HAL
open science

Le sens au coeur des systèmes d'information

Cyril Labbé

To cite this version:

Cyril Labbé. Le sens au coeur des systèmes d'information. Informatique ubiquitaire. Université de Grenoble, 2010. tel-00809360

HAL Id: tel-00809360

https://theses.hal.science/tel-00809360

Submitted on 9 Apr 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Le sens au cœur des systèmes d'information

Cyril Labbé

Laboratoire d'Informatique de Grenoble, équipe Sigma

6 décembre 2010

Université de Grenoble

École doctorale MSTII
Mathématique, Sciences et Technologies de l'Information, Informatique

Habilitation à diriger des recherches

Jury :

Rapporteurs :	Abdelkader Hameurlain	-	Univ. Paul Sabatier Toulouse
	Jacques Savoy	-	Univ. de Neuchâtel
	Pierre Sens	-	Univ. Paris VI
Président :	Jean-Claude Fernandez	-	Univ. de Grenoble
Examinateurs :	Isabelle Demeure	-	Institut Telecom ParisTech
	Claudia Roncancio	-	Univ. de Grenoble

Remerciements

En premier lieu, mes remerciements vont aux rapporteurs qui ont lu et évalué ce document. Les points de vue et les commentaires qu'ils ont exprimés sont à la fois riches et constructifs. Mes remerciements vont aussi aux autres membres du jury qui sont venus exprimer leur avis sur mon travail. Tous ont permis des échanges abondants et une discussion productive.

Mes remerciements aussi, à tous ceux qui ont participé – de près ou de loin, à Grenoble ou ailleurs dans le monde – aux recherches qui sont présentées dans ce document : étudiants, co-encadrants, co-auteurs, collègues d'enseignement et/ou de recherche et/ou de l'industrie. Enfin, remerciement à tous les membres des différentes équipes auxquelles j'ai appartenu. En particulier à ceux de l'équipe SIGMA qui m'ont fait l'honneur de leur soutien de tous les instants.

Il faut noter que ce document n'aurait pas vu le jour sans la volonté et la persévérance de Claudia Roncancio.

Je tiens aussi à remercier mes parents pour toute l'aide qu'ils m'apportent et mes deux sœurs pour leur présence même lointaine !

Un grand merci enfin à ceux sans qui rien n'est possible, dans l'ordre d'apparition : Caroline, Thaïs, Elian et Lully...

Table des matières

1 Introduction **1**

2 Lexicométrie : à la recherche du sens perdu **5**
2.1 Sens et codage 6
2.2 Retrouver le sens des mots 7
2.3 Replacer un mot dans la structure de la langue 9
2.4 Mesurer l'attraction et la répulsion entre deux mots. 12
2.5 *Amour* dans l'œuvre de Corneille. 13
2.6 Conclusion 15

3 Cache sémantique : le sens au cœur des grilles **17**
3.1 Structure des fichiers : une trace du sens 19
3.1.1 Modèle de données un moyen de prendre en compte le sens . 19
3.1.2 Requêtes : accès par le sens 19
3.1.3 Composition des sources 21
3.2 Exemple d'une source de données composée de plusieurs fichiers . . . 22
3.3 Cache et sens 23
3.3.1 Cache sémantique 23
3.3.2 Cache coopératif 24
3.4 Expérimentations 24
3.5 Conclusion 26

4 Le Pair à Pair et la perte de sens **29**
4.1 Les systèmes P2P structurés détruisent le sens 29
4.2 Systèmes P2P basés sur des DHT 31
4.2.1 Architecture des DHT et sens de l'information 31
4.2.2 Différentes sémantiques pour tirer profit du sens des données 32
4.3 Modèle basé attributs 33
4.4 Modèle relationnel 34
4.5 Données semi-structurées 36
4.6 Conclusion 37

5 Donner du sens au flux de données issues de capteurs **39**
5.1 Traitement des données issues des capteurs 40
5.2 Réseaux de capteurs. Wireless Sensor Networks (WSN) 41
5.2.1 Traitement des données dans les réseaux de capteurs 41
5.2.2 Caractéristiques des réseaux de capteurs 42
5.2.3 Bases de données capteurs 43
5.2.4 Discussion 45
5.3 Systèmes de Gestion de Flux de Données (SGFD) 45

5.3.1 SGFD vs SGBD et flux de données 46
5.3.2 Requêtes continues sur les flux de données 47
5.3.3 Résistance à la charge 48
5.3.4 Discussion 49
5.4 Intégration SGFD et réseaux de capteurs 49
5.5 Conclusion 51

6 Conclusion **53**

A Univers lexical de *amour* chez Pierre Corneille's. **57**
A.1 Attraction : mots sur-utilisés 57
A.2 Répulsion : mots sous-utilisés 57

Bibliographie **59**

Chapitre 1

Introduction

Positionnement

La science du traitement de l'information est largement basée sur la définition de méthodes et d'infrastructures génériques (machines, programmes, codage et stockage de l'information) qui peuvent être utilisées pour différents buts. La recherche de cette généralité est indispensable puisque, pour des raisons évidentes de rapidité, de développement et de coût, on ne construit pas une infrastructure dédiée à chaque problème que l'on souhaite résoudre.

C'est pourquoi les solutions génériques ont toujours revêtu une importance particulière en informatique. Elles permettent la réutilisation de l'existant, le partage de l'information et des infrastructures entre de multiples applications qui ont des objectifs différents. Par définition, une infrastructure de traitement de l'information n'a donc pas d'objectif final extrêmement précis, même si son champ d'action est toujours réduit à certains types d'applications.

De plus, une même information peut être plus ou moins importante et avoir un sens différent selon qu'elle est utilisée par une application ou par une autre. De la même façon, un même processus de traitement peut être utilisé par plusieurs applications, chacune d'elles y associant une importance et un sens particulier. C'est donc au cours de l'utilisation de l'infrastructure, quand les processus de traitement sont en cours d'exécution, que l'information et les traitements associés prennent, pour un temps, un sens extrêmement précis, et, plus l'infrastructure utilisée est dédiée à un champ d'applications spécifiques, plus elle y est performante. C'est ainsi que, l'utilisation du sens associé à l'information ou du sens associé aux processus de traitement, dans les couches basses de l'infrastructure, permet une amélioration qualitative et quantitative de la gestion de l'information.

On a souvent conscience de l'importance de la généricité pour le développement des infrastructures de traitement de l'information. Cependant, on n'a pas toujours conscience que, ayant à disposition ces machines sans but et une information privée de sens, une bonne partie du travail consiste à utiliser le sens associé à l'information. C'est pourquoi les outils qui permettent de donner du sens à l'information et aux processus de traitement qui lui sont associés, font partie intégrante des infrastructures de traitement de l'information et y joueront un rôle de plus en plus central.

Ecrire un algorithme et programmer revient à agencer des opérations élémentaires dans un but précis en utilisant des structures de données génériques en leur associant un sens bien particulier. En effet, aucun sens n'est, à priori, associé à

ces structures de données. De la même manière, un fichier permet le stockage des données sans tenir compte du sens associé à ces données.

Il existe de nombreux cas où le sens de l'information est utilisé au cœur des systèmes pour améliorer la qualité des traitements tout en gardant un degré de généralité important. Ainsi, dans le monde des bases de données, la structuration de l'information à l'aide du modèle relationnel permet de capturer et de mettre à profit le sens associé à l'information. Cette structuration des données est complétée par la structuration des traitements. De la même manière, l'utilisation du concept de transaction permet de regrouper les opérations élémentaires en ensembles d'opérations qui ont un sens au niveau de l'application.

Dans de nombreux domaines de l'informatique, le sens associé à l'information ou aux processus de traitement est capturé et mis à profit à l'aide de techniques de structuration (programmation, modèles de données, transactions, protocoles...). D'autre part, quand le sens associé à l'information est perdu lors du processus d'encodage, la (re)découverte de ce sens peut être nécessaire. Souvent, le sens associé aux données demeure implicitement présent et doit être découvert, par exemple grâce à des techniques de fouille de données. La recherche d'un bon équilibre entre des méthodes, qui font abstraction du sens de l'information et celles qui en tiennent compte est donc une constante dans le développement des systèmes de gestion de l'information.

La mise en réseau des dispositifs de gestion de l'information, qu'ils soient de petite taille (capteurs) ou de grande taille (cluster -super calculateur) accompagne et accélère l'émergence de l'informatique ubiquitaire. Ce mouvement de fond entraîne une explosion, tant de la quantité que de la diversité de l'information disponible. Une des principales difficultés réside dans la capacité à déterminer l'information pertinente pour une application et à la traiter de manière adaptée. Les outils permettant de prendre en compte le « sens » associé à l'information au cœur des systèmes deviennent indispensables.

L'objectif de ce document est d'illustrer, au travers de quatre exemples, l'importance que prend le sens dans le développement des nouveaux systèmes de gestion de l'information. Ces systèmes émergeants forment le socle des futures infrastructures de gestion de l'information. Plusieurs caractéristiques dirigent ces recherches. Il convient de faire face à un grand nombre de sources de données hétérogènes allant du document en langue naturelle aux données issues de capteurs en passant par les grandes bases de données scientifiques. Les contours des systèmes deviennent de plus en plus flous et dynamiques, les participants rejoignant et quittant les systèmes de manière non maîtrisée. Les capteurs sont de plus en plus partie intégrante des grands systèmes d'information et apportent avec eux des problématiques évoquées précédemment comme l'hétérogénéité, la dynamicité mais aussi des problématiques qui leur sont propres comme l'évaluation en flux et la gestion de l'énergie. Les nouveaux systèmes d'information font face à des flux croissant d'informations produites à des rythmes de plus en plus élevés et les processus de traitement associés se doivent de fonctionner de manière fiable et continue.

Toutes ces caractéristiques rendent les méthodes usuelles de gestion de données

peu ou pas du tout opérantes. L'interrogation et l'analyse des données doivent non seulement être adaptée au grand nombre et à l'hétérogénéité des sources mais aussi aux nouvelles structuration des systèmes. Il en est de même pour le transfert et le stockage ou encore pour l'administration des sources de données.

Résoudre ces problèmes ne doit cependant pas faire oublier que les infrastructures de traitement de l'information doivent permettre l'utilisation efficace du sens associé à l'information et que ce sens peut aussi être utilisé pour améliorer les processus de traitement. Nous pensons que chacun des travaux présentés dans ce document illustre ces affirmations dans un contexte particulier.

Portée du document

Nos travaux en lexicométrie nous ont permis de préciser la notion de sens associé à l'information et l'on montrera comment le processus le plus naturel d'encodage de l'information peut entraîner une perte ou une détérioration de ce sens associé à l'information. On verra alors les efforts qu'il est nécessaire de produire pour retrouver ce sens. Ces travaux ont pour objectif global de fournir des outils d'analyse de grandes collections (ou corpus) de textes. Ils sont menés en grande partie en collaboration avec le laboratoire PACTE de Grenoble II. Plus récemment, des techniques similaires ont donné lieu à une collaboration avec le DSSE de Monash University, Melbourne.

Nous pensons que le sens associé à l'information peut être pris en compte dans les couches basses d'une infrastructure pour améliorer le traitement de l'information. C'est ce que nous illustrerons au travers de la présentation de nos travaux dont l'objectif vise à améliorer la gestion de grandes bases de données scientifiques dans les grilles. Ces travaux ont été menés dans le cadre du projet Gedeon ANR masse de données avec l'IBCP, le LIP6 et l'ISREC.

Les travaux que nous avons menés sur les systèmes Pair-à-Pair nous permettront de montrer que la prise en compte du sens dans les couches basses des infrastructures est parfois difficile. Ces systèmes ont comme caractéristiques d'être massivement distribués avec des participants autonomes et hétérogènes. Ils offrent de bonnes propriétés génériques, mais, en contrepartie, ils imposent de gros efforts dans les couches supérieures qui nécessitent l'utilisation du sens associé à l'information. Nos travaux sur ce sujet ont été réalisés dans le cadre de DataGraal puis WebContent.

Nos travaux sur les données de capteurs illustrent comment les techniques de structuration des données peuvent être utilisées pour permettre aux applications de donner un sens particulier aux mesures provenant de capteurs. La prise en compte du sens associé à l'information qu'ils produisent est un des éléments indispensables à leur bonne intégration dans les grands systèmes d'information. Ces travaux ont été menés en grande partie en collaboration et avec le soutien de France-Telecom R&D. Ils font aussi l'objet d'un projet ANR jeune chercheur dont je suis responsable autour des équipes LINA et FT-R&D. Nos travaux dans ce cadre ont aussi bénéficié de collaborations avec le NII de Tokyo et DSSE de Monash University, Melbourne.

Plusieurs étudiants ont participé à ces travaux, en particulier cinq étudiants de

doctorat (deux sont actuellement en cours). Il faut aussi signaler la participation de sept étudiants de Master Recherche (dont un en collaboration avec Monash University et un en collaboration avec le laboratoire PACTE).

Organisation du document

Le chapitre 1 présente certains de nos travaux en lexicométrie. Le chapitre 2 donne un aperçu de nos travaux dans les grilles. Les chapitres 3 et 4 présentent un panorama général du cadre que nos travaux – sur les systèmes Pair à Pair et les capteurs – ont contribué à élaborer.

Chapitre 2

Lexicométrie : à la recherche du sens perdu

Sommaire

2.1 Sens et codage 6
2.2 Retrouver le sens des mots 7
2.3 Replacer un mot dans la structure de la langue 9
2.4 Mesurer l'attraction et la répulsion entre deux mots. 12
2.5 *Amour* dans l'œuvre de Corneille. 13
2.6 Conclusion 15

Ce chapitre précise la notion de sens de l'information. Il montre comment le sens associé à une information est perdu lors du processus d'encodage. Ce sens peut cependant être retrouvé à l'aide de techniques de fouille de données. La méthode présentée ici a été développée en collaboration avec Dominique Labbé ([Labbé 2005]). Son originalité réside dans le fait qu'elle met en valeur autant le sens associé à un mot que le sens qui ne lui est pas associé.

Contributions au domaine

Les travaux présentés dans ce chapitre s'insèrent dans un ensemble plus large de recherches qui ne sont pas discutées dans ce document. Elles visent la mise à disposition de vastes corpus électroniques qui soient facilement exploitables par tous ceux dont l'étude des textes est le métier : terminologues, lexicographes, traducteurs, sociologues, politologues, critiques littéraires... Dans ce cadre, nos contributions ont porté sur la définition et le calcul de distances entre textes [Labbé 2001, Labbé 2003], sur les méthodes de classification automatique dans de grandes collections de textes [Labbé 2006a, Labbé 2008], sur les effets du temps dans un ensemble d'œuvres [Hubert 2004, Hubert 2002, Labbé 2006b] et enfin, nous avons aussi cherché à travailler sur des macro-unités de sens plus grande que l'unité minimale d'information [Monière 2008, Labbé 2010]. Le travail réalisé par Dwi Rahayu (Stage de M2R, en 2010 à Monash University, co-encadrement avec Shonali Krishnaswamy) à lui porté sur l'extraction du sens d'un ensemble d'avis formulés par des utilisateurs [Rahayu 2010a, Rahayu 2010c, Rahayu 2010b].

2.1 Sens et codage

D'après le modèle classique dérivé des travaux de Shanon, la transmission d'un message se compose des phases suivantes :

- encodage des informations à transmettre par l'émetteur,
- transmission,
- décodage du message par le récepteur.

Pour que le message soit compris par le récepteur, il faut que le protocole (moyen) et le format (encodage) soient connus et compris par les deux participants.

Dans les systèmes informatiques ces protocoles et formats sont extrêmement codifiés et ne laissent pas place à des interprétations différentes. Hormis d'éventuels bugs et/ou pannes matérielles, la seule interférence qui peut brouiller le message provient d'un éventuel bruit sur le canal de transmission.

Il en est tout autrement quand on généralise ce modèle pour la transmission d'un message en langue naturelle. Le contenu du message et son sens sont transcrits dans un code beaucoup moins formel. L'émetteur tente de transcrire le sens de son message à l'aide de sa maîtrise de la langue et du sens qu'il associe aux mots. Le décodage du message par le récepteur est lui aussi affecté par les mêmes phénomènes.

L'interprétation d'un message en langue naturelle, qu'il soit écrit ou oral, peut être affecté non seulement par un éventuel bruit sur le canal mais aussi par un bruit qui affecte les processus de décodage et d'encodage. Les processus de décodage et d'encodage dépendent de contextes complexes. La différence entre le contexte associé au codage et le décodage est à l'origine d'un bruit qui peut entraîner une différence d'interprétation du sens porté par le message. Ce bruit peut être de différentes natures. L'émetteur et le récepteur n'associant pas le même sens à un même mot. Il peut y avoir plusieurs raisons à cela : différences culturelles, histoire personnelle et connaissance technique. Mais aussi, comme les langues sont vivantes, la différence temporelle entre le décodage et l'encodage a une influence sur le sens associé aux mots.

Le sens codé/décodé à partir d'un mot dépend donc d'un contexte complexe composé des mots qui l'entourent mais aussi des participants et des époques de codage et d'encodage.

Encoder/Décoder = perte de sens : tout codage et/ou décodage d'une information entraine une perte de sens. Les couches basses des systèmes de transmission de l'information font abstraction du sens originel de l'information. L'information est codée dans un format pratique pour coder et transmettre l'information. Ces formats sont génériques et ne tiennent pas compte du sens originel de l'information. Ainsi, plus on se rapproche des couches basses, plus le sens est oublié et plus la généricité est importante.

C'est ce qui se passe pour les langues naturelles où des symboles permettent de coder des sons qui permettent eux-mêmes de coder des mots qui codent des concepts qui sont l'information réelle.

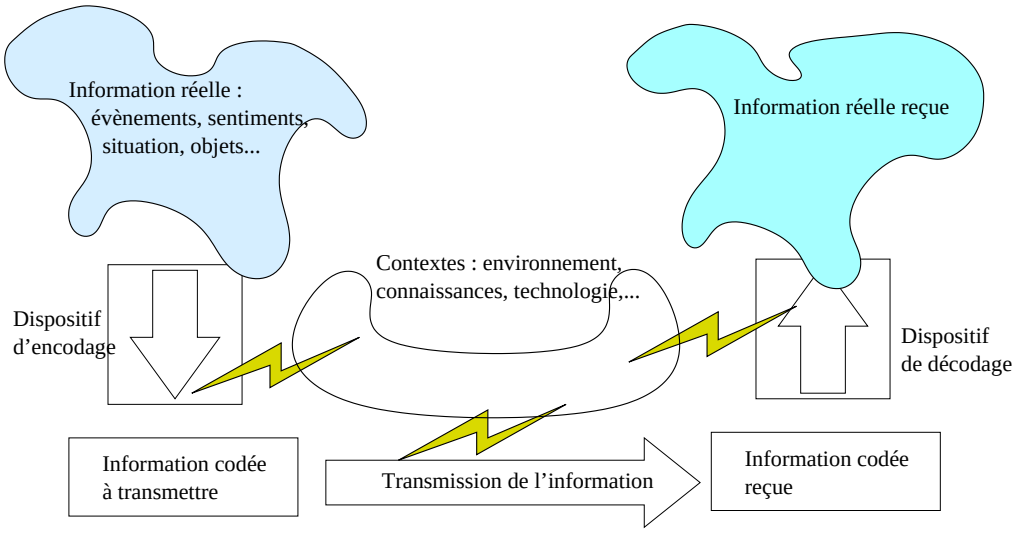

FIGURE 2.1 – Transmission d'information.

C'est aussi ce qui se passe pour toutes les données véhiculées dans les infrastructures modernes de traitement de l'information : les codes binaires qui sont manipulés n'ont qu'un lointain rapport avec le sens originel de l'information.

2.2 Retrouver le sens des mots

La perte de sens qui résulte de l'encodage en langue naturelle donne toute sa dimension aux études littéraires. Ainsi, il peut être particulièrement passionnant de se livrer à l'étude du sens associé à un mot dans un corpus particulier. La méthode décrite dans la suite de ce chapitre montre comment il est possible de détailler le sens qu'un auteur associe à un mot. Les détails de la méthode pourront être consultés dans [Labbé 2005].

Le vocabulaire des phrases contenant le mot étudié est comparé au vocabulaire de l'ensemble du corpus de manière à mettre en lumière les mots qui sont significativement sur-employés dans son voisinage (ils lui sont associés, ils ont un sens qui a un lien avec le mot étudié), mais aussi les mots qui sont significativement sous-employés dans son voisinage (ils ne sont pas associés, leur sens a un lien de « répulsion »).

Cette méthode permettant de calculer « l'univers lexical » d'un mot est un outil intéressant qui permet de découvrir le sens associé à ce mot ainsi que le sens qui n'y est pas associé. La pertinence de la méthode a été démontrée en l'appliquant au mot *amour* dans les œuvres de Pierre Corneille [1]. Cet outil vient compléter les outils développés en recherche d'information qui ont pour objectif de générer des *lexical frameworks* [Grefenstette 1994, Weeds 2005] permettant de regrouper des mots en ensembles selon le sens qui leur est associé.

1. 33 pièces composées entre 1633 et 1674, soit 583 451 mots.

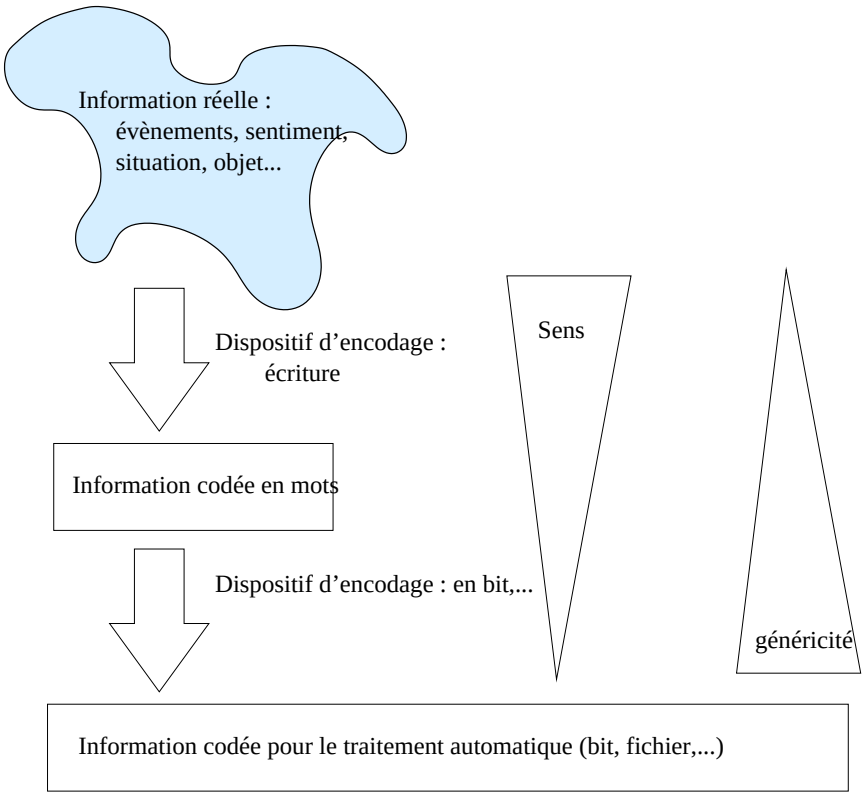

FIGURE 2.2 – Encodage d'information.

2.3 Replacer un mot dans la structure de la langue

Comment déterminer le sens associé à un mot dans une pièce de théâtre du 17e siècle ? La réponse la plus simple consiste à utiliser un dictionnaire de cette période. Par chance les premiers dictionnaires français apparaissent à cette époque. Ainsi, la définition du mot *amour* donnée par l'un de ces dictionnaires [Furetière 1690] est reproduite dans la figure 2.3 et repris pour plus de lisibilité dans la table 2.1.

