UNIVERSITÉ DE SHERBROOKE
Faculté des sciences appliquées
Département de génie électrique et de génie informatique

CONCEPTION ET APPLICATION D'UNE MÉTHODE DE CLASSIFICATION UTILISANT LA PROBABILITÉ ET LE FLOU DÉDIÉE À LA RECHERCHE DOCUMENTAIRE

LE SYSTÈME CLASSFLOU

Mémoire de maîtrise es sciences appliquées
Spécialité: génie électrique

Neïla FAKHFAKH

Sherbrooke (Québec), Canada Février 1998

À mon père Ameur, À ma mère Radhia,

À mon frère Naoufel,

À tous mes ami(e)s au Canada et en Tunisie,

Je dédie ce mémoire...

RÉSUMÉ

Ce mémoire présente un système de classification automatique floue basé sur des relations probabilistes entre documents, dédié à la recherche documentaire. Le projet de recherche présenté dans ce mémoire a pour objectif d'assurer le regroupement des documents d'une base documentaire dans un ensemble de classes disjointes, selon une relation de ressemblance déterminée en fonction des termes d'indexation.

Notre système de classification CLASSFLOU constitue une composante importante d'un projet qui vise à réunir une multitude d'outils qui selon nous concourent aux buts que nous nous sommes fixés : autoriser une consultation multilingue des milliers de sites d'informations disponibles sur le Web, tout en essayant de réduire au maximum la quantité de documents non pertinents.

L'essentiel de notre travail a porté sur la conception et la réalisation d'un tel système. Comparé à un système de classification binaire, notre système a réussi à améliorer la qualité de la classification. De plus, nous l'avons étendu afin qu'il puisse traiter les bases documentaires dynamiques sans pour autant être contraint d'effectuer une reclassification complète.

REMERCIEMENTS

Je profite de ces lignes pour souligner l'apport des personnes qui m'ont appuyé dans l'accomplissement de ce travail durant toute la période de mes recherches.

Je tiens à remercier mon directeur de recherche RUBEN GONZALEZ-RUBIO pour sa grande disponibilité, ses encouragements et les précieux conseils prodigués tout au long de ce travail.

J'exprime ma profonde reconnaissance à mes parents pour l'encouragement et le soutien moral et financier accordé tout au long de mes études.

Enfin, une attention toute spéciale à tous mes ami(e)s au Canada et en Tunisie qui m'ont aidé, supporté et encouragé tout au long de mes travaux.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

LEXIQUE

CC : C'est l'abréviation de coefficient de couverture, il exprime le degré de liaison d'un document avec un autre.

FCM : C'est l'abréviation de "Fuzzy C-Means", un algorithme de classification floue.

Pro-floue : C'est le nom que nous avons donné à notre approche de classification, puisqu'elle combine deux techniques : la probabilité et le flou.

Chapitre 1

INTRODUCTION

Le réseau Internet constitue une source extraordinaire de ressources de toutes sortes. Chaque jour, de nouveaux documents et outils deviennent accessibles par le biais du réseau, alors que d'autres disparaissent ou sont déplacés. Les changements se font à une telle vitesse qu'il est impossible à un seul individu d'être constamment à jour, même s'il y consacre tout son temps.

Des ressources phénoménales, un nombre astronomique de sites, des changements souvent fréquents... Comment, dans un tel contexte, trouver rapidement des informations sur Internet? Mais surtout, comment trouver l'information pertinente au milieu d'une telle quantité de ressources, dont la vaste majorité ne présente pour nous aucun intérêt immédiat?

L'une des techniques permettant l'exploration de l'information sur Internet est la classification documentaire. Dans ce même ordre d'idées, ce mémoire présente un système de

classification automatique floue basé sur des relations probabilistes, dédié à la recherche documentaire. Ce système assure, entre autres, le regroupement des documents sous forme de classes disjointes selon une relation de ressemblance entre documents en fonction de leurs termes d'indexation. L'application du concept de la classification floue, pour cataloguer systématiquement des documents, constitue un nouveau champ d'investigation et c'est d'ailleurs l'objet de notre recherche.

1.1 Contexte général du travail

Notre système de classification CLASSFLOU fait partie intégrante d'un grand projet pilote mené en collaboration avec le laboratoire d'informatique de Marseille (LIM). Ce projet réunit une multitude d'outils qui selon nous concourent au but que notre équipe de recherche s'est fixé: autoriser une consultation multilingue des milliers de sites d'informations offerts par le Web, tout en essayant de réduire au maximum la quantité de documents non pertinents.

Le projet comprend alors cinq composantes principales [figure 1.1]. Nous allons décrire dans ce qui suit chacune d'entre elles:

- **composante 1**: L'outil INTEX. C'est un processeur de corpus multilingue présenté dans [Sil93] permettant d'extraire avec une grande fiabilité les mots pleins ou les locutions non nécessairement connexes, de les filtrer selon leurs catégories syntaxiques, de localiser et regrouper toutes les flexions d'une forme lemmatisée ou encore, à partir d'une grammaire locale, de déterminer toutes les constructions correspondantes apparaissant dans le texte.

- **composante 2** : L'outil ILLICO. C'est un outil qui a été développé par l'équipe langue naturelle du LIM [PS97]. Ce système trouve son adéquation à notre problème dans l'analyse des requêtes données en langue naturelle. La requête étant donc fournie en langue naturelle, après analyse, ce système propose à l'utilisateur une formule logique compatible avec ce qui doit être soumis au serveur supportant ce type de requête.

- **composante 3** : Faute de pouvoir modifier le fonctionnement actuel des engins de recherche du Web, nous allons procéder à une simple utilisation de l'un des moteurs existant dans le Web.

- **composante 4** : Au cours d'une session de recherche, l'utilisateur peut se trouver confronté à une grande masse d'adresses redondantes ou encore qui ne correspondent pas à son choix au niveau de sa langue d'intérêt ou qui ne pointent pas vers des adresses actives. Une option de filtrage lui serait d'une grande aide. Cette option a été conçue de manière à pouvoir accroître le taux de pertinence.

- **composante 5** : L'outil CLASSFLOU. Basé sur la technique d'agrégation en classes de ressemblance, il vise à déterminer une relation de ressemblance floue entre documents et par conséquent à organiser les réponses obtenues.

Dans le cadre de ce mémoire de maîtrise, nous avons été amenés à nous intéresser aux problèmes de classification des documents, comme phase finale du processus de recherche visant à organiser les informations obtenues à la suite de lancement d'une requête.

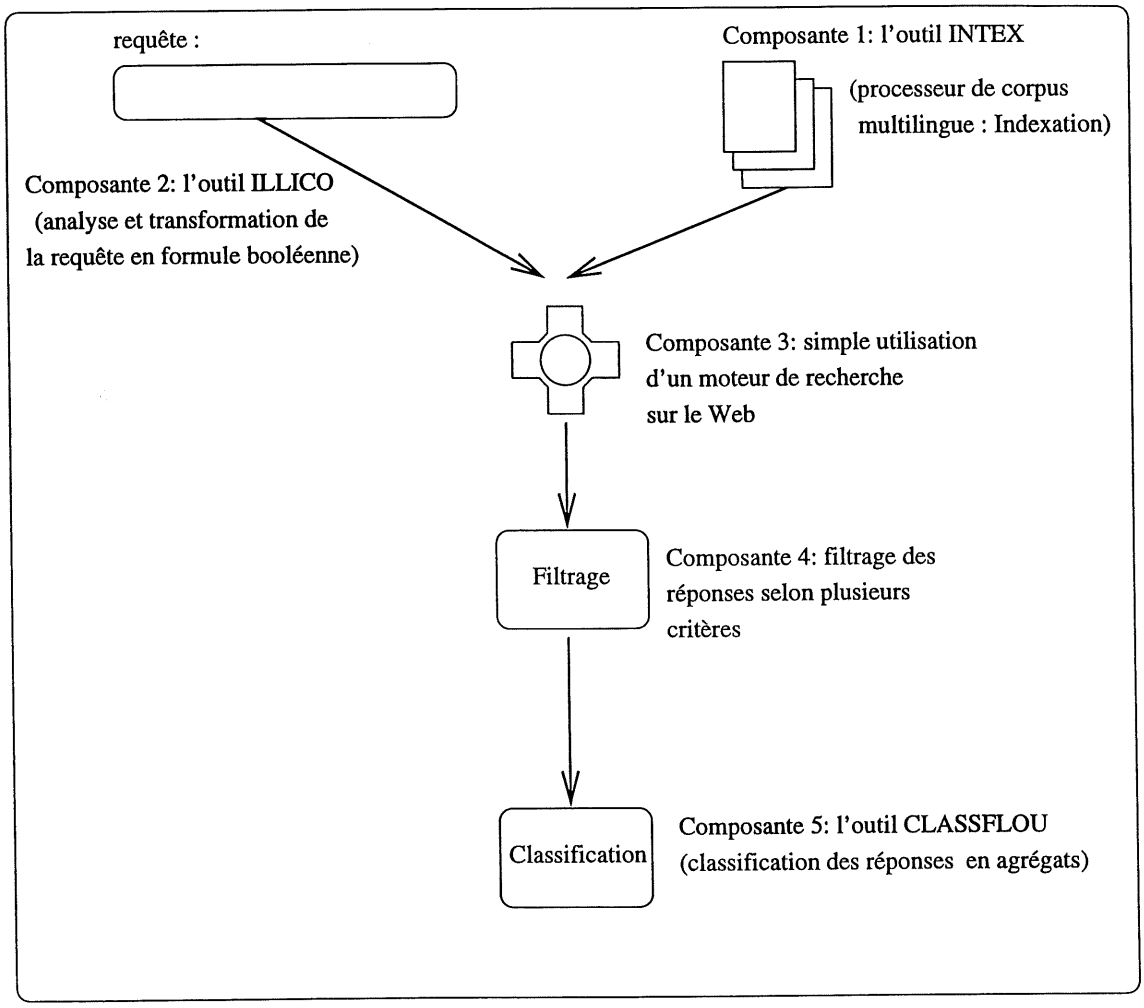


Figure 1.1 - *Maquette générale du projet.*

1.2 Présentation de la problématique

La manière dont les informations sont organisées et présentées à l'utilisateur à la suite d'une recherche, peut être de deux types :

- soit qu'elles sont présentées à l'utilisateur à l'état brut, c'est-à-dire sans aucune structuration, ni organisation, elles sont seulement ordonnées par leurs degrés de pertinence par rapport à la requête.
- soit qu'elles sont organisées par un système de classification calqué généralement sur les systèmes de classification hiérarchiques traditionnels comme Dewey [CCS95] et "library of congress" [CS68]. Ces derniers suscitent aussi certaines critiques : ils présentent, en effet, plusieurs niveaux de ramifications hiérarchiques entre classes rendant ainsi la recherche d'un document complexe à atteindre.

De ces considérations est née l'idée de créer un outil, de type "classificateur répartitionniste" basé sur la technique d'agrégation en classes de ressemblance, construit à partir de termes d'indexation et de données statistiques devant aider à la structuration et à l'organisation d'une base de données documentaire. Nous en avons fait l'objet de notre mémoire de maîtrise dont nous présentons dans ce chapitre les grandes lignes.

1.3 Méthodologie de travail

On se propose tout au long de ce mémoire de décrire notre approche de classification, dédiée à la recherche documentaire, dont le principe se base sur la combinaison de la probabilité et la classification floue. On donnera le nom de classification "Pro-Floue" à notre méthode.

Une comparaison est ensuite établie entre la méthode Pro-Floue et une autre méthode de classification binaire décrite dans l'article de Can et Ozkarahan [CO83]. Les deux méthodes ont été programmées afin de pouvoir juger la qualité de la classification de chacune d'elles.

Afin de concrétiser ces concepts, on va illustrer notre démarche par les étapes suivantes :

- représenter les connaissances relatives à la base documentaire [figure 1.2](1);
- déterminer par une approche probabiliste, les degrés de corrélation entre documents [figure 1.2](2).

Ces étapes vont servir de base commune pour l'implantation des deux algorithmes de classification, à savoir l'algorithme Pro-Floue et l'algorithme de Can et Ozkarahan.

Une fois que ces étapes ont été effectuées, il s'agit, d'une part, pour le cas de la méthode Pro-Floue de [figure 1.2]:

- construire à partir des degrés de corrélation des documents, la matrice des entrées E_0 qui sera soumise à l'algorithme de classification floue FCM, connu sous le nom de "Fuzzy C-Means", [Bez81];

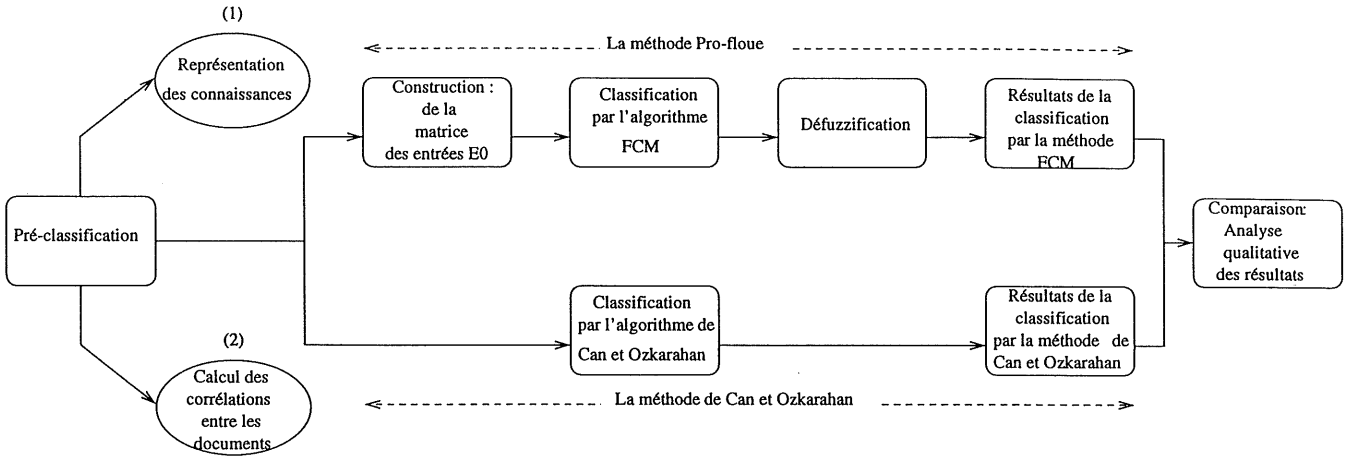


Figure 1.2 - *Schéma représentant les différentes étapes du travail*

- appliquer l'algorithme FCM sur cette matrice;
- déffuzzifer les résultats obtenus.

D'autre part, afin de pouvoir comparer les deux méthodes, il s'agit d'implémenter l'algorithme de classification binaire de Can et Ozkarahan [CO90], basé aussi sur le principe de partition. Cette méthode utilise, contrairement à l'algorithme FCM qui est itératif, un algorithme à une seule passe.

Mise à part la confrontation de notre approche Pro-Floue avec celle de Can et Ozkarahan, nous avons conçu et implanté une méthode de maintenance pour la classification des bases documentaires dynamiques. Le principe de cette méthode est détaillé dans le chapitre 5.

1.4 Réalisation

L'essentiel de notre travail a porté sur la conception et la réalisation d'un système de classification floue appelé CLASSFLOU. Cet outil a été conçu pour satisfaire deux objectifs :

- organiser une base documentaire en la divisant en classes partageant des sujets communs;
- tenir compte des mises à jour de documents sans pour autant être contraint à reclassifier la base entière.

Contrairement aux systèmes de classification peu nombreux qui existent sur Internet, le système que nous proposons évite tous les problèmes des ramifications hiérarchiques en suggérant une classification agglomérative organisée. Cette organisation au niveau de chaque classe prend sa source depuis les valeurs des degrés d'appartenance des ensembles flous déterminés par l'algorithme FCM.

En comparant notre approche Pro-Floue avec celle de Can et Ozkarahan, nous en concluons que la nôtre est simple et fiable. De plus, les résultats expérimentaux montrent qu'elle permet de bien structurer des bases de données documentaires en attribuant, dans la majorité des cas, à chaque classe les documents qui en font partie. Ces résultats sont justifiés par les points suivants :

- La méthode Pro-floue utilisée, basée d'abord sur le calcul des degrés de corrélation entre documents, permet de déterminer en s'appuyant sur une approche probabiliste, le nombre de classes nécessaires pour assurer la construction des classes.

- L'autre composante de la méthode Pro-floue, c'est l'algorithme FCM qui réalise l'étape de mise en correspondance de manière itérative. Puisque l'appariement est remis en cause à chaque étape, il n'est pas gênant que la mise en correspondance ne soit qu'approximative dans les premières étapes de l'algorithme. Un critère explicite du plus proche voisin convient bien pour faire ces appariements et déterminer la classe exacte à laquelle appartient le document, ce qui nous permet d'obtenir une meilleure qualité de classification.

- L'avantage de l'utilisation de l'algorithme FCM par rapport aux autres techniques de classification "hard" ou binaires est l'information inhérente que portent les degrés d'appartenance sur le jugement d'attribuer une donnée à une classe.

- L'algorithme FCM fournit, outre la sortie : matrice des degrés d'appartenance, la matrice des centres des classes. Ces derniers vont servir de documents pivots pour la classification de la base en cas d'ajout de termes et/ou de documents lorsque la base est dynamique.

1.5 Description du mémoire

Comme le titre de ce mémoire l'indique, il s'agit dans ce travail de vérifier si la stratégie de la classification Pro-Floue peut être utilisée pour structurer une base de données documentaire

indexée. Pour ce faire, nous décrirons les grandes lignes de notre mémoire :

- Dans le deuxième chapitre, nous présenterons un survol des recherches en informatique documentaire où nous aborderons essentiellement deux aspects : la représentation des documents d'une part, et l'accès à l'information d'autre part. Pour le premier thème, nous traiterons les techniques de représentation du contenu et les avancées réalisées ces dernières années pour les approches statistique et linguistico-conceptuelle.
  S'agissant de l'accès à l'information, on montre que les dernières années ont vu la problématique de l'appariement d'une requête à un ensemble de documents, dont nous décrirons les progrès techniques, se positionner progressivement à l'intérieur d'un cadre plus vaste, celui de la satisfaction du besoin d'information de l'utilisateur. Ainsi, nous avons vu apparaître plusieurs systèmes d'aide à l'activité de recherche, dont notamment les systèmes de navigation dans les hypertextes, les systèmes de classification automatique, les thesaurus, etc.

