoma, Lionel Sacks & Ian Marshall. *Biologically Inspired Models for Sensor Network Design.* In Proceedings of the London Communications Symposium 2002 (LCS 2002), 2002.

[Writer 06] Staff Writer. *Mandriva means to help Semantic desktop project.* http://www.cbronline.com/article_news.asp?guid=7DD9518A-EEA2-417A-980F-73C2628D8EB1, 20th February 2006.

[Yolum 03] Pinar Yolum & Munindar P. Singh. *Emergent Personalized Communities in Referral Networks.* In IJCAI Workshop on Intelligent Techniques for Web Personalization (ITWP), 2003.

[Yu 03] Bin Yu, Mahadevan Venkatraman & Munindar P. Singh. *An Adaptive Social Network for Information Access : Theoretical and Experimental Results.* Applied Artificial Intelligence, vol. 17, no. 1, pages 21–38, 2003.

[Zeiliger 05] Romain Zeiliger. *NESTOR The Web Browser and Cartographer.* http://www.gate.cnrs.fr/~zeiliger/nestor/nestor.htm, 2005.

[Zhao 01] Ben Y. Zhao, John Kubiatowicz & Anthony D. Joseph. *Tapestry : An Infrastructure for Fault-tolerant Wide-area Location and Routing.* Rapport technique UCB/CSD-01-1141, Computer Science Division (EECS), University of California, April 2001.

[Zhong 03] Ning Zhong, Jiming Liu & Yiyu Yao, editeurs. Web intelligence. Springer-Verlag, 2003. ISBN : 3-540-44384-3.

Annexe A

Annexes

A.1 Algorithmes de génération de données

Algorithme 5 : Génération des jeux de données

entrée : $Themes$ le nombre de thèmes à générer
entrée : $Documents$ le nombre de document par thème
entrée : $Mots$ le nombre de mots par document
entrée : $Bruit$ le nombre de mots bruit par document
entrée : $TailleDico$ la taille du dictionnaire
entrée : $FreqMot$ fréquence maximum d'un mot du theme
entrée : $FreqBruit$ fréquence maximum d'un mot bruit (hors theme)
sortie : Un ensemble de vecteurs $doc^{theme}_{document}(mot)$

```
1  Listeglobale ← ∅
2  /* Choisir les mots clé de chaque thème */
3  pour theme ← 1 à Themes faire
4  |  Listemots(theme) ← ∅
5  |  pour document ← 1 à Documents faire
6  |  |  pour mot ← 1 à TailleDico faire  doc^theme_document(mot) ← 0
7  |  fin
8  |  tant que |Listemots(theme)| ≠ Mots faire
9  |  |  répéter Choisir aléatoirement mot entre 1 et TailleDico jusqu'à
   |  |  mot ∉ Listeglobale
10 |  |  Listemots(theme) ← Listemots(theme) + {mot}
11 |  |  Listeglobale ← Listeglobale + {mot}
12 |  |  pour document ← 1 à Documents faire
13 |  |  |  Choisir aléatoirement freq entre 1 et FreqMot
14 |  |  |  doc^theme_document(mot) ← freq
15 |  |  fin
16 |  fin
17 fin
18 /* Ajouter du bruit et normaliser */
19 pour theme ← 1 à Themes faire
20 |  pour i ← 0 à Bruit faire
21 |  |  répéter
22 |  |  |  Choisir aléatoirement mot entre 1 et TailleDico
23 |  |  jusqu'à mot ∈ Listemots(theme)
24 |  |  pour document ← 1 à Documents faire
25 |  |  |  Choisir aléatoirement freq entre 1 et FreqBruit
26 |  |  |  doc^theme_document(mot) ← freq
27 |  |  fin
28 |  fin
29 |  pour document ← 1 à Documents faire Normaliser doc^theme_document(mot)
30 fin
```

Algorithme 6 : Génération des topologies initiales

entrée : $Pairs$ le nombre de pairs à générer
entrée : $Addresses$ la taille du carnet d'adresses
entrée : $Interets$ le nombre de centres d'intérêt différents