Table 2.1 – *Amour* dans le dictionnaire de Antoine Furetière (1690)

AMOUR. Subst. m. et f. Passion de l'âme qui nous fait aimer quelque personne ou quelque chose. L'amour divin est le seul qui nous doit enflammer. Les Romains se font sacrifier pour l'amour de la patrie. Il faut donner l'aumône pour l'amour de dieu. L'amour paternel, l'amour conjugal sont les amours les plus violentes. L'amour des richesses est la cause de tous les vices, l'amour de la gloire est la cause de toutes les belles actions. On dit aussi, il aime d'amour, pour dire d'une amitié violente. Ce prince est l'amour des peuples.

Se dit principalement de cette violente passion que la nature inflige aux jeunes gens de divers sexes pour se joindre, afin de perpétuer l'espèce. On dit qu'un jeune homme fait l'amour à une fille, quand il la recherche en mariage. On le dit aussi odieusement, quand il tâche de la suborner. Il s'est marié par amour, c'est-à-dire désavantageusement et par l'emportement d'une passion aveugle. On dit qu'une femme fait l'amour quand elle se laisse aller à quelque galanterie illicite. Il y a aussi des amours brutaux, monstrueux et contre nature. On dit aussi des animaux qui sont en chaleur, qu'ils entrent en amour, lorsqu'ils recherchent leurs femelles.

AMOURS se dit aussi au pluriel. Les livres, les tableaux sont ses amours, il nourrit de folles amours, c'était ses jeunes amours, ses tendres amours. Il signifie aussi l'objet aimé. Mon cœur, mes amours, m'aimerez-vous toujours ?

AMOUR. Subst m. Se prend encore pour la divinité fabuleuse des païens, qu'ils s'imaginaient présider à l'amour. Cupidon est le dieu d'amour. L'amour est tout nu, les flambeaux de l'amour, les flèches de l'amour, le bandeau de l'amour, l'amour est aveugle.

Il signifie aussi en ce sens, tous les petits agréments qui naissent de la beauté. Les jeux, les ris, les amours et les grâces. Vénus est la mère des amours.

AMOUR, se dit proverbialement en ces phrases : il n'est point de belle prison, ni de laides amours. On dit encore : tout par amour et rien par force. On dit encore qu'une femme laide est un remède d'amour. On dit aussi : à battre faut l'amour.

Voici quelques remarques qui peuvent venir à l'esprit à la lecture de cette définition :

- à cette époque, *amour* est un substantif (un nom) masculin ou féminin alors qu'aujourd'hui le féminin n'est plus utilisé ;
- *amours* au pluriel a une signification légèrement différente de *amour* au singulier ;
- comme la plupart des mots en français, *amour* a plusieurs significations.

Un des points important est que le mot est placé dans une catégorie grammaticale : substantif. Ces catégories grammaticales n'ont que peu été affectées par les évolutions de la langue et si le sens d'un mot change au cours des siècles sa catégorie grammaticale, à quelques exceptions près (comme *amour* féminin), est restée stable.

Mais la définition donnée par Furetière, et plus généralement par un dictionnaire,

A M O.

que le Roy qui puisse *amortir* des fiefs. les fiefs *amortis* ne doivent plus rien au Roy.

AMORTIR, signifie aussi, Esteindre, racheter une rente, une pension, une dette. On fait souvent revivre des rentes qui ont été *amorties* ou rachetées. cet homme a *amorti* plusieurs dettes des deniers dotaux de sa femme. il est permis d'*amortir* à prix d'argent une pension sur un Benefice, parce que c'est une chose temporelle.

AMORTIR, signifie aussi, Esteindre une chose allumée, affoiblir, rabatre la violence d'une chose; & se dit souvent avec le pronom personnel. Cet incendie a été grand, mais il s'*amortit*. l'ardeur de la fievre s'*amortit* par la saignée. la natte d'un jeu de paume *amortit* le coup de la balle, empêche sa reflexion. le mouvement le plus violent diminuë toûjours, & enfin s'*amortit* tout à fait. on garnit un santereau d'espinette d'un morceau d'estoffe pour *amortir* soudain le son de la corde.

AMORTIR, se dit figurément en Morale. L'âge *amortit* les plus violentes passions, *amortit* l'ardeur de la jeunesse. son amour s'est fort *amorti*, pour dire, s'est fort ralentie. ce Ministre a sagement *amorti*, appaisé la sedition.

AMORTI, IE. part. pass. & adj.

AMORTISSEMENT. s. m. est une grace ou concession que fait le Roy par lettres patentes aux gens de main morte, comme Eglises & Communautés, de tenir des fiefs & heritages à perpetuité, sans être obligés de les mettre hors de leurs mains, moyennant une somme qu'on luy paye pour le desdommager des profits & confiscations qui luy appartiendroient dans les mutations qui se feroient, s'ils demeuroient dans le commerce ordinaire. L'*amortissement* est dû au Roy, & l'indemnité au Seigneur immediat dont releve le fief. la Chambre des francs-fiefs & *amortissemens*.

AMORTISSEMENT, signifie aussi, Adoucissement d'une douleur, d'une inflammation. Les Medecins saignent pour procurer l'*amortissement* de l'ardeur de la fievre. si cette emplastre ne guerit pas, elle cause du moins l'*amortissement* de la douleur. sans l'*amortissement* du coup de cette balle, il auroit été plus grievement blessé.

AMORTISSEMENT, signifie aussi, Extinction, rachat. L'*amortissement* d'une rente se fait en remboursant le sort principal. l'*amortissement* d'une pension se fait par la mort du pensionnaire, ou par la renonciation qu'il y fait moyennant quelque argent qu'on luy donne pour la racheter.

AMORTISSEMENT. Terme d'Architecture. C'est ce qui termine quelque ouvrage au haut d'un bastiment, ou d'une menuiserie, ou d'une corniche, comme quelque vase, ou quelque figure; & generalement tout ce qui fait saillie ou ornement en cet endroit-là.

Tous ces mots viennent de *mors*.

AMOVIBLE, ou *Amobile*. adj. masc. & fem. Ter. Ecclesiastique, qui se dit de celuy qu'on établit en quelque charge ou employ par commission, & pour un temps seulement, & qui peut être revoqué & destitué, quand il plaist au superieur. Les Vicaires des Parroisses n'ont pas une charge ou un Benefice en titre, ils sont *amovibles ad nutum*, toutefois & quantes qu'il plaist aux Curez. tous les Obedienciers ou Religieux qu'on envoye desservir un Benefice sont *amovibles* ou *amobiles*.

AMOUR. s. m. & f. Passion de l'ame qui nous fait aimer quelque personne, ou quelque chose. L'*amour* divin est le seul qui nous doit enflammer. les Romains se sont sacrifiez pour l'*amour* de la Patrie. il faut donner l'aumosne pour l'*amour* de Dieu. l'*amour* paternelle, l'*amour* conjugale sont les *amours* les plus violentes. l'*amour* des richesses est la cause de tous les vices. l'*amour* de la gloire est la cause des plus belles actions. On dit

A M O. A M P.

aussi, Il aime d'*amour*, pour dire, d'une amitié violente. Ce Prince est l'*amour* des peuples.

AMOUR, se dit principalement de cette violente passion que la nature inspire aux jeunes gens de divers sexes pour se joindre, afin de perpetuer l'espece. On dit, qu'un jeune homme fait l'*amour* à une fille, quand il la recherche en mariage. On le dit aussi odieusement, quand il tasche de la suborner. Il s'est marié par *amour*, c'est à dire, desavantageusement, & par l'emportement d'une aveugle passion. On dit, qu'une femme fait l'*amour*, quand elle se laisse aller à quelque galanterie illicite. Il y a aussi des *amours* brutaux, monstrueux & contre nature.

On dit aussi des animaux qui sont en chaleur, qu'ils entrent en *amour*, lors qu'ils recherchent leurs femelles.

AMOURS, se dit aussi au pluriel. Les livres, les tableaux sont ses *amours*. il nourrit de folles *amours*. c'étoient ses jeunes *amours*, ses tendres *amours*. Il signifie aussi, l'objet aimé. Mon cœur, mes *amours*, m'aimerez-vous toûjours?

AMOUR. s. m. se prend encore pour la Divinité fabuleuse des Payens, qu'ils s'imaginoient presider à l'amour. Cupidon est le Dieu d'*amour*. l'*Amour* est tout nud. les flambeaux de l'*Amour*, les fleches de l'*Amour*. le bandeau de l'*Amour*. l'*Amour* est aveugle.

Il signifie aussi en ce sens, tous les petits agreements qui naissent de la beauté. Les jeux, les ris, les *amours*, & les graces. Venus est la mere des *amours*.

AMOUR, se dit proverbialement en ces phrases. Il n'est point de belle prison, ni de laides *amours*. On dit encore, Tout par *amour*, & rien par force. On dit encore, qu'une femme laide est un remede d'*amour*. On dit aussi, A battre faut l'*amour*.

AMOURACHER. v. n. qui ne se dit qu'avec le pronom personnel & en mauvaise part de ceux qui sont amoureux d'une personne de ville ou d'inégale condition. Cette femme s'est *amourachée* de son valet.

AMOURETTE. s. f. qui ne se dit aussi qu'en mauvaise part des amours illicites, ou entre personnes disproportionnées. Ce vieillard a encore une petite *amourette* en teste. il s'est marié par *amourette*.

AMOUREUX, EUSE. s. m. & f. Qui a de la passion pour quelque chose, ou quelque personne. Il est *amoureux* des tableaux, *amoureux* de toutes les femmes qu'il voit. *amoureux* de bonne foy. *amoureux* transi.

AMOUREUX, EUSE. adj. se dit des choses qui sont les instrumens de l'amour, ou qui concernent l'amour. Regards *amoureux*. desirs *amoureux*. vers & billets tendres & *amoureux*. faveurs *amoureuses*. les femmes Maures sont de complexion *amoureuse*. On dit poetiquement, Languir dans l'empire *amoureux*. Les Medecins appellent les deux muscles obliques de l'œuil, *amoureux*, circulaires & rotateurs, parce que leur mouvement marque de la tendresse ou de la passion.

On dit proverbialement d'un homme qui aime en plusieurs lieux, que c'est un *amoureux* des onze mille Vierges; & de celuy qui n'aime point du tout, qu'il est *amoureux* comme un chardon.

AMOUREUSEMENT. adv. D'une maniere amoureuse. Cet amant regardoit *amoureusement* sa maîtresse. un goinfre regarde *amoureusement* les bons morceaux. le ciel regarde *amoureusement* la terre.

A M P.

AMPHIBIE. adject. & subst. masc. Animal qui vit tantost dans l'eau, tantost sur la terre. Les crocodilles, les castors, les loutres, les grenouilles, les tortuës, le veau marin, sont des animaux *amphibies*. Ce mot vient du Grec, où il signifie, *Vie en deux manieres*, ou *en deux endroits*.

K 5 AM-

FIGURE 2.3 – *Amour* dans le dictionnaire de Furetière (1690)

n'est que le produit d'une impression, d'une interprétation des auteurs du dictionnaire. Cette interprétation est-elle correcte ? Présente-t-elle des biais ? La phrase *Il y a aussi des amours brutaux, monstrueux et contre nature* ne prend tout son sens qu'avec la connaissance des codes moraux de l'époque et une interprétation basée sur les moeurs de notre temps ne peut être qu'imparfaite.

A défaut d'interroger les contemporains, une réponse à ces questions peut être apportée par l'étude des textes d'époque. Ainsi pour le français du 17e peut-on se référer aux textes des plus illustres hommes de lettres de l'époque : Pierre Corneille (1606-1684), Jean-Baptiste Poquelin Molière (1622-1673) et Jean Racine (1639-1699).

Une première étape consiste à étudier l'entourage du mot *amour*. L'entourage consistant classiquement à regarder un empan fixe (par exemple dix derrière, dix devant) ou la phrase complète. L'étude de la fréquence avec laquelle les mots apparaissent dans l'entourage va donner une idée du sens dans lequel le mot est employé. Cependant cet exercice reste limité car, en français, les mots les plus fréquents sont les mots outils (cf. table 2.2). Si cet outil permet de relever la présence de certains mots dans l'entourage du mot étudié, cela n'apporte que peu d'information. Ainsi, on ne sera pas étonné de trouver des articles et les auxiliaire *être* et *avoir* dans cet entourage.

La question importante est de savoir si ces mots apparaissent significativement plus souvent dans l'entourage de *amour* que ce qui est normalement attendu. De même, il est intéressant d'exhiber les mots qui sont attendus dans l'entourage de par leur fréquence dans le corpus et qui n'apparaissent pas (ou peu) dans cet entourage.

Mot avant		Mot après
l' (686)		de (59)
d' (216)		qui (48)
mon (200)	amour (1888)	est (40)
votre (110)		pour (40)
cet (82)		a (24)

Table 2.2 – Entourage immédiat du mot *amour* dans l'œuvre de Corneille's (entre parenthèses : la fréquence absolue)

Un autre point important est de replacer les formes graphiques dans la structure de la langue. En effet, en français, il existe un grand nombre de formes ambigües. Environ un tiers des mots sont homographes : deux mots différents, deux sens différents mais une même écriture. Par exemple *(un) (une) somme*, *(des) (nous) sommes*, *(un) (il a) été*, ... Si l'on s'intéresse au sens et non uniquement aux formes, la normalisation (Mr. Monsieur M.) et la « lemmatisation » sont indispensables. Par exemple si l'on souhaite un jour retrouver la saison *été* au milieu des participes passés *été* ou encore pouvoir différencier *le pouvoir* du verbe *pouvoir*. Ceci illustre la manière dont les règles de codage sont importantes pour déterminer le sens d'un mot *code*.

Une fois ces opérations de normalisation et de lemmatisation effectuées (ces processus sont décrits dans [Labbé 1990] ; une discussion sur l'importance de ces

processus dans [Pincemin 2004]), il est possible de retrouver « l'univers lexical » d'un mot, ce qui permet de mieux saisir le sens qui lui est associé dans l'esprit de l'auteur.

2.4 Mesurer l'attraction et la répulsion entre deux mots.

La première étape consiste à identifier les phrases où le mot étudié apparaît. L'ensemble des phrases où le mot apparaît est appelé l'univers lexical. Par exemple l'univers lexical du mot *amour* dans l'ensemble de l'œuvre Corneille est constitué de 1822 phrases et d'environ 60000 mots.

En comparant les fréquences d'apparition des mots dans l'univers lexical et leurs fréquences d'apparition dans l'ensemble de l'œuvre, il est non seulement possible d'étudier les mots qui sont associés à *amour* dans l'œuvre (dans l'esprit ?) de Corneille, mais aussi ceux qui y sont « opposés ».

Pour savoir si un mot apparaît plus ou moins dans l'univers lexical d'un mot étudié, il est nécessaire de se doter d'un modèle de répartition des mots de manière à être capable de calculer la probabilité de l'apparition de ce mot dans l'univers lexical.

Soit un mot i (*haine* par exemple) dont on veut connaître le lien d'attraction/répulsion avec *amour* dont l'univers lexical sera noté U. Si on note X la variable aléatoire « fréquence du mot i dans U », choisir un modèle revient à choisir une méthode pour calculer la probabilité suivante :

$$P(X = x) = probabilité\ que\ le\ mot\ \texttt{i}\ apparaisse\ x\ fois\ dans\ U$$

Une fois ce choix fait, il est alors possible de mesurer par un indice $L_{i/U}$ la force ou la faiblesse du lien entre le mot i et l'univers lexical U du mot étudié en se basant sur la valeur observée $f_{i/U}$ pour la variable aléatoire X :

$$L_{i/U} = P(X \leq f_{i/U}) = \sum_{j=0}^{j=f_{i/U}} P(X = j)$$

Si $L_{i/U}$ est grande (proche de 1) on peut en déduire que les deux mots s'attirent mutuellement et que l'apparition de l'un entraine l'apparition de l'autre de manière plus importante que ne le prévoit le modèle. Au contraire, si $L_{i/U}$ est faible (proche de 0), les deux mots ont tendance à se repousser, à s'éviter plus que ne le prévoit le modèle. Cela laisse supposer que dans le premier cas les mots s'associent dans l'esprit de l'auteur alors que dans le second ils s'évitent et sont donc plutôt opposés.

$L_{i/U}$ est une probabilité qui varie entre 0 et 1. De manière assez usuelle, on peut considérer que 0.99 est une valeur anormalement haute alors que 0.01 est une valeur excessivement petite. On peut alors considérer, avec moins de 1% de chances de se tromper, qu'il y a « trop » ou « trop peu » d'emplois de ce mot dans les phrases contenant *amour*. Ces valeurs peuvent être utilisées pour l'étude de mots peu fréquents. Cependant, les règles du langage s'écartent grandement des modèles

« aléatoires » et quand on étudie de grands corpus et des mots très fréquents – comme *amour* dans l'œuvre de Corneille – un trop grand nombre de mots vérifient ces conditions et il est alors nécessaire de resserrer ces seuils de manière assez stricte. (cf. [Labbé 2005] pour plus de détails).

2.5 *Amour* dans l'œuvre de Corneille.

L'annexe A donne l'ensemble des mots qui sont associés ou repoussés par *amour*. Il sont classés en tenant compte de leur fonction grammaticale et par indice de lien décroissant. Par exemple, les substantifs (noms) qui ont un lien d'attraction le plus fort avec *amour* sont *haine*, *cœur*, *feu* ; les verbes sont *céder*, *éteindre* ; les adjectifs sont *conjugal*, *parfait*, *paternel*. Mais il est aussi intéressant de découvrir que *amour* a un lien de répulsion avec *seigneur*, *dieu* ou *roi*.

Il faut cependant remarquer qu'une lecture attentive de cette liste laisse suggérer que tous les liens positifs ne proviennent pas forcement d'une association porteuse de sens. Par exemple, la présence des mots *cour* et *retour* peut être expliquée par leur lien sémantique avec l'*amour* mais leur présence est sans doute renforcée par la terminaison en « our» qui permet la rime. Ainsi peut-on comprendre la présence des mots *tour*, *jour* ou *séjour*. Bien évidemment, les contraintes de la versification ne sont pas capturées par le modèle aléatoire.

La liste fournie en annexe confirme beaucoup de points de la définition du dictionnaire de Furetière. Par exemple, il n'est pas surprenant de découvrir *Vénus* ou *Psyché* comme noms propres associés à *amour*. L'association avec le vocabulaire du feu (cf. table 2.3) est aussi un cliché bien connu est peut être confirmé par l'application de la méthode aux œuvres de Corneille et de Racine. L'ensemble suggère une vision de l'*amour* assez en phase avec les canons de l'époque.

verbes	éteindre, allumer, éclater, étouffer, rallumer, brûler
substantif	feu, amorce, ardeur, froideur
adjectifs	éteint, puissant

Table 2.3 – L'univers lexical de *amour* et le thème du feu

L'utilisation du lien « négatif » permet de renforcer les informations données par les associations positives. Par exemple, la sur-utilisation dans l'entourage du mot *amour* de la troisième personne *il*, *lui* répond à l'absence de la deuxième personne *tu* et *vous* qui sont tous deux sous-utilisés. On met ainsi en lumière que, dans l'œuvre de Corneille, le mot *amour* est peu utilisé face-à-face par les personnages concernés. Il apparaît de manière plus notable dans les dialogues entre un des deux personnages concernés et une tierce personne (généralement un confident).

Ces listes apportent aussi des informations surprenantes au premier abord et qui ne peuvent pas être découvertes qu'avec les liens positifs et que la seule utilisation

des « collocations » ne peut mesurer.

Ainsi la liste des liens négatifs fait apparaître en bonne position : *seigneur, dieu, ciel, roi.* Ce qui peut être surprenant au vu de la définition fournie par Furetière qui commence par l'amour de dieu. Ces mots sont sous-employés dans l'entourage d'*amour* et au vu de leur fréquence dans l'ensemble de l'œuvre ils devraient apparaître plus souvent dans l'entourage de ce mot. Il semble qu'ici les associations négatives rassemblent ce que l'on considère généralement comme les obstacles à l'amour chez Corneille : *dieu, ciel, roi, mort, combat, guerre.*

L'apparition du mot *haine* est aussi intéressante. En effet, le mot *haine* est peu fréquent dans l'ensemble du corpus et il semble difficile de mettre à jour la relation avec *amour* à l'aide des méthodes classiques. La méthode proposée ici permet de mettre en avant la sur-utilisation de ce mot dans l'entourage de *amour* au regard de l'étendue du corpus complet. Autrement dit, *haine* apparaît peu dans l'ensemble de l'œuvre mais, quand il apparaît, il est souvent dans le voisinage de *amour.*

CHIMENE. C'est peu de dire aimer, Elvire : je l'adore ; Ma passion s'oppose à mon ressentiment ; Dedans mon ennemi je trouve mon amant ; Et je sens qu'en dépit de toute ma colère, Rodrigue dans mon coeur combat encor mon père : Il l'attaque, il le presse, il cède, il se défend, Tantôt fort, tantôt faible, et tantôt triomphant ; Mais en ce dur combat de colère et de flamme, Il déchire mon coeur sans partager mon âme ; Et quoi que mon amour ait sur moi de pouvoir, Je ne consulte point pour suivre mon devoir : Je cours sans balancer où mon honneur m'oblige. Rodrigue m'est bien cher, son intérêt m'afflige ; Mon coeur prend son parti ; mais malgré son effort, Je sais ce que je suis, et que mon père est mort. (Le Cid, Acte III, scène 3, vers 810-824)

Table 2.4 – La phrase la plus caractéristique de l'univers lexical de *amour* dans l'œuvre de Corneille.

Pour aider à l'interprétation de ces listes, il peut être intéressant de fournir des phrases caractéristiques de l'emploi du mot étudié. Cela peut être fait en recherchant dans les phrases de l'univers lexical celles comportant le plus d'associations positives et le moins d'associations négatives. La table 2.4 montre la phrase ayant le plus grand score pour le mot *amour.* Dans quelques vers, on retrouve : *cœur*, *flamme*, l'opposition *passion* vs *devoir* et *honneur.* Il est important de noter que ces vers constituent l'un des passages importants de la plus fameuse pièce de Corneille. Il s'agit de l'acte III scène III du Cid où Chimène explique qu'elle n'arrive pas à haïr

l'assassin de son père. On redécouvre ainsi de manière automatique un des passages les plus souvent cités pour illustrer le Cid.

2.6 Conclusion

Le codage d'une information entraîne une perte de sens. Pour les textes en langues naturelles cette perte peut être due aux différences dans le sens que chacun de nous associe à un mot. Plus globalement, le contexte d'utilisation d'un mot va influer sur le sens qui lui est donné. Le sens donné et transmis par le texte dépend intimement d'un ensemble complexe de contextes : l'époque, le concepteur, ceux à qui il s'adresse, les mots entourant le « code », le support (écrit ou oral), le genre (vers, prose, théâtre, roman,...). Le sens originellement associé à un mot peut être retrouvé, ou du moins précisé, par l'étude des mots présents dans son voisinage.

L'exemple de l'étude du mot *amour* dans l'œuvre de Corneille montre qu'il est important de retrouver non-seulement les mots qui sont sur-représentés dans son entourage mais aussi les mots qui sont sous-représentés. Ceci permet une étude plus fine des concepts mis en jeu et surtout, cela permet une mise en évidence riche et contrastée du sens qui est associé au mot dans le corpus.