- Après cet aperçu sur l'état de l'art de la recherche en informatique documentaire, nous aborderons le troisième chapitre où il sera question d'entrer dans le vif du sujet en présentant les différentes méthodes existantes en classification automatique. Ce chapitre constituera un point de départ pour expliciter le principe de notre méthode.

- Le quatrième chapitre décrit en détail, à l'aide d'exemples, les différentes phases nécessaires pour l'étape de pré-classification. Le choix du modèle de représentation des connaissances, le principe du calcul des degrés de liaison entre documents ainsi que la détermination d'un estimatif du nombre de classes, constituent les principaux éléments développés dans ce chapitre.

- Dans le cinquième chapitre, il s'agit de décrire en détail l'algorithme de classification floue FCM, de l'appliquer sur un exemple et d'interpréter ses sorties. Dans ce même chapitre, nous introduisons un algorithme que nous avons créé pour traiter le cas de base documentaire dynamique. Finalement, afin d'évaluer notre méthode, nous présenterons l'algorithme de classification binaire de Can et Ozkarahan présenté dans [CO83], que nous nous proposons d'implanter et de comparer ses résultats avec ceux de nos algorithmes.
- Le sixième chapitre fournit, enfin, une évaluation du système CLASSFLOU à travers :
  - la confrontation de ses résultats avec ceux de l'algorithme de Can et Ozkarahan [CO83];
  - les résultats des tests de sa performance (juger la qualité de la classification).
- Finalement, nous conclurons ce mémoire en rappelant les originalités du système CLASSFLOU et en proposant des extensions futures de ce travail.

Chapitre 2

REPRÉSENTATION ET ACCÈS À L'INFORMATION

Afin de pouvoir traiter automatiquement un document, il est nécessaire que ce dernier subisse des transformations des informations depuis leur forme réelle vers la forme reconnue par le système. En effet, un système de recherche d'information ne travaille pas dans le monde réel, mais seulement sur une représentation des informations. Cette représentation peut varier selon deux axes. Elle recouvre en partie une dichotomie qui oppose sens et valeur. C'est ce que nous allons essayer de présenter dans le premier thème de ce chapitre.

Quant au second thème : la recherche d'information, nous nous intéresserons dans cette partie à l'évolution et aux tendances manifestées dans l'appariement entre requêtes et représentants des textes d'une base de données. Nous essaierons de montrer par la suite, de quelles évolutions, les outils de l'appariement c'est-à-dire les interfaces, ont fait l'objet ces

dernières années. En effet, on parle plutôt, de nos jours, de systèmes multi-experts ou hybrides qui rassemblent une multitude d'outils d'aide à l'activité de recherche dont nous exposerons quelques uns.

2.1 La représentation des documents

La représentation des documents est l'opération qui consiste à décrire et à caractériser un document à l'aide de représentation des concepts contenus dans ce texte, c'est à dire à transcrire en langage documentaire les concepts après les avoir extraits du document par une analyse. La transcription en langage documentaire se fait grâce à des outils d'indexation [Sal72].

2.1.1 Représenter un texte

Les mots :

Les techniques d'indexation automatique classiques voient la représentation des documents comme étant constituée de la totalité des formes que ceux-ci contiennent. De cet ensemble sont retranchés les mots grammaticaux trop fréquents et sans poids sémantique. Par forme il faut entendre séquence de caractères délimitée par des blancs ou des signes de ponctuation.

Les tenants de cette approche se sont bien vite rendu compte que des formes voisines,

les formes conjuguées d'un verbe par exemple, ont entre elles une intersection formelle et sémantique qui est celle du mot, d'où la mise au point d'algorithmes d'élimination des séquences terminales. Toutefois, la notion de séquence terminale n'est qu'une approximation de la notion de suffixe. Pour qu'une séquenence terminale soit un suffixe, il faut reconnaître un mot. Pour cela, il faut disposer d'un dictionnaire qui est le répertoire des mots, de leur signification et des flexions dont chacun d'eux peut être affecté. Ce sont là les premiers pas d'un traitement linguistique des textes. Aujourd'hui on peut considérer comme acquis et maitrisé ce type de traitement même s'il n'est pas toujours mis en oeuvre dans les systèmes commerciaux [Sal89].

Les termes :

Ce qui est toujours objet de recherche, c'est l'extraction automatique de séquences de mots. On sait que dans les domaines techniques surtout, les unités sémantiquement chargées sont les termes, c'est-à-dire des unités syntagmatiques dont la taille est supérieure au mot.

Deux types de techniques sont mises en oeuvre pour effecturer ce genre de traitement : des techniques statistiques d'une part et des techniques linguistiques d'autre part.

Les techniques statistiques repèrent dans les textes les "mots" fréquemment co-occurents pour élaborer des "cartes de termes associés". Le diagnostic porté sur l'utilisation de ces méthodes par [Sal86] est modérément positif : d'une part, les mots ainsi associés ne présenteraient pas souvent une unité de sens, d'autre part, l'augmentation de la précision[1] des

1. Le taux de précision est défini comme le rapport du nombre de documents pertinents retrouvés sur le nombre de documents retrouvés.

documents rappelés lors d'une recherche ne serait pas concluante.

La mise en oeuvre des "termes associés" peut conduire à une amélioration du taux de rappel[2] en accroissant les possibilités d'appariement entre les termes des requêtes et les termes affectés aux documents. Malheureusement l'expérimentation montre que seules 20% des associations entre paires de mots élaborées automatiquement sont sémantiquement légitimes.

Les techniques linguistiques quant à elles, se fondent sur un traitement morphologique et syntaxique pour extraire des unités syntagmatiques syntaxiquement et sémantiquement valides [Rou], [Ant88]. Les problèmes à résoudre sont : la profondeur de l'analyse syntaxique à mettre en oeuvre, la recherche des sous-unités syntagmatiques sémantiquement pertinentes pour représenter le document - les descripteurs - et l'expansion du vocabulaire d'indexation par paraphrasage.

Le sens :

La représentation du sens des documents textuels est le problème clé de l'indexation. On assiste à plusieurs types de tentatives dans l'univers documentaire. Les tentatives strictement linguistiques s'opposent à celles qui font appel, pour représenter le sens, à des objets de type conceptuel dont la validité n'est pas d'ordre linguistique; ce sont les modes de représentation des connaissances de l'intelligence artificielle. Autrement dit on peut opposer l'approche linguistique de la représentation du sens à l'approche intelligence artificielle.

2. Le taux de rappel est défini comme le rapport du nombre de documents pertinents retrouvés sur le nombre de documents pertinents de la collection.

L'approche linguistique consiste à représenter la sémantique des énoncés à l'aide de la logique. On procède alors aux analyses : lexicale, morphologique et syntaxique. La représentation syntaxique permet de dériver une représentation sémantique sous forme logique. Les efforts entrepris ici sont ceux de l'informatique linguistique. (Voir à ce sujet [CK86]). Il règne beaucoup d'incertitudes sur les modèles logiques à adopter. La sémantique des langues naturelles est un champ actif de la recherche en linguistique. Il ne semble pas raisonnable à l'heure actuelle d'escompter des résultats lorsque les textes sont longs et nombreux.

De l'approche linguistique, nous passons insensiblement à l'approche intelligence artificielle. La représentation et la modélisation des connaissances est l'objet de cette discipline. À coté de la logique comme mode de représentation des connaissances, d'autres formalismes ont été élaborés : frames, réseaux sémantiques, scripts [SKJ81]. Les bases de connaissances obtenues sont constituées de concepts entre lesquels divers types de relation sans statut linguistique sont établies. Si la traduction des phrases d'un texte en forme logique est chose difficile, l'extraction automatique de la représentation d'une phrase dans les termes d'une base de connaissances l'est tout autant. Elle n'est certainement pas envisageable actuellement comme technique opérationnelle de l'informatique documentaire à cause de l'ordre de grandeur des textes à traiter. Jusqu'ici, les textes que l'intelligence artificielle a réussi à traiter sont courts et traitent de domaines restreints.

2.1.2 Représenter une base de textes

Implicitement, la représentation du contenu sémantique d'une base de données textuelles devrait être la somme des représentations individuelles de chacun des textes qui la constitue.

Or ceci est loin d'être évident. Le développement historique de l'informatique documentaire nous invite à considérer les choses autrement.

En effet, que ce soit en indexation manuelle ou en indexation automatique, les outils de la représentation du contenu considèrent la représentation d'un document particulier comme une fonction de l'ensemble des documents contenus dans la base. Il en est ainsi du thesaurus. Salton [Sal86] considère que la fonction du thesaurus est de normaliser le vocabulaire des documents. À notre avis, la normalisation n'est qu'un effet de l'objectif plus général qui est de représenter les documents non pas de façon atomique un par un, mais en les situant les uns par rapport aux autres. Le thesaurus est un outil forgé pour la représentation de la base. Il caractérise lexicalement la sémantique de la base. La sémantique de chacun des documents est une valeur particulière de cette base sémantique globale.

Les approches statistiques ont bien ce mérite de prendre en compte la totalité des documents de la base pour représenter le contenu de chacun d'eux. En effet, ces approches éliminent totalement le sens des termes d'indexation pour ne considérer que leur valeur. Chacun des mots extraits de l'ensemble des documents se voit assigner une valeur, un poids représentant la valeur discriminative de chaque mot. La mesure retenue en général pour assigner le poids de chaque mot est : $tf \times idf$ où tf représente la fréquence du terme dans le document et idf, l'inverse de la fréquence relative du terme dans la collection [Sal89].

2.1.3 La représentation du sens : sens commun vs valeur

On le voit, à ce jour, la représentation du contenu des documents d'une base de données textuelles est partagée, et le demeure, entre deux conceptions du sens : l'une, l'approche statistique, est indissociable de la notion de valeur, l'autre, qu'elle soit traditionnelle (mots-clés) ou plus récente (intelligence artificielle) met en jeu des concepts, concepts appartenant à une sémantique générale dont on voit mal comment elle pourrait reposer sur autre chose que le sens commun.

Bien qu'actuellement tous les efforts en recherche, se sont orientés vers l'application des techniques de traitement automatique du langage et de représentation des connaissances [CROF87], le champs d'application de ces techniques reste limité, et présente beaucoup d'inconvénients : la richesse de la langue, la complexité et la diversité des relations sémantiques entre les concepts et l'importance des ressources de stockage pour la représentation, contribuent au rejet de ces techniques. L'approche statistique et probabiliste continuent à susciter l'intérêt de plusieurs applications commerciales.

2.2 Accès à l'information

Nous nous intéresserons dans cette partie à l'évolution et aux tendances manifestées dans l'appariement entre requêtes et représentants des textes d'une base de données. Ensuite, nous essaierons de montrer de quelles évolutions, les outils de recherche, ont fait l'objet ces dernières années.

2.2.1 Les modes d'appariement entre requêtes et documents

Apparier des formes

Appariement approché I : booléen et booléen étendu. La technique traditionnelle de recherche d'information est la recherche des solutions d'une équation booléenne de descripteurs.

C'est la technique qui prévaut très largement dans les systèmes commerciaux opérationnels. Belkin [BC87] trouve à la pérennité de cette situation les raisons suivantes :

- l'investissement réalisé dans ces systèmes est si considérable que les modifier ne serait pas économiquement viable;
- les techniques alternatives n'ont pas été testées dans des environnements en grandeur réelle;
- les résultats obtenus par des techniques alternatives ne sont pas suffisamment supérieurs, même au niveau expérimental, pour justifier les changements;
- les structures des requêtes booléennes expriment des aspects importants des besoins des utilisateurs.

Les inconvénients de cette technique sont bien connus [BC87] :

- de nombreux textes pertinents dont la représentation ne correspond qu'approximativement à la requête, ne sont pas retrouvés;

- les textes retrouvés ne sont pas ordonnés selon leur degré de pertinence;
- l'importance relative des concepts à l'intérieur de la requête ou à l'intérieur du texte n'est pas pris en compte;
- la formulation des requêtes est complexe;
- le résultat est fonction des deux représentations comparées qui doivent faire appel au même vocabulaire.

La rigidité de cette technique avait déjà été assouplie par l'utilisation des jokers et autres masques ainsi que par la mise en oeuvre plus ou moins automatique du thesaurus pour développer une requête.

Les techniques statistiques ne présentent pas ces inconvénients: l'appariement entre requête et documents peut être plus ou moins strict, l'importance relative des documents et des concepts est prise en compte, des règles rationnelles sont mises en oeuvre pour ordonner les documents retrouvés, les résultats sont meilleurs ou au moins comparables.

C'est pour remédier aux inconvénients des techniques booléennes de façon économiquement viable qu'une technique dite "booléenne étendue" a été mise au point [SFW78],[SV85]. C'est un cas particulier des techniques statistiques et probabilistes. Le principe est d'une part, de conférer aux termes de recherche de l'équation booléenne des poids et d'autre part, d'interpréter les opérateurs de l'équation comme des distances entre requêtes et documents.

Il s'agit là d'un compromis entre une technique d'appariement exact, la technique booléenne, et une technique d'appariement approché, la technique statistique, qui apparaît in-

dustriellement crédible. La recherche a démontré la supériorité des techniques d'appariement approché. Aux techniques statistiques vectorielles exposées en détail par Salton [Sal89] sont venues s'ajouter d'autres techniques d'appariement approché, les techniques de clusterisation.

Appariement approché 2 : clusters. L'hypothèse de groupement émise en premier lieu par [JR71] est que des documents très semblables les uns aux autres sont susceptibles d'être davantage pertinents pour une même requête que des documents ne témoignant pas du même degré de similitude entre eux. La clusterisation est une technique de classification qui permet d'identifier et de constituer des groupes, des clusters d'objets, ici de documents. En confrontant une requête aux clusters - à dire vrai à un représentant abstrait de ceux-ci - plutôt qu'à chaque document individuellement, on retrouve plus facilement et plus rapidement les documents pertinents. Le degré de similitude inter-document est calculé sur la base des descripteurs des documents. Ces descripteurs peuvent avoir été attribués manuellement ou avoir été obtenus par indexation automatique. De nombreux travaux actuels portent sur les mérites comparés de diverses techniques de clusterisation [Voo86],[Wil87],[Pan87],[EHW87].

Il faut remarquer que le principe de ces techniques de classification est mis à profit dans l'univers des hypertextes. L'utilisation conjointe des techniques statistiques et des techniques de clusterisation est souhaitable car toutes deux permettent de retrouver des ensembles de documents qui peuvent être différents.

La justification pour la mise en oeuvre de cette stratégie est fournie par des expériences qui ont montré que les deux stratégies tendent à retrouver des ensembles différents de documents et que, en particulier, la stratégie fondée sur les clusters parvient à retrouver des documents là où la recherche probabiliste échoue [CT87].

Apparier du sens

Inférence : Inférence est le nom donné par l'intelligence artificielle à la déduction des connaissances. Les opérations concrètes que ce nom recouvre dépendent du formalisme adopté pour représenter les connaissances. S'agissant d'un réseau sémantique, une opération dite d'activation par proximité est souvent utilisée en recherche documentaire. Une requête est utilisée pour activer les parties du réseau sémantique qui décrit le contenu des documents.

Dans les réseaux simples, les noeuds représentent les termes d'indexation qui sont connectés aux documents. Dans les réseaux plus riches en connaissances, les noeuds et les arcs représentent des concepts du domaine, les relations qu'ils entretiennent entre eux et les documents sur lesquels ils pointent. Une fois les noeuds de départ activés, l'activation se propage vers les autres noeuds en suivant les liens établis et en respectant certaines contraintes. Dans le système GRANT décrit par [CK87], les containtes sont de trois types: contrainte de distance, contrainte de branchement (l'activation est interrompue aux noeuds dont sont issus un trop grand nombre d'arcs), contrainte de chemin qui représente des méta-connaissances sur les cheminements privilégiés dans le réseau en fonction du domaine représenté. Selon les auteurs, l'activation par proximité sous contrainte obtient, avec des scores raisonnables de rappel et de précison, des résultats qui n'auraient pu être obtenus au moyen d'une simple recherche par mots-clés [CK87].

2.2.2 Les systèmes intelligents de recherche d'information

Une deuxième génération de systèmes intelligents de recherche d'information est en train de voir le jour. Ces systèmes prennent davantage en compte la situation de recherche d'information, tirant en cela partie des recherches cognitives qui conduisent à concevoir des systèmes qui modélisent les comportements observés de l'utilisateur, ici le demandeur de l'information.

Avant d'exposer plus en détail les types d'expertise incorporés dans les sytèmes, nous voudrions attirer l'attention sur le fait qu'un certain nombre des modules de ces systèmes implémentent des techniques éprouvées. Ainsi, les techniques statistiques ou probabilistes sont mises à contribution.

Mettre en oeuvre ensemble des outils et des techniques éprouvés :

Lors d'une session de recherche, nous pouvons recenser trois grands types de besoins des utilisateurs :

- besoins de vérification : l'utilisateur veut vérifier ou retrouver de l'information sur des éléments d'information aux caractéristiques connues.
- besoins conscients concernant un sujet : l'utilisateur veut clarifier, passer en revue ou approfondir certains aspects d'un sujet bien connu.
- besoins flous concernant un sujet : l'utilisateur veut explorer de nouveaux concepts sur des sujets non connus.

Nous allons à présent passer en revue des techniques bien éprouvées, qui ont l'avantage de correspondre à des situations concrètes de recherche d'information, en montrant comment chacune d'elle permet de répondre à l'un ou l'autre des types de besoins énumérés plus haut :

Browsing : C'est le troisième type de besoin qui visent à satisfaire les techniques de "browsing", de navigation, techniques qui constituent le mode typique d'accès à l'information dans les hypertextes.

Certains systèmes fondent leur programme sur cette situation de recherche d'information dans laquelle un utilisateur n'est pas à même de formuler une requête précise mais sera capable de reconnaître ce qu'il cherchait en le voyant. Le programme épargne à l'utilisateur qui, au démarrage de sa recherche n'est pas à même d'exprimer précisément son besoin, la nécessité de formuler d'entrée de jeu une requête précise et lui donne la possibilité, lorsqu'il découvre peu à peu qu'il sait ce qu'il veut, de formuler celle-ci.

Relevance feedback : Parmi les techniques pré-existantes à l'introduction de l'intelligence artificielle, et compatibles avec elle, la mise en oeuvre du "relevance feedback" est certainement celle dont l'utilité est unanimement admise. Elle constitue bien une stratégie naturelle de recherche d'information.

