OPTIMISATION DE REQUETES DANS UN SYSTEME DE RECHERCHE D'INFORMATIONAPPROCHE BASEE SUR L'EXPLOITATION DE TECHNIQUES AVANCEES DE L'ALGORITHMIQUE GENETIQUE

Lynda Tamine

N° d'ordre : Laboratoire IRIT – Pôle SIG

THESE

Présentée devant

l'UNIVERSITE PAUL SABATIER DE TOULOUSE

en vue de l'obtention du

DOCTORAT DE L'UNIVERSITE PAUL SABATIER

Spécialité **INFORMATIQUE**

Par

Lynda Tamine

OPTIMISATION DE REQUETES DANS UN SYSTEME DE RECHERCHE D'INFORMATION

APPROCHE BASEE SUR L'EXPLOITATION DE TECHNIQUES AVANCEES DE L'ALGORITHMIQUE GENETIQUE

Soutenue le 21/12/ 2000 devant le jury composé de :

M. A. Flory	Professeur à l'INSA de Lyon
M. J. Savoy	Professeur à l'Université de Neuchâtel
M. C. Chrisment	Professeur à l'Université Paul Sabatier de Toulouse
Mme F. Sedes	Professeur à l'Université Paul Sabatier de Toulouse
M. M. Boughanem	HDR, Maître de conférences à l'Université de Toulouse II
Mme. C.SouleDupuy	Maître de Conférences à l'Université de Toulouse I

INSTITUT DE RECHERCHE EN INFORMATIQUE DE TOULOUSE
Centre National de la Recherche Scientifique- Institut National Polytechnique- Université P. Sabatier

Université P. Sabatier, 118 route de Narbonne, 31062 Toulouse Cedex. Tel : 61.55.66.11

Résumé

Les travaux présentés dans cette thèse traitent des apports de l'algorithmique génétique à la conception de systèmes de recherche d'information adaptatifs aux besoins des utilisateurs.
Notre étude se focalise en premier lieu, sur l'analyse des différents modèles et stratégies de représentation et de recherche d'information. Nous mettons notamment en évidence, leur contribution à la résolution des problèmes inhérents à la recherche d'information.
En second lieu, notre intérêt s'est porté sur l'étude des algorithmes génétiques. Nous basant alors sur leur robustesse, théoriquement et expérimentalement prouvée, nous intégrons ces algorithmes à la mise en oeuvre de notre approche d'optimisation de requête.
Nous présentons une approche de recherche d'information qui intègre une stratégie de reformulation de requête par injection de pertinence, fondée sur l'hybridation d'un processus d'optimisation génétique, à un modèle de recherche de base. Nous proposons un algorithme spécifique à la recherche d'information, par l'intégration d'opérateurs génétiques augmentés par la connaissance du domaine d'une part, et d'une heuristique permettant de résoudre le problème de multimodalité de la pertinence d'autre part. L'heuristique de nichage en l'occurrence, est diffusée globalement lors de l'évolution de l'AG. La population est alors organisée en niches de requêtes effectuant une recherche parallèle et coopérative d'informations

Nous évaluons enfin notre approche de recherche d'information, au travers d'expérimentations réalisées à l'aide du système Mercure, sur la collection de référence TREC.

Mots-clés : **Système de Recherche d'information** **Algorithme Génétique** **Reformulation de Requête**

A Mourad et Katia

Remerciements

L'achèvement de tout travail mené sur plusieurs années procure une grande satisfaction. Il est l'occasion de se remémorer les étapes passées et les personnes qui y ont contribué.

Aussi, j'adresse mes sincères remerciements à mon Directeur de thèse Monsieur Claude Chrisment, ainsi qu'à Monsieur Jacques Luguet et Monsieur Gilles Zurfluh, Professeurs à l'Université Paul Sabatier de Toulouse qui m'ont accueillie au sein de leur équipe et m'ont apportée une aide très précieuse lors de la réalisation de ce travail.

Je tiens à exprimer ma profonde gratitude à Monsieur Mohand Boughanem, Maître de Conférences à l'Université de Toulouse II pour avoir dirigé mes recherches. Ses conseils, ses critiques, sa ferme volonté de collaboration ainsi que la confiance qu'il m'a toujours témoignée m'on été d'un grand apport tout au long de mes recherches. Qu'il soit assuré de mon très grand respect.

Je souhaite exprimer ma gratitude à Monsieur Jacques Savoy Professeur à l'Université de Neuchâtel (Suisse) et Monsieur André Flory, Professeur à l'INSA de Lyon qui ont accepté d'évaluer ce travail afin d'en être les rapporteurs. Je les remercie pour l'honneur qu'ils me font en participant à ce jury tout comme Mme Florence Sedes, Professeur à l'Université Paul Sabatier de Toulouse et Mme Chantal Soule-Dupuy, Maître de Conférences à l'Université de Toulouse I.

Je tiens à renouveler toute ma reconnaissance à Monsieur Karim Tamine, Maître de Conférences à l'Université de Limoges pour ses encouragements, son investissement personnel et son fidèle soutien.

Je tiens également à remercier Monsieur Malik Si-Mohammed, Maître Assistant Chargé de Cours à l'Université de Tizi-Ouzou (Algérie) qui m'a intégrée au sein de son équipe de recherche et n'a cessé depuis, à me témoigner sa confiance.

Un grand merci à Melle Karima Amrouche pour ses encouragements renouvelés tout au long de cette thèse.
Mes remerciements vont de même à tous les membres de l'équipe IRIT-SIG et particulièrement à Melle Nawal Nassr.

Je souhaite exprimer ma profonde gratitude à toute ma famille et belle famille pour leur soutien indéfectible.

Enfin, je tiens à exprimer toute ma reconnaissance à Mourad pour ses conseils, ses encouragements, son dévouement et sa bravoure.

La diversité des contributions apportées tout au long de ce travail en firent une expérience très intéressante et formidablement enrichissante. Merci à tous.

Sommaire

Partie 1

Recherche d'Information et Algorithmique Génétique

Chapitre 1 : ***Recherche d'Information : Modèles et Techniques***

Chapitre 2 : ***Concepts et Principes des Algorithmes Génétiques***

Chapitre3 : *Application des Algorithmes Génétiques à la Recherche d'Information*

Partie 2
Mise en Œuvre d'un Algorithme Génétique Adapté à l'Optimisation de Requête dans un Système de Recherche d'Informations

Partie 1

Recherche d'Information
Et
Algorithmique Génétique

Introduction

Les récentes innovations technologiques ont redonné à l'information au sens large, de nouveaux contours. L'information n'est plus confinée au seul domaine des spécialistes ; elle est en effet devenue une ressource stratégique convoitée par divers profils d'utilisateurs, en nombre sans cesse croissant.
Dés lors, nous assistons depuis une décennie à la prolifération des services de l'information. Internet est à ce titre, le réseau d'informations le plus sollicité de nos jours. C'est un outil de communication potentiel qui offre de très nombreux avantages : ergonomie, accès aisé à l'information, structure décentralisée etc…
L'immense vague de ses utilisateurs génère ainsi une masse d'informations phénoménale qui impose des outils efficaces d'organisation et de recherche.

Dans un contexte très large, un système de recherche d'information capitalise un volume d'informations relativement considérable et offre des techniques et outils permettant de localiser l'information pertinente relativement à un besoin en information, exprimé par l'utilisateur. Dans un contexte précis, la recherche documentaire est une activité quotidienne très largement pratiquée par des catégories d'utilisateurs très diversifiées : entreprises, particuliers, banques, institutions scolaires etc..
Un système de recherche d'information manipule dans ce cas, une collection de documents traduisant des connaissances hétérogènes et indépendantes qu'il convient d'homogénéiser à travers la découverte d'associations sémantiques, dans le but de structurer la réponse au besoin exprimé par l'utilisateur.

Parmi les grandes questions qui agitent à ce jour, les travaux dans le domaine de la recherche d'information documentaire, deux revêtent une importance déterminante, de lourd impact sur l'efficacité des systèmes : interfaçage utilisateur - système, représentation et recherche d'information.
La première question traite des difficultés inhérentes à la communication homme-machine. Cette dernière cible deux principaux objectifs : le premier est de permettre à l'utilisateur une expression aisée de son besoin en information. Le vœu de répandre l'utilisation des systèmes de recherche d'information à un grand nombre d'utilisateurs, plaide en faveur de la mise en oeuvre d'interfaces conviviales et langages d'interrogation libres. Ceci met alors en amont le problème de perception du système qui est la cible du second objectif. Dans ce cadre, les travaux ciblent la compréhension fidèle et exhaustive de la requête utilisateur. Des modèles de représentation sémantique et mécanismes d'indexation ont été à cet effet, proposés. Cependant, la modélisation des unités textuelles représentatives du contenu sémantique des requêtes, ne saurait être concluante sans la mise en oeuvre de modèles représentatifs des documents d'une part, et mécanismes d'appariement requête-document d'autre part. Ceci est le domaine d'intérêt de la seconde question.

Les modèles de recherche et représentation d'information sont basés sur un processus de mise en correspondance entre requêtes utilisateurs et documents de la collection. Le mécanisme de recherche détermine alors, sur la base d'un degré de pertinence supposé des documents, ceux qui répondent au besoin de l'utilisateur. De nombreux modèles et stratégies sont développés dans la littérature.
Les modèles classiques (vectoriel, booléen, probabiliste) sont fondés sur l'utilisation de théories mathématiques tant pour la représentation des unités textuelles que pour le calcul de la pertinence des documents. L'inconvénient majeur de ces modèles, réside notamment dans le fait qu'ils induisent une manipulation de concepts de manière indépendante. L'intégration de l'aspect associatif, au travers de liaisons statistiques, ne pallie toutefois pas aux problèmes de dépendance des structures initiales, les liaisons entre informations y sont en effet difficiles à maintenir.
Le modèle vectoriel généralisé et modèle LSI apportent une solution judicieuse à ce problème. Basés sur des techniques d'algèbre linéaire, ces modèles présentent l'originalité et avantage de rapprocher, dans l'espace documentaire défini par la collection, les documents conceptuellement ressemblants.

Dans le but d'accroître les performances des modèles de recherche de base, de nombreuses stratégies sont mises en œuvre afin d'y être greffées. Ces stratégies exploitent diverses sources d'évidence : relations sémantiques définies dans le thesaurus, classes et contextes d'utilisation des concepts, résultats de recherche, jugement de pertinence des utilisateurs, éléments de la théorie de l'information, heuristiques etc…
Pour notre part, nous nous intéressons à la mise en œuvre d'une stratégie d'optimisation de requête basée sur les algorithmes génétiques. Ces derniers sont des métaphores biologiques inspirées des mécanismes de l'évolution darwinienne et de la génétique moderne, utilisées comme un outil puissant d'optimisation.
Nous exploitons les concepts et techniques de l'algorithmique génétique afin de mettre en œuvre un processus d'optimisation de requête caractérisé par une exploration efficace du fond documentaire et recherche graduelle et coopérative d'informations.

L'organisation retenue pour la présentation de nos travaux et le domaine dans lequel ils s'inscrivent, s'articule en deux parties.
La première traite de la recherche d'information et des algorithmes génétiques.
Le premier chapitre présente la problématique de recherche d'information ainsi que les différents modèles et stratégies de recherche et de représentation d'information proposés dans la littérature.

Le second chapitre présente les algorithmes génétiques sous l'angle de l'optimisation. Nous décrivons le processus d'évolution qu'il induisent à travers la description de la structure des différents opérateurs génétiques. La preuve théorique des résultats d'un

algorithme génétique est présentée à travers l'étude du théorème fondamental. Nous examinons ensuite, quelques techniques et opérateurs avancés d'exploration génétique.

Le troisième chapitre rapporte les principaux travaux d'application des algorithmes génétiques à la recherche d'information.

La deuxième partie présente notre contribution à la mise en œuvre de stratégies de recherche d'information à travers la description de notre approche d'optimisation de requête.
Le quatrième chapitre présente globalement notre approche. Nous y décrivons notamment nos motivations, caractéristiques fondamentales de l'AG d'optimisation de requête que nous proposons, fonctionnement général du SRI et structures et objectifs des opérateurs génétiques proposés. Enfin, un bilan critique et préliminaire de notre approche est présenté et sert de cadre de réflexions pour définir de nouvelles directions à nos travaux.

Le cinquième chapitre présente une nouvelle version de notre approche. Nous y présentons alors les principaux éléments révisés, justifiés à l'aide d'arguments expérimentaux ou théoriques, liés au nouveau mode d'exploration de l'espace documentaire.
On y présente également les résultats d'expérimentations réalisées dans le but de valider notre approche d'optimisation de requête. Cette évaluation a pour but de mesurer l'efficacité de l'algorithme proposé et estimer l'impact de chacune de ses caractéristiques sur les résultats de la recherche.

En conclusion, nous dressons un bilan de nos travaux, en mettant en exergue les éléments originaux que nous introduisons. Nous présentons ensuite les perspectives d'évolution de ces travaux.

Une annexe est enfin présentée pour décrire le système de recherche d'information Mercure, utilisé pour la réalisation de nos expérimentations.

Chapitre 1

Recherche d'Information : Modèles et Techniques

Introduction

Dans le contexte particulier de l'information documentaire, les systèmes de recherche d'information sont au centre des préoccupations des entreprises, administrations et grand public. Un Système de recherche d'information manipule un ensemble de documents sous forme d'unités informationnelles ou conteneurs sémantiques, non décomposables. L'objectif d'un système de recherche d'information est d'aiguiller la recherche dans le fond documentaire, en direction de l'information pertinente relativement à un besoin en information exprimé par une requête utilisateur. A cet effet, le système assure les fonctionnalités de communication, stockage, organisation et recherche d'information.

Les modèles de recherche d'information s'associent généralement à des modèles de représentation et poursuivent l'objectif de mise en correspondance des représentants de documents et représentants de requêtes. Le mécanisme de recherche identifie ainsi, l'information susceptible de répondre à la requête, en associant un degré de pertinence supposé aux documents restitués.

De nombreux modèles et stratégies sont proposés dans la littérature. Les approches sont basées sur des modèles formels (booléen, vectoriel, probabiliste) et techniques qui puisent dans une large mesure d'heuristiques de recherche, théorie de l'information, réseaux de neurones et algorithmes génétiques.

A ce titre, les algorithmes génétiques sont une reproduction artificielle des mécanismes naturels d'adaptation issue de la théorie darwinienne. Sous le double aspect d'optimisation et d'apprentissage, les algorithmes génétiques ont pour objectif fondamental de faire évoluer une population de connaissances vers des connaissances idéales, grâce à des transformations génétiques analogues au croisement et mutation biologiques. La robustesse de leur principe d'exploration d'espaces complexes et efficacité de l'évolution qu'ils induisent, sont des éléments qui justifient leur application au domaine de la recherche d'information.

Dans le premier chapitre, nous présentons les principaux modèles et techniques utilisés en recherche d'information. Une attention particulière y sera portée sur les stratégies de reformulation de requête.

Dans le second chapitre, nous présentons les concepts et principes de base des algorithmes génétiques. Nous y décrivons également les principales techniques avancées utilisées pour l'adaptation de ces algorithmes à la résolution de problèmes d'optimisation spécifiques.

Nous présentons enfin dans le troisième chapitre, les principaux travaux d'application des algorithmes génétiques à la recherche d'information.

1. Introduction

L'intérêt stratégique porté à l'information sous ses différentes facettes, combinée à l'avènement explosif d'Internet et autres services de l'information sont des facteurs déterminants qui justifient la multiplication de directions de recherche ayant pour objectif de mettre en œuvre des processus automatiques d'accès à l'information, sans cesse plus performants.
Un système de recherche d'information nécessite la conjugaison de modèles et algorithmes permettant la représentation, le stockage, la recherche et la visualisation d'informations. L'objectif fondamental de la recherche d'information consiste à mettre en œuvre un mécanisme d'appariement entre requête utilisateur et documents d'une base afin de restituer l'information pertinente.
L'élaboration d'un processus de recherche d'information pose alors des problèmes liés tant à la modélisation qu'à la localisation de l'information pertinente. En effet, la recherche d'information induit un processus d'inférence de la sémantique véhiculée par l'objet de la requête, en se basant sur une description structurelle des unités d'informations.

1.1. Motivations

Avant la dernière décennie, le champ d'exploitation des SRI[1] a connu une croissance graduelle qui a motivé les travaux de recherche en la matière et abouti à la définition de nombreux modèles de représentation, recherche, architecture et interfaces.
Toutefois, depuis le début des années 1990, le monde assiste à un processus de maturation qui se traduit par une production massive d'informations et d'une explosion du nombre de ses consommateurs. Actuellement, les facteurs déterminants qui plaident pour une réflexion plus mûre afin d'asseoir des modèles et techniques efficaces de recherche d'information sont principalement :

1. L'introduction généralisée de l'informatique personnelle et des réseaux de communication. On cite particulièrement le réseau Internet qui constitue un réservoir universel d'informations et un moyen de communication très convivial, utilisé à une très grande échelle. La structure très décentralisée d'Internet a pour corollaire une grande hétérogénéité de son contenu. Ceci, combiné à la grande diversité des profils de ses utilisateurs, rend le processus de recherche d'information plus ardu.

2. L'explosion des sources d'informations et par voie de conséquence une croissance déterminante de ses consommateurs.

[1] Système de Recherche d'information

3. La nécessité économique actuelle de disposer et exploiter l'information à la même cadence que sa dynamique de production.

1.2. Notions de base

Un SRI intègre un ensemble de modèles pour la représentation des unités d'information (documents et requêtes) ainsi qu'un processus de recherche/décision qui permet de sélectionner l'information pertinente en réponse au besoin exprimé par l'utilisateur à l'aide d'une requête. Le processus U de recherche d'information est illustré sur la figure1.1. On y dégage les principaux mots clés suivants :

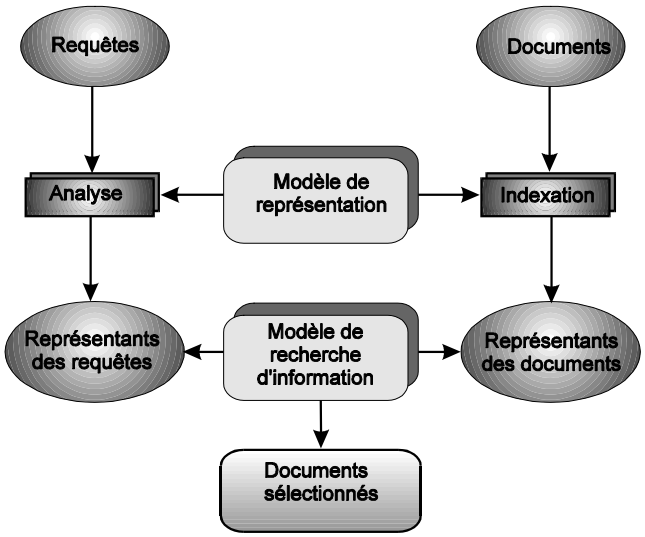


Figure 1.1 : Processus U de recherche d'information

1. Document

Le document constitue le potentiel d'informations élémentaire d'une base documentaire. La taille d'un document et son contenu sémantique dépendent en grande partie du domaine d'application considéré. On distingue principalement deux types de bases documentaires : les référothèques et bibliothèques.

- Les référothèques

Une référothèque est constituée d'un ensemble d'enregistrements faisant référence au document dans lequel se trouve l'information intégrale [Mothe, 1994]. Une unité d'informations est composée d'un résumé du texte intégral (abstract) et de données factuelles complémentant la description du document.

- Les bibliothèques

Le document comprend dans ce cas, le texte intégral (full text). Ceci pose alors des problèmes de structuration de documents pour la localisation des parties pertinentes, et stockage optimisant l'accès à l'information.

2. Requête

La requête est l'expression du besoin en informations de l'utilisateur . A cet effet, divers types de langages d'interrogation sont proposés dans la littérature. Une requête peut être décrite :

- par une liste de mots clés : cas des systèmes SMART [Salton, 1971] et Okapi[Robertson & al, 1999],
- en langage naturel : cas des systèmes SMART [Salton, 1971] et SPIRIT [Fluhr & Debili, 1985],
- en langage bouléen : cas des systèmes DIALOG [Bourne & Anderson, 1979]
- en langage graphique : cas du système issu du projet NEURODOC [Lelu & François, 1992].

3. Modèle de représentation

Le modèle de représentation constitue un référentiel qui définit un ensemble de règles et notations permettant la traduction d'une requête ou document à partir d'une description brute vers une description structurée. Ce processus de conversion est appelé **indexation**. L'indexation est un processus permettant d'extraire d'un document ou d'une requête, une représentation paramétrée qui couvre au mieux son contenu sémantique. Le résultat de l'indexation constitue le **descripteur** du document ou requête.

Le descripteur est une liste de **termes** ou **groupes de termes** significatifs pour l'unité textuelle correspondante, généralement assortis de poids représentant leur degré de représentativité du contenu sémantique de l'unité qu'ils décrivent.

L'indexation est une étape fondamentale dans la conception d'un SRI puisqu'elle est à l'origine de la constitution des représentants de documents qui sont susceptibles de s'apparier avec les requêtes des utilisateurs. De la qualité de l'indexation, dépend en partie la qualité des réponses du système.

L'ensemble des termes reconnus par le SRI sont rangés dans une structure appelée **dictionnaire** constituant le **langage d'indexation**. Le langage d'indexation peut être contrôlé ou libre. Dans le cas d'un langage contrôlé, une expertise préalable sur le domaine d'application considéré, établit un vocabulaire exhaustif représenté dans une structure dénommée le **thesaurus**. La description des documents n'est effectuée que moyennant les termes de ce vocabulaire.

Ce type de langage garantit le rappel de documents lorsque la requête utilise dans une large mesure les termes du vocabulaire. En revanche, il y a risque important de perte d'informations lorsque la requête s'éloigne du vocabulaire et qu'il y a absence de relations sémantiques entre termes. Dans le cas d'un langage libre, le dictionnaire est enrichi en cours d'exploitation du système. Il en ressort une difficulté dans la maîtrise du vocabulaire en raison des différences de perceptions des utilisateurs et des indexeurs.

4. **Modèle de recherche**
   C'est le modèle noyau d'un SRI. Il comprend la fonction de décision fondamentale qui permet d'associer à une requête, l'ensemble des documents pertinents à restituer. Notons que le modèle de recherche d'information est étroitement lié au modèle de représentation des documents et requêtes

1.3. Problématique

Nous présentons dans ce qui suit, les principaux points problématiques de la recherche d'information.

- Le besoin en informations formulé par une requête utilisateur est généralement **vague**, **imprécis**; il s'ensuit que **l'objet de la recherche d'information est à priori inconnu**. Le processus de recherche d'information doit s'appuyer sur des hypothèses de description incertaine des requêtes

- **Les univers de référence des auteurs et utilisateurs sont différents**. Les auteurs sont des spécialistes du domaine alors que les utilisateurs n'en sont pas forcément. L'utilisateur interroge le système en méconnaissance du contenu sémantique de la base documentaire (termes, concepts,...). Le lexique de l'utilisateur et lexique du système étant différents, **l'appariement requête-documents est alors approximatif**.

- Les procédures d'indexation automatique sont fondées sur l'utilisation du mot ou groupe de mots pour la représentation sémantique des documents et requêtes. Ce procédé est toutefois mis en échec par les propriétés d'**ambiguïté** et de **recouvrement de concepts** connus en langage naturel. Les méthodes linguistiques nécessitent pour cela, un volume important de connaissances induisant une complexité souvent rédhibitoire pour le traitement.

- La notion de **pertinence** dépend étroitement de l'utilisateur. La relation qu'elle induit est non intrinsèque est par conséquent difficile à formaliser.

La conséquence immédiate et perceptible à cet ensemble de difficultés est qu'un ensemble de documents pertinents à la requête utilisateur n'est pas sélectionné par le SRI. Ceci traduit le phénomène non désirable de silence du système.

2. Les modèles de recherche d'information

Un modèle de recherche d'information est formellement décrit par un quadruple **[D, Q, F, R(q_i , d_j)]** [Yates & Neto, 1999] où :

D : Ensemble des représentants de documents de la collection
Q : Ensemble de représentants des besoins en informations
F : Schéma du support théorique de représentation des documents, requêtes et relations associées
R(q_i , d_j) : Fonction d'ordre associée à la pertinence

La définition d'un modèle de recherche d'information induit ainsi la détermination d'un support théorique comme base de représentation des unités d'informations et de formalisation de la fonction pertinence du système. De très nombreux modèles sont proposés dans la littérature. Le présent paragraphe a pour objectif d'en présenter les principaux modèles de base et modèles dérivés construits sur chacun d'eux.

Nous adoptons dans la suite, les principales notations suivantes :

Q_k : kième requête
D_j : jème document de la collection
RSV(Q_k,D_j) : Valeur de pertinence associée au document D_j relativement à la requête Q_k
q_{ki} : Poids d'indexation du terme t_i dan la requête Q_k
d_{ji} : Poids d'indexation du terme t_i dans le document D_j
T : Nombre total de termes d'indexation dans la collection
N : Nombre total de documents dans la collection
n_i : Nombre de documents de la collection contenant le terme t_i

2.1. Le modèle booléen

2.1.1. Le modèle de base

Le modèle booléen propose la représentation d'une requête sous forme d'une équation logique. Les termes d'indexation sont reliés par des connecteurs logiques *ET*, *OU* et *NON*. Le processus de recherche mis en oeuvre par le système consiste à effectuer des opérations sur ensembles de documents définis par l'occurrence ou absence de termes d'indexation afin de réaliser un appariement exact avec l'équation de la requête. De manière formelle, le modèle de recherche booléen est défini par un quadruplet *(T, Q, D, F)*

Où :
T: Ensemble des termes d'indexation
Q : Ensemble de requêtes booléennes
D : Ensemble des documents de la collection

F : Fonction présence définie par : D x Q ⟶ {0 ,1}
F(d,t) = 1 si t occurre dans D
= 0 sinon

Sur la base de cette fonction, on calcule la ressemblance relativement à la forme de la requête comme suit :

Formulation booléenne	***Formule d'évaluation***
$F(d_k , t_i$ et $t_j)$	Min $(F(d_k , t_i) , F(d_k , t_j)) = F(d_k , t_i) * F(d_k , t_j)$
$F(d_k , t_i$ ou $t_j)$	Max $(F(d_k , t_i) , F(d_k , t_j)) = F(d_k , t_i) + F(d_k , t_j)$- $F(d_k , t_i) * F(d_k , t_j)$
$F(d_k$, Non $t_i)$	$1 - F(d, t_i)$

Le modèle booléen présente le principal avantage de simplicité de mise en oeuvre. Toutefois, il présente les principaux inconvénients suivants :
- les formules de requêtes sont complexes, non accessibles à un large public,
- la réponse du système dépend de l'ordre de traitement des opérateurs de la requête,
- la fonction d'appariement n'est pas une fonction d'ordre,
- les modèles de représentation des requêtes et documents ne sont pas uniformes. Ceci rend le modèle inadapté à une recherche progressive.

2.1.2. Le modèle booléen étendu

Le modèle booléen étendu [Fox,1983] [Salton, 1989] complète le modèle de base en intégrant des poids d'indexation dans l'expression de la requête et documents. Ceci a pour conséquence la sélection de documents sur la base d'un appariement rapproché (fonction d'ordre) et non exact.
A cet effet, l'opérateur L_p-*Norm* est défini pour la mesure de pertinence requête-document. Cette mesure est évaluée pour des requêtes décrites sous la forme conjonctive ou disjonctive, comme suit :

Opérateur OR : $$RSV(Q_k,D_j)=\left[\frac{\sum_{i=1}^{T}\sum_{i=1}^{T} q_{ki}^{P} d_{ji}^{P}}{\sum_{i=1}^{T} q_{ki}^{P}}\right]^{1/P}$$

Opérateur AND : $$RSV(Q_k,D_j)=\left[\frac{\sum_{i=1}^{T}\sum_{i=1}^{T} q_{ki}^{P}(1-d_{ji}^{P})}{\sum_{i=1}^{T} q_{ki}^{P}}\right]^{1/P}$$

Où :
P : Constante

La littérature rapporte qu'aucune méthode formelle n'est proposée pour la détermination de la valeur du paramètre *P* [Ponte, 1998].

2.1.3. Le modèle des ensembles flous

La théorie des ensembles flous est due à Zadeh [Zadeh, 1965]. Elle est basée sur l'appartenance probable, et non certaine, d'un élément à un ensemble. Un ensemble flou est formellement décrit comme suit :

$$EF=\{(e_1, f_{EF}(e_1)) , ... (e_n, f_{EF}(e_n)) \}$$

Où :

e_i : Elément probable de E

f_{EF} : E ⟶ [0, 1]

e_i ⟶ $f_{EF}(e_i)$ = degré d'appartenance de e_i à E

Les opérations de base sur les ensembles flous sont alors définies comme suit :

Intersection : $f_{A\cap B}(e_i)=Min(f_A(e_i), f_B(e_i)) \quad \forall e_i \in E$

Union : $f_{A\cup B}(e_i)=Max(f_A(e_i), f_B(e_i)) \quad \forall e_i \in E$

Complément : $f_{A'}(e_i)=1- f_B(e_i) \quad \forall e_i \in E \;\; avec\; A'=\{x \in A \wedge x \notin B\}$

Une extension du modèle booléen basée sur les ensembles flous est proposée par Salton [Salton, 1989]. L'idée de base est de traiter les descripteurs de documents et requêtes comme étant des ensembles flous. L'ensemble flou des documents supposés pertinents à une requête est obtenu en suivant les étapes suivantes :

1. Pour chaque terme t_i de Q_k, construire l'ensemble flou D_t des documents contenant ce terme.
2. Effectuer sur les ensembles D_t, les opérations d'intersection et union selon l'ordre décrit dans l'expression de Q_k relativement aux opérateurs *ET* et *OU* respectivement.
3. Ordonner l'ensemble résultat de la précédente opération selon le degré d'appartenance de chaque document à l'ensemble associé à chaque terme.

Exemple

Soit la requête $Q_k = t_1 \wedge (t_2 \vee t_3)$

En posant :

$Q_{k1} = t_1 \wedge t_2 \wedge t_3$, $Q_{k2} = t_1 \wedge t_2 \wedge \neg t_3$, $Q_{k3} = t_1 \wedge \neg t_2 \wedge t_3$,

On obtient l'expression disjonctive suivante de la requête Q_k : $Q_k = Q_{k1} \vee Q_{k2} \vee Q_{k3}$

1. On construit les ensembles flous d'occurrence des termes t_1 , t_2 et t_3 dans les documents D_1, D_2, D_3, D_4 et D_5, soient :
   D_{t1} ={ 0.2, 0.4, 0.2, 0.6, 0.8}
   D_{t2} = {0.1, 0.8, 0.4, 0.3, 0 }
   D_{t3} = { 0.4, 1, 0.1, 0.1, 0.2}
   On construit les ensembles compléments :
   D_{t2} = {0.9, 0.2, 0.6, 0.7, 1 }
   D_{t3} = { 0.6, 0, 0.9, 0.9, 0.8 }

2. Les ensembles pertinents associés à la requête Q_k sont obtenus par application des opérations sur les ensembles flous, comme décrit dans sa forme disjonctive
   D_{Qk1} = {0.1, 0.4, 0.1, 0.2, 0}
   D_{Qk2} = {0.1, 0, 0.2, 0.3, 0}
   D_{Qk3} = {0.4, 0.2, 0.1, 0.1, 0.2}
   On a lors : D_{QK} = {0.1, 0.2, 0.1, 0.1, 0.2}

3. On obtient ainsi la liste ordonnée des documents pertinents à la requête Q_k :
   $DP(Q_k)$ = { D_2 , D_5 , D_1, D_3, D_4 }

Le principal intérêt de ce modèle est l'application d'opérations algébriques sur les ensembles de documents plutôt qu'une simple maximisation ou minimisation de valeurs d'ensembles [Yates & Neto, 1999].

Lucarella & Morara [Lucarella & Morara, 1991] ont exploité le modèle des ensembles flous pour mettre en œuvre le système FIRST. Les auteurs ont proposé l'utilisation d'un réseau où chaque nœud représente un terme de document ou requête et un lien représente une relation sémantique entre termes. Chaque document D_j est décrit par un ensemble flou comme suit :

$$D_j = \{(t_1, d_{j1}), \ldots, (t_T, d_{jT})\}$$

Une liaison entre concepts est valorisée de manière directe , ou dérivée par transitivité floue :

$$F(t_i , t_k) = Min (F(t_i, t_j) , F(t_j , t_k))$$

Où F : Fonction de valorisation des liens

L'ensemble flou des documents pertinents à une requête Q_k est obtenu comme suit :

1. Pour chaque terme t de Q_k , construire l'ensemble des documents D_t reliés par lien direct ou transitif.

2. Pour chaque couple (t , D_t), associer un degré d'appartenance égal à la valeur minimale de tous les liens qui figurent sur le chemin $t – D_t$

3. Effectuer sur les ensembles D_t, les opérations d'intersection et union selon l'ordre décrit dans l'expression de Q_k relativement aux opérateurs *ET* et *OU* respectivement.

4. Ordonner l'ensemble résultat de la précédente opération selon le degré d'appartenance de chaque document à l'ensemble associé à chaque terme.

Des expérimentations réalisées sur une collection de test italienne comprenant *300* documents, *175* concepts et *15* requêtes ont montré que le modèle offre de meilleures valeurs de rappel relativement au modèle vectoriel.

Chen & Wang [Chen & Wang, 1995] ont étendu ce modèle à l'utilisation d'intervalles de poids admissibles aux concepts, par opposition à l'utilisation de valeurs uniques, ainsi qu'à l'utilisation d'une matrice de concepts. La clôture transitive de cette matrice, soit *T*, est obtenue par multiplications successives de cette même matrice. La valeur de pertinence requête-document est obtenue selon la formule suivante :

$$RSV(Q_k, D_j) = \sum_{ti \in Qk} T(t_{ji}, q_{ki})$$

Où :
$T(x,y)=1-|x-y|$
t_{ji} : Minimum des poids des liens du document D_j au terme t_i

2.2. Le modèle vectoriel

1.2.1. Le modèle de base

Ce modèle préconise la représentation des requêtes utilisateurs et documents sous forme de vecteurs, dans l'espace engendré par les *N* termes d'indexation [Salton, 1968] [Salton, 1989]. De manière formelle, les documents et requêtes sont des vecteurs dans un espace vectoriel de dimension *N* et représenté comme suit :

$$D_j = \begin{bmatrix} d_{j1} \\ d_{j2} \\ \\ d_{jT} \end{bmatrix} \qquad Q_k = \begin{bmatrix} q_{k1} \\ q_{k2} \\ \\ q_{kT} \end{bmatrix}$$

Sous l'angle de ce modèle, le degré de pertinence d'un document relativement à une requête est perçu comme le degré de corrélation entre les vecteurs associés. Ceci nécessite alors la spécification d'une fonction de calcul de similarité entre vecteurs mais également du principe de construction qui se traduit par la fonction de pondération.

1- Fonction de pondération

La fonction de pondération la plus répandue est $d_{ji}=tf_{ji}*idf_i$ [Sparck Jones & Needham, 1972]

Où :

tf_{ji} : Décrit le pouvoir descriptif du terme t_i dans le document D_j

idf_i : Décrit le degré de généralité du terme t_i dans la collection

De nombreuses autres fonctions d'indexation sont basées sur une variante du schéma balancé tf.Idf, on cite notamment :

Formule [Salton & Buckley, 1988]

$$d_{ji}=\left(0.5+\frac{0.5*freq_{ij}}{Max_l\ freq_{jl}}\right)*\log\frac{N}{n_i}$$

Formule [Salton & Allan, 1994]

$$d_{ji}=\frac{freq_{ij}}{\sqrt{\sum_{j=1}^{N} freq_{ji}*\log_2^2(\frac{N}{n_i})}}*\log\frac{N}{n_i}$$

Où :

$freq_{ij}$: Fréquence d'apparition du terme t_i dans le document D_j

Ces mesures supposent que la longueur d'un document n'a pas d'impact sur la mesure de pertinence ; or des expérimentations réalisées par Singhal [Singhal & al, 1997] ont montré que les documents longs ont plus grande probabilité de pertinence parce que contenant plus de termes d'appariement avec la requête.

L'analyse de la corrélation entre probabilité de sélection et probabilité de pertinence a permis la détermination d'une valeur pivot permettant d'ajuster la fonction de pondération par un facteur de normalisation lié à la longueur d'un document. Les auteurs proposent la fonction suivante :

$$d_{ji}=\frac{tf_{ji*}\log\left(\frac{N-n_i+0.5}{n_i+0.5}\right)}{2*(0.25+0.75*\frac{|D_j|}{\overline{|D_j|}}\ )+tf_{ji}}$$

Où :

$|D_j|$: Longueur du document D_j

$\overline{|D_j|}$: Longueur moyenne des documents dans la collection

2- Fonction de similarité

La fonction de similarité permet de mesurer la ressemblance des documents et de la requête. Les types de mesures les plus répandus sont :

Mesure du cosinus [Salton, 1971]

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki} d_{ji}}{\left(\sum_{i=1}^{T} q_{ki}^2\right)^{1/2}\left(\sum_{i=1}^{T} d_{ji}^2\right)^{1/2}}$$

Mesure de Jaccard

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki} d_{ji}}{\sum_{i=1}^{T}(d_{ji})^2+\sum_{i=1}^{T}(q_{ki})^2+\sum_{i=1}^{T} q_{ki} d_{ji}}$$

[Singhal & al, 1995] proposent une fonction de pertinence normalisée par la longueur de document, définie comme suit :

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki} d_{ji}}{(1-s)+s*\frac{\sqrt{\sum_{k=1}^{N} d_{kj}^2}}{|D_j|}}$$

Où :

$|D_j|$: Longueur du document D_j

s : Constante

L'utilisation répandue du modèle vectoriel en recherche d'information est principalement due à l'uniformité de son modèle de représentation requête-document, l'ordre induit par la fonction de similitude ainsi que les possibilités aisées offertes pour ajuster les fonctions de pondération afin d'améliorer les résultats de la recherche.

Toutefois, le modèle présente un inconvénient majeur lié au traitement des termes de documents de manière indépendante. Ceci ne permet pas en effet de reconstituer à travers le processus de recherche, la sémantique associative de termes et ainsi, de la comparer à celle véhiculée par la requête.

2.2.2. Le modèle vectoriel généralisé

Dans le but de pallier au problème d'indépendance des termes, posé par le modèle vectoriel classique, Wong [Wong & al, 1985] a proposé une nouvelle base de référence pour la représentation des documents et requêtes. A cet effet, il définit sur une collection de termes d'indexation $\{t_1, \ldots, t_T\}$:

1. Une base de vecteur binaires, non orthogonaux $\{m_i\}_{i=1..2^T}$

2. Un ensemble de min-termes associé à la base ; chaque min-terme correspond à l'ensemble de documents comprenant les termes d'indexation positionnés à *1* dans le vecteur de base correspondant

3. Une fonction de pondération $g_i(m_j)$ qui donne le poids du terme t_i dans le min-terme m_j, soit w_{ij}

La base ainsi décrite supporte la représentation de la cooccurrence entre termes. Chaque document et requête est décrit dans la nouvelle base comme suit :

$$D_j = \sum_{i=1}^{T} d_{ji} K_i \qquad Q_k = \sum_{i=1}^{T} q_{ki} K_i$$

Où :

$$K_i = \frac{\sum_{\forall r, gi(mr)=1} C_{ir} m_r}{\sqrt{\sum_{\forall r, gi(mr)=1} C_{ir}^2}}$$

Avec :

$$C_{ir} = \sum_{dj / gl(dj)=gl(mr) \forall l} w_{ij}$$

Le calcul de pertinence *RSV(Q,D)* combine alors le poids des documents w_{ij} et facteur de corrélation entre termes C_{ir} . Malgré un accroissement du coût de calcul pour la mesure de similarité, relativement au modèle vectoriel classique, le modèle vectoriel généralisé a l'intérêt d'introduire l'idée de considérer la relation entre termes de manière inhérente au modèle de la fonction de pertinence.

2.2.3 Le modèle LSI

L'objectif fondamental du modèle LSI [Dumais, 1994] est d'aboutir à une représentation conceptuelle des documents où les effets dus à la variation d'usage des termes dans la collection sont nettement atténués. Ainsi, des documents qui partagent des termes cooccurents ont des représentations proches dans l'espace défini par le modèle.

La base mathématique du modèle LSI est la décomposition par valeur singulière SVD de la matrice Terme-Document. La SVD identifie un ensemble utile de vecteurs colonnes de base de cette matrice. Ces vecteurs de base couvrent le même espace de vecteurs associé à la représentation des documents, car ils sont obtenus par rotation (multiplication par une matrice orthogonale) des vecteurs d'origine. On pose :

X : Matrice Terme-Document
T_0 : Matrice avec colonnes orthonormées qui couvre l'espace des colonnes de X
D_0 : Matrice avec colonnes orthonormées qui couvre l'espace des lignes de X
S_0 : Matrice diagonale formée des valeurs singulières qui résultent de la normalisation de T_0 et D_0
On a : $$X = T_0 S_0 D_0^T$$

La propriété de la SVD pour le modèle LSI est qu'en raison du tri des valeurs singulières dans l'ordre décroissant, la meilleure approximation de *X* peut être calculée comme suit :

$$X = \sum_{i=1}^{k} T_{0x1i} S_{0x1i} D_{0xi}^T$$

La matrice T_0 a deux principales propriétés [Oard, 1996] :

1. Chaque paire de représentations de documents obtenue par combinaison linéaire des lignes de T_0 , a la même valeur de degré de similitude que les représentations associées classiques Document-Termes
2. La suppression des composants de faible poids dans un vecteur ligne améliore l'ordre lors du calcul de similitude requête-document.

On passe ainsi d'une représentation de documents à base de termes, vers une représentation à base de concepts, dans un espace de dimension plus réduite.

La méthode LSI a été appliquée dans la collection TREC pour la tâche de croisement de langues [Deerwester & al, 1990]. L'application de la méthode dans un corpus parallèle a permis d'identifier les principaux composants de l'espace vectoriel, associés à chaque langue, produisant ainsi une représentation unifiée de documents écrits dans différentes langues. En utilisant des requêtes en anglais, la méthode sélectionne en début de liste les versions traduites en français dans *92%* des cas.

Outre, l'accroissement de performances dûe à son utilisation [Deerwester & al, 1990] [Dumais, 1994], le modèle LSI présente l'intérêt majeur d'introduire la notion de concept en recherche d'information à travers l'utilisation de la théorie relative à la décomposition par valeurs singulières.

Le modèle LSI probabiliste a été proposé par Hofman [Hofman, 1999]. La particularité de ce modèle relativement au modèle LSI classique, est l'intégration de techniques statistiques pour le traitement des mots polysèmes. A cet effet, le modèle utilise des critères d'optimalité de la décomposition/approximation basée sur une distribution de probabilités.
Les expérimentations réalisées sur les collections MED, CACM, CRANFIELD et CISI prouvent la consistance du modèle et son impact positif sur le rappel du système relativement au modèle LSI classique [Hofman, 1999].

2.3. Le modèle probabiliste

2.3.1. Le modèle de base

Le modèle de recherche probabiliste utilise un modèle mathématique fondé sur la théorie de la probabilité [Robertson & Sparck Jones, 1976]. Le processus de recherche se traduit par calcul de proche en proche, du degré ou probabilité de pertinence d'un document relativement à une requête. Pour ce faire, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

$P(w_{ji} / Pert)$: Probabilité que le terme t_i occure dans le document D_j sachant que ce dernier est pertinent pour la requête

$P(w_{ji} / NonPert)$: Probabilité que le terme t_i de poids d_{ji} occure dans le document D_j sachant que ce dernier n'est pas pertinent pour la requête

Le calcul d'occurrence des termes d'indexation dans les documents est basée sur l'application d'une loi de distribution (type loi de poisson) sur un échantillon représentatif de documents d'apprentissage.

En posant les hypothèses que :

1. La distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents
2. Les variables « document pertinent », « document non pertinent » sont indépendantes,

la fonction de recherche est obtenue en calculant la probabilité de pertinence d'un document *D*, notée *P(Pert/D)* [Risjbergen, 1979] :

$$P(Pert/Dj)=\sum_{i=1}^{T}\log\frac{P(w_{ji}/Pert)}{P(w_{ji}/NonPert)}$$

L'ordre des documents est basé sur l'une des deux méthodes :

1. Considérer seulement les termes présents dans les documents et requêtes

2. Considérer les termes présents et termes absents dans les documents et requêtes

Croft & Harper [Croft & Harper, 1979] intègrent au modèle les mesures de fréquence plutôt, que de considérer seulement la présence ou l'absence des termes. La similitude requête document est calculée comme suit :

$$RSV(Q_k,D_j)=C\sum_{i=1}^{T} q_{ki}d_{ji}+\sum_{i=1}^{T} f_{ji}q_{ki}d_{ji}\log\frac{N-n_i}{n_i}$$

Où :

$$f_{ji}=\frac{tf_{ji}}{\max tf_j}$$

C : Constante

Robertson & Walker [Robertson &Walker, 1994] intègrent la fréquence d'apparition des termes dans la formule de calcul de poids et ce, en se basant sur le modèle de Poisson

$$w_t=\log\frac{\left(p'+(1-p')\left(\frac{\mu}{\lambda}\right)^{tf}e^{j}\right)\left(q'e^{-j}+1-q'\right)}{q'+(1-q')\left(\frac{\mu}{\lambda}\right)^{tf}e^{j}((p'e^{-j})+(1-p'))}$$

Où :

λ : Paramètre de la loi de poisson pour les documents contenant t

μ : Paramètre de la loi de poisson pour les documents ne contenant pas t

j : Différence λ - μ

p' : Probabilité qu'un document contenant t soit pertinent

q' : Probabilité qu'un document contenant t ne soit pas pertinent

La difficulté majeure du modèle réside dans la détermination des valeurs p, p', λ et μ

Dans [Robertson & Walker, 1997], les auteurs montrent que le schéma de pondération de Croft et Harper, peut sous certaines conditions, produire des valeurs négatives. Ils proposent alors une fonction de pertinence d'un document relativement à une requête, basé sur le calcul de chacun des poids des termes d'indexation comme suit :

$$d_{ji}=\frac{k_5}{k_5+R}(k_4+\log\frac{N}{N-n_i})+\frac{R}{k_5+R}\log(\frac{r_i+0.5}{R-r_i+0.5})-\frac{k_6}{k_6+S}\log(\frac{n_i}{N-n_i})-\frac{S}{k_6+S}\log(\frac{s_i+0.5}{S-s_i+0.5})$$

Où :
R : Nombre de documents pertinents
r_i : Nombre de documents pertinents contenant le terme t_i
S : Nombre de documents non pertinents
s_i : Nombre de documents non pertinents contenant le terme t_i
k_4 , k_5 , k_6 : Constantes

De manière générale, le modèle probabiliste présente l'intérêt d'unifier les représentations des documents et concepts. Cependant, le modèle repose sur des hypothèses d'indépendance des variables pertinence non toujours vérifiées, ce qui entâche les mesures de similitude d'imprécision.
En outre, le modèle ne prend pas en compte les liens de dépendance entre termes, et engendre des calculs de probabilité conditionnelles complexes.