On soulignera également qu'il est important de dissocier le sens associé à l'information et le codage de celle-ci. En effet, le sens d'un mot code va dépendre d'un ensemble de règles de codage et de décodage qui ne sont pas incluses dans le mot code lui-même.

Ainsi pour les langues naturelles il est important de ne pas confondre la graphie d'un mot et le sens qui lui est associé. Par exemple, en français [2], le nom *été* et le participe passé *été* du verbe être ne peuvent se distinguer qu'en utilisant les règles de codage de l'information qui, en l'occurrence, sont les règles grammaticales du français. Ces règles de codage/décodage font partie intégrante de la transmission du sens associé à l'information. Les prendre en compte et les utiliser pour le traitement de l'information est particulièrement important.

Les chapitres suivants montrent comment les règles de codage de l'information peuvent être mises à profit dans d'autres contextes.

2. Ceci n'est pas spécifique au français. En anglais *desert* est à la fois un verbe (déserter) et un nom (désert).

Chapitre 3

Cache sémantique : le sens au cœur des grilles

Sommaire

3.1 Structure des fichiers : une trace du sens 19
3.1.1 Modèle de données un moyen de prendre en compte le sens . 19
3.1.2 Requêtes : accès par le sens 19
3.1.3 Composition des sources 21
3.2 Exemple d'une source de données composée de plusieurs fichiers 22
3.3 Cache et sens 23
3.3.1 Cache sémantique 23
3.3.2 Cache coopératif 24
3.4 Expérimentations 24
3.5 Conclusion 26

Une fois l'information codée, elle peut être manipulée, transportée et/ou transformée en faisant abstraction de son sens originel. Un fichier ou un paquet réseau est manipulé comme une suite de « bits » indépendamment du sens et de la nature de l'information. Cette suite de codes n'a qu'un rapport indirect avec le sens originel de l'information.

Cependant, il n'est pas rare de voir le « sens » et la nature de l'information réintégrés dans les processus de traitement. Il existe, par exemple, des protocoles dédiés pour les réseaux selon la nature de l'information envoyée. De la même manière, un système de stockage peut tenir compte de la nature et du sens des informations pour en permettre une meilleure manipulation. C'est l'approche qui est classiquement adoptée par les Systèmes de Gestion de Bases de Données (SGBD) où la structure du codage de l'information (typiquement à l'aide du modèle relationnel) est intimement liée au sens qui lui est associé. En utilisant cette structuration, il est possible d'améliorer les traitements (recherche, mise à jour,..) tant d'un point de vue qualitatif que quantitatif.

De nombreux fichiers informatiques contiennent des données qui sont structurées en fonction de leurs sens, à l'aide d'un protocole de codage spécifique. Ce protocole est utilisé et mis à profit par les applications du domaine. Cependant, les couches basses des systèmes de gestion de l'information manipulent l'information avec la

granularité du fichier sans tenir compte de cette structuration, c'est-à-dire, sans tenir compte du sens associé à l'information.

C'est dans ce contexte, que l'idée principale du projet Gedeon (auquel nous avons participé) était de permettre une manipulation à un grain plus fin de l'information, en proposant un système de gestion de fichiers hybride pour les grilles [Globus , Foster 2006, gLite , EGEE , SRB , Mobius] qui tire profit de la structuration inhérente de l'information.

Dans ce contexte de grilles, les techniques de cache sont un élément particulièrement important et on les retrouve à plusieurs niveaux dans ces architectures. Les caches ne sont efficaces que quand ils sont bien configurés au regard du contexte d'utilisation. La présence évidente de nombreux caches distribués sur les différents nœuds d'une grille implique de manière assez naturelle la mise en place de stratégies de coopération entre eux. Le fait de tenir compte du sens associé à l'information pour définir ces coopérations ne peut être que bénéfique.

L'idée globale est donc de tenir compte, au cœur du système, de la manière dont le sens de l'information est codé et est utilisé pour améliorer les fonctionnalités et l'efficacité du système.

Ce chapitre présente le système de gestion de fichiers hybride pour les grilles ainsi que des techniques de cache sémantique qui permettent de tirer profit de la structuration de l'information. Ainsi, le cache sémantique en conservant des résultats de requêtes permet de tirer profit du sens pour améliorer les performances. Les techniques de coopération entre caches permettent aussi d'utiliser ce « sens » à bon escient de manière à « diffuser » les données là où elles seront le plus utiles.

Contributions au domaine

Le projet ANR masse de données Gedeon (11/2004 – 8/2008) a fédéré les efforts des laboratoires Laboratoire d'Informatique de Paris VI (LIP6), de l'Institut de Biologie et Chimie des Protéine (IBCP) de Lyon, du Laboratoire d'Informatique de Grenoble (LIG) et L'Institut Suisse de Recherche Expérimentale sur le Cancer (ISREC). Au sein de ce projet, nos contributions au travers de la thèse de Laurent d'Orazio [d'Orazio 2007a] (co-encadré avec Claudia Roncancio) ont plus particulièrement porté sur la réalisation d'un canevas [1] de cache [d'Orazio 2005]. Ce canevas facilite le développement de différentes techniques de caches et a permis de tester et de proposer une nouvelle technique de cache sémantique adapté aux grilles légères : le cache dual. Différentes politiques de coopération entre les caches ont été proposées [d'Orazio 2010, d'Orazio 2007b, d'Orazio 2008]. Enfin, le cache dual et les techniques de coopération ont été insérés dans l'intergiciel Gedeon pour fournir un système de gestion de fichiers hybride [d'Orazio 2006, Vanlentin 2006, Denneulin 2010].

1. Framework.

3.1 Structure des fichiers : une trace du sens

Les fichiers manipulés par une application sont structurés en fonction du sens associé à l'information que ces fichiers représentent. Cette section montre comment la connaissance de ce sens peut être mise à profit pour améliorer les fonctionnalités d'un système de gestion de fichiers. La section 3.1.1 présente le modèle de données utilisé par l'intergiciel Gedeon et la section 3.1.2 présente les capacités d'interrogation intégrées dans ce système de fichiers qui gère des sources de données distribuées.

3.1.1 Modèle de données un moyen de prendre en compte le sens

La grande majorité des données scientifiques brutes se trouve dans des fichiers dit *plats* et de nombreuses applications scientifiques se nourrissent de ces fichiers plats pour réaliser leurs calculs. Ces gros ensembles de données peuvent être vus comme de grandes collections d'enregistrements contenant un mélange de données et de méta-données. Le modèle utilisé pour structurer ces données est, la plupart du temps, simple. Les méta-données peuvent, souvent, être comparées à une collection de couples (attribut-valeur). Ce modèle fondamental est très largement utilisé dans de nombreux domaines. En effet, il a l'avantage de la simplicité et reste largement ouvert à des modifications futures en autorisant, de manière simple, l'ajout de nouveaux couples.

A partir de ce constat, l'idée est d'améliorer la gestion de ces données en offrant la possibilité de les parcourir à l'aide d'un langage de requêtes portant sur les valeurs et les noms d'attribut (langage de deuxième ordre). Aucune tâche additionnelle de modélisation n'est demandée. On fait donc l'économie de ces tâches qui sont souvent lourdes et complexes et nécessitent le travail d'experts. L'approche préserve aussi l'interface fichier traditionnellement utilisée par les applications existantes.

Trois niveaux d'abstraction sont ainsi proposés pour accéder aux données. Premièrement, une source de données est vue comme un ensemble de fichiers qui sont accessibles localement ou de manière distante au travers de la grille. Deuxièmement, ces fichiers sont constitués d'enregistrements et un enregistrement est une liste de couple (attribut-valeur). Troisièmement, le plus bas niveau donne accès aux sources, par une interface standard de type fichier alors que l'interface de plus haut niveau permet un accès qui tient compte du sens. Ainsi au plus bas niveau, une source Gedeon est composée de fichiers standards des systèmes d'exploitation. Ces fichiers sont manipulés efficacement par les systèmes d'exploitation et donc aucun surcoût n'est introduit par la couche de stockage.

3.1.2 Requêtes : accès par le sens

Avec Gedeon, les requêtes sont utilisées pour sélectionner des enregistrements de sources de données distribuées et/ou dupliquées sur la grille. Dans une requête, les sources de données peuvent être des fichiers (locaux ou distants) ou des alias qui définissent des compositions de sources (similaires à des vues dans les systèmes de gestion de bases de données). La composition des sources (détaillée dans 3.1.3) peut

être de différents types : *union*, *round-robin* et *join*. Le résultat des requêtes est un ensemble d'enregistrements présenté sous forme de fichier. La navigation dans cet ensemble d'enregistrements peut être réalisée de la même manière que la navigation dans un ensemble de fichiers. On obtient ainsi un système de fichiers hybride à l'échelle de la grille enrichie par des requêtes sémantiques.

Ces requêtes, principalement des requêtes de sélection, améliorent la navigation en autorisant l'utilisateur à énoncer des prédicats pour sélectionner les enregistrements les plus pertinents d'une source de données. Une requête permet de créer un ensemble virtuel d'enregistrements qui peut faire partie de l'arborescence des fichiers. Les prédicats sont des expressions régulières sur les noms d'attributs et sur leur valeur. La figure 3.1 illustre cette navigation. Par exemple :

```
> cd BigFlatDataFile$Date== 1991
```

désigne un nœud virtuel du système de fichiers qui contient le résultat de la sélection des enregistrements ayant un attribut *date* égal à *1991* dans la source de données *BigFlatDataFile*. La visualisation des enregistrements de ce nœud virtuel se fait par extraction dans la (ou les) sources puis par concaténation de ces enregistrements (commande unix standart *ls* par exemple).

`>ls` `BigFlatDataFile`	Liste les fichiers du répertoire courant.
`>cd BigFlatDataFile`	Se déplacer *dans* le fichier.
`BigFlatDataFile> ls` `Data1 Data2 Data3 ...`	Liste des enregistrements du fichier courant.
`>cat Data1` `...`	Affiche l'enregistrement Data.
`BigFlatDataFile>cd $Date==/1991/`	Sélectione les enregistrements avec une valeur de l'attribut *Date* contenant la chaîne de caractères *1991*.

`BigFlatDataFile/$Date==/1991//>ls`
`Data2 Data21 Data322 ...`
`...`
	Liste l'ensemble des enregisterments sélectionnés.

`BigFlatDataFile/$Date==/1991//>cd /size/==123`	Parmi les enregistrements déjà sélectionnés, sélectionner ceux qui ont un nom d'attribut contenant la chaîne de caractères *size* et ayant *123* comme valeur.

FIGURE 3.1 – La navigation dans les enregistrements d'un grand fichier plat.

La mise à jour des fichiers est aussi possible. Bien que la modification des données scientifiques soit assez improbable (elles sont le résultat d'expériences), cette opération est nécessaire dans le système Gedeon. En effet, l'opération de composition de sources *Join* a pour principale fonctionnalité de permettre d'enrichir une source de données par des informations fournies par l'utilisateur. Ces informations, fournies dans un fichier, ont de fortes chances de ne pas être figées.

Pour des raisons d'efficacité, Gedeon utilise de manière assez intensive des techniques de cache et de réplication. La cohérence entre, les données stockées dans les sources et les répliquas dans les caches, doit être assurée et dépend de la politique de

cohérence choisie au niveau des caches. Comme dans le cas des données scientifiques, le taux de mise à jour étant jugé faible, une politique « paresseuse » a été utilisée.

3.1.3 Composition des sources

Une des fonctionnalités importantes de l'intergiciel est de pouvoir composer des sources de données éparpillées sur plusieurs sites. Ces opérations de composition permettent à l'utilisateur de transmettre au système une partie du sens qu'il associe aux données. Cette composition permet ainsi de mettre à profit le sens de l'information au cœur du système.

Une source de données peut être créée par composition de plusieurs sources. La description d'une source de données, comme une composition de sources, peut être vue comme un plan d'exécution de requête. Chaque requête comporte une description des sources qui doivent être employées pour évaluer cette requête. Une même source peut être exploitée de différentes manières dans plusieurs descriptions de sources.

Le sens associé aux données peut ainsi être utilisé pour composer les sources. Par exemple, si plusieurs sources sont identiques (elles contiennent le même ensemble de données, la même information), elles peuvent être composées pour fournir de la tolérance aux fautes ou pour paralléliser les traitements. Si elles sont disjointes, elles peuvent être regroupées par une opération d'union (regroupement des enregistrements) pour former une seule source virtuelle. Si elles sont complémentaires, le résultat peut être construit en joignant certains des enregistrements provenant des différentes sources (opération de jointure). Ainsi les opérations possibles sont :

local La source est un fichier local. Toute description de source se termine forcément par une telle source.

remote La source est distante, la requête est juste transférée au site distant qui envoie les résultats qui sont ensuite exploités localement.

union La même requête est exécutée sur les sources de données et les résultats sont agrégés. Cette opération permet de répartir la charge de travail sur plusieurs sites et ainsi de tirer profit du parallélisme. Elle permet aussi de regrouper des enregistrements pour former une source plus complète.

round robin exécute la même requête tour à tour sur différentes sources pour répartir la charge de travail entre plusieurs sites.

join Similaire à l'équi-jointure bien connue dans les bases de données. La figure 3.2 montre un exemple de l'opération de jointure sur les attributs désignés avec clef et ID. Il est intéressant de noter que les attributs des différentes sources peuvent être différents. Il n'y a pas de notion stricte de schéma.

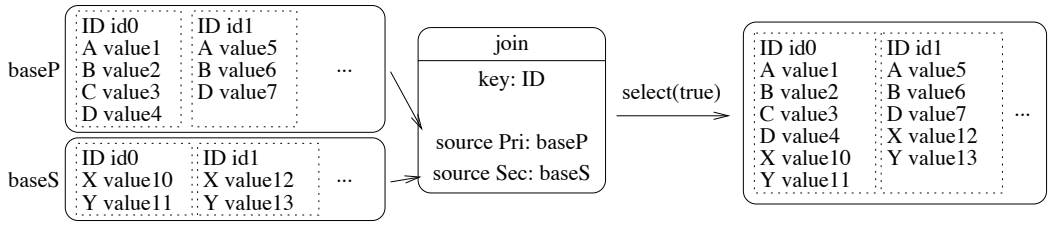

FIGURE 3.2 – Exemple de l'opération de jointure entre deux fichiers Gedeon

3.2 Exemple d'une source de données composée de plusieurs fichiers

Cette section montre comment plusieurs fichiers peuvent être utilisés pour créer des sources de données. L'utilisateur injecte du sens dans le système de gestion de fichiers qui est capable de l'utiliser pour améliorer le traitement de l'information.

Un exemple typique d'utilisation est le partage d'enregistrements à l'échelle de la planète. Chaque chercheur a besoin de créer, selon ses besoins propres, une vue particulière des informations disponibles dans le système. Il doit être en mesure de définir « sa » source de données en fonction de l'ensemble des sources à sa disposition. Habituellement, un chercheur voudra utiliser une base de référence (*Bref*) enrichie avec ses propres annotations (*Banot*) et celle de la communauté à laquelle il appartient (*Bcom*). Ces trois sources existant déjà, l'utilisateur souhaite pouvoir définir une nouvelle source comme une composition de celles-ci.

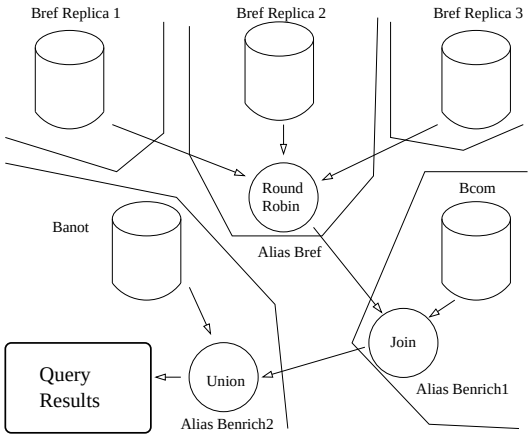

FIGURE 3.3 – Composition de sources de données dans Gedeon.

Il y a de fortes chances pour que la source de référence (*Bref*) soit largement utilisée et donc dupliquée. Il est donc intéressant d'utiliser l'opération de round-robin sur ces différentes copies. *Bcom* apporte de nouveaux attributs aux enregistrements de la source de référence. Ces attributs sont spécifiques à la communauté du cher-

cheur. Enrichir *Bref* avec *Bcom* peut être fait en définissant une source de données *Benrich1* utilisant l'opération de *jointure*. En considérant que *Banot* ne contient que des attributs déjà présents dans *Bcom*, *Banot* contient uniquement de nouveaux enregistrements qu'il faut ajouter à *Benrich1*. Ceci est réalisé en créant une source de données *Benrich2* qui utilise l'opérateur *union*. La figure 3.3 illustre la composition résultant de ces choix et montre comment le sens que l'utilisateur associe aux données peut être utilisé pour améliorer la gestion des données. L'*union* permet de tirer profit du parallélisme, le *roundrobin* permet d'obtenir une meilleure tolérance aux fautes et une meilleure répartition de la charge.

3.3 Cache et sens

L'intergiciel Gedeon est conçu pour être utilisé dans des systèmes à large échelle avec un grand volume de données. Dans ce contexte, l'objectif est d'optimiser l'évaluation des requêtes et de limiter le temps de réponse. Cette section présente une solution à base de caches sémantiques [Keller 1996, Dar 1996, Ren 2003, Roussopoulos 1991, Keller 1996, Lee 1999] qui consiste à prendre en compte le sens associé à l'information pour construire le contenu des caches.

3.3.1 Cache sémantique

Au cours du projet Gedeon, un cache sémantique spécifique a été développé : le *dual cache* [d'Orazio 2008, d'Orazio 2007a, d'Orazio 2006]. Ce cache est basé sur une coopération entre deux caches qui sont de nature différente : un cache de requêtes et un cache d'enregistrements.

D'une part, le cache de requêtes permet de se souvenir des résultats des requêtes déjà posées. Pour cela on associe à une requête la liste des identifiants d'enregistrements qui composent le résultat. D'autre part, le cache d'enregistrements permet de conserver un ensemble d'enregistrements et leurs identifiants.

Quand une requête est soumise au cache dual, elle est transférée au cache de requêtes. Selon que la requête est présente ou non dans ce cache, il se produit un succès ou un échec. Un succès peut être total, quand la requête a déjà été évaluée, et l'ensemble des enregistrements résultats sont alors identifiés. Ce succès peut n'être que partiel quand une requête ayant un sens voisin a déjà été évaluée, seul un sous-ensemble des enregistrements résultats est alors identifié. Ainsi, en cas de succès (même partiel) un ensemble d'enregistrements est identifié comme faisant partie de la réponse. Avec un peu de chance, ces enregistrements peuvent être trouvés dans le cache d'enregistrements. Les identifiants, ainsi que les parties de requêtes qui restent à évaluer, doivent être transmises aux sources de données concernées.

Structuration en enregistrements et requêtes sémantiques sont des manifestations du sens associé aux données qui sont prises en compte au sein du cache pour permettre d'améliorer des processus de traitement de l'information. Ainsi on limite les transferts de données en ne transférant pas tous les fichiers (les données recherchées

peuvent venir de plusieurs sources/fichiers) mais uniquement les enregistrements intéressants. De même en cachant des résultats de calcul, on allége la charge de travail sur les sources de données, une partie du calcul étant économisé et déporté vers le client.

Ces bonnes propriétés peuvent encore être améliorées en définissant des méthodes de coopération entre les caches.

3.3.2 Cache coopératif

La coopération des caches est basée sur la notion de proximité [d'Orazio 2007b]. En effet, la coopération entre deux caches ne sera efficace que s'ils ont une certaine complémentarité. La notion de proximité est une tentative de mesurer cette complémentarité.

Cette proximité entre caches peut être de différentes natures. Elle peut être purement « physique ». Par exemple deux caches d'un même site seront considérés comme complémentaires puisque leur coopération permettra d'éviter le transfert de données vers leur site commun. Dans le cadre du cache dual, cette notion semble la plus adaptée au cache d'enregistrements.

La proximité peut aussi être liée au sens des données qui intéressent un utilisateur. Deux caches ayant des données de sens « similaire » pourront répondre à des questions « similaires ». Ainsi deux caches de requêtes, s'ils sont accédés par des utilisateurs d'une même communauté d'intérêts, peuvent coopérer efficacement puisque les résultats des calculs qu'ils stockent ont plus de chances d'être pertinents.

3.4 Expérimentations

Cette section donne un rapide aperçu d'une série d'expérimentations menées pour observer l'efficacité d'une architecture de caches coopératifs utilisant notre approche de proximité, qu'elle soit physique, sémantique ou une combinaison des deux. Nous poserons le contexte bio-informatique de ces expérimentations ainsi que l'architecture où sont déployés les caches. Les différentes métriques retenues pour l'interprétation des résultats viendront valider l'évaluation de performance de notre proposition.

Contexte applicatif d'expérimentations

Nous nous plaçons dans un contexte d'accès à des masses d'information par des biologistes pour le traitement de données génomiques. Ces opérations sont coûteuses en temps de calcul et en transfert de données. Une approche de cache coopératif semble pertinente dans ce contexte. Si nous considérons aussi les besoins des biologistes, en terme de recherche d'information, qui accèdent au contenu de fichiers plats de très grande taille à l'aide de requêtes successives sémantiquement proches, des caches sémantiques sont requis pour exploiter au mieux les données cachées.

Les expériences utilisent comme support la banque biologique de séquences de protéines *Swissprot* [Swiss-Prot]. Cette source d'informations contient un grand nombre d'annotations sur chacune des séquences répertoriées. Cette banque de données est un fichier contenant 750Mo d'enregistrements. Un enregistrement se compose d'une séquence de protéines et de ses annotations. Le modèle utilisé pour décrire les enregistrements est de type attribut / valeur, chaque ligne étant composée du nom de l'attribut et de la valeur associée (ou de l'ensemble de valeurs associées). Le contexte applicatif est principalement en lecture et se prête bien à l'utilisation de caches.

Aspect matériel, déploiement sur grille

L'intergiciel Gedeon a été déployé sur Grid5000, la plate-forme française d'expérimentations sur grille. Des clusters sur trois sites différents (Nancy, Rennes et Sophia-Antipolis) ont été utilisés.

Expérience grille

	Temps de réponse	Évaluations sur les serveurs	Données transférées
Sans coopération	44,1 s	35 %	9.0 Go
Coopération physique	43,7 s	35 %	8.8 Go
Coopération sémantique	28,4 s	17 %	7.9 Go
Coopération sémantique et physique	23,4 s	17 %	5.1 Go

Table 3.1 – Métriques de performances de protocoles coopératifs dans un contexte grille. 5000 requêtes sur 20 clients répartis uniformément à Nancy, Rennes et Sophia-Antipolis. Taux de raffinement des requêtes 40% [d'Orazio 2007a].

Cette expérience a été réalisée pour étudier les différents protocoles de résolution proposés pour le cache dual dans un contexte grille. Le tableau 3.1 présente les résultats comparant différents protocoles de coopération (en ligne dans le tableau) : sans coopération, coopération physique, coopération sémantique et enfin un mixte des coopérations sémantique-physique selon la proximité. Les colonnes du tableau représentent les métriques considérées : le temps moyen de réponse à une requête en secondes, le taux de requêtes évaluées sur les serveurs, donné en pour cent, et finalement le volume de données transférées entre les serveurs et un cache.

Globalement, le tableau montre qu'utiliser une résolution coopérative permet de réduire le temps de réponse, cette diminution étant de l'ordre de 50 % dans le cas d'une combinaison des coopérations. Plus généralement, toutes les coopérations permettent une diminution (plus ou moins grande) du temps de réponse. Cette diminution provient de différents facteurs : un gain en termes de transfert de données et d'évaluation sur les serveurs.