Dans un système statistique, le "relevance feedback" est effectué de la manière suivante [Sal86] : les résultats d'une première recherche sont utilisés automatiquement pour reformuler la requête en accroissant les poids des termes de la requête qui sont présents dans les documents retrouvés jugés pertinents par l'utilisateur, et à l'inverse, en diminuant les poids des

termes de la requête qui sont également présents dans les documents non pertinents retrouvés. La modification des poids peut s'accompagner d'une expansion de la requête à laquelle on adjoint des termes caractérisant les documents pertinents retrouvés.

Agrégation des termes et documents : Un autre type de technique permettant l'exploration d'une base de données est la mise en oeuvre de ce que Salton appelle "co-word analysis" et "document clustering". Il s'agit de rapprocher, les uns des autres, des descripteurs sur la base de leurs fréquences de co-occurence. Les ensembles de descripteurs constitués ainsi sur des bases purement statistiques (techniques de classification automatique) constituent des clusters. Salton [Sal89] examine l'utilisation de ces clusters pour constituer automatiquement des thesaurus alors que Michelet [MT85] et Can [CO90] envisagent d'en tirer parti pour donner à l'utilisateur des moyens pour constituer des classes de documents semblables en mesurant des liens de similitude de sujet entre ces documents.

2.3 Conclusion

Ce survol des recherches en informatique documentaire a abordé essentiellement deux thèmes : la représentation du contenu des documents d'une part, l'accès à l'information d'autre part. Les documents que nous avons envisagés sont essentiellement les documents textuels.

Les deux approches traditionnelles de la représentation du contenu des documents ont été confrontées : statistiques d'un côté, linguistico-conceptuelles de l'autre. Il est apparu que

cette opposition d'approche recouvrait en partie une dichotomie qui opposait sens et valeur.

S'agissant de l'accès à l'information, nous avons d'abord passé en revue les techniques d'appariement requête-documents. Nous avons distingué celles qui visaient à apparier des formes et celles qui visaient à apparier des sens.

Finalement, nous avons mis l'accent sur une nouvelle philosophie de recherche d'information qui vise à faire coopérer des outils, pour certains, déjà éprouvés: "relevance feedback", "browsing", "document clustering". Ces outils se donnent pour objectifs de prendre en compte les comportements de l'utilisateur en l'assistant dans l'activité de recherche.

Nous allons, dans les chapitres suivants, mettre l'accent sur la technique d'agrégation de documents, puisqu'elle constitue l'objet de notre recherche.

Chapitre 3

Classification automatique

La classification devient la base indispensable de nos activités, dès l'instant où nous avons à faire à de grands ensembles de documents. La division en classes facilite la recherche d'un document dans un ensemble.

La classification permet principalement de trouver des régularités dans un grand tableau de nombres en dégageant des nuages de points homogènes auxquels on essaiera de donner un sens. On parle ainsi de classification automatique : les classes seront obtenues au moyen d'algorithmes formalisés et non par des méthodes subjectives ou visuelles faisant appel à l'initiative du praticien.

Le but de ce chapitre étant de présenter le principe de ces différents algorithmes de classification automatique.

3.1 Les méthodes de classification automatique

On distingue deux grands types de méthodes de classification :

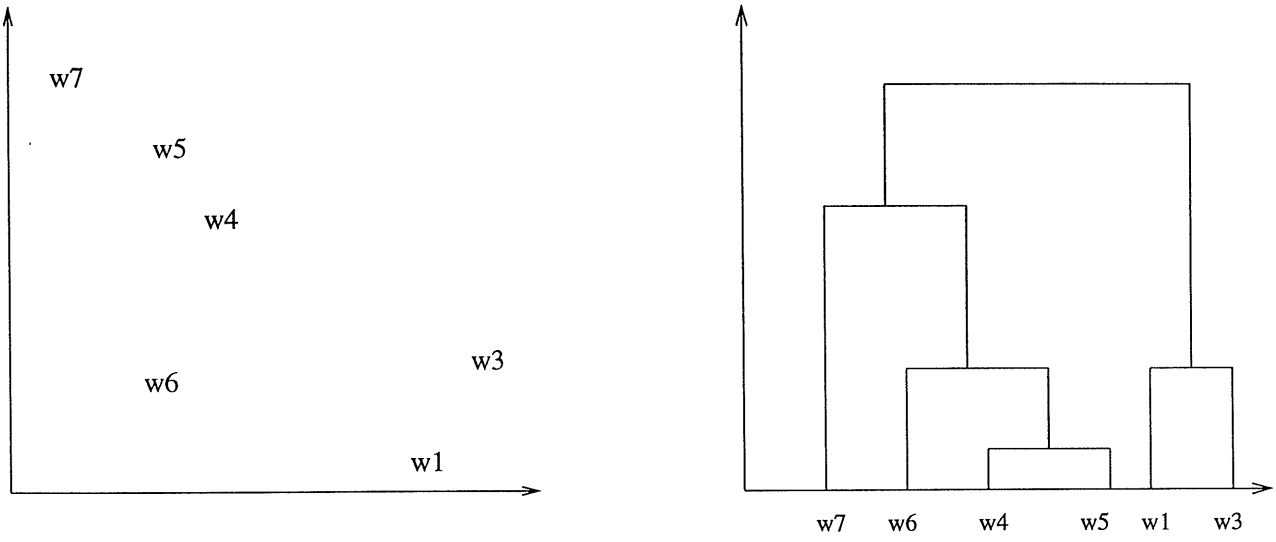
- les méthodes hiérarchiques qui produisent des suites de partitions en classes de plus en plus vastes à l'image des célèbres classifications des zoologistes en espèces, genres, familles, ordre, etc;
- les méthodes non hiérarchiques qui produisent directement une partition en un nombre fixé de classes.

3.1.1 Classification par hiérarchie

On parle de classification hiérarchique ou de hiérarchie, car chaque classe d'une partition est incluse dans une classe de la partition suivante. La suite des partitions obtenues est usuellement représentée sous la forme d'un arbre de classification analogue à l'organigramme d'une entreprise.

On cherche à représenter des points $w_1,...,w_n$ par un ensemble de parties hiérarchiquement emboîtées. Par exemple, la hiérarchie, indiquée [figure 3.1(b)], représente l'ensemble des points du plan donnés en [figure 3.1(a)], munis de la distance euclidienne.

L'interprétation d'une hiérarchie telle que celle indiquée [figure 3.1(b)] est la suivante : chaque palier de la hiérarchie couvre un groupe de points. La hauteur du palier est une mesure du degré d'agrégation du groupe qu'il couvre. Ainsi le groupe $\{w_4,w_5\}$ est plus agrégé



(a) nuage de points

(b) hierarchie

Figure 3.1 - *Classification hiérarchique.*

que le groupe $\{w_1,w_3\}$.

De manière plus précise, on peut définir une hiérarchie de la façon suivante :

Définition d'une hiérarchie

Soit Ω un ensemble fini, H un ensemble de parties (appelées paliers) de Ω, H est une hiérarchie sur Ω si :

1. $\Omega \in H$: c'est-à-dire le palier le plus haut contient tous les individus;
2. $\forall$ w $\in \Omega$,{w} $\in$ H (les points terminaux);
3. $\forall h, h' \in H$ on a : $h \cap h' \neq \emptyset \Rightarrow h \subset h'$ ou $h' \subset h$.

Exemple [figure 3.2] :

Soit $\Omega = \{w_1, w_2, w_3, w_4, w_5\}$, H est donné par le graphique de la figure 3.2.

On a : $h_1 = \{w_1, w_2, w_3, w_4, w_5\}$, $h_2 = \{w_1, w_2, w_3, w_4\}$, $h_3 = \{w_2, w_1\}$, $h_4 = \{w_1\}$, $h_5 = \{w_2\}$, $h_6 = \{w_3\}$, $h_7 = \{w_4\}$, $h_8 = \{w_5\}$.

H = $\{h_1,h_2,h_3,h_4,h_5,h_6,h_7,h_8\}$.

On vérifie facilement que les propriétés 1, 2 et 3 de la définition sont satisfaites.

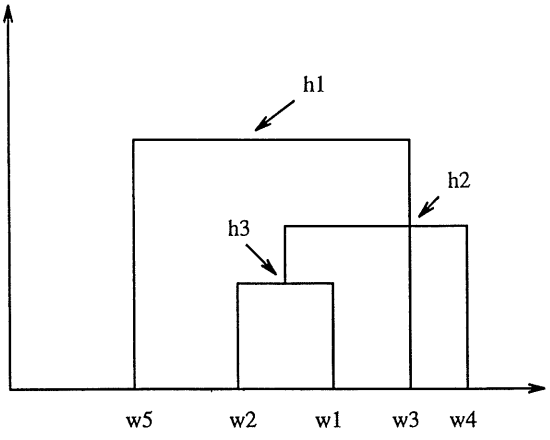


Figure 3.2 - *Représentation par hiérarchie de l'exemple.*

Indice d'agrégation

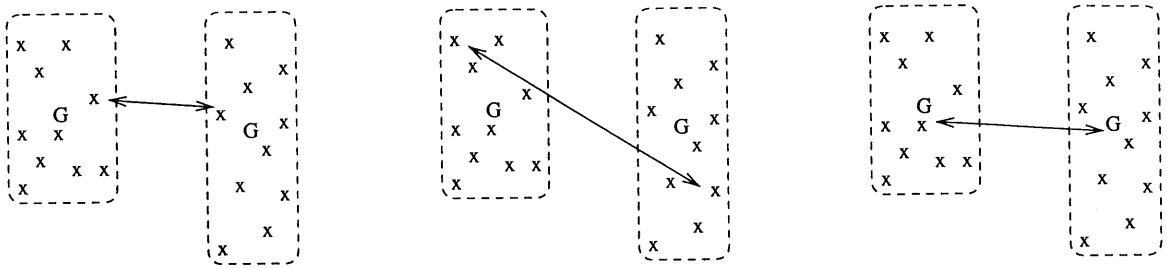
La construction d'une hiérarchie nécessite la connaissance d'une mesure de ressemblance entre individus entre eux, entre groupes entre eux et entre groupe et individu. Cette mesure de ressemblance s'appelle aussi indice d'agrégation. Elle est définie à partir des distances entre individus et des poids des parties de la hiérarchie. Parmi les plus utilisés, on peut citer :

- l'indice d'agrégation du lien minimum entre 2 parties h_1 et h_2 illustré par la [figure 3.3(a)] :

$$\delta(h_1, h_2) = min_{w_i \in h_1, w_j \in h_2} d(w_i, w_j)$$

- l'indice d'agrégation du lien maximum illustré par la [figure 3.3(b)] :

$$\delta(h_1, h_2) = max_{w_i \in h_1, w_j \in h_2} d(w_i, w_j)$$



(a) lien minimum (b) lien maximum (c) entre centres de gravite

Figure 3.3 - *Distances entre groupes.*

– l'indice des moyennes des distances illustré par la [figure 3.3(c)] :

$$\delta(h_1, h_2) = \frac{1}{\|h_1\|\|h_2\|} \sum_{w_i \in h_1, w_j \in h_2} d(w_i, w_j)$$

avec $\|h_i\|$ *le cardinal de* h_i

Autrement dit, l'indice d'agrégation entre deux parties est la moyenne des distances entre les individus pris dans les deux classes.

– l'indice d'agrégation des centres de gravité :

$$\delta(h_1, h_2) = d(G(h_1), G(h_2))$$

Algorithme

L'algorithme est simplement expliqué dans [LMF82]. Il procède par regroupements successifs :

1. partir d'une partition dont les classes sont réduites à un élément ;
2. rechercher le couple de classes dont l'indice d'agrégation est minimum (par exemple : agrégation en fonction du lien minimum);
3. construire la classe résultant de la fusion des deux précédentes;
4. recommencer la recherche jusqu'au regroupement final.

Le principal problème des méthodes de classification hiérarchique consiste à définir le critère de regroupement de deux classes, ce qui revient à définir une distance entre classes. Tous les algorithmes de classification hiérarchique se déroulent de la même manière : on recherche à chaque étape les deux classes les plus proches, on les fusionne, et on continue jusqu' ce qu'il n'y ait plus qu'une seule classe.

Cet algorithme est rapide. Au fur et à mesure des regroupements, le nombre d'itérations pour calculer les nouveaux indices d'agrégation diminue. L'algorithme permet de s'arrêter sur un nombre de classes donné a priori ou sur un critère calculé à partir de l'indice d'agrégation. Son pricipal inconvénient est son encombrement. De ce fait, il est déconseillé d'utiliser ce type d'algorithme pour la classification d'un grand nombre d'objets.

3.1.2 Classification par partitions

Il s'agit de regrouper n individus (ici documents) en k classes de telle sorte que les individus d'une même classe soient le plus semblables possible et que les classes soient bien séparées. Ceci suppose la définition d'un critère global mesurant la proximité des individus d'une même classe et donc la qualité d'une partition. Si on dispose d'un tel critère on pourrait imaginer d'examiner toutes les partitions possibles et de choisir la meilleure. Cette tâche est en réalité impossible, même avec les plus gros ordinateurs, dès que le nombre des individus dépasse quelques dizaines : pour 14 individus seulement il y a plus de 10 millions de partitions possibles en 4 classes!

Il est donc à peu près exclu de trouver la meilleure partition possible et il faudra se contenter d'algorithmes aboutissant à des solutions approchées.

Inertie interclasse et inertie intraclasse

Si on peut considérer les individus (ici des documents) comme des points d'un espace euclidien le problème de la classification peut se décrire comme la recherche d'une partition d'un nuage de n points en k sous-nuages. La dispersion d'un nuage de points est caractérisée par son inertie qui est la moyenne des carrés des distances au centre de gravité. Une classe sera donc d'autant plus homogène que son inertie sera faible. Appelons ξ_1, ξ_2,...,ξ_k, les inerties de chaque classe, calculées par rapport à leurs centres de gravité respectifs g_1, g_2,...,g_k et notons P_1, P_2,...,P_k les poids de ces classes : P_j est la somme des poids des individus de la classe j. La moyenne de ces inerties est appelée inertie intraclasse et est notée ξ_w :

$$\xi_w = \sum_{j=1}^{k} P_j \xi_j$$

Il est donc souhaitable que ξ_w soit la plus petite possible pour avoir un ensemble de classes très homogènes.

Considérons maintenant l'ensemble des k centres de gravité $g_1,...,g_k$, leur dispersion autour de g, centre de gravité du nuage total des n individus, est appelée inertie interclasse et est notée ξ_B :

$$\xi_B = \sum_{j=1}^{k} P_j d^2(g_j, g)$$

Une grande valeur de ξ_B indique une bonne séparation des classes et il conviendra donc que ξ_B soit la plus grande possible. Or ξ_B et ξ_W sont reliées par une importante formule généralisant le théorème de Huyghens :

$$\xi_B + \xi_W = \xi$$

où ξ est l'inertie totale du nuage des n points.

Rendre maximale ξ_B est donc équivalent à rendre minimale ξ_w puisque leur somme est constante.

Nous chercherons désormais à obtenir une partition en k classes où k a été fixé a priori. La plupart des techniques procèdent par améliorations successives d'une partition de départ : nous décrirons d'abord celle des centres mobiles puis la méthode des nuées dynamiques qui en est une variante.

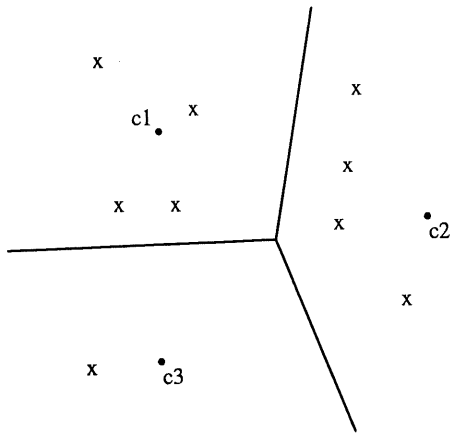


Figure 3.4 - *Partition associée à trois centres dans un plan.*

Regroupement autour de centres mobiles

Le déroulement de cet algorithme, dû à l'Américain Forgy [For65], est le suivant : dans un premier temps on regroupe les individus autour de k centres arbitraires c_1, c_2,..., c_k de la manière suivante : la classe associée à c_j est constituée de l'ensemble des individus plus proches de c_j que de tout autre centre. Géométriquement ceci revient à partager l'espace des individus en k zones définies par les plans médiateurs des segments $c_i c_j$. La [figure 3.4] ci-dessus donne un exemple d'une partition associée à trois centres dans un plan.

On calcule ensuite les centres de gravité g_1, g_2, ..., g_k des classes que l'on vient de former. On effectue alors une deuxième partition en regroupant les individus autour des g_j qui prennent alors la place des centres c_j de la première étape. On calcule les centres de gravité $g_1^{(2)}$,$g_2^{(2)}$, ...,$g_k^{(2)}$ de ces nouvelles classes, on regroupe les individus autour d'eux et ainsi de suite jusqu'à ce que la qualité de la partition mesurée par l'inertie intraclasse ne s'améliore

plus. Comme il suffit à chaque étape de calculer les nk distances entre les individus et les centres, il n'est pas nécessaire de conserver en mémoire les $\frac{n(n-1)}{2}$ distances différentes, ce qui est avantageux si n est grand.

L'inconvénient de cette méthode, à part le risque d'obtenir des classes vides, donc d'aboutir à moins de k classes, est de fournir une partition finale qui dépend de la partition de départ : on n'atteint pas l'optimum global mais seulement la meilleure partition possible à partir de celle de départ. De plus, la partition initiale est souvent arbitraire car il est courant de choisir les centres c_i par tirage au sort de k individus parmi n.

La méthode des nuées dynamiques

Sous ce nom évocateur, E. Diday a développé une méthode efficace de partitionnement que l'on peut considérer comme une généralisation de la méthode des centres mobiles [DL82]. Cette méthode est aussi connue sous le nom de "K-means clustering" en Amérique du nord. La différence fondamentale entre cette méthode et celle des centres mobiles est la suivante :

Au lieu de définir une classe par un seul point, son centre, qui peut ne pas être un des individus de l'ensemble à classer, on la définit par q individus formant un noyau qui, s'ils sont bien choisis, seront plus représentatifs de la classe qu'un simple centre de gravité. Ces noyaux permettront par la suite d'interpréter les classes.