2.3.2. Le modèle de réseau infèrentiel bayésien

Un réseau bayésien est un graphe direct acyclique où les nœuds représentent des variables aléatoires et les arcs des relations causales entre nœuds. Ces derniers sont pondérés par des valeurs de probabilités conditionnelles.
Le travail original en recherche d'information, et basé sur le modèle des réseaux bayésiens, est développé par Turtle [Turtle & Croft, 1991].
Dans l'espace défini par les termes d'indexation, on définit :
- T variables aléatoires binaires $t_1, \ldots, t_T$ associés aux termes d'indexation
- D_j : Variable aléatoire associée à un document
- Q_k : Variable aléatoire associée à une requête

On calcule alors la mesure de pertinence de Q_k relativement à D_j en traitant les probabilités conditionnelles de Bayes selon la formule :

$$RSV(Q_k, D_j) = 1 - P(Q_k \wedge D_j)$$

Où :

$$P(Q_k \wedge D_j) = \sum_{i=1}^{T} P(Q_k/t_i) * \left(\prod_{ti \in Dj} P(t_i/D_j) * \prod_{ti \notin Dj} P(\overline{t_i}/D_j) \right) * P(D_j)$$

Avec :
$P(Q_k/t_i)$: Probabilité que le terme t_i appartienne à un document pertinent de Q_k
$P(t_i/ D_j)$: Probabilité que le terme t_i appartienne au document D_j sachant qu'il est pertinent
$P(\overline{t_i} /D_j) = 1- P(t_i/D_j)$
$P(D_j)$: Probabilité d'observer D_j

Les probabilités conditionnelles de chaque nœud sont calculées par propagation des liens de corrélation entre eux.
Le modèle présente l'intérêt de considérer la dépendance entre termes mais engendre une complexité de calcul importante.
D'autres modèles sont décrits dans [Savoy & Dubois, 1991] et [Haines & Croft, 1993].

2.4. Le modèle connexioniste

Les SRI basés sur l'approche connexioniste utilisent les fondements des réseaux de neurones tant pour la modélisation des unités textuelles que pour la mise en oeuvre du processus de recherche d'information.
Après un bref aperçu des concepts clés du modèle, nous décrivons avec plus de détails les modèles connexionistes pour la recherche d'information.

2.4.1. Le modèle de base

Le fonctionnement d'un neurone formel est inspiré du fonctionnement connu d'un neurone biologique. Un neurone biologique est le processeur élémentaire de traitement de l'information; il est composé d'un [Bourret & Samuelides, 1991] :
- Réseau convergent d'entrée : **dendrites**
- Elément de traitement de l'information : **corps cellulaire**
- Réseau divergent : **axone**

La connexion de l'axone d'un neurone aux dendrites d'un autre neurone est appelé **synapse**. Les neurones sont connectés pour former un réseau. La transmission des signaux d'activation est effectuée par propagation depuis les entrées jusqu'aux sorties.
Par analogie, un neurone formel reçoit des entrées des neurones auxquels il est connecté en tant que successeur; le neurone calcule une somme pondérée des potentiels d'action de ses entrées puis calcule une valeur de sortie correspondant à son niveau d'activation.
Si le niveau dépasse un seuil, le neurone est activé et transmet une réponse; si cela n'est pas le cas, le neurone est dit inhibé et ne transmet aucun signal.
Un neurone est caractérisé par :
1- La fonction d'entrée totale $E = f(e_1, e_2, ..., e_n)$ qui peut être :

- Linéaire $E = \sum_{i=1}^{n} w_{ij} e_j$

- Affine $E = \sum_{j=1}^{n} w_{ij} e_j - a$

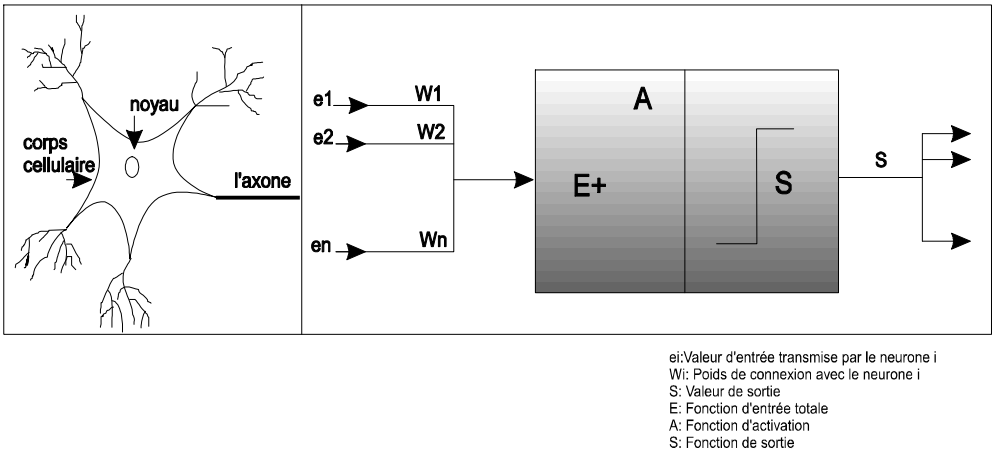


Figure 1. 2 : Modèle de neurone formel et modèle de neurone biologique

2- La fonction d'activation $A= A(E)$ qui peut être

- La fonction binaire Heaviside $A = 0$ Si $E \leq 0$, $A= 1$ Si $E > 0$
- La fonction signe $A = -1$ Si $E \leq 0$, $A=1$ Si $E > 0$
- La fonction sigmoïde $A = a * \frac{ekx-1}{ekx+1}$ $A \in [-a \ a]$

3- La fonction de sortie $S= S(A)$ qui est généralement la fonction identité

Un réseau de neurones est caractérisé par deux propriéts fondamentales : dynamique des états et dynamique des connexions

- Dynamique des états

La dynamique des états correspond à l'évolution des états des différents neurones qui composent le réseau. Cette évolution est modulée d'une part par l'architecture du réseau et d'autre part, par la structure des poids des connexions et nature de la fonction d'activation

- Dynamique des connexions

La dynamique des connexions correspond à l'évolution des poids des connexions en cours du temps. Ceci traduit **l'apprentissage** du réseau par changement de son comportement d'après les résultats de son expérience passée. Comme pour l'activation des noeuds, les poids sont modifiés en parallèle mais varient généralement plus lentement que les niveaux d'activation.

De nombreuses approches de l'apprentissage ont été proposées; on y présente généralement des règles de modification de poids telles que la règle de Hebb [Hebb, 1949], Windrow-Hoff et rétropagation du gradient [Bourret & Samuelides, 1991].

2.4.2. Le modèle à couches

Les modèles connexionistes à couches sont d'utilisation répandue en recherche d'information [Wilkinson & Hingston, 1991] [Boughanem, 1992] [Kwok, 1995]. Le réseau est construit à partir des représentations initiales de documents et informations descriptives associées (termes, auteurs, mots clés ...). Le mécanisme de recherche d'information est fondé sur le principe d'activation de signaux depuis les neurones descriptifs de la requête, et propagation de l'activation à travers les connexions du réseau. La pertinence des documents est alors mesurée grâce à leur niveau d'activation.

Le réseau construit dans Kwok [Kwok, 1995] utilise trois couches interconnectées dans le sens requête - termes - documents. Les connexions sont bidirectionnelles et de poids assymétriques.

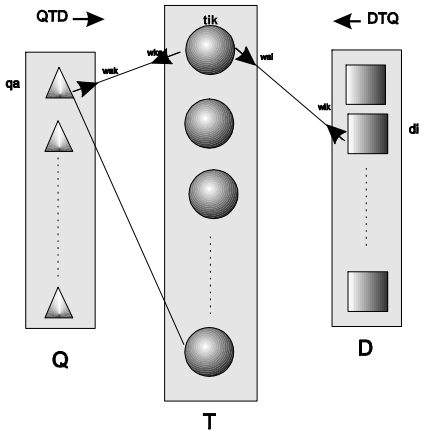


Figure 1.3 : Le modèle de réseau Kwok [Kwok, 1995]

L'approche de Kwok est fondée sur l'idée que les requêtes et documents sont similaires, en ce sens qu'ils sont tous deux représentants de concepts. Sur cette base, il reprend des éléments du modèle probabiliste pour classer les neurones documents selon la probabilité $W_i = W_{j/Q} + W_{j/D}$

Où :

$W_{j/Q}$: Probabilité pour que la requête Q soit pertinente pour le document D_j

$W_{j/D}$: Probabilité pour que le document D_j soit pertinent pour la requête Q

La valeur de $W_{j/Q}$ est obtenue dans le sens **DTQ**, par simulation de la pertinence du document D_j ; on injecte à l'entrée du neurone d_i un signal de *1*, puis on évalue la valeur d'activation du neurone q_a :

$$W_{qa} = \sum_{k=1}^{m} W_{ka} S_{ik} \qquad W_{jQ} = \sum_{k=1}^{T} W_{j/Q} d_{jk}$$

Ce processus est itéré pour chaque neurone document D_i.

D'autre part, on effectue la propagation dans le sens **QTD** par injection d'un signal de *1* à l'entrée du neurone q_a . On évalue alors les valeurs de sortie aux neurones documents $W_{j/D}$

$$W_{j/D} = \sum_{k=1}^{T} W_{kj} S_{jk} \qquad S_{jk} = \sum_{k=1}^{T} W_{ak} q_{jk}$$

Où :

$W_{jk} = d_{jk}/L_j$

$W_{ak} = q_{ak}/L_a$

$q_{ak} = \text{Log}(r_{ak}/(1-r_{ak})) + \text{Log}((1-S_{ak})/S_{ak})$

$d_{jk} = \text{Log}(r_{jk}/(1-r_{jk})) + \text{Log}((1-S_{jk})/S_{jk})$

Avec

L_j : Nombre de termes du documents d_j

q_{ak} : Fréquence du terme du terme t_k dans la requête Q_a

L_a : Nombre de termes de la requête Q_a

$r_{ik} = r_{ak} = 1/40$

$S_{jk} = (F_k - d_{ik})/(T - L_i)$

$S_{ak} = F_k / T$

F_k : Fréquence du terme t_k dans la collection

En outre, le modèle est doté de la capacité d'apprentissage par modification des poids de connexions suite à la perception des jugements de pertinence de l'utilisateur.

2.5. Synthèse

La définition formelle d'un modèle de recherche d'information ainsi que l'étude des principaux modèles proposés dans la littérature, nous amène à mettre en évidence les caractéristiques qualitatives suivantes :

1. pouvoir de représentation des unités d'information : termes isolés, termes pondérés, groupes de termes, concepts …
2. capacité de modélisation des liens sémantiques termes-documents, termes-termes,
3. pertinence du principe d'appariement requête – document.

Il en ressort que chacun de ces modèles s'investit à couvrir l'une ou l'autre de ces caractéristiques afin d'aplanir les difficultés inhérentes à la recherche d'information.
Du point de vue de la qualité de représentation et capacité de modélisation de l'information, le modèle booléen présente une remarquable insuffisance. Le modèle étendu et celui des ensembles flous y apportent des améliorations qui portent cependant sur le principe d'appariement.

Le modèle probabiliste modélise quant à lui l'information, en se basant sur des collections d'apprentissage ainsi que sur des hypothèses d'indépendance peu réalistes, ce qui diminue de la qualité de représentation . Bien que le processus de recherche d'information soit relativement coûteux, les travaux montrent que c'est l'un des modèles les plus performants.
Le modèle vectoriel LSI ainsi que le modèle connexioniste offrent des atouts intéressants de conceptualisation de par respectivement, le principe de décomposition de la matrice de représentation termes-documents et principe d'activation par propagation lors de la recherche.
Des modèles s'intéressant d'avantage à la représentation sémantique des documents, sont proposés dans [Puget, 1992] et [Genest, 1999]. L'idée fondamentale de ces modèles est l'utilisation de graphes conceptuels pour l'indexation automatique des documents. La recherche d'information y est basée sur l'application de règles de transformation de graphes.
Concernant la formalisation de la fonction de pertinence, on note que cette dernière est communément dérivée du support théorique du modèle de représentation et généralement paramétrée en fonction de l'environnement d'exploitation.
Dans ce contexte, Bartell [Bartell & al, 1994] propose la combinaison de fonctions de pertinence liées à différents algorithmes de recherche ; l'idée est alors d'exploiter différentes sources d'interprétation de la pertinence permettant d'approcher d'avantage

la réelle probabilité de pertinence. L'évaluation de l'utilité conjointe des méthodes combinées a permis d'ajuster automatiquement les paramètres de la fonction de pertinence globale.
Les expérimentations réalisées sur la collection Encyclopédie Britannique ont révélé un accroissement des performances de la fonction pertinence combinée de *12%* à *14%* que les fonctions de pertinence isolées.
D'autres travaux de Bartell [Bartell & al, 1998] proposent une méthode automatique pour l'ajustement de paramètres d'une fonction pertinence. La méthode est basée sur l'utilisation du point d'aliénation de Guttman et du gradient conjugué afin d'optimiser les paramètres de la fonction tout en préservant l'ordre des performances réalisées sur les collections d'apprentissage.

Enfin, les travaux en recherche d'information ont mis en évidence l'impact considérable de la rigueur du modèle et support théorique associé, sur les performances de recherche. Cependant, l'efficacité d'un modèle reste limitée face aux nombreuses difficultés de recherche d'information liées notamment à l'ambiguïté du langage.
A cet effet, de nombreux travaux ont investi l'idée d'intégrer à un modèle de base, des stratégies de recherche qui traduisent des croyances et heuristiques quant à la description des unités d'informations et de la sémantique globale véhiculée par une recherche.

3. Stratégies de recherche

Une stratégie de recherche représente un ensemble d'heuristiques et algorithmes permettant d'améliorer les performances du processus de recherche d'information. Une stratégie de recherche est généralement définie dans le cadre de différents modèles moyennant quelques adaptations.
Pour notre part, nous nous intéressons principalement aux stratégies de :

- Reformulation de requête : mécanisme adaptatif de modification de requête qui a des conséquences très avantageuses sur les résultats de recherche.
  Cette modification de requête en poids et/ou structure peut être basée sur diverses techniques : utilisation du thesaurus, utilisation des résultats de recherche locale, injection de pertinence de l'utilisateur etc…

- Recherche de passage de document : technique qui consiste à limiter les disparités de pertinence dans un document par définition d'une méthode de partitionnement ; des problèmes liés à la délimitation de passages documentaires et localisation de l'information pertinente y sont alors posés.

3.1. La reformulation de requête

La reformulation de requête est proposée comme une méthode élaborée pour la recherche d'information s'inscrivant dans la voie de conception des SRI adaptatifs aux besoins des utilisateurs. C'est un processus permettant de générer une requête plus adéquate à la recherche d'information dans l'environnement du SRI, que celle initialement formulée par l'utilisateur. Son principe est de modifier la requête de l'utilisateur par ajout de termes significatifs et/ou réestimation de leur poids.
La dimension de l'espace de recherche étant élevée, la difficulté fondamentale de la reformulation de requête est alors la définition de l'approche à adopter en vue de réduire l'espace de recherche par la détermination de [Efthimiadis, 1996] :
1. critères de choix des termes de l'expansion,
2. règles de calcul des poids des nouveaux termes,
3. hypothèse de base quant aux liens entre termes et documents.

3.1.1. Les outils de base

Les techniques de reformulation de requête ont généralement recours à l'utilisation de techniques de classification et du thesaurus.

3.1.1.1. La classification

La classification découpe l'espace des documents en sous-espaces homogènes appelés classes [Salton & MacGill, 1983] [Risjbergen, 1979] [Aboud, 1990]. Celles-ci sont constituées à partir de critères discriminatoires restreignant l'espace de recherche à un échantillon plus pertinent; les documents d'une même classe sont caractérisés par la même valeur du critère.
Plusieurs stratégies de classification ont été proposées; nous présentons brièvement les techniques basées sur les attracteurs de groupes, similarité de documents et pertinence par rapport à une requête.

1- Classification par choix d'attracteurs de groupes

Le principe de classification consiste, dans ce cas, à choisir un document attracteur pour chaque groupe de documents. Un document est rattaché au groupe dont l'attracteur est le plus similaire. Dans [Blosseville & al, 1992], les pôles attracteurs sont déterminés préalablement par échantillonnage de documents classés par un expert. L'analyse des documents est basée sur un modèle mathématique polynomial portant sur la distribution des termes dans les documents. Le calcul de probabilité $P(D_i , C_k)$ pour que le document D_i appartienne à la casse C_k est effectué selon la formule suivante :

$$P(D_j,C_k)=N_k\prod_{i=1}^{m}\frac{d_{ji}*N_k(i)+(1-d_{ji})*(N_k-N_k(i))}{N_k}$$

Où :
m : Nombre total de termes qui occurrent dans l'échantillon
C_k : kième classe
N_k : Nombre de documents de la classe C_k appartenant à l'échantillon
N_k (i) : Nombre de documents de la classe C_k appartenant à l'échantillon et contenant le terme t_i

La même approche a été étudiée par Lewis [Lewis & Ringuette, 1994] en utilisant les arbres de décision. Les documents attracteurs peuvent également être choisis de manière aléatoire [Razouk, 1990] ou selon des algorithmes basés sur le contenu de la collection [Can & Ozkarahan, 1990].

2- Classification hiérarchique

Cette technique est basée sur le calcul d'une matrice de similitude entre documents [Salton & MacGill, 1983]. Dans la stratégie de classification avec un seul passage, on construit, à partir de la matrice de similitude, un graphe de classement où les nœuds représentent des documents. Deux sommets sont reliés par une arête si le degré de ressemblance entre documents correspondants est supérieur à un seuil établi. La décomposition du graphe obtenu en classes, utilise des techniques liées à la théorie des graphes; on citera notamment les définitions suivantes [Aboud, 1990] :

- Une classe est une composante connexe du graphe. Une classe forme un groupe de sommets dans lequel chaque sommet est connecté à tous les autres.
- Une classe est une étoile du graphe. Une étoile est un ensemble de sommets tel qu'il existe un sommet central connecté à tous les autres.

Le nombre de classes est ainsi dépendant du seuillage appliqué.

La stratégie de classement séquentiel suppose l'existence d'un critère de classification pour le critère à utiliser. On définit ainsi une hiérarchie de classes définie chacune par un descripteur constitué de l'ensemble des termes d'indexation des documents qu'elle contient. Le classement d'un document dans une classe s'effectue par calcul du degré de ressemblance au centroïde correspondant. Le centroïde d'une classe est l'ensemble des termes représentatifs de ses documents.
Des travaux plus récents [Shutze & Silverstein, 1997] adoptent ce type de classification en utilisant une représentation des documents basée sur le modèle LSI ; cependant aucun accroissement de performances significatif n'a été atteint.

3- Classification basée sur la pertinence des documents

Une méthode de classification adaptative a été introduite dans [Yu & Chen, 1985]. A l'origine, on associe à chaque document une coordonnée aléatoire sur un axe réel; les coordonnées des documents pertinents pour une requête sont ensuite modifiées en vue de les rapprocher les uns des autres. Dans le but d'éviter la concentration de documents, le centroïde de ces documents est éloigné de celui de la collection.

Raghavan & Deogun [Raghavan & Deogun, 1986] ont également développé une méthode de description de classes basée sur la description des documents pertinents aux requêtes. L'originalité de leur approche est de définir les classes de documents copertinents par fusions progressives de documents jugés pertinents mais éloignés des requêtes en cours. Les auteurs ont mis au point des heuristiques basées sur des calculs statistiques de distribution des termes dans la collection et dans les classes afin de maintenir un équilibre entre leurs tailles.

3.1.1.2. Le thesaurus

Un thesaurus est un outil permettant de représenter la proximité ou voisinage sémantique entre termes de la collection. Nous synthétisons ci dessous les principales approches adoptées pour sa construction .

1- Thesaurus manuel

Consiste à définir intéractivement divers liens linguistiques entre mots : synonymes, hypernyms, hyponyms, polysèmes etc… [Wang & al, 1985][Roget, 1988].

Le thesaurus de Roget [Roget, 1988] est organisé en catégories de mots; chaque catégorie correspond à un sens bien défini par les indexeurs. La polysémie y est traduite par la possibilité d'associer à chaque mot, *n* catégories différentes représentant ses différents sens.

Ce mode de construction est généralement adapté à des collections de petites tailles, à domaine spécifique [Suy&Lang, 1994].

2- Thesaurus automatique

Consiste à déterminer une hypothèse de liaison sémantique et l'utiliser pour la génération automatique du thesaurus. Cette liaison est généralement basée sur la cooccurence, contexte des termes ou leur combinaison.

- ***Thesaurus basé sur la cooccurrence***

  Consiste généralement à combiner une mesure seuillée de cooccurrence entre descripteurs des termes dans la collection et un algorithme de classification.

  Dans [Chen & Ng, 1995], la mesure utilisée pour le calcul de la cooccurrence est la suivante :

$$SC(t_i,t_j)=\left(\frac{\sum_{k=1}^{N}\min(tf_{ik},tf_{jk})\log(\frac{N}{df_{ij}}*p_j)}{\sum_{k=1}^{N}d_{ik}}\right)*W_j$$

Où :
df_{ij} : Nombre cooccurrences entre les termes t_i et t_j
p_j : Longueur du descripteur du terme t_j
d_{ik} : Poids du terme i dans le document D_k
f_{ik} : Fréquence du terme i dans le document k

Avec :

$$W_j=\frac{\log(\frac{N}{df_j})}{\log N}$$

Les travaux de Peat & Willet [Peat & Willet, 1991] et de Schutze & Pederson [Schutze & Pederson, 1997] ont cependant montré les limites d'utilisation de ce type de thesaurus. Le problème fondamental posé est que l'usage de la cooccurrence ne permet pas d'identifier les termes caractéristiques de la recherche en cours. En effet, la valeur de la cooccurrence est généralement élevée pour des termes génériques, de fréquences d'apparition élevées dans la collection, ce qui ne permet pas d'améliorer les valeurs de rappel/précision.

- Thesaurus basé sur le contexte

L'idée de base est de distinguer les polysèmes par définition de contextes d'utilisation des termes dans la collection. A chaque terme est ainsi associé plusieurs vecteurs contexte dépendants de leur usage dans les documents.
Dans [Gauch & Wang, 1996], les auteurs définissent le contexte d'un terme t_l à une position voisine i , $VC_i = (W_{i1}, \ldots, W_{i,200})$ comme formé des *200* termes à plus grande valeur de cooccurrence avec le terme t à la position i,
Où :

$$W_{lk}=\log\left(\frac{N*df_{lk}}{tf_l*tf_k}+1\right)$$

Avec :
df_{lk} : Fréquence de cooccurrence de contexte du terme t_l avec le terme t_k
tf_l : Nombre total d'occurrences du terme t_l dans la collection
tf_k : Nombre total d'occurrences du terme t_k dans la collection

Le vecteur descripteur d'un terme t_l est composé de vecteurs contextes situés aux *3* positions précédentes et *3* positions successives :

$$t_l = < VC_{-1}\ VC_{-2}\ VC_{-3}\ VC_1\ VC_2\ VC_3 >$$

L'utilisation d'une description contextuelle des termes est également proposée dans [Jing & Tzoukerman, 1999]. Dans l'approche présentée, un sens dominant est associé à chaque terme t_i $(t_{i-P+1}, \ldots, t_1, \ldots, t_P)$ dans un document, et représenté comme suit :
Pour chaque occurrence du terme t_i :

1. créer un contexte local constitué par *P* termes à droite et *P* termes à gauche,

2. calculer pour chaque terme t_v voisin de t_i dans le contexte local, le poids $Frequence(t_v, t_i)/Frequence(t_i)$,

3. constituer le vecteur normalisé des dix termes de plus grands poids.

Des travaux présentés ci dessous montrent l'intérêt de la représentation contextuelle des termes

3.1.2. La reformulation automatique

La reformulation automatique de requête induit un processus d'expansion et/ou repondération de la requête initiale en utilisant des critères de choix définis sans intervention de l'utilisateur. Ce type de reformulation peut être défini dans un contexte global, basé sur le thésaurus, ou alors local, basé sur les résultats de la recherche en cours.

3.1.2.1. Reformulation basée sur le contexte global

Cette stratégie de recherche fait référence à l'exploitation d'informations préalablement établies dans la collection, et non dépendantes de la recherche en cours, en vue de réaliser la reformulation. Ceci fait alors appel essentiellement à l'utilisation de thesaurus.

1- Utilisation d'un thesaurus manuel

Le principe fondamental est d'ajouter à la requête initiale, les termes voisins définis dans le thesaurus et sélectionnés par l'application d'un seuil et d'un algorithme de choix.
[Suy & Lang, 1994] proposent une expansion de requête basée sur l'utilisation du thesaurus manuel de Roget. La recherche d'information est effectuée selon les principales étapes suivantes :

1- Expansion de requête en utilisant les liens sémantiques prédéfinis dans le thesaurus. Plus précisément, la requête utilisateur Q_k est étendue avec les termes de l'ensemble défini comme suit :

$$C^{+} = \bigcup_{ti \in Qk} C_i^{+}$$

où :

$C_i^{+} = \{ t \in C_i / (t_{ij} \neq 0) \wedge (C_j = C_i) \}$, C_i : Catégorie de Roget du terme t_i

En fait, on intègre à la requête utilisateur l'ensemble des termes qui traduisent la couverture sémantique de chacun de ses termes

2- Calcul de pertinence des documents selon un mécanisme d'activation propagation basé sur le modèle connexioniste

Des expérimentations réalisées sur les collections standards CACM et CISI ont révélé que l'expansion de requête a permis d'obtenir de meilleurs résultats que ceux fournis par le modèle de réseau bayésien [Suy & Lang, 1994].

2- Utilisation d'un thesaurus automatique basé sur la similarité

Qiu & Frei [Qiu & Frei, 1993] proposent une expansion de requête basée sur un thesaurus construit de façon automatique, modélisant des liens de similarité entre termes. A chaque terme t_i, on associe un descripteur vectoriel $\vec{t_i}(d_{1i} \ldots, d_{Ni})$

Où :

$$d_{ji} = \frac{(0.5 + 0.5 * \frac{f_{ji}}{Max f_{ji}})}{\sqrt{\sum_{j=1}^{N} (0.5 + 0.5 * \frac{f_{jl}}{Max_l(f_{jl})})^2 itf_j^2}}$$

Avec

f_{ji} : Fréquence du terme t_i dans le document D_j

itf_j : Fréquence inverse du document D_j

La relation entre termes est représentée par un facteur de corrélation calculé comme suit :

$$C_{UV} = \vec{t_U}\,\vec{t_V} = \sum_{j=1}^{N} d_{uj} d_{vj}$$

L'expansion de requête est alors effectuée selon les étapes suivantes :

1. Représenter sous forme vectorielle, la requête initiale $\vec{Q_k} = \sum_{ti \in Qk} q_{ki} \vec{t_i}$
2. Utiliser le thesaurus pour calculer $Sim(\vec{Q_k}, \vec{t_j}) = Q_k t_j = \sum_{ti \in Qk} q_{ki} C_{ij}$
3. Ajouter à la requête les *r* top termes t_s sélectionnés par $Sim\ (Q_k, k_s)$. A chaque terme ajouté t_a, on utilise un poids donné par :

$$q_{ai'} = \frac{Sim(Q_k, t_a)}{\sum_{ti \in Qk} q_{ki}}$$

Les expérimentations réalisées sur *3* collections de test standards montrent un accroissement de l'ordre de *20%* relativement à la baseline [Yates & Neto, 1999].
Le modèle vectoriel généralisé est considéré comme une généralisation de cette technique, en ce sens que la principale différence est l'utilisation restreinte des *r* top termes pour l'expansion [Yates & Neto, 1999].

Dans [Schutze & Pedersen, 1997], les auteurs proposent l'utilisation de termes d'expansion issus d'une classification basée sur une décomposition par valeur singulière (SVD) de la matrice de cooccurrences entre termes. Cette matrice est le résultat de l'application de l'algorithme du backchot qui est une variante de l'algorithme de classification hiérarchique. Chaque document est alors représenté par la somme des vecteurs contexte déterminés par la SVD et utilisée pour le calcul de la similitude avec la requête.

3- Utilisation d'un thesaurus basé sur le contexte

L'idée essentielle est d'étendre une requête par intégration de termes de même contexte que ceux qui la composent. Dans ce cadre, les approches différent principalement relativement au principe adopté pour la définition d'un contexte de mot.

Une expansion de requête basée sur l'utilisation d'un thesaurus organisé en classes définissant des contextes, a été proposée par [Carolyn & Yang, 1992]. Les travaux présentés proposent l'application d'un algorithme de classification pour l'organisation de documents ; les termes d'expansion sont issus d'un thesaurus constitué des termes de faible fréquence associés à chaque requête.
La sélection des termes à ajouter, est basée sur le poids de la classe calculé par la formule : $W_c = \frac{w_{tc}}{|C|} * 0.5$

Où
| C | : Cardinal de la classe C

W_{tc} : Poids du terme t dans la classe C, calculé selon la formule $W_{tc}=\frac{\sum_{i=1}^{|C|} w_{ic}}{|C|}$

Avec w_{ic} : Poids du terme t_i dans la classe C

Les expérimentations réalisées dans des collections standards montrent l'intérêt de cette stratégie d'expansion. Cependant, cette dernière donne des résultats très dépendants des paramètres de l'algorithme de classification : nombre de classes, taille min d'une classe… Ces paramètres sont en outre très variables en fonction des collections interrogées [Yates & Neto, 1999].

L'expansion de requête proposée par [Gauch & Wang, 1996] est effectuée comme suit :

1. Construction préalable du thesaurus de contexte de termes (Cf. paragraphe 3.1.1.2)

2. Calcul de similarités vecteur contexte – requête

3. Ajout des n top termes dont la valeur de similarité avec la requête est supérieure à un seuil déterminé

Cependant, les résultats d'expérimentations n'ont pas montré un accroissement significatif de la précision lié à l'utilisation de la requête étendue

L'approche proposée dans [Jing & Tzoukermann, 1999] est basée sur la distance contextuelle et la proximité morphologique entre termes.
La principale motivation pour l'intégration de ces deux aspects dans le modèle, est que la corrélation basée sur la morphologie d'un mot fait augmenter le rappel alors que la corrélation basée sur le sens fait augmenter la précision.

L'algorithme de recherche est effectué en deux étapes :

Etape 1 : ***Préambule à la recherche***

1. Construction de la base documentaire en utilisant un analyseur morphologique

2. Construction du vecteur contexte de chaque document constitué par les vecteurs contexte de chacun de ses termes (Cf. 3.1.1.2)

$$VC_i (t_1 (W_{i1}), ..., t_N (W_{iN}))$$

Où :
W_{ij} : poids du terme t_i dans le document D_j

3. Calcul des cooccurrences locales, dans la collection, pour toute paire de mots :

$$R(t_1,t_2)=\frac{IDF(t_1,t_2)}{IDF(t_1)+IDF(t_2)-IDF(t_1,t_2)}$$

Où : $IDF(t_i, t_j)$: Fréquence inverse de la cooccurrence des termes t_i et t_j dans la collection

Etape 2 : *Recherche*

Pour chaque requête et chaque document :

1. Calculer la distance contextuelle moyenne entre chaque terme de la requête et ses variantes morphologiques dans le document selon la formule :

$$Dist(VC_1,VC_2)=\sum_{i=1}^{|VC|} R(t_i,B_m(i))*W_{ti}W_{2m(i)}$$

Où :
|VC| : taille des vecteurs contexte

$B_{m(i)}$: terme le plus cooccurent avec t_i, ie $B_{m(i)}$ / $R(t_i, B_{m(i)})= Max_{j=1}^{T} R(t_i,t_j)$

2. Si (distance contextuelle moyenne est supérieure à un seuil) ou (taille du vecteur contexte est inférieur à un seuil) Alors
   Considérer les deux termes équivalents *(Expansion)*
   Sinon
   Considérer les deux termes différents

3. Calculer la similarité requête – document

Cet algorithme permet ainsi de corréler des termes sur la base de leur morphologie mais aussi de leur sens, ceci par opposition aux algorithmes d'indexation classiques qui tronquent les termes morphologiquement reliés au même mot même s'ils ne véhiculent pas le même sens.
Des expérimentations réalisées sur les collections AP88 (Associated Press) et AP90 de TREC4 montrent un accroissement de la précision moyenne de *8,6%*.

3- Reformulation basée sur une combinaison de thesaurus

Intéressé par l'aspect sémantique couvert par chacun des types de thesaurus, Mandala [Mandala & al, 1999] propose une méthode d'expansion de requête en utilisant un thesaurus combiné de manière à conjuguer leurs caractéristiques.
Plus précisément, trois types de thesaurus sont utilisés dans l'approche :

- Thesaurus manuel

Un terme y est représenté selon différentes taxonomies. On y associe un graphe sémantique où les nœuds représentent les termes et liens des relations de synonymie entre termes inter-taxonomies et intra-taxonomies.

La similitude entre deux mots est définie comme le chemin le plus court dans le graphe :

$$Sim(t_i,t_j)=Max_{pathP}(-\log\frac{N_P}{2*D})$$

Où :

Np : Nombre de nœuds entre t_i et t_j selon le chemin P

D : Hauteur maximale de la taxonomie

- Thesaurus basé sur la cooccurrence

On évalue la cooccurrence de pseudo-phrases de taille fixe *T* dans des blocs adjacents de documents

$$Sim(b_i,b_j)=\frac{\sum_{l=1}^{T} w_{lbi}w_{lbj}}{\sqrt{\sum_{l=1}^{T} w_{lbi}^2 \sum_{l=1}^{T} w_{lbj}^2}}$$

Où :

b_i : ième bloc

W_{tbi} : Fréquence du terme t dans le bloc b_i

- Thesaurus basé sur le contexte linguistique

Les mots sont classés par contexte grammatical verbe, sujet, adjectif etc… puis on calcule la cooccurrence relative entre deux termes, dans chaque classe, selon une formule appropriée.

Exemple : classe Adjectif

$$I(a_i,adj,n_j)=\log\frac{f_{adj}(a_i,n_j)/N_{adj}}{(f_{adj}(n_j)/N_{adj})*(f(a_i)/N_{adj})}$$

Où :

$I(a_i , adj, n_j)$: Valeur de cooccurence de a_i en qualité d'adjectif du nom n_j

$f(a_i,n_j)$: Fréquence d'occurrence de a_i en qualité d'adjectif de n_j

$f_{adj}(n_j)$: Fréquence d'occurrence de n_j en qualité d'objet de tout adjectif

N_{adj} : Nombre total d'adjectifs dans la collection

$f(a_i)$: Fréquence de l'adjectif a_i dans la collection

On calcule la similitude entre deux termes selon la formule :

$$Sim(t_i,t_j)=\frac{\sum_{(c,t)\in T(ti)\bigcap T(tj)}(I(t_i,c,t)+I(t_j,c,t))}{\sum_{(c,t)\in T(ti)} I(t_i,c,t)+\sum_{(c,t)\in T(tj)} I(t_j,c,t)}$$

Où :

c : Classe grammaticale (adjectif, nom, verbe …)

T(t) = { (c,t') / I(t,c,t') > 0 }

Le principe de recherche / expansion est alors le suivant :

1. Représenter la requête sous forme vectorielle $Q(q_{k1}, \ldots, q_{kt})$

Où :

$$q_{ki}=\frac{(\log(tf_{ki})+1.0)*\log(N/n_i)}{\sqrt{\sum_{j=1}^{T}[\log(tf_{kj}+1.0)*\log(N/n_j)]^2}}$$

tf_{ki} : Fréquence d'occurrence du terme t_i dans la requête Q_k

2. Calculer la similitude entre termes de la requête et termes du thesaurus combiné comme suit :

$$Sim(Q_k,t_i)=\sum_{tj\in Qk} q_{ki}\overline{Sim}(t_i,t_j)$$

Où :

$\overline{Sim}(t_i,t_j)$: Similitude moyenne entre les termes t_i et t_j relativement au 3 types de thesaurus

Avec :

$\overline{Sim}(t_i,t_j)=\frac{\sum Sim(t_i,t_j)}{Typethesaurus}$, $Sim(t_i, t_j)$ est normalisée comme suit :

$$Sim(t_i,t_j)=\frac{Sim(t_i,t_j)_{old}-Sim(t_i,t_j)_{\min}}{Sim(t_i,t_j)_{\max}-Sim(t_i,t_j)_{\min}}$$

Avec :

$Sim_{old}(t_i,t_j)$: Valeur de similititude calculée selon la formule non normalisée associée au type de thesaurus

$Sim_{min}(t_i,t_{j)}$: Valeur de similititude minimale calculée selon la formule non normalisée associée au type de thesaurus

$Sim_{max}(t_i,t_{j)}$: Valeur de similititude maximale calculée selon la formule non normalisée associée au type de thesaurus

3. Ordonner les termes par valeurs croissantes de $Sim\ (Q_k\ ,\ t_j)$. Retenir les *r* top termes pour l'expansion de requête avec un poids calculé comme suit :

$$q_{ki}=\frac{Sim(Q_k,t_i)}{\sum_{tj\in Qk} q_{kj}}$$

Ainsi, le poids d'un nouveau terme dépend de l'ensemble des poids de la requête mais également de la similitude relativement à chacun des types de thesaurus.
Les expérimentations réalisées sur la base TREC7 ont montré l'intérêt de cette stratégie relativement à l'utilisation d'un seul type de thesaurus [Mandala &al, 1997].

3.1.2.2. Reformulation basée sur le contexte local

Dans le cas de cette stratégie de recherche plus connue sous l'expression anglaise « adhoc feedback », les informations utilisées pour la reformulation de requête dépendent en grande partie de la recherche en cours : documents retrouvés, termes et poids associés .
A l'origine, les travaux relatifs à l'utilisation de cette stratégie consistent essentiellement en l'application de techniques de classification de termes issus des *n* tops documents retrouvés [Attar & Fraenkel, 1977]. Actuellement, de nouvelles techniques sont mises en œuvre en vue d'analyser le contexte local de la recherche et de l'exploiter pour l'expansion de requête.

L'approche proposée par Xu & Croft [Xu & Croft, 1996] combine les atouts de l'analyse globale et analyse locale en procédant comme suit :

1. Identification des *n* tops passages de documents par appariement vectoriel avec la requête

2. Pour chaque concept identifié, calculer

$$Sim(Q_k,t_i)=\prod_{tj\in Qk}(\delta+\frac{\log(f(t_i,t_j)*idf_i}{\log(Ns)})^{idfj}$$

Où :
Ns : Nombre de top documents sélectionnés
idf_i : Fréquence inverse du terme t_i
δ : Constante

$$f(t_i,t_j)=\sum_{k=1}^{T} pf_{jk}*pf_{ik}$$

Avec :
Pf_{jk} : Fréquence du terme t_i dans le kème passage

3. Les *r* top termes sont ajoutés à la requête avec un poids : $q_{ki}=(1-0.9*i)/m$
où i est la position du terme dans la liste

Les expérimentations réalisées sur la base TREC ont montré la difficulté d'ajustement des paramètres de la fonction de similitude relativement à d'autres types de collection.

Par ailleurs, plutôt que d'utiliser uniquement l'hypothèse de pertinence des *n* tops documents, Mitra [Mitra & al, 1998] propose l'utilisation d'expression de filtres pour la sélection de documents utilisés pour l'expansion. Plus précisément, les auteurs comparent l'utilisation de filtres traduits sous forme d'expressions booléennes, de contraintes de proximité et de corrélation entre termes.
Comme la présence d'une proportion importante de documents non pertinents en début de liste est la raison principale de la dérive de requête [2], les auteurs proposent un procédé permettant d'augmenter la précision au rang top comme suit :

1. Soit *T* documents retrouvés par la recherche initiale et *K (K<T)* à utiliser pour le feedback. Pour chaque document, calculer un nouveau degré de similitude avec la requête, basé sur la présence d'indicateurs de pertinence. A cet effet, les auteurs ont expérimenté différentes expressions de filtre : expression booléenne manuelle, contrainte de proximité et mesure de corrélation.

   - Expression booléenne construite manuellement

   Une requête Q_k est traduite sous forme d'une conjonction d'aspects, décrit chacun par une disjonction de termes : $(t_{11}\ OR \ldots\ t_{1k1})\ AND \ldots\ (t_{m1}\ OR \ldots.\ t_{mkm})$

   Où :

   t_{inj} : ième terme du jème aspect de la requête

   m : Nombre d'aspects exprimés dans la requête

   - Contrainte de proximité

   On organise le document en passages de taille fixe puis on les ordonne sur la base de la formule :

$$Sim_{new}(D)=\sum_{i=1}^{M}\max_{k=1}^{ni} idf(t_{ik})$$

   Où :

   $idf(t_{ik})$: Fréquence de document inverse du terme t_{ik}

   - Corrélation

   On calcule pour chaque document top retrouvé une nouvelle valeur de similitude :

$$Sim_{new}(D)=idf(t_1)+\sum_{i=2}^{T} idf(t_i)*\min_{j=1}^{i-1}(1-P(t_i/t_j))$$

   Où :

   $\{t_1 \ldots t_{tn}\}$: : Ensemble des termes du document D, ordonnés dans le sens croissant de dfs

   $P(t_i / t_j)$: Coefficient de cooccurrence des termes t_i et t_j dans l'ensemble dfs

   $Idf(t_i)$: Fréquence inverse du terme t_i

   Avec :

   dfs: Ensemble de documents initialement retrouvés

[2] Phénomène dû à une expansion qui éloigne le descripteur de la requête de sa structure initiale

Cette formule permet d'atténuer la contribution des termes les plus fréquents dans les documents retrouvés lors de l'expansion de requête.

2. Réordonner les *T* documents sur la base du résultat d'application du filtre

3. Sélectionner les termes issus des *K* top documents de la liste nouvellement ordonnée et les utiliser pour construite la requête étendue

4. Effectuer la recherche avec la requête étendue

Une analyse minutieuse des résultats d'expérimentations sur les bases TREC3, TREC4, TREC5 et TREC6 ont permis aux auteurs de conclure que :

1. Le réordonnancement d'un plus petit nombre de documents *(50-100)* produit de meilleurs résultats dans le cas de méthodes automatiques et ce, à l'inverse des méthodes manuelles. Ceci est expliqué par le fait, qu'étant basé sur des suppositions de pertinence sujettes à l'erreur, l'algorithme automatique, filtre moins bien que l'algorithme manuel et de là, échoue d'avantage dans le cas d'utilisation d'un nombre plus important de documents supposés pertinents.

2. L'usage de la formule de cooccurrence proposée est très appropriée. Elle permet en effet, d'éliminer du filtre des documents portant sur le même aspect de la requête. Ceci a pour conséquence, une augmentation de la précision.

3. L'expansion est particulièrement avantageuse pour les requêtes ayant une faible précision initiale.

D'autres travaux ont porté sur l'utilisation d'éléments de la théorie de l'information pour la sélection des termes candidats à l'expansion [Carpineto & al, 1999]. Dans ce cadre, en se basant sur la notion d'entropie relative entre deux distributions, mesurée par la distance de Kullback-Leiber, les auteurs proposent une formule théoriquement justifiée pour le calcul et comparaison de distributions des termes dans les documents sélectionnés relativement aux documents de la collection. Plus précisément, on calcule le score de sélection d'un terme comme suit :

$$\sigma(t) = \left(\delta \frac{f(t)}{NR} - Pc(t) \right) \log \frac{\delta \frac{f(t)}{NR}}{Pc(t)}$$

Où :

δ : Constante

f(t) : Fréquence du terme t dans top documents retrouvés

P_c (t) : Probabilité que le terme t appartienne aux top documents retrouvés

NR : Nombre total de termes dans les top documents retrouvés

Les termes sélectionnés doivent satisfaire la condition $\delta(t) > m(\sigma)$
Où :

$$m(\delta)=\max_{t\in v(R)} P_c(t)\frac{1-\delta-A}{A}\log\frac{1-\delta}{A}$$

avec :
V(R) : Ensemble des termes appartenant aux top documents retrouvés

$$A=\sum_{t\in V(R)} P_c(t)$$

Cette condition signifie que la contribution de tout terme sélectionné par l'expansion, doit être supérieure à la contribution de tout terme n'appartenant pas aux top documents retrouvés. Le poids des termes ajoutés à la requête est obtenu par division du score sur le poids maximal de la requête.
Cette approche a été expérimentée sur la base TREC7 et a été à l'origine d'un accroissement de *11,92%* relativement à la baseline. Il a été observé, en outre, que les résultats de l'expansion dépendaient largement de la qualité de la recherche initiale estimée en précision moyenne. Cependant, aucune relation formelle n'a été établie entre ces deux paramètres.

3.1.3. La reformulation par injection de pertinence

La reformulation de requête par injection de pertinence est plus connue sous le nom de *Relevance Feedback* [Rocchio, 1971]. Cette méthode permet une modification de la requête initiale, sur la base des jugements de pertinence de l'utilisateur sur les documents restitués par le système. La relevance feedback est une forme de recherche évolutive et interactive. Son principe fondamental est d'utiliser la requête initiale pour amorcer la recherche d'information puis exploiter itérativement les jugements de pertinence de l'utilisateur afin d' « ajuster » la requête par expansion ou repondération.
La nouvelle requête obtenue à chaque itération de feedback, permet de « corriger » la direction de recherche dans le fond documentaire, et ce, dans le sens des documents pertinents.
Un nombre considérable de travaux se sont intéressés à l'intégration de techniques de relevance feedback à des modèles de recherche de base.

3.1.3.1. Reformulation dans le modèle vectoriel

Les stratégies de reformulation développées dans le modèle vectoriel induisent une repondération de requête avec expansion. La reformulation consiste alors à orienter le vecteur requête vers les vecteurs documents pertinents et de l'éloigner des vecteurs documents non pertinents.
Rocchio [Rocchio, 1971] décrit une stratégie permettant de dériver itérativement le vecteur requête optimal à partir d'opérations sur les vecteurs documents pertinents et vecteurs documents non pertinents. La formule posée est la suivante :

$$Q_{i+1} = \alpha Q_i + \frac{\beta}{P} \sum_{dp \in Dp} dp - \frac{\delta}{Np} \sum_{dnp \in Dnp} dnp \qquad (1)$$

Où :
Q_{i+1} : Requête construite à la i+1 ème itération de feedback
Q_i : Requête construite à la i ème itération de feedback
Dp : Ensemble des documents jugés pertinents
Dnp : Ensemble des documents jugés non pertinents
P : Nombre de documents jugés pertinents
Np : Nombre de documents jugés non pertinents
α, β, δ : Constantes

Le jugement de l'utilisateur est ainsi exploité pour :
- ajouter des termes issus des documents pertinents : leurs coefficients deviennent non nuls dans le nouveau vecteur requête,
- repondérer les termes de la requête : les poids des termes de la requête sont réevalués sur la base de leur fréquence d'occurrence dans les documents pertinents et documents non pertinents.