La résolution physique, qui ne tient pas compte du sens, est l'approche présentant le plus faible gain. La coopération entre les caches d'objets n'a, en effet, aucun impact sur le taux de requêtes évaluées sur les serveurs. De plus, le gain en termes de volume de données transférées est assez faible (environ 200 Méga octets). Ce faible gain peut s'expliquer par le fait que la plupart des résolutions au sein du cache concerne le cache de requêtes. Dans le cas d'une coopération physique, la résolution de défaut pour le cache de requêtes se fait auprès du serveur, les objets étant récupérés par ce même cache. Par conséquent, les caches d'objets sont moins contactés que dans le cas d'une double coopération.

L'utilisation d'une coopération, entre cache de requêtes, basée sur la proximité sémantique permet de réduire le nombre de requêtes à évaluer sur les serveurs. On observe effectivement, que le taux de requêtes évaluées sur les serveurs est de 35 % sans coopération, alors qu'il est diminué de moitié (17 %) lorsque celle-ci est activée. Il est également possible de remarquer que la coopération entre caches de requêtes permet de diminuer le volume des données transférées, celui-ci passant de 9 à 7.9 Giga octets lorsque la coopération est prise en compte. En effet, lorsque les caches de requêtes coopèrent, les sources de données sont plus souvent accédées à l'aide d'identifiants, évitant de récupérer des objets déjà stockés.

La résolution sémantique et physique représente le cas de la double coopération. Les résultats montrent que le gain de la coopération entre cache d'objets est bien plus significatif que précédemment, diminuant de 2.8 Giga octets le volume de données transférées lorsque l'on passe de la seule coopération sémantique à une association des deux types de coopération. A noter que le résultat en terme de taux d'évaluation reste inchangé entre la résolution sémantique et la résolution physique. Ce résultat est logique, puisque l'accès au cache de requêtes est le même dans les deux protocoles.

3.5 Conclusion

Les infrastructures de gestion de l'information sont par définition génériques elles sont ensuite configurées/déclinées/paramétrées en fonction de l'usage que l'on souhaite en faire.

Un système de gestion de fichiers pour grille ne tient pas forcément compte de la nature ou du sens des informations pour stocker ou transférer les données. Pourtant les applications qui utilisent ces données connaissent la manière dont elles sont structurées et savent en tirer profit. Le protocole de codage étant connu et explicité, il est alors possible d'en tenir compte pour améliorer le traitement de l'information. C'est un peu du sens de l'information qui est utilisé pour améliorer les performances et les fonctionnalités globales du système.

L'exemple de la navigation intra-fichiers, inter-fichiers distribués, de la composition des sources de données et de la coopération des caches proposées par le projet Gedeon montre que la prise en compte du sens améliore le niveau de fonctionnalité et permet aux données de « diffuser » dans l'architecture pour se retrouver au plus près de l'endroit où elles vont être utiles.

Le chapitre suivant porte également sur la gestion de données au sein d'infrastructures de grande taille et fortement distribuées. Il va montrer comment une infrastructure conçue indépendamment de la structure de l'information, en ne tenant pas compte du sens, rend difficile une manipulation riche et efficace de l'information.

Chapitre 4

Le Pair à Pair et la perte de sens

Sommaire

4.1 Les systèmes P2P structurés détruisent le sens 29
4.2 Systèmes P2P basés sur des DHT 31
4.2.1 Architecture des DHT et sens de l'information 31
4.2.2 Différentes sémantiques pour tirer profit du sens des données 32
4.3 Modèle basé attributs 33
4.4 Modèle relationnel 34
4.5 Données semi-structurées 36
4.6 Conclusion 37

Ne pas tenir compte du sens dans les couches basses des systèmes de traitement de l'information peut se révéler pénalisant lorsque l'on cherche à accéder à l'information en utilisant le sens qui lui est associé. C'est ce problème qui se pose pour un certain type de systèmes Pair à Pair (P2P) qui ont été conçus pour résoudre des problèmes sans lien avec le sens de l'information. Pour palier à cette déficience un ensemble de solutions a été développé ces dernières années. Un panorama de ces solutions est présenté dans ce chapitre.

Contributions au domaine

Nos contributions dans ce domaine, au travers de la thèse de María del Pilar Villamil (co-encadrée avec Claudia Lucia Roncancio) [Villamil 2006a], ont permis le développement d'algorithmes permettant l'évaluation de requêtes déclaratives ainsi que des stratégies d'indexation permettant d'améliorer les performances de l'évaluation. C'est l'une des grandes originalités de ce travail de fournir plusieurs stratégies d'exécution [Villamil 2005a, Villamil 2005b, Villamil 2006b, Roncancio 2009]. Un prototype (PinS) [Villamil 2004] a été réalisé et il figure dans le panel des systèmes étudiés ici. La validation expérimentale de ce type de systèmes reste toujours extrêmement problématique. Il est important de noter que nous avons réalisé une étude comparative, tant théorique qu'expérimentale, des coûts associés aux différentes alternatives proposées. Cette étude détaillée peut être trouvée dans [Villamil 2006a].

4.1 Les systèmes P2P structurés détruisent le sens

Les systèmes pair à pair, P2P, sont des systèmes massivement distribués composés d'un très grand nombre de sites qui coopèrent afin de partager des ressources

– documents, données, etc. La composition de tels systèmes est dynamique et autorise le départ et l'arrivée de pairs à tout moment. Les pairs sont considérés comme potentiellement volatiles ce qui impose la mise en place d'une gestion de la coopération décentralisée et robuste. De plus, tous les sites, bien qu'hétérogènes et autonomes, sont considérés comme « équivalents » : ils peuvent être client, serveur ou client/serveur au même moment. Aucun contrôle ou serveur centralisé ne peut donc être envisagé [Androutsellis-Theotokis 2004].

Les systèmes P2P diffèrent sur divers aspects notamment dans leur architecture. Les systèmes dits *non structurés*, comme Gnutella [Gnutella], utilisent un réseau logique couvrant [1] dont la topologie n'a pas de régularité particulière. Dans les systèmes dits *structurés* la topologie du réseau logique est très organisée, par exemple en anneau (Chord [Stoica 2001], Pastry [Rowstron 2001] et Baton [Jagadish 2005]). Enfin, dans certains systèmes, tels que Piazza [Gribble 2001] et Somewhere [Adjiman 2004], les liens de voisinage au sein du réseau logique sont de nature sémantique.

Nous nous intéressons ici aux systèmes P2P structurés basés sur des tables de hachage distribuées (DHT) [Androutsellis-Theotokis 2004, Steinmetz 2005]. Ils proposent un espace logique permettant de localiser les pairs et les données. Ils permettent de trouver, de manière totalement distribuée, des ressources partagées dans le système avec une complexité faiblement dépendante du nombre d'objets stockés dans l'infrastructure. Cette bonne caractéristique permet d'envisager la réalisation de systèmes de gestion de données passant à l'échelle pour un grand nombre d'objets et de pairs.

Malheureusement, les systèmes P2P-DHT, avec l'utilisation de tables de hachage, ne tiennent pas compte de la structuration des données, et donc du sens associé à l'information. Et pourtant, pour pouvoir retrouver les objets intéressants, perdus au milieu d'un très grand nombre d'objets enregistrés dans ces structures, il est nécessaire de disposer d'outils permettant de retrouver ces informations en utilisant le sens qui leur est associé.

Ainsi, l'utilisation des tables de hachage distribuées permet d'obtenir de bonnes propriétés génériques mais, en ne prenant pas en compte le sens associé aux données, elles rendent une recherche par le sens beaucoup plus difficile. D'où un grand effort pour permettre de redonner/retrouver du sens dans les données qui sont stockées dans les systèmes P2P-DHT.

La section 4.2 propose de montrer pourquoi les bonnes propriétés génériques de ces systèmes ne permettent pas de tirer profit du sens de l'information et comment il est alors nécessaire de développer des techniques permettant l'utilisation de ce sens. Ces techniques sont classées en trois grands groupes selon le modèle de données et les langages utilisés, l'ensemble permettant de retrouver l'information en fonction de son sens. L'étude de ces travaux est l'objet des sections 4.3, 4.4 et 4.5.

1. *Overlay network.*

4.2 Systèmes P2P basés sur des DHT

La section 4.2.1 présente un découpage en couches fonctionnelles des systèmes P2P de type DHT. La plus haute de ces couches est celle qui cherche à tirer profit du sens de l'information. La section 4.2.2 présente les trois grands types de structuration des données qui permettent de retrouver l'information en fonction de son sens.

4.2.1 Architecture des DHT et sens de l'information

L'architecture en couches fonctionnelles présentée ici permet de regrouper, au sein d'une même couche, les fonctions et les propriétés de même nature qui sont garanties par les différents systèmes.

Au niveau le plus bas (figure 4.1) se trouvent les services de gestion du réseau logique entre les pairs, puis on trouve les services qui prennent en charge la gestion du stockage des données et enfin, au niveau le plus haut, les services qui concernent la gestion de ces données en tant que ressources structurées. Ce sont ces derniers qui utilisent la structuration de l'information et le sens qui lui est associé.

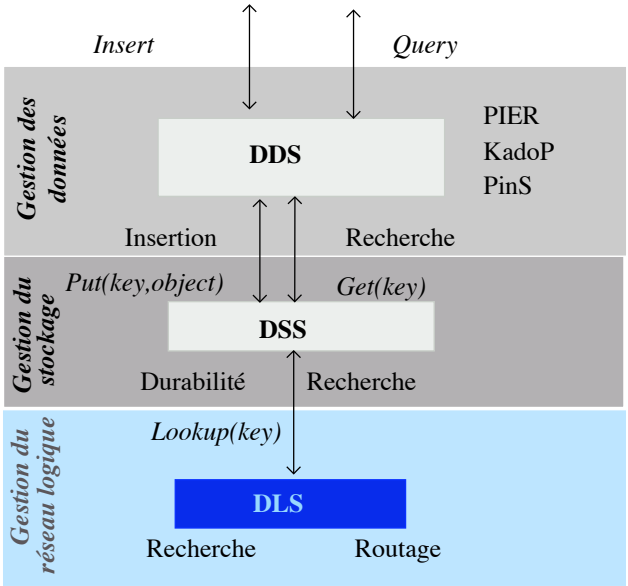

FIGURE 4.1 – Architecture globale pour la gestion de données en P2P structuré

La couche gestion du réseau logique gère la communication entre les pairs, l'insertion et la disparition des pairs dans le système ainsi que leur localisation. Ces fonctionnalités sont regroupées au sein d'un service nommé *Distributed Lookup Service* (DLS). La fonction `lookup` est responsable de la localisation les pairs du système en utilisant une description du réseau logique (données de routage). La couche gestion du stockage est, elle, responsable de la « permanence » des données, le stockage et la recherche à l'aide des clés de hachage. Le service *Distributed Storage Service* (DSS) regroupe les fonctions `put` et `get`. Ces fonctions permettent respectivement

le stockage et la recherche de données. Contrairement aux systèmes non structurés, il est possible de garantir que toute réponse présente (stockée) quelque part dans le système sera retournée par les requêtes.

Dans la configuration de base, les systèmes P2P DHT restent limités en termes de recherche. Pour localiser un objet, il est nécessaire de connaître son identifiant (sa clé). Rechercher de l'information en fonction de son sens n'est pas facilité. Une recherche multi-attributs basée sur des critères d'égalité, d'inégalité, jointure ou agrégat, n'est pas possible.

Le découplage complet entre les couches basses (DSS, DLS) et le sens des données rend difficile l'évaluation d'opérateurs – même simples comme par exemple l'inégalité. Pour la localisation des données par le sens, la méthode classique consiste à tirer profit de la structuration utilisée pour coder le sens de l'information. Par exemple le système prendra en compte des mots-clés ou des couples attribut-valeur associés aux données. Ces méta-données sont utilisées dans des requêtes permettant de localiser les données pertinentes. Ces requêtes sont prises en charge par le niveau le plus haut *Distributed Data Service* (DDS) de la figure 4.1. En général, le résultat est constitué par l'ensemble des identifiants d'objets vérifiant les conditions exprimées dans la requête.

Pour encoder le sens de l'information, le choix d'une structuration, d'un « modèle » est indispensable. Ce choix aura un impact sur le degré d'expressivité des requêtes. Plusieurs choix sont possibles, l'objet de la section suivante est d'expliciter certain d'entre eux..

4.2.2 Différentes sémantiques pour tirer profit du sens des données

Les possibilités d'interrogation proposées par un DDS sont étroitement liées à la nature des données partagées qui va influer sur le choix des métadonnées et du modèle choisi. Cette structuration détermine les critères utilisables pour formuler les requêtes. On trouve ainsi des approches utilisant des mots-clés [Gnawali 2002, Shing 2004] ou basées sur des attributs [Gnawali 2002, Triantafillou 2003, Huebsch 2003, Galanis 2003]. Dans la plupart des cas [Cai 2003, Gnawali 2002, Triantafillou 2003, Huebsch 2003, Galanis 2003], les métadonnées sont stockées à l'aide du DSS. Elles sont identifiées par une clé obtenue à l'aide d'une fonction de hachage. Un tel identifiant sera associé, selon les cas, à un mot-clé [Shing 2004, Gnawali 2002], un attribut [Gnawali 2002, Garcés-Erice 2004] ou un couple attribut-valeur.

Les opérateurs proposés par les langages de requêtes sont aussi un élément important. L'opérateur d'égalité est proposé par la majorité des systèmes [Gnawali 2002, Shing 2004]. L'opérateur d'inégalité, malgré son apparente simplicité, est bien plus difficile à mettre en œuvre. Ceci résulte de la politique de stockage « aveugle » de l'information. En effet, la couche DSS prend en charge le placement des ressources et le seul critère de placement utilisé est la clé de hachage. Cette approche ne considère aucune sémantique pouvant servir à cerner les pairs utiles à l'évaluation d'une requête d'inégalité. Certains, comme [Galanis 2003, Huebsch 2005], proposent des

solutions intéressantes pour l'évaluation de requêtes d'inégalité. Par ailleurs, d'autres proposent des opérateurs de jointure et de tri [Triantafillou 2003, Huebsch 2003].

Dans la suite de ce chapitre, trois grandes familles de travaux qui cherchent à utiliser le sens de l'information pour améliorer la gestion de l'information dans les systèmes P2P sont présentées. Les systèmes sont classés en fonction du modèle de données choisi. C'est le choix de ce modèle qui fixe une sémantique permettant de coder le sens des données et donc de tenir compte de ce sens au sein du système :

- métadonnées de type mots-clés ou couples attribut-valeur,
- modèle relationnel (tableau),
- modèle navigationnel et/ou semi-structuré (arbres).

4.3 Modèle basé attributs

Nous nous intéressons ici à des systèmes qui adoptent des métadonnées simples sous la forme de couples <attribut, valeur>. Ces systèmes, contrairement aux requêtes supportées par la couche DSS, permettent une recherche tirant profit du sens des données à l'aide de requêtes exprimées sur ces attributs.

KSS [Gnawali 2002] et MLP [Shing 2004] proposent des langages de requêtes qui permettent d'utiliser des conditions d'égalité sur les valeurs des attributs. PinS [Villamil 2005a] et MAAN [Cai 2003] proposent en plus des techniques d'indexation et d'optimisation de requêtes. Ces systèmes autorisent des requêtes avec des conditions d'égalité et d'inégalité.

A Keyword-Set Search System for Peer-to-Peer Networks (KSS)

Le système de recherche KSS s'appuie sur DHash [Dabek 2001] pour la gestion du stockage, et sur Chord [Stoica 2001] pour la gestion du réseau logique. Ces systèmes représentent respectivement les couches DSS et DLS.

KSS est un des premiers travaux permettant l'évaluation des requêtes d'égalité au-dessus des systèmes P2P basés sur des DHT. Bien qu'il propose une stratégie d'évaluation performante (basée sur la matérialisation des requêtes), pour des requêtes incluant des conjonctions de mots-clés ou de termes, le critère de choix des requêtes à matérialiser est indépendant de la fréquence d'utilisation de la requête. Cela entraine la matérialisation d'un grand nombre de requêtes peu ou pas utilisées. Par ailleurs, la duplication des métadonnées proposées augmente le coût de stockage. En effet, toutes les métadonnées d'un objet sont dupliquées dans tous les pairs stockant une de ces métadonnées.

MLP

MLP est une solution dédiée tirant profit de propriétés particulières fournies par SkipNet, le système P2P sous-jacent. En effet, des éléments comme la topologie hiérarchique et la diffusion des messages sont la base de la stratégie d'évaluation d'une requête.

Bien que MLP permette l'évaluation d'une requête d'égalité de façon parallèle, le nombre de messages et le nombre de pairs contactés augmentent par rapport aux travaux qui n'utilisent pas de groupes de diffusion. De plus, il n'est pas possible de garantir des réponses compréhensives. Ceci est dû aux départs potentiels de pairs pendant le processus de construction de la réponse dans les niveaux de la hiérarchie.

PinS

PinS [Villamil 2005a, Villamil 2006a] est un service de localisation de données pour des systèmes P2P utilisant une DHT. Il est portable et a été expérimenté au-dessus de Past/Pastry. Les requêtes s'expriment par des critères d'égalité, d'inégalité et de jointure [Prada 2008] sur les métadonnées.

Diverses stratégies d'évaluation des requêtes sont proposées. La stratégie *catalogue local* utilise une approche hybride, *data shipping* et *query shipping*) [Kossmann 2000] pour l'évaluation des requêtes. La *stratégie basée sur index distribué par attribut* permet l'évaluation des requêtes ayant des termes d'inégalité en ne contactant que des pairs qui possèdent l'information nécessaire pour répondre aux requêtes et enfin la *stratégie multiniveaux* distribue la recherche sur les index par attribut pour limiter le risque de surcharge des pairs stockant ces index.

Elles peuvent être utilisées en accord avec le contexte d'évaluation pour améliorer les temps de réponse. PinS n'ajoute pas de contraintes sur la configuration du système P2P ni sur la fonction de hachage utilisée. Néanmoins, des problèmes de saturation des pairs peuvent apparaître, problèmes générés par la taille et le placement des index.

MAAN

MAAN [Cai 2003] utilise une approche P2P pour l'enregistrement et la découverte de ressources partagées au sein d'une grille. Comme PinS, les requêtes s'expriment avec des critères d'égalité ou d'inégalité sur des métadonnées.

Néanmoins, cette solution impose la connaissance préalable des domaines des attributs ainsi que de la distribution des données. De plus, un domaine très vaste peut engendrer des temps d'évaluation des requêtes importants du fait de l'augmentation du nombre de pairs à contacter. Par ailleurs, la duplication des métadonnées a un impact sur le volume des données à stocker et sur le temps de création lors de l'enregistrement d'une ressource.

L'évaluation des requêtes s'appuie sur la fonction `successeur` qui permet de contacter un pair contenant des valeurs supérieures aux valeurs stockées par le pair invoquant la fonction. La portabilité de la solution est ainsi restreinte.

4.4 Modèle relationnel

Le deuxième groupe correspond aux travaux qui considèrent le système pair à pair comme une base de données relationnelle distribuée. Ces travaux permettent

l'évaluation des requêtes qui utilisent des opérateurs de sélection, projection, jointure et agrégation. Les solutions combinent des algorithmes classiques et des fonctions de hachage.

Cette section présente PIER [Huebsch 2005], DHTop [Akbarinia 2007] et FCQ [Triantafillou 2003]. La description inclut la gestion de données, le contenu des index et les hypothèses ajoutées aux couches sous-jacentes pour permettre l'évaluation de requêtes. Une importance particulière est donnée au support de requêtes complexes. Notons que l'évaluation d'un opérateur simple, tel que la sélection avec une condition d'inégalité, est difficile et coûteuse dans un environnement P2P. Donc, contrairement à l'évaluation au sein d'un SGBD, une telle requête pourrait être considérée ici comme complexe. Notons que la jointure qui a priori est considérée comme complexe peut bénéficier dans ce contexte d'une implémentation utilisant la fonction de hachage. De tels algorithmes sont bien connus dans les SGBD relationnels.

PIER

PIER [Huebsch 2003, Huebsch 2005] est un moteur de recherche massivement distribué. Il suppose l'existence des services DSS et DLS avec des propriétés particulières. Le service DSS doit permettre la notification des migrations des données dans le réseau, suite aux changements de la configuration du système. De plus, le service DLS doit fournir la fonctionnalité de suppression de données. Ces propriétés permettent de nouvelles fonctionnalités comme par exemple la gestion de données temporaires. Les données restent stockées seulement pour un temps défini et l'utilisateur doit renouveler leur existence.

PIER propose un grand nombre d'opérateurs relationnels ainsi que des algorithmes pour les évaluer. Il propose divers index logiques de différents niveaux en visant l'optimisation des différents types de requêtes. L'adaptation des algorithmes basés sur des fonctions de hachage permet l'évaluation des opérateurs relationnels incluant des termes d'inégalité. Néanmoins, le nombre de pairs à contacter et le nombre de messages générés dépendent du nombre des n-uplets des tables participant à l'évaluation de la requête et du nombre de pairs inclus dans le groupe. Ce sont des éléments qui peuvent entraîner des problèmes de performance pour l'évaluation des requêtes. Des optimisations pour l'évaluation de requêtes incluant des termes d'inégalité sont proposées en utilisant *Prefix Hash Tree* [Ramabhadran 2004] (PHT) comme index de type *range predicate*.

DHTop

DHTop [Akbarinia 2007] met l'accent sur l'évaluation performante de requêtes incluant l'opérateur top-k [Michel 2007]. Ce problème a été fortement étudié pour des systèmes centralisés et distribués mais peu dans des contextes de large échelle comme les systèmes P2P basés sur une DHT.

Les structures d'indexation présentées ont besoin d'une connaissance préalable d'histogrammes permettant d'estimer la distributions des valeurs des attributs qui

sont inclus dans les fonctions de score. Cela peut générer des problèmes car l'information à stocker n'existe pas forcément de manière préalable au chargement des données. De plus, la stratégie de duplication proposée considère une connaissance du comportement des utilisateurs, qui peut être difficile à obtenir, et des problèmes de surcharge de stockage peuvent apparaître.

FCQ

FCQ [Triantafillou 2003] est un canevas pour l'évaluation de requêtes SQL sur des systèmes P2P qui stockent des tables relationnelles. Il fournit les opérateurs de sélection, de projection, de jointure, de *group by*, etc.

FCQ utilise la même structure d'indexation pour évaluer tous les types de requêtes (égalité, inégalité et jointure). Ces requêtes portent sur un ou plusieurs attributs. Néanmoins, FCQ introduit des changements dans l'architecture du système et des contraintes sur la fonction de hachage utilisée. En effet, FCQ utilise une couche de super pairs et une fonction de hachage qui préserve l'ordre. De plus, il est nécessaire de connaître au préalable le domaine des valeurs des n-uplets pour désigner les intervalles des valeurs associées aux super pairs.

Comme MAAN, FCQ s'adresse aux attributs numériques et utilise la fonction `successeur` du DSS pour la propagation des requêtes dans les couches des pairs et des super pairs. Cette fonction n'est pas offerte par tous les DSS.

4.5 Données semi-structurées

Cette section concerne des travaux autour de XML. Elle présente les systèmes DIP [Garcés-Erice 2004], KadoP [Abiteboul 2005] et LDS [Galanis 2003] avec leurs choix concernant les données, la manière de les décrire, l'indexation et les processus d'évaluation de requêtes. Un certain nombre d'autres travaux se sont attachés à l'amélioration de ces techniques d'évaluation, par exemple à l'aide de *bloom filter* [Jamard 2007, Abiteboul 2008] ou bien en améliorant les mécanismes d'indexation [Bonifati 2006, Dragan 2006].