Description de l'algorithme des nuées dynamiques : À partir d'un système initial de k noyaux on obtient une partition en regroupant les individus autour de ces noyaux. On calcule alors de nouveaux noyaux représentatifs des classes ainsi formées et recommence jusqu'à ce

que la qualité de la partition ne s'améliore plus. Formellement il faut donc disposer de trois fonctions :

- la première qui calcule la distance d'un individu à un noyau ;
- la deuxième qui à une partition en k classes associe les k noyaux de q points, représentatifs de ces classes ;
- la troisième qui mesure la qualité d'une partition.

Connaissant ces trois fonctions, le nombre de classes et l'effectif des noyaux, l'algorithme est entièrement déterminé.
Comme pour la méthode de Forgy la partition finale dépend du choix initial des noyaux. Afin de limiter cet inconvénient on procède à plusieurs tirages au sort des noyaux de départ et on compare les partitions finales obtenues : les individus qui ont toujours été classés ensemble définissent des "formes fortes" qui sont en quelque sorte les parties vraiment homogènes de l'ensemble des individus car elles ont résisté aux aléas des tirages des noyaux.

Les méthodes de partitionnement permettent de traiter rapidement de grands ensembles d'individus mais elles supposent que le nombre k de classes est fixé. Toutes les frontières entre les classes sont les droites médiatrices des segments joignant les centres de gravité. Leurs équations en x et y sont linéaires. Une variante de cette méthode "Fuzzy C-Means", faisant intervenir la logique floue, consiste à autoriser une appartenance partielle de chaque point aux différentes classes, ce qui peut facilement éclairer la décision d'attribuer un document à une classe. Nous allons présenter brièvement dans ce qui suit, le principe de cette méthode.

Classification floue

L'algorithme de la classification floue FCM, permet de construire à partir de la donnée d'un certain nombre d'observations, un ensemble de points qui gravitent autour d'un centroïde et définissent une classe. L'algorithme procède par une sélection arbitraire de noyaux, auxquels il agrège les observations disponibles. Il calcule ensuite, à partir des ensembles affectés à chaque noyau, une nouvelle position de ce noyau, et réitère le processus d'affectation des observations. Le principe de la classification floue est d'autoriser une appartenance partielle ou distribuée d'une forme donnée à l'ensemble des classes de l'espace des individus. Afin de minimiser l'erreur de misclassification, on fait appel aux techniques de classification adaptative floue introduites initialement par Bezdek, qui minimisent la fonctionnelle suivante [Bez73, BP92] :

$$J(u_{ij}, v_k) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} \| x_j - v_i \|^2, \ m > 1 \qquad (1)$$

où $\| x_j - v_i \|^2$ est la distance entre le point x_j et le centroïde v_i de la classe i, u_{ij} est le degré d'appartenance du point j à la classe i. u_{ij} est toujours positif et sa somme sur toutes les classes est égale à 1. Cette contrainte $\sum_{i=1}^{c} u_{ij} = 1$ force la méthode de classification à distribuer la totalité des données sur c sous-ensembles flous [BP92]. L'algorithme de classification se déduit de la fonctionnelle (1) qui peut être résumé de la manière suivante :

- étape 1: Fixer le nombre de classe c (le nombre de classes est déterminé grâce à un algorithme de pré-classification, qu'on se propose d'implémenter) et le facteur flou m. Calculer la position initiale des centres v_i des classes.

- étape 2: Générer une partition en utilisant l'équation suivante du degré d'appartenance:

$$u_{ij} = \frac{(\frac{1}{\|x_j - v_i\|^2})^{\frac{1}{m-1}}}{\sum_{k=1}^{c}(\frac{1}{\|x_j - v_k\|^2})^{\frac{1}{m-1}}}, \quad i = 1, ..., c; j = 1, ..., n$$

- étape 3: Calculer les nouveaux centres des classes en utilisant l'équation suivante (pour i=1 à c)

$$v_i = \frac{1}{\sum_{j=1}^{n} u_{ij}^m} \sum_{j=1}^{n} u_{ij}^m x_j, \quad i = 1, ..., c$$

- étape 4: S'arrêter quand la partition est stable, sinon retourner à l'étape 2. x_j est un élément de l'ensemble de n points, m est le facteur flou généralement fixé à 2. $\| x_j - v_k \|^2$ est défini comme étant la distance euclidienne entre le point x_j de l'ensemble des données et le centre v_k de la classe, pour (i=1 à c).

La sortie finale de l'algorithme FCM est une liste constituée par les centres des différentes classes et les degrés d'appartenance de chaque point de l'espace. Ces informations sont obtenues par application de l'algorithme FCM que nous nous proposons de l'implanter sur une matrice documents-termes, ayant subi des traitements préalables et étant transformée en matrice documents-documents. Ces mêmes informations sont utilisées pour construire un système d'inférence flou, dont le but est de défuzzifier les résultats de la classification, en interprétant les résultats des calculs de la distance de rapprochement des documents entre eux.

3.2 Conclusion

L'algorithme FCM offre, dans le cadre de la classification documentaire, l'énorme intérêt d'avoir un fonctionnement itératif. En effet, la variation quasi permanente de la source des documents et la versatilité de l'utilisateur réclament une méthode adaptative ne nécessitant pas une remise à zéro totale du système.

Outre cet aspect, primordial selon nous, la classification floue présente l'avantage, par rapport aux autres techniques de classification, de mieux éclairer la décision de classement grâce à l'information inhérente véhiculée par les degrés d'appartenance.

Chapitre 4

Étape de pré-classification

Avant d'entamer le processus de classification, un travail de préparation des données est nécessaire. Il s'agit d'abord de choisir le modèle de représentation des connaissances, de calculer les degrés de corrélation entre documents, d'expliciter le fondement théorique derrière cet aspect et de déterminer finalement un estimatif du nombre des classes, paramètre essentiel pour l'algorithme de classification floue FCM. Nous allons, au cours de ce chapitre, détailler tous ces points tout en les illustrant par des exemples.

4.1 Choix du modèle de représentation des connaissances

Parmi les modèles utilisés pour représenter une base documentaire, on a retenu le modèle vectoriel [Sal89]. Ce dernier associe à chaque document un vecteur numérique de dimension égale au nombre de termes d'indexation du système. Ce vecteur représente soit d'une façon binaire la présence ou l'absence du terme dans le document, soit la pondération de chacun des termes d'indexation par rapport au document.

Le principal inconvénient de ce modèle réside dans la nécessité de définir des espaces vectoriels où la dépendance sémantique entre les dimensions (termes) reste faible.

Dans un contexte de recherche et de classification documentaire, utilisant un modèle de représentation vectoriel, une collection de documents est représentée par une matrice documents-termes appelée la matrice D. Chaque ligne de cette matrice représente un seul document et est définie par un ensemble de termes. L'ensemble de tous les documents de la matrice constituent la base de données documentaire. Les éléments composant une ligne sont appelés des termes index. La matrice D peut être générée manuellement ou automatiquement par une procédure d'indexage [Sal78]. Dans le cas d'un indexage binaire, les entrées d_{ij} $(1 \leq i \leq m, 1 \leq j \leq n)$ de la matrice D indiquent la présence ou l'absence d'un terme t_j dans un document d_i. Le terme t_j constitue un membre du vocabulaire d'indexage $T = \{t_1, t_2, t_3, ..., t_n\}$. La matrice D possède aussi deux propriétés essentielles :

- $\sum_{j=1}^{n} d_{ij} \geq 1,\ i = 1, ...m$ (chaque document est décrit par au moins un terme)

$$D = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Figure 4.1 - *La matrice documents-termes D.*

- $\sum_{i=1}^{m} d_{ij} \geq 1,\ j = 1, ...n$ (chaque terme est assigné à au moins un document).

Les concepts de notre algorithme vont être accompagnés par une illustration [figure 4.1], soit la matrice D, représentant une collection de 5 documents. Ces derniers sont décrits par un ensemble de 6 termes.

4.2 Construction de la matrice CC (coefficients de couverture)

La matrice CC est une matrice (documents-documents) dérivée à partir de la matrice D (documents-termes), dont les entrées c_{ij} $(1 \leq i, j \leq m)$ indiquent la probabilité de sélectionner un terme de d_i à partir de d_j.
En d'autres termes, la matrice CC exprime les relations qui existent entre les documents basées sur le calcul de probabilités en deux étapes [CO83]. La première étape consiste à choisir

arbitrairement un terme t_k du document d_i; la deuxième étape consiste à choisir le même terme t_k sélectionné à partir du document d_j.

En terme de probabilité, ceci se traduit par la considération de deux évènements :

- s_{ik} représente l'évènement de sélectionner t_k à partir de d_i.
- s_{jk} représente l'évènement de sélectionner d_j à partir de t_k.

Dans cette expérience, la probabilité de l'évènement "s_{ik} et s_{jk}" notée P(s_{ik}, s_{jk}) est égale à $P(s_{ik}) \times P(s_{jk})$.

$$P(s_{ik}) = d_{ik} \times \left[\sum_{h=1}^{n} d_{ih}\right]^{-1}, \quad P(s_{jk}) = d_{jk} \times \left[\sum_{h=1}^{m} d_{hk}\right]^{-1}$$
$$pour\ 1 \leq i, j \leq m,\ 1 \leq k \leq n.$$

Ainsi, en considérant un document d_i, on peut représenter la matrice D avec un modèle probabiliste hiérarchique à deux étapes [CO90]. Si l'on choisit un chemin qui commence à partir d'un document d_i et qui se termine par un des documents d_j de la base, il existe n façons possibles $(s_{i1}, s_{i2}, ..., s_{in})$ pour atteindre d_j $(1 \leq j \leq m)$ à partir de d_i [figure 4.2].

Supposons qu'on choisisse s_{ik}, ainsi l'étape d'arrêt intermédiaire serait t_k. À partir de ce terme, et pour atteindre d_j, il faudra choisir le chemin s_{jk}. En utilisant la [figure 4.2], et pour calculer les entrées c_{ij} de la matrice CC, représentant la probabilité de choisir un terme de d_i à partir de d_j, il faudra calculer la somme des probabilités des chemins individuels de d_i à d_j. Pour illustrer le concept de coefficient de couverture, on va utiliser la matrice D [figure 4.1] pour le calcul de l'entrée c_{12}. En se référant à la matrice D, on remarque que le document d_1

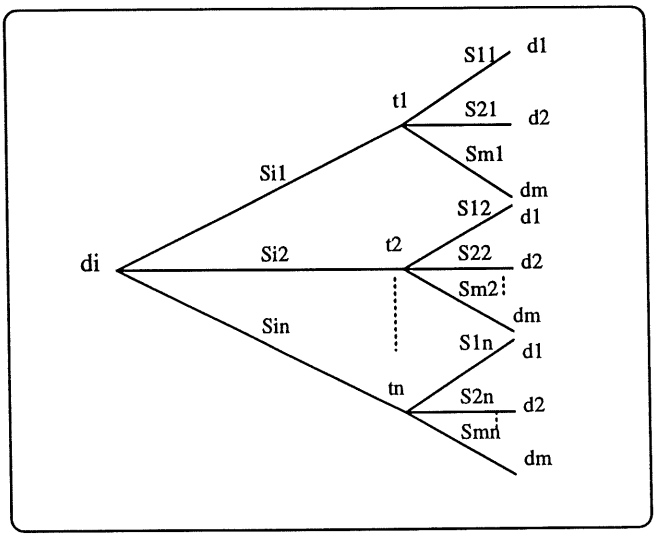


Figure 4.2 - *Représentation hiérarchique en deux étapes du modèle probabiliste des entrées d_i de la matrice D.*

contient trois termes t_1, t_2, t_5. En suivant le principe du modèle probabiliste à deux étapes pour calculer l'entrée c_{12}, on doit d'abord sélectionner arbitrairement chaque terme de d_1, ensuite essayer de sélectionner d_2 à partir de la sélection du terme résultat de la première étape. À la première étape, la probabilité de choisir chaque élément de t_1, t_2, t_5 à partir de d_1 est de $\frac{1}{3}$. Si l'on sélectionne t_1 ou t_5, alors d_2 aura $\frac{1}{2}$ de chance d'être sélectionné, par contre, si on sélectionne t_2, alors la probabilité de choisir d_2 à la seconde étape sera de $\frac{1}{4}$. Ceci est dû au fait que t_1 et t_5 apparaissent dans d_1 et d_2, alors que t_2 apparaît dans d_1, d_2 et dans deux autres documents. L'entrée c_{12} est calculée alors comme suit :

$$c_{12} = \sum_{k=1}^{6} s_{1k} \times s_{2k} = \frac{1}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{1}{4} + 0 \times 0 + 0 \times \frac{1}{2} + \frac{1}{3} \times \frac{1}{2} + 0 \times 0 = 0.417.$$

Le calcul des entrées c_{ij} peut être résumé par la formule suivante :

$$c_{ij} = \alpha_i \times \left[\sum_{k=1}^{n} d_{ik} \times \beta_k \times d_{jk} \right],$$

$$avec\ 1 \leq i, j \leq m,$$

avec α_i et β_k, représentant respectivement les inverses de la somme de la ième ligne et la kième colonne, comme indiqué ci-dessous :

$$\alpha_i = \left[\sum_{j=1}^{n} d_{ij} \right]^{-1}, \qquad 1 \leq i \leq m,$$

$$\beta_k = \left[\sum_{j=1}^{m} d_{jk} \right]^{-1}, \qquad 1 \leq k \leq n.$$

4.3 Propriétés de la matrice CC

Les propiétés suivantes caractérisent la matrice CC [figure 4.3] :

1. pour $i \neq j$, $0 \leq c_{ij} \leq c_{ii}$ et $c_{ii} > 0$.
2. $c_{i1} + c_{i2} + ... + c_{im} = 1$ (la somme des valeurs d'une ligne i est égale à 1, pour $1 \leq i \leq m$).
3. si aucun des termes de d_i n'est utilisé par les autres documents, alors $c_{ii} = 1$, sinon, $c_{ii} < 1$.
4. si $c_{ij} = 0$, alors $c_{ji} = 0$, similairement, si $c_{ij} > 0$, alors $c_{ji} > 0$; mais généralement $c_{ij} \neq c_{ji}$.
5. $c_{ii} = c_{jj} = c_{ij} = c_{ji}$, si et seulement si d_i et d_j sont identiques.

- Propriété (1) : $c_{ij} \geq 0$ signifie que soit $(c_{ij} > 0)$, et dans ce cas, on peut sélectionner des termes de d_i à partir de d_j, soit $(c_{ij} = 0)$ et ceci implique que d_i et d_j ne partagent aucun terme en commun. L'inégalité $c_{ij} \leq c_{ii}$ est évidente dans le sens où le document d_i aura plus de chances d'être sélectionné à partir des termes qui le représentent, que n'importe quel autre document. Cependant, si d_j contient tous les termes de d_i alors $c_{ij} = c_{ii}$. Nous pouvons, bien sûr toujours sélectionner un terme de d_i à partir de lui même, d'où $c_{ii} > 0$.

- Propriété (2) : Elle s'explique par le fait que les entrées de chaque ligne de la matrice CC sont les résultats d'expériences probabilistes en deux étapes, et que la somme de toutes ces probabilités est égale à 1.

- Propriété (3) : Elle s'explique par le fait que si des termes de d_i apparaîssent dans un autre document, par exemple d_j, l'entrée c_{ij} correspondante aura une valeur différente de zéro pour $i \neq j$.

- Propriété (4) : Elle s'explique par le fait que s'il n'est pas possible de sélectionner un terme de d_i à partir de d_j alors il ne sera pas possible de sélectionner un terme de d_j à partir de d_i (d_i et d_j ne partagent aucun terme en commun). Par contre, si d_i et d_j ont des termes en commun mais ne sont pas pas identiques, alors c_{ij} et c_{ji} auront des valeurs supérieures à zéro mais non nécessairement égales. Expliqué autrement, nous pouvons remarquer que, contrairement aux mesures de similarité comme le cosinus, les mesures d'association entre deux documents définies par le concept CC (coefficient de couverture) ne sont pas symétriques. Ceci est dû à la nature probabiliste du concept, qui n'est autre qu'un exemple de probabilité conditionnelle pour des évènements aléatoires [And73].

- Propriété (5) : Si deux documents sont parfaitement identiques, alors leurs coefficients de couverture mutuels (c_{ij} et c_{ji}) et leurs coefficients de couverture sur eux-mêmes (c_{ii} et c_{jj}) sont égaux.

Ces propriétés montrent que si un document partage des termes en commun avec d'autres documents dans la base, alors on observera beaucoup de valeurs non nulles de part et d'autre de l'entrée de la diagonale de la ligne correspondante [CO85, CO90]. Étant donné que la somme des valeurs d'une ligne est égale à 1, l'entrée à la diagonale est nécessairement inférieure à 1. Dans le cas contraire, c'est-à-dire lorsqu'un document partage seulement quelques termes avec les autres documents, on observera beaucoup de valeurs nulles de part et d'autre

de l'entrée de la diagonale, celle-ci aura une valeur proche de 1. Si, par contre, un document ne partage aucun de ses termes avec les autres documents, alors l'entrée à la diagonale sera égale 1 et les autres entrées de la ligne auront une valeur nulle. À partir de ces propriétés, les entrées c_{ij} peuvent être interprétées de la manière suivante :

- pour $i \neq j$, l'étendue de la couverture de d_i par d_j (degré de couplage de d_i avec d_j).
- pour i = j, l'étendue de la couverture de d_i par lui même (degré de découplage de d_i par rapport au reste des documents).

Pour obtenir une meilleure explication de la signification de la matrice CC, nous considérons deux vecteurs documents d_i et d_j, auxquels nous allons définir quatre types de relations possibles en termes d'entrées de la matrice CC (c_{ij}, c_{ji}, c_{ii} et c_{jj}) :

- **Documents identiques :** Les coefficients de couplage et de découplage des deux documents sont équivalents ($c_{ij} = c_{ji}$). De plus, les degrés de couverture de ces deux documents par les autres documents de la base sont identiques ($c_{ik} = c_{jk}, 1 \leq k \leq m$). Similairement, les degrés de couverture des autres documents par d_i et d_j sont les mêmes ($c_{ki} = c_{kj}, 1 \leq k \leq m$).
- **Documents chevauchés :** Chaque document couvre lui-même plus que n'importe quel autre document. Ceci ne nous donne pas suffisamment d'informations pour comparer c_{ii} avec c_{jj} et c_{ij} avec c_{ji}. Ceci est dû au fait que ces valeurs sont affectées par leurs degrés de couplage avec les autres documents de la base.