Salton et Buckley [Salton & Buckley, 1990] ont comparé l'effet de formule de Rocchio, Ide-Regular et Ide-Dechi, sur différentes collections.

$$Q_{i+1} = \alpha Q_i + \beta \sum_{dp \in Dp} dp - \delta \sum_{dnp \in Dnp} dnp \qquad \text{Ide-Regular (2)}$$

$$Q_{i+1} = \alpha Q_i + \beta \sum_{dnp \in Dp} dp - \delta dnp \qquad \text{Ide-Dec-Hi}^3 \quad (3)$$

Où :
dnp : premier document jugé non pertinent

Les auteurs ont mené une série d'expérimentations pour évaluer la reformulation par injection de pertinence, en comparant l'impact de l'utilisation des trois formules sur les résultats de recherche d'information, effectuées dans les collections CRANFIELD, CISI et MED. Les résultats présentés montrent que la formule Ide-Dec-Hi donne les meilleurs résultats avec les paramètres $\alpha = 1, \beta = 0.75, \delta = 0.25$.

Buckley & al [Buckley & al, 1994] se sont intéressés à l'application de la technique de relevance feedback dans la base TREC. La nouvelle requête est obtenue selon la formule Ide_Regular avec les paramètres $\alpha = 8, \beta = 16, \delta = 4$.

Les résultats obtenus dans la base TREC2, pour la tâche de routing, montrent un accroissement de performance de *24%* lors de l'expansion et repondération de requêtes.

[3] Dec-Hi : **Dec**rease using **Hi**ghest ranking non relevant documents

Par ailleurs, les auteurs définirent des mini-documents par groupage de *200* mots interconnectés dans le but de restituer les parties de documents pertinents (passage retrieval). Chaque terme du mini-document est localement pondéré. La combinaison de la similitude locale et similitude globale dans le calcul de similitude requête/documents, a permis d'améliorer d'avantage la reformulation de requête avec un taux de *16%* .

La reformulation de requête est de mise en œuvre aisée dans le modèle vectoriel. Cependant, le problème de traitement des termes de manière indépendantes, demeure non résolu.

3.1.3.2. Reformulation dans le modèle probabiliste

Sur la base du modèle probabiliste, Harman [Harman, 1992], Haines [Haines & Croft, 1993] et Robertson [Robertson & al, 1995] ont développé des formules de pondération de requête en utilisant le jugement de l'utilisateur sur la pertinence des documents restitués par le système.
Robertson calcule la similitude initiale Document-requête selon la formule :

$$Sim(Q_k,D_j)=\sum_{i=1}^{T} q_{ki}*d_{ji}*\log\frac{P_i(1-U_i)}{U_i(1-P_i)}+C$$

Où :
P_i : Probabilité (d_{ji} = 1/ D_j est Pertinent)
U_i : Probabilité (d_{ji} = 1/ Dj est Non Pertinent)
C : Constante
Avec :
D_{ji} = 1 si t_i occurre dans D_j, 0 sinon
P_{init} = 0.5
U_{init} = n_i / N

Les recherches ultérieures exploitent l'occurrence des termes dans les documents jugés pertinents et documents jugés non pertinents. Une liste de termes candidats à l'expansion de requête, sont triés selon une valeur de sélection donnée par la formule :

$$VS(t_i)=\log\frac{p_i*r_i(1-q_i)}{q_i*R(1-p_i)}$$

Où :
r : Nombre de documents jugés pertinents, contenant le terme candidat t_i
R : Nombre de documents jugés pertinents

Les liens de dépendance conditionnelles sont repondérés et calculés comme suit :

$$P_i=\frac{r_i}{R} \qquad U_i=\frac{n_i-r_i}{NR-R}$$

Où :
NR : Nombre de documents non pertinents retrouvés

La fonction de similitude utilisée lors des itérations feedback devient alors la suivante :

$$SIM(Q_k,D_j)=\sum_{i=1}^{T} q_{ki}\log\left(\left(\frac{r_i}{R-r_i}\right)+\frac{(n_i-r_i)}{(N-NR-n_i+r_i)}\right)$$

Il en résulte ainsi une repondération de la requête avec expansion.

Haines & Croft [Haines & Croft, 1991] étendérent le modèle d'inférence introduit par Turtle & Croft [Turtle & Croft, 1991] en incluant des techniques de reformulation de requête. Un nouveau type de nœud et deux couches associées y ont été intégrés dans le réseau afin de traduire le jugement de pertinence de l'utilisateur, relativement à la présence des concepts modélisés dans les documents restitués.
Ces travaux ont montré la faisabilité de la relevance feedback dans le modèle d'inférence bayésien. Cependant, la complexité engendrée dans la structure du réseau et dans le calcul de probabilités conditionnelles dégradent les performances globales de la stratégie [Ponte, 1998].

Dans le cas du modèle probabiliste, la stratégie d'injection de pertinence présente l'intérêt d'être directement reliée à la dérivation de poids des termes de la requête. Cependant, elle pose le problème de la complexité de calcul des probabilités conditionnelles.

3.1.3.3. Reformulation dans le modèle connexioniste

La relevance feedback a été également expérimentée pour améliorer les résultats de la recherche d'information dans un modèle connexioniste [Wilkinson & Hingston, 1991] [Kwok, 1995] [Boughanem & Soule-Dupuy, 1997].

Le modèle de Wilkinson et Hingston [Wilkinson & Hingston, 1991] est basé sur un réseau à deux couches : couche de mots et couche de documents. Les connexions entre ces deux couches sont bidirectionnelles et pondérées par des formules classiques.
L'interrogation du réseau à l'aide d'une requête, provoque la propagation des signaux d'activation depuis les neurones termes de la requête jusqu'aux neurones documents. Le jugement de pertinence est basé sur une estimation interne; les documents dont le niveau d'activation est supérieur à un seuil sont considérés pertinents. L'itération feedback correspond alors à la poursuite de la recherche par propagation des signaux d'activation à travers les connexions inverses, depuis les neurones documents pertinents jusqu'aux neurones termes. Il en résulte le calcul d'une nouvelle activation des neurones termes qui correspond ainsi à la constitution d'une nouvelle requête.
Les travaux réalisés sur la collection CACM montrent qu'en moyenne, deux itérations améliorent les performances de *12%* par rapport à la mesure du cosinus.

Des travaux de Boughanem & Soule-Dupuy [Boughanem & Soule-Dupuy, 1999] ont porté sur l'utilisation du modèle connexioniste Mercure [Boughanem & Soule-Dupuy, 1997] pour l'expansion de requête.
Cette dernière est le résultat de l'application de l'algorithme de rétropropagation à partir de neurones documents associés aux *n* top documents retrouvés à la recherche initiale et ce comme suit :

1. Construction de la sortie désirée $Desired\ Output = (rel_1 , rel_2 , ..., rel_n)$

2. Application de la sortie désirée à la couche documents. Chaque neurone calcule une valeur d'entrée $In (Nd_j) = rel_i$ et un signal de sortie $Out (Nd_j) = g (In (Nd_j))$
   Où :
   Nd_j : Neurone document associé au document D_j
   G : Fonction de sortie

3. Rétropropagation des signaux de sortie vers la couche termes. Chaque neurone terme calcule une valeur d'entrée

$$In(Nt_i)=\sum_{j=1}^{N} d_{ji} * Out(Nd_j)$$

   puis calcule un signal de sortie $Out (Nt_i) = g(In (Nt_i)$
   Où :
   W_{ij} : Poids de la connexion du neurone terme t_i au neurone terme t_j

4. Calcul de la nouvelle entrée selon la formule $Q_{new} = \alpha\ Q_{old} + * \beta Out (N_t)$

   Où :
   α, β : Constantes

Les expérimentations réalisées sur la base TREC6 ont révélé un accroissement de performances de *16%* relativement à la baseline et ont permis aux auteurs de conclure que la qualité des documents utilisée pour la reformulation, décrits en termes et poids, a un impact plus considérable sur les résultats de la recherche, que le nombre associé.

Le modèle connexioniste présente l'avantage de disposer d'un support théorique rigoureux pouvant être mis à profit pour le développemnt de la stratégie de reformulation par injection de pertinence : algorithme de rétropagation, règles d'apprentissage de Hebb etc… Cependant, il pose également une complexité de calcul lors du mécanisme d'activation propagation.

3.1.3. Paramètres de performance

Un nombre considérable d'expérimentations ont été effectuées sur les collections de documents pour l'évaluation de l'impact induit par la reformulation de requête sur le processus de recherche d'information. Une étude synthétique de ces différents travaux, nous amène à dire que l'ordre des performances imputées à l'intégration de cette stratégie est variable, dépendant de divers conditions d'exploitation :

- modèle de recherche,
- hypothèse de base quant à la distribution des termes dans les documents, sémantique d'un terme, concept, phrase et relation sémantique entre eux,
- caractéristiques des collections de documents : taille, nombre, source etc…

En faisant abstraction des paramètres caractéristiques inhérents à chacune des techniques de reformulation de requête présentées, nous tentons de dégager dans ce qui suit, les paramètres de performance intrinsèques.

1. Nombre de termes ajoutés à la requête

L'ajout de termes à la requête accroît la performance du SRI dans le cas des deux stratégies décrites précédemment. Buckley & al [Buckley & al, 1994 a] ont expérimenté la relevance feed-back dans l'environnement multi-fond documentaire TREC; ils ont montré que le taux de performance est d'avantage corrélé avec le nombre de termes ajoutés qu'avec le nombre de documents initialement retrouvés.

Ils ont abouti à la mise au point de l'équation de variation :

$$RP(N) = A\ Log(Ns) + B\ Log(X) + C$$

Où :

RP (N) : Performance du système pour N documents restitués
Ns: Nombre de documents restitués
X : Nombre de termes ajoutés à la requête
A, B, C : Constantes

Ils ont conclu que le seuil critique du nombre de termes à ajouter à la requête dépend des caractéristiques de la collection

Harman [Harman, 1992] a par ailleurs montré, que la meilleure méthode de sélection des termes issues des documents pertinents devient inefficace après l'ajout de *20* à *40* termes à la requête initiale, sur des bases de tailles moyenne (CACM, Cranfield ...). En outre, la pondération différenciée des termes ajoutés à la requête accroît la performance du système. On attribue un poids moins important aux termes ajoutés [Haines & Croft, 1993], plus important aux termes issus des documents pertinents que ceux issus des documents non pertinents [Salton & Buckley, 1990].

2- Méthode de sélection des termes

La méthode de sélection des termes à ajouter à la requête est aussi importante que le choix de leur seuil. Nous citerons les principales méthodes expérimentées.

Salton et Buckley [Salton & Buckley, 1990] ont expérimenté séparément, l'ajout de tous les nouveaux termes, tous les termes issus des documents pertinents et les termes les plus fréquents dans les documents restitués à la requête initiale.
L'expansion de la requête avec tous les nouveaux termes offre de meilleurs résultats que les autres méthodes; toutefois l'écart de performance n'est pas très considérable relativement aux exigences de temps et d'espace mémoire.

Robertson [Robertson & al , 1995] et Haines [Haines & Croft, 1993] adoptent une méthode de sélection de nouveaux termes sur la base d'une fonction qui consiste à attribuer pour chaque terme un nombre traduisant sa valeur de pertinence. Les termes sont alors triés puis sélectionnés sur la base d'un seuil.
Robertson propose la formule suivante pour le calcul de la valeur de sélection d'un terme :

$$SV(i) = w\,(P_i - U_i)$$

Où :

$$w = \log \frac{P_i(1-U_i)}{U_i(1-P_i)}$$

Avec :

P_i : Probabilité (d_i = 1/ D est Pertinent)
U_i : Probabilité (d_i = 1/ D est Non Pertinent)

Harman [Harman, 1992] propose les fonctions suivantes :

1. $SV(i) = \frac{RT_j * df_i}{N}$

Où :
RT_j : Nombre total de documents retrouvés par la requête
df_i : Fréquence d'occurrence du terme t_i dans la collection
N : Nombre total de documents dans la collection

2. $SV(i) = \frac{r_i}{R} - \frac{df_i}{N}$

Où :
r_i : Nombre de documents pertinents contenant t_i
R : Nombre de documents pertinents

3. $SV(i)=\log_2 \frac{p_i(1-q_i)}{(1-p_i)}$

Avec :
p_i : Probabilité que t_i appartienne aux documents pertinents
q_i : Probabilité que t_i appartienne aux documents non pertinents

Les expérimentations réalisées sur différentes collections standards, ont révélé que la troisième fonction est la meilleure.

[Lundquist & al, 1997] ont expérimenté la fonction p_i*$nidf$ en utilisant la fonction de pondération des documents normalisée par la longueur [Singhal & al, 1995]. Les résultats montrent un accroissement de *31%* des performances à l'ajout des *10* top termes et ce, dans des collections moyennes.

3- Longueur moyenne de requête

L'accroissement des performances est plus important lorsque les collections sont interrogées par des requêtes de longueur relativement petite [Buckley & al, 1994]. Dans ce sens, des expérimentations intéressantes ont été réalisées sur la base TREC7 et présentées dans [Cormack & al, 1999]. Les auteurs montrent en effet que la dérivation automatique de courtes requêtes à partir de documents jugés ou supposés pertinents à la suite d'une recherche initiale, permettent d'atteindre des résultats très performants pour différentes tâches : recherche, filtrage et routing.

3.2. Recherche basée sur le passage de document

Dans ce cadre, la recherche d'information est établie à partir d'une stratégie de recherche et d'une technique de structuration de documents en vue d'en restituer des ***parties pertinentes.*** La principale motivation de cette stratégie est la difficulté des algorithmes classiques d'appariement à localiser les « régions » d'appariement dense, relativement à l'intégralité du document. Cette difficulté se présente avec plus d'acuité dans les documents présentant une structure complexe ou une disparité dans les sujets contenus [Grossman & Frieder, 1998].

De ce point de vue, on perçoit la recherche d'information comme un processus de recherche de passage pertinent plutôt que de document. Durant l'indexation, un document est subdivisé en passages avec liens éventuels ; chaque passage est repéré comme unité distincte. Cette approche a l'avantage de maintenir les algorithmes classiques applicables mais soulève le problème d'identification adéquate de passage [Salton & al, 1993]. On distingue principalement deux types de passages : passage fixe et passage dynamique.

3.2.1. Passage fixe

Le passage est dans ce cas préalablement défini comme étant délimité au paragraphe ou section [Zobel & al, 1995] où à une plage de mots [Callan, 1994].
Cette technique de délimitation de passages n'a cependant pas permis d'accroître de manière significative les performances de recherche. Une principale raison évoquée par les auteurs, est que cette structuration n'est pas équivalente à une partition sémantique et ne peut donc pas convenir de manière uniforme à toute les requêtes.

3.2.2. Passage dynamique

L'approche consiste dans ce cas à partitionner le document en fonction de la requête en cours. Ceci permet de remédier à la technique de passage fixe en adoptant un principe de partitionnement dépendant de la partition sémantique de la requête en cours.
Le principe de partitionnement de document présenté dans [Callan, 1994] est le suivant :

1. Identifier, dans le document, la position du premier terme contenu dans la requête.

2. Identifier des passages successifs de taille fixe *P* comprenant les termes de positions *[n+(i-1)P/2 … n+(i+1)P/2]*.
   Il en résulte un découpage du document en passages non disjoints permettant ainsi de répondre à des requêtes qui s'apparient à différents passages.
   L'auteur propose un calcul d'appariement basé sur l'estimation de la valeur de pertinence locale au passage et globale au document. La fonction pertinence est de la forme :

$$RSV(Q_k,D_j) = RSV(Q_k,D_j)_G + \theta\, RSV(Q_k,D_j)_L$$

Où :
$RSV(Q_k,D_j)_G$: Valeur de pertinence relativement à la globalité du document
$RSV(Q_k,D_j)_L$: Valeur de pertinence relativement au passage du document
θ : Facteur de pondération

Les expérimentations réalisées sur les collections locales Federal Register et base de documents légaux, montrent que les meilleurs résultats sont atteints pour une longueur moyenne de passage documentaire de *100* à *300* mots et un poids plus important pour la pertinence locale.
Des travaux analogues sont présentés dans [Wilkinson, 1994] et [Knaus & al, 1994].

4. Evaluation de la recherche d'information

L'évaluation constitue une étape importante lors de la mise en œuvre d'un modèle de recherche d'information puisqu'elle permet de paramétrer le modèle, d'estimer l'impact de chacune de ses caractéristiques et enfin de fournir des éléments de comparaison entre modèles.
L'évaluation nécessite alors la définition d'un ensemble de mesures et méthodes d'évaluation et bases de test assurant l'objectivité de l'évaluation.

4.1. Les mesures de rappel/précision

Ce sont les mesures les plus utilisées pour l'évaluation d'un modèle de recherche d'information. De manière classique, ils sont obtenus en partitionnant l'ensemble des documents restitués par le SRI en deux catégories : documents pertinents et documents non pertinents.

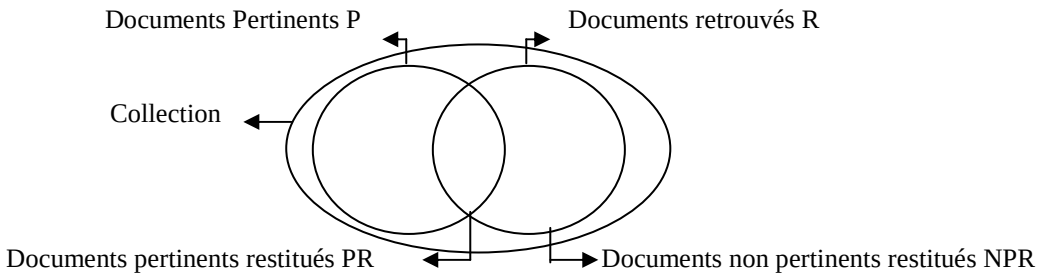


Figure 1. 4. : Partition de la collection pour une requête

On définit :

Rappel : Proportion de documents pertinents restitués par le système relativement à l'ensemble des documents pertinents contenus dans la base. Le rappel est calculé selon la formule suivante :

$$Rappel = \frac{|PR|}{P}$$

Précision : Proportion de documents pertinents relativement à l'ensemble des documents restitués par le système. La précision est calculée selon la formule suivante :

$$Pr\acute{e}cision = \frac{|PR|}{|R|}$$

Toutefois, seule une partie des documents restitués par le système est examinée par l'utilisateur. Dans ce cas, la paire de mesures (taux de rappel, taux de précision) est calculée à chaque point de rappel (document pertinent restitué) comme le montre l'exemple illustré sur le tableau 1.1.

Documents restitués	*Rappel*	*Précision*
[D1] P	1/3 = 0.33	1/1=1
[D2] NP		
[D3] NP		
[D4] P	2/3=0.66	2/4=0.5
[D5] P	3/3=1	3/5=0.6
[D6] NP		

Tableau 1.1. : Evaluation du taux de rappel/précision par rapport à la réponse du système

P désigne Pertinent, NP désigne Non Pertinent

Par ailleurs, l'évaluation d'un modèle de recherche d'information est effectué sur la base d'une collection de requêtes test. La précision moyenne au taux de rappel *rp* est calculée comme suit :

$$\overline{P(rp)}=\sum_{i=1}^{Nq}\frac{P_i(rp)}{Nq}$$

Où :
N_q : Nombre total de requêtes
$P_i(r)$: Précision de la requête au niveau de rappel rp

Comme les niveaux de rappel ne sont pas unifiés pour l'ensemble des requêtes, on retient dans la littérature, *11* points de rappel standards *0.00 à 1.00* à pas de *0.1*.
On procède alors par une méthode d'interpolation, présentée ci après, qui permet de dresser le graphique standard rappel/précision.

On a alors : $$\overline{P(rp)}=\sum_{r\in\{0...1.0\}}\frac{P_i(rp)}{N_q}$$

Une variante de la précision moyenne standard consiste à calculer la précision moyenne à un nombre fixe de documents restitués. Cette dernière mesure nous permet en outre d'évaluer la qualité de l'ordre des documents restitués par le système. Un des objectifs essentiels d'un modèle de recherche d'information est en effet de positionner les documents pertinents en début de liste.
Sur la base de ces mesures, des outils variés sont utilisés pour l'évaluation : courbes, histogrammes, tableaux statistiques etc…

4.1.1. Méthode d'évaluation par interpolation

Cette méthode est basée sur le principe d'ordonnancement des documents restitués et respecte l'hypothèse de variation inverse du taux de rappel et de précision.
Soit une requête *Q* pour laquelle il existe *Dp* documents pertinents dans la base; on calculera alors les taux de précision pour *Dp* valeurs de rappel: *1/DP, 2/DP, ..., DP/DP*
Le principe de l'extrapolation est le suivant : chaque fois que le taux de précision pour un taux de rappel *k / DP (k≥2)* est supérieur au taux de précision pour le taux de rappel *(k-1) / DP*, sa valeur est remplacée par celle du point *k / DP*. Ce processus est ainsi itéré jusqu'au point de rappel *1*. Le tableau 1.2 illustre ce principe de calcul.

Documents restitués	*Rappel Calculé*	*Précision Calculée*	*Rappel fixé*	*Précision extrapolée*
[D1] P			0.1	1
[D2] NP	1/6=0.17	1/1=1	0.2	0.6
[D3] P			0.3	0.6
[D4] NP	3/6=0.5	3/5=0.6	0.4	0.6
[D5] P	4/6=0.67	4/7=0.57	0.5	0.6
[D6] P			0.6	0.57
[D7]NP	6/6=1	6/67=0.09	0.7	0.09
[D8]NP			0.8	0.09
			0.9	0.09
			1.0	0.09

Tableau 1.2 : Principe de l'extrapolation de la précision

4.1.2. Méthode d'évaluation résiduelle

Cette méthode est adaptée à l'évaluation d'un mécanisme de recherche d'information basé sur la relevance feedback. La mesure des taux de rappel et précision doit être effectuée avec précaution lorsqu'on évalue particulièrement la performance induite par l'intégration de la relevance feedback. En effet, le procédé de feedback est tel que tout document initialement restitué à une requête, paraît à nouveau avec un rang amélioré (effet de rang) dans les itérations ultérieures du feedback. Les valeurs de rappel/précision croissent en valeur absolue mais ne reflètent pas effectivement la satisfaction de l'utilisateur. La mesure doit plutôt estimer la capacité de la relevance feedback à rappeler de nouveaux documents. L'une des solutions apportées est la méthode de collection résiduelle . Cette méthode préconise de ne pas considérer les documents préalablement jugés pour l'évaluation des résultats de l'itération feedback courante. Les valeurs de rappel/précision décroissent en valeur absolue mais la mesure du pourcentage de nouveaux documents entre la réponse à la requête initiale et la réponse à la requête étendue, traduit la performance effective due à la relevance feedback.

4.2. Les mesures combinées

L'idée de définir de nouvelles mesures qui combinent les mesures standards de rappel/précision est principalement motivée par [Korfhage, 1997] :

1. la difficulté de calcul du rappel maximal dans les collections volumineuses (dénombrement des documents pertinents à une requête),

2. l'inadéquation de ces mesures dans le cas où la fonction d'appariement n'est pas une fonction d'ordre faible,

3. nécessité de combiner les deux aspects rappel/précision.

Dans le but d'y pallier, deux principales mesures combinées ont été définies : mesure harmonique et mesure orientée utilisateur.

1. Mesure harmonique

Cette mesure est proposée par shaw & al [Shaw & al, 1997]

$$F(j)=\frac{2}{\frac{1}{R(j)}+\frac{1}{P(j)}}$$

Où
R(j) : Valeur de rappel au $j^{ème}$ document restitué
P(j) : Valeur de précision au $j^{ème}$ document restitué

On note ainsi que la valeur de la mesure harmonique est élevée pour des valeurs de précision et de rappel élevées, ce qui assure que la mesure garantit le compromis entre les deux aspects.

2. Mesure orientée utilisateur

Le jugement de pertinence des documents sélectionnés étant dépendant de l'utilisateur, de nouvelles mesures ont été proposées afin de relativiser l'évaluation de recherche à l'utilisateur [Korfhage , 1997]. A cet effet, deux nouvelles mesures ont été définies :

Coverage : Proportion de documents pertinents connus de l'utilisateur et restitués par le système

$$Coverage=\frac{|R_k|}{|U|}$$

Novelty : Proportion de documents pertinents inconnus de l'utilisateur, et restitués par le système

$$Novelty = \frac{|R_u|}{|R_u| + |R_k|}$$

Où

R_u : Documents restitués, pertinents et inconnus de l'utilisateur

R_k : Documents restitués, pertinents et connus de l'utilisateur

U : Documents pertinents connus de l'utilisateur

Ces mesures restreignent les valeurs standards de rappel et précision au champ de vision d'un utilisateur.

4.3. La collection TREC

Les collections de test ont été traditionnellement utilisées en recherche d'information pour évaluer les stratégies de recherche. Cependant, en vue d'être une base de travail fiable, une collection de test doit constituer une référence sûre de comparaison entre stratégies, en accord avec leur efficacité . Plus particulièrement, une collection de référence doit traduire la subjectivité de pertinence des utilisateurs d'une part, et contenir, d'autre part, une masse d'informations assez importante et variée pour constituer un environnement standard d'interrogation.

Dans ce cadre, une série de conférences annuelles TREC [Voorhees, 1999] pour *Text REtrieval Conference* a été lancée en 1990 dans le but de conjuguer les efforts de la communauté en recherche d'information et uniformiser les outils d'évaluation.

4.3.1. Structure

TREC offre une très large collection de documents de sources très variées : Financial Time, Résumés de publications USDOE, SAN JOSE Mercury news etc… organisées en sous collections, qui évoluent d'année en année.

Le tableau 1.3 présente les caractéristiques de la collection TREC6.

Un document TREC est généralement présenté sous le format SGML, identifié par un numéro et décrit par un auteur, une date de production et un contenu textuel.

Le tableau 1.4 présente à titre d'exemple, un document TREC.

Une requête TREC est également identifiée par un numéro et décrite par un sujet générique, une description brève et une description étendue sur les caractéristiques des documents pertinents associés à la requête.

Le tableau 1.5 présente à titre d'exemple, une requête TREC.

Disque	*Contenu*	*Taille Mb*	*Nombre de documents*	*Nombre moyen de termes par document*
1	WSJ, 1987-1989	267	98732	245
	AP, 1989	254	84678	446
	ZIFF	242	75180	200
	FR, 1989	260	25960	391
	DOE	184	226087	111
2	WSJ, 1990-1992	242	74520	301
	AP, 1988	237	79919	438
	ZIFF	175	56920	182
	FR, 1988	209	19860	396
3	SJMN, 1991	287	90257	379
	AP, 1990	237	78321	451
	ZIFF	345	161021	122
	PAT, 1993	243	6711	4445
4	FT, 1991-1994	564	210158	316
	FR, 1994	395	55630	588
	CR, 1993	235	27922	288
5	FBIS	470	130471	322
	LAT	475	131896	351
6	FBIS	490	120653	348

Tableau 1.3. : Structure de la collection TREC6

```
<doc>
<docno> WSJ880406-</docno>
<hl>AT&T Unveils Services to Upgrade Phone Networks Under
Global Plan </hl>
<author> Janet Guyon (WSJ Staff)</author>
<dateline> NewYork < /dateline>
<text>
American Telephone & Telegraph Co. introduced the first of a new
Generation of phone services with broad …
</text>
</doc>
```

Tableau 1.4 : Structure du document TREC identifié WJS880406-0090

<top>
<num> Number : 168
<title >Topic : Financing AMTRAK
<desc> Description :
A document will address the role of the Federal Govrnment in financing the operation of the National Railroad Transportation Corporation (AMTRAK)
<narr> Narrative : A relevant document mus provide information on the govrnment's responsibility to make AMTRAK an economically
viable entity. It could also discuss the privatisation of AMTRAK as an alternative to continuing government subsidies. Documents comparing government subsidies given to air and bus transportation with those provided to AMTRAK would also be relevant
<top>

Tableau 1.5 : Structure de la requête 168 dans la collection TREC

4.3.2. Principe de construction

Le processus de construction d'une collection TREC est le suivant :

1. On constitue un groupe d'assesseurs de pertinence. Chacun d'eux gère un ensemble d'en moyenne *10* sujets de requête et détermine les documents pertinents associés dans la collection. On sélectionne finalement *50* sujets de requêtes sur la base du nombre de documents pertinents estimé

2. Dans l'année, les participants à TREC utilisent les *50* requêtes pour leur SRI et proposent la liste des *1000* top documents obtenus pour chaque requête

3. NIST constitue un document de synthèse où figure pour chaque système et chaque requête les *100* premiers documents restitués.

4. L'assesseur de pertinence de chaque sujet de requête évalue les résultats de synthèse pour chacun des documents. On évalue alors chaque système, en considérant que tout document qui n'apparaît pas parmi les *100* premiers est non pertinent, en utilisant les mesures standards de précision moyenne à la requête et précision moyenne du système.

Voorhees [Voorhees, 1998] a mené une série d'expérimentations dans le but de vérifier la stabilité d'évaluation d'un SRI en utilisant la collection TREC.
A cet effet, l'auteur a procédé à la construction de sous-collections issus de TREC4 et TREC6, en respectant le principe général adopté dans TREC, mais en faisant varier les conditions liées à la source des jugements de pertinence.
Plus précisément, les collections construites utilisent une combinaison des jugements de pertinence provenant :

- des auteurs de documents / autres
- différentes catégories de personnes
- personnes de même environnement / différents environnements
- une personne/ groupes de personnes différentes

Les résultats ont montré que les conditions expérimentales ont un impact sur les valeurs de précision moyenne obtenues pour différents systèmes. Cependant, la quantification de la corrélation, moyennant la mesure de Kendall, montre que les écarts sont très corrélés entre les différents systèmes.
Globalement, l'auteur confirme, à travers les résultats obtenus, la fiabilité de la collection TREC pour l'évaluation et comparaison des stratégies de recherche d'information, en notant cependant qu'il convient d'être particulièrement prudent pour la comparaison de résultats obtenus avec :

- des requêtes avec peu de documents pertinents : la précision moyenne étant non stable dans ce cas, un nombre important de ce type de requêtes rendrait les résultats d'évaluation très variables,
- des méthodes fortement manuelles : l'évaluation est en effet instable dans ce cas, et ce, en raison de la subjectivité de l'utilisateur qui y intervient.

5. Conclusion

Ce premier chapitre a porté essentiellement sur l'étude des SRI de manière générale et modèles de recherche et de représentation d'information de manière particulière. Il en ressort que chacun de ces modèles ou stratégies contribue en partie à la résolution des problèmes inhérents à la recherche d'information : perception du besoin en information, représentation du sens véhiculé par les documents, formalisation de la pertinence etc…
Pour ce faire, les auteurs puisent dans une large mesure d'un support théorique permettant d'associer les différentes fonctions de calcul de poids des termes, liens terme-terme, terme-document, appariement requête-document etc…
Nous avons analysé les apports et limites des différents modèles et stratégies. On conclut que la pluralité des difficultés liées à la localisation de l'information pertinente à un besoin en information d'un utilisateur donné, plaide pour la coopération de diverses techniques de représentation, analyse et optimisation.

A l'issue de cette étude, nous nous intéressons à la conception de SRI adaptatifs aux besoins des utilisateurs. Plus précisément, nous ciblons l'objectif de mettre en œuvre un processus d'optimisation de requête, que nous greffons à un modèle de recherche d'information de base .

Dans le chapitre suivant, nous présentons les principaux concepts d’une approche novatrice d’optimisation qui est en l’occurrence matérialisée par les algorithmes génétiques. L’étude menée dans ce chapitre, nous permet de justifier puis de définir une approche d’optimisation génétique de requêtes dans les SRI.

Chapitre 2

Concepts et Principes des Algorithmes Génétiques

1. Introduction

L'homme et la nature constituent incontestablement, la source d'inspiration fondamentale de l'intelligence artificielle. Cette dernière étant une science visant la reproduction automatique de comportements humains, est essentiellement basée sur la modélisation des phénomènes naturels.
L'origine de l'inspiration couvre très souvent les pensées d'une école au sein de cette discipline. Ainsi, et à titre illustratif, les neurones biologiques et modes d'activations associés, les fonctionnements cognitifs humains et les systèmes immunitaires constituent l'arrière base naturelle des systèmes artificiels que sont respectivement les réseaux neuronaux, les systèmes experts et les réseaux immunitaires.

Pour notre part, nous nous intéressons aux travaux fondés sur l'exploitation des lois naturelles de la sélection énoncées par Darwin, pour la conception de systèmes artificiels. Ainsi, le principe de survie des plus adaptés d'une part et les mécanismes de transmission génétique générationnelle d'autre part, constituent les idées novatrices véhiculées par la classe des algorithmes d'évolution artificielle.
Au sein de cette large classe d'algorithmes, nous focalisons notre intérêt sur les algorithmes génétiques. Ces derniers sont nés des réflexions darwiniennes relatives à la théorie de l'évolution des espèces. L'idée clé de cette théorie est que, sous les contraintes imposées par l'environnement, les espèces d'êtres vivants se sont progressivement automodifiées dans le but de s'adapter à leurs milieux naturels. Ce processus d'évolution, induit la régénération de populations, basées sur la combinaison des caractéristiques de base des individus qui les composent, en vue de les rendre de plus en plus adaptés.
Dans cet ordre d'idées, est né l'intérêt de définir des algorithmes génétiques artificiels permettant d'améliorer en cours de générations, des solutions candidates à un problème, vers des solutions de plus en plus adaptées. Ceci justifie finalement, l'application des algorithmes génétiques à la résolution de problèmes d'optimisation.

On attribue la parenté des AG's à Jhon Holland et son équipe de l'université de Michigan où les concepts y afférents sont apparus aux années 1960 puis synthétisés en 1975 dans le livre intitulé *Adaptation in Natural and Artificial Systems* [Holland, 1975].

Les AG's ont connu un grand essor dans les années 1980 suite aux travaux de Goldberg; celui-ci a en effet réalisé un AG fournissant des résultats probants pour la résolution d'un problème complexe qu'est celui de la commande optimale d'un réseau de pipelines. Plusieurs recherches ont apporté plus de crédibilité aux AG's en étudiant leur preuve de convergence.

Leur application à la résolution de problèmes divers se répand, à l'heure actuelle de façon considérable. Cet engouement est d'autant plus soutenu que d'autres concepts biologiques sont introduits dans les modèles d'optimisation par algorithme génétique (niches écologiques, diploïdie, dominance, etc...) ouvrant la voie à des pistes de recherche prometteuses.

Nous décrivons dans ce chapitre les principaux paradigmes de l'algorithmique évolutive puis détaillons, au sein de cette large classe d'algorithmes, les principes et concepts des algorithmes génétiques. Nous présentons également des éléments de la théorie qui supporte leur principe d'optimisation, des heuristiques et techniques d'adaptation et enfin des possibilités d'implémentation sur des architectures parallèles.

2. L'algorithmique évolutive

Les algorithmes évolutifs sont des algorithmes stochastiques fondés sur la simulation du processus d'évolution et d'adaptation des organismes dans les milieux naturels. Ces algorithmes sont adaptés à la résolution de problèmes dont l'espace de recherche est caractérisé par un grand nombre de dimensions et de nombreux optima locaux [Preux, 1995].
Les techniques d'algorithmique évolutive ont attiré une attention considérable en raison des potentialités qu'elles offrent pour la résolution de problèmes complexes. Ces techniques basées sur le principe puissant de « survie du meilleur », modélisent les phénomènes naturels liés à la génétique darwinienne ; elles constituent une catégorie intéressante d'heuristiques de recherche et d'optimisation modernes.

Du point de vue de l'optimisation, les algorithmes évolutifs sont des méthodes d'ordre *0,* pouvant retrouver l'optimum global de problèmes. Un ensemble potentiel de solutions est renouvelé à chaque génération en favorisant la survie des populations les plus performantes. Par analogie aux mécanismes de la génétique, ces algorithmes combinent de manière pseudo-aléatoire, les informations portées par des solutions servant de base pour la construction de solutions plus adaptées.
La structure générique d'un algorithme évolutif est présentée sur la figure 2.1.

Les algorithmes évolutifs sont caractérisés par :

- la manipulation d'une **population d'individus** représentant les solutions candidates au problème posé,
- **l'évaluation de la qualité des individus** grâce à une fonction d'adaptation,
- la détermination d'une **stratégie de sélection** des individus d'une génération à une autre,
- l'application **d'opérateurs de transformation d'individus** entre générations.

```
Algorithme Evolutif
  Début
   t := O
   Initialiser la population
   Evaluer la population
   Tant que ( ⌉Condition de terminaison) Faire
     Début
      t :=  t+1
     Sélectionner les individus pour la reproduction
     Appliquer les opérateurs d'évolution
     Evaluer la génération enfant
     Fin
   Fait
   Fin
```

Figure 2.1. : Structure d'un algorithme évolutif

La littérature sur l'algorithmique évolutive fait état de travaux se rapportant à différentes variantes du modèle générique dont on cite principalement : les algorithmes génétiques [Holland, 1975][Goldberg, 1989], les stratégies d'évolution [Baeck & al, 1991], la programmation évolutive [Fogel & al, 1966] et la programmation génétique [Koza, 1992].
Ces modèles ont été définis indépendamment les uns des autres et ce n'est qu'au début des années 1990 que les communautés ont commencé à conjuguer leur réflexions.
Nous allons décrire dans ce qui suit, les principales classes d'algorithmes évolutifs puis mettrons en évidence les principaux rapprochement et différences les caractérisant.

2.1. Les algorithmes génétiques

En se rapportant à la version canonique des AG[4]'s [Holland, 1975], les individus sont des chaînes construites sur l'alphabet binaire {0, 1} et sont de longueur fixe, formant une population de taille constante en cours du temps. Trois principaux opérateurs d'évolution sont appliqués à chaque génération :

1. sélection basée généralement sur la valeur d'adaptation des individus,
2. croisement qui consiste à combiner, de manière pseudo-aléatoire, les informations portées par deux individus parents, pour former un individu enfant,
3. mutation qui consiste à altérer, de manière pseudo-aléatoire, la structure d'un individu.

[4] Algorithme Génétique

Basées sur le principe de l'AG canonique, de nombreuses variantes d'AG ont été proposées de manière à résoudre efficacement des problèmes divers d'optimisation.
Une description détaillée des AG's est présentée dans des paragraphes ultérieurs.

2.2. Les stratégies d'évolution

Les stratégies d'évolution ont été destinées à l'origine pour l'optimisation de fonctions [Baeck & al, 1991]. Initialement, les stratégies d'évolution manipulaient un individu représenté par un point dans un espace multidimensionnel et un opérateur de mutation qui agit en ajoutant un bruit gaussien à chacune des composantes de l'individu.
Les étapes de l'algorithme d'optimisation sont les suivantes :

1. Génération aléatoire du parent Ind_0
2. Génération aléatoire du descendant Ind_1 en fonction de l'individu Ind_0 selon la formule :

$$Ind_1 = Ind_0 + N_0 (\sigma)$$

avec $N_0 (\sigma)$: Bruit gaussien dont l'écart type se réduit en fonction du temps.

3. *Si* $f (Ind_1) > f (Ind_0)$ *Alors* $ind_0 := ind_1$
avec f : fonction à optimiser

4. Aller à *2* ou arrêt

Par la suite, les stratégies d'évolution sont devenues des techniques proches des AG's, en ce sens qu'elles manipulaient une population d'individus, un opérateur de croisement et des heuristiques de sélection. Cependant, elles en différent principalement par les points suivants [Preux, 1995] :

- le croisement peut muter des valeurs de gènes des parents,
- la sélection des individus est aléatoire, non basée sur leur valeur d'adaptation. Le dilemme exploration contre exploitation n'est donc pas résolu de façon optimale,
- les paramètres de contrôle s'intègrent systématiquement dans la représentation des individus.

En raison d'un principe de sélection non basé sur la valeur d'adaptation des individus, les stratégies d'évolution sont d'avantage destinées à des applications du type optimisation de fonctions réelles ou discrètes où la fonction est stable dans le temps, qu'à des applications nécessitant une exploitation et une exploration simultanées des résultats [Venturini, 1996].

2.3. La programmation évolutive

La programmation évolutive était destinée à l'origine au développement d'automates d'états finis [Fogel & al, 1966] et partage de nombreuses similarités avec les stratégies d'évolution. Les individus sont des variables multidimensionnelles réelles transformées que par mutation . Chaque individu génère un enfant et les meilleurs *P* individus parmi la génération parent et génération enfant, sont sélectionnés pour la génération suivante.

Des versions plus récentes [Fogel, 1995] proposent le contrôle de la mutation par un paramètre endogène, intégré à une structure d'individus, non limitée à l'espace des réels.

2.4. La programmation génétique

La programmation génétique a été fondée par Koza [Koza, 1992]. A l'origine, le but était de synthétiser des programmes LISP devant effectuer un traitement donné. Les individus étaient alors des *S* expressions arborescentes LISP, de taille non limitée, définies dans un espace de recherche virtuellement illimité.

Le principe de sélection est basé sur le déroulement d'un tournoi pour la détermination des parents. Un enfant est retenu dans la population selon la méthode de surpeuplement : *T* individus sont uniformément choisis, le plus mauvais est remplacé par un enfant.

Alors qu'initialement, seul le croisement était appliqué aux opérateurs, des travaux plus récents [Kinnear, 1996] proposent l'application de la mutation.

2.5. Synthèse et directions de recherche actuelles

L'étude des principaux paradigmes de l'algorithmique évolutive nous permet de mettre en évidence qu'outre le respect du principe fondamental de la génétique darwinienne, les différents types d'algorithmes présentent des similarités qui rendent leur caractérisation d'autant plus difficiles que les travaux actuels s'orientent vers l'intégration de nouvelles techniques et adaptations, sans être limités par le contexte d'utilisation à l'origine.

Sur la base des principaux aspects d'un algorithme évolutif, nous présentons dans ce qui suit, une brève comparaison entre les différentes classes.

1. Représentation des individus

On ne rapporte aucune différence importante entre les différents types d'algorithmes. L'essentiel étant que la représentation retenue soit adéquate relativement à la fonction d'évaluation et structure des opérateurs appliqués.

2. Opérateurs d'évolution

- *Croisement*

  Ce type d'opérateur est important pour les AG's puisqu'à l'origine de la création des briques élémentaires traduisant des solutions partielles. Concernant les autres types d'algorithmes, ces solutions n'existeraient pas pour des problèmes pratiques. Il est supposé que l'effet de la sélection combiné à la variation génotypique due à la mutation suffisent [Schoenauer & Michalewicz, 1997].

- *Mutation*

  Concernant les AG's, la probabilité d'application et champ d'action (nombre de bits mutés par individu) est statique. Or, pour les autres classes d'algorithmes, des heuristiques d'adaptation sont généralement mises en œuvre de manière fixe, par définition d'un pas de variation, ou alors adaptative par intégration de paramètres de contrôle à la structure des individus.

- *Sélection-Remplacement*

  Leur principe est généralement basé sur la valeur d'adaptation des individus. Les AG's ont tendance cependant à remplacer l'intégralité de la population courante, alors que les autres stratégies en préservent une partie.

La tendance actuelle dans la communauté des algorithmes évolutifs est la fusion et adaptation des différentes techniques empruntées aux différentes classes d'algorithmes de manière à atteindre le meilleur schéma de résolution du problème posé.
A ce titre, notons que les AG's sont les algorithmes évolutifs qui ont connu une plus large utilisation et adaptation [Michalewicz, 1996] [Schoenauer & Michalewicz, 1997].

Dans ce contexte, nous distinguons deux principales approches pour l'adaptation des AG's :

Première approche

Consiste à proposer de nouvelles variantes basées sur le principe de l'AG canonique. En effet, la résolution efficace de problèmes d'optimisation nécessite parfois :

- la définition d'un alphabet non binaire,
- la variation de la taille de la population en cours d'évolution,
- l'intégration de connaissances du domaine dans la structure des opérateurs,
- la coopération de plusieurs sous-populations,
- la considération d'environnements évolutifs dans le temps et non déterministes.

Deuxième approche

Consiste à hybrider un AG avec des méthodes d'optimisation locale. Dans un contexte plus large, Talbi [Talbi, 1999] propose une taxonomie des métaheuristiques hybrides d'évolution, et qui constitue un mécanisme fort intéressant pour la comparaison qualitative entre algorithmes hybrides. L'auteur base sa classification sur différents critères : nombre de métaheuristiques en cours d'exécution, principe d'hybridation (séquentiel, coopératif), nature des métaheuristiques hybridées (homogène, hétérogène).

Chacune de ces approches pose des problèmes spécifiques lors de la mise en œuvre du modèle d'évolution à retenir et ce tant sur le volet théorique (rigueur du modèle, garantie de convergence etc …) que sur le volet pratique (coût de production de la solution, qualité de la solution etc…). La résolution de ces problèmes constitue des pistes de recherche sans doute prometteuses qui élargiront considérablement le champ d'application des AG's en particulier et algorithmes évolutifs en général.

Nous nous intéressons dans ce qui suit, aux AG's. Nous décrivons les principaux concepts y afférents ainsi que leur principe général de leur optimisation.

3. Présentation générale des AG's

Un AG [Holland ,1975] [Goldberg, 1989] a pour but de faire évoluer un ensemble de solutions candidates à un problème posé vers la solution optimale. Cette évolution s'effectue sur la base de transformations inspirées de la génétique, assurant de génération en génération, l'exploration de l'espace des solutions en direction des plus adaptées.

L'approche est fondée sur deux points [Kettaf, 1995] :

- la capacité de représentations simples à encoder des structures complexes,
- l'efficacité de transformations simples pour améliorer de telles structures.

Le processus de résolution d'un problème d'optimisation sous l'angle de la génétique est illustré sur la figure 2.2.

Considérons un problème d'optimisation donné; sa résolution sous l'angle de la génétique se résume par la succession des étapes suivantes :

1. Modélisation

Consiste à identifier le génotype d'une solution candidate à travers un ensemble de caractéristiques, puis d'associer une fonction analytique permettant de mesurer sa capacité à résoudre le problème posé.