DIP

DIP [Garcés-Erice 2004] est une solution d'indexation et d'interrogation de données partagées dans un système P2P. DIP utilise seulement les fonctions minimales `put` et `get` des systèmes P2P structurés ce qui le rend portable.

DIP propose comme optimisation la création d'index de requêtes « intéressantes » ainsi que l'utilisation d'un cache. Ces optimisations permettent respectivement l'évaluation itérative des requêtes et l'amélioration du temps de réponse. Néanmoins, il ne fournit pas de mécanismes pour identifier les requêtes « intéressantes », et la manière de le faire peut affecter les performances et la viabilité de la solution. Par ailleurs, la suppression des données ne fait pas partie des fonctions communes à la plupart des DSS ce qui restreint la portabilité de la solution proposée.

KaDoP

KadoP [Abiteboul 2008, Abiteboul 2005] utilise Active XML [Abiteboul 2002], XML contenant des appels à des services Web. Il a été développé au-dessus de FreePastry [Rice University 2002].

KadoP fournit l'évaluation de requêtes sémantiquement riches en utilisant des concepts et des relations entre concepts. La représentation des index ainsi que l'évaluation proposée sont très proches de LDS présenté au paragraphe suivant.

KadoP propose un mécanisme automatique pour la maintenance des index lors de l'enregistrement de nouvelles données. Par ailleurs, la quantité d'information gérée, les concepts et relations entre concepts peuvent représenter un volume de stockage important et allonger le processus d'évaluation. Des techniques permettant d'améliorer ces points peuvent être trouvées dans [Abiteboul 2008].

LDS

LDS [Galanis 2003] est un service d'indexation de données XML qui utilise XPath pour l'interrogation. LDS est indépendant du système P2P sous-jacent.

Bien que l'indexation proposée permette l'évaluation de tout type de requêtes de manière homogène, les index contiennent une grande quantité de données ce qui peut rendre difficile leur mise en œuvre et leur mise à jour. Le système *XPath for P2P* (XP2P) présenté dans [Bonifati 2006] propose d'améliorer l'indexation en utilisant uniquement des fragments de document XML. De plus, dans LDS, les index sont affectés par l'arrivée et le départ des pairs, et cela peut être problématique dans un système très dynamique.

L'utilisation du réseau est moindre comparée aux stratégies d'évaluation de requêtes d'égalité proposées par d'autres travaux qui stockent dans l'index les identifiants des données. En effet, LDS ne transfère pas les identifiants des objets satisfaisant une partie de la requête. Les réponses intermédiaires contiennent uniquement des identifiants de pairs et non des identifiants d'objets. En contrepartie, LDS introduit une phase supplémentaire pour l'évaluation des requêtes d'égalité en raison du contenu des index. LDS contacte les pairs contenant les index pour trouver les pairs susceptibles d'avoir des réponses et interroge ensuite ces derniers. Dans les autres travaux (e.g. KSS et PinS), il faut contacter uniquement les pairs contenant un index pour trouver la réponse à la requête.

4.6 Conclusion

De nombreux travaux sur la gestion des données dans les systèmes P2P structurés ont été menés ces dernières années. Ces systèmes gèrent de manière performante des pairs et des données dans un contexte très dynamique et hétérogène. Cela a permis de proposer des solutions P2P de partage de données à très grande échelle. On s'est ici principalement focalisé sur les systèmes P2P DHT « classiques » qui constituent un panel très riche.

Dans leur configuration de base, ces systèmes font abstraction du sens des données et de la sémantique utilisée pour coder ce sens. Ils ne permettent donc pas,

sous cette forme originelle, de retrouver de l'information en fonction de son sens. En tirant profit de la connaissance de la sémantique choisie pour coder le sens des données, un certain nombre de systèmes ont été proposés pour améliorer la gestion de données dans ce contexte. Les données partagées, le langage d'interrogation proposé, les domaines des métadonnées diffèrent selon les travaux.

Les métadonnées, qui permettent de décrire le sens associé aux données, peuvent être des mots-clés ou des concepts très structurés. L'évaluation des requêtes de sélection avec des conditions d'égalité s'est généralisée mais l'opérateur d'inégalité est plus rare et plus difficile dans ces environnements. Cette difficulté provient principalement de la distribution massive des données et de leur placement aveugle (sans critère de sens). Pour permettre l'évaluation de ces termes d'inégalité, divers index sont proposés. On trouve des index logiques, dont les entrées ne dépendent pas de la localisation physique des pairs, et aussi des index physiques qui accélèrent les recherches mais sont sensibles à la volatilité des pairs.

De manière générale, les index sont volumineux et des problèmes liés à l'espace de stockage peuvent apparaître. La maintenance des index peut aussi être problématique du fait de leur taille (comme dans LDS et PHT) et de la volatilité des pairs (LDS).

Dans certains cas, l'évaluation de l'opérateur d'inégalité repose sur des caractéristiques particulières de l'architecture du système P2P sous-jacent (existence de super pairs dans FCQ) ou sur l'utilisation de fonctions de hachage qui préservent l'ordre (comme dans MAAN et FCQ). Ces choix affectent la portabilité de la solution ainsi que, dans certains cas, l'efficacité de l'évaluation des requêtes. En effet, le nombre de pairs à contacter pour évaluer une requête d'inégalité dépend de la distribution des données dans le système. Si le domaine théorique utilisé pour définir la fonction de hachage est vaste, et si les valeurs des objets partagés sont très dispersées dans ce domaine, le nombre de pairs à contacter augmente et le temps de réponse également.

Dans la plupart des systèmes, une seule stratégie est disponible pour évaluer une requête. Cette rigidité peut être pénalisante, par exemple lorsque la charge que doit soutenir le système augmente, le système devant évaluer un grand nombre de requêtes qui peuvent être en concurrence avec l'enregistrement de données. Certains systèmes (comme MAAN et PinS) proposent diverses stratégies d'évaluation des requêtes. Néanmoins ils ne proposent pas d'heuristiques solides pour le choix de la stratégie à utiliser à un instant donné.

Ainsi, la recherche par le sens de données stockées dans les systèmes P2P-DHT reste une problématique importante, ouverte et clairement opportune dans ces systèmes qui restent difficiles à maîtriser, à mettre en œuvre et à tester. Ces difficultés sont pour partie dues au grand nombre et à l'autonomie des participants. Ces deux caractéristiques se retrouvent sous des formes différentes dans les grands systèmes à base de capteurs. Le chapitre suivant se propose de montrer comment la problématique de la prise en compte du sens de l'information est traitée dans le cadre de ces systèmes.

Chapitre 5

Donner du sens au flux de données issues de capteurs

Sommaire

5.1 Traitement des données issues des capteurs 40
5.2 Réseaux de capteurs. Wireless Sensor Networks (WSN) . . 41
5.2.1 Traitement des données dans les réseaux de capteurs 41
5.2.2 Caractéristiques des réseaux de capteurs 42
5.2.3 Bases de données capteurs 43
5.2.4 Discussion 45
5.3 Systèmes de Gestion de Flux de Données (SGFD) 45
5.3.1 SGFD vs SGBD et flux de données 46
5.3.2 Requêtes continues sur les flux de données 47
5.3.3 Résistance à la charge 48
5.3.4 Discussion 49
5.4 Intégration SGFD et réseaux de capteurs 49
5.5 Conclusion 51

L'apparition des flux de données de capteurs entraîne un mouvement similaire à celui observé lors de la création du modèle relationnel. On observe un glissement des traitements, à l'origine spécifiques aux capteurs utilisés, vers des traitements plus généraux qui sont conçus pour manipuler l'information en fonction du sens qui lui est associé. Ce mouvement passe par une structuration des données qui permet de capturer le sens de l'information transmise par les capteurs. Cette structuration, tout en gardant une certaine généricité, doit prendre en compte la spécificité de ces données qui sont produites par des dispositifs instables, en flux continus trop intenses pour être traités de manière classique par les SGBD. Ce chapitre présente un panorama des solutions mises en œuvre pour traiter ces problèmes [Gürgen 2010b].

Contributions au domaine

Nos contributions dans ce domaine ont débuté avec les travaux autour de la thèse de Levent Gürgen [Gürgen 2007], co-encadrement Claudia Lucia Roncancio et en coopération avec Vincent Olive (Orange Labs – France Telecom R&D). Ces travaux ont abouti à des propositions en termes d'architecture et de modèle de flux [Gürgen 2005a, Gürgen 2005b, Gürgen 2006a] ainsi qu'à un intergiciel SStreaMWare [Gürgen 2008c, Gürgen 2008a]. L'aspect le plus original de ce travail est de

mettre en lumière une problématique de contrôle de concurrence liée à l'administration des capteurs [Gürgen 2006b, Gürgen 2008b].

Ces travaux se poursuivent, dans le cadre du projet ANR Jeunes Chercheurs TransCap (01/2008 – 10/2011), dont je suis responsable, avec les thèses d'Amin Cherbal et de Loic Petit (co-encadrement Claudia Lucia Roncancio).

La thèse d'Amin Cherbal porte plus particulièrement sur les aspects liés à l'administration [Gürgen 2009a, Gürgen 2010a] (en collaboration avec le NII de Tokyo). Celle de Loic Petit (en collaboration avec François-Gaël Ottogali de Orange Labs–France Telecom R&D) explore les problématiques liées aux modèles utilisés pour interroger les données provenant de capteurs avec la définition de l'algèbre de flux Astral [Petit 2010].

Nos recherches se poursuivent aussi avec des travaux sur l'économie d'énergie sur les capteurs [Sagen 2010], d'une part en collaboration avec Monash University et La Trobe University (Melbourne) et d'autre part avec le LCIS de Valence au travers du BQR ARTEco.

5.1 Traitement des données issues des capteurs

Avec l'avancement des technologies, en particulier leur miniaturisation et la communication sans fil, les dispositifs de calcul deviennent de plus en plus autonomes, petits, peu coûteux et puissants. Les capteurs sont devenus partie intégrante de notre vie de tous les jours, ils facilitent la surveillance des événements physiques survenant dans notre environnement. Il existe de nombreux dispositifs de capture : température, pression, composition chimique, localisation, dispositif audiovisuel, lecteur d'étiquettes RFID, etc. Ces dispositifs peuvent être sans-fil, autonomes ou non. Ils sont utilisés dans beaucoup d'applications scientifiques et industrielles [Bonivento 2006] : applications médicales [Gao 2005, Lorincz 2004], surveillance de l'environnement pour des applications scientifiques [Mainwaring 2002, Biagioni 2002, Ulmer 2003], pour le trafic urbain [Coleri 2004] ou encore des applications domotiques [Estrin 2002] [1].

Le traitement des données générées par les capteurs, a apporté de nouveaux problèmes. Parmi ces problèmes, la gestion de l'énergie, l'évaluation de requêtes déclaratives sur des flux de données de capteurs, la gestion de l'hétérogénéité, et le passage à l'échelle des systèmes de capteurs [Estrin 1999, Akyildiz 2002, Golab 2003, Dejene 2008]. Ce chapitre présente les différents types de systèmes qui ont été conçus pour permettre l'exploitation de l'information issue des capteurs.

Trois grandes classes de systèmes sont présentées, chacune de ces classes correspondant à des choix différents d'architecture, mais toutes ont pour objectif principal d'offrir des capacités de gestion de l'information qui ne fassent pas complètement abstraction du sens associé à l'information.

– *Les réseaux de capteurs* [Akyildiz 2002] sont des réseaux ad hoc composés de capteurs intelligents ayant une certaine autonomie d'énergie ainsi qu'une

1. Pour plus d'exemple, le lecteur pourra se référer a [Chong 2003] et [Römer 2004]

capacité pour stocker, traiter et transmettre leurs mesures (température, pression, localisation, etc.) grâce à un réseau sans fil. En introduisant la notion de « bases de données de capteurs », ils permettent une prise en compte du sens au plus proche de la génération de données tout en exploitant au mieux les capacités de calcul des capteurs. La section 5.2 présente ces travaux.

– *Les systèmes de gestion de flux de données (SGFD)* [Golab 2003] sont des systèmes qui permettent d'exploiter au mieux le sens associé aux flux de données. Un *flux de données* est une séquence non bornée de données ordonnées par leur estampille de temps. Ces flux sont générés de manière continue par des sources (logs de transactions web, applications financières, trafics de réseau, etc.). Les SGFD peuvent aussi être utilisés pour traiter les données des capteurs sous forme de flux. Cette approche est décrite dans la section 5.3.
– *Les systèmes hybrides* concernent l'intégration des deux premières approches pour tenter de bénéficier de leurs avantages respectifs pour des problématiques spécifiques. L'objectif est de mieux traiter les aspects liés au passage à l'échelle, à la gestion de l'hétérogénéité et à la dynamicité tout en permettant l'évaluation de requêtes sémantiquement riches. Ces approches hybrides sont discutées dans la section 5.4.

La section 5.5 conclut le chapitre en posant la question de la place de ces technologies au sein de systèmes plus vastes que sont les systèmes d'information ubiquitaires.

5.2 Réseaux de capteurs. Wireless Sensor Networks (WSN)

Cette section définit et présente les caractéristiques essentielles des réseaux de capteurs (sections 5.2.1 et 5.2.2 respectivement). La section 5.2.3 présente des solutions de plus haut niveau pour l'interrogation, qui adoptent une vision « bases de données » des réseaux de capteurs. Une discussion en section 5.2.4 clôt cette présentation.

5.2.1 Traitement des données dans les réseaux de capteurs

Un *réseau de capteurs* est un réseau *ad hoc*, autoconfigurable, sans infrastructure fixe, composé de nœuds ayant une certaine capacité de calcul, de stockage, et de communication sans fil. Des capteurs dispersés dans l'environnement mesurent des caractéristiques physiques ou détectent des événements. Les données sont envoyées à une ou plusieurs stations de base qui jouent le rôle d'interface entre les applications et les capteurs (voir la figure 5.1).

Contrairement aux réseaux traditionnels, utilisés pour divers buts, la construction des réseaux de capteurs n'est motivée que par le but de « collecter des données de capteurs sans avoir besoin d'une infrastructure fixe ». Le routage et la collecte des données sont étroitement liés, et le terme « réseaux de capteurs » est très souvent

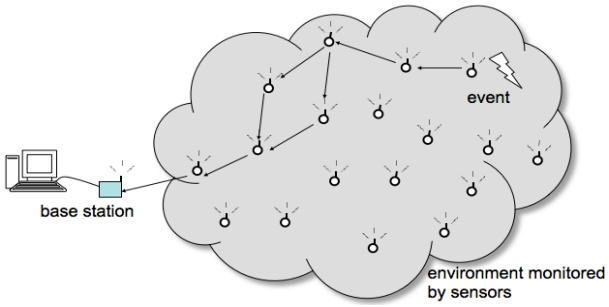

FIGURE 5.1 – Réseaux de capteur : chemin des données vers la station de base.

utilisé pour désigner une plateforme pour la collecte de données. Signalons également le rôle essentiel du système d'exploitation des capteurs dans la collecte de données. Un des buts premiers de ces systèmes d'exploitation est d'abstraire l'acquisition des mesures et de faciliter la transmission des données. Son rôle est d'autant plus important qu'il n'y a pas toujours « d'évaluateur de requêtes » sur les capteurs [2].

Collecter et traiter les données brutes des capteurs pour les transformer en « information » n'est pas une tâche facile. De plus, les caractéristiques spécifiques des réseaux de capteurs (voir la section suivante) introduisent des défis qui doivent être relevés conjointement par différents domaines de l'informatique : réseaux, systèmes d'exploitation, bases de données, systèmes distribués, etc. Une séparation claire de ces domaines est difficile dans le cadre des réseaux de capteurs.

5.2.2 Caractéristiques des réseaux de capteurs

Les réseaux de capteurs partagent certaines des caractéristiques des réseaux *ad hoc* sans fil mais possèdent des caractéristiques spécifiques, notamment au niveau des ressources disponibles, du mode de communication et de l'adressage. Les caractéristiques principales des réseaux de capteurs sont les suivantes.

Ressources limitées : la taille des capteurs peut varier de quelques centimètres jusqu'à quelques millimètres cubes. Naturellement, ceci limite les capacités de calcul, de stockage et de communication, malgré les remarquables progrès de miniaturisation. Ici, l'énergie est la ressource la plus précieuse. La durée de vie d'un système entier dépend des réserves d'énergie de chacun des capteurs. En effet, comme les capteurs participent au routage, à l'agrégation des résultats et à l'autoconfiguration du réseau, l'indisponibilité de certains nœuds peut provoquer le dysfonctionnement du système.

Grand nombre de capteurs : la diminution continue des coûts de production permet désormais de déployer de nombreux capteurs. D'une part, le nombre croissant

2. Une discussion sur les systèmes d'exploitation de capteurs dépasse le cadre de document. Le lecteur intéressé peut consulter [Levis 2004], [Han 2005], [Bhatti 2005], TinyOS [Levis 2004].

des nœuds augmente la redondance, améliore la tolérance aux fautes et la précision des mesures, mais d'autre part, il complique la tâche d'auto-organisation, augmente le nombre de données à traiter et le nombre de messages échangés. Les protocoles de routage et les techniques de gestion de données doivent donc prendre en compte le facteur « grande échelle » des réseaux de capteurs [Estrin 1999].

Routage data-centric : contrairement aux réseaux traditionnels, les nœuds dans les réseaux de capteurs peuvent ne pas posséder une adresse ou un identifiant unique. Les applications dites *data-centric* [Estrin 1999] ne s'intéressent pas en général à des capteurs spécifiques, mais plutôt aux données qu'ils génèrent. Par exemple, une requête de style `« Quelle est la mesure de température du capteur avec l'ID 3424 »` n'a pas beaucoup d'intérêt pour une application, alors qu'une requête telle que `« A quel endroit, se trouvent les capteurs qui mesurent plus de` $40°C$ `»` sera plus significative. Les sources des données sont identifiées par des propriétés (dans l'exemple, la localisation et la température), d'où le terme *data-centric*. Les mécanismes de routage constituent donc la base de l'évaluation de requêtes et de la prise en compte du sens associé aux données.

5.2.3 Bases de données capteurs

Historiquement les technologies utilisées pour l'interrogation des capteurs étaient spécifiques à l'application : chacune avait ses propres capteurs avec une méthode d'interrogation et/ou des protocoles propres influençant le routage. Cependant, la multiplication des applications de capteurs a fait apparaître le besoin d'une mise en commun des moyens d'interrogation par des solutions génériques. Les « bases de données capteurs » [Bonnet 2001, Madden 2002b] tentent de répondre à ce besoin, en fournissant une approche déclarative pour décrire et traiter les requêtes. Cette évolution est similaire à celle des systèmes de gestion de bases de données (SGBD) qui a eu lieu il y a maintenant plus de trois décennies et qui permet de découpler le stockage des données de leur traitement.

Une base de données capteurs *est une combinaison des données persistantes contenant l'information sur les capteurs (ID, localisation, etc.) et des données instantanées et transitoires qui concernent les mesures relevées de l'environnement surveillé.*

Cette approche considère chaque capteur comme une source de données. Ainsi, le réseau de capteurs est vu comme une BD distribuée. Les requêtes sont introduites dans le réseau à partir d'une station base et elles sont propagées avec des mécanismes de routage. Les requêtes sont évaluées par les capteurs en tirant profit de leurs capacités de calcul et de stockage. Chaque capteur envoie ses données si et seulement si elles sont conformes à la requête reçue. La charge d'évaluation des requêtes est distribuée, ce qui diminue la taille et le nombre des données circulant dans le réseau. Cependant, les techniques d'évaluation des requêtes, telles qu'elles sont utilisées pour les bases de données classiques, doivent être revues pour les capteurs. Leurs ressources limitées et la nature continue, dense, imprécise, et parfois bruitée, des

données doivent être prises en compte par ces nouvelles techniques.

Interrogation déclarative

De nombreux travaux de recherche sur la gestion des données dans les réseaux de capteurs ont été menés. TinyDB [Madden 2005] et COUGAR [Yao 2002] sont les premiers à adopter l'approche déclarative d'acquisition de données. L'approche la plus utilisée s'appuie sur le modèle relationnel et utilise des requêtes *SQL-like* [Madden 2005]. Les données des capteurs sont sous forme de n-uplets conformes à un schéma qui est généralement lié à l'application. Par exemple, les n-uplets produits par un capteur de température peuvent être de la forme *<sensorId, location, temperature, timestamp>*. Les n-uplets d'un ensemble de capteurs forment une table virtuelle qui est partitionnée horizontalement sur les capteurs du réseau. Cette table est mise à jour, de manière continue, par les mesures des capteurs. Les requêtes doivent donc être continues pour prendre en compte les valeurs les plus récentes.

Les requêtes continues *sont évaluées au fur et à mesure sur les données très variables issues des capteurs afin de refléter l'état le plus actuel de l'environnement surveillé.*

En effet, contrairement aux requêtes traditionnelles qui portent sur l'état actuel d'une base de données, les *requêtes continues* doivent être évaluées de manière à informer, en temps réel, des changements enregistrés par les capteurs. Ceci est particulièrement vrai pour les applications de surveillance. Par exemple, avec TinyDB il est possible d'indiquer la périodicité de la remontée des résultats et la durée de vie de la requête. La figure 5.2 donne l'identificateur des capteurs de température qui mesurent plus de 40° dans la chambre C toutes les 3 secondes pendant 2 minutes.

```
SELECT sensorId, temperature
FROM sensors
WHERE temperature > 40 AND room = C
SAMPLE PERIOD 30s FOR 360s
```

FIGURE 5.2 – TinyDB query

Les travaux [Madden 2005] permettent aussi d'indiquer un événement qui déclenche l'évaluation d'une requête pour mieux cerner les mesures pertinentes. Des fenêtres temporelles s'appliquant à un capteur particulier peuvent être utilisées. Une telle fenêtre contient les mesures récentes relevées pendant un intervalle de temps.

Une manière d'économiser l'énergie pendant la collecte des données est de les agréger pour limiter le nombre de messages nécessaires à leur remontée vers la passerelle d'accès. Cette approche est décrite dans la plupart des propositions récentes [Sharaf 2004]. Il s'agit de profiter de la nature multisaut des protocoles de routage. Chaque capteur agrège sa mesure avec celle de son prédécesseur avant de transmettre la réponse au capteur suivant. Cette agrégation peut soit utiliser un opérateur classique (somme, moyenne, minimum, etc.) soit effectuer un simple ajout d'une mesure à une autre dans un même paquet réseau. Ainsi le nombre de messages envoyés est minimisé et l'énergie consommée est moindre comparée à une solution où tous les capteurs communiquent directement avec la passerelle [Krishnamachari 2002].

En effet, l'exécution des instructions consomme moins d'énergie que la transmission sans fil des données. Il est montré dans [Madden 2002b] que l'énergie consommée pour transmettre un bit est équivalente à celle consommée pour exécuter environ 800 instructions. Il faut noter qu'avec cette technique, il n'est pas possible de calculer les fonctions d'agrégation qui ont besoin de l'ensemble des mesures pour être évaluées (par exemple histogramme, médiane ou quartile).