- **Un document, sous-ensemble d'un autre document :** Supposons que d_i est un sous-ensemble de d_j, le degré de couverture de d_i par lui-même (c_{ii}) sera égal à celui du degré de couverture de d_i par d_j (c_{ij}). De plus, du fait que d_j contient tous les termes de d_i et même plus, son degré de couverture par lui-même sera plus élevé que celui de d_i par lui-même ($c_{jj} > c_{ii}$). Par le même raisonnement, le degré de couverture de d_j par d_i est plus élevé que celui de d_i par d_j, ($c_{ij} > c_{ji}$).

- **Documents disjoints :** Du fait que d_i et d_j ne partagent aucun terme en commun, leurs degrés de couverture mutuels seront nuls ($c_{ij} = c_{ji} = 0$). Evidemment, ils se couvrent eux-mêmes, mais chacun d'eux peut être couplé avec les autres documents de la base. C'est pour cette raison que c_{ii} et c_{jj} auront une valeur inférieure ou égale à 1.

D'après ce qu'il a été montré dans les discussions précédentes, la matrice D [figure 4.1] est une matrice relativement distincte, dans la mesure où les d_i ne partagent que quelques descripteurs avec les autres documents de la base. Comme illustration de tous ces concepts décrits plus haut, nous présenterons la matrice CC [figure 4.3] dérivée à partir de la matrice D donnée dans la section (4.1).

Ce phénomène est traduit au niveau de la matrice CC par les entrées c_{ii} qui détiennent les valeurs les plus élevées. L'entrée c_{ii} est appelée alors coefficient de découplage, δ_i de d_i. Notons que δ_i exprime le degré de liaison d'un document aux autres documents et c'est pourquoi le mot coefficient est attribué.

La somme sur une ligne des entrées autres que la diagonale indique le degré de couplage de d_i avec les autres documents de la base. Cette mesure est aussi appelée coefficient de couplage ψ_i de d_i. À partir des propriétés de la matrice CC :

$$CC = \begin{bmatrix} 0.417 & 0.417 & 0.000 & 0.083 & 0.083 \\ 0.313 & 0.438 & 0.000 & 0.063 & 0.188 \\ 0.000 & 0.000 & 0.333 & 0.333 & 0.333 \\ 0.083 & 0.083 & 0.111 & 0.361 & 0.361 \\ 0.063 & 0.188 & 0.083 & 0.271 & 0.396 \end{bmatrix}$$

Figure 4.3 - *La matrice documents-documents CC.*

$\delta_i = c_{ii}$: coefficient de découplage de d_i;

$\psi_i = 1 - \delta_i$: coefficient de couplage de d_i.

Les domaines de δ_i et ψ_i sont $0 < \delta_i \leq 1$ et $0 \leq \psi_i < 1$, pour $1 \leq i \leq m$.

En utilisant les valeurs individuelles de δ_i et ψ_i, le coefficient moyen de couplage et de découplage des documents sera calculé de la manière suivante :

$$\delta = \frac{\sum_{i=1}^{m} \delta_i}{m} \quad 0 < \delta \leq 1;$$

$$\psi = \frac{\sum_{i=1}^{m} \psi_i}{m} \quad 0 \leq \psi < 1.$$

Les coefficients de couplage et de découplage de la base documentaire entière sont calculés comme suit :

$$\delta = \frac{\sum_{i=1}^{5} \delta_i}{5} = 1.945/5 = 0.389, \quad \psi = 1 - \delta = 0.611.$$

En examinant la matrice D, nous pouvons remarquer que d_1 est un sous-ensemble de d_2. Ceci est vérifié par les valeurs $c_{11} = c_{12}$. Les autres relations sont validées par les valeurs suivantes : $c_{22} > c_{21}$, $c_{22} > c_{11}$ et $c_{12} > c_{21}$.

4.4 Calcul du nombre de classes

L'estimation du nombre de classes pour une base de données, et en général pour un ensemble de données multivariées, n'est pas une tâche facile. Dans les méthodes de classification hiérarchique, le nombre de classes peut varier de 1 (la base entière) à m, (m étant le nombre d'objets contenus dans la base). Dans un contexte de classification documentaire, le calcul de la distance euclidienne entre documents ne constitue pas un bon estimateur du nombre de classes. En effet, deux documents peuvent s'avérer très similaires même s'ils ne partagent aucun terme en commun.

Théoriquement, le nombre de classes doit être supérieur ou égal à 1 et inférieur ou égal à m, (la taille de la base). Or, en réalité, les documents présents dans la base ne sont pas tous identiques, ni complètement distincts.

En partant de l'hypothèse que le nombre de classes doit être élevé, si les documents ne sont pas similaires, nous pouvons calculer alors le nombre de classes sur la base des coefficients de découplage. Comme l'on vient de le mentionner précédemment, les c_{ii} représentent des

coefficients de découplage et indiquent le degré de non-liaison d'un document par rapport aux autres documents de la base.

$\delta_i = c_{ii}$: coefficient de découplage de d_i.

$\psi_i = 1 - \delta_i$: coefficient de couplage de d_i.

Le nombre de classes est alors déterminé par la formule suivante :

n_c = (coefficient de découplage de la base) * (nombre de documents)

On peut ainsi calculer le nombre de classes de la matrice D :

$$
\begin{aligned}
n_c = & \\
& = \sum_{i=1}^{m} \delta_i \times m = \delta \times m \\
& = trace(CC) \\
& = (0.417 + 0.438 + 0.333 + 0.361 + 0.396) = 0.389 \times 5 = 1.945 \\
& \approx 2
\end{aligned}
$$

4.5 Construction de la matrice des entrées E_0

À ce niveau, on a calculé un estimatif du nombre des classes (n_c) de la base documentaire. Comme deuxième étape, on va essayer de déterminer les documents ayant les plus forts poids, et qui sont candidats à occuper les centres de ces classes-là. La mesure $\psi_i \delta_i t_i$ peut être

appropriée pour choisir les documents pivots :

- Le paramètre ψ_i, appelé également coefficient de couplage, indique le degré de connexion entre les documents;
- Le paramètre δ_i, appelé également coefficient de découplage, nous donne une idée sur le degré de séparation entre les documents;
- Le paramètre t_i (nombre de termes indexés par document), est un paramètre de normalisation du produit $\psi_i\delta_i$ qui peut être grand, dans certains cas, sans que ce dernier soit un véritable document pivot. Un document pivot doit alors couvrir un grand nombre de termes afin de permettre une association ou un lien avec les différents autres documents.

Le produit $\psi_i\delta_i t_i$ est appelé alors poids des documents pivots. Cette mesure est calculée pour chaque document, et les n_c premiers documents ayant les plus forts poids seront retenus comme documents centres ou pivots.

En se référant à l'exemple, on va calculer sur la base de la matrice D et la matrice CC, les différents poids des documents. Ainsi les P_i sont calculés dans un ordre décroissant comme suit :

$$P_2 = 0.438 \times (1 - 0.438) \times 4 = 0.985$$
$$P_5 = 0.396 \times (1 - 0.396) \times 4 = 0.957$$
$$P_1 = 0.417 \times (1 - 0.417) \times 3 = 0.730$$
$$P_4 = 0.361 \times (1 - 0.361) \times 3 = 0.693$$

$P_3 = 0.333 \times (1 - 0.333) \times 1 = 0.222$

Sachant que la base va être décomposée en deux classes et que les documents d_2 et d_5 sont les documents ayant les plus forts poids, ces derniers vont constituer les documents pivots de la matrice des entrées E_0.

Pour construire la matrice des entrées, on va conserver les entrées c_{ii} de la matrice CC, à savoir c_{22} et c_{55}. Les entrées e_{ij} seront calculées de la manière suivante :

$$e_{ij} = max(c_{ij}, c_{ji})$$

$$i = \{2, 5\},\ j = 1, 2, ..., 5$$

La matrice E_0 résultat est la suivante :

$$E_0 = \begin{bmatrix} 0.417 & 0.083 \\ 0.438 & 0.188 \\ 0.083 & 0.333 \\ 0.083 & 0.361 \\ 0.188 & 0.396 \end{bmatrix}$$

4.6 Complexité de l'étape de pré-classification

La complexité de l'algorithme de pré-classification peut être déterminée à partir des données suivantes :

- le nombre du vocabulaire d'indexation : n;

– le nombre de documents composant la base : m.

Pour cette étape, les coefficients de couverture, constituant les entrées de la matrice CC, sont calculés en faisant le parcours de chacune des lignes et des colonnes de la matrice D. La complexité de l'algorithme est alors de l'ordre de $O(mn)$.

Chapitre 5

Classification floue

Utilisé généralement dans la reconnaissance de formes, l'algorithme de regroupement flou FCM dispose, par rapport aux algorithmes de regroupement non flou, d'une information additionnelle qui est le degré d'appartenance d'un objet aux différents groupements.

En faisant l'analogie avec les travaux qui ont été menés dans ce domaine, il s'agit d'appliquer le même processus sur les bases de données documentaires. L'objectif étant de structurer et d'organiser ces bases afin d'augmenter l'efficacité de la recherche des documents pertinents répondant aux besoins des utilisateurs.

Après avoir préparé le terrain avec l'étape de pré-classification et surtout déterminé le nombre de classes, il s'agit, dans ce chapitre, de mettre l'accent sur l'analyse de l'algorithme de classification floue. Ce dernier va être appliqué sur un exemple afin de montrer et d'interpréter ses résultats, c'est la phase de défuzzification. Un autre volet consiste à présenter un

algorithme que nous avons conçu pour traiter le cas des bases documentaires dynamiques. Les différentes étapes de cet algorithme vont être accompagnées par un exemple. Finalement, afin de valider notre travail, nous allons présenter un algorithme de classification non floue basé toujours sur le concept de coefficient de couverture [CO90]. Ce dernier a été implanté afin de quantifier les avantages de notre système.

5.1 L'algorithme de classification floue basé sur des relations probabilistes

À travers le processus d'acquisition de données, l'environnement observé est constitué par la matrice des entrées E_0 calculée à l'étape de pré-classification. Cette matrice n'est autre qu'un ensemble de documents $E_0 = \{d_1, d_2, ..., d_n\}$, définissant un espace euclidien de dimension p, $\mathcal{R}^p$, c'est à dire $d_j \in \mathcal{R}^p, j = 1, 2, ..., n$.

Le problème est de faire une partition de cette collection en c ensembles flous selon un critère donné, ici c est le nombre donné de groupements. Le critère est une fonction objective qui agit comme indice de performance du groupement. Le résultat du groupement flou peut s'exprimer par une matrice de partition U ainsi :

$$U = [u_{ij}]_{i=1..c,j=1..n} \tag{1}$$

où u_{ij} est une valeur numérique dans [0,1] qui exprime le degré avec lequel le document d_j appartient au ième groupement. Il y a des contraintes sur u_{ij}.

Tout d'abord, l'appartenance totale du document $d_j \in E_0$ dans toutes les classes est égale à 1 :

$$\sum_{i=1}^{c} u_{ij} = 1 \; pour \; tout \; j = 1, 2, ..., n \tag{2}$$

ensuite chaque groupement construit est non vide et différent de l'ensemble complet :

$$0 < \sum_{j=1}^{n} u_{ij} < n \; pour \; tout \; i = 1, 2, ..., c \tag{3}$$

Une forme générale de fonction objective est :

$$J(u_{ij}, v_k) = \sum_{i=1}^{c} \sum_{j=1}^{n} \sum_{k=1}^{c} g[w(d_i), u_{ij}] d(d_j, v_k) \tag{4}$$

où $w(d_i)$ est le poids à priori de chaque d_i et $d(d_j, v_k)$ est le degré de non similitude entre la donnée d_j et l'élément v_k qui est le vecteur central du k^{ieme} groupement. Le degré de non-similitude est une mesure qui satisfait les deux axiomes :

$$(i) \; d(d_j, v_k) \geq 0, \tag{5}$$

$$(ii) \; d(d_j, v_k) = d(v_k, d_j) \tag{6}$$

Avec ceci, le groupement flou peut se formuler comme un problème d'optimisation :

$$Minimiser \; J(u_{ij}, v_k), \; i, k = 1, 2, ..., c \; : j = 1, 2, , ..., n \tag{7}$$

avec les contraintes des équations (2) et (3).

Une méthode largement répandue basée sur l'équation (7) est le "Fuzzy C-Means" (FCM) de Bezdek. La fonction d'optimisation de l'algorithme FCM prend la forme de :

$$J(u_{ij}, v_k) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m \| d_j - v_i \|^2, \ m > 1 \tag{8}$$

Ici m est appelé poids exponentiel, il influence le degré de flou de la matrice d'appartenance. Pour solutionner ce problème de minimisation, il faut différentier la fonction objective de l'équation (8) par rapport à v_i (pour u_{ij} fixe, $i = 1, ..., c$, $j = 1, ..., n$) et à u_{ij} (pour v_i fixe, $i = 1, ..., c$) et appliquer les conditions de l'équation (2). On obtient :

$$v_i = \frac{1}{\sum_{j=1}^{n} u_{ij}^m} \sum_{j=1}^{n} u_{ij}^m d_j, \quad i = 1, ..., c \tag{9}$$

$$u_{ij} = \frac{(\frac{1}{\|d_j - v_i\|^2})^{\frac{1}{m-1}}}{\sum_{k=1}^{c} (\frac{1}{\|d_j - v_k\|^2})^{\frac{1}{m-1}}}, \quad i = 1, ..., c; j = 1, ..., n \tag{10}$$

Le système d'équations précédent ne peut pas se résoudre analytiquement. L'algorithme FCM donne une approche itérative pour minimiser la fonction objective à partir d'une position donnée.

Algorithme FCM :

- **Étape 1 :** Choisir un nombre de clusters c ($2 \leq c \leq n$) et un poids exponentiel m ($1 < m < \infty$). Choisir une matrice initiale de partition $U^{(0)}$ et un critère d'arrêt ϵ. Mettre l'indice l à 0.

- **Étape 2 :** Calculer les centres des clusters flous $\{v_i^{(l)} \mid i = 1, 2, ..., c\}$ en utilisant $U^{(l+1)}$ et l'équation (9).
- **Étape 3 :** Calculer une nouvelle matrice de partition $U^{(l+1)}$ en utilisant $\{v_i^{(l)} \mid i = 1, 2, ..., c\}$ et l'équation (10).
- **Étape 4 :** Calculer $\Delta = \| U^{(l+1)} - U^{(l)} \| = max_{i,j} \| u_{ij}{}^{(l+1)} - u_{ij}^{(l)} \|$. Si $\Delta > \epsilon$, poser $l = l + 1$ et retourner à l'étape 2. Si $\Delta \leq \epsilon$, arrêter.

Cette procédure itérative minimise la fonction objective de l'équation (8) et mène à un minimum local.

5.2 Analyse de la complexité de l'algorithme

Bien que l'algorithme FCM est implanté sur MATLAB, nous avons préféré de le coder en C++. Les raisons pour lesquelles nous avons opté pour ce choix sont :

- La fonction FCM implantée sur MATLAB est utilisée comme une boîte noire, ce qui nous contraint à l'utiliser sans pouvoir changer ses paramètres par défaut ou essayer d'optimiser l'une de ses étapes;
- Par souci d'interfaçage, nous avons jugé plus adéquat l'implantation de cet algorithme en C++ vu que tous nos autres programmes ont été implantés en C++.

Théoriquement l'algorithme FCM converge vers un minimum local de $J(u, v)$ à cause de sa descente par la technique du gradient. De ce fait, la convergence de l'algorithme est

fortement liée au choix des initialisations. Plus le point sélectionné est proche du minimum local, plus la convergence est rapide [Bez81].

Pratiquement, l'implantation de l'algorithme nécessite en moyenne aussi bien le parcours des vecteurs de données que les vecteurs centres choisis initialement de façon arbitraire. La complexité de l'algorithme est alors de l'ordre de $O(nc)$.

5.3 Défuzzification de la classification

Les niveaux d'appartenance d'un document aux différentes classes déterminent le degré par lequel ils peuvent être attribués comme membres à une classe donnée. La matrice des degrés d'appartenance résultat de l'application de l'algorithme FCM sur la matrice U_0 est projetée sur chaque dimension (document).

Il existe, en fait, deux méthodes pour attribuer un élément donné à une classe. Bien qu'apparemment elles sont différentes, elles aboutissent généralement au même résultat :

- La première, (**méthode du plus proche voisin**), consiste à calculer la distance entre l'élément et le centre de chaque classe. L'élément sera alors attribué à la classe ayant la distance minimale entre son centre et l'élément en question.

$$\| d_i - v_j \| = Min_{k=1..c} \| d_i - v_k \|$$

- La deuxième, (**le maximum des membres**), consiste à attribuer l'élément à la classe

ayant la plus grande valeur de degré d'appartenance.

$$\mu(d_p) = max \left[\frac{1}{\sum_{j=1}^{c} (\frac{d_{ik}}{d_{jk}})^{\frac{2}{m-1}}} \mid d = (d_1, ..., d_{p-1}, d_p, d_{p+1}, ..., d_m) \right]$$

Dans notre cas, nous avons opté pour la deuxième méthode vu qu'elle est plus simple et nécessite moins de calcul. En appliquant l'opération Max sur chaque ensemble flou, nous obtenons la plus grande valeur des degrés d'appartenance, les index de cette valeur indiquent la classe à laquelle appartient le document en question [KG85].

On va montrer, tout au long de cette partie, la démarche entreprise lors de la classification automatique des documents de la base.

L'exemple retenu pour illustrer cette démarche est celui traité le long de ce mémoire.

Les entrées de l'algorithme FCM sont constituées par la matrice U_0, le nombre de classes et le poids exponentiel m (la valeur de ce poids est égale à 2 par défaut).