2. Génération de la population initiale

Consiste à créer de manière aveugle ou guidée, par application d'heuristiques, la population initiale d'individus

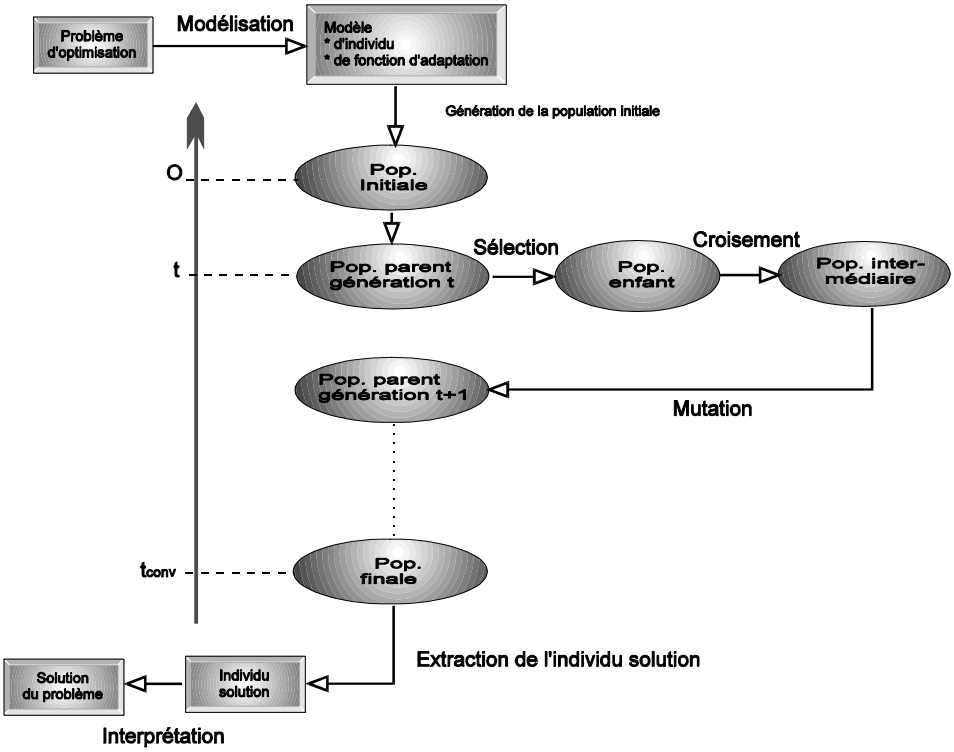


Figure 2.2 : Schéma de résolution d'un problème d'optimisation par AG

3. Sélection

Détermine, par application d'une méthode probabiliste, les individus jugés adaptés et ce, en vue de les cloner à la génération suivante

4 . Croisement

Consiste à appliquer sur la population enfant, un opérateur de combinaison des caractéristiques, avec une probabilité Pc donnée

5. Mutation

Consiste à muter chaque individu issu de la population croisée, avec une probabilité Pm donnée

6. Extraction de l'individu solution

Détermine l'individu solution, caractérisé par la meilleure valeur d'adaptation

7. Interprétation

Consiste à décrire le phénotype de l'individu sur la base de son modèle

Les AG's différent fondamentalement des autres méthodes d'optimisation selon les principaux axes suivants [Goldberg, 1994] :

1. les AG's utilisent un **codage des paramètres**, et non les paramètres eux même,

2. les AG's travaillent sur une **population de points**, au lieu d'un point unique,

3. les AG's n'utilisent que les **valeurs de la fonction étudiée**, pas sa dérivée, ou une autre connaissance auxiliaire,

4. les AG's utilisent des **règles de transition probabilistes**, et non déterministes.

3.1. Concepts de base

La caractérisation du processus d'évolution sous l'angle de la génétique, nécessite sans doute, la transposition des concepts biologiques dans un cadre artificiel. Plus précisément, on se pose les questions suivantes :

Que représente un individu ?
Que signifie l'adaptation d'un individu ?
Comment évoluent les individus ?

Ce paragraphe tente d'expliciter les réponses à ces questions.

3.1.1. Individu et population

Les systèmes génétiques artificiels puisent de la terminologie des systèmes biologiques. Ainsi, un individu étant l'unité fondamentale supportant le matériel génétique en biologie, il représente pour les systèmes artificiels, la structure permettant d'encoder une solution candidate.

Chaque individu ou ***chromosome*** exprimé par un ***génotype***, est constitué d'un ensemble fixe de ***gènes*** représentant chacune de ses caractéristiques. Le décodage d'un individu produit son ***phénotype***. Un gène identifié par sa position appelée ***locus***, peut prendre plusieurs valeurs dénommées ***allèles*** constituant ainsi l'alphabet de l'individu. Initialement, on adopta particulièrement la représentation binaire, ce qui correspond à l'alphabet minimal {0,1}; on parle alors de *version canonique* des AG's. Par la suite, d'autres représentations étendues ont été présentées.

Exemples

1. Optimisation de la fonction $f(x)= x^2$ sur l'intervalle *[a b]*
   L'individu est représenté par un nombre en binaire; sa taille est alors de $Ent(Log_2 b)$
   Soit pour *a= 0*, *b=30*, taille d'un individu est égale à *5*
   Soit l'individu Ind représenté par | 1 | 0 | 0 | 1 | 0 | génotype du nombre *18*

2. Résolution du problème du voyageur de commerce : dans ce problème, un voyageur de commerce hypothétique doit réaliser une tournée complète d'un ensemble de *N* villes en minimisant la distance totale parcourue. Plusieurs codages ont été proposés, on cite particulièrement :
   Représentation par chemin : consiste à représenter un individu (tournée) par une liste ordonnée de villes par lesquelles passe le voyageur de commerce.
   La liste de référence | 1 | 5 | 4 | 3 | 2 | 6 | représente la tournée des villes dans l'ordre *1, 5, 4, 3, 2 et 6*

L'efficacité d'un AG dépend en grande partie du codage retenu pour la représentation des solutions candidates du problème posé. A ce titre, Goldberg [Goldberg, 1994] préconise le respect de deux principes fondamentaux :

1. Principe de pertinence des briques élémentaires

L'utilisateur doit sélectionner un codage de façon à ce que les schèmes[5] courts et d'ordre[6] faible soient pertinents pour le problème sous-jacent, et relativement indépendants des schèmes aux autres positions instanciées.

Cependant, ce principe s'étant avéré difficile à respecter dans le contexte de nombreux problèmes, on atténua le biais du codage par la mise au point d'opérateurs recherchant les bons codages en cours de générations.

2. Principe des alphabets minimaux

L'utilisateur doit choisir le plus petit alphabet qui permette une expression naturelle du problème. Ceci se justifie en effet par le fait que l'adaptation d'un schème est d'autant plus significative et par conséquent plus fiable pour la sélection, que le cardinal de l'espace qu'elle définit est petit.

3.1.3. Fonction d'adaptation

Les transformations qu'opère un AG sur une population d'individus est régie par une mesure de leur adaptation ou capacité à résoudre le problème posé.
Chaque individu solution a une valeur **Fitness** retournée par l'application d'une fonction d'évaluation. Celle-ci agit en deux temps [Preux, 1995] :

[5] Ensemble d'individus qui ont une partie commune de leur code
[6] Taille de la partie commune mesurée en nombre de gènes

1. décodage de l'individu, c'est à dire interprétation de la chaîne de bits[7]. Cela peut être vu comme l'exhibition du phénotype de l'individu,

2. calcul de la valeur de ce phénotype comme une solution au problème, fournissant la performance de l'individu ou capacité d'adaptation.

La fonction d'adaptation ou fonction objectif, est un élément de réflexion fondamental lors de la modélisation d'un AG car elle définit les contours de l'environnement dans lequel évolue la population d'individus. Cette fonction doit être capable de favoriser la sélection d'individus dans la direction de l'optimum qui est, à priori, inconnue.
A ce titre, Mansanne & al [Mansanne & al, 1999] montrent à travers un cas pratique dans le domaine de la géodésie, la nécessité de formaliser prudemment la fonction d'adaptation . En effet, les auteurs montrent que la non considération de critères évidents pour les experts en la matière, lors de l'expression de la fonction, a fait aboutir l'AG à une solution « absurde », donnant la répartition optimale de la vélocité souterraine.
En cas de difficulté de formalisation de tous les critères, les auteurs proposent le bornage de l'espace de recherche.

3.1.3. Opérateurs génétiques

Les opérateurs génétiques représentent des procédures de transformation des individus entre deux générations. Les AG's exploitent principalement trois types d'opérateurs visant chacun d'eux un objectif spécifique relativement à la couverture de l'espace des solutions. Ces opérateurs sont la sélection, le croisement et la mutation.

3.1.3.1. Sélection

La sélection est le premier opérateur génétique appliqué à une population d'individus en vue de la renouveler. Cet opérateur base la constitution de la nouvelle population sur le Fitness des individus de la population qui la précède.
Le principe de la sélection est tel que les individus les mieux adaptés fournissent la descendance la plus nombreuse. Notons que l'opération de sélection couvre au sens que nous évoquons, deux étapes :

Première étape **:** correspond au **clonage**. Consiste en la reproduction de copies intégrales d'un individu; le nombre de copies dépend de sa performance relative dans la population,

Deuxième étape **:** correspond à la **sélection** proprement dite. Consiste en l'intégration d'individus clonés dans la nouvelle population.

[7] Dans le cas d'un codage binaire

La sélection peut être réalisée selon différentes méthodes, dont nous citons principalement:

- Méthode de la roulette

Connue également sous le nom de « *Roulette Wheel Selection* » [Goldberg, 1983]. Cette méthode est ainsi dénommée car elle consiste à attribuer à chaque individu, un secteur de la roue de loterie proportionnel à son fitness relatif. La sélection de *N* individus, *N* étant la taille de la nouvelle population, est réalisée en effectuant *N* tirages de la roue biaisée. En pratique, ceci revient à calculer pour chaque individu, une probabilité p_i de survie proportionnelle à sa performance et calculée comme suit :

$$p_i = \frac{Fitness(Ind_i)}{\sum_{j=1}^{N} Fitness(Ind_i)} \qquad 0 < p_i < 1$$

On effectue alors un calcul d'une probabilité de sélection $q_i / q_i = \sum_{j=1}^{i-1} P_j$ puis on génère aléatoirement un nombre *r* sur l'intervalle *[0 1]*, *N* fois de suite. Un individu Ind_i est sélectionné lorsque $q_{i-1} < r < q_i$.

La méthode de la roulette présente l'inconvénient majeur suivant : s'il existe un individu de performance relativement dominante par rapport aux autres individus de la population, les générations suivantes compteront essentiellement des descendants hybrides de cet individu. Ceci, restreint de fait, l'espace des solutions exploré et l'algorithme risque alors d'être piégé dans un optimum local.

- Méthode du rang

Connue également sous le nom de « *Ranking Fitness* ». Cette méthode a été proposée par J.BAKER en 1985 [Baker, 1985] afin de pallier aux problèmes soulevés par la méthode de la roue de loterie. La méthode du rang consiste principalement à ordonner les individus en fonction de leur performance, et ce du meilleur, de rang *1*, jusqu'au moins performant, de rang *N*. La probabilité de sélection d'un individu de rang i est calculée comme suit [Preux, 1995] :

$$Ps(i) = D_{max} - (D_{max} - D_{min}) * \frac{(i-1)}{(N-1)}$$

Où :

Ps(i) : Probabilité de sélection de l'individu de rang i

D_{max} : Nombre maximal de descendants par individu

D_{min} : Nombre minimal de descendants par individu

Avec :

$D_{min} = 2 - D_{max}$, $0 \leq D_{max} \leq 2$

Chaque individu Ind_i engendre un nombre de clones calculé selon la formule

$$V(Ind_i) = [Ps(i) + 1((Ps(i) - [Ps(i)]) \geq r_i)]$$

Où :

r_i : Nombre aléatoire compris entre 0 et 1

1(P) : Prédicat retournant 1 si P vrai, 0 sinon

[Ps(i)] : Retourne la partie entière de Ps(i)

La méthode ainsi définie empêche la dominance d'un individu en préservant des proportions de descendants adéquates. En outre, elle offre une prédisposition à une exécution massivement paralléle des AG's [Dejong & Sarma, 1995]

3.1.3.2. Croisement

Le croisement[8] est le deuxième opérateur génétique appliqué à la population d'individus enfants issue de la sélection. Cet opérateur consiste, par analogie aux systèmes biologiques, à effectuer un échange de matériel génétique entre individus choisis avec une probabilité *Pc*. C'est un opérateur de combinaison qui agit généralement par paire d'individus en déterminant un ou plusieurs points de coupure, délimitant la frontière des parties à échanger. On distingue principalement le croisement à points et le croisement uniforme.

- Le croisement à point (s)

Les individus sont coupés en un ou plusieurs points aléatoires dits **site(s) de croisement**. Les segments situés à partir de l'extrémité (cas d'un croisement à un point) ou entre les points (cas d'un croisement à plusieurs points), sont échangés entre eux.

La figure 2.3 présente le principe de croisement à un point.

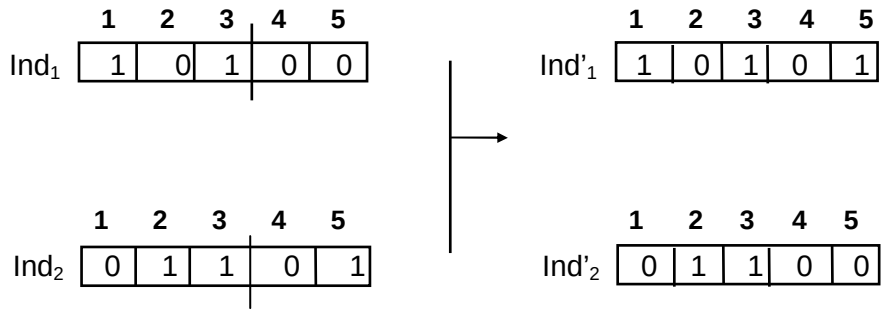


Figure 2.3 : Principe du croisement à un point

Echange des valeurs des gènes situés après le site de croisement 3 (00 et 01), entre les individus Ind_1 et Ind_2

[8] Connu également sous le nom de Cross Over

- Le croisement uniforme

Cet opérateur exploite une chaîne binaire masque générée aléatoirement. Un gène est échangé entre la paire d'individus sélectionnés pour le croisement, si le gène à la même position dans la chaîne masque a pour allèle la valeur *1*. Dans le cas contraire,

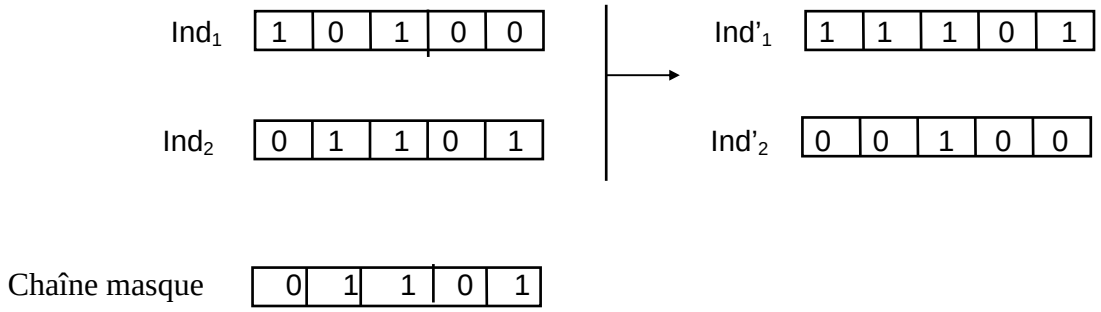


Figure 2.4 : Principe du croisement uniforme

La chaîne masque indique qu'il faut échanger les valeurs de gènes situées aux positions 2 (1 et 0), 3 (1 et 1) et 5 (0 et 1) entre les individus ind_1 et ind_2

l'échange n'est pas effectué. La figure 2.4. présente le principe du croisement uniforme.
En réponse à des besoins suscités par des applications spécifiques, de nombreuses variantes du croisement ont été mises au point afin de :

- garantir l'intégrité des individus résultats du croisement,
- exploiter une connaissance auxiliaire du domaine de l'application dans le but d'améliorer l'exploration de l'espace des solutions,
- intégrer des concepts récents issus des systèmes génétiques naturels.

3.1.3.3. Mutation

La mutation s'applique à un individu issu de la population ayant subi le croisement. Cet opérateur consiste à modifier un gène selon une probabilité Pm généralement inférieure à la probabilité de croisement Pc ($Pm \approx Pc/100$).
La figure 2.5. Présente le principe de mutation.

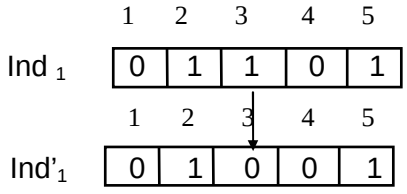


Figure 2.5 : Principe de la mutation

Le gène situé à la position 3 de l'individu Ind_1 (valeur 1) a été muté dans l'individu Ind'_1 (valeur 0).

On admet que la mutation joue un rôle secondaire mais nécessaire dans la mise en oeuvre des AG's. Alors que le croisement a pour but de recombiner efficacement les informations portées par des individus parents, l'opérateur de mutation nous permet quant à lui, de se prémunir contre la perte prématurée d'allèles, origine du phénomène de dérive génétique[9] et d'explorer aléatoirement de nouvelles régions de l'espace des solutions.

3.2. Analyse formelle

La mise en oeuvre des AG's semble assez aisée. On admet qu'ils représentent des procédures classiques, stochastiques puisant des principes approuvés de l'exploration humaine. Toutefois, la crédibilité scientifique impose une justification rationnelle du processus d'optimisation qu'ils préconisent.

A cet effet, nous distinguons trois types de travaux sur la formalisation des AG's. Les premiers travaux de Holland et Goldberg [Holland, 1975] [Goldberg, 1989] ont eu pour objet d'étayer les fondements mathématiques des propriétés fondamentales des AG's dans leur version canonique.

Par la suite, des travaux [Ankenbrandt, 1990][Cerf, 1994] se sont intéressés à leur preuve de convergence. Enfin, une autre catégorie de travaux [Goldberg, 1989][Hartman & Belew, 1991] ont développé une théorie de la complexité génétique qui caractérise les fonctions faciles ou difficiles à optimiser par un AG et ce, par analogie aux problèmes P et NP complexes pour les autres méthodes d'optimisation.

Nous présentons dans ce qui suit, la preuve détaillée du théorème fondamental des AG's.

3.2.1. Analyse par schème et théorème fondamental

Le théorème fondamental des schèmes [Holland, 1975] explique, à l'aide de faits mathématiques, la rigueur des résultats d'un AG. Le théorème formalise plus particulièrement la notion de « briques élémentaires », bases de construction adaptative des solutions retournées par un AG. L'exposé du théorème fondamental exploite des concepts que nous définissons dans ce qui suit.

3.2.1.1. Eléments de base

- Schème

Un schème est un motif de similarité décrivant un sous-ensemble de chaînes avec des similarités à des positions définies. Un schème est construit sur la base de l'alphabet de la population, étendu par le caractère joker # qui peut être remplacé par tout caractère de l'alphabet.

[9]Phénomène dû au caractére fini de la population; traduit la disparition de certaines allèles dans la population du fait de la non sélection répétée des individus qui les portent et d'un taux de mutation relativement faible

Exemple

Schème binaire : *# 0 1 0 1* peut être remplacé par les chaînes *0 0 1 0 1* et *1 0 1 0 1*

Un schème décrit ainsi une région de l'espace de recherche et de fait, un motif de similarité entre chaînes construites sur un alphabet fini.

On peut noter dès à présent que les AG's manipulent des chaînes appartenant à des schèmes différents et par conséquent à des régions différentes de l'espace des solutions. Ceci leur confère la puissante propriété connue sous le parallélisme implicite (Cf 3.2.2)

- Ordre d'un schème

L'ordre d'un schème *S* noté *o(S)* représente le nombre de caractères différents du caractère #

Exemple *S = # 1#011 o(S) = 4*

- Longueur utile d'un schème

La longueur utile d'un schème *S* notée $\delta(S)$ est la distance entre la première et la dernière position instanciées du schème.

Exemples *:* *S = # # 0 1 0 , $\delta(S) = 2$*

S = # 1 # 0 1 1 , $\delta(S) = 4$

- Adaptation d'un schème

L'adaptation d'un schème est la moyenne des adaptations de tous ses représentants dans la population. En clair, l'adaptation d'un schème rend compte de la distribution qualitative des individus appartenant à la région de l'espace associée.

3.2.1.2. Le théorème fondamental

Les concepts que nous venons de définir, sont des éléments simples qui nous permettent d'analyser l'effet des opérateurs génétiques sur le processus de convergence d'un AG vers des schèmes performants.

Notons :

N(S,t) : Nombre d'exemplaires du schème S à la génération t

F (S,t) : Adaptation d'un schème S à la génération t

Ind_i^t : ième individu de la population à la génération t

Taille_Pop : Taille de la population

On calcule la probabilité de sélection d'un schème comme suit [Goldberg, 1994] :

$$P_{Select} = \frac{F(S,t)}{F(t)} \qquad (1)$$

Où :
F(t) : Valeur totale des adaptations des individus de la population à la génération t
Avec :

$F(t)= \sum_{i=1}^{Taille-Pop} F(Ind_i^t)$, $F(Ind_i^t)$ *: Valeur d'adaptation de l'individu* Ind_i^t

L'expression de l'effet de la reproduction sur le nombre attendu de schèmes est la suivante :

$$N(S,t+1)=N(S,t)*Taille\_Pop*F(S,t)/F(t) \qquad (2)$$

En effet, la sélection clone les individus en fonction de leur performance; par conséquent un schème se développe au rythme du rapport de l'adaptation moyenne de la population par rapport à l'adaptation de la population.

Durant la phase de sélection, chaque individu est reproduit avec un nombre de copies qui dépend de sa valeur d'adaptation. Le nombre d'individus du schème *S*, attendu à la génération *t+1* est calculé comme suit :

$$N(S,t+1)=N(S,t)*F(S,t)/\overline{F(t)} \qquad (3)$$

Où :
$\overline{F(t)}$: Adaptation moyenne de la population à la génération t
Avec :

$\overline{F(t)}=\frac{F(t)}{Taille\_Pop}$

En clair, les schèmes dont la valeur d'adaptation est supérieure à la moyenne recevront plus de copies que les schèmes dont la valeur d'adaptation est au dessous de la moyenne.
Soit le schème *S* d'adaptation supérieure à la moyenne de valeur, $C^* \overline{F(t)}$ *avec* $C > 0$. En réecrivant l'équation (3), on obtient :

$$N(S,t+1)=N(S,t)*\overline{F(t)}+C*\overline{F(t)}/\overline{F(t)} \quad =(1+C)*N(S,t) \qquad (4)$$

en commençant à l'instant $t= 0$ et en supposant que *C* est constant, on obtient :

$$N(S,t+1)=N(S,0)*(1+C)^t \qquad (5)$$

Il en ressort que le nombre de copies d'un schème d'adaptation supérieure (resp. inférieure) à la moyenne, croît(resp. décroît) avec une variation exponentielle. Le croisement intervient à ce niveau pour échanger les informations entre les chaînes afin d'explorer de nouvelles régions de l'espace.
Examinons à présent les effets des opérateurs génétiques sur les schèmes. La longueur d'un schème intervient dans la probabilité de destruction suite à un croisement. Un long schème a plus de chances d'être détruit qu'un schème de longueur plus petite.
On calcule la probabilité de destruction d'un schème de longueur *l* comme suit [Goldberg, 1994]

$$P_d = \frac{\delta(S)}{(l-1)} \qquad (6)$$

et par conséquent, sa probabilité de survie est :

$$P_s = 1 - \frac{\delta(S)}{(l-1)} \qquad (7)$$

L'opérateur de croisement intervient avec une probabilité *Pc*; la probabilité de survie d'un schème est alors :

$$P_s(S) = 1 - Pc * \frac{\delta(S)}{(l-1)} \qquad (8)$$

L'effet combiné de la sélection et du croisement peut alors s'exprimer comme suit :

$$N(S,t+1) \geq N(S,t) * \frac{F(S,t)}{\overline{F(t)}} * (1 - (Pc * \frac{\delta(S)}{(l-1)})) \qquad (9)$$

L'expression *N(S,t+1)* devient une inégalité car la probabilité de survie est une minoration de la probabilité de survie réelle. La formule *(4)* ne tient pas compte en effet du cas de parents identiques, pour lequel, le schème survit sûrement [Kettaf, 1995].

Le schème *S* se développe avec un facteur multiplicatif qui dépend de deux faits :
- le schème a une adaptation au dessus ou au dessous de la moyenne,
- le schème a une longueur relativement courte (il est moins probable qu'il soit détruit).

Considérons à présent l'opérateur de mutation appliqué avec un taux *Pm*. Ce dernier, peut éventuellement provoquer la destruction d'un schème. La probabilité de survie d'un gène est *1 - Pm* . Il s'ensuit que la probabilité de survie de tous les gènes et donc du schème, est : $(1 - Pm)^{0(S)}$
La probabilité de mutation étant très faible, on peut effectuer l'approximation suivante :

$$P_s = (1 - Pm)^{0(S)} = 1 - o(S) * Pm \qquad (10)$$

En intégrant l'effet de la mutation à l'équation (9), on obtient [Goldberg, 1994]:

$$\mathbf{N(S,t+1) \geq N(S,t) * F(S,t) / F(t) * (1 - (Pc * (\delta(S)/ (l - 1) - o(S) * Pm)))} \qquad (11)$$

En définitive, on montre un résultat de grande portée :

Les schèmes courts, d'ordre faible, font l'objet d'un nombre de tests exponentiellement croissants dans les générations suivantes [Goldberg, 1994]. Ces schèmes sont qualifiés de **briques élémentaires**.
Notons toutefois que le théorème des schèmes analyse globalement le processus d'évolution d'un AG mais ne précise rien concernant sa convergence qui a été le centre de réflexion d'autres travaux [Aarts, 1989] [Cerf, 1994].

Par ailleurs, N.Radcliffe [Radcliffe, 1991a] a proposé une extension du théorème des schèmes dans le cas d'un espace de recherche, produit cartésien d'espaces finis $\Phi = \Phi_1 x....x \Phi_n$. L'équivalent de la notion de schème est appelée *forme*. Une forme H est alors une chaîne $X_1 ...X_n$ où X_i est une valeur définie dans Φ_i où indéfinie #. En imposant des conditions de clôture des formes, fermeture du croisement, ergodicité du croisement et mutation, Radcliffe démontra le théorème des formes de façon analogue au théorème des schèmes.

En définitive, le résultat fondamental est d'une part, qu'un schème pertinent est d'autant plus aisément découvert que l'adaptation de ses représentants ne dépend pas du contexte. D'autre part, un schème est d'autant plus résistant à la destruction sous l'effet d'opérateurs génétiques, qu'il est court et comporte peu de positions définies [Sebag & Schoenauer, 1996].
Cependant, il existe des problèmes trompeurs dits **AG-difficiles** ou ***« deceptive problems »*** qui mettent à défaut l'hypothèse des briques élémentaires. Ces problèmes déroutent l'algorithme en le dirigeant vers un point autre que la solution escomptée. Cette notion de problème trompeur a été définie dans la version canonique des AG's [Goldberg, 1989] puis étendue par Radcliffe [Radcliffe, 1991b] au cas d'une représentation quelconque. De façon simple, illustrons cette notion par l'exemple suivant :
Soit l'espace de recherche $\Phi = \{0,1\}^3$, la fonction d'adaptation F définie par :

$$F(x) = \begin{cases} 3, si.x = 111 \\ 2, si.x \in 0\#\# \\ 0, \text{sinon} \end{cases}$$

et les schèmes définis par $H_1 = 1\#\#$, $H_2 = 0\#\#$
F est alors une fonction trompeuse puisqu'on a l'optimum (3 dans ce cas) qui appartient au schème H_1 mais $F(H_1) = ¾ < F(H_2) = 2$
L'exploration génétique risque d'être ainsi piégée dans la région sous-optimale H_2. Formellement, une fonction F est dite partiellement trompeuse s'il existe des schèmes d'ordre K de performances supérieure à celles des schèmes de même ordre, contenant les optima globaux. F est dite globalement trompeuse si tout schème d'ordre K est de performance supérieure à celle des schèmes de même ordre contenant les optima globaux. Toutefois, Grefenstette [Grefenstette & al, 1985] soutient l'idée que la dynamique de l'évolution étant fondée sur une performance moyenne observée à travers

des générations successives, l'algorithme est ainsi armé de fortes chances d'échapper aux « trappes » des problèmes trompeurs.

3.2.2. Propriétés générales

Les propriétés générales d'un AG sont principalement les suivantes :

1. Parallélisme implicite

En manipulant une population de taille *N*, un AG traite efficacement un nombre de directions de recherche de l'ordre de N^3. Ce résultat dû à Holland [Holland, 1975] et confirmé par Godberg [Goldberg & Lingle, 1985], traduit la propriété fondamentale des AG's, connue sous le qualificatif de parallélisme implicite

Illustrons les grandes étapes du raisonnement permettant d'aboutir à cette estimation.

Soit une population de *N* individus de taille *l*, *N(S)* le nombre de schèmes. Considérons les schèmes ayant une probabilité de survivre au croisement, supérieure à une probabilité *Ps*. Ces schèmes ont une longueur de définition strictement inférieure à $ls= (1- Ps)(l-1)+1$. Goldberg estime le nombre de schèmes de longueur inférieure à *ls* construits à partir d'une chaîne binaire donnée de longueur *l* par $2*ls - 2\ (l - l_s +1)$. L'estimation d'une majoration du nombre de tels schèmes dans la population donne alors $N(S) = n* 2*ls-2\ (l-ls +1)$. Dans le but d'annuler l'effet d'une répétition de comptage d'individus, on *considère* $N=2*ls/2$; l'estimation de *N(S)* devient alors : $N(S) > N^3\ (l - ls + 1)/ 4$ et donc $N(S) > O (N^3)$

En évaluant n individus, l'*AG* canonique considère implicitement au moins N^3 directions de recherche. Davidor [Davidor, 1990] conteste toutefois ces résultats en arguant des erreurs d'échantillonnage susceptibles d'être commises par l'algorithme.

2. Equilibre entre exploration et exploitation

L'algorithmique génétique résout un problème qui résiste depuis longtemps aux méthodes de programmation classique : la détermination d'un **équilibre** entre l'**exploration** et l'**exploitation** [Holland, 1992]. Le mot équilibre est justifié par le fait que les deux procédures sont antagonistes. L'exploitation d'une direction de recherche consiste essentiellement à encourager l'apparition de ses représentants dans la population, tandis que l'exploration plaide en faveur de nouvelles directions de recherche. Une forte exploitation conduit à une convergence prématurée à fortiori vers un optimum local; à l'inverse, une forte exploration conduit, du fait d'un balayage hâtif de l'espace de recherche, à une lente convergence.

L'AG apporte une solution élégante à ce dilemme et ce, rappelons le, en allouant un nombre d'essais exponentiellement croissant à la meilleure direction observée [Goldberg, 1994]. Ceci permet en effet de minimiser les pertes dues à l'exploration de mauvaises directions.

A ce titre, notons que la sélection est orientée vers l'exploitation seule. Le croisement et la mutation permettent tant l'exploration que l'exploitation avec la différence

suivante : les enfants obtenus par croisement sont en général loin des parents mais appartiennent à une région réduite de l'espace de recherche. Par opposition, les enfants obtenus par mutation sont en général proches du parent mais peuvent être situés dans tout l'espace de recherche [Sebag & Schoenauer, 1996].

3. Non optimalité

Un AG ne garantit pas de trouver l'optimum global de la fonction d'adaptation associée. La pratique a toutefois montré que l'algorithme aboutit à une solution appréciable, proche de l'optimum global [Venturini, 1996].

4. Epistasie

L'épistasie est un phénomène fort connu de la génétique naturelle; il traduit la non additivité des performances des allèles dans le génotype. Ceci signifie que l'adaptation d'un individu ne varie pas linéairement avec son génotype; des combinaisons d'allèles présentes dans un génotype peuvent modifier de façon considérable la performance de l'individu comparativement à un individu de génotype quasi-similaire [Goldberg, 1994].

5. Adaptabilité dans le temps

L'exploration étant éventuelle (probabilité non nulle) dans toute direction de recherche, les AG's sont ainsi capables de retourner un optimum variable dans le temps [Cobb & Grefenstette, 1993].

3.2.3. Convergence d'un AG

La preuve de convergence des AG's a été le centre d'interêt d'un nombre considérable de travaux. Si on définit la diversité génotypique Δ d'une population P_t comme suit [Preux, 1995] :

$$\Delta(P_t) = \sum_{\{(Ind1, Ind2) \in PtxPt\}} \delta(Ind_1, Ind_2)$$

Où :

$\delta(x,y)$: Distance de Hamming entre x et y

Ind_1 , Ind_2 : Individus de la population

et si l'on mesure la diversité génotypique de la population en cours du temps, on observe que celle-ci diminue rapidement en début d'évolution pour stagner ensuite. Cette diminution s'explique par la phase de clonage qui reproduit, selon une stratégie élitiste, les « bons » individus. La diversité une fois largement diminuée, demeure relativement stable ce qui traduit la convergence.

Aarts & al [Aarts & al, 1989] prouve , en utilisant la théorie des chaînes de Markov, la convergence des AG's sous conditions que les stratégies de reproduction et sélection soient élitistes.

Carol Anckenbrandt [Anckenbrandt, 1990] a étudié les théorèmes de convergence pour l'AG canonique sans la mutation; il a abouti à la formalisation de l'ordre de la solution produite par l'algorithme et ce comme suit :

$$O\ (\ |complexité\ de\ calcul\ de\ F|\ *n\ log(l))/\ Log(r)$$

Où :

F : Fonction d'adaptation

l : Longueur d'un individu

r : Rapport de qualité où $r = Min\ (\ F1(Ii)\ /\ F0(Ii)$

Avec :

$F_1(Ind_i)$ *(resp.* $F_0(Ind_i)$ *)* moyenne d'adaptation des individus ayant la valeur du gène à la position i égale à (resp. à 0)

$$F_0(Ind_i)=\frac{1}{\left|Ind_i^0\right|}\sum_{Ind\in Ind_i^0}F(Ind)\ ,\quad F_1(Ind_i)=\frac{1}{\left|Ind_i^1\right|}\sum_{Ind\in Ind_i^1}F(Ind)\ ,\quad Ind_i^v = \{Ind/\text{ valeur du gène à la position i est v}\}$$

Ce théorème prouve la convergence de l'algorithme mais pas nécessairement vers l'optimum. R.Cerf [Cerf, 1994] démontra effectivement la convergence de l'algorithme vers l'optimum sous les conditions que la taille de la population soit suffisamment importante et que le processus de sélection soit caractérisé d'élitiste.

Par ailleurs, un AG est caractérisé par sa rapidité de convergence, auquel cas on qualifie la convergence lente ou à l'inverse prématurée. Pour illustrer intuitivement ces phénomènes, notons en premier lieu, que la pression sélective mesure l'écart entre le nombre de clones de l'individu de meilleure valeur d'adaptation et nombre de clones de l'individu de plus faible valeur d'adaptation, à une génération donnée. Lorsque la pression sélective est élevée, ce qui traduit la présence d'un individu de performance relativement importante, la population a tendance à être dominée par ce dernier et les opérateurs génétiques ne peuvent y apporter du nouveau (échange entre individus quasi-similaires). Aussi, la population convergera t-elle rapidement vers un optimum local.
A l'inverse, lorsque la pression sélective est faible, ce qui traduit une distribution presque uniforme d'individus de faible performance et individus de forte performance dans la population, la sélection devient latente. En ce sens, l'algorithme ne favorise pas particulièrement la constitution de meilleurs individus, ce qui rend la convergence lente.

3.3. Heuristiques d'adaptation d'un AG

De nombreuses heuristiques ont été mises en œuvre dans le but de réguler l'évolution d'un AG en adaptant ses éléments à la nature du problème posé
L'idée est en effet, d'utiliser la notion d'adaptation véhiculée par les AG's, non pas seulement pour trouver la meilleure solution à un problème mais à un niveau d'abstraction plus élevé, d'adapter les éléments de l'AG aux particularités du problème posé.

Ces heuristiques portent essentiellement sur la fonction fitness, opérateurs génétiques et paramètres de contrôle.

3.3.1. Adaptation de la fonction fitness

L'adaptation porte dans ce cas, sur l'utilisation de techniques d'ajustement de la fonction fitness. On décrit dans ce qui suit les techniques de changement d'échelle, de nichage et spéciation et enfin intégration de critères.

1. Le changement d'échelle[10]

Consiste à ajuster la fonction d'adaptation dans le but de maintenir un écart de performances entre les individus de meilleure valeur d'adaptation et les individus de valeur d'adaptation moyenne. Son principe est le suivant : procéder à une transformation linéaire de la fonction d'adaptation $f' = A * f + B$

Où :
f ': Fonction fitness ajustée
f : Fonction fitness à ajuster
A et B : Paramètres à déterminer tels que :
- la moyenne d'adaptation transformée est égale à la moyenne d'adaptation initiale
- le rapport entre le nombre de clones de l'individu de meilleure valeur d'adaptation et nombre de clones des individus de valeur d'adaptation moyenne, est un facteur de contrôle C_{mult} tel que

$f'max = C_{mult} * f'moy$

Avec :

C_{mult} *: Nombre de copies souhaité pour l'indindividu de meilleure valeur d'adaptation,*

$0 < Cmult < 2$

f'_{moye} *: Adaptation moyenne de la population en utilisant l fonction d'adaptation ajustée f'*

On trouvera dans [Michalewitcz, 1996] d'autres possibilités de mise à l'échelle.

2. Technique de nichage et spéciation

On peut représenter une niche écologique comme la fonction ou le rôle d'un organisme dans un environnement donné. On peut envisager l'espèce comme une classe d'organismes ayant des caractéristiques communes.

Une population d'individus peut être ainsi répartie en sous-populations ou espèces situées à des domaines ou niches différentes. Ces dernières sont considérées comme des points ressources que se partagent les individus qui y sont présents. Il s'en suit que la ressource par individu diminue d'autant plus que la niche associée est peuplée; ceci incite alors la migration d'individus vers d'autres niches.

Sur la base de cette inspiration écologique, des techniques appropriées améliorent l'exploration de l'espace des solutions d'un problème d'optimisation de fonctions multimodales, caractérisées par la présence de plusieurs optimums.

[10] Connue également sous l'expression anglaise « Fitness Scaling »

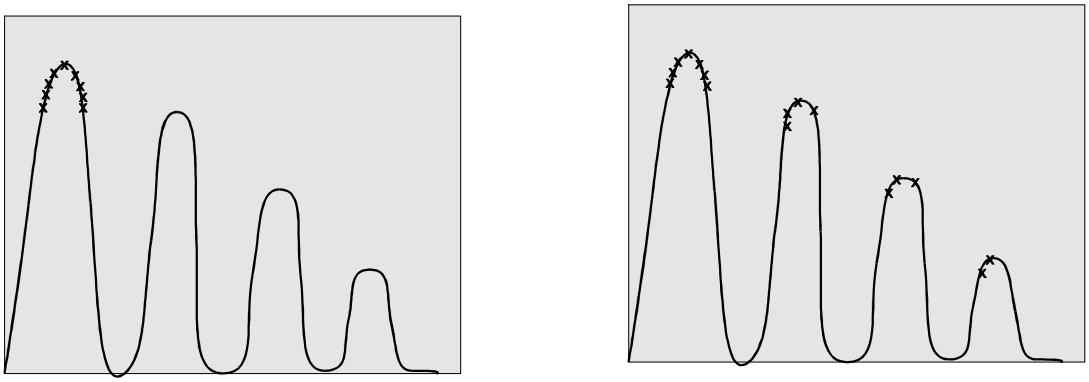
Dans ce cadre, des méthodes de formation d'espèces et niches écologiques ont été mises au point afin d'être exploitées dans un AG.

Goldberg et Richardson [Goldberg & Richardson, 1987] proposent d'ajuster la fonction d'adaptation par un facteur lié au nombre d'individus dans un voisinage

$$f'(Ind) = f(Ind) / | (Ind'/ \sigma(Ind,Ind') < seuil|$$

Où :
f'(Ind) : Fonction fitness ajustée appliquée à l'individu Ind
f(Ind) : Fonction fitness à ajuster appliquée à l'individu Ind
σ(Ind, Ind') : Distance de Hamming entre les individus Ind et Ind'

Ainsi, plus une sous-population est nombreuse, plus le fitness de ses individus se dégrade. Ceci encourage alors la reproduction d'individus dans d'autres niches et conduit à la découverte et conservation de plusieurs solutions optimales. Théoriquement, le partage permet de déterminer les principaux pics de la fonction à à optimiser, le nombre de représentants d'un pic étant inversement proportionnel à la hauteur relative du pic.
La figure 2.6. illustre le principe de partage.



(a) Fonction d'adaptation sans partage (b) Fonction d'adaptation avec partage

Figure 2. 6: Optimisation d'une fonction multimodale par application d'une fonction d'adaptation avec et sans partage

3. Intégration de critères

L'expression de la fonction fitness utilise dans ce cas précis, des critères secondaires, non directement liés à la mesure de la qualité des individus.
Dans [Bean & Hadj-Alouane, 1992], les auteurs intègrent à la fonction fitness, une mesure de pénalité des individus, basée sur le feedback des individus des générations précédentes et ce comme suit :

$$f'(Ind_i) = f(Ind_i) + \lambda(g) \sum_{j=1}^{Taille\_Pop} f_i^2(Ind_i)$$

Où :

$$\lambda(g+1) = \begin{cases} (1/\beta 1) * \lambda(g) \ si \ Ind^* \in F \ \forall g-k+1 \le i \le t \\ \beta 2 - \lambda(g) \ si \ Ind^* \in S-F \ \forall g-k+1 \le i \le t \\ \lambda(g) \ sinon \end{cases}$$

Avec :
f'(Ind) : Fonction fitness ajustée appliquée à l'individu Ind
f(Ind) : Fonction fitness à ajuster appliquée à l'individu Ind
S : Espace de recherche global
F : Espace des solutions réalisables (ie possibles en pratique)
Ind^* : Meilleur individu de la génération courante
g : Numéro de génération courante
k, β1,β2 : Constantes.

En fait, la stratégie est de décroître la pénalité si tous les individus des *k* dernières générations sont réalisables, la croître dans le cas contraire et enfin la stabiliser en cas d'équilibre.

Duvivier & al [Duvivier & al , 1998] proposent l'intégration de critères secondaires pour l'évaluation des individus, à une étape où l'algorithme atteint un *plateau*, qui traduit la quasi-égalité de leur valeur d'adaptation sur un voisinage de recherche.
A cet effet, les auteurs définissent le pouvoir de discrimination d'un critère :

$$\varphi\#(c) = \overline{\phi\#(c)}$$

Où :

$\phi\#(c)$=Nombre de valeurs différentes prises par les individus pour le critère c.

$\overline{\phi\#(c)}$: Nombre moyen de $\phi\#(c)$ pour un nombre moyen d'individus de la population

La fonction fitness est ajustée comme suit :

$$f'(Ind) = f(Ind) + \sum_{i=1}^{C} \alpha_i C_i(Ind)$$

Où :
f'(Ind) : Fonction fitness ajustée appliquée à l'individu Ind
f(Ind) : Fonction fitness à ajuster appliquée à l'individu Ind
C : Nombre de critères considérés
Ci(Ind) : Valeur du critère Ci prise par l'individu Ind
αi : Poids du critère C_i fixé manuellement en fonction de son importance

Les expérimentations réalisées pour la résolution du problème du Job shop ont permis de conclure sur l'intérêt de la corrélation entre critères retenus et fonction fitness brute.

3.3.2. Adaptation des opérateurs

Dans ce cadre, l'idée clé est d'intégrer aux opérateurs génétiques classiques, une connaissance issue du domaine d'application.

Grefenstette et al [Grefenstette & al, 1985] ont élaboré un croisement heuristique pour résoudre le problème du voyageur de commerce. Basé sur une représentation en proximité, l'opérateur construit un descendant à partir de deux tournées parentes selon un principe que nous décrivons par l'algorithme suivant :

Début
Choisir au hasard une ville initiale
Intégrer la ville choisie dans la tournée fille
Ville_Courante=Ville-Choisie
Tant que *la tournée est incomplète* **faire**
Comparer les trajets quittant la ville courante à partir des tournées parentes
Retenir la ville V_{opt} *la plus proche*
Si V_{opt} *existe déjà dans la tournée* **Alors**
{ formation d'une boucle}
Intégrer dans la tournée une ville V absente
Ville_Courante = V
Sinon
Intégrer V_{opt} *dans la tournée*
Ville_Courante=V_{opt}
Finsi
Fin
Fin

Des travaux récents [Carvalho & Freitas, 2000] proposent l'application d'un nouvel opérateur augmenté par la connaissance et qui ne constitue ni un croisement, ni une mutation.

Les auteurs s'intéressent à la découverte de règles d'inférence intéressantes, à faible disjonction, à partir d'une base de données (data mining).

L'opérateur est basé sur des éléments de la théorie de l'information, et appliqué selon les étapes suivantes :

1. Calculer le gain d'information associé à chaque condition exprimée par l'individu

$$Info\_Gain(C)=Info(G)-Info(G/C)$$

Où :

$$Info(G)=-\sum_{j=1}^{Cl}|Gj|/|T|*\log_2(|Gj|/|T|)$$

Avec :
G : Attribut but de la classe
Cl : Nombre de classes (valeurs de G)
|Gj| : Nombre de tuples d'apprentissage ayant la jème valeur du domaine de G
|T| : Nombre total de tuples d'apprentissage
$|V_i]$: Nombre de tuples d'apprentissage qui satisfont la condition d'une règle d'inférence de la forme $<A_i\ Op_i\ V_i>$ avec A_i : ième attribut prédicat, Op_i : opérateur relationnel, V_j : valeur réelle
$|V_{ij}|$: Nombre de tuples d'apprentissage qui satisfont la condition $<A_i\ Op_i\ V_i>$ et qui ont la jème valeur de l'attribut G
$|\neg\ V_i|$ :: Nombre de tuples d'apprentissage qui ne satisfont pas la condition $<A_i\ Op_i\ V_i>$
$|\neg\ V_{ij}|$: Nombre de tuples d'apprentissage qui satisfont la condition $<A_i\ Op_i\ V_i>$ et qui ont la jème valur de l'attribut G

2. Sélectionner la condition C_{min} telle que :

$$Info\_Gain(C_{min}) = Min(Info\_Gain(C)) \forall C \in Individu\ règle$$

3. Eliminer C_{min} de l'expression de l'individu règle

4. Réevaluer les conditions exprimées par la règle

Ces étapes sont itérées jusqu'à atteindre une taille minimale de la règle ou que le nombre d'itérations dépasse la taille maximale de la règle.
Les expérimentations réalisées sur la base de données standard UCI [http://www.ICS.uci.edu/~mlearn/MLRepository.html] à *48842* tuples et *14* attributs, ont montré l'intérêt de l'application de cet opérateur.