5.2.4 Discussion

Cette section a donné un aperçu de l'état actuel des travaux sur les réseaux de capteurs, en particulier sur les aspects liés à l'interrogation. Une vue « base de données » sur les capteurs a été proposée pour autoriser l'interrogation déclarative des capteurs. Ces propositions cherchent à adapter des techniques utilisées par les SGBD. Le but est de pouvoir découpler les besoins applicatifs et les aspects d'interrogation afin de faciliter le partage des capteurs entre applications. La recherche sur les réseaux de capteurs continue à explorer de nouvelles problématiques, telles que, la qualité de service [Chen 2004], la gestion de la mobilité [Dantu 2005], l'auto-administration [Liu 2003], l'hétérogénéité [Awan 2007], la fiabilité [Wan 2002] et la sécurité [Perrig 2004]. Les solutions doivent considérer les contraintes d'énergie des capteurs et ceci ne semble pas pouvoir changer dans un futur proche.

Les capteurs commencent à être utilisés dans des domaines d'application demandant de plus en plus de garantie de fiabilité, d'intégrité, de sécurité et de privacité. Parmi eux, les applications industrielles, médicales et militaires. Ces problématiques gagneront donc en importance. Le passage à l'échelle est une autre problématique émergente et importante à considérer puisque la réduction des prix des capteurs encourage une large utilisation. Ceci génère aussi la question de l'administration à grande échelle des parcs de capteurs [Gürgen 2005a, Gürgen 2009a]. En effet, l'administration demande des modifications qui peuvent mettre en cause la cohérence de ces systèmes, elles doivent donc être faites de manière à conserver l'intégrité du système [Gürgen 2006b, Gürgen 2008b].

5.3 Systèmes de Gestion de Flux de Données (SGFD)

Nous abordons des solutions qui, bien qu'elles puissent supporter le traitement des données de capteurs, sont conçues pour l'interrogation de données quelconques générées continuellement en flux [Golab 2003]. Les travaux sur ce thème convergent pour définir les flux de données de la manière suivante :
Un flux de données *est une séquence non bornée de données ordonnées par leur estampille de temps.*

L'interrogation de tels flux de données par les SGBD n'est pas immédiate. En section 5.3.1 nous discutons de différences entre les SGFD et les SGBD, ce qui nous amène à préciser la notion de flux. La section 5.3.2 introduit les opérateurs nécessaires pour pouvoir bénéficier d'une interrogation continue et riche sur les flux. De plus, dans ces systèmes, les requêtes doivent être traitées en quasi-temps réel et

l'aspect résistance à la charge générée par un grand nombre de sources de données est un point très important discuté en section 5.3.3. La section 5.3.4 propose une synthèse sur ces travaux.

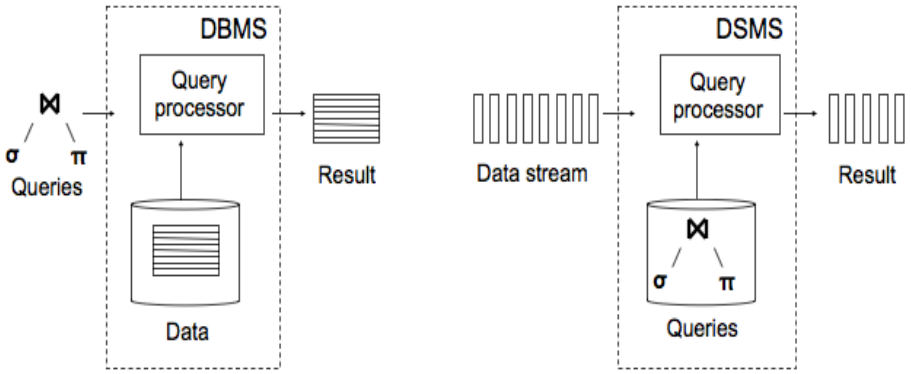

FIGURE 5.3 – DBMS : données persistantes, requêtes transitoires / DSMS : données transitoires, requêtes persistante.

5.3.1 SGFD vs SGBD et flux de données

Les SGFD et les SGBD fournissent aux applications des moyens génériques de gestion des données. Cependant, il existe des différences fondamentales au niveau de la nature des données et des requêtes traitées. Les SGBD classiques offrent des moyens d'interroger des données *persistantes*. Une requête est *transitoire* dans le sens qu'elle est évaluée « une fois » sur les données au moment de la réception de la requête puis elle est éliminée. Les données traitées par les SGFD sont des flux *transitoires* et temporairement stockés dans la mémoire, en général dans une file d'attente. Puisque le volume de données qui peut être stocké est limité, les données sont simplement éliminées après leur traitement. Les requêtes doivent être *persistantes* et s'évaluer continuellement afin de prendre en compte les nouvelles valeurs arrivant au système (cf. figure 5.3).

Parmi les applications les plus populaires nous pouvons citer le contrôle du trafic réseau [Cranor 2002], l'analyse de journaux transactionnels (transactions web, bancaires ou de télécommunication) [Cortes 2000], le suivi d'actions sur des pages web dynamiques [Chen 2000], ou encore la gestion des données de capteurs [Arasu 2003b]. Ces applications ont motivé les propositions de plusieurs SGFD dans la communauté académique, parmi lesquelles, STREAM [Arasu 2003b], Aurora [Abadi 2003], et TelegraphCQ [Chandrasekaran 2003]. Les offres commerciales issues de ces recherches commencent aussi à émerger [Coral8 2008, StreamBase 2008, Aleri 2008, Almagamated 2008]. Pour les applications d'affaires, le temps de traitement de données est très critique afin de permettre une prise de décision et des réactions rapides.

Pour prendre en compte le sens des données, les SGFD adoptent un modèle de

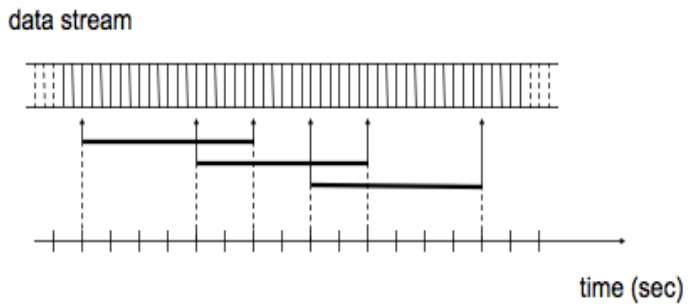

FIGURE 5.4 – Exemple d'une fenêtre de taille 6 secondes qui glisse toute les 4 secondes.

flux de données inspiré du modèle séquentiel [Seshadri 1996] qui est une extension du modèle relationnel. Les n-uplets sont ordonnés par la valeur d'un attribut (en général par l'estampille). Ils sont conformes à un schéma qui dépend de l'application. Par exemple, pour une application de surveillance environnementale [Stream 2008], les flux de données contiennent des mesures faites par les capteurs ainsi que d'autres informations comme l'identifiant et la localisation du capteur. Les requêtes continues permettent une surveillance en temps réel pour, par exemple, détecter des anomalies. Voici un exemple de schéma pour représenter le sens des données de capteurs : $sensors(id, location, temperature, timestamp)$ $\{< i, l, t, ts > \in sensors$ ssi le capteur d'identifiant i et de localisation l mesure la température t au moment $ts.\}$

La représentation relationnelle est adoptée par la plupart des propositions du domaine. Cependant, le caractère infini des flux introduit des problèmes pour l'exécution d'opérateurs pourtant classiques dits *bloquants*. En effet, ils nécessitent en entrée un ensemble fini de données. Une des solutions se base sur la gestion de « fenêtres » appliquée aux flux. Ainsi, les propositions diffèrent quant aux techniques d'évaluation des requêtes, de représentation des fenêtres et de gestion de la charge générée par les flux. Ces techniques sont présentées dans les deux sections qui suivent.

5.3.2 Requêtes continues sur les flux de données

Les requêtes sur les flux de données doivent s'évaluer continuellement afin de prendre en compte les nouvelles valeurs s'ajoutant aux flux. Comme dans les réseaux de capteurs, les requêtes dans les SGFD sont *continues*. Cependant, il faut noter une petite nuance sur *la périodicité* d'exécution des requêtes. En effet, dans les réseaux de capteurs cette périodicité est définie par le taux d'envoi de données des capteurs, tandis que les SGFD ne maîtrisent pas ce taux d'envoi. En conséquence, la périodicité n'est pas à définir dans les requêtes, sauf pour celles qui s'évaluent sur les *fenêtres périodiquement glissantes* ([Petit 2010] et 5.5). Les réponses aux requêtes sont des flux. Les requêtes sont formées d'un ensemble d'opérateurs, et chaque opérateur consomme des données en entrée, exécute des opérations, et produit des résultats

de manière continue. Un flux résultat peut servir d'entrée pour un autre opérateur.

Des variantes de SQL sont fortement utilisées dans ce contexte (e.g. STREAM [Arasu 2003b] et TelegraphCQ [Chandrasekaran 2003]). Cependant, même si, au niveau syntaxique, les requêtes sont similaires à celles destinées aux SGBD traditionnelles, leur évaluation est bien différente. L'évaluation se fait sur le flux de n-uplets et non sur une relation. Les langages ne sont pas tous formalisés. L'effort le plus important vient de STREAM qui propose un formalisme pour son langage CQL (*Continuous Query Language*) [Arasu 2003a, Jain 2008].

Comme alternative à l'approche SQL étendue Aurora [Abadi 2003] propose la construction des requêtes *via* une interface graphique où l'on place des boîtes représentant des opérateurs et des flèches représentant des flux de données. Une approche semi-structurée vient du projet Niagara [Chen 2000], où le flux de documents XML est interrogé par un langage similaire à XML-QL [Deutsch 1998].

```
SELECT AVG(temperature)
FROM sensors [RANGE 30]
```

FIGURE 5.5 – Requête avec une fenêtre dans STREAM.

5.3.3 Résistance à la charge

Le rythme d'arrivée des flux de données n'est pas maîtrisé par les SGFD. La nature imprédictible du taux de délivrance des n-uplets implique que le système soit capable de s'adapter dynamiquement en cas de surcharge. Si le système n'est pas assez rapide pour traiter tous les n-uplets d'un flux, plusieurs solutions peuvent être envisagées. Par exemple, il est possible d'utiliser des files d'attente, et en cas de débordement de ces files, ignorer certains n-uplets ou agréger des données (par exemple par une moyenne ou un histogramme). Ainsi STREAM [Widom 2003] sacrifie l'exactitude des réponses en proposant des réponses approximatives aux requêtes pour mieux gérer les ressources du système. La ré-optimisation dynamique des requêtes pour s'adapter aux changements du système et des flux est possible grâce au moteur d'évaluation adaptatif StreaMon [Babu 2004]. Pour la même raison, TelegraphCQ utilise les *eddies* [Avnur 2000]. Un *eddy* est un routeur de n-uplets vers les opérateurs selon une politique qui doit faire passer les n-uplets par tous les opérateurs de la requête. L'*eddy* peut changer dynamiquement l'ordre d'exécution des opérateurs et ainsi permettre une ré-optimisation continue des requêtes. Aurora utilise un mécanisme de délestage (*load shedder*) [Tatbul 2003] qui, en cas de surcharge du système, ignore certains n-uplets du flux. Le *shedding* se fait aléatoirement ou sémantiquement selon l'exigence de QdS donnée par l'application.

Pour finir, notons que Aurora, TelegraphCQ et STREAM proposent des mécanismes permettant un partage des requêtes entre les flux et un partage des flux entre requêtes. Ceci est particulièrement important pour une bonne utilisation des ressources.

5.3.4 Discussion

Généralement, dans les systèmes de gestion de flux actuels, les données sont envoyées vers un serveur central où se déroule le traitement des données, d'où leur exposition au problème de surcharge. En effet, le défi principal de ces systèmes est de résister au nombre croissant de n-uplets et à leur taux d'arrivée, ainsi qu'au nombre de requêtes à évaluer. Des *SGFD distribués* sont proposés [Abadi 2005, Logothetis 2008, Xiong 2007] pour faire face à ce problème. Dans un tel contexte l'importance de l'optimisation devient très forte [Madden 2002c]. L'optimisation de requêtes dans les SGBD traditionnels s'appuie sur des connaissances concernant la structure physique et la distribution des données. L'accès aux données se fait de manière directe et les réponses aux requêtes sont précises. Ceci diffère de l'accès aux flux de données qui est séquentiel, en général le taux d'arrivée des données n'est pas maîtrisé et les données sont parfois redondantes et bruitées. Les techniques classiques d'optimisation ne sont donc pas applicables pour les requêtes sur les flux de données. L'optimisation doit aussi être continue et dynamique pour s'adapter aux conditions variables de ces systèmes distribués [Seshadri 2006].

Les techniques de traitement de flux de données sont sans doute très utiles dans le contexte des capteurs. Le problème de surcharge peut être réduit en exploitant les capacités d'évaluation de requêtes des capteurs et/ou les proxies gérant les capteurs. Cette idée est exploitée par les systèmes hybrides présentés dans la section suivante.

5.4 Intégration SGFD et réseaux de capteurs

Les réseaux de capteurs et les systèmes de gestion de flux de données sont deux familles de solutions qui donnent naissance à une troisième : les systèmes hybrides. Ces derniers est de tenter d'intégrer les avantages des deux premières approches pour mieux répondre aux problématiques de l'interrogation d'*un grand nombre* de capteurs *hétérogènes* et de l'évaluation d'un nombre *maximum* de *requêtes complexes* qui filtrent, agrègent et intègrent les flux de données provenant des capteurs. Certains de ces systèmes visent des capteurs spécifiques à un domaine d'application, tandis que d'autres visent des capteurs variés allant des microcapteurs à des capteurs de plus grande taille comme des stations de météo.

Ces architectures hybrides sont principalement construites sur un principe qui suppose un regroupement géographique des capteurs. Les données de ces capteurs sont ensuite traitées par un vaste système distribué dont l'architecture doit faciliter le passage à l'échelle. L'hétérogénéité des capteurs est cachée grâce au choix d'un modèle commun qui permet de traiter l'information en fonction du sens associé aux données fournies par les capteurs

On se propose ici de comparer, pour l'évaluation des requêtes complexes, les avantages et les inconvénients des réseaux de capteurs comparés aux SGFD puis de montrer le positionnement des systèmes hybrides. Cette analyse peut être faite selon deux autres aspects importants : le passage à l'échelle et la gestion de l'hétérogénéité des capteurs [Gürgen 2010b]. Cette comparaison se base sur l'étude de sept systèmes,

à savoir, Fjords [Madden 2002a], Borealis [Abadi 2005], IrisNet [Gibbons 2003], Hourglass [Shneidman 2004], GSN [Aberer 2006], HiFi [Franklin 2005b] et SStreaMWare [Gürgen 2008a].

Réseaux de capteurs et SGFD Les solutions de type réseaux de capteurs utilisent des algorithmes relativement complexes pour la coopération des sources. Le routage *ad hoc* et la synchronisation des sources apportent une surcharge de travail à des sources ayant des ressources limitées. De plus, certains opérateurs tels que les jointures entre les données de différents capteurs et certaines agrégations qui nécessitent intégralité des mesures (histogramme, médiane, quartile, etc.) ne peuvent pas s'exécuter « dans le réseau ».

Au contraire, l'exécution centralisée des SGFD ne nécessite pas de tels algorithmes complexes pour l'acquisition de données. Les serveurs de SGFD sont des machines puissantes (contrairement aux capteurs) qui ont un contrôle sur tous les flux et peuvent évaluer des requêtes complexes sur plusieurs flux.

Systèmes hybrides Les solutions hybrides utilisent des unités relativement puissantes afin d'exécuter des opérateurs qui peuvent nécessiter une certaine puissance de calcul et de stockage importante. Ces unités peuvent aussi centraliser partiellement des flux pour évaluer, par exemple, des jointures entre flux ou des agrégations. Cependant, la possibilité d'exécuter des opérations sur les capteurs n'est pas exclue. Elle est prise en compte par certaines des propositions lorsque les capteurs utilisés le permettent.

L'utilité des requêtes déclaratives pour interroger les capteurs est maintenant bien admise. Les langages *SQL-like* sont très répandus. C'est le cas de HiFi qui utilise un tel langage pour toutes les étapes de son processus *CSAVA* (*Clean, Smooth, Arbitrate, Validate, Analyze*) pendant le traitement des données. GSN, lui aussi, utilise des requêtes SQL embarquées dans ses *descripteurs de déploiement des capteurs virtuels* écrits en XML. Cependant, IrisNet choisit d'utiliser des requêtes *XPath* sur ces descriptions de données structurées en XML. Les *circuits* de Hourglass sont des plans de requêtes exprimés en XML, qui contiennent quelques prédicats pour le service de capteurs désiré. Borealis et SStreaMWare proposent un outil graphique qui permet de créer les plans d'exécution des requêtes à l'aide de graphes : arcs pour les flux, nœuds pour les opérateurs. Et finalement, Fjords n'utilise pas de langage particulier, et affirme qu'il offre une construction architecturale adaptée à tous les langages.

Toutes les solutions étudiées proposent une évaluation distribuée des requêtes. Dans IrisNet, Borealis, Fjords, Hifi et SStreaMWare les adaptateurs et éventuellement les capteurs et/ou réseaux de capteurs participent à l'évaluation des requêtes. Fjords, Borealis et SStreaMWare distribuent l'évaluation *d'une requête* à proprement parler alors que HiFi évalue *différentes requêtes* sur plusieurs niveaux du système. Dans IrisNet, des *Organizing Agents* géographiquement distribués sont chargés de *stocker* des données de capteurs et d'évaluer les requêtes sur ces données.

IrisNet intègre un mécanisme de cache sémantique pour optimiser l'évaluation des requêtes similaires [Deshpande 2003]. Dans GSN, *les capteurs virtuels* évaluent les requêtes sur le(s) flux provenant de(s) capteur(s). Hourglass utilise la publication et la souscription de services distribués pour évaluer des requêtes. Précisons que GSN (respectivement IrisNet) évalue les requêtes style SQL (respectivement XPath) sur des données *persistantes* tandis que les autres évaluent les requêtes sur les flux de données *transitoires*.

5.5 Conclusion

Avec l'émergence des capteurs petits, autonomes et « intelligents », des applications de divers domaines se sont intéressées aux opportunités qu'ils offrent pour améliorer et accélérer le traitement de l'information. Ces innovations ont fait émerger de nouvelles problématiques liées à la gestion de données. Nous avons présenté trois familles de solutions. La première concerne les *réseaux de capteurs*. Ces réseaux exploitent les capacités croissantes des capteurs et leur délèguent une partie de la tâche de l'évaluation de requêtes. La deuxième famille, issue de la communauté des bases de données, sont *les systèmes de gestion de flux de données*. Il s'agit de proposer de nouvelles techniques pour traiter en temps réel des flux infinis. Finalement, la troisième famille vise à intégrer les avantages des deux précédentes et propose des *architectures hybrides*. Nous avons présenté ces familles et donné une comparaison autour des trois problématiques qui se manifestent avec la multiplication des capteurs et la diversification des applications : le passage à l'échelle des systèmes de capteurs, l'hétérogénéité des capteurs et l'évaluation de requêtes complexes.

L'information omniprésente est introduite par de nombreux dispositifs hétérogènes (ordinateurs, équipements maison et industriel, unités mobiles) dont l'interconnexion prend des formes variées. Les capteurs deviennent à leur tour des acteurs importants dans les systèmes d'information ubiquitaires. Dans leur insertion dans ces systèmes, le sens, associé à l'information qu'ils produisent, devient un élément central.

On notera en particulier qu'un certain nombre de recherches sur la gestion de données de capteurs se sont principalement focalisées sur des propositions permettant une interrogation déclarative continue des mesures variables des capteurs. Un tel support est en effet nécessaire pour de nombreuses raisons. Les applications ont besoin de traiter les données de capteurs hétérogènes en quasi temps réel et il est nécessaire d'optimiser l'utilisation des ressources des capteurs. La prise en compte du sens associé à l'information, dans les couches basses des systèmes, permet d'améliorer le traitement de l'information en la traitant au plus près des sources. Cette prise en compte doit cependant conserver un certain degré de généricité pour permettre le partage des infrastructures entre différentes applications.

Outre la fonctionnalité de l'interrogation, d'autres fonctionnalités doivent être introduites pour une bonne gestion des données issues de capteurs. Ces fonctionnalités incluent par exemple des *opérations d'administration* [Jurdak 2009, Gürgen 2008b] des capteurs ou encore un support amélioré de la gestion de la dynamicité (aspect *plug-and-play* [Gürgen 2009b]). En effet, ces opérations peuvent rendre incohérents

les résultats fournis par les requêtes continues en cours d'exécution : en modifiant la configuration d'un système, elles sont susceptibles d'entraîner une modification du sens associé à l'information.

Les aspects liés à la confidentialité et à la sécurité sont aussi extrêmement sensibles. Dans un contexte d'utilisation, tant industriel que grand public, ces propriétés doivent être assurées de manière inhérente par construction. Notons également que la grande échelle dans ce contexte n'est pas encore maîtrisée. Les expériences grandeur nature sont encore difficiles et les efforts dans ce sens sont toujours nécessaires.

Le chemin est donc encore long avant de disposer de véritables « systèmes de gestion de données de capteurs » qui ne recouvrent pas uniquement les aspects interrogation mais bien l'ensemble des aspects liés à la gestion des données issues des capteurs. De plus, c'est l'intégration de tels systèmes dans des systèmes d'information plus classiques qui permettra l'émergence de véritables systèmes d'information ubiquitaires. Cette émergence sera facilitée par la prise en compte du sens associé à l'information dans les couches les plus basses des systèmes. Ainsi, des plates-formes de support d'*espaces de données* (*dataspaces* [Franklin 2005a]) ne peuvent voir le jour que si le sens associé à l'information est mis au centre du traitement de l'information.

Chapitre 6

Conclusion

Nos recherches nous permettent de conclure que la mise à profit du sens associé à une information peut recouvrir différentes formes.

Le sens par la structure.

A minima, la structure du codage qui code le sens de l'information peut être utilisée pour améliorer les traitements. Cette approche « structurale » est classiquement adoptée en bases de données. C'est cette technique qui a été appliquée au sein du projet Gedeon pour réaliser le système de fichiers hybride présenté au chapitre 3. C'est en exploitant la notion d'enregistrement préexistante dans les données que ce système de fichiers offre des fonctionnalités de composition des sources tout en améliorant les performances globales du système (tolérance aux pannes, parallélisme). C'est la même approche qui a été adoptée pour nos travaux sur les flux de données de capteurs présentée au chapitre 5. C'est la structuration des flux qui permet le traitement des requêtes déclaratives et les travaux sur l'administration (évoqués au début du chapitre 5) permettent le partage des infrastructures entre différentes applications.

Cependant, avec cette approche, les résultats sont plus difficiles à obtenir pour des systèmes qui font abstraction de la structure de l'information comme ceux présentés au chapitre 4. Cette difficulté provient sans doute principalement de la nature des réseaux couvrants considérés ici. Ces derniers sont en effet construits de manière complètement orthogonale au sens associé à l'information, ce qui rend, par la suite, difficiles et coûteux le traitement des données.

Le sens émerge d'un réseau d'interactions.

De manière plus large, le sens associé à une information varie en fonction d'un ensemble de contextes complexes. Le sens d'une unité d'information (mot, enregistrement, fichier, mesure d'un capteur,..) est fonction de l'ensemble des interactions que cette unité entretient avec celles qui l'entourent. Elle est aussi fonction des acteurs qui utilisent et traitent cette unité d'information. C'est à l'aide de cette optique plus large « réseau d'interactions » que les travaux présentés dans le chapitre 2 ont permis d'affiner le sens d'un mot. C'est en étudiant les relations qu'un mot entretient avec son entourage que le sens particulier qui lui est associé dans une œuvre est précisé. D'une manière un peu similaire, la coopération des caches sémantique présentée au chapitre 3 permet de mettre à profit les interactions observées entre unités d'information. La coopération des caches de requêtes permet ainsi de

regrouper les enregistrements ayant de grandes affinités qui dépendent de l'intérêt des utilisateurs.