L'application du programmme FCM sur la matrice U_0 nous a conduit aux résultats suivants :

$$U = \begin{bmatrix} 0.9865 & 0.9840 & 0.0179 & 0.0044 & 0.0096 \\ 0.0135 & 0.0160 & 0.9821 & 0.9956 & 0.9904 \end{bmatrix}$$

La défuzzification de l'étape de classification nous permet de distribuer les documents

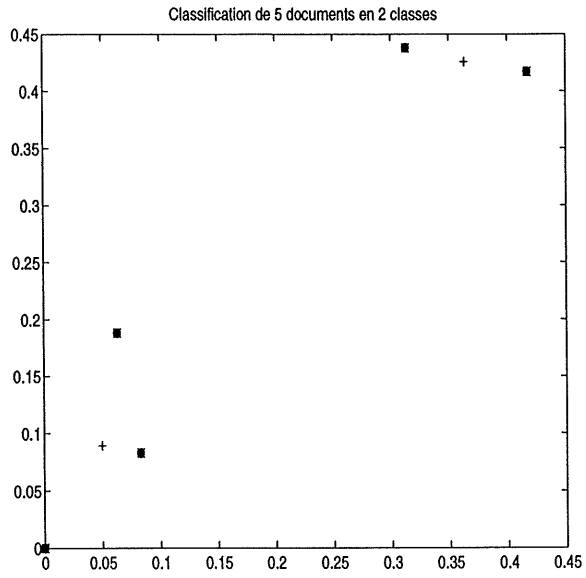


Figure 5.1 - *Classification de 5 documents par l'algorithme FCM.*

aux classes correspondantes [figure 5.1]. Ainsi, on obtient les deux classes suivantes :

$$C_1 = \begin{bmatrix} d_1 & d_2 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} d_3 & d_4 & d_5 \end{bmatrix}$$

5.4 Maintenance d'une base documentaire dynamique

La plupart des algorithmes de classification ne s'adaptent pas à des environnements dynamiques en perpétuels changements [Rij79]. Or, dans un contexte de recherche ou de classification documentaire, la suppression ou l'ajout de nouveaux documents est un phénomène très fréquent surtout sur le réseau Internet. La raison pour laquelle ces algorithmes ne peuvent pas supporter l'expansion de la base, c'est que la majorité d'entre eux se basent sur des heuristiques. Il serait alors plus pratique, dans ce cas, d'envisager la reclassification de la base entière que de penser à faire la maintenance par des algorithmes plus complexes, bien que c'est une opération coûteuse en temps et en place mémoire.

L'algorithme FCM offre, dans le cadre qui nous intéresse, l'énorme intérêt d'avoir un fonctionnement itératif. En effet, la variation quasi permanente de la source des documents et la versatilité de l'utilisateur réclament une méthode adaptative ne nécessitant pas une remise à zéro totale du système.

L'opération de la maintenance sur les classes commence par m_1 documents déjà existants que nous appelerons "anciens documents". Ces documents ont été déjà classifiés par notre algorithme basé sur les coefficients de couverture et les distributions floues, déjà détaillé précédemment.

Nous allons, dans ce qui suit, suivre pas à pas le déroulement de notre algorithme à l'aide d'un exemple d'une petite base.

5.4.1 Principe de l'algorithme de maintenance

Ayant déjà appliqué l'algorithme de pré-classification sur la matrice initiale D_{m1}, matrice des anciens documents, nous obtenons alors la matrice CC correspondante et, par conséquent, un estimatif n_{cm1} du nombre des classes. Outre la matrice des degrés d'appartenance, résultat de l'application de l'algorithme FCM sur D_{m1}, nous obtenons aussi une matrice correspondant aux centres des différentes classes qui constituent, en fait, les documents pivots ou les représentants de chaque classe.

Ayant déjà calculé au préalable, le nombre de classes de la nouvelle base, composée par les anciens et les nouveaux documents, nous pouvons associer la matrice des centres des "anciens documents" avec la matrice des "nouveaux documents" rajoutés dans la base. Cette nouvelle matrice ainsi constituée sera alors soumise à l'algorithme FCM pour une nouvelle classification. La phase de défuzzification consiste à déterminer quel document, récemment introduit dans la base, est intégré dans quelle ancienne classe. Bien evidemment, il peut y avoir le cas où un ou plusieurs nouveaux documents forment une classe indépendante des autres déjà formées dans l'ancienne base.

Cet algorithme a l'avantage de réduire le nombre de vecteurs documents à soumettre à l'algorithme de classification floue FCM, puisque seules les représentants des anciennes classes seront présents dans la nouvelle base. Ceci nous fait gagner bien sûr, tant au niveau de l'espace de stockage, qu'au niveau du temps de calcul, puisque le nombre de comparaisons des distances entre vecteurs va considérablement diminuer.

$$D = \begin{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} & m2 \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} & m1 \end{bmatrix}$$

Figure 5.2 - *La matrice documents-termes D (m = 7, n = 8).*

5.4.2 Exemple

Dans ce qui suit, nous allons illustrer par un exemple, le mécanisme de la maintenance d'une base documentaire composée par quatre documents indexés par six termes. Cette base est représentée par une matrice qu'on appellera D_{m1}.

On se propose alors de rajouter à cette base trois documents indexés par huit termes. On traitera les documents additionnels par la matrice D_{m2}. L'ensemble des documents représenté par la matrice D sera alors entièrement formé par sept documents indexés par huit termes [figure 5.2].

Le nombre de classes n_{cm1} relatif à la base D_{m1} composée des "anciens documents" est supposé être déjà calculé en faisant la somme des coefficients de découplage δ_i. Ces derniers sont considérés comme des entrées à la diagonale de la matrice des coefficients de couverture CC_{m1} [figure 5.3].

$$CC_{m1} = \begin{array}{c} \begin{array}{cccc} \text{d4} & \text{d5} & \text{d6} & \text{d7} \end{array} \\ \left[\begin{array}{cccc} 0.750 & 0.125 & 0.125 & 0.000 \\ 0.250 & 0.500 & 0.000 & 0.250 \\ 0.500 & 0.000 & 0.500 & 0.000 \\ 0.000 & 0.250 & 0.000 & 0.750 \end{array}\right] \end{array} \begin{array}{c} \\ \text{d4} \\ \text{d5} \\ \text{d6} \\ \text{d7} \end{array}$$

Figure 5.3 - *La matrice documents-documents* CC_{m1}.

$$MC_{m1} = \begin{bmatrix} 0.9991 & 0.9944 & 0.9944 & 0.0000 & 0.9944 & 0.0008 & 0.0000 & 0.0055 \\ 0.4595 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.5404 & 0.0000 & 1.0000 \\ 0.0103 & 0.0000 & 0.9845 & 0.0000 & 0.0000 & 0.0051 & 0.0000 & 0.0155 \end{bmatrix}$$

Figure 5.4 - *La matrice des centres des classes* MC_{m1}.

$$n_{cm1} = \sum_{i=4}^{7} \delta_i = \delta \times m_1 = 2.50 \approx 3.$$

En ayant déjà appliqué l'algorithme FCM sur l'ancienne base, nous obtenons, mise à part, la matrice des degrés d'appartenance U_{m1}, la matrice des centres de chaque classe MC_{m1}. Cette dernière aura les valeurs suivantes [figure 5.4] :

Le résultat de la classification des "anciens documents" est présenté alors comme suit :

$$C_1 = [1]$$

$$C_2 = [2,\ 4]$$

$$CC_{m2} = \begin{bmatrix} 0.4444 & 0.4444 & 0.1111 \\ 0.3333 & 0.5833 & 0.0833 \\ 0.1111 & 0.1111 & 0.7777 \end{bmatrix}$$

Figure 5.5 - *La matrice documents-documents CC_{m2}.*

$$C_3 = [3]$$

Il faut noter que le rang 1 de la classe C_1 correspond au document d_4 de la nouvelle base documentaire D après l'ajout des trois documents d_1, d_2 et d_3. La classe C_2 est formée par les rangs 2 et 4 qui correspondent aux documents d_5 et d_7 de cette même base. Quant à la classe C_3, elle est formée par le rang 3, qui correspond au document d_6.

Toutes ces étapes sont fournies comme des résultats lors de la classification de l'ancienne base. Elles ne constituent pas un traitement particulier qui sera inclus dans l'algorithme de maintenance.

Par contre, les nouveaux documents rajoutés à la base nécessitent la compilation de la matrice des coefficients de couverture CC_{m2} et la détermination du nombre de classes qui leur est associé [figure 5.5].

$$n_{cm2} = \sum_{i=1}^{3} \delta_i = \delta \times m_2 = 1.805 \approx 2.$$

L'étape suivante consiste à déterminer la matrice des centres des classes associée à ces nouveaux documents MC_{m2} [figure 5.6].

$$MC_{m2} = \begin{bmatrix} 1.0000 & 0.0011 & 0.0000 & 0.0003 & 0.0011 & 0.0000 & 0.9988 & 0.9988 \\ 1.0000 & 1.0000 & 0.0000 & 0.5038 & 1.0000 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$$

Figure 5.6 - *La matrice des centres des classes MC_{m2}.*

$$MC_{D} = \begin{bmatrix} 1.0000 & 0.0011 & 0.0000 & 0.0003 & 0.0011 & 0.0000 & 0.9988 & 0.9988 \\ 1.0000 & 1.0000 & 0.0000 & 0.5038 & 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.9991 & 0.9944 & 0,9944 & 0.0000 & 0.9944 & 0.0008 & 0.0000 & 0.0055 \\ 0.4595 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.5404 & 0.0000 & 1.0000 \\ 0.0103 & 0.0000 & 0.9845 & 0.0000 & 0.0000 & 0.0051 & 0.0000 & 0.0155 \end{bmatrix}$$

Figure 5.7 - *La matrice des centres des classes MC_D.*

Le résultat de la classification de la nouvelle base est présenté alors comme suit :

$$C_1 = [3]$$

$$C_2 = [1,\ 2]$$

Il faut noter que le rang 3 de la classe C_1 correspond au document d_3 de la nouvelle base documentaire D. La classe C_2 est formée par les rangs 1 et 2 qui correspondent aux documents d_1 et d_2 de cette même base.

La combinaison des matrices centres MC_{m1} et MC_{m2} formera la matrice des représentants de la base entière MC_D [figure 5.7].

$$CC_D = \begin{array}{c} \begin{array}{ccccccc} d1 & d2 & d3 & d4 & d5 & d6 & d7 \end{array} \\ \begin{bmatrix} 0.29 & 0.29 & 0.07 & 0.29 & 0.07 & 0.00 & 0.00 \\ 0.22 & 0.47 & 0.05 & 0.22 & 0.05 & 0.00 & 0.00 \\ 0.07 & 0.07 & 0.51 & 0.07 & 0.18 & 0.00 & 0.11 \\ 0.22 & 0.22 & 0.05 & 0.34 & 0.05 & 0.13 & 0.00 \\ 0.10 & 0.10 & 0.27 & 0.10 & 0.27 & 0.00 & 0.17 \\ 0.00 & 0.00 & 0.00 & 0.50 & 0.00 & 0.50 & 0.00 \\ 0.00 & 0.00 & 0.17 & 0.00 & 0.17 & 0.00 & 0.67 \end{bmatrix} \end{array}$$

Figure 5.8 - *La matrice documents-documents CC_D.*

Afin de déterminer le nombre de classes n_{cm} associé à la nouvelle base, il sera nécessaire de construire la matrice des coefficients de couverture CC_D s'y afférant [figure 5.8].

$$n_{cm} = \sum_{i=1}^{7} \delta_i = \delta \times m = 3.05 \approx 3.$$

Ayant en main, la matrice des représentants des classes MC_D ainsi que le nombre de classes n_{cm} de la base D, nous pouvons alors soumettre ces paramètres à l'algorithme FCM afin de déterminer les nouvelles classes relatives à la nouvelle base. Les résultats obtenus sont comme suit :

$$C_1 = [1,\ 4]$$

$$C_2 = [2,\ 3]$$

$$C_3 = [5]$$

Défuzzification :

Pour donner un sens à cette classification, il s'agit de faire la liaison entre les représentants des classes et les documents qui les constituent. Par exemple, la classe C_1 formée par le couple [1, 4] est relative aux documents de la première classe de la nouvelle base, à savoir d_3, ainsi qu'aux documents de la deuxième classe de l'ancienne base, à savoir d_5 et d_7.

La deuxième classe formée par le couple [2, 3] est relative aux documents de la deuxième classe de la nouvelle base, à savoir d_1 et d_2. Elle est aussi relative à la première classe de l'ancienne base composée du document d_4.

La troisième et la dernière classe composée du singleton 5 est relative aux documents de la troisième classe de l'ancienne base, à savoir d_6.

5.4.3 Validation de l'algorithme

Pour valider notre algorithme, on se propose de reclassifier la base entière formée par 7 documents et indexée par 8 termes. Nous rappelons que cette base est en fait, la combinaison d'anciens et de nouveaux documents et que le vocabulaire d'indexation a été enrichi par 2 termes. La base D subira alors le même processus que celui appliqué lors d'une première classification. Il s'agit alors de suivre les mêmes étapes que celles détaillées dans le chapitre 4.

La matrice CC_D, matrice des coefficients de couverture, dérivée à partir de la matrice correspondante à la base D aura alors les valeurs présentées ci-dessus [figure 5.8, page 72].

$$n_{cm} = \sum_{i=1}^{7} \delta_i = \delta \times m = 3.05 \approx 3.$$

La matrice des entrées E_0, matrice dérivée de CC_D et relative aux entrées de l'algorithme FCM est la suivante :

$$E_0 = \begin{bmatrix} 0.29 & 0.29 & 0.07 \\ 0.47 & 0.22 & 0.07 \\ 0.07 & 0.07 & 0.51 \\ 0.22 & 0.34 & 0.07 \\ 0.10 & 0.10 & 0.27 \\ 0.00 & 0.50 & 0.00 \\ 0.00 & 0.00 & 0.17 \end{bmatrix}$$

L'application de l'algortihme FCM sur E_0 donnera la classification suivante :

$$C_1 = \begin{bmatrix} d_3 & d_5 & d_7 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} d_1 & d_2 & d_4 \end{bmatrix}$$

$$C_3 = \begin{bmatrix} d_6 \end{bmatrix}$$

Notons que nous obtenons la même classification que celle obtenue avec l'algorithme de maintenance de la base dynamique, ce qui confirme la validité de notre algorithme.

5.5 Implémentation de l'algorithme de Can et Ozkarahan

Afin d'évaluer la qualité de la classification obtenue par application de l'algorithme FCM sur une base documentaire, on a opté pour sa comparaison avec un autre algorithme de classification non floue.

On a d'abord commencé par étudier les publications qui ont porté sur la classification documentaire et on a retenu deux articles, [CO90, CO83], d'un même auteur, qui offraient un algorithme détaillé de classification binaire, c'est alors qu'on a décidé de l'implémenter et de comparer les résultats obtenus des deux algorithmes.

L'algorithme se base sur le concept de coefficient de couverture. Il assure, entre autres, une stratégie de partition qui repose sur la sélection d'un certain nombre de documents considérés comme des documents pivots, et assigne par la suite les documents ordinaires aux classes initialisées par ces documents pivots afin de former les classes entières. Le concept de coefficient de couverture est la base de cette stratégie dans la mesure où il sert à :

1. identifier les relations entre les documents de la base, en utilisant la matrice CC définie précédemment dans le chapitre 4, la section 4.2;
2. déterminer le nombre de classes de la base qui en résulte, se référer à la section 4.4;
3. sélectionner les documents pivots, en utilisant le concept de calcul de poids, section 4.5;
4. former les classes en se basant sur l'algorithme décrit ci-dessous.

5.5.1 Algorithme de Can et Ozkarahan

AGRÉGATION-CLASSES()

```
1  [a] Déterminer les documents pivots de la base;
2  [b] i = 1 ;
3  while i < m
4      do /*m étant le nombre de documents*/
5          /*construction des classes*/
6          if d_i n'est pas un document pivot
7            then trouver le document pivot qui couvre au maximum d_i;
8                 s'il existe plus qu'un document pivot qui satisfasse cette condition alors
9                 assigner d_i à la classe qui a le plus fort poids parmi les candidats;
10         i ← i + 1;
11
12 [c] s'il reste des documents qui ne sont pas affectés,
13 on les regroupe sous une même classe.
```

Il faut noter que l'algorithme ci-dessus essaie de concentrer les documents ordinaires autour d'un document pivot. Il y a des fois où ces documents ne peuvent pas être associés à un document pivot. La solution, comme le montre l'étape (c), est d'insérer ces documents dans un même "sac", c'est-à-dire dans une même classe.

Chapitre 6

Résultats expérimentaux

Tous les algorithmes de classification automatique peuvent à leur tour être classifiés selon le nombre d'itérations qu'ils nécessitent pour accomplir l'agrégation et par conséquent, ils se réfèrent à des algorithmes itératifs ou à une seule passe.

Les propriétés suivantes sont importantes pour déterminer la validité d'un algorithme de classification [CO89] :

1. la composition des classes est indépendante de l'ordre dans lequel les documents ont été traités.

2. les algorithmes de classification devront être capables de supporter l'expansion ou le retrait de certains documents efficacement. La maintenance des classes doit être pratique et efficace.

3. les classes sont stables, dans la mesure où il est peu probable qu'elles changent de composition lorsque de nouveaux documents viennent s'ajouter à la base ou des anciens sont retirés.

4. la distribution des documents dans chaque classe doit être la plus uniforme possible.

Nous allons montrer au cours de ce chapitre que ces propriétés sont bien vérifiées par notre algorithme de classification floue.

6.1 Sélection de la base documentaire

Afin de valider notre travail, nous avons mené nos expériences sur une base de 100 documents indexés par 425 termes. Ces documents ont été choisis arbitrairement de la revue "Communications of the ACM" s'étalant sur la période du mois d'Avril 1978, volume 21, jusqu'au numéro du mois de Janvier 1983, volume 26. Le choix de la revue s'est basé sur le fait que ses articles sont préalablement classifiés sous des rubriques générales faisant allusion au sujet que traite l'article. Ceci facilitera le calcul du taux d'erreur de la classification automatique par rapport à la classification réelle. De plus, chaque article présente une série de mots clés l'identifiant. L'indexation de chaque article est déjà fournie dans leurs entêtes. Notre travail s'est résumé alors à rassembler tout le vocabulaire d'indexation dans un fichier puis à attribuer chaque mot à l'article lui correspondant. Une description complète, à l'aide des mots clés des documents ainsi que leurs domaines d'intérêt sont donnés en annexe.

6.2 Test sur la composition des classes

Rappelons que pour représenter nos connaissances, nous avons fait appel au modèle vectoriel. Les documents identifiés par leurs mots clés forment une matrice documents-termes. Les entrées de cette matrice indiquent la présence ou l'absence du terme faisant partie du vocabulaire d'indexation dans le document.

Au point de vue mathématique, ces documents forment des vecteurs dans un espace multidimentionnel. La dimension du vecteur est en fait la dimension du vocabulaire d'indexation. Le choix des termes est fait de manière à ce qu'il n'y ait pas de dépendances sémantiques entre eux. Les documents représentent alors des vecteurs libres donc indépendants.