$Listepairs \leftarrow \emptyset$
/ Créer la liste des pairs */*
pour $index \leftarrow 0$ **à** $Pairs$ **faire**
  Créer un nouveau noeud $pair$
  $pair.interet \leftarrow index \pmod{Interets}$
  $Listepairs \leftarrow Listepairs + \{pair\}$
fin
Mélanger l'ordre des pairs de $Listepairs$
/ Créer un réseau régulier */*
pour $index \leftarrow 0$ **à** $Pairs$ **faire**
  $pair \leftarrow Listepairs[index]$
  pour $i \leftarrow 1$ **à** $\frac{Addresses}{2}$ **faire**
    $voisin \leftarrow Listepairs[index + i \pmod{|Listepairs|}]$
    $pair.voisins \leftarrow pair.voisins + \{voisin\}$
  fin
fin
/ Brasser les connexions */*
pour $i \leftarrow 0$ **à** $\frac{Addresses}{2}$ **faire**
  $pair \leftarrow Listepairs[index]$
  pour $index \leftarrow 0$ **à** $Pairs$ **faire**
    $r \leftarrow$ Valeur aléatoire entre 0 et 1
    si $r < p$ **alors**
      répéter
        $pair\_valide \leftarrow true$
        $nouveau\_voisin \leftarrow Listepairs$[Valeur aléatoire entre 1 et $|Listepairs|$]
        si $nouveau\_voisin = pair$ **alors** $pair\_valide \leftarrow false$
        si $nouveau\_voisin \in pair.voisins$ **alors** $pair\_valide \leftarrow false$
        si $pair \in nouveau\_voisin.voisins$ **alors** $pair\_valide \leftarrow false$
      jusqu'à $pair\_valide$
      $pair.voisins \leftarrow pair.voisins \setminus ancien\_voisin$
      $pair.voisins \leftarrow pair.voisins + \{nouveau\_voisin\}$
    fin
  fin
fin

Table des figures

1.1 Exemple de recherche sur le site `google.fr` 6

1.2 Processus de recherche mené par l'utilisateur 7

1.3 Le Navigateur Internet comme intermédiaire dans la démarche d'accès à l'information 9

1.4 Nouveau schéma d'intéractions et nouvelle place du navigateur Internet 12

1.5 Exemple d'utilisation de Footprints [Wexelblat 99] 14

1.6 Exemple d'utilisation de Nestor [Zeiliger 05] 15

1.7 Capture d'écran du métamoteur de recherche « Webcrawler » 19

1.8 Capture d'écran du métamoteur de recherche « Kartoo » 19

1.9 Capture d'écran du logiciel de lecture de flux RSS « Akregator » 20

1.10 Exemple d'affichage de Margin Notes [Rhodes 00] 23

1.11 Représentation simplifiée de l'architecture **I-KnowUMine**. D'après [Vlachakis 03] 26

1.12 Illustration des schémas de communication 27

1.13 Architecture d'un système stigmergique (adapté de [Parunak 05]) 29

1.14 Système d'écoute d'un dialogue entre deux services. Le conseiller s'abonne auprès de l'agent d'écoute chargé d'observer les messages echangés. Grâce aux informations qu'il récupère, le conseiller peut ainsi suggérer des actions aux services. D'après [Busetta 01] 31

1.15 Graduation de l'intelligence d'un assistant 36

2.1 Modèle simplifié des échanges liés au Web sur Internet 41

2.2 Exemple de fichier HTML (sans indication d'en-tête). `body` indique le corp de la page, `h1` et `h2` sont deux formats de mise en page de section. 41

2.3 Exemple de fichier CSS 42

2.4 Exemple de fichier xml 42
2.5 Exemple de fichier DTD 43
2.6 Exemple de fichier RDF 43
2.7 Exemple (simplifié) de requête et de réponse SOAP 44
2.8 Localisations et échanges de fichiers avec un serveur centralisé 45
2.9 Classification des systèmes P2P 46
2.10 Relations entre « concentrateurs » et « autorités » dans un réseau social 46
2.11 Circulation d'une requête dans un réseau de référents (d'après [Yu 03]) 48
2.12 Les quatre niveaux d'approche de la Web Intelligence (WI), adapté de [Zhong 03] 50
2.13 Le développement du wisdom Web (adapté de [Liu 05]) 51
2.14 La mise en place d'un bureau sémantique social (adapté de [Decker 04]) 52
2.15 Exemple de portail netvibes 54