3.3.3. Adaptation des paramètres de contrôle

La stratégie consiste à adopter une méthode qui permet de déterminer des valeurs fixes ou évolutives des paramètres de l'AG [Freitas, 1999] [Michalewitz, 1996] [Yang & Korfhage, 1993] : taille de la population, nombre de générations, probabiltés d'application des opérateurs, structure et masque des opérateurs etc…

Freitas [Freitas, 1999] propose un pas de variation progressif pour la mutation, dépendant linéairement du nombre de clones du meilleur individu de la génération courante. Les expérimentations réalisées sur une base de test standard en data mining, ont montré que l'application d'une probabilité adaptative pour la mutation a permis d'éviter la convergence de la population vers un individu de forte valeur de fitness ; l'AG a en effet abouti à sept meilleurs individus relativement différents et de fitness comparables.

Sebag et Schoenauer [Sebag & Schoenauer, 1996] proposent quant à eux, un contrôle inductif basé sur l'apprentissage à partir d'exemples. Intuitivement, l'apprentissage inductif permet à partir d'événements produits en cours d'évolution de l'algorithme (croisement, mutation...), de bâtir, puis exploiter des règles caractérisant les classes d'événements bons ou mauvais.

4. Les AG's parallèles

Les applications actuelles des AG's s'inscrivent dans des domaines d'études aussi variés que les sciences de l'ingénieur, informatique, reconnaissance de formes, sciences physiques et sciences sociales. Ces applications ont montré l'efficacité des AG's pour la résolution de problèmes d'optimisation difficiles, caractérisés par des espaces de recherche complexe.

Cependant, les AG's présentent un coût d'exécution important pour des tailles de population importantes, ce qui a motivé leur adaptation sur des architectures parallèles.

Trois modèles d'AGP[11] sont proposés dans la littérature : le modèle centralisé, le modèle distribué et le modèle totalement distribué.

4.1. Le modèle centralisé

Ce modèle consiste à utiliser l'algorithme standard en effectuant les étapes d'évaluation, de sélection et de reproduction en parallèle. L'étape de sélection nécessite une connaissance globale des coûts de tous les individus, toute paire d'individus dans la population étant potentiellement candidate [Talbi, 1995]

En raison de la centralisation de la population, ce modèle permet d'obtenir facilement et à tout moment des informations y afférentes : meilleur individu, valeur d'adaptation moyenne de la population etc…

Cependant, la distribution de la sélection et reproduction engendrent un coût de communication élevé.

4.2. Le modèle distribué

Ce modèle consiste à diviser la population sur différents processeurs. Chaque processeur exécute alors l'algorithme standard sur la sous-population qui lui est affectée. De nombreux paramètres interviennent alors dans la définition du modèle [Talbi, 1995] [Schoenauer &Michalewicz, 1997] : nombre total de sous-populations, topologie de connectivité entre sous-populations, fréquence de migration entre sous-populations, mécanisme de remplacement de la sous-population etc…

[11] Algorithme Génétique Parallèle

4.3. Le modèle totalement distribué [Talbi, 1995]

Ce modèle est basé sur une architecture massivement parallèle où le nombre de processeurs peut être modulé en fonction de la taille de la population désirée. La granularité de ce modèle est fine, en ce sens que la population est placée sur un graphe connexe, non complètement connecté, à raison d'un individu par nœud.
Le choix du voisinage est un paramètre important de l'algorithme. Dans le but d'éviter le coût et la complexité des algorithmes de routage dans les architectures parallèles à mémoire distribuée, un bon choix peut être de restreindre le voisinage aux individus directement connectés .

5. Conclusion

Ce chapitre est une présentation sommaire des principes de base nécessaires à la compréhension du fonctionnement des AG's. Nous retenons fondamentalement qu'un AG est une reproduction artificielle de mécanismes naturels liés à la génétique. Basé sur l'application de trois opérateurs que sont la sélection, croisement et mutation, l'AG est un processus cyclique manipulant une population de solutions candidates à un problème dans le but d'optimiser la fonction d'adaptation associée, définie dans un espace éventuellement complexe.
Comparativement à d'autres méthodes d'optimisation, les AG's permettent de résoudre de façon optimale le dilemme de l'exploration contre l'exploitation et d'être dotés de la propriété fondamentale de parallélisme implicite. Outre ces propriétés intéressantes, la robustesse d'exploration des AG's peut être largement améliorée, en qualité de la solution produite et coût de calcul associé, et ce, en adaptant ses éléments au problème posé : conception prudente de l'espace de recherche (espace génotype défini par la représentation), définition ajustée de la fonction fitness (nombre de variables, nombre et type de contraintes et critères à considérer), intégration de la connaissance du domaine à la structure des opérateurs.
Par ailleurs, de nombreux travaux montrent que la parallélisation des AG's permet d'atteindre de bons résultats sur toutes les classes de problèmes. Cependant, la recherche dans ce domaine reste empirique, dominée par la description des résultats que par l'analyse formelle des facteurs observés [Schoenauer & Michalewicz, 1997].

A la lumière de cette présentation des AG's, ces derniers semblent être une solution intéressante pour approcher le problème d'optimisation des performances d'un SRI.
Le chapitre suivant montre l'intérêt et directions d'application des AG's à la recherche d'information de manière générale.

Chapitre 3

Application des Algorithmes Génétiques à la Recherche d'Information

1. Introduction

C'est Jhon Holland qui mit les bases des premières applications des AG's dans ses écrits sur la théorie des systèmes adaptatifs [Holland, 1962]. Parmi les applications historiques des AG's, on cite notamment les travaux de :

- Bagley [Bagley, 1967] sur la conception d'un banc d'évaluation des stratégies de jeu. Bagley a utilisé les AG's afin de rechercher des ensembles de paramètres dans des fonctions d'évaluation de jeux et les a comparés à des algorithmes de corrélation de procédures d'apprentissage inspirées des algorithmes de changement de poids

- Rosenberg [Rosenberg, 1967] sur la simulation de la cellule biologique. Son utilisation des AG's visait la recherche d'un ensemble de concentrations chimiques minimisant la fonction d'antiadaptation des cellules.

Les applications des AG's s'inscrivent dans des domaines d'études très variés : problèmes combinatoires [Oliver & al, 1987] [Montana & Davis, 1989][Duvivier &al, 1995], théorie des jeux [Holland, 1992], extraction de connaissances dans les bases de données [Freitas, 1999] [Fidelis & al, 2000].

Pour notre part, nous nous intéressons au domaine précis de la recherche d'information. Vu sous l'angle de l'optimisation, les techniques de recherche d'information ciblent trois principaux objectifs :

1. ***Représentation optimale des documents*** :
   Consiste à couvrir de manière fidèle la sémantique véhiculée par un document en considérant le contenu de la collection

2. ***Représentation optimale des requêtes***
   Consiste à traduire l'intégralité de la sémantique véhiculée par la requête en considérant le véritable besoin en informations de l'utilisateur ainsi que le contenu de la collection.

3. ***Formalisation optimale de la fonction pertinence***
   Cette dernière traduit une combinaison formelle de critères permettant d'estimer la pertinence d'un document relativement à une requête.

Le présent chapitre a pour objectif de présenter les principaux travaux d'application des AG's dans ce contexte.

2. Recherche d'information basée sur la génétique : travaux et résultats

Dans le cadre de l'application des AG's à la recherche d'information, on recense dans la littérature, les travaux s'intéressant principalement à la description optimale de documents, l'optimisation de requêtes et recherche interactive dans le WEB.

2.1. Représentation des documents

La présentation des différents modèles de recherche d'information au premier chapitre, nous a permis de mettre en exergue l'impact considérable de la qualité de représentation des documents et requêtes sur les résultats de la recherche.
Dans ce sens, Gordon affirme [Gordon, 1988] :
« *… if forming adequate representations of both document and user's need were completely understood, there would be no need for further research in this field …* »

Poursuivant ainsi l'objectif d'atteindre une "bonne" représentation des documents, Gordon [Gordon, 1988] propose une méthode adaptative de redescription des documents, dans le modèle probabiliste, basée sur les AG's.
L'AG standard opère sur chaque document en lui associant *N* descriptions dont chacune est définie par une liste de termes d'indexation non pondérée.
Le renouvellement des générations de descripteurs de documents est basée sur l'utilisation de requêtes d'apprentissage pertinentes et autres non pertinentes.
Plus précisément, l'AG a la structure générale suivante :

Début
1. *Générer la population initiale de descripteurs du document D*

Répéter
2. *Evaluer chaque descripteur*
3. *Remplacer la génération courante de descripteurs en considérant*
   - *a- la structure des descripteurs courants*
   - *b- le degré de ressemblance avec les requêtes pertinents et requêtes non pertinentes*

Jusqu'à *atteindre un critère d'arrêt*
Fin

Nous présentons ci-dessous une description plus détaillée des étapes de l'AG.

1. Génération de la population initiale

L'auteur initialise la population de descripteurs du document ciblé aux descripteurs des requêtes pertinentes associées.

Un descripteur a la forme générale suivante :

$$Desc\_D_i = \begin{array}{cccccc} & t_1 & t_2 & & t_T & \\ < & I_1 & I_2 & & I_t & > \end{array}$$

Où :

$Desc\_D_i$: Descripteur i du document D

$t_1, \ldots, t_T$: Termes d'indexation de la collection

I_i : Indicateur binaire de présence du terme t_i dans le descripteur

Les populations de descripteurs associées à chaque document sont indépendantes.

2. Evaluation des descripteurs

La qualité de chaque descripteur de chaque document *D* est évaluée à l'aide de la formule suivante :

$$Fitness(Desc\_D_i)=Score(Desc\_D_i,R\_P)+w*(G_g^{NP}-(Score(Desc\_D_i,R\_NP)-G_g^{NP}))$$

Où :

R_P : Ensemble des requêtes pertinentes pour le document D

R_NP : Ensemble des requêtes non pertinentes pour le document D

Score ($Desc\_D_i$, R_P) : Score moyen de ressemblance du descripteur avec les requêtes pertinentes

Score ($Desc\_D_i$, R_NP) : Score moyen de ressemblance du descripteur avec les requêtes non pertinentes

G_g^{NP} : Score de ressemblance moyen de la population de descripteurs, à la génération g, avec les requêtes non pertinentes

w : Constante

Avec :

$$Score(Desc\_D_i,R\_P)=\frac{1}{M}\sum_{k=1}^{M}J(Desc\_D_i,q_k^P)$$

$$G_j^{NP}=\frac{1}{M*N}\sum_{i=1}^{N}\sum_{k=1}^{M}J(Desc\_D_i,q_k^{NP})$$

Où :

q_k^P : Requête pertinente pour D

q_k^{NP} : Requête non pertinente pour D

M : Nombre de requêtes pertinentes pour D

N : Nombre de descripteurs du documents D

J : Mesure de Jaccard

La fonction d'adaptation ainsi définie combine l'intérêt de ressemblance de chaque descripteur aux requêtes pertinentes et dissemblance aux requêtes non pertinentes.

3. Renouvellement de la génération de descripteurs

Cette étape est réalisée selon deux opérations : reproduction et croisement.

a- Reproduction

Cette opération aboutit au clonage de chaque descripteur. Le nombre de clones est calculé sur la base du fitness relatif du descripteur associé, calculé comme suit :

$$Fitness\_relatif \quad (Desc\_D_i) = G_q^p \Big/ Score(Desc\_D_i, R\_P)$$

Où :

$$G_g^p = \frac{1}{M*N} \sum_{i=1}^{N} \sum_{k=1}^{M} J(Desc\_D_i, q_k^P)$$

La taille de la population est constante de génération en génération.

b- Croisement

Le principe est de constituer de manière aléatoire une partition de *N/2* paires de descripteurs. Pour chaque paire, on établit un croisement à un point.

Les expérimentations réalisées sur une base de test locale font état d'un accroissement de performances évalué à *25%* à la *40ème* génération. L'auteur montre que l'AG produit des descriptions de documents plus performantes que celles générées dans le modèle probabiliste.

Gordon exploite ces premiers résultats pour définir un mécanisme de classification des documents [Gordon, 1991] basé sur le regroupement de documents pertinents à une même requête. L'auteur élabore une technique de classification qui permet :

- le regroupement de documents copertinents dans la même classe,
- l'identification des classes de documents à partir de descriptions partielles,
- l'association de requêtes aux classes pertinentes.

L'expérimentation de l'approche sur une base de tes locale révèle que la redescription « génétique» des documents , permet d'atteindre *39,74%* d'accroissement des performances au bout de 20 générations et *56.61%* au bout de *40* générations

2.2. Optimisation de requête

L'optimisation de requête traduit le processus de reformulation de requête. Ce dernier a été préalablement défini comme un mécanisme de modification de requête par expansion et/ou réestimation des poids d'indexation.
Dans ce cadre, différents travaux ont étudié la faisabilité du processus en utilisant les principes de la génétique.

Yang & Korfhage [Yang & Korfhage, 1993] développèrent un AG pour une optimisation de requête par réestimation des poids d'indexation sans induire une expansion. Un individu requête est représenté comme une liste pondérée de termes d'indexation. Les générations de requêtes sont renouvellées par application :
- d'une fonction d'adaptation basée sur la formule :

$$Fitness(q)=\alpha R_{p}^{(q)}-\beta R_{Np}^{(q)}$$

Où :
q : Individu requête
$R_p^{(q)}$: Nombre de documents pertinents retrouvés
$R_{Np}^{(q)}$: Nombre de documents non pertinents retrouvés

- d'une sélection basée sur un échantillonnage stochastique [Baker, 1985],
- d'un croisement à deux points et d'une mutation classique.

Les expérimentations préliminaires réalisées sur deux collections : IPM (Information Processing & Management) à *85* documents et une requête et NPL (National Physical Laboratory) à plus de *1000* documents et *100* requêtes, ont montré d'une part, l'intérêt de l'approche et d'autre part, la difficulté d'ajustement des probabilités de croisement et mutation en fonction des collections et des générations de l'AG.
Ceci a motivé les auteurs pour la variation de ces paramètres en cours d'évolution de l'AG et ce, afin de prévenir une convergence prématurée vers des optima locaux.
D'autres expérimentations réalisées sur la collection TREC ont montré que cette technique d'adaptation des probabilités de croisement et mutations, assure la convergence des générations de requêtes en moyenne au bout de *3* à *6* générations.

Chen [Chen, 1995] a développé le système dénommé GANNET pour mettre en œuvre une reformulation de requête avec expansion. Le système exploite un réseau de neurones Hopfield pour la modélisation des associations sémantiques entre concepts et un AG pour la sélection de concepts candidats à l'expansion.
La représentation des individus requêtes est binaire, la population initiale de requêtes est constituée de documents jugés pertinents à l'issue de la recherche initiale.
Le processus de recherche d'information est cyclique, opérant en trois phases .

1. Phase d'optimisation de concepts

Chaque document issu d'un cycle de recherche est évalué en qualité de requête. L'évaluation est basée sur la fonction d'adaptation proposée par Gordon [Gordon, 1988].

Le document de plus grande valeur d'adaptation constitue alors la source d'activation du réseau de neurones

2. Phase d'exploration de concepts

L'activation, en entrée, des termes issus de la phase précédente, produit par propagation des signaux d'activation, une liste de nouveaux termes pertinents en sortie.

3. Phase de sélection

Une nouvelle population de requêtes est constituée par :

- sélection des descripteurs de documents décrits par au moins un des nouveaux termes en sortie du réseau, et dont la valeur d'adaptation est supérieure à un seuil fixé,
- application d'un croisement à un point et mutation classique.

Le processus se poursuit jusqu'à ce que l'accroissement de performances entre deux cycles successifs, ne soit plus significatif.

Le système GANNET a été évalué en utilisant une base de test de *3000* articles issus de la collection DIALOG et qui a permis de générer un réseau de *1488* concepts et *44486* liens pondérés.

Les expérimentations réalisées révèlent que les performances du système sont très dépendantes des résultats de la recherche d'information. On enregistre en effet une variation d'accroissement de performances de *7%* à *48%* en fonction du nombre de cycles de recherche.

Kraft & al [Kraft & al, 1995] ont appliqué les techniques de programmation génétique dans le but d'optimiser la représentation des requêtes dans le modèle booléen. Un individu requête est représenté sur la base du modèle génétique de Koza [Koza, 1991].

Leurs premières expérimentations ont montré la faisabilité de l'approche pour dériver des requêtes qui accroissent les performances du système en termes de rappel et précision.

2.3. Recherche interactive dans le WEB

Dans le contexte de la recherche interactive d'informations dans le WEB, Menczer & Belew [Menczer & Belew, 1999] proposent une méthode adaptative de recherche basée sur la coopération d'agents qui effectuent une recherche à contexte local.

Soit un réseau d'informations modélisant des liens référentiels entre documents. On calcule la probabilité de consulter un document pertinent selon la formule :

$$\partial = \eta R + (1-\eta) G$$

Où **:**

η : Probabilité que le document courant soit pertinent

G : Probabilité que tout document consulté à partir du document courant, soit pertinent

R : Probabilité conditionnelle d'atteindre un document pertinent à partir du document courant et ce, en choisissant aléatoirement un lien de référence

La consultation récurrrente de tous les documents référés est de complexité rhédibitoire (ordre de e^{k2}). L'objectif est alors de mettre en œuvre une population d'agents de recherche qui naviguent à travers le réseau d'informations. Ces agents évoluent selon un algorithme évolutif qui optimise la pertinence supposée des documents visités, en réduisant les coûts de recherche.

La recherche d'information est coopérative, en ce sens que chaque agent est autonome, effectue des décisions locales sur les liens de référence à emprunter et ajuste sa stratégie relativement au contexte local (nœud courant) et besoins de l'utilisateur à travers les jugements de pertinence.

La structure générale de l'algorithme est présentée ci-dessous [[Menczer & Belew, 1999] :

Les principales étapes de l'algorithme sont les suivantes :

1. **Génération de la population initiale**
   L'utilisateur produit initialement une liste de mots clés et documents pertinents *(D_1, …, D_p).* Chaque agent est alors positionné sur un de ces documents avec une énergie initiale $E_0 = \theta/2$.

2. **Sélection d'un lien de navigation**
   Un agent est situé sur un document présentant plusieurs liens de référence. Chacun d'eux est caractérisé par les termes les plus proches $k_1 \ldots k_l$, et modélisé à l'aide d'un réseau de neurones illustré sur la figure 3. 1. La sélection du lien de navigation est effectué en deux opérations : calcul de sa valeur d'activation puis calcul de sa probabilité de sélection.

Notations :

a : Agent

D : Document

E_a : Energie de l'agent a

c(D) : Coût d'atteindre le document D : durée de transfert, longueur du document …

e(D) : Valeur de pertinence d'un document D. Cette valeur provient de l'utilisateur ou estimée

θ : Constante

Début

1. *Initialiser une population de N agents avec une énergie initiale $E = \theta / 2$*
2. *Refaire :*
3. *Pour chaque agent a*
4. *Sélectionner un lien à partir du document courant*
5. *Consulter le document référé D*
6. *Mettre à jour $E_a = E_a - c(D) + e(D)$*
7. *Apprendre par renforcement le signal e(D)*
8. ***Si** ($E_a \geq \theta$) **Alors***
9. *à = Mutation (Croisement (Clone (a))*
10. *$E_{à} = E_a / 2$*
11. ***Sinon** Si ($E_{à} < 0$)*
12. *Eliminer (a)*
13. *Récupérér le jugement de pertinence de l'utilisateur*

Fin

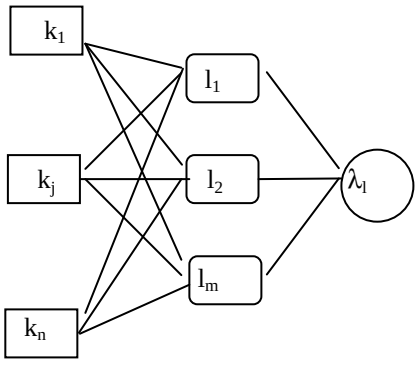


Figure 3.1 : Réseau descriptif d'un agent

[Menczer & Belew, 1999]

a- Calcul de la valeur d'activation des liens

Pour chaque lien de référence *l* figurant dans le document courant, on calcule la valeur d'adaptation propagée :

$$\lambda_l = \sum_{j=1}^{n} (b_j + \sum_{k=1}^{m} w_{jk} In_k^l)$$

Où :
λ_l : Valeur d'activation du lien l
b_j : facteur de biais numéro j
w_{jk} : Poids de la liaison du nœud lien j et terme k
in_k^l : Valeur d'activation en sortie du lien l à partir du terme k
n : Nombre de termes du lien l
m : Nombre total de liens dans le document

Avec :

$$In_k^l = \sum_{i / dist(ki,l) < \rho} \frac{1}{dist(k_i,l)}$$

Où :
dist(k_i , l) : Nombre de liens où intervient le terme k_i.

La distance $dist(k_i , l) < \rho$ limite le comptage de liens à une fenêtre de ρ liens

b- Calcul de la probabilité de sélection des liens

L'agent calcule de manière stochastique un score de sélection de liens de navigation selon la distribution de probabilité suivante :

$$\Pr[l] = \frac{e^{\beta \lambda_l}}{\sum_{l' \in D} e^{\beta \lambda_{l'}}}$$

Où :
β : Constante

Le lien de plus grande valeur de *Pr[l]* est alors emprunté par l'agent.

3. Calcul de la valeur d'adaptation d'un agent

Suite à la sélection d'un lien de référence, le document atteint est consulté puis on met à jour l'énergie de l'agent qui traduit sa valeur d'adaptation. La mise à jour exprime un gain ou une perte en fonction des jugements de pertinence préalables ; elle est le résultat de l'application de la formule :

$$Ea = Ea - c(D) + e(D)$$

Où :

$$e(D)=\begin{cases}\lambda*\phi(D) & si \quad D \quad est \quad préalablement \quad jugé \\ \tanh(\sum_{k\in D} freq(k,D)*I_k) & \sin on\end{cases}$$

Avec :

tanh : Fonction tangeante hyperbolique $\in$ [-1 +1]

freq(k,D) : Fréquence du terme k dans le document D, normalisée par la longueur du document

I_k : Facteur de pertinence du terme k, variable en cours d'évolution de l'algorithme selon la formule :

$$I_k=\alpha I_{k+}+(1-\alpha)*w_{k*}(1+\log(\frac{1}{C_k}))$$

Avec :

α : Terme d'inertie

w_k : Valeur de pertinence cumulée du terme k

$w_k=w_k+\phi(D)$, $\phi(D)\in$ *[-1, 0, +1] : jugement de pertinence de l'utilisateur*

C_k : Proportion de documents pertinents sauvegardés et contenant le terme k

4. Apprentissage du réseau de liens de référence

Un signal de renforcement est calculé selon la formule :

$$\delta(D)=e(D)+\mu Max_{l\in D}\{\lambda_l\}-\lambda_D$$

Où :

μ : Constante

Le poids des liens de référence sont alors corrigés par application de l'algorithme de rétropropagation.

5. Application des opérateurs génétiques

L'évolution génétique de chaque agent est basée sur la valeur de son énergie ; il est reproduit dans le cas où son énergie est positive, éliminé dans le cas contraire. Les deux opérateurs de croisement et mutation sont appliqués aux agents. Deux types de croisement sont définis :

- croisement local : un agent n'est combiné qu'avec un agent situé sur le même document,
- croisement global : un agent peut être combiné avec tout autre agent.

La sortie de l'algorithme est un flot de liens de référence ordonnés par les valeurs de pertinence estimée en partie en fonction du jugement de pertinence de l'utilisateur. L'algorithme s'arrête au vœu de l'utilisateur ou par absence de liens pertinents.

L'approche a été évaluée sur une sous collection de la collection « Human Society », contenant *19427* documents organisés en hypergraphe.

Les expérimentations réalisées montrent globalement l'intérêt de l'approche. Plus précisément, les résultats montrent l'avantage de combiner la capacité de recherche d'information locale des agents et capacité de recherche d'information globale et hétérogène due à l'application de la génétique d'une part et d'une vision de pertinence plus large de l'utilisateur, d'autre part.

3. Conclusion

Il ressort principalement de cette étude, que les AG's trouvent un champ d'application adéquat dans le domaine de la recherche d'information. Il est à noter en effet, que d'un angle de vue très large, leurs propriétés inhérentes de parallélisme implicite et construction adaptative des solutions, est très avantageuse pour réduire la complexité d'exploration d'un espace documentaire d'une part, et construction graduelle et coopérative de la requête optimale d'autre part.
D'un angle de vue plus particulier, les travaux dans le domaine ont indéniablement montré l'intérêt des AG's à apporter, principalement, des solutions judicieuses :

- au problème de représentation des documents [Gordon, 1988] [Gordon, 1991],
- à la difficulté de sélection des termes candidats à l'expansion de requête [Chen, 1995],
- à l'inconvénient de traitement des termes de manière indépendantes, posé par le mécanisme classique de reformulation de requête [Yang & Korfhage, 1993].

Outre ces propriétés avantageuses et intrinsèques des AG's, nous proposons d'y apporter des adaptations qui puisent dans une large mesure des résultats d'application des stratégies de recherche d'information de manière générale, et reformulation de requête de manière particulière.
Plus précisément, notre approche est particulièrement caractérisée par l'intégration de la technique de nichage lors de la mesure de l'adaptation des requêtes, et intégration des heuristiques de recherche d'information dans la structure des opérateurs.
La technique de nichage a pour but fondamental le rappel de documents pertinents de descripteurs relativement différents.
La connaissance intégrée aux opérateurs permet de guider l'exploration en vue d'atteindre des résultats appréciables en peu de générations, et ce, en égard à la contrainte de limitation des itérations feedback.
Notre approche est présentée de manière détaillée dans les chapitres suivants.

Partie 2

Mise en Œuvre d'un Algorithme Génétique Adapté à l'Optimisation de Requête dans un Système de Recherche d'Information

Introduction

Nous décrivons dans cette partie, notre approche pour l'optimisation de requête dans un SRI.

Dans le quatrième chapitre, nous décrivons les caractéristiques de l'approche, présentons nos motivations, fonctionnement général du SRI et principaux éléments de l'AG : individu, fonction d'adaptation, structure et objectifs des opérateurs proposés.

Nous y présentons également les résultats d'évaluation de l'approche sur les collections CACM, ADI et TREC6. Un bilan critique de ces résultats nous a mené à dresse une nouvelle ébauche pour le processus génétique de recherche d'information. Ceci fait l'objet du cinquième chapitre.

Nous nous y intéressons notamment à décrire les principales révisions apportées à l'algorithme général d'optimisation de requête, principe d'organisation de la population, mode d'exploitation des résultats de recherche, structure et principe d'application des opérateurs génétiques.

Nous présentons ensuite nos expérimentations pour l'évaluation de notre approche améliorée sur la collection AP88 de TREC. Cette évaluation nous permet d'une part, de mesurer l'apport de l'optimisation génétique des requêtes à la recherche d'information et, d'autre part, de déterminer l'impact des paramètres et heuristiques de l'algorithme sur les résultats de la recherche. Une évaluation comparative des approches proposées y est également présentée.

Chapitre 4

Présentation de notre Approche : Description Générale et Evaluation Préliminaire

1. Introduction

Nous avons étudié dans le premier chapitre, différents modèles de recherche et de représentation d'information, présentés dans la littérature. Les différents modèles sont essentiellement basés sur une représentation formelle des requêtes et documents. Le mécanisme d'appariement proposé est fondé sur une mesure analytique de la pertinence suivie de l'application d'un seuillage.
Cette étude nous a permis de conclure que la conception d'un SRI dans sa globalité, nécessite l'intégration de stratégies de recherche et mérite l'expérimentation de techniques hybrides de l'intelligence artificielle.
A ce propos, notre intérêt s'est porté sur l'exploitation des concepts de la génétique pour greffer un mécanisme de reformulation de requête à un modèle de recherche de base.
Dans ce cadre, nous avons proposé une approche de reformulation de requête, par injection de pertinence, basée sur les AG's. Nous exploitons la robustesse de ces algorithmes dans l'objectif d'assurer une exploration efficace du fond documentaire, soutenue par des transformations génétiques inspirées des résultats expérimentaux des approches classiques de reformulation de requête.

Le présent chapitre décrit de manière détaillée la structure de base et éléments de l'AG d'optimisation de requête que nous préconisons, ainsi que le fonctionnement général du SRI.
Les résultats des expérimentations réalisées à l'aide du système Mercure [Boughanem & Soule-Dupuy, 1997] sur les collections CACM, ADI et TREC6 y sont également présentés.
Ces résultats préliminaires montrent globalement l'intérêt de notre approche.

2. Motivations

Nous exploitons les techniques et concepts de l'algorithmique génétique en vue de mettre en œuvre un processus d'optimisation de requête, motivé par les éléments de réflexion suivants :

1. Le fond documentaire peut être perçu comme un espace de dimension élevée. La recherche de(s) requête(s) optimale(s) permettant de capturer des voisinages de documents pertinents à la requête utilisateur, évoque à plus d'un titre la puissante capacité d'exploration des AG's. Leur parallélisme implicite permettrait en effet, d'orienter la recherche simultanée à travers plusieurs régions de documents, caractérisées par différents termes descriptifs.

2. Par opposition aux modèles classiques qui focalisent le recherche d'information sur une unique requête, l'AG manipule une population de requêtes dont chacune d'elles peut être à l'origine de la restitution de documents pertinents.
   Le rapprochement d'une requête à un ensemble de documents de structures différentes, nous semble plus lent et moins prometteur que le rapprochement entre ensemble de requêtes et sous-ensembles de documents pertinents.
   A titre illustratif, nous avons calculé le nombre de documents pertinents qui n'ont aucun terme commun avec la requête initiale ni avec celle déduite par modification pour les topics *350-450* sur la collection adhoc8 (CD4 et CD5) de TREC. On a constaté qu'il y a une requête qui n'a aucun terme commun avec *50%* des documents pertinents, *7* requêtes avec *20%* et *16* requêtes avec *10%* . Ceci encourage alors l'idée d'une recherche multi-requêtes.

3. La reformulation de requête telle que préconisée dans le modèle vectoriel, manipule les termes indépendamment les uns des autres. Or la pratique a montré que les termes occurrent dans les documents par combinaison. L'AG apporterait dans ce cas précis, une contribution considérable pour la préservation de « briques élémentaires » qui constituent, dans notre cas, des groupes de termes occurrant par combinaison dans les documents pertinents

4. L'efficacité de la technique classique d'injection de pertinence dépend étroitement du degré d'exhaustivité du mécanisme d'indexation. Plus précisément, l'inconvénient majeur de cette approche est le rappel de documents ressemblants. L'intégration de l'heuristique de nichage dans la modélisation d'un AG nous permettrait de remédier à ce problème en encourageant l'exploration dans des directions différentes de l'espace documentaire.

5. L'intégration de la connaissance dans la structure des opérateurs permettrait de faire converger l'algorithme de recherche vers les documents pertinents et ce, à un nombre plus réduit d'itérations feedback.

Comparativement aux autres travaux dans le domaine, notre approche est caractérisée par l'évolution d'un **AG adapté**, spécifique à la recherche d'information en ce sens qu'il vise une recherche d'information coopérative, régulée par le jugement de pertinence de l'utilisateur et caractérisée par l'utilisation :

1. d'une **fonction d'adaptation ajustée par le nichage** permettant de faire progresser la recherche d'information dans des sous espaces documentaires différents; ceci offre alors des possibilités intéressantes de rappel de documents de descripteurs non proches,

2. d'opérateurs génétiques non aveugles, augmentés par une connaissance théoriquement et expérimentalement approuvée dans le domaine de la relevance feedback, accélérant le processus de recherche d'information par une exploration guidée de l'espace des documents.

3. Le processus génétique de recherche d'information

La première partie de notre travail s'est essentiellement articulée sur l'étude des modèles de recherche d'information documentaire. Nous avons alors analysé les limites des approches usuelles et les diverses stratégies élaborées dans le but d'améliorer les performances d'un SRI.
Plus particulièrement, le mécanisme de reformulation de requête est un moyen permettant d'adapter graduellement le processus de recherche d'information tant au besoin de l'utilisateur qu'à l'environnement linguistique du système.
Nous exploitons la puissante capacité d'optimisation des AG's en vue de mettre en œuvre un mécanisme de reformulation de requête qui en termes du bilan de notre précédente étude, possède les caractéristiques avantageuses suivantes :

- **Recherche simultanée** et **guidée** dans des directions différentes de l'espace documentaire

- **Recherche coopérative** permettant le rappel de documents de descripteurs différents et pertinents pour une même requête. Ceci est réalisé par l'intégration de la technique de nichage dans la mesure d'adaptation des requêtes.

- **Reformulation par injection de pertinence** permettant une expansion et repondération contextuelle de requêtes. Le jugement de pertinence de l'utilisateur est en effet exploité pour effectuer des croisements et mutations traduisant un procédé de reformulation de requête qui puise dans une large mesure des techniques du domaine.

- **Manipulation des termes par combinaison** et non de manière indépendante. L' évolution des générations de l'AG induit en effet de manière inhérente à son fonctionnement, la préservation de blocs de termes jugés intéressants.

3.1. L'approche adoptée

Le processus de recherche d'information que nous proposons est essentiellement basé sur le déroulement d'un AG, qui par essence est cyclique, et vise dans notre cas **l'optimisation de requête**. Celle-ci consiste à construire de génération en génération, la (les) requête(s) permettant de rappeler le maximum de documents pertinents associés au besoin en information exprimé par l'utilisateur. Cet algorithme coordonne les activités de trois unités fonctionnelles fondamentales :

1. L'utilisateur

Constitue l'acteur actif, prépondérant dans le fonctionnement du SRI de façon générale et évolution du processus de recherche d'information de façon particulière.

Les interactions utilisateur-SRI sont à la base du mécanisme de reformulation de requête induit par l'AG de recherche d'information. Plus précisément, l'intervention de l'utilisateur dans le fonctionnement du système se justifie par :

- la formulation du besoin en information : correspond à l'étape classique d'expression de requête,
- l'expression d'un jugement de pertinence : ceci évoque le feedback utilisateur à travers lequel le SRI capte le jugement de pertinence de l'utilisateur quant aux documents restitués par le système, à une itération donnée du cycle de recherche.

Dans notre contexte, ce jugement est exploité d'une part, dans le but d'attribuer une valeur d'adaptation (fitness) à des individus requêtes. Ceci permet de générer de nouvelles requêtes (reformulation) plus adaptées en ce sens, qu'elles s'orientent d'avantage vers les directions des documents pertinents.

D'autre part, le jugement de pertinence de l'utilisateur est exploité dans le but d'adapter la structure des opérateurs génétiques.

2. Un modèle de recherche de base

C'est la composante permettant de modéliser le fond documentaire et supporter le mécanisme inhérent de sélection des documents. Un cycle de recherche correspond à une itération feedback durant laquelle chaque individu requête de la population est présenté à l'entrée du modèle de recherche de base.

Le principe de recherche – sélection d'informations associé, produit une liste partielle et ordonnée de documents.

3. Un AG adapté à la recherche d'information

Traduit le processus d'optimisation de requête. Sur la base d'une fonction d'adaptation qui valorise la pertinence de chaque requête dans le sous espace documentaire correspondant, des opérateurs génétiques augmentés par la connaissance, effectuent une reformulation de requête par application des opérateurs

classiques de sélection, croisement et mutation des individus requêtes de la population courante.

En somme, l'approche est caractérisée par les principales propriétés suivantes :

1. Exploitation du mécanisme de relevance feedback : constitue un mécanisme de reformulation de requête fort intéressant. Son impact est d'autant plus considérable dans le cadre de notre approche que les transformations effectuées en terme de repondération et expansion, agissent sur un ensemble potentiel de requêtes et non une unique requête. Ceci offre l'avantage majeur d'ajuster la composante de chaque individu requête en corrélation avec le sous espace documentaire de recherche qu'il définit.

2. Possibilité d'intégration à tout modèle de recherche d'information de base, qui met en oeuvre un appariement requête – document basé sur une fonction d'ordre.

3. Mise en œuvre d'un AG adapté à la problématique de recherche d'information.

3.2. Fonctionnement général

Le processus de recherche d'information que nous mettons en oeuvre est adaptatif aux besoins de l'utilisateur. En ce sens qu'une session d'interrogation correspond à une suite d'interactions utilisateur-SRI exploitées par l'AG de recherche d'information.

Nous décrivons dans ce qui suit le mode de fonctionnement global du SRI.

En premier lieu, le besoin en information de l'utilisateur est présenté au système sous forme d'une expression en langage naturel ou une liste de mots clés. Cette expression est alors analysée par une procédure d'indexation qui la traduit, sur la base des termes d'indexation identifiés, en une liste de termes significatifs. La requête indexée est alors présentée à l'entrée du modèle de recherche de base. Son évaluation conduit à la restitution d'une liste ordonnée de documents en sortie; cette première réponse du système est soumise au jugement de pertinence de l'utilisateur et sera la base de la construction de la population initiale de requêtes. Suite à l'évaluation de la valeur d'adaptation de chaque individu requête, interviennent les opérateurs génétiques qui construisent de nouvelles requêtes étalonnées sur leur valeur de pertinence.

Une recherche coopérative d'informations est alors menée en présentant chaque individu requête à l'entrée du processus de recherche de base. Les listes partielles issues de l'évaluation de chaque requête sont alors fusionnées pour constituer une liste unique présentée à l'utilisateur. Les itérations feedback délimitent ainsi dans notre contexte, la période de vie des générations de requêtes. Le jugement de pertinence de l'utilisateur a un impact direct sur le calcul de l'adaptation des individus requêtes et structures des opérateurs génétiques mis en jeu.

La figure 4.1 illustre le principe général de fonctionnement du SRI .

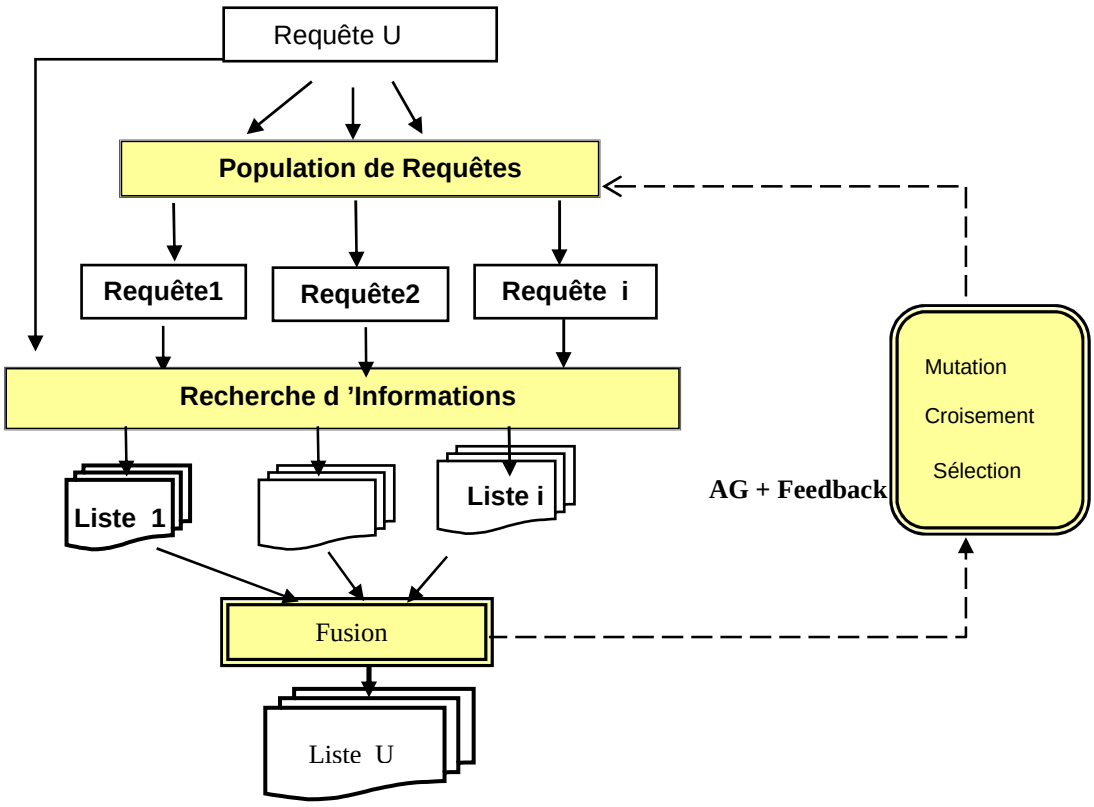


Figure 4.1 : Processus général de fonctionnement du SRI

3.3. Algorithme de base

Le processus génétique d'optimisation de requête, se déroule selon l'algorithme présenté sur la figure 4.2.

Le but de l'algorithme est de générer la (les) requête (s) optimale(s) à partir du besoin en information exprimé par l'utilisateur. En ce sens, l'AG tente de faire évoluer de génération en génération, une population de requêtes vers la ou les requête(s) à même d'accroître les performances du SRI. Le processus s'arrête à un nombre d'itérations feedback déterminé par l'utilisateur.

Début
Evaluer Requête utilisateur
t :=0
Construire la population initiale de requêtes $Pop^{(0)}$
Répéter
Pour *chaque requête de* $Pop^{(0)}$
Effectuer la recherche
Fait
Effectuer la fusion
Retenir le jugement utilisateur sur les NbJug top documents
Calculer l'adaptation de chaque requête
Appliquer les opérateurs génétiques
t :=t+1
Jusqu'à *arrêt*
Fin

Figure 4.2 : Structure de base de l'AG d'optimisation de requête

4. Description de l'AG d'optimisation de requête

Nous présentons dans cette section les éléments caractéristiques de l'AG d'optimisation de requête.

4.1. Individu requête

A la lumière de l'analyse des AG's de manière générale et de leur principe d'efficacité de manière particulière, il s'avère que le codage des individus est une étape de modélisation fondamentale, possédant un impact majeur sur le déroulement de l'algorithme ainsi que sur la qualité des résultats obtenus.
Goldberg préconise à cet effet le respect de principes fondamentaux (Cf. Chapitre 2, paragraphe 3.1.1) que nous discutons dans le cadre du codage que nous retenons.
Un individu requête est représenté comme suit :

	t_1	t_2		t_T
$Q_u^{(s)}$	(q_{u1}	q_{u2}		q_{uT})

Figure 4.3 : Code d'un individu requête

Où :
$Q_U^{(s)}$: Individu requête n° u de la population à la génération s
$t_1, t_2, ..., t_T$: Liste de termes d'indexation
q_{ui} : Poids du terme t_i dans la requête individu $Q_u^{(s)}$

Le génotype d'un individu requête est caractérisé par :

- une taille effective limitée au nombre de termes d'indexation de poids non nuls. Les termes d'indexation de poids nuls ne sont en effet pas considérés lors des traitements génétiques.
- une représentation réelle : les gènes ont des valeurs bornées dans l'intervalle *[0 1]* en accord avec la fonction de pondération de la requête initiale puis des transformations génétiques opérées sur les individus,

- des locus non fixes puisque les termes ne sont pas représentés uniformément à la même position pour l'ensemble des requêtes.

En respect de l'aspect formel du principe de codage dans un AG, nous convenons de caractériser le présent codage relativement aux principes fondamentaux de pertinence des briques élémentaires et minimisation de l'alphabet.

- Principe de pertinence des briques élémentaires [Goldberg, 1994]

La preuve formelle des AG's montre que la recherche de l'optimum comporte un biais en faveur des schèmes courts et d'ordre faible. Or la longueur d'un schème dépend de la représentation retenue; pour notre part, la taille d'un individu requête est celle de sa représentation minimale à savoir un descripteur construit sur ses propres termes d'indexation et non sur l'ensemble des termes d'indexation reconnus dans le système. Ceci conduit à la constitution de schèmes relativement courts qui revêtent une sémantique particulière dans le cadre de notre présent problème d'optimisation.

Un schème représente en effet, un groupe de termes éventuellement pertinent dans une direction de recherche donnée; de plus, les locus n'étant pas fixes, la représentation offre des possibilités intéressantes de réarrangement pour la création de combinaisons de termes à effet épistatique, en ce sens qu'elles constituent des contextes sémantiques liés aux concepts véhiculés par la requête utilisateur.

En outre, le principe de recherche coopérative menée par l'ensemble des niches, suppose que chacune de ces dernières couvre un sous-espace relativement élargi de l'espace complexe défini par l'espace documentaire. Ceci rejoint alors, l'intérêt de la constitution de schèmes d'ordre faible, délimitant les frontières du voisinage de recherche.

- Principe des alphabets minimaux [Goldberg, 1994]

Ce principe plaide sans doute pour une représentation binaire des individus requêtes. Cependant, ce type de représentation pose deux inconvénients majeurs :

1. La représentation binaire repose sur la composition de la requête en termes et non en poids et n'a par conséquent pas un effet discriminatoire majeur. L'évaluation de requête binaire conduirait alors à un taux de bruit considérable.

2. La représentation binaire ne prédispose pas à des transformations génétiques conséquentes. Plus précisément, ce type de représentation ne supporterait que des altérations dans la composition des termes de la requête, ce qui réduit le champ d'action des opérateurs génétiques que nous voulons plus performants de par l'exploitation des résultats expérimentaux issus de l'étude des processus de reformulation de requêtes, de représentations réelles.

Nous avons alors opté pour une représentation réelle qui, force est de constater qu'elle est redondante (un phénotype est associé à plusieurs génotypes). Cependant, la représentation respecte le principe palliatif de redondance minimale [Sebag & Schoenauer, 1996] puisque la restitution redondante de documents est à priori le résultat de l'évaluation d'individus requêtes voisins.

4.2. Population de requêtes

La population est renouvelée à chaque génération sur la base des résultats de recherche et des transformations génétiques opérées sur les individus.
La génération de la population initiale s'effectue selon deux étapes : évaluation de la requête initiale et sélection des individus requêtes de la population initiale.

1. **Evaluation de la requête initiale**
   Une requête utilisateur est à l'origine de la génération de la population initiale. La procédure d'indexation, produit une requête représentée selon la description suivante :

$$Q_I(q_{I1}, q_{I2,}\ldots, q_{IT})$$

Où :
Q_I : Requête initiale
q_{Ij} : Poids du terme t_j dans la requête Q_I, calculé selon la formule de pondération :

$$q_{ij}=\begin{cases}\dfrac{nq*qtf}{nq-qtf} & si\ (nq>qtf)\\ qtf & \text{sinon}\end{cases}$$

Où :
qtf : Fréquence d'un terme dans une requête
nq : Nombre de termes dans la requête

La requête ainsi présentée est évaluée par le processus de recherche de base, qui produit une liste ordonnée de documents sélectionnés et soumis au jugement de pertinence de l'utilisateur.