En pratique, le théorème d'indépendance entre les vecteurs libres se confirme par le fait que quelle que soit la position du document dans la matrice, la répartition des documents en agrégats est toujours la même. Ce test a été effectué sur la base de 100 documents et a montré que quelle que soit la disposition du document dans la base ce qui veut dire que, quelle que soit la position du vecteur dans la matrice, la décomposition de l'ensemble de la base est toujours la même.

6.3 Test sur la maintenance et la stabilité de la base

Comme il a été mentionné dans le chapitre précédent, notre approche permet facilement l'ajout aussi bien de termes que de documents. En effet après avoir compilé et testé notre algorithme sur les 100 documents tirés de la revue "Communications of the ACM", nous

avons voulu enrichir cette base par 15 documents additionnels indexés par 55 termes. Il existe parmi ces derniers 25 nouveaux termes. La base s'est alors élargie, et s'est rendue à 115 documents indexés par 450 termes.

En suivant les étapes de l'algorithme de maintenance d'une base dynamique, il s'agit de considérer uniquement les 15 documents récemment ajoutés puisque tous les traitements ont été déjà effectués sur les 100 anciens documents. Tout le travail se résume donc à chercher les centres des nouvelles classes et de les combiner avec ceux des anciens. L'étape suivante consiste alors à appliquer l'algorithme FCM puis défuzzifier les résultats obtenus non pas sur les documents proprement dit mais sur leurs représentants. Bien évidemment, ceci va contribuer sensiblement à la diminution du temps de calcul et à l'espace de stockage puisque seuls les vecteurs centres sont appelés lors de l'opération d'agrégation. Lors des tests effectués, nous avons remarqué que le temps de calcul diminue proportionnellement au nombre de vecteurs centres de la nouvelle base récemment modifiée. Nous nous sommes rendus compte de cette observation en comparant le temps de calcul nécesaire pour la reclassification de la base entière par rapport au temps de calcul requis pour l'exécution de l'algorithme de maintenance.

$$t_e \approx \frac{t_{reclassification}}{nombre - vecteurs - centres}$$

Il faut noter que t_e correspond au temps d'exécution de l'algorithme de maintenance de la base dynamique et $t_{reclassification}$ correspond au temps d'exécution nécessaire si nous optons pour la reclassification de la base entière.

Au point de vue stabilité de la base, vu que les nouveaux documents couvrent dans la majorité des cas la matière evoquée par les anciens documents, ces derniers ont été distri-

bués sur les différentes classes déjà existantes selon leurs domaines d'intérêt. Trois parmi ces nouveaux documents ont été attribués à une sorte de classe fourre tout "rag bag". Ceci peut s'expliquer par le fait que ces documents contiennent la plupart des termes non existants dans les autres documents.

6.4 Test sur l'uniformité de la distribution des documents entre les classes

L'hypothèse de l'uniformité dans une base documentaire est une notion relative. Selon les tests, la concentration de documents par classe varie sensiblement en fonction de la sémantique des documents, elle dépend en fait, dans le cas de la classification documentaire, du nombre de termes en commun entre les documents.

Plus le nombre de termes syntaxiquement identiques est élevé dans les documents, plus ces derniers ont de chance à être rassemblés dans une même classe. L'inconvénient majeur de la classification basée sur les termes d'indexation est qu'elle ne tient pas compte de la sémantique des termes. En effet, deux documents peuvent s'avérer très semblables même s'ils ne partagent aucun terme en commun. De ce fait, nous avons tenté l'expérience de remplacer les termes syntaxiquement différents mais que nous jugeons sémantiquement équivalents par un seul terme standard qui sera partagé par tous les documents contenant son synonyme. Le résultat de l'expérience a complètement changé et le nombre de classes a considérablement diminué. Il est passé de 39 classes à 20 classes. Par conséquence nous observerons un changement au niveau de la distribution des documents entre les classes comme, par exemple, la fusion de

certaines classes. Dans le cas où nous avons affaire à une base documentaire très diversifiée dans la mesure où elle rassemble des documents de divers domaines non nécessairement liés, nous pouvons observer dans ce cas un grand nombre de classes de petites tailles.

La conclusion que nous pouvons tirer de cette expérience, c'est qu'il est fortement conseillé d'avoir recours à un thesaurus pour établir les liens entre les termes d'indexation. Ceci aura une grande influence sur le résultat de la classification et par conséquent, sur la diminution du taux d'erreur de la classification automatique par rapport à la classification réelle.

6.5 Test sur le taux d'erreur

L'un des paramètres tests utilisé pour s'assurer de la validité d'un algorithme de classification est de calculer son taux d'erreur par rapport à la classification réelle. Le taux d'erreur est calculé en comptant le nombre de documents mal classifiés par rapport au nombre total de documents.

Les tests effectués sur la base de 100 documents déjà classifiés ont permis de constater et d'interpréter l'écart entre la classification automatique et la classification réelle de la revue "Communications of the ACM". Nous présentons ci-dessus, un tableau comparatif des résultats obtenus lors de la classification automatique avec ceux de la classification réelle [tableau 6.1].

Il faut noter que n_d correspond au nombre de documents, n_t le nombre de termes, n_{cr} le nombre de classes réelles, n_{ce} le nombre de classes effectives, ter_{nc} le taux d'erreur sur le

	n_d	n_t	n_{cr}	n_{ce}	ter_{nc}	ter_{cl}
Classification réelle	100	425	8	-	0%	0%
Classification automatique avec thesaurus	100	297	8	20	25%	11%
Classification automatique sans thesaurus	100	425	8	30	37.5%	13%

TABLEAU 6.1 - *Tableau comparatif des résultats de la classification automatique avec ceux de la classification réelle.*

nombre de classes et finalement ter_{cl} le taux d'erreur sur la classification.

L'examen des résultats obtenus permet de constater l'importance d'avoir un thesaurus associé à l'algorithme de classification. Un tel outil, bien qu'il ait été construit manuellement par un non-linguiste, a permis d'établir des liens d'équivalence entre les termes. Ce qui nous a conduit à de meilleurs résultats plus proches de la classification réelle.

Le second point qu'il faut préciser, c'est que pour le cas de la classification sans thesaurus, le taux d'erreur calculé au niveau des regroupements des documents dans une classe (13%) est comparable à celui de la classification avec thesaurus (11%) sauf qu'il faut souligner la présence d'un grand nombre de classes avec une petite concentration de documents. Nous en concluons alors que les classes réelles ont été éclatées en petites sous-classes comportant des termes identiques en commun.

6.6 Comparaison et interprétation des résultats

Il s'agit dans cette expérience de comparer les résultats de l'algorithme de Can et Ozkarahan, déjà présenté dans le chapitre précédent avec ceux de la méthode Pro-Floue. Le critère de comparaison étant par rapport à la classification réelle. Le résultat de l'application des deux algorithmes sur la même base a donné deux résultats totalement différents. On s'est basé alors sur la description des contenus des documents ainsi que leurs domaines d'intérêt pour juger la qualité de la classification.

L'application de l'algorithme de Can et Ozkarahan, sur cette base a donné la classification suivante :

$$
\begin{array}{l}
C_1 = \left[\begin{array}{cc} d_5 & d_{34} \end{array}\right], C_2 = \left[\begin{array}{cc} d_{52} & d_{53} \end{array}\right], C_3 = \left[\begin{array}{cc} d_{33} & d_{42} \end{array}\right], \\
C_3 = \left[\begin{array}{ccc} d_{72} & d_{85} & d_{100} \end{array}\right], C_4 = \left[\begin{array}{cc} d_{60} & d_{73} \end{array}\right], C_5 = \left[\begin{array}{ccc} d_{75} & d_{91} & d_{36} \end{array}\right], \\
C_6 = \left[\begin{array}{ccc} d_{14} & d_{20} & d_{26} \end{array}\right], C_7 = \left[\begin{array}{ccc} d_{55} & d_{90} & d_{71} \end{array}\right], C_8 = \left[\begin{array}{ccc} d_{55} & d_{90} & d_{71} \end{array}\right], \\
C_9 = \left[\begin{array}{cccc} d_{27} & d_{95} & d_{93} & d_{92} \end{array}\right], C_{10} = \left[\begin{array}{ccc} d_{38} & d_{57} & d_{69} \end{array}\right], C_{11} = \left[\begin{array}{cc} d_{58} & d_{59} \end{array}\right], \\
C_{12} = \left[\begin{array}{cc} d_{45} & d_{78} \end{array}\right], C_{13} = \left[\begin{array}{cccc} d_7 & d_{82} & d_{40} & d_{86} \end{array}\right], C_{14} = \left[\begin{array}{ccc} d_4 & d_{56} & d_{32} \end{array}\right], C_{15} = \left[\begin{array}{cc} d_8 & d_{28} \end{array}\right] \\
C_{16} = \left[\begin{array}{ccccccccccccccccc}
d_1 & d_2 & d_3 & d_6 & d_9 & d_{10} & d_{11} & d_{12} & d_{13} & d_{15} & d_{16} & d_{17} & d_{18} & d_{19} & d_{21} & d_{22} & d_{23} \\
d_{24} & d_{25} & d_{29} & d_{30} & d_{31} & d_{35} & d_{37} & d_{39} & d_{41} & d_{43} & d_{44} & d_{46} & d_{47} & d_{48} & d_{49} & d_{50} & d_{51} \\
d_{54} & d_{56} & d_{57} & d_{61} & d_{62} & d_{63} & d_{64} & d_{65} & d_{66} & d_{67} & d_{68} & d_{70} & d_{74} & d_{76} & d_{77} & d_{78} & d_{79} \\
d_{80} & d_{81} & d_{83} & d_{84} & d_{87} & d_{88} & d_{89} & d_{94} & d_{96} & d_{97} & d_{98} & d_{99} & & & & &
\end{array}\right]
\end{array}
$$

Cette même base, a été soumise aussi à l'algorithme FCM, qui nous a donné la classification suivante :

$$C_1 = \left[d_{95} \right], C_2 = \left[d_{10} \quad d_{25} \quad d_{59} \right],$$
$$C_3 = \left[d_9 \quad d_{11} \quad d_{13} \quad d_{32} \quad d_{40} \quad d_{43} \quad d_{78} \quad d_{79} \quad d_{80} \quad d_{81} \quad d_{82} \quad d_{86} \quad d_{96} \quad d_{97} \right],$$
$$C_4 = \left[d_{16} \quad d_{21} \quad d_{22} \quad d_{24} \quad d_{26} \quad d_{67} \quad d_{70} \quad d_{71} \quad d_{84} \quad d_{90} \quad d_{94} \right],$$
$$C_5 = \left[d_{44} \quad d_{45} \quad d_{47} \quad d_{48} \right], C_6 = \left[d_7 \right],$$
$$C_7 = \left[d_{34} \quad d_{35} \quad d_{65} \quad d_{68} \quad d_{73} \quad d_{76} \quad d_{77} \right], C_8 = \left[d_{89} \right], C_9 = \left[d_1 \quad d_{37} \quad d_{87} \right],$$
$$C_{10} = \left[d_3 \quad d_{12} \quad d_{19} \quad d_{23} \quad d_{38} \quad d_{39} \quad d_{41} \quad d_{56} \quad d_{58} \quad d_{63} \quad d_{72} \quad d_{85} \quad d_{99} \quad d_{100} \right],$$
$$C_{11} = \left[d_{64} \right], C_{12} = \left[d_2 \quad d_{14} \quad d_{15} \quad d_{20} \right], C_{13} = \left[d_{46} \quad d_{49} \quad d_{50} \quad d_{51} \quad d_{52} \quad d_{53} \right],$$
$$C_{14} = \left[d_{91} \right], C_{15} = \left[d_4 \quad d_{18} \quad d_{57} \quad d_{60} \quad d_{69} \right], C_{16} = \left[d_{27} \quad d_{31} \quad d_{54} \quad d_{93} \right],$$
$$C_{17} = \left[d_{17} \quad d_{29} \quad d_{30} \quad d_{55} \quad d_{62} \quad d_{88} \right], C_{18} = \left[d_{28} \right],$$
$$C_{19} = \left[d_5 \quad d_6 \quad d_{33} \quad d_{36} \quad d_{42} \quad d_{66} \quad d_{74} \quad d_{75} \quad d_{98} \right], C_{20} = \left[d_8 \quad d_{61} \quad d_{83} \quad d_{92} \right]$$

En examinant la classification donnée par les deux algorithmes, on constate, dans le cas de l'algorithme de Can et Ozkarahan, que la distribution des documents au sein des classes est faible (en moyenne 3 à 4 documents par classe) et que seule une classe échappe à cette règle. Il s'agit de ce qu'on appelle la classe "fourre-tout" où 63% des documents y sont inclus. Ceci est très pénalisant dans la mesure où la classe "fourre-tout" est retenue utile lorsque, dans le cas où on ne réussit pas à attribuer un document à la bonne classe, on le place dans une sorte de classe additionnelle, qui théoriquement doit avoir une faible distribution. Or, dans notre cas, plus de la moitié des documents y font partie. Ceci peut s'interpréter de la façon suivante : lorsque les documents ne partagent aucun ou très peu de termes en commun, seul le calcul des coefficients de couplage et de découplage n'assurera pas la bonne agrégation.

Contrairement aux résultats obtenus par l'algorithme de Can et Ozkarahan, les classes construites par l'approche Pro-Floue montrent une certaine uniformité au niveau de la distribution des documents entre les classes. En se référant à la classification réelle, on a pu

établir une comparaison au niveau des contenus des classes [tableau 6.1]. Le taux d'erreur s'est révélé très acceptable. En effet, grâce à son fonctionnement itératif, l'algorithme FCM permet de réduire l'erreur à chaque itération en calculant le regroupement le plus efficace.

La conclusion avec laquelle on peut sortir après l'étude de ces résultats est le fait que la classification effectuée par l'algorithme FCM est assez proche de la réalité. Bien qu'elle ne soit pas parfaite et tolère une certaine marge d'erreur, notre approche est beaucoup plus réaliste que la classification donnée par l'algorithme de Can et Ozkarahan.

Chapitre 7

Conclusion

Ce mémoire a présenté un système de classification documentaire utilisant une approche combinant la probabilité et le flou. Rappelons aussi que ce travail constitue une partie intégrante d'un grand projet de recherche et de filtrage d'information multilingue.

7.1 Bilan du travail

L'objectif principal du système CLASSFLOU était d'aboutir à partir d'une base documentaire assez importante à une classification agglomérative qui s'approche le plus possible de la classification réelle. Suite à un lancement d'une requête de la part de l'usager, l'ensemble des documents résultats de la recherche et du filtrage est soumis au processus de classification. Cela vise à répondre de la manière la plus efficace et rapide possible au besoin

d'information de l'usager.

Le système CLASSFLOU a réussi à répondre à ces exigences en obtenant un taux d'erreur respectable de 13% représentant les documents mal classifiés. La comparaison des résultats de notre système avec ceux de Can et Ozkarahan nous a permis de confirmer avec rigueur l'amélioration de la qualité de la classification.

Un autre point tout aussi important est celui de l'extension de notre système afin qu'il puisse traiter le cas de la classification d'une base dynamique, spécification qui a été respectée, puisque les documents manipulés sur Internet sont sujets à des mises à jour assez fréquentes.

7.1.1 Amélioration de la qualité de la classification

L'amélioration de la qualité de la classification a été atteinte grâce :

- au bon choix de la représentation vectorielle des correlations probabilistes entre documents;
- à la mise en correpondance itérative floue dans le but de décomposer la base entière en regroupements cohérents.

En effet, la méthode de classification floue utilisée réalise l'étape de mise en correspondance de manière itérative puisque l'appariement est remis en cause à chaque étape. Un critère explicite du plus proche voisin convient bien pour décider de l'appartenance d'un document à une classe.

La comparaison des résultats escomptés avec ceux d'une classification réelle établie par la revue "Communications of the ACM" nous a permis de conclure que, malgré que notre approche n'est pas parfaite et tolère une certaine marge d'erreur, elle reste néanmoins nettement plus efficace si on la compare avec les résultats de l'algorithme de Can et Ozkarahan.

7.1.2 Extension du système pour la classification d'une base dynamique

L'avantage de l'utilisation d'un algorithme de classification floue par rapport aux autres techniques de classification "hard" ou binaires est l'information inhérente que portent les degrés d'appartenance sur le jugement d'attribuer une donnée à une classe.

Nous avons exploité le fait que l'algorithme FCM retourne en sortie l'ensemble des vecteurs centres des classes constituées. Ces derniers sont considérés comme éléments pivots représentatifs de chaque classe. Ils sont intégrés en cas de mise à jour de la base pour représenter les regroupements obtenus lors de la classification de l'ancienne base. Ainsi, de cette manière, nous pourrons éviter la reclassification complète de la base.

Bien évidemment, ceci va contribuer sensiblement à la diminution du temps de calcul et l'espace de stockage puisque seuls les vecteurs centres de l'ancienne base sont appelés lors de l'opération d'agrégation.

7.2 Travaux futurs

Au cours des tests expérimentaux effectués sur la base de 100 documents, nous avons souligné l'importance de l'introduction d'un thesaurus dans notre système. Des travaux futurs suivant cet axe pourront raffiner davantage le système. En effet, le recours à un thesaurus constitue un complément particulièrement efficace. Les résultats de classification que nous avons obtenus sont largement en faveur de cette solution sur deux points essentiels :

- d'une part, dans la classification non floue où un objet doit forcément appartenir à une classe et une seule, nous constatons l'apparition d'une classe "fourre-tout" assez importante;
- d'autre part, dans la classification floue, nous observons un taux d'erreur de classification supérieur à celui d'un système flou utilisant un thesaurus.

À l'heure actuelle, l'assemblage de tous les outils formant le système de recherche et de filtrage d'information multilingue est partiel. En effet, chaque composante du projet a été testée à part. Il serait alors judicieux de pouvoir intégrer les différentes parties et de faire les tests nécessaires afin de constater l'impact de la réunion de tous ces outils sur l'efficacité et la rapidité de la recherche de l'information pertinente.

BIBLIOGRAPHIE

[And73] M. R. Anderberg. *Cluster analysis for applications.* Academic Press, New York, 1973.

[Ant88] G. Antoniadis. French text recognition model for information retrieval systems. In Y. Chiaramella, editor, *Proceedings of the 11th international conference on research and development in information retrieval.*, Grenoble, France, June 8-10 1988. Presses Universitaires.