Les systèmes d'information de demain et le sens.

Aujourd'hui, l'explosion du nombre et de la diversité des sources d'information fait surgir des problématiques « système » importantes. La problématique liée à la grande échelle se retrouve sur de nombreuses dimensions : masses de données, nombres de participants et distribution importante tant d'un point de vue quantitatif que géographique. Les contours des systèmes sont de plus en plus flous, leur organisation doit être souple pour supporter des participants qui arrivent/partent à volonté. D'autre part, l'information est fortement hétérogène et doit être traitée de manière continue. La problématique de l'économie d'énergie à différents niveaux prend aussi de plus en plus d'importance. Les recherches exposées dans ce document sont centrées sur la gestion de données dans différents types de systèmes informatiques (traitement de la langue, Grilles, P2P, capteurs). Ces expériences nous ont permis d'explorer différentes caractéristiques qui interviennent dans la construction des grands systèmes d'information ubiquitaires : masse importante d'informations, contours des systèmes peu ou pas stables et enfin gestion des données en flux continus.

On peut considérer que les problématiques soulevées par ces caractéristiques sont en grande partie indépendantes du sens associé aux données. Il peut donc sembler naturel de tenter de s'abstraire du sens pour résoudre ces problèmes. C'est ainsi que le sens associé à l'information est souvent ignoré par les traitements opérés dans les couches basses des systèmes de gestion de l'information. Les difficultés qui surgissent pour résoudre ces problèmes proviennent en partie du fait que l'information et les infrastructures qui la traitent sont par nature mutualisées. Des applications très variées, toutes ayant des centres d'intérêt et des attentes différentes doivent être construites et exécutées à l'aide d'infrastructures communes. C'est ainsi que fleurissent les normes et les formats de description de ressources permettant d'intégrer dynamiquement les sources de données et les services aux infrastructures de traitement de l'information. Ces descriptions elles-mêmes hétérogènes font surgir des types de problèmes qui sont intimement liées au sens associé à ces descriptions. C'est pourquoi, de plus en plus, les systèmes doivent apprendre à tirer profit des informations sur le sens associé à l'information par les applications.

Cette approche et les travaux de recherche que nous avons réalisés ouvre la voie à un certain nombre de perspectives de recherche importantes. Nous nous contenterons ici de n'en citer que quelques-unes que nous distinguerons en fonction de la vision « structurale » ou « réseaux d'interactions » du sens de l'information.

perspectives : approche structurale.

Tirer profit de la structure de l'information peut se faire de différentes manières et dans différents contextes. Par exemple, le modèle de système de fichiers hybrides

peut-être utilisé pour d'autres types de fichiers ayant une structuration plus complexe, par exemple des fichiers XML ou encore des fichiers textuels étiquetés. En ce qui concerne les flux de données, la recherche d'un modèle bien adapté à leur insertion dans les grands systèmes d'information reste indispensable. Les opérations sur les flux, rendues possible par la structuration, ne sont pas encore complètement formalisées et ce travail est aussi indispensable. La recherche de modèles de description des capacités des différents acteurs (capteurs, services, sources de données,...) est aussi importante. Ce sont elles qui vont permettre d'exhiber le sens associé à ces ressources. Ces ressources et leurs descriptions doivent pouvoir s'administrer voir s'auto-administrer et ces opérations d'administration peuvent avoir un impact fort sur le sens qui est associé aux ressources. Ceci pose des problèmes de type concurrence d'accès qui nécessitent aussi d'être résolus.

perspectives : approche réseaux d'interactions.

L'approche cache sémantique et coopération entre caches peut être aussi étudiés et appliqués dans d'autres contextes : Pair-à-Pair, textes, flux de données... Ce dernier contexte est particulièrement intéressant puisque l'approche devrait aboutir au partage d'opérateurs sur les flux et non au partage de résultats pré-calculés. L'étude des interactions entre unité de sens (à l'image de nos travaux réalisés sur les textes) peut aussi être mis à profit pour déterminer le sens associé à une source de données ou encore pour mieux comprendre comment ces ressources sont utilisées. Ainsi, la manière dont les données issues de capteurs interagissent entre elles peut être explicitée et utilisée pour améliorer la consommation de leur ressource énergétique.

L'ensemble des travaux présentés ici montrent que le « sens » associé à l'information est l'une des dimensions critiques qui doit immanquablement être prise en compte dans le développement des couches basses des futurs systèmes de traitement de l'information. C'est cette prise en compte qui permettra de résoudre les principaux problèmes auxquels ils doivent faire face. Il semble donc indispensable de participer à une évolution majeure à venir : l'intégration, au cœur des systèmes, du sens associé à l'information et aux processus de traitement.

Annexe A

Univers lexical de *amour* chez Pierre Corneille's.

Par ordre grammatical, poids décroissant, intervalle de confiance 1%.

A.1 Attraction : mots sur-utilisés

Noms propre : Vénus, Psyché, Léon, Placide, Créuse, Amarante, Phinée, Daphnis

Verbes : céder, éteindre, opposer, allumer, naître, éclater, trahir, paraître, intéresser, aimer, surmonter, succéder, croître, vaincre, étouffer, pardonner, inspirer, tourner, déférer, rallumer, gémir, éprouver, couronner, mériter, brûler, souffrir, presser, faire, fléchir, produire, combattre, seoir, changer, traiter, flatter, vouloir, préférer, devoir, renaître, unir, animer

Substantifs : haine, coeur, tour, jour, retour, excès, amitié, amorce, cour, noeud, estime, tendresse, objet, beauté, séjour, douceur, ardeur, soin, force, pitié, feu, espérance, respect, désir, devoir, loi, idolâtrie, aile, espoir, dépit, mère, excuse, prix, caresse, cause, impatience, faveur, discours, partage, jeunesse, balance, violence, patrie, divorce, feinte, conduite, lien, mérite, raison, transport, froideur, amant, effort, hymen, ambition, maîtresse, passion

Adjectif : conjugal, parfait, paternel, véritable, fort, extrême, tendre, aimé, doux, éteint, chaste, puissant, fou, mutuel, forcé, vertueux, éternel, solide, aveugle, feint, simple, léger, indigne, aimable, beau, ferme

Pronoms : dont, se, qui, il, lui

Adverbes : peu, toujours, plus, d'autant, aussi, ensemble, tant, quelquefois, si, auprès

Articles et déterminants : mon, premier, tel

Prépositions and conjonctions : que, malgré, ni, soit, pour, vers, dans, quand, contre, car

A.2 Répulsion : mots sous-utilisés

Noms propre : Romain, Rome, César, Pompée

Verbes : être, dire, aller, laisser, venir, prendre, arriver, attendre, connaître, penser, pouvoir, sortir, revoir, sembler, amener, craindre, hâter, choisir, trancher, couler, falloir, recevoir, prétendre, défaire, secourir, tomber, plaindre, rougir, marcher, pousser, suivre, fuir, punir, ouvrir, chercher, éviter, perdre, garantir, vanter, mentir, achever, pleurer

Substantifs : seigneur, dieu, ciel, roi, adieu, madame, mot, prince, gens, terre, pied, monsieur, humeur, ordre, heure, ami, homme, comte, mort, lieu, temps, sort, tête, loisir, malheur, mort, avis, coup, combat, soeur, traître, destin, frère, ouvrage, bonheur, guerre, fer, zèle, sang, foudre, bataille, ombre, assassin, main, mal, monstre, père, événement, réponse, fois, oeil, victime, chef, vie, nombre, bourreau, soldat, avenir, affection, fils, pas, châtiment, place, porte, conseil, peuple, épée, parole, effroi, sujet, fortune, état, point, autel, comble, artifice, encens, gendre, assurance, vérité

Adjectif : funeste, prêt, autre, bon, faux, las

Pronoms : tu, vous, ils, en, quoi, nous, y, cela, autre, leur, vous-même

Adverbes : là, demain, bien, vrai, pas, déjà, bas, trop, encore, oui, ici, pourtant, mieux, tout

Articles et déterminants : quel, second, ce, ton, tout, trois

Prépositions and conjonctions : donc, après, voici, jusque, avec, mais, si, sur

Bibliographie

[Abadi 2003] Daniel Abadi, Donald Carney, Ugur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Michael Stonebraker, Nesime Tatbul et Stanley Zdonik. *Aurora : a new model and architecture for data stream management.* VLDB Journal, vol. 12, no. 2, pages 120–139, 2003. 46, 48

[Abadi 2005] Daniel J Abadi, Yanif Ahmad, Magdalena Balazinska, Ugur Cetintemel, Mitch Cherniack, Jeong-Hyon Hwang, Wolfgang Lindner, Anurag S Maskey, Alexander Rasin, Esther Ryvkina, Nesime Tatbul, Ying Xing et Stan Zdonik. *The design of the borealis stream processing engine.* In Second Biennial Conference on Innovative Data Systems Research (CIDR 2005), Asilomar, CA, January 2005. 49, 50

[Aberer 2006] Karl Aberer, Manfred Hauswirth et Ali Salehi. *The Global Sensor Networks middleware for efficient and flexible deployment and interconnection of sensor networks.* Rapport technique LSIR-REPORT-2006-006, Ecole Polytechnique Fédérale de Lausanne (EPFL), 2006. 50

[Abiteboul 2002] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo et R. Weber. *Active XML : A data-centric perspective on web services.* In Demo Proc. of Int'l Conference on VLDB, Hong Kong, Chine, Août 2002. 37

[Abiteboul 2005] S. Abiteboul, I. Manolescu et N. Preda. *Sharing content in structured P2P networks.* In Journées Bases de Données Avancées, Saint Malo, France, Octobre 2005. 36, 37

[Abiteboul 2008] S. Abiteboul, I. Manolescu, N. Polyzotis, N. Preda et Chong Sun. *XML processing in DHT networks.* In IEEE 24th International Conference on Data Engineering (ICDE 2008), pages 606–615, April 2008. 36, 37

[Adjiman 2004] P. Adjiman, P. Chatalic, F. Goasdoué, M.C. Rousset et L. Simon. *Distributed reasoning in a peer-to-peer setting.* In Poster of European Conference on Artificial Intelligence (ECAI), Valence, Espagne, Août 2004. 30

[Akbarinia 2007] R. Akbarinia, E. Pacitti et P. Valduriez. *Processing top-k queries in distributed hash tables.* In Parallel Processing, 12th International Euro-Par Conference, Rennes, France, Août 2007. 35

[Akyildiz 2002] Ian F. Akyildiz, W. Su, Yogesh Sankarasubramaniam et Erdal Cayirci. *Wireless sensor networks : a survey.* Computer Networks, vol. 38, no. 4, pages 393–422, 2002. 40

[Aleri 2008] Aleri. *Aleri, Inc. http ://www.aleri.com/*, 2008. 46

[Almagamated 2008] Almagamated. *Amalgamated Insight, Inc. http ://www.aminsight.com/*, 2008. 46

[Androutsellis-Theotokis 2004] S. Androutsellis-Theotokis et D. Spinellis. *A survey of peer-to-peer content distribution technologies.* ACM Computing Surveys, vol. 36, no. 4, pages 335–371, 2004. 30

[Arasu 2003a] A. Arasu, S. Babu et J. Widom. *The CQL Continuous Query Language : semantic foundations and query execution.* Rapport technique 2003-67, Stanford University, 2003. 48

[Arasu 2003b] Arvind Arasu, Brian Babcock, Shivnath Babu, Mayur Datar, Keith Ito, Rajeev Motwani, Itaru Nishizawa, Utkarsh Srivastava, Dilys Thomas, Rohit Varma et Jennifer Widom. *STREAM : The Stanford Stream Data Manager.* IEEE Data Eng. Bull., vol. 26, no. 1, pages 19–26, 2003. 46, 48

[Avnur 2000] Ron Avnur et Joseph M. Hellerstein. *Eddies : continuously adaptive query processing.* In Weidong Chen, Jeffrey F. Naughton et Philip A. Bernstein, editeurs, Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pages 261–272, USA, May 16-18 2000. The ACM Press. 48

[Awan 2007] Asad Awan, Suresh Jagannathan et Ananth Grama. *Macroprogramming heterogeneous sensor networks using cosmos.* SIGOPS Oper. Syst. Rev., vol. 41, no. 3, pages 159–172, 2007. 45

[Babu 2004] Shivnath Babu et Jennifer Widom. *StreaMon : an adaptive engine for stream query processing.* In SIGMOD '04 : Proceedings of the 2004 ACM SIGMOD international conference on Management of data, pages 931–932, New York, NY, USA, 2004. The ACM Press. 48

[Bhatti 2005] Shah Bhatti, James Carlson, Hui Dai, Jing Deng, Jeff Rose, Anmol Sheth, Brian Shucker, Charles Gruenwald, Adam Torgerson et Richard Han. *MANTIS OS : an embedded multithreaded operating system for wireless micro sensor platforms.* Mobile Networks and Applications, vol. 10, no. 4, pages 563–579, 2005. 42

[Biagioni 2002] E. Biagioni et K. Bridges. *The Application of remote sensor technology to assist the recovery of rare and endangered species.* International Journal of High Performance Computing Applications, vol. 16, no. 3, pages 315–324, 2002. 40

[Bonifati 2006] Angela Bonifati et Alfredo Cuzzocrea. *Storing and retrieving XPath fragments in structured P2P networks.* Data Knowl. Eng., vol. 59, no. 2, pages 247–269, 2006. 36, 37

[Bonivento 2006] Alvise Bonivento, Luca P. Carloni et Alberto Sangiovanni-Vincentelli. *Platform-based design of wireless sensor networks for industrial applications.* In DATE '06 : Proceedings of the conference on Design, automation and test in Europe, pages 1103–1107, Leuven, Belgium, 2006. European Design and Automation Association. 40

[Bonnet 2001] Philippe Bonnet, Johannes Gehrke et Praveen Seshadri. *Towards sensor database systems.* In MDM '01 : Proceedings of the Second International Conference on Mobile Data Management, pages 3–14, London, UK, 2001. Springer-Verlag. 43

[Cai 2003] M. Cai, M. Frank, J. Chen et P. Szekely. *MAAN : A Multi-Attribute Addressable Network for grid information services.* In Proc. of Int'l WS on

Grid Computing, pages 184–191, Phoenix, Arizona, Novembre 2003. 32, 33, 34

[Chandrasekaran 2003] Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael Franklin, Joseph Hellerstein, Wei Hong, Sailesh Krishnamurthy, Samuel Madden, Vijayshankar Raman, Fred Reiss et Mehul Shah. *TelegraphCQ : continuous dataflow processing for an uncertain world.* In CIDR, 2003. 46, 48

[Chen 2000] Jianjun Chen, David J. DeWitt, Feng Tian et Yuan Wang. *NiagaraCQ : a scalable continuous query system for Internet databases.* In SIGMOD'00 : Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pages 379–390, New York, NY, USA, 2000. The ACM Press. 46, 48

[Chen 2004] Dazhi Chen et Pramod K. Varshney. *QoS Support in wireless sensor networks : a survey.* In International Conference on Wireless Networks, pages 227–233, Las Vegas, Nevada, USA, 2004. 45

[Chong 2003] Chee-Yee Chong et S. P. Kumar. *Sensor networks : evolution, opportunities, and challenges.* Proceedings of the IEEE, vol. 91, no. 8, pages 1247–1256, 2003. 40

[Coleri 2004] S. Coleri, S. Cheung et P. Varaiya. *Sensor networks for monitoring traffic.* In 42nd Allerton Conference on Communication, Control and Computing, 2004. 40

[Coral8 2008] Coral8. *http ://www.coral8.com/*, 2008. http ://www.coral8.com/. 46

[Cortes 2000] Corinna Cortes, Kathleen Fisher, Daryl Pregibon et Anne Rogers. *Hancock : a language for extracting signatures from data streams.* In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 9–17, NY, USA, 2000. The ACM Press. 46

[Cranor 2002] Chuck Cranor, Yuan Gao, Theodore Johnson, Vlaidslav Shkapenyuk et Oliver Spatscheck. *Gigascope : high performance network monitoring with an SQL interface.* In Proceedings of the ACM SIGMOD international conference on Management of data, pages 623–623, New York, NY, USA, 2002. The ACM Press. 46

[Dabek 2001] F. Dabek, E. Brunskill, M. Kaashoek, D. Karger, R.Morris, I.Stoica et H.Balakrishnan. *Building peer-to-peer systems with Chord, a distributed lookup service.* In Hot Topics in Operating Systems-VIII, Elmau/Oberbayern, Allemagne, Mai 2001. 33

[Dantu 2005] Karthik Dantu, Mohammad Rahimi, Hardik Shah, Sandeep Babel, Amit Dhariwal et Gaurav S. Sukhatme. *Robomote : enabling mobility in sensor networks.* In IPSN '05 : Proceedings of the 4th international symposium on Information processing in sensor networks, page 55, Piscataway, NJ, USA, 2005. The IEEE Press. 45

[Dar 1996] Shaul Dar, Michael J. Franklin, Bjorn Thorn Jonsson, Divesh Srivastava et Michael Tan. *Semantic data caching and replacement.* In Proc. of the Int. Conf. on VLDB, pages 330–341, 1996. 23

[Dejene 2008] E Dejene, V. Scuturici et L. Brunie. *Hybrid approach to collaborative context-aware service platform for pervasive computing.* Journal of Computers (JCP), vol. 3, no. 1, pages 40–50, 2008. 40

[Denneulin 2010] Yves Denneulin, Cyril Labbé, Laurent d'Orazio et Claudia Roncancio. *Merging file systems and data bases to fit the grid.* In Globe, Conference on Data Management in Grid and P2P Systems, pages 13–25, 2010. 18

[Deshpande 2003] Amol Deshpande, Suman Nath, Phillip B. Gibbons et Srinivasan Seshan. *Cache-and-query for wide area sensor databases.* In Proceedings of the ACM SIGMOD international conference on Management of Data (SIGMOD'03), pages 503–514, New York, NY, USA, 2003. The ACM Press. 51

[Deutsch 1998] A. Deutsch, M. F. Fernandez, D. Florescu, A. Y. Levy et D. Suciu. *XML-QL : A Query Language for XML*, 1998. 48

[d'Orazio 2005] Laurent d'Orazio, Fabrice Jouanot, Cyril Labbé et Claudia Roncancio. *Building adaptable cache services.* In 3rd International Workshop on Middleware for Grid Computing (MGC'05), pages 1–6, Grenoble, Novembre 2005. The ACM Press. 18

[d'Orazio 2006] Laurent d'Orazio, Olivier Valentin, Fabrice Jouanot, Yves Denneulin, Cyril Labbé et Claudia Roncancio. *Services de cache et intergiciel pour grilles de données.* In 22ème journées Bases de Données Avancées, 2006. 18, 23

[d'Orazio 2007a] Laurent d'Orazio. *Caches adaptables et applications aux systèmes de gestion de données répartis à grande échelle.* PhD thesis, Institut National Polytechnique de Grenoble, Decembre 2007. 18, 23, 25

[d'Orazio 2007b] Laurent d'Orazio, Fabrice Jouanot, Yves Denneulin, Cyril Labbé, Claudia Roncancio et Olivier Valentin. *Distributed semantic caching in grid middleware.* In Proceedings of the International Conference on Database and Expert Systems Applications, pages 162–171, Regensburg, Germany, 2007. 18, 24

[d'Orazio 2008] Laurent d'Orazio, Claudia Roncancio, Cyril Labbé et Fabrice Jouanot. *Semantic caching in large scale querying systems.* Revista Colombiana De Computación, vol. 9, no. 1, 2008. 18, 23

[d'Orazio 2010] Laurent d'Orazio, Claudia Roncancio et Cyril Labbé. *Adaptable cache service and application to grid caching.* Concurrency and Computation : Practice and Experience, vol. 22, no. 9, pages 1118–1137, June 2010. 18

[Dragan 2006] Florin Dragan, Georges Gardarin, Benjamin Nguyen et Laurent Yeh. *On indexing multidimensional values in a P2P architecture.* In 22ème journées Bases de Données Avancées, 2006. 36

[EGEE] EGEE. *EGEE enabling grids for e-science. Web page.* http ://public.eu-egee.org/. 18

[Estrin 1999] Deborah Estrin, R. Govindan, J. Heidemann et S. Kumar. *Next century challenges : scalable coordination in sensor networks.* In Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking, pages 263–270, Seattle, USA, 1999. 40, 43

[Estrin 2002] Deborah Estrin, David Culler, Kris Pister et Gaurav Sukhatme. *Connecting the physical world with pervasive networks.* IEEE Pervasive Computing, vol. 1, no. 1, pages 59–69, 2002. 40

[Foster 2006] Ian T. Foster. *Globus toolkit version 4 : software for service-oriented systems.* Journal of Computer Science and Technology, vol. 21, no. 4, pages 513–520, 2006. 18

[Franklin 2005a] Michael Franklin, Alon Halevy et David Maier. *From databases to dataspaces : a new abstraction for information management.* SIGMOD Rec., vol. 34, no. 4, pages 27–33, 2005. 52

[Franklin 2005b] Michael J. Franklin, Shawn R. Jeffery, Sailesh Krishnamurthy, Frederick Reiss, Shariq Rizvi, Eugene Wu 0002, Owen Cooper, Anil Edakkunni et Wei Hong. *Design considerations for high fan-in systems : the HiFi approach.* In CIDR, pages 290–304, 2005. 50

[Furetière 1690] Antoine Furetière. Dictionnaire universel. Rotterdam : Augsgaden Den Haag, 1690. 9

[Galanis 2003] L. Galanis, Y. Wang, S. Jeffery et D. DeWitt. *Locating data sources in large distributed systems.* In Proc. of Int'l Conference on VLDB, pages 874–885, Berlin, Allemagne, September 2003. 32, 36, 37

[Gao 2005] Tia Gao, Dan Greenspan, Matt Welsh, Radford R. Juang et Alex Alm. *Vital signs monitoring and patient tracking over a wireless network.* In 27th Annual International Conference of the IEEE EMBS, pages 102–105, Shanghai, September 2005. 40

[Garcés-Erice 2004] L. Garcés-Erice, P. Felber, E. Biersack et G. Urvoy-Keller. *Data indexing in peer-to-peer DHT networks.* In In Proc. Int'l Conference on Distributed Computing Systems, pages 200–208, Columbus, Ohio, Etats-Unis, Juin 2004. 32, 36

[Gibbons 2003] Phillip B. Gibbons, Brad Karp, Yan Ke, Suman Nath et Srinivasan Seshan. *IrisNet : an architecture for a worldwide sensor web.* IEEE Pervasive Computing, vol. 02, no. 4, pages 22–33, 2003. 50

[gLite] gLite. *gLite lightweight middleware for grid computing. Web page.* http ://glite.web.cern.ch/glite/. 18

[Globus] Globus. *Globus.* http ://www.globus.org. 18

[Gnawali 2002] O. Gnawali. *A keyword-set search system for peer-to-peer networks.* Master thesis, Massachusetts Institute Of Technology, Massachusetts, Etats-Unis, Juin 2002. 32, 33

[Gnutella] Gnutella. *The Gnutella Protocol Specification v0.41.* http ://www9.limewire.com/developer/gnutella_protocol_0.4.pdf. 30

[Golab 2003] Lukasz Golab et M. Tamer Özsu. *Issues in data stream management.* SIGMOD Rec., vol. 32, no. 2, pages 5–14, 2003. 40, 41, 45

[Grefenstette 1994] Gregory Grefenstette. Explorations in automatic thesaurus discovery. Kluwer Academic Publishers, Norwell, MA, USA, 1994. 7