2. Sélection des individus requêtes de la population initiale

A ce stade est effectivement constituée la composante de la population initiale notée $Pop^{(0)}$. On pose :

$Dr^{(0)}$: Ensemble de documents pertinents à la première itération feedback

$Ds^{(0)}$: Ensemble des documents sélectionnés à la recherche initiale

Taille_Pop : Taille de la population

On construit alors :

$$Pop^{(0)} = Dr^{(0)} \bigcup (Ds^{(0)}, (Taille\_Pop + 1) - |Dr^{(0)}|)$$

Où :

$(Ds^{(0)}, E)$: Ensemble des E premiers documents issus de l'ensemble $Ds^{(0)}$

En clair, la population initiale est constituée des descripteurs des documents pertinents, complétée par les documents situés en début de la liste présentée à l'utilisateur. Précisons qu'un descripteur de document est une liste pondérée de termes d'indexation et peut donc exprimer une requête.

Ainsi, la génération de la population initiale de requêtes n'est pas aléatoire. L'exploration de l'espace documentaire est amorcée à partir de régions que nous jugeons prometteuses.

4.3. Fonction d'adaptation

De manière générale, la fonction d'adaptation mesure la performance d'un individu dans la résolution du problème posé. Dans le contexte précis du problème d'optimisation de requête, la fonction d'adaptation doit s'étalonner à la mesure des performances du système en taux de rappel/précision.

A cet effet, la liste fusionnée de documents restituée par le système suite à l'évaluation d'une génération de requêtes, est éclatée par l'utilisateur à l'itération de feedback correspondante en deux ensembles : documents pertinents et documents non pertinents.

Une requête est d'autant plus adaptée qu'elle est à l'origine de plus de documents pertinents et moins de documents non pertinents. Sur cette base, et à la suite d'un choix issu de nombreuses expérimentations, nous proposons, en premier lieu, la formulation suivante de la fonction d'adaptation [Tamine, 1997] :

$$QFitness(Q_u^{(s)}) = \frac{\frac{1}{|Dr|} * \sum_{Dj \in Dr} J(D_j, Q_u^{(s)})}{\frac{1}{|Dnr|} * \sum_{Dj \in Dnr} J(D_j, Q_u^{(s)})}$$

Où :
Dr : Ensemble de documents pertinents retrouvés à travers les générations de l'AG
Dnr : Ensemble de documents non pertinents retrouvés à travers les générations de l'AG
$Q_u^{(s)}$: Individu requête à la génération s de l'AG

$J(D_j , Q_u^{(s)})$: Mesure de Jaccard définie par $J(D_j, Q_u^{(s)}) = \frac{\sum_{i=1}^{T} q_{ui}^{(s)} d_{ji}}{\sum_{i=1}^{T} q_{ui}^{2} + \sum_{i=1}^{T} d_{ji}^{2} - \sum_{i=1}^{T} q_{ui}^{(s)}}$

Où :
$q_{ui}^{(s)}$: Poids du terme t_i dans la requête $Q_u^{(s)}$
d_{ji} : Poids du terme t_i dans le document D_j

La fonction d'adaptation ainsi formulée, avantageraient la reproduction de requêtes dont la composition pondérée de termes se rapproche des documents pertinents et s'éloigne des documents non pertinents.
Cependant, nous pensons que la fonction « pertinence » est multimodale, en ce sens que des documents pertinents à un même besoin en information peuvent avoir des descripteurs différents et par conséquent être situés à des régions différentes de l'espace documentaire. Or, L'AG classique fait évoluer la population de requêtes en convergeant vers **une requête optimale**, qui forcément, est à l'origine du rappel de documents de structures proches, ignorant éventuellement des documents pertinents mais de structures spécifiques différentes. A cet effet, la fonction d'adaptation sera ajustée de manière à favoriser la conservation de requêtes de performances comparables explorant dans des directions différentes. Les techniques de nichage et de spéciation [Goldberg, 1989] nous conduisent à définir des niches de requêtes. Une niche de requêtes est, dans notre cas, un ensemble de requêtes restituant des documents de génotype ressemblants.
Pour notre part, nous proposons la définition suivante de la fonction de partage :

$$Niche(Q_u^{(s)}) = \{ Q_v^{(s)} / Dist\_Euclid(Q_u^{(s)}, Q_v^{(s)}) \leq Seuil\ de\ nichage \}$$

Où :
Dist_Euclid est la fonction distance Euclidienne définie par :

$$Dist\_Euclid(Q_u^{(s)}, Q_v^{(s)}) = sqrt(\sum (q_{ui}^{(s)} - q_{vi}^{(s)})$$

On donne alors la formulation suivante pour la fonction d'adaptation ajustée :

$$Fitness(Q_u^{(s)}) = \frac{QFitness(Q_u^{(s)})}{\left| Niche(Q_u^{(s)}) \right|}$$

Il en résulte que le fitness d'un individu requête est proportionnel à son rapport de similitude aux documents pertinents et documents non pertinents et inversement proportionnel au nombre de requêtes de sa niche. Ceci encouragerait la sélection vers la reproduction dans des niches moins peuplées; aussi, de nouvelles régions seraient elles explorées en accord avec la valeur moyenne des *fitness des requêtes associées.*

4.4. Les Opérateurs génétiques

Un nombre considérable de techniques ont été mises au point pour accroître les performances du processus de recherche d'information notamment par reformulation de requête. Les expérimentations ont montré que certains paramètres conditionnent les performances du système : méthode de sélection des nouveaux termes de la requête, méthode de repondération des termes de la requête, nombre de termes ajoutés à la requête. Nous avons alors exploité ces résultats pour mettre en oeuvre des opérateurs génétiques non aveugles, augmentés par une connaissance issue des techniques d'expansion et repondération de requête. L'utilisation d'une connaissance auxiliaire propre au problème de la recherche d'information permettrait d'accélérer l'exploration génétique par une recherche guidée dans l'espace des documents.

4.4.1. La sélection

Après calcul de la valeur d'adaptation de chaque individu requête, intervient l'opération de sélection.

Nous avons opté pour une sélection basée sur la méthode usuelle de la roue de loterie. Dans notre cadre d'application, la méthode se traduit par le déroulement de l'algorithme suivant :

1. ***Pour*** chaque individu requête $Q_i^{(s)}$, calculer la valeur d'adaptation relative selon la formule :

$$Fitness\_Rel(Q_i^{(s)}) = Fitness(Q_i^{(s)}) \Big/ \sum_{j=1}^{Taille\_Pop} Fitness(Q_j^{(s)})$$

2. ***Pour*** chaque individu requête $Q_i^{(s)}$, calculer la probabilité de sélection :

$$Pselect(Q_i^{(s)}) = \sum_{j=1}^{i-1} Fitness\_Rel(Q_j^{(s)})$$

3. Taille_New_Pop := 0

4. ***Répéter***
5. Générer un nombre aléatoire *r* sur l'intervalle *[0 1]*
6. ***Si*** $Pselect(Q_{i-1}^{(s)}) < r < Pselect(Q_i^{(s)})$ ***Alors***
   6.1. Générer un clone de $Q_i^{(s)}$
   6.2. . *Taille_New_Pop := Taille_New_Pop +1*
7. ***Jusqu'à*** *Taille_New_Pop =Taille_Pop*

La méthode consiste essentiellement à attribuer à chaque requête de la génération courante, un nombre de clones proportionnel à sa valeur d'adaptation relative avec un biais lié au tirage aléatoire.

4.4.2. Le croisement

Les opérateurs de croisement que nous définissons dans la suite ont pour but d'exploiter au mieux la distribution des termes dans les documents pertinents, l'occurrence des termes par combinaison (non de manière indépendante) ainsi que les associations sémantiques établies selon des formules de cooccurrence dans la collection. Nous définissons trois types de croisement : croisement basé sur le poids des termes, croisement basé sur la cooccurrence des termes dans la collection et croisement aveugle.

4.4.2.1. Croisement basé sur la pertinence des termes

Ce type de croisement est sans site , visant la modification des poids des termes d'indexation des requêtes sélectionnées et ce, sur la base de leur distribution dans les documents pertinents et documents non pertinents.
Ce croisement revient à augmenter les poids d'indexation des termes fréquents dans les documents pertinents et, diminuer les poids des termes d'indexation relativement fréquents dans les documents non pertinents.

$$\left.\begin{array}{l} Q_u^{(s)}\,(\,q_{u1}^{(s)}, q_{u2}^{(s)}, \ldots, q_{uT}^{(s)}) \\ Q_v^{(s)}\,(\,q_{v1}^{(s)}, q_{v2}^{(s)}, \ldots, q_{vT}^{(s)}) \end{array}\right\} \longrightarrow Q_p^{(s+1)}\,(\,q_{p1}^{(s+1)}, q_{p2}^{(s+1)}, \ldots, q_{pT}^{(s+1)})$$

$$q_{pi}^{(s+1)} = Max\,(q_{ui}^{(s)}, q_{vi}^{(s)}) \text{ si } Poids\,(t_i, Dr^{(s)}) \geq Poids\,(t_i, Dnr^{(s)})$$
$$Min\,(q_{ui}^{(s)}, q_{vi}^{(s)}) \text{ sinon}$$

Où : $Poids(t_i,D)=\sum_{dj\in D} d_{ji}$

Exemple

Soient :

- Les requêtes suivantes sélectionnées pour le croisement :

$$Q_u^{(s)} = (t_1\ 0.2\ \ t_2\ 0.6\ \ t_{13}\ 0.8\ \ t_{15}\ 0.1) \qquad Q_v^{(s)} = (t_1\ 0.4\ \ t_3\ 0.1\ \ t_{10}\ 0.8\ \ t_{12}\ 0.6\ \ t_{15}\ 0.4)$$

- Les documents pertinents sélectionnés

$$Dr_1 = (t_1\ 0.2\ \ t_2\ 0.6\ \ t_{13}\ 0.8\ \ t_{15}\ 0.5)$$

$$Dr_2 = (t_1\ 0.3\ \ t_3\ 0.4\ \ t_{10}\ 0.8\ \ t_{12}\ 0.4)$$

$$Dr_3 = (t_1\ 0.4\ \ t_3\ 0.4\ \ t_{10}\ 0.8\ \ t_{15}\ 0.2)$$

- Les documents non pertinents sélectionnés :

$$Dnr_1 = (t_1\ 0.1\ \ t_8\ 0.8\ \ t_{10}\ 0.4\ \ t_{12}\ 0.5\ \ t_{15}\ 0.4)$$

$$Dnr_2 = (t_1\ 0.2\ \ t_4\ 0.3\ \ t_8\ 0.1\ \ t_{10}\ 0.1\ \ t_{15}\ 0.8)$$

Les poids des termes d'indexation dans les documents pertinents et documents non pertinents sont alors les suivants :

t_i	Poids(t_i , Dr)	Poids(t_i , Dnr)
t_1	0.65	0.15
t_2	0.4	0.8
t_3	0.4	0
t_{10}	0.7	0.25
t_{12}	0.1	0.4
t_{13}	0.5	0
t_{14}	0.9	0
t_{15}	0.2	0.6

L'individu requête issu du croisement est alors le suivant :

$$Q_p^{(s+1)} = (t_1\ 0.4\ \ t_3\ 0.1\ \ t_{10}\ 0.8\ \ t_{13}\ 0.8\ \ t_{15}\ 0.1)$$

On retient les valeurs maximales des poids pour les termes t_1, t_3, t_{10} et t_{13} et valeurs minimales pour les termes t_2, t12 et t_{15}

4.4.2.2. Croisement basé sur la cooccurrence des termes dans la collection

Après définition d'un site de croisement à deux requêtes sélectionnées, ce type de croisement établit entre ces dernières, un échange croisé de poids entre termes sémantiquement liés.

Les requêtes sélectionnées pour le croisement s'échangent éventuellement, une partie de leur structure délimitée par le site de croisement. Les poids des termes issus de la partie à croiser, sont échangés sous condition que leur poids de cooccurrence dans la collection, soit supérieur à un seuil significatif relativement à une relation sémantique.

Ce croisement vise à réduire les effets dûs à un éventuel mauvais choix des termes d'indexation, par composition avec des termes associés. Sa performance dépend, dans une large mesure, de la performance des formules de cooccurrence utilisées.

$$Q_u^{(s)}(q_{u1}^{(s)}, q_{u2}^{(s)}, \ldots, q_{uT}^{(s)}) \quad \leftrightarrow \quad Q_{p1}^{(s+1)}(q_{p11}^{(s+1)}, q_{p2}^{(s+1)}, \ldots, q_{p1T}^{(s+1)})$$

$$Q_v^{(s)}(q_{v1}^{(s)}, q_{v2}^{(s)}, \ldots, q_{vT}^{(s)}) \quad \leftrightarrow \quad Q_{p2}^{(s+1)}(q_{p21}^{(s+1)}, q_{p22}^{(s+1)}, \ldots, q_{p2T}^{(s+1)})$$

Soit *c* : le site de croisement, alors on a :

$$Si \quad (c>i)\vee(Cooc(t_{ui},t_{vi})_{tui\neq tvi}<Seuil\_Cooc) \quad Alors\begin{cases} q_{p1i}^{(s+1)}=q_{ui}^{(s)} \\ q_{p2i}^{(s+1)}=q_{vi}^{(s)} \end{cases} \quad Sinon\begin{cases} q_{p1i}^{(s+1)}=q_{vi}^{(s)} \\ q_{p2i}^{(s+1)}=q_{ui}^{(s)} \end{cases}$$

Où :

t_{ui} : ième terme de la requête $Q_u^{(s)}$

$t_{vi}^{(s)}$: ième terme de la requête $Q_v^{(s)}$

$Cooc(t_i,t_j)$: Poids de l'association de cooccrrence entre les termes t_i et$_j$ dans la collection calculée selon la formule :

$$Cooc(t_i,t_j)=\alpha*\frac{\sum_{k=1}^{N}(d_{ki}*d_{kj})}{\sqrt{\sum_{k=1}^{N}d_{ki}^2+\sum_{k=1}^{N}d_{kj}^2-\sum_{k=1}^{N}(d_{ki}*d_{kj})}}$$ avec α : Constante réelle

Seuil_Cooc : Seuil de coccurrence entre termes

Exemple

Soient les requêtes exemples $Q_u^{(s)}$ et $Q_v^{(s)}$:

$$Q_u^{(s)} = \underset{t_1\ t_2\ t_{13}\ t_{15}}{(0.2\ \ 0.6\ \ 0.8\ \ 0.1)} \qquad Q_v^{(s)} = \underset{t_1\ t_3\ t_{10}\ t_{12}\ t_{15}}{(0.4\ \ 0.1\ \ 0.8\ \ 0.6\ \ 0.4)}$$

On suppose en outre que :
- Le site de croisement est *3*
- Le seuil de cooccurrence est *0.5*
- Les poids de cooccurrence des paires de termes (t_2,t_3) et (t_{10}, t_{13}) sont :
  $Cooc(t_2,t_3) = 0.4$, $Cooc(t_{10},t_{13}) = 0.7$.

On obtient alors les requêtes enfants suivantes :

$$Q_{p1}^{(s+1)} = (\underset{t_1}{0.2}\ \underset{t_2}{0.6}\ \underset{t_{10}}{0.8}\ \underset{t_{15}}{0.1}) \qquad Q_{p2}^{(s+1)} = (\underset{t_1}{0.4}\ \underset{t_3}{0.1}\ \underset{t_{12}}{0.6}\ \underset{t_{13}}{0.8}\ \underset{t_{15}}{0.4})$$

On note l'échange des termes t_{10} et t_{13} entre les requêtes $Q_u^{(s)}$ et $Q_v^{(s)}$

4.4.2.3. Croisement aveugle

Cet opérateur reprend le procédé de croisement classique défini dans un AG standard. Après détermination aléatoire d'un site de croisement *c*, deux requêtes sélectionnées échangent une partie de leur structure. Cet opérateur est défini comme suit :

$$Q_u^{(s)}(q_{u1}^{(s)}, q_{u2}^{(s)}, q_{uc}^{(s)}, q_{uc+1}^{(s)}, .., q_{uT}^{(s)}) \leftrightarrow Q_{p1}^{(s+1)}(q_{u1}^{(s+1)}, q_{u2}^{(s+1)}, q_{vc}^{(s)}, q_{vc+1}^{(s)} ...., q_{vT}^{(s)})$$

$$Q_v^{(s)}(q_{v1}^{(s)}, q_{v2}^{(s)}, q_{vc}^{(s)}, q_{vc+1}^{(s)}, .., q_{vT}^{(s)}) \leftrightarrow Q_{p2}^{(s+1)}(q_{v1}^{(s+1)}, q_{v2}^{(s+1)}, q_{uc}^{(s)}, q_{uc+1}^{(s)}, \ldots, q_{uT}^{(s)})$$

$$Si \quad (c<i) \quad Alors \begin{cases} q_{p1i}^{(s)} = q_{ui}^{(s)} \\ q_{p2i}^{(s)} = q_{vi}^{(s)} \end{cases} \quad Sinon \begin{cases} q_{p1i}^{(s)} = q_{vi}^{(s)} \\ q_{p2i}^{(s)} = q_{ui}^{(s)} \end{cases}$$

Exemple

Soient les requêtes exemples $Q_u^{(s)}$ et $Q_v^{(s)}$:

$$Q_u^{(s)} = (\underset{t_1}{0.2}\ \underset{t_2}{0.6}\ \underset{t_{13}}{0.8}\ \underset{t_{15}}{0.1}) \qquad Q_v^{(s)} = (\underset{t_1}{0.4}\ \underset{t_3}{0.1}\ \underset{t_{10}}{0.8}\ \underset{t_{12}}{0.6}\ \underset{t_{15}}{0.4})$$

On suppose en outre que :
- Le site de croisement est *3*

On obtient alors les requêtes enfants suivantes :

$$Q_{p1}^{(s+1)} = (\underset{t_1}{0.2}\ \underset{t_2}{0.6}\ \underset{t_{12}}{0.6}\ \underset{t_{15}}{0.4}) \qquad Q_{p2}^{(s+1)} = (\underset{t_1}{0.4}\ \underset{t_3}{0.1}\ \underset{t_{10}}{0.8}\ \underset{t_{15}}{0.1})$$

Les poids d'indexation des termes t_{12} et t_{15} situés après le site de croisement (3) sont échangés entre les requêtes $Q_u^{(s)}$ et $Q_v^{(s)}$

4.4.3. La mutation

Nous définissons deux types de mutation : mutation basée sur la pertinence des termes et mutation aveugle.

4.4.3.1. Mutation basée sur la pertinence des termes

Cet opérateur consiste essentiellement à exploiter les termes occurrents dans les documents pertinents dans le but d'ajuster les valeurs des gènes correspondants dans les requêtes sélectionnées pour la mutation.

Son procédé, induit de fait une **expansion contextuelle** de requête avec modification des poids d'indexation. Il convient, en outre de noter que la probabilité de mutation est non uniforme entre les termes d'indexation; elle dépend étroitement de la composition de la requête sélectionnée pour l'opération. Plus précisément, cet opérateur établit une expansion de requête dont la source est basée sur la constitution d'une liste *Lmut* de nouveaux termes issus des documents pertinents et triée selon un score que l'on calcule comme suit :

$$Score(t_i)=\frac{\sum_{Dj\in Dr^{(0)}} d_{ji}}{\left\| Dr^{(s)} \right\|}$$

La mutation ainsi définie consiste à établir une expansion de requête basée sur le contexte courant. En outre, la liste *Lmut* est limitée à une taille issue des expérimentations préalables concernant le paramètre *nombre de termes d'expansion.*

Le poids d'un terme t_i ajouté à la requête est calculé selon la formule :

$$q_{ui}=\frac{Score(t_i)}{Max_{tj\in Q_u^{(s)}} q_{uj}}$$

Plus précisément, la mutation est effectuée selon l'algorithme suivant :

Début

Pour chaque t_i dans la liste Lmut ***Faire***

Si (random(p)<Pm) ***Alors***

$q_{ui}^{(s)}$ = Poids_Moyen($Q_u^{(s)}$)-δ

Finsi

Fait

Fin

Avec :

Random(p) : Fonction qui génére un nombre aléatoire p dans l'intervalle [0 1]

$$Poids\_Moyen(Q_u^{(s)})=\frac{\sum_{j=1}^{T} q_{ui}^{(s)}}{n_{q_{ui}^{(s)}}}$$

Où

$n_{qui}^{(s)}$: Taille effective de l'individu requête $Q_u^{(s)}$ (nombre de termes d'indexation de poids non nuls)

δ : Paramètre de contrôle du poids moyen

Exemple

t_1 t_2 t_{13} t_{15}

Soit la requête exemple $Q_u^{(s)} = (0.2\ 0.6\ 0.8\ 0.1)$ et distributions des termes dans les documents pertinents, précédemment présentée (Cf. paragraphe 4.4.2.2).

On construit alors la liste ordonnée $Lmut\ (t_{14}, t_{10}, t_1, t_{13}, t_3, t_{15}, t_{12})$

En supposant que la génération du nombre aléatoire indique la mutation des termes t_{14} et t_3 , on calcule $q_{i14} = 0.4$ et $q_{i3} = 0.4$ et obtient la requête enfant :

t_1 t_2 t_3 t_{10} t_{13} t_{14} t_{15}

$Q_u^{(s+1)} = (0.2\ 0.6\ 0.4\ 0.6\ 0.8\ 0.4\ 0.1)$

Expansion de la requête avec les termes t_{14} et t_3

4.4.3.2. Mutation aveugle

Cet opérateur reprend le procédé de mutation classique défini dans un AG standard. Après détermination aléatoire d'un terme de requête sélectionnée, son poids d'indexation est muté à une valeur aléatoire comprise dans l'intervalle *[0 1*]. Notons que dans le cas de cet opérateur, tous les termes de la requête sélectionnée ont une probabilité de mutation uniforme.

Exemple

t_1 t_2 t_{13} t_{15}

Soit la requête $Q_u^{(s)}$ exemple $Q_u^{(s)} = (0.2\ 0.6\ 0.8\ 0.1)$. On suppose que la génération d'un nombre aléatoire indique la mutation du terme t_{13} et que le poids aléatoire tiré est *0.4*, la requête enfant est lors :

t_1 t_2 t_{13} t_{15}

$Q_u^{(s+1)} = (0.2\ 0.6\ 0.4\ 0.1)$

5. Principe de fusion des résultats de recherche

A l'issue de l'évaluation de chaque individu requête, nous obtenons des ensembles non disjoints de documents restitués par le processus de recherche de base. Ces listes partielles sont fusionnées dans le but de construire une liste unique de documents présentée à l'utilisateur, lors d'une itération feedback donnée.

La liste est obtenue en respectant l'ordre donné par la formule :

$$Rel(D_j) = \sum_{Q_u^{(s)} \in Pop^{(s)}} RSV(Q_u^{(s)}, D_j)$$

Où :

$RSV(Q_u^{(s)}, D_j)$: Valeur de pertinence du document D_j en présentant l'individu $Q_u^{(s)}$ au processus de recherche de base

$Pop^{(s)}$: Population à la génération s

La valeur de pertinence d'un document dépend ainsi des valeurs d'adaptation des requêtes à l'origine de sa restitution.

6. Evaluation globale de l'approche

L'évaluation constitue sans doute une étape importante lors de la mise en œuvre de modèles et techniques de recherche d'information. L'évaluation permet en effet d'estimer l'impact de chacun des éléments de toute approche proposée sur les résultats de recherche et de fournir des bases de comparaison des performances entre différentes approches.
Nous avons procédé à un ensemble d'expérimentations, en utilisant le SRI Mercure [Boughanem & SouleDupuy, 1997] présenté en annexe. Nous avons en premier lieu, expérimenté notre approche sur les petites collections CACM et ADI [Tamine, 1997]. Cette première série d'expérimentations a montré globalement l'intérêt de notre approche. En effet, on a enregistré un taux d'accroissement moyen des performances de *58%* en utilisant les différents opérateurs et ce, comparativement à une recherche simple sans intervention de l'AG. De plus, les résultats obtenus montrent l'intérêt de l'intégration de la connaissance dans la structure des opérateurs et ce par comparaison aux résultats obtenus par application des opérateurs aveugles.

En second lieu, nous avons procédé à une série d'expérimentations sur la collection TREC6 French Data avec *21* requêtes [Boughanem & al, 1999] décrite sur le tableau 4.2. Les expérimentations ont pour but d'évaluer l'apport de l'approche de manière générale sur une collection volumineuse et impact des paramètres et heuristiques de l'AG sur les résultats de la recherche. Ces expérimentations sont détaillées dans ce qui suit.

6.1. Conditions expérimentales

Nous avons procédé à un ensemble d'expérimentations dont l'objectif est d'évaluer :

1. L'apport de notre approche d'optimisation de requête pour l'amélioration des performances du SRI. Cette évaluation est effectuée en utilisant des estimations comparatives des nombres de documents pertinents restitués par itération ainsi que nombre de documents pertinents cumulé et ce, relativement à une recherche de base menée dans le SRI Mercure.

2. L'impact des éléments de l'AG sur les résultats de recherche : probabilité de croisement, probabilité de mutation, taille de la population et types d'opérateurs génétiques appliqués.

6.1.1. Paramètres de l'AG

Notre processus de reformulation de requête étant essentiellement basé sur l'exécution d'un AG, nous avons été dès lors, confronté à la tâche ardue de détermination des paramètres adéquats.
A cet effet, nous avons adopté une approche expérimentale qui consiste à effectuer de nombreux essais pour la sélection des meilleures conditions de déroulement de l'algorithme.
Sur la base de la structure de l'algorithme d'une part, et objectifs de l'expérimentation d'autre part, nous avons caractérisé deux catégories de paramètres de contrôle de l'algorithme :

1. ***Première catégorie***
   Cette catégorie comprend les paramètres taille de la population (*Taille_Pop*), probabilité de croisement (*Pc*), probabilité de mutation (*Pm*) et taille de la liste des termes d'expansion *Lmut*. Les paramètres *Taille_Pop, Pc* et *Pm* ont fait l'objet de nombreuses expérimentations développées dans la suite.
   En ce qui concerne le paramètre taille de la liste *Lmut*, nous avons retenu la valeur *30* qui correspond à à une valeur standard recommandée pour l'expansion de requêtes [Harman, 1992].

2. ***Deuxième catégorie***
   Cette catégorie concerne des paramètres de niveau de contrôle supérieur, permettant de définir l'ossature de l'algorithme.
   Cette catégorie comprend le type d'opérateurs génétiques appliqués qui a fait l'objet d'expérimentations développées dans la suite.

6.1.2. Jeu d'opérateurs génétiques

Le tableau 4.1 présente les différents jeux d'opérateurs génétiques, que nous avons expérimentés. Le choix des jeux d'essai des opérateurs génétiques est ainsi établi afin d'évaluer l'apport intrinsèque des différents types d'opérateurs génétiques, en considérant l'éventuel effet de dépendance entre type de croisement et type de mutation.

	Croisement	***Mutation***
G1	Croisement basé sur le poids des termes	Mutation basée sur la pertinence des termes
G2	Croisement basé sur la cooccurrence	Mutation basée sur la pertinence des termes
G3	Croisement aveugle	Mutation aveugle

Tableau 4.1 : Groupes d'opérateurs génétiques

6.1.3. Méthode d'évaluation

La mesure de performances du processus de recherche d'information a été effectuée moyennant la méthode d'évaluation de la collection résiduelle. Cette méthode est adoptée afin d'évaluer le processus d'optimisation de requête induit par l'AG et ce, à des itérations feedback successives correspondant à des générations de l'algorithme.
Nous avons, à cet effet, mis en œuvre la base de référence d'évaluation notée *Baseline*, qui considère comme référence, à chaque génération *s* de l'AG, l'apport de nouveaux documents relativement à la génération *s-1*.
Ce type d'évaluation mesure ainsi la capacité de l'AG à atteindre de nouveaux documents pertinents lors de l'évaluation de chaque nouvelle génération de requêtes.
On calcule pour chaque requête de la base de test, la précision à 11 points de rappel fixes *0.0, 0.1, ..., 0.9,1* puis le nombre de documents pertinents restitués et nombre de documents pertinents cumulé à chaque itération et ce, sur l'ensemble des requêtes de la base.
En outre, l'accroissement des performances est calculé sur la base d'une :
- liste de ***15*** documents présentés et soumis au jugement de l'utilisateur : cette valeur est communément utilisée pour l'évaluation des techniques de recherche d'information [Robertson & Walker, 1997] [Harman, 1992] [Boughanem & all, 199].
- série de *5* itérations feddback,
- recherche initiale se limitant à l'interrogation du système Mercure avec la requête utilisateur.

6.1.4. Collection de test

Nous avons utilisé la collection de test TREC6 French Data décrite sur le tableau 4.2. Les documents de la collection sont issus de la collection SDA News (Schwerzerischen Depeschenagentue swiss news Agency). Les requêtes sont issues des topics numérotées 1-21 de la collection TREC ; nous avons utilisé les champs titre, description et narrative.

TREC6 French Data		
Nombre de documents dans la collection	***Nombre de termes dans la collection***	***Taille moyenne d'un document***
141490	128922	127.70

Tableau 4.2 : Description de la collection de test

6.1.5. L'algorithme de recherche

L'expérimentation est basée sur l'exécution de l'algorithme de recherche suivant :

0. Iter :=*0*
1. Présenter puis évaluer la requête initiale
2. Juger les *15* premiers documents
3. Construire la population initiale de requêtes
4. Calculer l'adaptation de chaque individu requête
5. Appliquer les opérateurs génétiques
6. Effectuer la recherche pour chaque individu requête
7. Fusionner les résultats
8. Juger les *15* premiers documents
9. Iter :=Iter+1
10. Si (Iter<5) Alors aller à 4

6.2. Evaluation des probabilités de croisement et probabilité de mutation

L'optimisation génétique est un processus stochastique régulé par les valeurs de probabilité de croisement *Pc* et probabilité de mutation *Pm*. Notre première série d'expérimentations consistait à évaluer l'impact de ces valeurs sur les résultats de recherche. A cet effet, nous avons fait varier les valeurs de *Pc* et *Pm* dans les ensembles respectifs de valeurs *{0.1,0.25,0.5,0.7}* et *{0.01,0.07,0.1,0.25,0.5}*.

Le tableau 4.3 [Boughanem & al, 1999] présente le nombre de documents pertinents et nombre de documents pertinents cumulé à chaque itération, pour le jeu d'opérateurs génétiques *G1*.

On constate que les valeurs des probabilité *Pc* et *Pm* ont un impact considérable sur les résultats de recherche. On note particulièrement une faible variation du nombre de documents pertinents cumulés pour différentes valeurs de *Pm* et même valeur de *Pc* : *259* à *260* pour *Pc=0.1*, *268 à 269* pour *Pc=0.25*, *266 à 275* pour *Pc=0.5* et *290* à *292* pour *Pc =0.7*. En revanche, la variation de la valeur de *Pc* conduit à une variation relativement notable du nombre de documents pertinents cumulés : *259* à *292*. On en conclut que la valeur de *Pc* joue un rôle plus important que la valeur de *Pm*.

	Iter1	Iter2	Iter3	Iter4	Iter5
Pc=0.1					
0.01	91(91)	36(127)	43(170)	53(223)	36(259)
0.07	91(91)	36(127)	43(170)	53(223)	37(260)
0.1	91(91)	36(127)	43(170)	53(223)	37(260)
0.25	91(91)	36(127)	44(171)	53(223)	35(259)
0.5	91(91)	37(128)	44(171)	49(224)	36(260)
Pc=0.25					
0.01	89(89)	60(149)	43(192)	36(228)	41(269)
0.07	89(89)	60(149)	43(192)	36(228)	40(268)
0.1	89(89)	60(149)	43(192)	36(228)	40(268)
0.25	89(89)	60(149)	43(192)	36(228)	40(268)
0.5	89(89)	60(149)	43(192)	38(230)	38(268)
Pc=0.5					
0.01	92(92)	49(141)	49(190)	48(238)	37(275)
0.07	92(92)	49(141)	49(190)	48(238)	36(274)
0.1	92(92)	49(141)	49(190)	48(238)	36(274)
0.25	92(92)	49(141)	49(190)	48(238)	37(275)
0.5	92(92)	49(141)	47(188)	46(234)	34(266)
Pc=0.7					
0.01	94(94)	54(148)	69(217)	39(226)	36(292)
0.07	94(94)	54(148)	69(217)	39(256)	36(292)
0.1	94(94)	54(148)	69(217)	39(256)	36(292)
0.25	94(94)	54(148)	69(217)	39(256)	34(290)
0.5	94(94)	54(148)	68(216)	40(256)	34(290)

Tableau 4.3 : Impact des valeurs *Pc* et *Pm* sur les résultats de recherche

Ceci confirme l'intérêt du croisement relativement à la mutation comme développé dans la littérature des AG's [Goldberg, 1989] [Sebag & Shoneauer, 1996]. Nous avons retenu pour les expérimentations suivantes la meilleure configuration, à savoir *Pc=0.7* et *Pm=0.07*.

6.3. Evaluation de la taille de la population

La littérature des AG's met en évidence la difficulté de déterminer la taille de population adéquate pour atteindre les résultats de recherche escomptés à un nombre de générations relativement réduit.

Une population de taille importante conduirait à un bruit considérable et coût d'exécution élevé. A l'inverse, une taille de population limitée restreindrait l'exploration à un voisinage très proche de la population initiale.

Nous avons expérimenté l'effet de la taille de la population sur les résultats de recherche en utilisant les jeux d'opérateurs *G1* et *G2*.

Le tableau 4.4 [Boughanem & al, 1999] présente le nombre de documents pertinents et nombre de documents pertinents cumulés retrouvés à chaque itération de l'AG pour ces différents jeux d'opérateurs.

Les résultats donnés pour une exécution *Sans AG* sont ceux obtenus par considération des *15* documents suivants à chaque itération et ce, à partir de la liste obtenue lors de la recherche initiale. Les résultats donnés pour une exécution *Avec AG* sont ceux obtenus par considération des *15* documents de la liste de référence *Baseline* issue de l'application de l'AG d'optimisation de requête.

	G1					**G2**				
	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**	**Iter1**	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Taille_Pop=2										
Sans AG	90(90)	48(138)	33(171)	46(217)	24(241)	90(90)	59(149)	36(185)	40(195)	21(216)
Avec AG	72(72)	52(124)	36(160)	46(206)	24(230)	72(72)	42(112)	42(154)	35(189)	21(210)
Taille_Pop=4										
Sans AG	90(90)	57(147)	50(197)	42(239)	42(281)	90(90)	54(144)	46(190)	40(230)	41(273)
Avec AG	96(96)	64(160)	40(200)	45(245)	42(287)	85(85)	40(125)	46(171)	58(229)	45(274)
Taille_Pop=6										
Sans AG	57(147)	39(186)	44(230)	46(217)	37(267)	90(90)	64(154)	40(194)	31(225)	31(256)
Avec AG	96(96)	64(160)	55(215)	51(266)	30(296)	85(85)	64(149)	41(190)	51(241)	37(278)
Taille_Pop=8										
Sans AG	90(90)	64(154)	42(196)	37(233)	42(275)	90(90)	64(154)	36(190)	40(230)	30(260)
Avec AG	96(96)	72(168)	50(218)	38(256)	29(285)	85(85)	64(159)	45(204)	36(240)	30(270)

Tableau 4.4 : Impact de la taille de la population sur les résultats de recherche

On constate que le nombre de documents pertinents cumulé pour une exécution *Avec AG* est plus élevé que celui enregistré avec une exécution *Sans AG* à toutes les itérations feedback et en appliquant les différents jeux d'opérateurs génétiques *G1* et *G2* et ce, dès que la taille de population dépasse *2*. En analysant l'impact de la taille de la population, on note que pour les tailles de population *4, 6 et 8*, on enregistre les séries de valeurs de nombres de documents pertinents cumulés *(287 ,296, 285)* et *(274 ,278, 270)* et ce respectivement pour les groupes d'opérateurs *G1* et *G2*. Un compromis entre les

performances et coût de recherche nous a amené à retenir la taille de population *6* pour les expérimentations ultérieures .
On note d'emblée que le groupe d'opérateurs *G1* donne de meilleurs résultas ; l'analyse est dans ce sens développée dans le paragraphe suivant.

6.4. Impact des opérateurs génétiques augmentés

Nous évaluons à présent l'intérêt d'intégrer une connaissance spécifique au domaine de la recherche d'information, dans la structure des opérateurs génétiques.
L'effet des opérateurs augmentés est évalué par comparaison des résultats de recherche obtenus avec ceux obtenus par application des opérateurs aveugles (classiques). Le tableau 4.5 présente les résultats d'exécution de l'AG utilisant les opérateurs augmentés par la connaissance, représentés par les groupes *G1* et *G2* et opérateurs classiques, représentés par le groupe *G3*. La ligne *Accroissement/Doc_Pert_Cum* traduit le taux d'accroissement du nombre de documents pertinents cumulés à chaque itération, par application d'un groupe d'opérateurs augmentés et ce, comparativement à l'application du groupe d'opérateurs aveugles.

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
G3	27(27)	28(55)	41(96)	28(126)	24(150)
G1	96(96)	64(160)	55(215)	51(266)	30(296)
Accroissement/Doc_Pert_Cum	255%	190%	123%	111%	97%
G2	85(85)	64(149)	41(190)	51(241)	37(278)
Accroissement/Doc_Pert_Cum	214%	170%	97%	91%	85%

Tableau 4.5 : Impact des opérateurs augmentés par la connaissance sur les résultats de recherche

Il apparaît clairement l'intérêt de l'application des opérateurs augmentés proposés et ce, à toutes les générations de l'AG. On note en effet, que le nombre de documents pertinents total cumulé en utilisant les jeux d'opérateurs *G1* et *G2*, est nettement supérieur à celui obtenu en utilisant le jeu d'opérateurs *G3*, correspondant à l'application des opérateurs aveugles présentés précédemment.
Plus précisément, on enregistre un accroissement à la $5^{ème}$ itération feedback de *97%* et *85%* respectivement pour les groupes d'opérateurs *G1* et *G2*.
Par ailleurs, on constate que le groupe *G1* donne de meilleurs résultats (*296* documents pertinents cumulés à la $5^{ème}$ itération) que le groupe *G2* (*278* documents pertinents cumulés à la $5^{ème}$ itération) aussi bien en mesurant les performances par itération qu'à l'issue de la $5^{ème}$ itération.
Ces groupes d'opérateurs différent par le type de croisement appliqué. Les résultats moins performants obtenus par le groupe *G2* peuvent être imputés à l'utilisation de

l'association de cooccurrence lors de transformations génétiques des individus requêtes. Cette technique d'expansion est effectivement critiquée par de nombreux auteurs du domaine [Peat & Willet, 1991] [Schutze & Pederson, 1997].

7. Bilan et nouvelles directions

Nous avons défini une approche d'optimisation de requête en s'appuyant sur deux volets d'étude : la stratégie de reformulation de requête d'une part et concepts de la génétique d'autre part.

L'approche de reformulation de requête que nous préconisons, vise l'efficacité de la recherche d'information à travers une exploration robuste du fond documentaire, l'indépendance de l'hypothèse de ressemblance des documents et l'exploitation de techniques d'injection de pertinence.

A cet effet, nous mettons en œuvre un AG d'optimisation de requête qui permet une recherche coopérative d'informations, pouvant s'intégrer à de nombreux modèles de recherche de base.

Il est caractérisé par l'intégration de la technique de nichage dans le but d'assurer le rappel de documents possédant des descripteurs relativement différents et pertinents pour une même requête. En outre, les opérateurs génétiques proposés, sont non aveugles, puisque traduisent des procédés de repondération et expansion de requêtes inspirés des techniques du domaine, et qui ont pour but d'assurer une convergence rapide des individus requêtes vers le rappel de documents pertinents.

Les expérimentations préliminaires réalisées sur les collections CACM, ADI et TREC6 ont montré l'intérêt de notre approche. Cependant, nous exploitons les résultats de ces expérimentations d'une part, et éléments de la théorie des AG's d'autre part, pour formuler des bases d'amélioration du processus génétique de recherche d'information proposé. Les nouvelles directions de nos travaux sont définies par révision de deux éléments fondamentaux de l'AG :

1. ***Exploitation de la technique de nichage***

   L'AG présente une faible exploitation de la technique de nichage. En effet, excepté une mesure d'adaptation d'individu ajustée par la taille de la niche associée, il n'existe pas d'identification effective des niches. La population est manipulée linéairement comme un ensemble d'individus aussi bien lors de l'application des opérateurs que lors de la définition de l'ordre global issu de l'évaluation des individus requêtes.

   En effet, la pertinence supposée d'un document, ne considère en aucun cas la valeur d'adaptation de la niche de requête à l'origine de sa restitution.

2. ***Mesure de l'adaptation des individus requêtes***

La mesure d'adaptation proposée (Cf 3.2.2.) pose deux principaux inconvénients :

- La mesure n'est pas corrélée aux mesures de performance rappel/précision. Il y a ainsi risque de sélectionner des requêtes dont l'adaptation n'est pas étalonnée à la mesure de rappel/précision.
- La formule de définition des niches n'est pas normalisée, et donc difficile à seuiller ; ceci engendre un paramètre supplémentaire, non évident à définir en fonction des caractéristiques des collections interrogées.

  De plus, la valeur d'un seuil, ne traduit , dans ce cas, aucun lien entre la composante d'une niche de requêtes et les documents résultant de leur évaluation.

Ces points de réflexion nous ont amené ainsi à définir une nouvelle structure et fonctionnement général de l'AG d'optimisation de requête. L'impact qui en découle sur les éléments de l'algorithme nous a conduit à proposer différentes variantes d'exécution.

Notre nouvelle approche est présentée de manière détaillée dans le chapitre suivant.

Chapitre 5

Vers une Approche Basée sur la Coopération de Niches de Requêtes

1. Introduction

Nous avons présenté dans le précédent chapitre les caractéristiques fondamentales de notre approche globale d'optimisation de requête.
L'approche proposée est basée sur l'exploitation des techniques d'algorithmique génétique d'une part, et de reformulation de requête par injection de pertinence d'autre part. Plus précisément, le processus de recherche d'information inhérent à cette approche est caractérisée par une exploration parallèle et guidée de l'espace documentaire. En effet, il est basé sur le déroulement d'un AG qui utilise le feedback utilisateur en vue de relativiser l'adaptation des requêtes dans le voisinage documentaire associé, et d'ajuster la structure des opérateurs en les spécifiant pour la résolution du problème d'optimisation de requête dans le SRI.
Les résultats expérimentaux réalisés, à l'aide du SRI Mercure sur les collections standards CACM, ADI et TREC6 montrent globalement l'intérêt de notre approche. L'analyse critique de ces résultats, nous ont amenés à dresser une nouvelle ébauche pour notre approche. Nous avons essentiellement focalisé notre intérêt sur l'exploitation de l'heuristique de nichage et calcul d'adaptation des requêtes.
Cette actualisation a eu pour conséquence, la révision du processus général de recherche d'information et définition de nouvelles fonctions appropriées pour la fusion des résultats de recherche.
Le présent chapitre décrit les principales caractéristiques de notre nouvelle approche, en mettant en exergue les différences existantes avec la précédente. Nous y décrivons également les différentes expérimentations que nous avons réalisées en utilisant la collection AP88, dans le but de valider notre approche.
L'évaluation porte particulièrement sur l'estimation des paramètres de l'AG, impact des opérateurs génétiques et heuristiques d'évolution sur les résultats de recherche.
Enfin, une évaluation comparative entre approche de base et approche améliorée est également présentée.

2. Description générale de l'approche

Le processus de recherche d'information que nous proposons demeure essentiellement basé sur le déroulement d'un AG d'optimisation de requête, qui coordonne les activités de l'utilisateur, d'un modèle de recherche de base et d'un AG adapté à la recherche d'information.
Cependant, comparativement à notre précédente approche, l'AG mis en œuvre à présent, est caractérisé par :

1. **l'exploitation de niches de requêtes formellement identifiées** relativement à l'évaluation des requêtes associées. Une telle caractérisation nous permet de cibler effectivement des sous-espaces de recherche différents, contenant des descripteurs de documents éventuellement pertinents pour une même requête et relativement dissemblants,

2. **la mise en œuvre d'une fonction d'ordre global des documents** basée sur l'adaptation relative des niches. Ceci nous permet alors de déterminer les directions de recherche à explorer et autres à exploiter en se basant sur une valeur de pertinence synthétique liée aux résultats de recherche menée par les différentes niches de la population,

3. **l'application restrictive d'opérateurs génétiques non aveugles,** augmentés par une connaissance liée aux techniques d'expansion et de de repondération de requêtes,

4. **l'intégration d'heuristiques d'évolution** en vue d'améliorer les conditions de convergence de l'AG en qualité et temps.

Nous présentons dans ce qui suit, le fonctionnement général du SRI et algorithme d'optimisation de requête qui en découlent.

2.1. Fonctionnement général du SRI

Le fonctionnement global du SRI, illustré sur la figure 5.1., est à présent fondé sur l'évolution génétique de niches de requêtes et non d'individus requêtes.
La niche constitue en effet, un groupe potentiel de requêtes qui investit une direction de recherche déterminée et évolue en accord avec les résultats d'évaluation de la recherche, traduit par deux facteurs :
- valeur d'adaptation relative des niches de requêtes,
- jugement de pertinence de l'utilisateur.

Le séquencement des opérations qui sont à la base du processus de recherche d'information est globalement analogue à celui de la précédente approche, à savoir :

1. Indexation de la requête utilisateur
2. Evaluation de la requête utilisateur
3. Organisation de la population en niches d'individus requêtes
4. Déroulement du processus de recherche de base
5. Fusion des résultats d'évaluation des niches d'individus requêtes
6. Renouvellement des niches d'individus requêtes
7. Feedback utilisateur

8. Calcul d'adaptation des niches de requêtes
9. Application des opérateurs génétiques

Cependant, de nombreuses révisions ont été apportées aux détails des différentes opérations, et seront présentées dans les paragraphes suivants.