3.1 Mode de diffusion de données (d'après [Franklin 98]) 56
3.2 Étapes du processus de comparaison de deux documents 57
3.3 Exemple d'application d'un filtre de Bloom 60
3.4 Différentes topologies 63
3.5 Exemple de graphe pour le calcul de coefficients 65
3.6 Les « Small World » : entre ordre et désordre [Watts 98] 67
3.7 Classification des systèmes de circulation de l'information. Le premier niveau de distinction correspond à l'initiative de l'échange (client / serveur). Le second niveau indique la nature du réseau (structuré / non structuré / hybride). 69
3.8 Distributions des identifiants dans Chord (d'après [IonStoica 01]) 71
3.9 Distribution des identifiants dans CAN 72
3.10 Exemple de routage selon l'algorithme de Plaxton *et al.* (adapté de [Plaxton 97]) 73
3.11 Comparatif des méthodes de mise en place d'une infrastructure de diffusion d'événement 78
3.12 Architecture pour le partage de ressources proposé par Koubarakis *et al.* (d'après [Koubarakis 04]) 79
3.13 Mise en place d'une diffusion dirigée. Exemple inspiré de celui de [Dressler 06] . . 80

3.14 Fourmis assurant la circulation de ressources entre différentes zones de stockage dans Anthill [Babaoglu 01, Babaoglu 02] 82

3.15 Ajout d'une structure de liens sémantiques supplémentaires 84

3.16 Exemple de raccourcis créés par SWOP (d'après [Hui 04]) 85

4.1 Contournement d'un obstacle par une population de fourmis (extrait de [Dorigo 96a]) 91

4.2 Utilisation des deux opérateurs d'API 95

4.3 Chaîne d'états du modèle épidémique SI 96

4.4 Chaîne d'états du modèle épidémique SIS 97

4.5 Chaîne d'états du modèle épidémique SIR 98

4.6 Chaîne d'états du modèle épidémique SEIR 99

4.7 Exemple de graphe mettant l'algorithme de bavardage en difficulté 100

5.1 Vision globale de l'implication de PIAF dans l'environnement de travail de l'utilisateur 105

5.2 Description des éléments du niveau « communication » de PIAF 108

5.3 Vision globale 109

5.4 Modélisation des traces de phéromones 110

5.5 Evolution de $\rho(x) = 0.9\mathrm{e}^{-\alpha x}$ 110

5.6 Activités des composants 111

5.7 Fonction de densité de probabilité de la loi triangle 112

5.8 Exemple d'une mémoire $E_{i \leftarrow j}(t)$ contenant 5 évaluations positives sur 12 114

5.9 Scénario conduisant à une situation de blocage pour une information 114

5.10 Variables entrant en jeu lors de la recommandation 116

6.1 Interface graphique contrôlant l'activité de PIAF dans sa version en PERL 125

6.2 Interface graphique permettant de piloter à distance un ou plusieurs PIAFs . . . 125

6.3 Interface graphique du simulateur 126

6.4 Représentation des éléments de PIAF implémentés dans le simulateur 127

6.5 Représentation d'un des réseaux de test de PIAF ainsi que des éléments externes de contrôle 128

6.6 Jeu de données 129
6.7 Représentation du contenu initial du carnet d'adresses - réseau 20 130
6.8 Représentation du contenu initial du carnet d'adresses - réseau 40 131
6.9 Indications de lecture pour les tableaux de résultats 132
6.10 Fenêtre de réglage de PIAF (esquisse) 150
6.11 Comparaison avec une diffusion aléatoire 152
6.12 Comparaison avec une diffusion aléatoire 153
6.13 Méthode d'attribution des informations pour la classification 155
6.14 Exemple de selection d'une destination 156
6.15 Exemple de résultat de classification de la base Iris (1) 157
6.16 Exemple de résultat de classification de la base Iris (2) 157
6.17 Exemple de résultat de classification de la base Iris (3) 158