[BC87] N. J. Belkin and W. B. Croft. Retrieval techniques. *Annual review of information science and technology*, (22):109–145, 1987.

[Bez73] J. C. Bezdek. *Fuzzy mathematics in pattern classification.* PhD thesis, Thesis applied, Mathematics center, Cornell University, Ithaca, 1973.

[Bez81] J. C. Bezdek. *Pattern recognition with fuzzy objective function algorithm.* New York: Plenum, 1981.

[BP92] J. C. Bezdek and S. K. Pal. *Fuzzy models for pattern recognition.* New York: IEEE Press, 1992.

[CCS95] L. M. Chan, J. P. Comaroni, and M. Satija. *Classification décimale de Dewey: guide pratique.* Montreal: Asted, 1995.

[CK86] D. Coulon and D. Kayser. Informatique et langage naturel: présentation générale des méthodes d'interprétation des textes écrits. *Techniques et sciences informatiques*, 5(2):103–128, 1986.

[CK87] P. R. Cohen and R. Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Information processing and management*, 23(4):255–268, 1987.

[CO83] F. Can and E. A. Ozkarahan. A clustering scheme. *In Proceedings of the 6th Annual International ACM-SIGIR Conference. (Bethesda, Md.)*, pages 115–121, June 1983.

[CO85] F. Can and E. A. Ozkarahan. Concepts of the cover coefficient based clustering methodology. *In Proceedings of the 8th Annual International ACM-SIGIR Conference. (Montreal, Quebec).*, pages 204–211, June 1985.

[CO89] F. Can and E. A. Ozkarahan. Dynamic cluster maintenance. *Information processing and management*, 25(3):295–291, 1989.

[CO90] F. Can and E. A. Ozkarahan. Concepts and effectiveness of the cover-coefficient-based clustering methodology for text databases. *ACM Transactions on Database systems*, 15(4):483–517, December 1990.

[CS68] C. D. Cook and R. H. Schimmelpfeng. *The use of the library of congress classification :proceedings.* Chicago: American library association, 1968.

[CT87] W. B. Croft and R. H. Thompson. A new approach to the design of document retrieval systems. *Journal of the American society for information science*, 38(6):389–404, 1987.

[DL82] E. Diday and J. Lemaire. *Éléments d'analyse de données.* Dunod, Paris, 1982.

[EHW87] A. El-Hamdouchi and P. Willet. Techniques for the measurement of clustering tendency in document retrieval systems. *Journal of information science*, (13):361–365, 1987.

[For65] E. W. Forgy. Cluster analysis of multivariate data : efficeing version interpretability of classifications. *Biometrica*, 21(3), 1965.

[JR71] N. Jardine and C. J. Van Rijsbergen. The use of hierarchic clustering in information retrieval. *Information storage and retrieval*, 7(5):217–240, 1971.

[KG85] J. M .Keller and M. R. Gray. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(4):580–585, July/August 1985.

[LMF82] L. Lebart, A. Morineau, and J. P. Fenelon. *Traitement des données statistiques : Méthodes et programmes.* Dunod, 1982.

[MT85] B. Michelet and W. A. Turner. Co-word search : a system for information retrieval. *Journal of information science*, (11):173–181, 1985.

[Pan87] J. Panyr. Conceptual clustering and relevance feedback. *International classification*, 14(3):133–137, 1987.

[PS97] R. Pasero and P. Sabatier. *Concurrent processing for sentences analysis, synthesis and guided composition.* Lecture Notes in Computer Science, Springer, 1997.

[Rij79] C. J. Van Rijsbergen. *Information retrieval.* London : Butterworths, 1979.

[Rou] J. Rouault. *Linguistique automatique.* Applications documentaires, Berne, peter lang edition.

[Sal72] G. Salton. A new comparaison between conventional indexing and automatic text processing. *American Society for Information Science*, 23(2):75–84, 1972.

[Sal78] G. Salton. Generation and search of clustered files. *ACM Trans on Database Systems*, 3(4):321–346, 1978.

[Sal86] G. Salton. Recent trends in automatic information retrieval. In F. Rabitti, editor, *Proceedings of the ACM conference on research and development in information retrieval.*, Pisa, Italy, September 8-10 1986.

[Sal89] G. Salton. *Automatic Text Processing, The transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[SFW78] G. Salton, E. A. Fox, and H .Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(12):1022–1036, 1978.

[Sil93] M. Silberztein. *Dictionnaires électroniques et analyse automatique de textes : le système INTEX.* Paris : Masson, 1993.

[SKJ81] R. C. Schank, J. L. Kolodner, and G. D. De Jong. Conceptual information retrieval. *Information retrieval research*, pages 94–116, 1981.

[SV85] G. Salton and E. Vorhees. Automatic assignment of soft boolean operators. In *Proceedings of the eighth annual international ACM SIGIR conference on research and development in information retrieval.*, pages 54–69, New York, 1985. Association for computing machinary.

[Voo86] E. M. Voorhees. The efficiency of inverted index and cluster searches. In F. Rabitti, editor, *Proceedings of the ACM conference on research and development in information retrieval.*, pages 164–174, Pisa, Italy, September 8-10 1986.

[Wil87] P. Willet. Effectiveness of retrieval in clustered document files. In Oxford : Learned information, editor, *11th Online information*, pages 137–146, December 8-10 1987.

ANNEXE

Description des documents

Documents	Titre	Indexes	Domaine d'intérêt
d_1	An improved hash code for scatter storage	hash code, hash table, scatter storage, searching	Programming techniques
d_2	Scatter storage techniques	scatter storrage, hash addressing, searching, file addressing, storage layout	Programming techniques
d_3	The working set model for program behavior	general operating system concepts, multiprocessing, multiprogramming, operating systems, program behavior, program models, ressource allocation, scheduling, storage allocation	Operating system principles
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_4	The structure of "the" multiprogramming system	operating system, multiprogramming system, system hierarchy, system structure, real-time debugging, program verification, synchronizing primitives, cooperating sequential process, system levels, input-output buffering, multiprogramming, processor sharing, multiprocessing	Operating system principles
d_5	An axiomatic basis for computer programming	axiomatic method, theory of programming, proofs of programs, formal language definition, programming language design, machine independent programming, program documentation	Programming languages
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_6	An efficient context-free parsing algorithm	syntax analysis, parsing, context-free grammar, compilers, computational complexity	Programming languages
d_7	The quadratic quotient method: A hash code eliminating secondary clustering	hashing, hash code, scatter storage, calculated address, clustering search, symbol table, collisions, keys, table look-up	Programming techniques
d_8	A relational model of data for large shared data banks	data bank, data base, data structure, data organization, hierarchies of data, networks of data relations, derivability, redundancy consistency, composition, join, retrieval language, predicate calculus security, data integrity	Information retrieval
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_9	Program development by stepwise refinement	education in programming, programming techniques, stepwise program construction	Education
d_{10}	A technique for software module specification with examples	software, specification modules, software engineering, software design	Programming techniques
d_{11}	A statistical study of the accuracy of floating point number systems	error analysis, floating point arithmetic, rounding, guard digits, number representation	Numerical mathematics
d_{12}	The Unix time-sharing system	time-sharing, operating system, command language, pdp-11	Operating system principles
d_{13}	Ethernet: Distributed packet switching for local computer networks	computer networks, packet switching, multiprocessing, distributed control, distributed computing, broadcast communication, statistical arbitration	Computer systems
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{14}	A method for obtaining digital signatures and public-key cryptosystems	digital signatures, public-key cryptosystems, privacy authentication, security, factorization, prime number, electronic-mail, message-passing, electronic funds transfer, cryptography	Programming techniques
d_{15}	Communicating sequential processes	programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, non determinacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{16}	A critique of the foundations of style programming logics	partial correctness, proofs, logic, defined functions	Programming techniques
d_{17}	A hash code method for detecting and correcting spelling errors	hash table, search process, rotation, dictionaries, spelling, spelling errors	Programming techniques
d_{18}	On the asymptotic behavor of time-sharing systems	operating systems, time sharing systems, system hierarchy, response times	Operating system principles
d_{19}	A dynamic storage allocation technique based on memory residence time	operating systems, dynamic storage allocation, memory allocation, memory fragmentation, storage fragmentation	Operating system principles
d_{20}	Computer-assisted microanalysis of programs	analysis of algorithms, time-formulas, program correctness, symbol table, programming techniques	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{21}	Combinatorially implosive algorithms	programming techniques, search, parallelism, heuristic search	Programming techniques
d_{22}	Plane-sweep algorithms for intersecting geometric figures	intersection problems, algorithms, maps, programming techinques	Programming techniques
d_{23}	A protection model and its implementation in a dataflow system	operating systems, operating system principles, protection, interprocess communication	Operating system principles
d_{24}	Efficient parallel algorithms for some graph problems	algorithms, parallel processing, graph algorithm, optimal algorithms, programming techinques	Programming techniques
d_{25}	The solution for the branching factor of the alpha-beta pruning algorithm and its optimality	heuristic search, proofs, graph algorithms, game searching, average case analysis, programming techniques	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{26}	Heuristics for the line division problem in computer justified text	algorithms, line division, text editing, layout spacing	Programming techniques
d_{27}	An efficient garbage compaction algorithm	algorithms, garbage collection, compaction, relocation, storage reclamation, programming techniques	Programming techniques
d_{28}	Transactions and consistency in distributed database systems	database management, distributed systems, data representation, data organization, recovry	Database management
d_{29}	Determining view dependencies using tableaux	database management, relational model, distributed systems	Database management
d_{30}	A new normal form for the design of relational database schemata	database management, relational model, functional dependencies, database schema	Database management
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{31}	Permutation of data blocks in a bubble memory	bubble memory, algorithm, memory loops, permutations, programming techniques	Programming techniques
d_{32}	The impact of distributions and disciplines on multiple processor systems	multiprogramming, multiprocessing, performance evaluation, computer systems	Computer systems
d_{33}	An event-driven compiling technique	compilers, programming language, process, parallelism, event	Programming languages
d_{34}	Syntactic source to source transforms and program manipulation	programming language, structured programming, program transforms, control language	Programming languages
d_{35}	Recursive data structures in APL	recursive data structures, programming languages, theory of programming, trees, data-driven algorithm	Programming languages
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{36}	Global optimization by suppression of partial redundancies	optimization, compilers, redundancy elimination, programming languages, data flow analysis	Programming languages
d_{37}	Comments on perfect hashing functions: A single probe retrieving method for static sets	hashing, hash-code, reduction, scatter storage, programming techinques, searching	Programming techniques
d_{38}	Synchronization with eventcounts and sequencers	process synchronization, interprocess communication, distributed systems, security kernel	Operating system principles
d_{39}	Optimal storage allocation for serial files	serial files, storage allocation, resource allocation, operating systems	Operating system principles
d_{40}	FOCUS microcomputer number system	number representation, computational accuracy, computational speed, computer system	Computer systems
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{41}	An improved algorithm for decentralized extrema-finding in circular configurations of processes	distributed systems, operating systems, decentralized algorithms, operating system	Operating system principles
d_{42}	Reasoning about arrays	arrays, program proving, binary search, ordering	Programming languages
d_{43}	A model for and discussion of multi-interpreter systems	interpreters, transfer-of-control, computer systems	Computer systems
d_{44}	The time and state relationships in simulation modeling	model representation, simulation programming languages, model documentation, time flow mechanisms	Simulation modeling and statistical computing
d_{45}	Concepts and criteria to access acceptability of simulation studies	model documentation, model validation, model robustness, software validation, software representation	Simulation modeling and statistical computing
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{46}	A methodology for cost-risk analysis in the statistical validation of simulation models	validation, simulation, cost-risk analysis	Simulation modeling and statistical computing
d_{47}	Asynchronous distributed simulation via a sequence of parallel computations	event simulation, distributed systems, deadlock, message-passing systems	Simulation modeling and statistical computing
d_{48}	Use of distributions in approximate solutions to nonstationary MMS queues	queueing delays, queueing approximation, system dynamics, event simulation	Simulation modeling and statistical computing
d_{49}	A conceptual framework for research in the analysis of simulation output	output analysis, confidence intervals, measures of effectiveness	Simulation modeling and statistical computing
d_{50}	A spectral method for confidence interval generation and run length control in simulations	simulation, confidence intervals, validation, batch means	Simulation modeling and statistical computing
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{51}	Control of initialization in multivariate simulation response	simulation, initialization, multivariate response, simulation modeling and statistical computing	Simulation modeling and statistical computing
d_{52}	Minimum mean-squared-error estimators for simulation experiments	surrogate process, simulation experiments, flow mecanisms	Simulation modeling and statistical computing
d_{53}	Regression-adjusted estimates for regenerative simulations with graphics	simulation, output analysis, ratio estimator, graphics	Simulation modeling and statistical computing
d_{54}	Multidimensional divide and conquer	analysis of algorithms, data structures, computational geometry, programming techniques, algorithms	Programmming techniques
d_{55}	A unifying look at data structures	data structures, dictionaries, linear search, merge, programming techniques, analysis of algorithms	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{56}	A virtual machine emulator for performance evaluation	performance control, virtual machines, operating system	Operating system principles
d_{57}	Pilot : An operating system for a personal computer	personal computer, operating system, virtual memory, file system, system structure	Operating system principles
d_{58}	Medusa : An experiment in distributed operating system structure	operating system, distributed systems, message systems, task, deadlock	Operating system principles
d_{59}	Experience with processes and monitors in MESA	concurrency, deadlock, monitor, operating system, process, task	Operating system principles
d_{60}	Specification and verification of the UCLA Unix security kernel	security, operating system, protection, programming methodology, formal specifications, kernel	Operating system principles
d_{61}	Hierarchical binary search	data structures, searching, binary search, file organization	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{62}	A linear algorithm for copying binary trees using bounded workspace	binary trees, tree traversal	Programming techniques
d_{63}	Password security: A case history	operating systems, computer security, passwords	Operating systems principles
d_{64}	Storing a sparse table	parsing, searching, programming techniques, table lookup	Programming techniques
d_{65}	How to share a secret	cryptography, key management, interpolation	Programming techniques
d_{66}	On the proof of correctness of a calendar program	program specification, program verification, inductive assertions	Programming languages
d_{67}	Computing connected components on parallel computers	algorithms, parallel processing, transitive closure, programming techniques	Programming techniques
d_{68}	Secure personal computing in an insecure network	personal computing, security, privacy, networks, operating systems	Operating system principles
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{69}	Proving termination with multiset orderings	program correctness, program transformation, program verification, well founded sets, bags, tree replacement systems	Programming languages
d_{70}	An optimal real-time algorithm for planar convex hulls	Computational geometry, convex hull, real-time algorithm, algorithm complexity, programming techniques	Programming techniques
d_{71}	Storage reorganization techniques for matrix computation in a paging environment	matrix multiplication, virtual memory, pagination, programming techniques	Programming techniques
d_{72}	The control of response times in multi-class systems by memory allocation	paging, virtual memory, performance control, operating system	Operating system principles
d_{73}	Algorithm = logic + control	control language, non-procedural language, program specification, programming languages	Programming languages
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{74}	A conceptual framework for a nonprocedural programming language	parallel programming, nonprocedural language, data flow analysis	Programming languages
d_{75}	A case study of a new code generation technique for compilers	compilers, code generation, concatenation, program documentation, optimization, data flow analysis	Programming languages
d_{76}	An exercise in proving parallel programs correct	garbage collection, multiprocessing, program correctness	Programming languages
d_{77}	A language for formal problem specification	formal specifications, program correctness, parallel processing, synchronization, programming languages	Programming languages
d_{78}	A methodology for interactive computer service measurement	computer networks, performance, measurement model, computer systems	Computer systems
d_{79}	The cray-1 computer system	computer system, architecture	Computer systems
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{80}	The development of the MU5 computer system	architecture, virtual storage, computer architecture, somputer system, associative storage	Computer systems
d_{81}	The evolution of the DEC system	computer networks, architeture, operating systems, time sharing, computer systems	Computer systems
d_{82}	Architecture of the IBM system/370	computer systems, architecture, instruction sets, virtual storage	Computer systems
d_{83}	Insertions and deletions in one-sided height-balanced trees	balanced trees, binary search, dynamic programming, programming techniques	Programming techniques
d_{84}	Preserving average proximity in arrays	algorithm, arrays, linear search, trees, programming techniques	Programming techniques
d_{85}	Anomalies with variable partition paging algorithms	operating system, memory management, program behavior, virtual memory, paging algorithms	Operating system principles
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{86}	Implications of structured programming for machine architecture	machine architecture, computer architecture, computer networks, computer systems, program characteristics	Computer systems
d_{87}	A technique for isolating differences between files	hash coding, file compression, programming techniques, text editing, program maintenance	Programming techniques
d_{88}	Optimal conversion of extended-entry decision tables with general cost criteria	optimal algorithm, dynamic programming, programming techniques, decision table	Programming techniques
d_{89}	List processing in real time on a serial computer	real-time, compacting, garbage collection, list processing, virtual memory, storage allocation, programming techniques, storage management	Programming techniques
d_{90}	Secure communications over insecure channels	security, cryptography, key management, programming techniques, public key cryptosystem	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{91}	Assembling code for machines with span-dependent instructions	code generation, assemblers, compilers, programming techniques, computational complexity	Programming techniques
d_{92}	A data structure for manipulating priority queues	data structures, priority queues, merge, programming techniques, binary trees	Programming techniques
d_{93}	Economical encoding of commas between strings	delimiters, encoding of the integers, commas, programming techniques	Programming techniques
d_{94}	Fast parallel sorting algorithms	parallel processing, algorithms, bucket sort, programming techniques	Programming techniques
d_{95}	A time -and space- efficient garbage compaction algorithm	garbage collection, compaction, storage reclamation, storage allocation, programming techniques, list processing, data structures	Programming techniques
...	...	...	...

Documents	Titre	Indexes	Domaine d'intérêt
d_{96}	Feedback coupled resource allocation policies in the multiprogramming multiprocessor computer system	multiprogramming systems, system scheduling, computer systems	Computer systems
d_{97}	Hybrid simulation models of computer systems	performance evaluation, simulation, central server, computer systems, queueing models	Computer systems
d_{98}	A practical interprocedural data flow analysis algorithm	data flow analysis, optimization, relations, programming languages, reference parameters	Programming languages
d_{99}	A model for verification of data security in operating systems	operating systems, security, program verification, protection	Operating system principles
d_{100}	Generalized working sets for segment reference strings	memory management, paging, program behavior, working sets, segmentation	Operating system principles