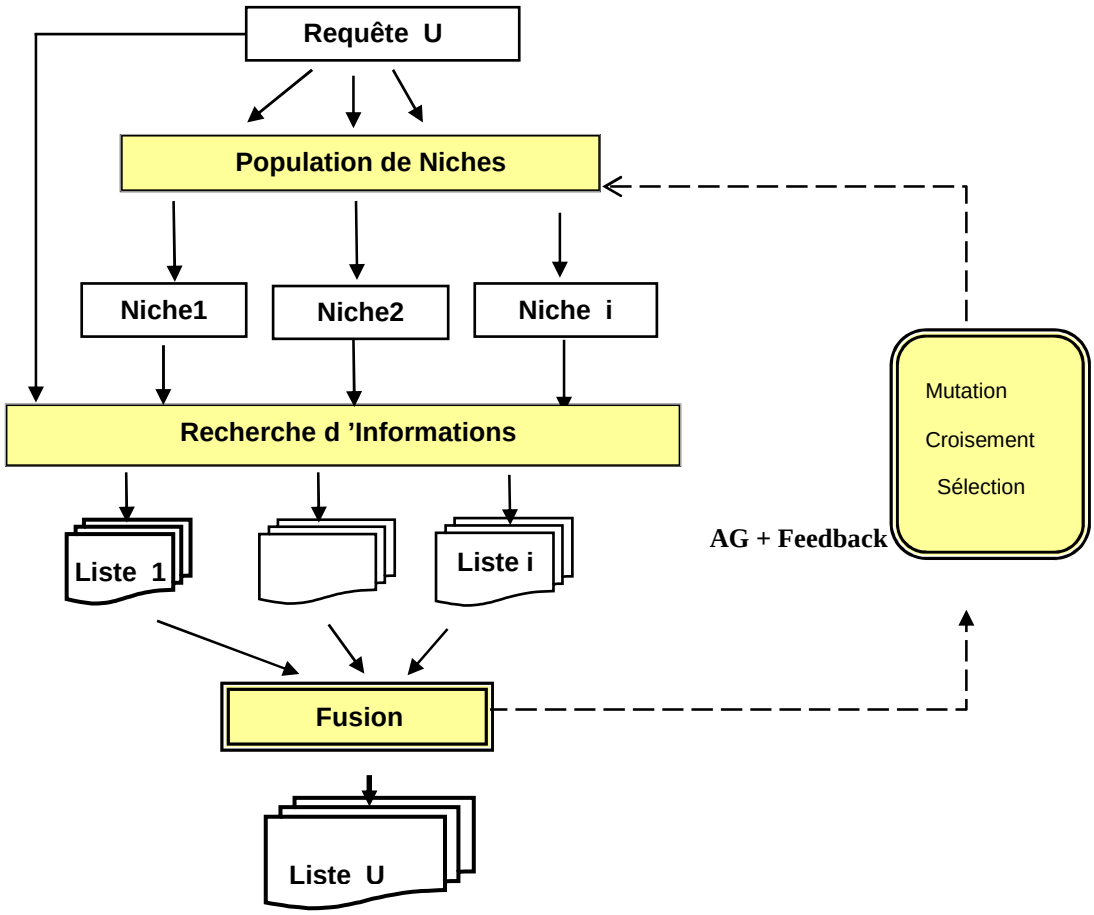


Figure 5.1 : Processus général du fonctionnement du SRI

2.2. Algorithme de base

L'AG d'optimisation de requête a la structure de base décrite sur la figure 5.2. L'algorithme ainsi décrit vise l'évolution régulée de niches de requêtes représentant des directions de recherche dans l'espace documentaire défini par la collection. La manipulation génétique de ces niches de requêtes est conditionnée par le jugement de pertinence de l'utilisateur d'une part, et résultats de la recherche, d'autre part.

3. Principaux éléments de l'AG d'optimisation de requête

Nous présentons dans cette section, les principaux éléments caractéristiques du nouvel AG d'optimisation de requête que nous proposons. Notons à cet effet, que nous détaillons d'avantage les éléments de différence entre la présente et précédente approche.

Début
Evaluer Requête-Utilisateur
t :=0
Construire la population initiale de niches de requêtes $Pop^{(0)}$
Répéter
Pour *chaque niche de requêtes* $N_i \in Pop^{(t)}$
Pour *chaque individu requête* $\in N_i$
Effectuer la recherche
Fait
Fait
Identifier les nouvelles niches N_new_i
Effectuer la fusion
Retenir le jugement utilisateur des NbJug top documents
Pour *chaque niche* N_new_i
Calculer l'adaptation des individus requêtes
Appliquer les opérateurs génétiques
Fait
t :=t+1
Jusqu'à arrêt
Fin

Figure 5.2. : Structure de base de l'AG d'optimisation de requête

3.1. Niche et population

La niche devient un concept qu'il convient de décrire dans le cadre de notre nouvelle approche.
Rappelons que la technique de nichage [Goldberg, 1994] est intégrée à un AG en vue de résoudre un problème d'optimisation multimodal (présence de différents optimums). Dans notre cadre de travail, nous persévérons dans la logique de multimodalité du problème d'optimisation de requêtes, de par la présence éventuelle de documents pertinents dans des sous espaces documentaires différents (caractérisées par des termes descriptifs différents), et que nous appelons ***« régions ».***

Ces régions sont, à notre sens, structurellement assez dissemblables, pour être atteintes par une « unique requête optimale ». A cet effet, l'AG favorisera la production de niches potentielles de requêtes associées aux différentes régions. C'est ainsi le principe de la ***recherche coopérative.***

En reprenant les concepts des AG's, une région sera définie par l'évaluation d'un schème qui décrit un hyperplan de l'espace documentaire de recherche. La recherche d'information globale est la résultante des recherches partielles et progressives effectuées par des générations de niches de requêtes dans des régions de l'espace documentaire.

La théorie des AG's assure alors la découverte de régions pertinentes correspondant aux briques élémentaires et ce, par une exploration parallèle de l'espace documentaire.

Nous décrivons dans les paragraphes suivant le principe de construction des niches de requêtes.

3.1.1. Identification d'une niche de requêtes

Une niche est composée d'un nombre déterminé d'individus requêtes codés selon le principe préalablement illustré dans la précédente approche (Cf chapitre 4, paragraphe 4.1) . La composante d'une niche évolue en taille (nombre d'individus requêtes) et structure (descripteurs des individus requêtes) sous l'effet des résultats de la recherche dans la base documentaire et des transformations génétiques opérées sur les individus requêtes. A cet effet, on définit la relation **Coniche** notée $\equiv_N$, comme suit :

$$(Q_u^{(s)} \equiv_N Q_v^{(s)}) \Leftrightarrow ((Ds(Q_u^{(s)} , L) \cap (Ds(Q_v^{(s)}, L) > Limite\_Coniche)$$

Où :

Ds(Q, L) : Ensemble des L premiers documents sélectionnés par la requête individu Q

Limite_Coniche : Nombre minimal de documents communs retrouvés par les individus requêtes d'une même niche.

Avec :

Limite_Coniche = NbJug*Prop_Coniche

Où :

Prop_Coniche: Constante réelle appartenant à l'intervalle [0 1]

NbJug : Nombre de documents jugés par l'utilisateur

En ce sens que deux requêtes appartiennent à une même niche, si et seulement si le cardinal de l'ensemble des premiers documents communs résultant de leur évaluation, est supérieur à une proportion fixée du nombre de documents jugés par l'utilisateur.

Le procédé d'identification des niches ainsi défini, présente l'avantage majeur d'être basé directement sur le résultat d'évaluation des individus requêtes et non sur une fonction caractéristique, qui impose la fixation d'un seuil en rapport avec le mécanisme d'appariement mis en œuvre dans le modèle de recherche de base.

On note que la relation $\equiv_N$ est réflexive, symétrique et non transitive.

Les niches sont identifiées après une phase d'interrogation de la base comme illustré sur la figure 5.3. D'après la définition de la relation $\equiv_N$, il résulte que les niches sont non forcément disjointes, de nombre et tailles variables en cours du processus génétique de recherche d'information.
Cependant, pour éviter une croissance incontrôlée de la taille population, sous l'effet des opérateurs génétiques, et qui engendrerait une diminution de la précision lors de la fusion des listes de documents issues des niches de requêtes, nous convenons de préserver une taille de population constante par construction de niches disjointes.

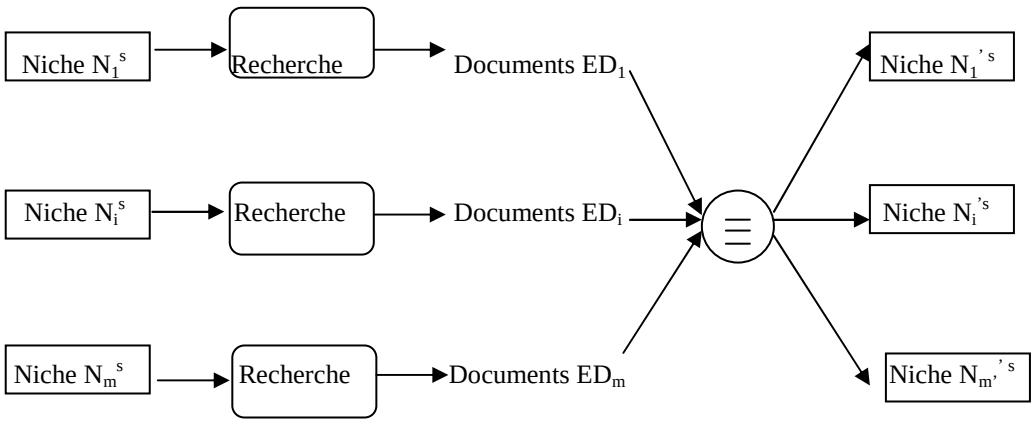


Figure 5.3 : Principe d'identification des niches

A cet effet, nous assumons de retenir les requêtes éventuellement communes entre niches, une seule fois dans la niche la moins peuplée, de manière à favoriser l'exploration.
Par ailleurs, nous considérons que les valeurs du paramètre *Prop_Coniche* dépend d'avantage des caractéristiques des collections interrogées que de la structure de la requête initiale. Il convient ainsi de déterminer sa valeur adéquate afin d'éviter de canaliser l'exploration dans une direction de recherche, dans le cas d'un nombre réduit de niches relativement à la taille de la population, ou à l'inverse, de disséminer l'exploration dans de nombreuses directions de recherche, dans le cas d'un nombre élevé de niches relativement à la taille de la population .
A cet effet, le paramètre *Prop_Coniche* étant borné, il nous paraît opportun de préconiser son apprentissage préalable sur les collections interrogées.

3.1.2. Population de niches

La population manipulée par l'AG, est à présent perçue comme un ensemble de niches de requêtes. La population est renouvelée à chaque génération sous l'effet de l'application des opérateurs génétiques et principe de construction des niches.
La population initiale est construite en deux étapes : évaluation de la requête initiale et construction de(s) niche (s) initiale (s).
L'évaluation de la requête initiale demeure fondée sur le jugement de pertinence de l'utilisateur sur les documents sélectionnés suite à l'évaluation de la requête initiale. La sélection des individus de la population initiale, notée $Pop^{(0)}$, est effectuée comme suit :
Soit :
$Ds^{(0)}$: Ensemble des documents sélectionnés à la recherche initiale
$Dr^{(0)}$: Ensemble des documents pertinents sélectionnés à la recherche initiale

$J(Dr^{(0)})$: Ensemble des documents , contenus dans l'ensemble complément $C_{D_r^{(0)}}^{D_s^{(0)}}$, les plus ressemblants à ceux contenus dans $D_r^{(0)}$, par application de la mesure de similitude de Jaccard

$N^{(0)}$: Niche constituée à la population initiale
Taille_Pop : Taille de la population
On construit alors:

$$N^{(0)} = (Dr^{(0)}, \beta * |Dr^{(0)}|) \bigcup (J(Dr^{(0)}, (Taille\_Pop + 1) - \beta * |Dr^{(0)}|)$$

Où :
(Dr, E): Ensemble des E premiers documents pertinents sélectionnés
β : Constante réelle qui détermine la proportion de documents pertinents à intégrer dans $N^{(0)}$
avec $0 \leq \beta \leq 1$

Ne se basant sur aucune hypothèse préalable quant à la ressemblance des documents pertinents , nous proposons d'intégrer tous les individus requêtes ainsi construits, dans une même niche, et la population initiale $Pop^{(0)}$ réduite à $N^{(0)}$.

Comparativement au principe adopté pour notre précédente approche, nous privilégions dans le cas présent, l'intégration de documents ressemblants aux documents pertinents plutôt que ceux qui sont en début de la liste de documents sélectionnés. Ceci est motivé par la conjonctions de deux facteurs :

1. La recherche est à présent basée sur le résultat d'évaluation d'une niche qui délimite une région de l'espace documentaire, contenant des descripteurs structurellement ressemblants.
2. La population initiale étant constituée d'une unique niche, nous pensons ainsi réduire les erreurs d'estimation de la pertinence supposée des directions de recherche initiales, en intégrant des descripteurs ressemblants aux descripteurs de documents effectivement jugés pertinents par l'utilisateur.

Dans le cas où aucun document pertinent n'est sélectionné lors de la recherche initiale, $N^{(0)}$ sera constitué des *Taille_Pop* premiers documents de la liste issue de la recherche initiale

L'objectif est ainsi de favoriser, dans la constitution de la population initiale, la présence significative de documents pertinents afin d'orienter d'emblée l'AG dans une direction de recherche prometteuse. Le principe de reproduction génétique déterminera, en cours de générations de niches de requêtes, l'équilibre entre directions à exploiter et autres à explorer.

En résumé, la génération de la population initiale n'est pas aléatoire. L'exploration de l'espace des documents est amorcée à partir d'une « région » que nous jugeons intéressante.

3.2. Fonction d'adaptation

Rappelons que dans le contexte précis du problème d'optimisation de requête, la fonction d'adaptation doit s'étalonner à la mesure des performances du système en taux de rappel/précision. Nous proposons une fonction objectif basée sur le modèle de coefficient de Guttman [Guttman, 1978] comme suit :

$$F\_Object(Q_u^{(s)}) = -\frac{\sum_{dr \in Dr,\, dnr \in Dnr} J(Q_u^{(s)}, dr^{(s)}) - J(Q_u^{(s)}, dnr^{(s)})}{\left| \sum_{dr \in Dr,\, dnr \in Dnr} J(Q_u^{(s)}, dr^{(s)}) - J(Q_u^{(s)}, dnr^{(s)}) \right|}$$

Où :
$Dr^{(s)}$: Ensemble des documents pertinents retrouvés à la génération s de l'AG
$Dnr^{(s)}$: Ensemble des documents non pertinents retrouvés à la génération s de l'AG

On a $F\_Object(Q_u^{(s)})$ définie dans l'intervalle *[-1 1]*. L'objectif est ainsi de minimiser $F\_Object(Q_u^{(s)})$. En transformant la fonction objectif en fonction d'adaptation positive sur l'intervalle de définition, on obtient la formule suivante [Tamine & Boughanem, 2000] :

$$Fitness(Q_u^{(s)}) = 1 + \frac{\sum_{dr \in Dr, dnr \in Dnr} J(Q_u^{(s)}, dr^{(s)}) - J(Q_u^{(s)}, dnr^{(s)})}{\left| \sum_{dr \in Dr, dnr \in Dnr} J(Q_u^{(Ss}, dr^{(s)}) - J(Q_u^{(s)}, dnr^{(s)}) \right|}$$

On note que la valeur d'adaptation des individus requêtes est relativisée aux résultats de la génération courante de manière à favoriser l'exploration en continuité, relativement aux directions définies par les niches de requêtes en cours d'évolution.
Les principaux avantages de ce type de fonctions est qu'il est basé sur le jugement de pertinence des utilisateurs et un modèle de fonction statistiquement corrélé aux mesures de taux de rappel/ précision [Bartell & al, 1998]. En effet, il paraît clairement que cette fonction considère l'ordre des documents sélectionnés, de manière analogue au principe de calcul des taux de rappel/précision.

Par ailleurs, l'ajustement de la fonction d'adaptation d'un individu requête par le cardinal de la niche associé , permet de favoriser le peuplement de niches isolées mais éventuellement intéressantes [Goldberg, 1989]. A cet effet, nous proposons la deuxième formulation suivante de la fonction d'adaptation :

$$Fitness_A(Q_u^{(S)}) = \frac{Fitness(Q_u^{(S)})}{\left| Niche(Q_u^{(s)}) \right|}$$

Où :
$Fitness_A(Q_u^{(s)})$: Adaptation ajustée de l'individu $Q_u^{(s)}$
$Niche(Q_u^{(s)})$: Niche associée à l'individu requête $Q_u^{(s)}$

3.3. Opérateurs génétiques

Nous poursuivons, dans notre nouvelle approche également, l'idée d'intégrer dans la structure des opérateurs, une connaissance liée aux techniques de reformulation de requête. Notons encore une fois, que l'utilisation d'une connaissance auxiliaire propre au problème de la recherche d'information permet d'accélérer l'exploration génétique par une recherche guidée dans l'espace documentaire.
L'application de ces opérateurs s'effectuera cependant**, de manière restreinte aux niches et non de manière uniforme sur toute la population**.

3.3.1. Sélection

Nous optons pour une sélection basée, non plus sur la méthode usuelle de la roue de loterie, mais sur le mode d'espérance mathématique [Goldberg, 1994]. La sélection est appliquée de manière restreinte à chaque niche selon les étapes suivantes :

Notations :
$N_i^{(s)}$: ième niche de la population à la génération *s*
$Pop^{(s)}$: Population de la génération *s*

1. Pour chaque individu requête de $N_i^{(s)}$, soit $Q_u^{(s)}$, calculer la valeur d'adaptation relative :

$$P_{select}(Q_u^{(s)}) = Fitness(Q_u^{(s)}) \Big/ Fitness_R(N_i^{(s)})$$

Où :
$Fitness_R(N_i^{(s)})$: Valeur d'adaptation relative de la niche $N_i^{(s)}$ calculée comme suit :

$$Fitness_R(N_i^{(s)}) = \frac{\sum_{Q_u^{(S)} \in N_i^{(S)}} Fitness(Q_u^{(s)})}{\sum_{N_j^{(s)} \in Pop^{(s)}} \sum_{Q_u^{(s)} \in N_j^{(s)}} Fitness(Q_u^{(s)})}$$

2. Calculer le nombre d'individus requêtes clones attendus

$$Nb\_Clones(Q_u^{(s)}) = P_{select}(Q_u^{(s)}) * \left| N_i^{(s)} \right|$$

3. Générer pour chaque individu requête $Q_u^{(s)}$, un nombre de clones égal à la partie entière de $Nb\_Clones\ (Q_u^{(s)})$

4. Trier les individus requêtes selon la partie décimale de $Nb\_Clones(Q_u^{(s)})$

5. Considérer les parties décimales de $Nb\_Clones(Q_u^{(s)})$ comme probabilités de sélection puis effectuer une sélection usuelle basée sur la roue de loterie.

Comparativement à la méthode de la roue de loterie, l'avantage de cette méthode est de réduire l'erreur stochastique qui induit un écart très variable entre nombre de clones attendus et nombre de clones effectivement produits pour chaque individu requête.

3.3.2. Croisement

L'opérateur de croisement est appliqué de manière restrictive à chaque niche dans le but produire une niche enfant de même taille.

En ce sens, on itère les opérations de sélection, croisement avec une probabilité *Pc* de deux individus requêtes candidats, puis remplacement des individus requêtes résultants dans la niche enfant, et ce, jusqu'à sa saturation.

A cet effet, nous retenons deux types de croisement qui traduisent des reformulations contextuelles de requêtes, relativement au voisinage de recherche défini par la niche associé : croisement basé sur le poids des termes et croisement aveugle. Le croisement basé sur la cooccurrence des termes dans la collection n'est pas retenu. En effet, son principe ne répond pas particulièrement au mode d'application restrictif aux niches. L'association de cooccurrence est inhérente à la structure des documents de la collection et indépendante des résultats d'évaluation de la niche en cours de croisement.

3.3.2.1.Croisement basé sur le poids des termes

C'est un croisement sans site qui produit, à partir de deux individus requêtes parents, un seul individu requête enfant , construit par ajustement des poids des termes d'indexation en accord avec leur distribution dans les documents pertinents et documents non pertinents.

C'est le type de croisement préalablement proposé dans notre précédente approche. Cependant nous y intégrons, outre la repondération des requêtes, la possibilité d'expansion par ajout, dans la requête enfant, de termes absents dans l'une ou l'autre des requêtes parents. Le principe de ce croisement est défini comme suit :

$Q_u^{(s)}(q_{u1}^{(s)}, q_{u2}^{(s)}, \ldots , q_{uT}^{(s)})$

$Q_v^{(s)}(q_{v1}^{(s)}, q_{v2}^{(s)}, \ldots , q_{vT}^{(s)})$

$\longrightarrow \quad Q_p^{(s+1)}(q_{p1}^{(s+1)}, q_{p2}^{(s+1)}, \ldots , q_{pT}^{(s+1)})$

Si $((q_{ui}^{(s)} \neq 0) \wedge (q_{vi}^{(s)} \neq 0))$ *Alors*

$q_{pi}^{(S+1)} =$ *Max* $(q_{ui}^{(s)}, q_{vi}^{(s)})$ *si Poids* $(t_i, D_r^{(s)}) \geq$ *Poids* $(t_i, D_{nr}^{(s)})$

Min $(q_{ui}^{(s)}, q_{vi}^{(s)})$ *sinon*

Sinon

Si $(q_{ui}^{(s)} = 0)$ *Alors*

$q_{pi}^{(s+1)} = q_{vi}^{(s)}$

Sinon

$q_{pi}^{(s+1)} = q_{ui}^{(s)}$

Fsi

Fsi

3.3.2.2. Croisement aveugle

C'est le type de croisement présenté préalablement dans le cadre de notre précédente approche.

3.3.3. Mutation

Nous retenons les opérateurs de mutation que nous avons préalablement défini dans le cadre de notre précédente approche.

3.4. Heuristiques d'évolution

Inspirés des heuristiques d'évolution proposés dans la littérature des AG's, nous préconisons d'intégrer à la population en cours, une niche composée de deux individus requêtes *artificiellement construits*. Ces individus sont construits sur la base :

- de la stratégie de sélection élitiste [Goldberg, 1994] qui consiste pour nous, à conserver dans la prochaine génération de requêtes, l'individu requête ayant obtenu la meilleure valeur d'adaptation dans la génération courante,

- de la structure d'une requête *« virtuelle »* constituée par les termes de meilleure valeur issue de la formule suivante :

$$Score(t_i)=\frac{\sum_{Dj\in Dr(s)} d_{ji}}{\left|Dr_r^{(s)}\right|}$$

  Les termes les mieux pondérés dans les documents pertinents, retrouvés à la génération courante de l'AG, sont ainsi structurés en individu requête à intégrer dans la prochaine génération.

Le but d'intégrer ces heuristiques est d'améliorer la convergence de l'AG en qualité et temps. La qualité de convergence traduit la valeur de précision/rappel obtenue à chaque génération alors que le temps traduit le nombre de générations nécessaires pour atteindre les valeurs de rappel/précision optimales.

4. Principe de fusion des résultats de recherche

La population d'individus requêtes est organisée en niches. A l'issue de l'évaluation de chaque niche, nous obtenons des ensembles non disjoints de documents restitués par le processus de recherche de base. Ces listes partielles sont fusionnées de manière à constituer une liste unique de documents, soumise au jugement de pertinence de l'utilisateur.

Le principe adopté pour la fusion doit être considéré avec prudence, puisqu'il a un impact direct sur la précision de la recherche. En effet, rappelons que les performances de recherche sont mesurées en valeur de rappel des documents pertinents mais également en termes de qualité de l'ordre des documents pertinents restitués.
Dans ce cadre, nous proposons deux types de fusion : fusion basée sur l'ordre local des niches et fusion basée sur l'ordre global de la population.

4.1. Fusion basée sur l'ordre local des niches

Dans ce cas, la fusion des documents restitués par les individus requêtes est effectuée en deux étapes.
La première étape détermine un premier ordre relatif à la supposition de pertinence, basée sur les résultats de sélection , au niveau de chaque niche.
La seconde étape, réordonne alors ces listes partielles sur la base de la mesure d'adaptation des individus requêtes et/ou de la valeur de pertinence des documents.
Nous proposons deux fonctions de fusion basées sur ce principe : fusion totale et fusion élitiste.

1. Fusion totale

Le principe de fusion totale consiste à considérer l'ordre restitué par toutes les requêtes des différentes niches. Les documents sont ordonnés localement à la niche sur la base d'une valeur de pertinence calculée comme suit :

$$\mathrm{Rel}(N_i^{(s)},D_j)=\frac{1}{\left|Ni^{(s)}\right|}\sum_{Q_u^{(s)}\in N_i^{(s)}} RSV(Q_u^{(s)},D_j)$$

Où :
$\mathrm{Rel}(N_i^{(s)},D_j)$: Valeur de pertinence locale du document D_j dans la niche $N_i^{(S)}$
$\mathrm{RSV}(Q_u^{(s)},D_j)$: Valeur de pertinence du document D_j en présentant l'individu requête $Q_u^{(s)}$ au processus de recherche de base

Ensuite, la liste présentée à l'utilisateur est obtenue par fusion des listes ordonnées et partielles obtenues précédemment . Cette fusion respecte l'ordre donné par la formule :

$$\mathrm{Rel}(D_j)=\sum_{Nj^{(s)}\in Pop^{(s)}} Fitness\_Moyen(N_j^{(s)})*\mathrm{Rel}(N_j^{(s)},D_j)$$

Où :
$\mathrm{Rel}(D_j)$: Valeur de pertinence globale du document D_j
$\mathrm{Fitness\_Moyen}(N_j^{(s)})$: Valeur d'adaptation moyenne de la niche $N_j^{(s)}$ avec :

$$Fitness\_Moyen(N_j^{(s)})=\frac{1}{\left|N_j^{(s)}\right|}\sum_{Q_u^{(s)}\in N_j^{(S)}} Fitness(Q_u^{(s)})$$

En somme l'ordre global des documents est obtenu moyennant une fonction linéaire de l'ordre local aux différentes niches, pondérée par la valeur d'adaptation moyenne associée. Ainsi l'objectif est de relativiser l'ordre de documents donné par une niche, en fonction de son adaptation relative dans la population.

2. Fusion élitiste

Le principe de fusion élitiste consiste, en revanche, à ne considérer que l'ordre local restitué par l'individu requête le plus adapté de chaque niche. L'ordre local étant basé sur la valeur de pertinence retournée par le processus de recherche de base, l'ordre global est alors obtenu en utilisant la formule :

$$\mathrm{Rel}(D_j)= \sum_{Nj \in Pop^{(S)} \wedge Q_u^* \in Nj} Poids(N_j^{(s)}) * RSV(Q_u^*, D_j)$$

où :

Q_u^* : Individu requête de meilleure valeur d'adaptation

$Poids(N_j^{(S)})$: Poids de la niche $N_j^{(S)}$ calculé comme suit :

$$Poids(N_j^{(S)}) = \frac{Fitness(Q_u^*)}{Fitness\_Moyen(N_j^{(s)})}$$

L'objectif dans ce cas est d'augmenter éventuellement la précision globale en accordant un degré de confiance important à l'ordre local donné par la meilleures requête de chaque niche.

4.2. Fusion basée sur l'ordre global de la population

Dans ce cas, la fusion de documents restitués par les individus requêtes est effectué en une seule étape.

L'idée est de réduire l'erreur commise sur l'ordre global en évitant de se baser sur une supposition peu justifiée de l'ordre local aux niches.

On propose alors une **fusion sélective** qui consiste à fusionner linéairement les documents restitués par les requêtes dont les valeurs d'adaptation sont supérieures à la moyenne d'adaptation dans la population.

L'ordre des documents restitués à l'utilisateur est obtenu en calculant :

$$\mathrm{Rel}(D_j)= \sum_{N_j^{(s)} \in Pop^{(s)}} \sum_{Q_u^{(s)} \in N_j^{(s)}} Fitness(Q_u^{(s)**} * RSV(Q_u^{(s)}, D_j)$$

Où :

$Q_u^{(S)**}$: Requêtes dont la valeur d'adaptation est supérieure à l'adaptation moyenne de la population

On note ainsi que c'est la valeur d'adaptation directe de chaque individu requête qui contribue à déterminer la qualité de l'ordre des documents restitués.

5. Evaluation globale de l'approche

Nous décrivons dans cette section les différentes expérimentations que nous avons réalisées dans le but de valider notre approche. L'évaluation porte particulièrement sur les paramètres de base de l'AG, impact de l'ajustement de la fonction d'adaptation, opérateurs génétiques appliqués et heuristiques d'évolution proposées, sur les résultats de recherche.
En considérant l'aspect amélioré de cette approche d'une part, et nombreux éléments intégrés d'autre part, nous convenons d'approfondir son évaluation et ce, en introduisant de nouvelles mesures d'évaluation relativement à une collection de test caractérisée par des documents plus longs que la collection précédente, qui est en l'occurrence la collection AP88 décrite sur le tableau 5.1.
Les conditions expérimentales sont globalement analogues à celles définies pour l'évaluation de la précédente approche. Nous décrivons dans ce qui suit les mesures intégrées.

5.1. Conditions experimentales

Nous avons procédé à un ensemble d'expérimentations en utilisant le SRI Mercure. L'objectif de ces expérimentations est d'évaluer :

1. L'apport de notre approche d'optimisation de requête pour l'amélioration des performances du SRI. Cette évaluation est effectuée en utilisant des mesures comparatives des taux de précision moyenne, précision à *15* documents, nombre de documents pertinents et nombre de documents prtinents cumulé restitués à chaque itération et ce, relativement à une recherche de base menée dans le SRI Mercure.

2. L'impact des éléments de l'AG sur les résultats de recherche : taille de la population, seuil de conichage , méthode de fusion des résultats de recherche, ajustement de la fonction d'adaptation, types d'opérateurs génétiques appliqués, technique de nichage et heuristiques d'évolution.

Nous présentons dans ce qui suit les conditions expérimentales qui caractérisent le schéma d'évaluation mis en œuvre.

5.1.1. Paramètres de l'AG

Nous avons caractérisé deux catégories de paramètres de contrôle de l'algorithme :

1. ***Première catégorie***

   Cette catégorie comprend les paramètres probabilité de croisement (*Pc*), probabilité de mutation (*Pm*), taille de la population de requêtes (*Taille_Pop),* proportion de conichage (*Prop_Coniche),* base d'application de l'opérateur de conichage (*L*) et nombre de termes d'expansion (*taille de la liste Lmut*).

   En ce qui concerne les paramètres *Pm*, *Pc* et *taille de la liste* L_{mut} , nous retenons le meilleur jeu de valeurs obtenues lors d'expérimentations préalables effectuées sur la collection TREC6 [Boughanem & all, 1999] ; ce sont respectivement les valeurs *0.7*, *0.07* et *30*. Cette dernière valeur correspond à une valeur standard recommandée pour l'expansion de requêtes [Harman, 1992].

   En ce qui concerne le paramètre *L* et suite à de nombreuses expérimentations préliminaires, nous avons retenu la valeur *50* qui correspond à une fraction de moitié du nombre maximal *(100)* de documents sélectionnés par chaque individu requête.

   Les paramètres *Taille_Pop* et *Prop_Coniche* ont fait l'objet d'expérimentations développées dans la suite.

2. ***Deuxième catégorie***

   Cette catégorie comprend les paramètres : type de méthode de fusion des résultats de recherche, type de fonction d'adaptation et types d'opérateurs génétiques appliqués. Ces paramètres ont fait l'objet de nombreuses expérimentations que nous développerons dans la suite.

5.1.2. Jeu d'opérateurs génétiques

Le croisement basé sur la cooccurrence n'étant pas retenu dans le cadre de cette approche, nous avons alors expérimenté l'impact des groupes d'opérateurs *G1* et *G3* définis dans le tableau 4.1. (Cf au chapitre 4), sur les résultats de la recherche.

5.1.3. Méthode d'évaluation

Nous retenons la méthode d'évaluation adoptée dans le cadre de la précédente approche. Cependant, nous considérons deux bases de référence d'évaluation :

- *Baseline* : Utilisée précédemment ; rappelons que cette base considère comme référence, à chaque génération *s* de l'AG, l'apport de nouveaux documents relativement à la génération *s-1*.

- *Baseline_Init* : considère comme référence à chaque itération, les *15* documents suivants de la liste ordonnée restituée à la recherche initiale. Ce type d'évaluation mesure ainsi la capacité de l'AG, à améliorer l'ordre des documents pertinents relativement à la liste issue de la recherche initiale.

Par ailleurs, on calcule pour chaque requête, en plus du nombre de documents pertinents et nombre de documents pertinents cumulé à chaque itération, la précision moyenne et précision à 15 documents.

5.1.4. Collection de test

Nous avons mené nos expérimentations en utilisant la collection de test *AP88* (Associated Press newswire, 1988) décrite sur le tableau 5.1.
Les requêtes sont issues des topics numérotées 1-24 de la collection TREC ; nous avons utilisé les champs titre, description et narrative.

AP 88		
Nombre de documents dans la collection	***Nombre de termes dans la collection***	***Taille moyenne d'un document***
79919	144186	235.085

Tableau 5.1 : Description de la collection de test

5.1.5. L'algorithme de recherche

L'expérimentation est basée sur l'exécution de l'algorithme de recherche suivant :

1. Iter :=*0*
2. Présenter puis évaluer la requête initiale
3. Juger les *15* premiers documents
4. Construire la population initiale de niches de requêtes
5. Pour chaque niche
6. Effectuer la recherche pour chaque individu requête
7. Fait
8. Fusionner les résultats
9. Construire les nouvelles niches

10. Juger les *15* premiers documents
11. Pour chaque niche
12. Calculer l'adaptation pour chaque individu requête
13. Appliquer les opérateurs génétiques
14. Fait
15. Iter :=Iter+1
16. Si (Iter<5) Alors aller à 4

5.2. Evaluation de la taille de population et seuil de conichage

Notre approche d'optimisation de requête s'articule sur une recherche coopérative menée par une population de niches de requêtes. L'AG explore l'espace documentaire défini par la collection interrogée, dans des directions définies par les niches de requêtes.

Dans le cadre précis de notre approche, le seuil de conichage est un facteur déterminant quant à l'organisation de la population en niches. Comme les résultats de recherche sont obtenus par fusion des résultats d'évaluation de chaque niche, nous estimons qu'il convient d'expérimenter en premier lieu, les effet combinés de la taille de population, proportion de conichage (détermine le seuil conichage) et méthode de fusion, sur les résultats de la recherche. Nous présentons deux types d'évaluation, l'une basée sur le nombre de documents pertinents et nombre de documents pertinents total obtenus à chaque itération, et l'autre basée sur les valeurs de précision moyenne et précision à *15* documents obtenues à chaque itération.

5.2.1. Evaluation basée sur le nombre de documents pertinents

Les tableaux 5.2.a, 5.2.b et 5.2.c présentent le nombre de documents pertinents et nombre de documents pertinents cumulé obtenus, à chaque itération, en expérimentant la variation de la taille de population et proportion de conichage pour, respectivement la méthode de fusion totale, fusion élitiste et fusion sélective.

La taille de population varie dans l'ensemble *{2, 4, 6}* et la proportion de conichage varie dans l'ensemble *{0.2, 0.6, 1}*.

La ligne *Ecart min_max* exprime l'écart obtenu entre nombre minimal et nombre maximal de documents pertinents obtenus à chaque itération pour les différentes valeurs de conichage. La ligne *Ecart global* exprime l'écart obtenu entre nombre minimal et nombre maximal de documents pertinents obtenus à chaque itération pour les différentes valeurs de conichage et différentes valeurs de la taille de population.

Nous constatons que le seuil de conichage a un effet intrinsèque sur les résultats de recherche (autres paramètres constants) et effet combiné à celui de la taille de population et méthode de fusion des résultats de recherche.

Les écarts en nombre minimal et nombre maximal de documents pertinents obtenus à chaque itération, sont très variables en fonction de la taille de population et type de méthode de fusion.
Notons, cependant, que pour une même méthode de fusion , les écarts entre résultats obtenus sont plus importants relativement à la taille de population qu'à la proportion de conichage, ce qui nous permet d'estimer les effets relatifs de ces deux paramètres.
La comparaison du nombre de documents pertinents cumulé obtenu à la cinquième itération, nous permet de déterminer les meilleures paires de valeurs des paramètres *Taille_Pop* et *Prop_Coniche*, associées à chaque type de fusion.

Prop_Coniche	**Iter1**	**Iter2**	**Iter3**	**Iter4**	**Iter5**
		Taille_Pop=2			
0.2	172(172)	56(228)	53(281)	76(357)	75(432)
0.6	172(172)	56(228)	57(285)	76(361)	68(429)
1	172(172)	56(228)	68(296)	78(374)	72(446)
Ecart min_max	0(0)	0(0)	15(15)	21(17	7(17)
		Taille_Pop=4			
0.2	180(180)	66(246)	76(322)	93(415)	57(472)
0.6	180(180)	65(245)	86(331)	76(407)	68(475)
1	180(180)	87(267)	97(364)	81(445)	**55(500)**
Ecart min_max	0(0)	22(22)	21(42)	27(38)	13(28)
		Taille_Pop=6			
0.2	177(177)	59(236)	74(310)	71(381)	70(452)
0.6	177(177)	59(236)	73(309)	61(370)	73(443)
1	177(177)	59(236)	65(301)	55(356)	68(424)
Ecart min_max	0(0)	0(0)	9(9)	16(25)	5(28)
Ecart global	8(8)	31(39)	44(83)	38(89)	20(71)

Tableau 5.2.a : Effet de la taille de population et proportion de conichage

Cas d'une fusion totale

Prop_Coniche	**Iter1**	**Iter2**	**Iter3**	**Iter4**	**Iter5**
		Taille_Pop=2			
0.2	172(172)	123(295)	115(410)	78(488)	70(558)
0.6	172(172)	123(295)	117(412)	73(485)	69(554)
1	172(172)	123(295)	115(410)	87(497)	**70(567)**
Ecart min_max	0(0)	0(0)	2(2)	14(12)	1(13)
		Taille_Pop=4			
0.2	180(180)	87(267)	100(367)	81(448)	62(510)
0.6	180(180)	87(267)	86(353)	64(417)	79(497)
1	180(180)	87(267)	97(364)	81(445)	55(500)
Ecart min_max	0(0)	0(0)	14(14)	17(28)	24(13)
		Taille_Pop=6			
0.2	177(177)	102(279)	92(371)	73(444)	56(500)
0.6	177(177)	102(279)	73(352)	74(426)	44(470)
1	177(177)	102(279)	75(354)	70(424)	68(492)
Ecart min_max	0(0)	0(0)	19(19)	4(20)	44(30)
Ecart global	8(8)	36(28)	42(58)	23(73)	26(75)

Tableau 5.2.b : Effet de la taille de population et proportion de conichage

Cas d'une fusion élitiste

Prop_Coniche	**Iter1**	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Taille_Pop=2					
0.2	172(172)	113(285)	87(372)	80(452)	**70(522)**
0.6	172(172)	113(285)	87(372)	75(447)	71(518)
1	172(172)	113(285)	89(374)	69(443)	69(513)
Ecart min_max	0(0)	0(0)	1(2)	11(9)	2(9)
Taille_Pop=4					
0.2	180(180)	88(268)	93(361)	87(448)	61(509)
0.6	180(180)	88(268)	98(366)	75(442)	78(520)
1	180(180)	88(268)	97(365)	75(440)	57(497)
Ecart min_max	0(0)	0(0)	4(4)	12(8)	21(33)
Taille_Pop=6					
0.2	177(177)	105(282)	80(362)	61(423)	68(491)
0.6	177(177)	105(282)	78(360)	64(424)	56(480)
1	177(177)	105(282)	60(342)	68(410)	50(460)
Ecart min_max	0(0)	0(0)	20(20)	9(14)	18(31)
Ecart global	8(8)	25(13)	33(30)	26(42)	20(62)

Tableau 5.2.c : Effet de la taille de population et proportion de conichage

Cas d'une fusion sélective

Plus précisément, on retient les paires de valeurs *(4,1), (2,1) et (2,0.2)* pour, respectivement, la méthode de fusion totale, fusion élitiste et fusion sélective.
Par ailleurs, l'estimation du nombre de niches de requêtes construites à chaque génération, pour les différentes valeurs de la taille de population et proportion de conichage, nous a permis de constater une dépendance linéaire entre ces paramètres. Plus précisément, le nombre de niches varie dans l'intervalle *[1 Taille_Pop]* avec un écart plus important pour des valeurs relativement élevées de la proportion de conichage. Ceci confirme bien deux faits :

1. L'augmentation de la taille de population augmente le risque de construction de requêtes dissemblables sous l'effet combiné du croisement et de la mutation et par conséquent, une croissance du nombre de niches.
2. L'augmentation de la proportion de conichage conduit à l'imposition d'une condition plus stricte quant au « rapprochement » des résultats d'évaluation des individus requêtes devant appartenir à la même niche. Ceci a pour conséquence, l'apparition de nouvelles niches qui explorent de nouveaux voisinages documentaires.

L'analyse de cette série d'expérimentations est complétée par l'examen des valeurs de précision obtenues pour les différentes conditions d'exécution de l'AG. Ceci est présenté dans le paragraphe suivant.

5.2.2. Evaluation basée sur la précision

Les tableaux 5.3.a, 5.3.b et 5.3.c présentent la précision moyenne et précision à *15* documents obtenues pour l'ensemble des requêtes en appliquant les différentes valeurs de la taille de population et proportion de conichage pour, respectivement, les méthodes de fusion totale, élitiste et sélective.
En premier lieu, nous constatons globalement que la variation de la taille de la population et proportion de conichage ont un effet considérable sur les mesures de précision, ce qui corrobore les résultats obtenus avec les expérimentations précédentes avec, cependant, un impact moins significatif de la proportion de conichage sur les résultats obtenus pour une même taille de population.
Notons, à l'aide des résultats mis en gras, et en privilégiant les précisions obtenues aux premières itérations, que la meilleure valeur de taille de population varie entre *2* et *4*.
En faisant un compromis entre les résultats obtenus en utilisant la mesure du nombre de documents pertinents et ceux obtenus en utilisant la mesure de précision, nous convenons de retenir pour les expérimentations ultérieures une taille de population de *4* avec la valeur médiane de *0.6* pour la proportion de conichage.

Prop_Coniche	**iter1**		**iter2**		**iter3**		**iter4**		**iter5**	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
					Taille_Pop=2					
0.2	0.20	0.47	0.06	0.15	0.05	0.14	0.07	0.20	0.06	0.20
0.6	0.20	0.47	0.06	0.15	0.06	0.15	0.08	0.21	0.05	0.18
1	0.20	0.47	0.06	0.15	0.06	0.18	0.07	0.21	0.05	0.20
					Taille_Pop=4					
0.2	**0.21**	**0.51**	**0.04**	**0.18**	0.05	0.21	0.06	0.25	0.03	0.15
0.6	**0.21**	**0.51**	**0.04**	**0.18**	0.07	0.23	0.05	0.20	0.3	0.19
1	**0.21**	**0.51**	**0.04**	**0.18**	0.06	0.20	0.06	0.25	0.04	0.14
					Taille_Pop=6					
0.2	0.22	0.49	0.05	0.16	0.06	0.20	0.04	0.19	0.05	0.19
0.6	0.22	0.49	0.05	0.16	0.06	0.20	0.04	0.16	0.05	0.20
1	0.22	0.49	0.05	0.16	0.05	0.18	0.04	0.15	0.05	0.18

Tableau 5.3.a : Variation de la précision en fonction de la taille de population et proportion de conichage

Cas d'une fusion totale

Prop_Coniche	**Iter1**		**Iter2**		**Iter3**		**Iter4**		**Iter5**	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
					Taille_Pop=2					
0.2	0.20	0.47	**0.11**	**0.34**	**0.09**	**0.31**	0.05	0.21	0.03	0.19
0.6	0.20	0.47	**0.11**	**0.34**	**0.09**	**0.32**	0.04	0.20	0.03	0.19
1	0.20	0.47	**0.11**	**0.34**	**0.09**	**0.31**	0.05	0.24	0.03	0.19
					Taille_Pop=4					
0.2	**0.21**	**0.51**	0.07	0.24	0.08	0.27	0.04	0.20	0.04	0.18
0.6	**0.21**	**0.51**	0.07	0.24	0.07	0.23	0.04	0.17	0.05	0.20
1	**0.21**	**0.51**	0.07	0.24	0.07	0.27	0.05	0.20	0.04	0.15
					Taille_Pop=6					
0.2	0.22	0.49	0.09	0.28	0.06	0.25	0.05	0.20	0.03	0.15
0.6	0.22	0.49	0.09	0.28	0.05	0.20	0.05	0.20	0.03	0.12
1	0.22	0.49	0.09	0.28	0.05	0.21	0.04	0.19	0.03	0.18

Tableau 5.3.b : Variation de la précision en fonction de la taille de population et proportion de conichage

Cas d'une fusion élitiste

Prop_Coniche	Iter1		Iter2		Iter3		Iter4		Iter5	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
					Taille_Pop=2					
0.2	0.20	0.47	**0.10**	**0.31**	0.07	0.24	0.05	0.22	0.03	0.19
0.6	0.20	0.47	**0.10**	**0.31**	0.07	0.24	0.05	0.20	0.03	0.19
1	0.20	0.47	**0.10**	**0.31**	0.07	0.24	0.05	0.19	0.04	0.19
					Taille_Pop=4					
0.2	**0.21**	**0.51**	0.07	0.24	**0.06**	**0.25**	0.06	0.24	0.03	0.16
0.6	**0.21**	**0.51**	0.07	0.24	**0.06**	**0.27**	0.06	0.21	0.04	0.21
1	**0.21**	**0.51**	0.07	0.24	**0.06**	**0.26**	0.05	0.20	0.03	0.15
					Taille_Pop=6					
0.2	0.22	0.49	0.09	0.29	0.05	0.22	0.04	0.16	0.03	0.18
0.6	0.22	0.49	0.9	0.29	0.05	0.21	0.04	0.17	0.03	0.15
1	0.22	0.49	0.09	0.29	0.04	0.16	0.04	0.18	0.03	0.13

Tableau 5.3.c : Variation de la précision en fonction de la taille de population et proportion de conichage

Cas d'une fusion sélective

5.3. Impact de l'optimisation génétique de requête

Nous nous intéressons à présent, à l'évaluation de l'impact du processus génétique d'optimisation de requête sur les résultats de la recherche. A cet effet, nous avons utilisé le groupe d'opérateurs G1 et appliqué les différentes méthodes de fusion puis présentons dans ce qui suit, les résultats obtenus en considérant l'apport relativement aux deux listes de référence préalablement citées : *Baseline* qui considère les résultats de l'itération précédente et *Baseline_Init* qui considère les résultats de la recherche initiale.

L'évaluation est basée sur l'estimation du nombre de documents pertinents et nombre de documents pertinents cumulés à chaque itération, ainsi que sur les mesures de précision, comme présentés dans les paragraphes suivants.

5.3.1. Evaluation basée sur le nombre de documents pertinents

Les tableaux 5.4.a, 5.4.b et 5.4.c servent de comparaison entre valeurs du nombre de documents pertinents et nombre de documents pertinents cumulé, obtenus à chaque itération en utilisant les différentes conditions d'exécution et ce, respectivement, pour la méthode de fusion totale, fusion élitiste et fusion sélective.