[Gribble 2001] S. Gribble, A. Halevy, Z. Ives, M. Rodrig et D. Suciu. *What can databases do for peer-to-peer.* In Proc. of WebDB, pages 31–36, Californie, Etats-Unis, Mai 2001. 30

[Gürgen 2005a] Levent Gürgen, Cyril Labbé, Vincent Olive et Claudia Roncancio. *A scalable architecture for heterogeneous sensor management.* In DEXA Workshops, pages 1108–1112, 2005. 39, 45

[Gürgen 2005b] Levent Gürgen, Cyril Labbé, Vincent Olive et Claudia Roncancio. *Une architecture hybride pour l'interrogation et l'administration des capteurs.* In 2ième conférence française ubiquité et mobilité (UbiMob'05), pages 37–44, 2005. 39

[Gürgen 2006a] Levent Gürgen, Cyril Labbé, Claudia Roncancio et Vincent Olive. *SStreaM : a model for representing sensor data and sensor queries.* In International Conference on Intelligent Systems And Computing : Theory And Applications (ISYC), July 2006. 39

[Gürgen 2006b] Levent Gürgen, Claudia Roncancio, Cyril Labbé et Vincent Olive. *Transactional Issues in Sensor Data Management.* In 3rd International Workshop On Data Management for Sensor Networks (DMSN'06) in conjunction with VLDB'06, pages 27–32, 2006. 40, 45

[Gürgen 2007] Levent Gürgen. *Gestion à grande échelle de données de capteurs hétérogènes.* PhD thesis, Institut National Polytechnique de Grenoble, Septembre 2007. 39

[Gürgen 2008a] Levent Gürgen, Claudia Roncancio, Cyril Labbé, André Bottaro et Vincent Olive. *SStreaMWare : a service oriented middleware for heterogeneous sensor data management.* In International Conference on Pervasive Services (ICPS), pages 121–130, July 2008. 39, 50

[Gürgen 2008b] Levent Gürgen, Claudia Roncancio, Cyril Labbé et Vincent Olive. *Update tolerant execution of continuous queries on sensor data.* In 5th IEEE International Conference on Networked Sensing Systems (INSS), pages 51 – 54, July 2008. 40, 45, 51

[Gürgen 2008c] Levent Gürgen, Claudia Roncancio, Cyril Labbé, Vincent Olive et Didier Donsez. *Scalable management of heterogeneous senor data in dynamic environments (Demonstration).* Fifth IEEE International Conference on Networked Sensing Systems (INSS), June 2008. 39

[Gürgen 2009a] Levent Gürgen et Shinichi Honiden. *Management of networked sensing devices.* In Proceedings of the 2nd. International Workshop on Sensor Network Technologies for Information Explosion Era, 2009. 40, 45

[Gürgen 2009b] Levent Gürgen, Johan Nyström-Persson, Amin Cherbal, Cyril Labbé, Claudia Roncancio et Shinichi Honiden. *Plug&manage heterogeneous sensing devices.* In DMSN, 2009. 51

[Gürgen 2010a] Levent Gürgen, Johan Nyström-Persson, Amin Cherbal, Cyril Labbé, Claudia Roncancio et Shinichi Honiden. *Service-oriented middleware for dynamic management of heterogeneous sensing devices.* In 7th ACM International Conference on Pervasive Services (ICPS 2010), 2010. 40

[Gürgen 2010b] Levent Gürgen, Claudia Roncancio, Cyril Labbé et Shinichi Honiden. *Data management solutions in networked sensing systems.* In Takahiro Hara, Vladimir Zadorozhny et Erik Buchmann, editeurs, Wireless Sensor Network Technologies for the Information Explosion Era, Studies in Computational Intelligence, pages 111–137. Springer Berlin / Heidelberg, 2010. 39, 49

[Han 2005] Chih-Chieh Han, Ram Kumar, Roy Shea, Eddie Kohler et Mani Srivastava. *A dynamic operating system for sensor nodes.* In MobiSys '05 : Proceedings of the 3rd international conference on Mobile systems, applications, and services, pages 163–176, New York, NY, USA, 2005. The ACM Press. 42

[Hubert 2002] Pierre Hubert, Cyril Labbé et Dominique Labbé. *Segmentation automatique des corpus. Voyages de l'autre côté de J.-M. Le Clézio.* In Annie et Sébillot Pascale, editeur, VIe Journées Internationales d'Analyse des Données Textuelles, volume 1, pages 359–369, 2002. 5

[Hubert 2004] Pierre Hubert, Cyril Labbé et Dominique Labbé. *Automatic segmentation of texts and corpora.* Journal of Quantitative Linguistics, vol. 11, no. 3, pages 193–213, december 2004. 5

[Huebsch 2003] R. Huebsch, J. Hellerstein, N. Lanham, B. Loo, S. Shenker et I. Stoica. *Querying the Internet with PIER.* In Proc. of Int'l Conference on VLDB, Berlin, Allemagne, Septembre 2003. 32, 33, 35

[Huebsch 2005] R. Huebsch, B. Chun, J. Hellerstein, B. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica et A.R. Ymerefendi. *The architecture of PIER : an internet-scale query processor.* In The Second Biennial Conference on Innovative Data Systems Research : CIDR, Californie, Etats-Unis, Janvier 2005. 32, 35

[Jagadish 2005] H.V. Jagadish, B.C. Ooi et Q.H. Vu. *Baton : a balanced tree structure for peer-to-peer networks.* In Proc. of Int'l Conference on VLDB, Trondheim, Norvège, Septembre 2005. 30

[Jain 2008] Namit Jain, Shailendra Mishra, Anand Srinivasan, Johannes Gehrke, Jennifer Widom, Hari Balakrishnan, Uğur Çetintemel, Mitch Cherniack, Richard Tibbetts et Stan Zdonik. *Towards a streaming SQL standard.* VLDB'08 : Proceedings of the 34th international conference on Very Large Data bases, vol. 1, no. 2, pages 1379–1390, 2008. 48

[Jamard 2007] Clément Jamard, Georges Gardarin et Laurent Yeh. *Indexing textual XML in P2P networks using distributed bloom filters.* In DASFAA, pages 1007–1012, 2007. 36

[Jurdak 2009] Raja Jurdak, A G Ruzzelli, Alessio Barbirato et S Boivineau. *Octopus : modular visualization and control for sensor networks.* Wiley Wireless Communications and Mobile Computing, 2009. 51

[Keller 1996] Arthur M. Keller et Julie Basu. *A predicate-based caching scheme for client-server DB architectures.* The VLDB Journal, vol. 5, no. 1, pages 35–47, 1996. 23

[Kossmann 2000] D. Kossmann. *The state of the art in distributed query processing.* ACM Computing Surveys, vol. 32, no. 4, pages 422–469, 2000. 34

[Krishnamachari 2002] Bhaskar Krishnamachari, Deborah Estrin et Stephen B. Wicker. *The impact of data aggregation in wireless sensor networks.* In ICDCSW'02 : Proceedings of the 22nd International Conference on Distributed Computing Systems, pages 575–578, Washington, DC, USA, 2002. IEEE. 44

[Labbé 1990] Dominique Labbé. *Normes de saisie et de dépouillement des textes politiques.* Rapport technique, Cahier du CERAT, 1990. 11

[Labbé 2001] Cyril Labbé et Dominique Labbé. *Inter-textual distance and authorship attribution. Corneille and Molière.* Journal of Quantitative Linguistics, vol. 8, no. 3, pages 213–231, 12 2001. version anglaise préliminaire à l'article paru sous ce titre dans le Journal of Quantitative Linguistics. 8-3, December 2001, p 213-231. 5

[Labbé 2003] Cyril Labbé et Dominique Labbé. *La distance intertextuelle.* Corpus, pages 95–118, 12 2003. 5

[Labbé 2005] Cyril Labbé et Dominique Labbé. *How to Measure the Meanings of Words ? Amour in Corneille's Work.* Language Resources and Evaluation, vol. 39, no. 4, pages 335–351, 2005. 5, 7, 13

[Labbé 2006a] Cyril Labbé et Dominique Labbé. *A tool for literary studies : intertextual distance and tree classification.* Literary and Linguistic Computing, vol. 21, no. 3, pages 311–326, 2006. 5

[Labbé 2006b] Cyril Labbé et Dominique Labbé. *La diachronie dans le discours politique. Le général de Gaulle.* In Nouvelles journées de l'ERLA. Aspects diachroniques du texte de spécialité, Brest France, 11 2006. 5

[Labbé 2008] Cyril Labbé et Dominique Labbé. *Peut-on se fier aux arbres ?* In Actes des 9e journées internationales d'analyse statistique des données textuelles, volume 2, pages 635–645, Lyon, 2008. 5

[Labbé 2010] Cyril Labbé et Dominique Labbé. *Ce que disent leurs phrases.* In Proceedings of the 10th International Conference Statistcal Analysis of Textual Data, volume 1, pages 297–307, 2010. 5

[Lee 1999] Dongwon Lee et Wesley W. Chu. *Semantic caching via query matching for web sources.* In Proceedings of the eighth international conference on Information and knowledge management, pages 77–85, 1999. 23

[Levis 2004] Philip Levis, Samuel Madden, David Gay, Joseph Polastre, Robert Szewczyk, Alec Woo, Eric A. Brewer et David E. Culler. *The emergence of networking abstractions and techniques in TinyOS.* In NSDI, pages 1–14, 2004. 42

[Liu 2003] Ting Liu et Margaret Martonosi. *Impala : a middleware system for managing autonomic, parallel sensor systems.* In PPoPP '03 : Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming, pages 107–118, New York, NY, USA, 2003. The ACM Press. 45

[Logothetis 2008] Dionysios Logothetis et Kenneth Yocum. *Wide-scale data stream management.* In ATC'08 : USENIX 2008 Annual Technical Conference on Annual Technical Conference, pages 405–418, Berkeley, CA, USA, 2008. USENIX Association. 49

[Lorincz 2004] Konrad Lorincz, David J. Malan, Thaddeus R. F. Fulford-Jones, Alan Nawoj, Antony Clavel, Victor Shnayder, Geoffrey Mainland, Matt Welsh et Steve Moulton. *Sensor networks for emergency response : challenges and opportunities.* IEEE Pervasive Computing, vol. 3, no. 4, pages 16–23, 2004. 40

[Madden 2002a] Samuel Madden et Michael J. Franklin. *Fjording the stream : an architecture for queries over streaming sensor data.* In Proceedings of the 18th International Conference on Data Engineering (ICDE'02), page 555, Washington, DC, USA, 2002. IEEE Computer Society. 50

[Madden 2002b] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein et Wei Hong. *TAG : A Tiny AGgregation service for ad-hoc sensor networks.* In OSDI, 2002. 43, 45

[Madden 2002c] Samuel Madden, Mehul Shah, Joseph M. Hellerstein et Vijayshankar Raman. *Continuously adaptive continuous queries over streams.* In Proceedings of the 2002 ACM SIGMOD international conference on Management of Data (SIGMOD'02), pages 49–60, New York, NY, USA, 2002. The ACM Press. 49

[Madden 2005] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein et Wei Hong. *TinyDB : an acquisitional query processing system for sensor networks.* ACM Trans. Database Syst., vol. 30, no. 1, pages 122–173, 2005. 44

[Mainwaring 2002] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk et John Anderson. *Wireless sensor networks for habitat monitoring.* In Proceedings of the 1st ACM international workshop on Wireless sensor Networks and Applications (WSNA'02), pages 88–97, NY, USA, 2002. The ACM Press. 40

[Michel 2007] S. Michel. *Top-k aggregation queries in large-scale distributed systems.* Phd thesis, Universität des Saarlandes, Saarbrücken, Allemagne, Mai 2007. 35

[Mobius] Mobius. *The Mobius project. Web page.* http ://projectmobius.osu.edu/. 18

[Monière 2008] Denis Monière, Cyril Labbé et Dominique Labbé. *Les styles discursifs des premiers ministres québécois de Jean Lesage à Jean Charest.* Canadian Journal of Political Science/ Revue canadienne de science politique, vol. 41, no. 1, pages 43–69, 03 2008. 5

[Perrig 2004] Adrian Perrig, John Stankovic et David Wagner. *Security in wireless sensor networks.* Communications of the ACM, vol. 47, no. 6, pages 53–57, 2004. 45

[Petit 2010] Loïc Petit, Cyril Labbé et Claudia Roncancio. *An algebric window model for data stream management.* In MobiDE, pages 17–24, 2010. 40, 47

[Pincemin 2004] Bénédicte Pincemin. *Lexicométrie sur corpus étiquetés.* In Le poids des mots, actes des 7ièmes journées internationales d'analyse statistique de données textuelles, 2004. 12

[Prada 2008] Carlos Prada, Maria-Del-Pilar Villamil et Claudia Roncancio. *Join queries in P2P DHT Systems.* In Proc. of 6th Int'l Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2008), Auckland, New Zealand, Août 2008. 34

[Rahayu 2010a] Dwi Rahayu, Shonali Krishnaswamy, Cyril Labbe et Oshadi Alahakoon. *RnR : a system for extracting rationale from online reviews and ratings.* In Social Interactions Analysis and Services Providers (SIAPS), workshop in conjunction with ICDM 2010, 2010. 5

[Rahayu 2010b] Dwi Rahayu, Shonali Krishnaswamy, Cyril Labbe et Oshadi Alahakoon. *RnR : a system for extracting rationale from online reviews and ratings.* In Demonstration eighth International Conference on Service on Service Oriented Computing (ICSOC), 2010. 5

[Rahayu 2010c] Dwi Rahayu, Shonali Krishnaswamy, Cyril Labbe et Oshadi Alahakoon. *Web services for analysing and summarising online opinions and reviews.* In ServiceWave 2010, 2010. 5

[Ramabhadran 2004] S. Ramabhadran, S. Ratnasamy, J.M. Hellerstein et S. Shenker. *Prefix hash trees an indexing data structure over Distributed Hash Tables*, 2004. http ://berkeley.intel-research.net/sylvia/pht.pdf. 35

[Ren 2003] Qun Ren, Margaret H. Dunham et Vijay Kumar. *Semantic caching and query processing.* IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 1, pages 192–210, 2003. 23

[Rice University 2002] Rice University. *FreePastry*, 2002. http ://freepastry.rice.edu/FreePastry/. 37

[Römer 2004] K. Römer et F. Mattern. *The design space of wireless sensor networks.* IEEE Wireless Communications, vol. 11, no. 6, pages 54–61, Dezember 2004. 40

[Roncancio 2009] Claudia Roncancio, Maria-Del-Pilar Villamil, Cyril Labbé et Patricia Serrano-Alvarado. *Data sharing in DHT based P2P systems.* Transactions on Large-Scale Data and Knowledge Centered Systems, pages 327–352, 2009. 29

[Roussopoulos 1991] Nicholas Roussopoulos. *An incremental access method for ViewCache : concept, algorithms, and cost analysis.* ACM Transactions on DB Systems, vol. 16, no. 3, pages 535–563, 1991. 23

[Rowstron 2001] A. Rowstron et P. Druschel. *Pastry : scalable, decentralized object location, and routing for large-scale peer-to-peer systems.* In IFIP/ACM Int'l Conference on Distributed Systems Platforms (Middleware), Heidelberg, Allemagne, Novembre 2001. 30

[Sagen 2010] Alexandre Sagen, Cyril Labbé, Mohamed Gaber, Shonali Krishnaswamy, Agustinus Borgy Waluyo et Seng Loke. *Saving energy on sensors through data clustering.* In UDM 2010 workshop in conjunction with ECAI 2010, 2010. 40

[Seshadri 1996] Praveen Seshadri, Miron Livny et Raghu Ramakrishnan. *The design and implementation of a sequence database system.* In Proceedings of VLDB'96, pages 99–110, San Francisco, USA, 1996. 47

[Seshadri 2006] Sangeetha Seshadri, Vibhore Kumar et Brian F. Cooper. *Optimizing multiple queries in distributed data stream systems.* In Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06), page 25, Washington, DC, USA, 2006. IEEE Computer Society. 49

[Sharaf 2004] Mohamed A. Sharaf, Jonathan Beaver, Alexandros Labrinidis et Panos K. Chrysanthis. *Balancing energy efficiency and quality of aggregate data in sensor networks.* The VLDB Journal, vol. 13, no. 4, pages 384–403, December 2004. 44

[Shing 2004] S. Shing, G. Yang, D. Wang, J. Yu, S. Qu et M. Chen. *Making peer-to-peer keyword searching feasible using multi-level partitioning.* In Proc. Int'l WS on Peer-to-Peer Systems (IPTPS), San Diego, CA, Etats-Unis, Février 2004. 32, 33

[Shneidman 2004] Jeffrey Shneidman, Peter Pietzuch, Jonathan Ledlie, Mema Roussopoulos, Margo Seltzer et Matt Welsh. *Hourglass : an infrastructure for connecting sensor networks and applications.* Rapport technique TR-21-04, Harvard University, 2004. 50

[SRB] SRB. *SRB the SDSC storage ressource broker. Web page.* http ://www.sdsc.edu/srb. 18

[Steinmetz 2005] R. Steinmetz et K. Wehrle. Peer-to-peer systems and applications. Springer-Verlag, 2005. 30

[Stoica 2001] I. Stoica, R. Morris, D. Karger, F. Kaashoek et H. Balakrishnan. *Chord : a scalable peer-to-peer lookup service for internet applications.* In ACM SIGCOMM Conference, pages 149–160, San Diego, CA, Etats-Unis, Août 2001. 30, 33

[Stream 2008] Stream. *Stream query repository, www-db.stanford.edu/stream/sqr*, 2008. 47

[StreamBase 2008] StreamBase. *StreamBase systems, Inc. http ://www.streambase.com/*, 2008. 46

[Swiss-Prot] Swiss-Prot. *La base de connaissances Swiss-Prot.* http ://www.ebi.ac.uk/swissprot/. 25

[Tatbul 2003] Nesime Tatbul, Ugur Çetintemel, Stanley B. Zdonik, Mitch Cherniack et Michael Stonebraker. *Load shedding in a data stream manager.* In Proceedings of the 29th VLDB Conference, pages 309–320, 2003. 48

[Triantafillou 2003] P. Triantafillou et T. Pitoura. *Toward a unifying framework for complex query processing over structured peer-to-peer data networks.* In VLDB WS on Databases, Information Systems, and Peer-to-Peer Computing, Berlin, Allemagne, Septembre 2003. 32, 33, 35, 36

[Ulmer 2003] C. Ulmer, L. Alkalai et S. Yalamanchili. *Wireless distributed sensor networks for in-situ exploration of mars.* Rapport technique 2003-67, NASA, 2003. 40

[Vanlentin 2006] Olivier Vanlentin, Fabrice Jouanot, Laurent d'Orazio, Yves Denneulin, Claudia Roncancio, Cyril Labbé, Christophe Blanchet, Pierre Sens et Claude Bonnard. *GEDEON, un Intergiciel pour grille de données.* In Conférence Française en Système d'Exploitation, 2006. 18

[Villamil 2004] Maria-Del-Pilar Villamil, Claudia Roncancio et Cyril Labbé. *PinS : peer to peer interrogation and indexing system.* In Proc. Int'l Database Engineering and Applications Symposium, pages 236 – 245, Coimbra, Portugal, Juin 2004. 29

[Villamil 2005a] Maria-Del-Pilar Villamil, Claudia Roncancio et Cyril Labbé. *Querying in massively distributed storage systems.* In Journées Bases de Données Avancées, Saint Malo, France, Octobre 2005. 29, 33, 34

[Villamil 2005b] Maria-Del-Pilar Villamil, Claudia Roncancio, Cyril Labbé et Carlos Afonso-Dos-Santos. *Location queries in DHT P2P systems.* In Demonstration Journées Bases de Données Avancées, Saint Malo, France, Octobre 2005. 29

[Villamil 2006a] Maria-Del-Pilar Villamil. *Service de localisation de données pour les systèmes P2P.* Phd thesis, Institut National Polytechnique de Grenoble, Grenoble, France, Juin 2006. 29, 34

[Villamil 2006b] Maria-Del-Pilar Villamil, Claudia Roncancio et Cyril Labbé. *Range queries in massively distributed data.* In Proc. Int'l WS on Grid and Peer-to-Peer Computing Impacts on Large Scale Heterogeneous Distributed Database Systems, pages 1529–4188, Cracovie, Pologne, Septembre 2006. 29

[Wan 2002] Chieh-Yih Wan, Andrew T. Campbell et Lakshman Krishnamurthy. *PSFQ : a reliable transport protocol for wireless sensor networks.* In Proceedings of the 1st ACM international workshop on Wireless Sensor Networks and Applications (WSNA'02), pages 1–11, New York, NY, USA, 2002. ACM. 45

[Weeds 2005] Julie Weeds et David Weir. *Co-occurrence retrieval : a flexible framework for lexical distributional similarity.* Comput. Linguist., vol. 31, no. 4, pages 439–475, 2005. 7

[Widom 2003] J. Widom et R. Motwani. *Query processing, resource management, and approximation in a data stream management system.* In Proceedings of The First Biennial Conference on Innovative Data Systems Research (CIDR), pages 245–256, 2003. 48

[Xiong 2007] Xiaopeng Xiong, Hicham G. Elmongui, Xiaoyong Chai et Walid G. Aref. *Place : a distributed spatio-temporal data stream management system for moving objects.* 8th International Conference on Mobile Data Management,, pages 44–51, 05 2007. 49

[Yao 2002] Yong Yao et Johannes Gehrke. *The cougar approach to in-network query processing in sensor networks.* SIGMOD Rec., vol. 31, no. 3, pages 9–18, 2002. 44

Le sens au cœur des systèmes d'information

Résumé : La mise en réseau des dispositifs de gestion de l'information, qu'ils soient de petite taille (capteur - dispositif) ou de grande taille (cluster -super calculateur) accompagnent et accélèrent l'émergence d'une informatique ubiquitaire. Ce mouvement de fond entraîne une explosion, tant de la quantité que de la diversité de l'information disponible. Le sens même de ces informations est souvent ignoré par les traitements opérés dans les couches basses des systèmes qui gèrent ces informations.

Dans un contexte où les sources d'information deviennent surabondantes, l'exécution de l'opération la plus élémentaire, portant sur la plus élémentaire des informations, passe par la maîtrise du sens associé aux données manipulées. Une des évolutions majeures à venir est donc, l'intégration, au cœur des systèmes, du sens associé à l'information et aux processus de traitement.

Les domaines plus particulièrement développés sont la recherche de sens dans les textes et la gestion de données dans les systèmes ubiquitaires à grande échelle. Ces recherches ont été conduites au sein de la communauté IMAG et se poursuivent aujourd'hui dans l'équipe SIGMA - laboratoire LIG et de l'Université Joseph Fourier.

Mots clés : Lexicométrie, gestion de données, grilles, systèmes pair à pair, capteurs.

Meaning at the heart of information Systems

Abstract : The setting in network of information management devices, which can be either of small size (sensor - device) or of large size (cluster) is going along and accelerate the emergence of data processing in a ubiquity way. This essential trend leads to an explosion, of the quantity as well as the diversity of available information. The meaning of information is often ignored by processing operated in lower layers of information managing systems.

In a context, where information sources become superabundant, the execution of the most elementary operation, bearing on most elementary informations, needs to be adapted to the meaning associated with the data

In this document, developed topics, are the seek for meaning in texts and data management in large scale ubiquitous systems. This research has been done at the LIG laboratory in the SIGMA team, Joseph Fourier University Grenoble.

Keywords : Text mining, Data management, gird, Peer to peer systems, sensors.