Liste des tableaux

1.1 Résultats d'une étude sur le Web datant de 1996 (adapté de [Bouras 01]) 10

1.2 Synthèse comparative de différents outils d'aide 36

3.1 Pondération en fréquence 59

3.2 Pondération binaire 59

5.1 Paramètres régissant le comportement des fourmis 122

5.2 Notations relatives à la définition des informations 122

5.3 Notations relatives à la définition du réseau 122

5.4 Définitions des probabilités relatives aux connexions 122

6.1 Paramétrage de l'algorithme de génération de données 129

6.2 Paramétrage de l'algorithme de génération de réseaux 130

6.3 Influence de s_{min} sur la valeur du coefficient de groupement pour un réseau de 20 pairs 134

6.4 Influence de s_{min} sur la valeur du coefficient de groupement pour un réseau de 40 pairs 135

6.5 Influence de s_{min} sur la valeur du coefficient de rappel pour un réseau de 20 pairs 136

6.6 Influence de s_{min} sur la valeur du coefficient de rappel pour un réseau de 40 pairs 137

6.7 Influence de s_{min} sur la valeur du coefficient de précision pour un réseau de 20 pairs 139

6.8 Influence de s_{min} sur la valeur du coefficient de précision pour un réseau de 40 pairs 140

6.9 Influence de n^+ sur la valeur du coefficient de groupement pour un réseau de 20 pairs 141

6.10 Influence de n^+ sur la valeur du coefficient de groupement pour un réseau de 40 pairs 141

6.11 Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 20 pairs 142
6.12 Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 40 pairs 142
6.13 Influence de n^+ sur la valeur du coefficient de rappel pour un réseau de 20 pairs . 143
6.14 Influence de n^+ sur la valeur du coefficient de rappel pour un réseau de 40 pairs . 143
6.15 Influence de η sur la valeur du coefficient de groupement pour un réseau de 20 pairs 145
6.16 Influence de η sur la valeur du coefficient de groupement pour un réseau de 40 pairs 145
6.17 Influence de η sur la valeur du coefficient de precision pour un réseau de 20 pairs 146
6.18 Influence de n^+ sur la valeur du coefficient de precision pour un réseau de 40 pairs 147
6.19 Influence de η sur la valeur du coefficient de rappel pour un réseau de 20 pairs . . 147
6.20 Influence de η sur la valeur du coefficient de rappel pour un réseau de 40 pairs . . 148

Résumé :

Dans ce travail de thèse, nous proposons l'architecture PIAF (Personnal Intelligent Agent Framework) dont l'objectif est de fournir aux utilisateurs un environnement d'échange d'informations non intrusif, autonome et polyvalent. Les problématiques de diffusion de l'information entre utilisateurs et d'optimisation de la topologie du réseau sont abordés avec un algorithme utilisant des fourmis artificielles. L'utilisation de phéromones artificielles déposées sur les connexions entre pairs lors des transferts autorise la constitution d'une mémoire globale des échanges et la détection de centres d'intérêts partagés. Comparativement aux solutions existantes, l'avantage de notre algorithme est d'affranchir l'utilisateur de la définition de profils. Ce dernier n'a besoin ni de s'abonner à un quelconque canal de diffusion ni de paramétrer ses centres d'intérêts pour pouvoir échanger de l'information.

Mots clés :

réseaux P2P – fourmis artificielles – petits mondes

Abstract :

In this thesis, we propose the architecture PIAF (Personnal Intelligent Agent Framework) whose objective is to provide users with an environment for nonintrusive, autonomous and general-purpose exchange of information. The problems of diffusion of information between users and optimization of the network's topology are approached with an algorithm using artificial ants. The use of artificial pheromones deposited on connexions between peers at the time of the transfers authorizes the constitution of a global memory of the exchanges and the detection of shared centers of interests. Comparatively with existing solutions, the advantage of our algorithm is to free the user from the definition of profiles. This last needs neither to subscribe with diffusion channel nor to define its centers of interests to be able to exchange information.

Key words :

P2P network – artificial ants – small worlds

Université François Rabelais Tours, Laboratoire d'Informatique, UPRES-EA 2101, Équipe Handicap et Nouvelles Technologies (`http://www.li.univ-tours.fr`). Polytech'Tours, Département Informatique, 64 Avenue Jean Portalis, 37200 Tours (`http://www.polytech.univ-tours.fr`).