La ligne *Avec_AG* exprime les résultats obtenus en utilisant notre approche. Les lignes *Sans_AG_Baseline* et *Sans_AG_Baseline_Init* expriment les résultas obtenus en interrogeant le SRI avec la requête initiale et en considérant les listes de référence respectives *Baseline* et *Baseline_Init.*

La ligne *Accroissement/Doc_Pert_Cum* exprime le taux d'accroissement enregistré en nombre de documents pertinents cumulé, à chaque itération, entre une exécution du processus de recherche *Avec_AG* et autre exécution *Sans_AG.*

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Avec_AG	180(180)	65(245)	86(331)	74(406)	69(475)
Sans_AG_Baseline	110(110)	114(225)	47(272)	69(342)	65(407)
Accroissement/Doc_Pert_Cum	63%	8%	22%	19%	16%
Sans_AG_Baseline_Init	110(110)	92(203)	82(285)	65(351)	61(412)
Accroissement/Doc_Pert_Cum	63%	20%	16%	15%	15%

Tableau 5.4.a : Comparaison entre nombres de documents pertinents retrouvés
Cas d'une fusion totale

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Avec_AG	180(180)	87(267)	86(353)	64(417)	79(497)
Sans_AG_Baseline	110(110)	114(225)	82(307)	79(376)	59(435)
Accroissement/Doc_Pert_Cum	63%	18%	14%	10%	14%
Sans_AG_Baseline_Init	110(110)	92(203)	82(285)	65(351)	61(412)
Accroissement/Doc_Pert_Cum	63%	31%	23%	18%	20%

Tableau 5.4.b : Comparaison entre nombres de documents pertinents retrouvés
Cas d'une fusion élitiste

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Avec_AG	180(180)	88(268)	97(366)	75(442)	78(520)
Sans_AG_Baseline	110(110)	114(225)	77(302)	69(371)	65(437)
Accroissement/Doc_Pert_Cum	63%	18%	21%	22%	22%
Sans_AG_Baseline_Init	110(110)	92(203)	82(285)	65(351)	61(412)
Accroissement/Doc_Pert_Cum	63%	32%	28%	25%	26%

Tableau 5.4.c : Comparaison entre nombres de documents pertinents retrouvés
Cas d'une fusion sélective

Les résultats montrent que la recherche par optimisation génétique de requête assure un accroissement des performances qui varie de *8%* à *63%* et ce, à chaque itération et pour toute méthode de fusion des résultats de recherche et relativement aux deux listes de référence d'évaluation.

Notons cependant, un pic de performances à la première itération puis diminution de l'accroissement des performances au niveau de la deuxième itération puis reprise aux itérations suivantes, comme illustré sur la figure 5.4. Ceci peut être justifié par la conjonction de deux causes. La première, concerne le principe adopté pour la construction de la population initiale. Cette dernière est en effet, constituée d'une niche composée d'individus requêtes sélectionnés dans le voisinage des documents jugés pertinents à la recherche initiale, ce qui nous permet d'atteindre immédiatement des nouveaux documents pertinents ressemblants.

La deuxième cause est liée à la première. En effet, en raison du rappel d'un nombre important de documents pertinents à la première itération, les itérations suivantes

révéleront des performances moindres, eu égard au nombre de documents pertinents total limité d'une part, et restriction de l'espace documentaire exploré relativement aux niches en cours d'évolution, d'autre part.

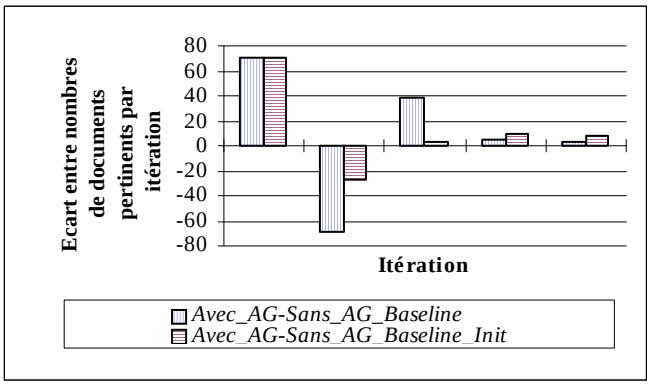


Figure 5.4.a : Comparaison entre nombres de documents pertinents retrouvés par itération
Cas fusion totale

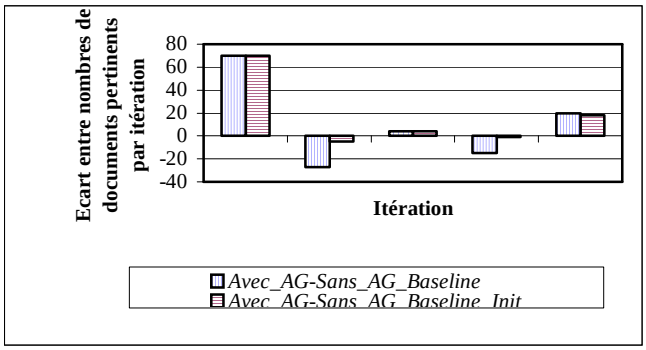


Figure 5.4.b : Comparaison entre nombres de documents pertinents retrouvés par itération
Cas fusion élitiste

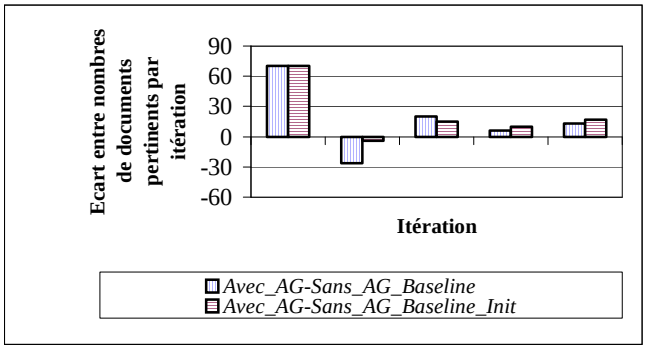


Figure 5.4.c : Comparaison entre nombres de documents pertinents retrouvés par itération
Cas fusion sélective

Par ailleurs, on constate que, pour toute méthode de fusion considérée, les accroissements de performances enregistrés relativement à la *Base_line_init* (*15% à 32%* à partir de la 2ème itération), ont tendance à être supérieurs à ceux enregistrés relativement à la *Baseline* (*8%* à *22%* à partir de la 2ème itération). Ceci traduit le fait que la recherche *Avec_AG* apporte, à chaque itération, mieux relativement aux listes successives à *15* documents issues de la liste initiale, que relativement à la liste obtenue à l'itération précédente.

Ces résultats sont prévisibles et confortent d'avantage notre approche. En effet, nous montrons ainsi que chaque génération de niches de requêtes apporte de nouveaux documents pertinents et que cet apport est évidemment moins important que si aucun traitement n'est effectué sur la liste des documents restituée à la recherche initiale (parcours simple en bas de liste), ce qui est le cas de la base de référence *Baseline_Init.*

5.3.2. Evaluation basée sur la précision

Nous complétons l'analyse précédente par l'examen de la précision enregistrée aux différentes itérations pour les différentes conditions . Les tableaux 5.5.a, 5.5.b et 5.5.c présentent les valeurs de précision moyenne et précision à *15* documents obtenues en appliquant respectivement, la méthode de fusion totale, fusion élitiste et fusion sélective.

	Iter1		**Iter2**		**Iter3**		**Iter4**		**Iter5**	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
Avec_AG	0.21	0.5	0.04	0.18	0.07	0.20	0.05	0.20	0.03	0.19
Sans_AG_Baseline	0.12	0.30	0.10	0.31	0.03	0.13	0.04	0.19	0.04	0.18
Sans_AG_Baseline_Init	0.12	0.30	0.07	0.25	0.05	0.22	0.03	0.18	0.02	0.17

Tableau 5.5.a : Comparaison de la précision moyenne et précision à 15 documents

Cas d'une fusion totale

	Iter1		**Iter2**		**Iter3**		**Iter4**		**Iter5**	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
Avec_AG	0.21	0.5	0.07	0.24	0.07	0.23	0.04	0.15	0.05	0.20
Sans_AG_Baseline	0.12	0.30	0.10	0.31	0.05	0.22	0.04	0.19	0.03	0.16
Sans_AG_Baseline_Init	0.12	0.30	0.07	0.25	0.07	0.22	0.03	0.18	0.02	0.17

Tableau 5.5.b : Comparaison de la précision moyenne et précision à *15* documents

Cas d'une fusion élitiste

	Iter1		**Iter2**		**Iter3**		**Iter4**		**Iter5**	
	Pmoy	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***	***Pmoy***	***P15***
Avec_AG	0.21	0.5	0.10	0.31	0.07	0.24	0.05	0.20	0.03	0.19
Sans_AG_Baseline	0.12	0.30	0.10	0.31	0.05	0.21	0.04	0.19	0.04	0.18
Sans_AG_Baseline_Init	0.12	0.30	0.07	0.25	0.05	0.22	0.03	0.18	0.02	0.17

Tableau 5.5.c : Comparaison de la précision moyenne et précision à *15* documents

Cas d'une fusion sélective

Ces résultats confirment bien ceux obtenus précédemment. Plus précisément, on calcule un taux d'accroissement à la 1ère itération de *75%* et *66%* respectivement pour la précision moyenne et précision à *15* documents.

5.3.3. Evaluation comparative des méthodes de fusion

Nous procédons à présent, à l'évaluation des différentes méthodes de fusion. Rappelons que la fusion porte sur les résultats d'évaluation locale à une niche ou globale à la population, et permet de constituer à chaque itération, une liste unique présentée à l'utilisateur.

Le tableau 5.6. présente une synthèse comparative des performances réalisées à l'aide des différentes méthodes de fusion proposées, en basant l'évaluation relativement à la *Baseline* et *Baseline_Init.*

Les lignes *Accroissement moyen/Baseline* et *Accroissement moyen/Baseline_Init* traduisent respectivement l'accroissement moyen réalisé sur les *5* itérations feedback, en considérant respectivement les listes *Baseline* et *Baseline_Init.*

La ligne *Accroissement moyen* traduit l'accroissement moyen obtenu relativement aux deux bases d'évaluation.

Performances	**Fusion totale**	**Fusion élitiste**	**Fusion sélective**
Accroissement/Baseline	25%	23%	29%
Accroissement/Baseline_Init	25%	31%	34%
Accroissement moyen	25%	23%	31%

Tableau 5.6 : Comparaison entre performances réalisées avec les différentes méthodes de fusion

Nous constatons globalement que la fusion sélective, qui est une méthode basée sur l'ordre global à la population, assure de meilleures performances *(31%)* que les autres méthodes basées sur l'ordre local aux niches *(23% et 25%).*

Ceci confirme bien nos suppositions initiales. En effet, nous pensons que l'ordonnancement des résultats d'évaluation des différentes niches, préconisé comme étape préliminaire dans le cas des méthodes basées sur l'ordre local, pourrait d'emblée éliminer des documents éventuellement pertinents , et donc non considérés lors de la seconde étape (liste partielles limitées). En outre, la considération de la valeur d'adaptation directe de l'individu requête lors de la constitution de l'ordre (choix des individus dont l'adaptation est supérieure à la moyenne de la population) permet, comparativement aux autres méthodes, de diminuer l'effet de variation des valeurs d'adaptation des individus requêtes d'une même niche.

Les deux autres méthodes, basées sur l'ordre local, présentent des performances comparables. On note cependant que la fusion élitiste apporte de meilleures performances relativement à la *Baseline_Init (31%)* que la fusion totale *(25%).*
Ceci peut s'expliquer par le fait que la fusion élitiste qui considère les résultats d'évaluation des requêtes de meilleure valeur d'adaptation de chaque niche, assure une qualité d'ordre meilleure que celle obtenue par considération des résultats d'évaluation de chaque individu requête de chaque niche. Dans ce cas en effet, l'échelle des valeurs obtenues grâce à la fonction d'adaptation ne permet pas d'atténuer suffisamment des valeurs de pertinence importantes obtenues par la fonction pertinence du SRI, mais ne traduisant pas la pertinence effective de l'utilisateur.

A l'issue de cette analyse, nous retenons la méthode de fusion sélective pour nos expérimentations ultérieures.

5.4. Impact de l'ajustement de la fonction d'adaptation

Nous avons préalablement proposé l'ajustement de la fonction d'adaptation par un facteur lié à la taille des niches. Nous évaluons dans ce paragraphe, l'impact de cet ajustement sur les résultats de recherche.
Le tableau 5.7 présente les résultats obtenus par application de la fonction d'adaptation ajustée et fonction d'adaptation brute (non ajustée) pour les conditions expérimentales retenues précédemment.

Type d'adaptation	**Iter1**	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Adaptation ajustée	180(180)	88(268)	97(366)	75(442)	77(519)
Adaptation brute	180(180)	88(268)	97(366)	75(442)	78(520)

Tableau 5.7 : Effet de l'ajustement de la fonction d'adaptation

On constate clairement que l'ajustement de la fonction d'adaptation n'a aucun impact notable sur les résultats de recherche. Ceci peut être aisément justifié par le fait que, dans le cadre précis de notre approche, la taille de population adéquate est petite *(2 à 4).* Par conséquent, le nombre et tailles des niches est réduit, ce qui ne permet pas de relativiser considérablement les valeurs d'adaptations des individus requêtes associés éventuellement à différentes niches. Ceci conduit alors à la génération de valeurs d'adaptation quasi égales à celles générées par la fonction d'adaptation brute et partant, de produire sous l'effet des opérateurs génétiques, des niches quasi similaires avec une différence peu significative, liée à leur mode d'application stochastique.

5.5. Impact des opérateurs génétiques augmentés

Les opérateurs génétiques proposés dans notre approche, sont augmentés par une connaissance liée aux techniques de reformulation de requête par injection de pertinence. Nous évaluons à présent, l'impact de ces opérateurs spécifiques sur les résultats de recherche. A cet effet, nous avons effectué une série d'expérimentations en utilisant les groupes d'opérateurs *G1* et *G3*.

Le tableau 5.8 présente le nombre de documents pertinents par itération et nombre de documents pertinents cumulé obtenus par application de chacun de ces groupes d'opérateurs. La ligne *Accroissement* précise le taux d'accroissement du nombre de documents pertinents, enregistré par application du groupe d'opérateurs augmentés *G1* et ce, comparativement à l'application du groupe d'opérateurs aveugles *G3*.

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
G3	171(171)	79(250)	65(315)	65(380)	68(449)
G1	180(180)	88(268)	97(366)	75(442)	78(520)
Accroissement	5,2%(5,2%)	5,2%(7,2%)	4,9%(16%)	15%(16%)	14%(15%)

Tableau 5.8 : Impact des différents jeux d'opérateurs génétiques sur les résultats de recherche

On constate clairement que le groupe d'opérateurs génétiques augmentés assure de meilleurs résultats de recherche. Plus précisément, on enregistre un accroissement de *5%* à *15%* du nombre de documents pertinents cumulés à chaque itération. Ceci confirme l'intérêt de l'intégration de la connaissance spécifique aux techniques de recherche d'information, à la structure des opérateurs.

5.6. Impact des heuristiques d'évolution

Les heuristiques d'évolution ont été intégrées à l'AG d'optimisation de requête en vue d'améliorer la qualité des résultats de recherche en termes de valeurs de rappel et précision obtenues à un nombre de générations relativement réduit.

Dans le but d'évaluer l'impact isolé et impact combiné de ces heuristiques sur la recherche, nous avons effectué des expérimentations qui consistent à intégrer chacune d'elles séparément, les deux conjointement et enfin, les éliminer toutes les deux dans le processus génétique d'optimisation.

Les résultats obtenus en termes de documents pertinents retrouvés et précisions à *15* documents enregistrées, sont présentés respectivement sur les tableaux 5.9 et 5.10.

Les lignes *Aucune, Req_Virt* et *Req_Opt* traduisent les résultats obtenus respectivement en intégrant aucune heuristique, en intégrant uniquement la requête virtuelle et en intégrant uniquement la requête optimale.

La ligne *Req_Virt + Req_Opt* traduit l'intégration conjointe des deux heuristiques. La ligne *Accroissement* traduit le taux d'accroissement enregistré relativement au nombre de documents pertinents et nombre de documents pertinents cumulé obtenus à chaque itération dans le cas où aucune heuristique n'est intégrée.

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Aucune	171(171)	84(255)	61(316)	60(377)	56(434)
Req_Virt	177(177)	112(289)	81(370)	74(444)	65(509)
Accroissement	3% (3%)	33% (13%)	32% (17%)	23% (17%)	16% (17%)
Req_Opt	177 (177)	95 (272)	98 (370)	57 (428)	69 (496)
Accroissement	3% (3%)	13% (6%)	60% (17%)	-5% (13%)	23% (14%)
Req_Virt + Req_Opt	180(177)	88(268)	97(366)	75(442)	78(520)
Accroissement	5% (5%)	4% (5%)	60% (15%)	25% (17%)	33% (15%)

Tableau 5.9 : Impact des heuristiques d'évolution sur le nombre de documents pertinents restitués

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Aucune	0.47	0.25	0.17	0.16	0.15
Req_Virt	0.49	0.31	0.22	0.20	0.18
Accroissement	4%	24%	29%	25%	20%
Req_Opt	0.49	0.26	0.27	0.16	0.19
Accroissement	4%	4%	58%	0%	26%
Req_Virt + Req_Opt	0.50	0.31	0.24	0.20	0.19
Accroissement	6%	24%	41%	25%	26%

Tableau 5.10 : Impact des heuristiques d'évolution sur la précision des résultats de recherche

Les résultats montrent que chacune des heuristiques a un impact positif sur les résultats de recherche, qui se traduit par un accroissement du rappel de *3%* à *17%* et de la précision de *0%* à *26%* avec un pic de *58%* pour la requête optimale à la troisième génération de l'AG.

Notons que l'intégration de la requête virtuelle améliore de manière considérable les valeurs de rappel et précision à la deuxième itération *(33%* et *24%)* relativement à l'effet d'intégration de la requête optimale à la même itération *(13%* et *4%)*. Ceci peut être dû au fait qu'étant constituées des meilleurs termes issus de la recherche initiale, cette requête offre l'avantage de cibler le voisinage documentaire *immédiat* du sous espace atteint par la requête initiale et *corrigé* par le jugement de pertinence obtenu à la première itération.

Par ailleurs, on remarque, en comparant l'impact d'intégration isolée et intégration combinée des deux heuristiques, que leurs effets ne sont pas simplement additifs, sur les résultats de recherche. A titre d'exemple, on note sur le tableau 6.10.a qu' à l'itération *2*, les heuristiques assurent isolément des accroissements de *33% et 13%* et conjointement, un accroissement de *4%*. Ceci est forcément dû aux effets de la méthode de fusion qui combine la valeur d'adaptation des requêtes et valeurs de pertinence supposée des documents issus de l'évaluation de ces même requêtes, et qui ne sont pas nécessairement corrélées.

Au delà de l'itération *2*, le taux d'accroissement enregistré en combinant les deux heuristiques a tendance à être supérieur ou égal à la meilleure performance enregistrée par intégration de l'une ou l'autre des heuristiques.

6. Evaluation comparative des approches proposées

Nous présentons dans ce paragraphe une analyse comparative des résultats obtenus en utilisant l'approche d'optimisation de requête de base, présentée dans le quatrième chapitre, et approche améliorée présentée dans le présent chapitre.
Rappelons que notre approche est initialement fondée sur la coopération d'individus requêtes alors que l'approche améliorée est d'avantage basée sur une recherche menée par des niches d'individus requêtes.
Le tableau 5.11 présente les résultats obtenus. Les lignes *Approche de base* et *Approche améliorée* traduisent le nombre de documents pertinents par itération et nombre de documents pertinents cumulés pour chacune des approches.
Les lignes *Accroissement/Baseline* et *Accroissement/Baseline_Init* présentent, pour chaque approche, le taux d'accroissement relativement au nombre de documents pertinents cumulé et ce, comparativement aux références d'évaluation respectives baseline et baseline_init.

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Approche de base	177(177)	114(291)	93(384)	69(453)	56(510)
Accroissement/Baseline	38%	23%	13%	15%	15%
Accroissement/Baseline_Init	38%	41%	24%	25%	22%
Approche améliorée	180(180)	88(268)	97(366)	75(442)	78(520)
Accroissement/Baseline	63%	18%	21%	22%	22%
Accroissement/Baseline_Init	63%	32%	28%	25%	26%

Tableau 5.11 : Comparaison des approches proposées

On constate clairement que l'approche basée sur les niches de requêtes donne de meilleurs résultats. Plus précisément, le nombre de documents cumulés à la cinquième itération est de *510* pour l'approche de base et *520* pour l'approche améliorée ; hormis l'itération *2*, le nombre de documents pertinents retrouvés par itération est plus élevé dans le cas de l'approche améliorée.
En outre, on note que les taux d'accroissement relativement aux listes de référence baseline et baseline_init, sont nettement meilleurs dans le cas de l'approche améliorée. Ceci montre que le principe d'optimisation de niches de requêtes offre de meilleures possibilités de rappel de nouveaux documents pertinents et ce, en comparant aussi bien aux étapes de recherche précédentes qu'à l'étape initiale par balayage en bas de liste.

Par ailleurs, nous avons évalué l'impact isolé de la technique de nichage. A cet effet, nous avons expérimenté le processus génétique de requête en retenant toutes les conditions d'expérimentation de l'approche basée sur le nichage (fonction d'adaptation, types d'opérateurs génétiques, heuristiques d'évolution) et avons comparé les résultats avec ceux obtenus en retenant ces mêmes conditions avec cependant, une évolution génétique fondée sur les individus requêtes comme proposé dans l'approche de base et non fondée sur les niches de requêtes. Les résultats sont présentés sur le tableau 5.12.

	Iter1	**Iter2**	**Iter3**	**Iter4**	**Iter5**
Sans Nichage	117(177)	124(302)	84(387)	64(451)	56(507)
Avec Nichage	180(180)	88(268)	97(366)	75(442)	78(520)

Tableau 5.12 : Impact de la technique de nichage sur les résultats de recherche

On constate que la technique de nichage a globalement un impact positif sur les résultats de recherche.
Cette technique étant intégrée en vue d'accroître les performances de recherche pour des requêtes dont les documents pertinents sont relativement dissemblants, nous avons alors effectué une analyse des résultats par requête, en considérant pour chacune d'elles, la distance moyenne entre ses documents pertinents. A cet effet, nous avons construit trois catégories de requêtes caractérisées par une **distance entre documents pertinents** inscrits dans les intervalles *[20 25[, [25 30[* et *[30 35]* et qui définissent ainsi des **régions de documents**.

La figure 5.5. illustre les résultats obtenus en termes de nombre de documents pertinents cumulés à la cinquième itération feedback correspondant à la cinquième génération de l'AG.

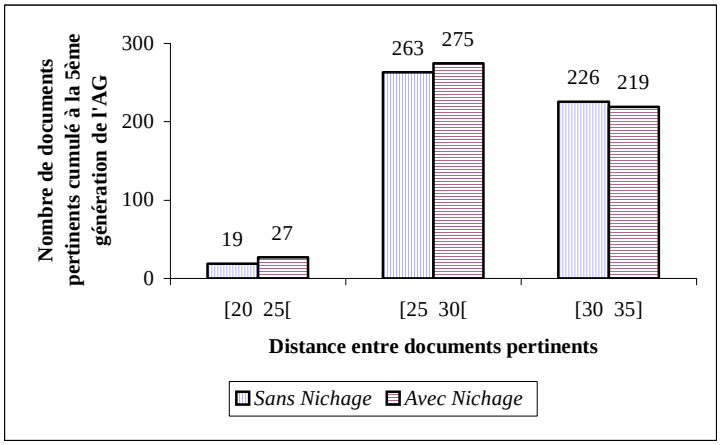


Figure 5.5 : Impact de la technique de nichage

On constate clairement que l'intégration de la technique de nichage assure, seule, de meilleures performances pour les deux premières catégories. Plus précisément, on note une augmentation de *19* à *27* documents pertinents pour la première catégorie, soit un accroissement de *42%* et, une augmentation de *263* à *275* pour la deuxième catégorie, soit un accroissement de *45%*.
Cependant, on constate une baisse de *226* à *219*, soit un décroissement de *31%* dans le cas de la troisième catégorie.
Ceci peut être imputé au fait qu'une distance relativement importante des documents pertinents rend la technique non concluante au bout d'un nombre réduit de générations de l'AG. L'analyse de cette catégorie particulière de requêtes à la sixième génération de l'AG, nous a permis de dresser le tableau 5.13. Les colonnes *Sans Nichage* et *Avec Nichage* donnent le nombre de documents pertinents cumulés à la sixième génération de l'AG.

N° Requête	Sans Nichage	Avec Nichage
22	64	*83*
11	41	*40*
25	14	*14*
10	40	*40*
16	6	*9*
12	37	*33*
21	9	*10*
17	55	*53*
14	16	*14*
Total	***282***	***296***

Tableau 5.13 : Impact du nichage à la 6ème génération de l'AG
Cas de la troisième catégorie de requêtes

Les résultats montrent en effet une augmentation de *282* à *296* documents pertinents ce qui traduit un accroissement de *4,9%* des performances. On montre ainsi que la technique de nichage a un impact intrinsèque sur les résultats de recherche ; cette technique permet effectivement le rappel de documents pertinents relativement dissemblants avec cependant, un temps de convergence dépendant du degré de dissemblance.
Cette expérimentation attire alors notre attention sur l'intérêt d'ajuster en cours de recherche, l'opérateur de conichage avec des paramètres liés à la mesure de distance entre documents jugés pertinents lors de précédentes itérations feedback.

7. Bilan

L'approche d'optimisation de requête présentée, demeure basée sur l'exploitation conjointe des techniques d'algorithmique génétique et de reformulation de requête par injection de pertinence.
Cependant, et comparativement à la précédente approche, l'heuristique de nichage est diffusée globalement durant l'évolution de l'AG. En ce sens que la population est manipulée en niches constituant des sous populations qui activent simultanément dans des régions différentes de l'espace documentaire.
L'AG standard opère alors sur chaque niche et exploite les résultats de sa recherche en accord avec sa valeur d'adaptation relative dans la population.
Les opérateurs génétiques proposés sont augmentés par une connaissance liée aux techniques de reformulation de requête et, adaptés à l'évolution de niches de requêtes.

Les expérimentations et évaluations que nous avons réalisées confirment l'intérêt de notre approche. Nous avons en particulier étudié l'apport de performances dû aux éléments de l'AG d'optimisation de requête que nous proposons et, évalué l'impact de chacun de ses paramètres sur les résultats de recherche.
Une première série d'expérimentations nous a permis d'estimer l'effet combiné de la taille de population et proportion de conichage. On y a alors montré la nécessité d'ajuster ces paramètres afin d'assurer un équilibre entre nombre de requêtes et facteur de multiplicité des directions de recherche.
L'évaluation de l'optimisation génétique de requête relativement aux références baseline et baseline_init montre un accroissement de *8%* à *63%* des performances en fonction des itérations feedback et des méthodes de fusion appliquées. Concernant ces dernières, une analyse comparative nous a permis de privilégier l'application d'une méthode basée sur l'ordre global à la population plutôt qu'une méthode locale aux niches.
Après détermination des meilleures conditions expérimentales, nous avons procédé à l'évaluation de l'impact de chacun des éléments caractéristiques de l'AG. Il en ressort principalement que l'intégration de la connaissance à la structure des opérateurs génétiques a un impact positif indéniable sur les résultats de recherche.
Concernant les heuristiques d'évolution, on montre qu'elles assurent une meilleure qualité des résultats. Cependant, leurs effets ne sont pas additifs en raison de l'action de la fusion sur les résultats d'évaluations des individus requêtes.
Enfin, une évaluation comparative des approches proposées nous permet de privilégier une recherche basée sur des niches de requêtes. L'estimation de l'impact isolé de la technique de nichage montre d'une part son efficacité à rappeler des documents pertinents dissemblants et qui se traduit par un accroissement des performances de recherche pour des catégories de requêtes situées à différentes régions de documents sur la base de la distance moyenne entre eux. D'autre part, les résultats montrent que le

seuil de convergence pourrait être dépendant du degré de dissemblance de ces documents. Ceci nous encourage alors à mener une réflexion sur le principe de construction des niches et ce, dans le sens d'y intégrer des paramètres liés aux mesures de distances entre documents préalablement jugés pertinents pour la requête en cours d'évaluation.

Conclusion et perspectives

Les travaux développés dans ce mémoire s'inscrivent dans le cadre de la conception de SRI adaptatifs aux besoins des utilisateurs. Nous y avons proposé une approche d'optimisation de requête basée sur les concepts de la génétique.

De nombreux modèles et stratégies de recherche d'information ont été proposés dans la littérature. Chacun d'eux utilise les éléments d'un théorie formelle afin de résoudre les problèmes inhérents à la recherche d'information : représentation du sens des documents et requêtes, traduction des liens sémantiques entre concepts, estimation de la pertinence …

La reformulation de requête a été proposée comme un mécanisme permettant de se greffer à un modèle de base, afin d'adapter la description de la requête à l'environnement linguistique du système. Bien que les techniques classiques d'expansion de requête apportent des améliorations appréciables aux résultats de recherche, force est de constater que le principe est fondé sur une manipulation des termes de manière indépendante. En outre, l'efficacité de la recherche dépend de deux facteurs : le premier porte sur la ressemblance des documents pertinents et le second porte sur la ressemblance du descripteur initial de la requête aux descripteurs des documents pertinents. En effet, dans le cas d'une dissemblance relativement importante, le processus de recherche ne peut être concluant à un nombre réduit d'itérations feedback.

Dans ce contexte de travail, nous nous sommes intéressés à la mise en œuvre d'un AG d'optimisation de requête. Les AG's sont des procédés d'optimisation qui présentent des propriétés fort intéressantes : exploration parallèle et efficace d'espaces complexes, construction graduelle de solutions partielles, recherche coopérative.

Vus sous l'angle de notre approche de recherche d'information, ces algorithmes sont exploités en vue de construire, dans l'espace défini dans la collection de documents, la structure de la requête optimale, permettant de rappeler le maximum de documents pertinents associés au besoin en information de l'utilisateur.

De manière inhérente à leur fonctionnement, nous mettons ainsi en œuvre un processus de recherche qui traite les dépendances entre termes et utilise le résultat d'évaluation des différentes requêtes qui explorent l'espace documentaire.

Le principe d'optimisation que nous préconisons est en outre caractérisé par l'intégration de techniques avancées de l'exploration génétique : technique de nichage et d'adaptation des opérateurs génétiques.

La technique de nichage nous permet le rappel de documents pertinents pour une même requête mais relativement dissemblants.

L'adaptation des opérateurs génétiques permet de réduire la complexité d'exploration en régulant les transformations génétiques opérées sur les individus requêtes, grâce à une connaissance approuvée liée aux techniques de reformulation de requête dans les SRI.

Une évaluation préliminaire a globalement montré l'intérêt de notre approche. Un bilan critique nous a conduit à réviser de nombreux éléments de l'AG : organisation de la population, fonction d'adaptation, structure et type d'application des opérateurs et enfin principe de fusion des résultats de recherche.
L'évaluation de notre nouvelle approche d'optimisation de requête a montré son impact positif sur les résultats de recherche.
Plus précisément, l'analyse des résultats révèle un accroissement considérable des performances lié au traitement des niches, utilisation de la connaissance dans les opérateurs génétiques et intégration d'heuristiques d'évolution à l'AG.

Les perspectives envisageables à nos travaux portent essentiellement sur deux volets. Le premier concerne l'établissement de relations formelles entre paramètres de l'AG : taille de population, proportion de conichage, taille de la liste des termes d'expansion, formules de pondération … et caractéristiques des collections interrogées : nombre total de documents, nombre total de termes, longueur moyenne de documents, longueur moyenne de requêtes…
L'analyse expérimentale des résultats issus de l'interrogation de différents types de collections d'une part, et analyse théorique par utilisation de test de corrélation d'autre part, nous permettront alors de définir des conditions d'exploitation génériques pour l'approche que nous proposons.
Le second volet porte sur l'évaluation de la faisabilité de l'approche globale pour une tâche de filtrage d'informations. Il est alors impératif, eu égard aux caractéristiques du processus de filtrage d'informations, d'approfondir la réflexion sur l'opportunité de nombreux éléments de l'AG dont on cite principalement : fonction d'adaptation et principe de fusion des résultats de recherche.
Une telle extension permettrait d'accroître de manière homogène et modulaire les fonctionnalités d'un SRI de manière générale.
Ces travaux élargiront sans doute le champ d'application des AG's à la recherche d'information.

REFERENCES

[Aarts & al, 1989] E.H.L. Aarts, A.E. Eiben, K.M Vanhee : *A General Theory of Genetic Algorithms*, Computing Science Notes, Eindhoven University of Technology Netherlands, 1989

[Aboud, 1990] M. Aboud : *Systèmes de Recherche d'informations : Thesaurus et Classification*, Thèse de Doctorat de l'Université Paul Sabatier, Toulouse II (France), Décembre 1990

[Ankenbrandt, 1990] Ankenbrandt C. : *An Extension to the Theory of Convergence and a Proof of the Time Complexity of Genetic Algorithms*, FOGA90, pp 53-58, 1990

[Attar & Fraenckel, 1977] R. Attar & S. Franenckel : *Local Feedback in Full Text Retrieval Systems.* Journal of the ACM, 397-417, 1977

[Baeck & Al, 1991] Baeck T., Hoffmeister F. & Schwefel H.P : *A Survey of Evolution Strategies*, ICGA 4, pp 2-9, 1991

[Bagley, 1967] Bagley J.D : *The Behavior of Adaptive Systems wich Employ Genetic and Correlation Algorithms*, Doctoral Dissertation, University of Michigan, Dissertation Abstracts International, 28(1), 510B, 1967

[Baker, 1985] J E.Baker : *Adaptive Sélection Methods for Genetic Algorithm,* ICGA1 1985

[Bartell & al, 1994] B. Bartell, G. Cottrel & R.K Belew : *Automatic Combination of Multiple Ranked Retrieval Systems*, In Proceedings of the 17th Annual ACM SIGIR, Conference on Research and Development in Information Retrieval

[Bartell & al, 1998] B. Bartell, G.Cottrell and R.K. Belew : *Optimizing Similarity Using Multiquery Relevance Feedback*, JASIS 49(8), 1998

[Bean & Hadj-Alouane, 1992] J.C. Bean, A.B Hadj-Alouane : *A dual Genetic Algorithm for Bounded Integer Programs*, Tr 92-53, Department of Industrial and Operations Engineering, the University of Michigan, 1992

[Blosseville & al, 1992] M.J Blosseville, G. Hébrail, M.G Monteil & N. Pénot : *Automatic Classification : Natural Langage Processing, Statical Analysis and Expert Techniques Used together*, Conference on Research and Development in Information Retrieval (SIGIR), pp 51- 58, 1992

[Boughanem, 1992] M. Boughanem : *Les Systèmes de Recherche d'informations : D'un Modèle Classique à un Modèle Connexioniste*, Thèse de Doctorat De l'Université Paul Sabatier, Toulouse (France), Décembre1992

[Boughanem & Soule-Dupuy, 1997] M. Boughanem, C. SouleDupuy : *Mercure at TREC 6 :* In Harman DK, ed. 6th International Conference on Text Retrieval TREC 6. November 21-23, NIST SP, pp. 321-328, 1997

[Boughanem & Soule-Dupuy, 1999] M. Boughanem & C. Soule-Dupuy : *Query Modification Based on Relevance Backpropagation in Adhoc Environment,* Information Processing and Management, 1999

[Boughanem & al, 1999] M. Boughanem, C. Chrisment & L.Tamine, Genetic Approach to Query Space Exploration. Information Retrieval Journal volume 1 N°=3 , pp175-192, 1999

[Bourne & Anderson, 1979] C. Bourne, B.Anderson : *DIALOG LabWorkbook,* second edition, Lockheed Information Systems, PaloAlto, Californie (USA), 1979

[Bourret & Samuelides, 1991] P.Bourret & J.Reggia Samuelides : *Réseaux de Neurones, une Approche Connexioniste de L'Intelligence Artificielle,* Edition TEKNEA 1991

[Buckley & al, 1994] C. Buckley, G. Salton & J. Allan : *The Effect of Adding Information in a Relevance Feedback Environment,* Conference on Research and Development in Information Retrieval (SIGIR), 1994

[Callan, 1994] J.P Callan : *Passage Level Evidence in Document Retrieval,* Conference on Research and Development in Information Retrieval (SIGIR), pp 302-309, 1994

[Callan & al, 1995] J.P. Callan, W.B Croft, J. Broglio : *TREC and TIPSTER Experiments with INQUERY,* Information Processing and Management

[Can & Ozkarahan, 1990] F. Can & E.A. Ozkarahan : *Concepts and Effectiveness of the Cover-Coefficient for Text Databases,* ACM Transactions Database Systems, Vol. 15 N°4 pp 483-517, 1990

[Carolyn & Yang, 1992] Carolyn J., B. Yang : *Experiments in Automatic Statistical Thesaurus Construction,* In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pp 77-88, Copenhage, Denmark, 1992

[Carpineto & al, 1999] C. Carpineto, R. DeMori & G.Romano : *Informative Term Selection for Automatic Query Expansion* , In the Proceedings of the 7th Text Retrieval Conference TREC7, July 1999

[Cavarlho & Freitas 2000] D.R. Cavarlho, A.A. Freitas : *A hybrid Decision Tree/Genetic Algorithm for Coping with the Problem of Small disjuncts in DataMining,* In Proceedings of Genetics Evolutionary Computation Conference, Las Vegas USA, July 2000

[Cerf, 1994] R. Cerf: *Une Théorie Assymptotique des Algorithmes Génétiques,* Thèse Phd, Université de Montpellier II, Mars 1994

[Chen , 1995] Chen H. : Machine Learning for Information Retrieval : Neural Networks, Symbolic Learning and Genetic Algorithms, JASIS, 46(3) : pp 194-216

[Chen & Ng, 1995] H. Chen & T. Ng : *An Algorithmic Approach to Concept Exploration in Large Knowledge Network (automatic thesaurus consultation) : Symbolic Branch and Bround Search vs Connexionist Hopfield net Activation* Journal of the American Society for Information Science, 348-369

[Chen &Wang, 1995] S. Chen & J.Y Yang : *Document Retrieval Using Knowledge Based Fuzzy Information Retrival Technique.* IEE Transactions on Systems, Man and Cybernetics, 793-803

[Cobb & Grefenstette, 1993] Cobb H. Et Grefenstette J. J : *Genetic Algorithms for Tracking Changing Environnements* , ICGA5 pp 523-530,1993

[Cormack & al, 1999] G. Cormack, C.R. Palme, M.V Biesbrouk & C.L.A. Clarck: *Deriving Very Short Queries for High Precision and Recall,* In Proceedings of the 7th Text Retrieval Conference TREC7, July 1999

[Croft & Harper, 1979] W. Croft & D. Haper : *Using Probabilistic Models for Document Retrieval without Relevance Information,* Journal of Documentation pp 185-202, 1979

[Davidor, 1990] Davidor Y. : *Epistasis Variance : A Viewpoint on GA-Hardness,* FOGA 90 pp 23-25, 1990

[Deerwester & al, 1990] S. Deerwester, S. Dumais, S. Furnas, G. Landauer & R. Harshman : *Indexing by Latent Semantic Analysis :* Journal of the American Society for Information Science, 391-407

[Dejong & Sarma, 1995] K.A Dejong and J. Sarma : *On Decentralizing Selection Algorithm,* In L.J. Eshelman, Editor, Proceedings of the 6th International Conference on Genetic Algorithms, Pages 17- 23 Morgan Kaufmann, 1995

[Dumais, 1994] S. Dumais : *Latent Semantic Indexing (LSI), TREC3 report.* In Proceedings of the 3rd Conference on Text Retrieval Conference, 1994

[Duvivier & al, 1998] D. Duvivier, Ph. Preux, C. Fonlipt, D. Robillard, E.G. Talbi : *The fitness Function and its Impact on Local Search Methods,* IEEE Systems, Man and Cybernetics, San Diego, USA, October, 1998

[Efthimidias, 1996] E. Efthimidias , *Query Expansion :* Annual Review on Technology (ARIST), Volume 31, p 121-187, 1996

[Fidelis & al, 2000] M.V. Fidelis, H.S. Lopes, A.A. Freitas : *Discovering Comprehensible Classification Rules with a Genetic Algorithm,* 1st IEEE Symposium on Combination, Evolutionary Computation & Neural Networks, San Antonio, May 2000

[Flurh & Debili, 1985] C. Flurh & F. Debili : *Interrogation en Langue Naturelle de Données Textuelles et Factuelles,* Intelligent Multimadia Information Systems and Management (RIAO), Grenoble (France), pp 548-556, 1985

[Fogel & al, 1966] L. Fogel, A.J Owens, M.J Walsh : *Artificial Intelligence through Simulated Evolution,* New York John Wiley, 1966

[Fogel, 1995] D.B Fogel : *Toward a new Philosophy of Machine Intelligence,* IEEE Press, Piscataway, NJ, 1995

[Fox, 1983] E.A. Fox : *Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types.* PhD thesis, Cornell University, Ithaca NewYork, 1983

[Freitas, 1999] A.A. Freitas : *A Genetic Algorithm for Generalized Rule Induction,* In R.Roy & al advances in soft computing engeering design & manufacturing pp 340-353, Proc WSC3, 3rd On line world conference on soft computing, Springer Verlag, 1999

[Gauch & Wang, 1996] S. Gauch & J. Wang : *Corpus Analysis for TREC 5 Query Expansion,* In Proceedings of the 5th Text Retrieval Conference (TREC5), 537-546, 1996

[Genest, 1999] D. Genest, *Vers un Système de Recherche Documentaire basé sur les Graphes Conceptuels :* Actes du congrès Inforsid, p115-131, Juin 1999, LAGARDE France

[Goldberg, 1983] Goldberg D.E : *Computer-Aided Pipeline Operation Using Genetic Algorithms and Rule Learning,* Doctoral Dissertation, University of Michigan. Dissertation Abstracts International, 44(10), 3174B, University Microfilms N° 8402282, 1983

[Goldberg & Lingle, 1985] Goldberg D.E & R.Lingle : *Alleles, Loci and the Travelling Salesman Problem,* Proceedings of the 1st International Conference on Genetic Algorithms, pp154-159, 1985

[Goldberg & Richardson, 1987] Goldberg D.E & Richardson : *Genetic Algorithms with Sharing for Multimodal Function Optimization,* ICGA 2, pp 41-49, 1987

[Goldberg, 1989] Goldberg D.E : *Genetic Algorithms in Search, Optimisation and Machine Learning,* Edition Addison Wesley 1989

[Goldberg, 1994] Goldberg D.E : *Algorithmes Génétiques, Exploration, Optimisation et Apprentissage Automatique,* Edition Addison Wesley, 1994

[Gordon, 1988] M. Gordon : *Probabilistic and Genetic Algorithms for Document Retrieval,* Communications of the ACM pp 1208-1218, October 1988

[Gordon, 1991]] M. D. Gordon : *User-Based Document Clustering By Redescribing Subject Descriptions with a Genetic Algorithm,* Journal of The American Society for Information Science, 42(5) pp 311 - 322, 1991

[Grefenstette & al, 1985] Grefestette J.J., R. Gopal, B. Rosmaita Et D Van. Guht : *Genetic Algorithms for the Travelling Salesman Problem,* 1 St International Conference on Genetic Algorithms, 1985

[Grossman & Frieder, 1998] D.A. Grossman & O. Frieder : *Information Retrieval : Algorithms and heuristics*, Kluwer Academic Publishers, 1998

[Guttman, 1978] Guttman L. : *What is Who What in Statistics*, the statistician, 26 : pp 81-107, 1978

[Haines & Croft, 1993] D. Haines & W.B Croft : *Relevance Feedback and Inference Networks*, Conference on Research and Development in Information Retrieval (SIGIR), pp 2-11, 1993

[Harman, 1992] D. Harman : *Relevance Feedabck Revisited :* Conference on Research and Development in Information Retrieval (SIGIR), pp 1-10, 1992

[Hartman & Belew, 1991] Hartman W.& Belew R.K : *Optimizing An Arbitrary Function is Hard for the Genetic Algorithm*, ICGA4 pp 190-195, 1991

[Hebb, 1949] : D.O Hebb : *The Organization of Behaviour :* J. Wiley & Sons, New York, 1949

[Hofman, 1999] T. Hofman, *Probabilistic Latent Semantic Indexing :* In the Proceedings of the 22nd Annual International ACM SIGIR, Conference on Research and Development in Information Retrieval, August, 1999, Buckley USA

[Holland, 1962] Holland J. : Concerning Efficicent Adaptive Systems.In M.C Yovits, G.T Jacobi, &G.D Goldstein(Eds) Self Organizing Systems pp 215-230 Washinton : Spartan Books, 1962

[Holland, 1975] Holland J. : *Adaptation In Natural and Artificial Systems,* University of Michigan Press, Ann Arbor, 1975

[Holland, 1992] Holland J. : *Les Algorithmes Génétiques*, Revue POUR LA SCIENCE N°=179 pp 44-51, Septembre 1992

[Jing & Tzoukerman, 1999] H. Jing & E. Tzoukermann : *Information Retrieval Based on Context Distance and Morphology,* In proceedings of the 22th Annual International ACM SIGIR, Conference on Research and Development in Information Retrieval, August, 1999, Buckley USA

[Kettaf, 1995] F.Z Kettaf : *Présentation des Algorithmes Génétiques, R*apport Interne N°=46, Université François Rabelais Septembre 1995

[Knaus & al, 1994] Knaus D., Mittendo, Shausle : *Improving Basic Retrieval Method by Links and Passage Level Evidence,* In Proceedings of the 3rd Text Retrieval Conference (TREC 3) pp 241-246, 1993

[Korfhage, 1997] R. Korfhage : *Information Storage and Retrieval.* Jhon Wiley & sons, Inc 1997

[Koza, 1991] Koza JR : *A Hierachical Approach to Learning the Boolean Multiplexer Function,* In Rawlins G. Ed., Foundations of Genetic Algorithms, Morgan Kaufman, San Mateo, CA, pp 171-192, 1991

[Koza, 1992] J.R. Koza : *Genetic Programming* A bradford book, MIT Press, Cambrige, MA, USA 1992

[Kraft & al, 1995] Kraft DH,Petry FE, Buckles BP and Sadisavan T : *Applying Genetic Algorithms to Information Rtrieval System Via Relevance Feedback*, In Bosc and Kacprzyk J eds, Fuzziness in Databse Management Systems Studies in Fuzziness Series, Physica Verlag, Heidelberg, Germany pp 330-344

[Kwok, 1995] K.L Kwok : *A Network Approach to Probabilistic Information Retrieval,* ACM Transactions on Information Systems, Vol 13 N3 pp 324 - 353, 1995

[Lelu & François, 1992]A. Lelu & C. François : *Information Retrieval Based on Neural Unsupervised Extraction of Thematic Fuzzy Clusters,* Les Réseaux Neuromimétiques et leurs Applications (Neuronîmes), pp 93-104, 1992

[Lewis & Ringuette, 1994] D.D Lewis & M. Ringuette : *A Comparison of two Learning Algorithms of Text Categorization,* Symposium on Document Analysis and Information Retrieval, 1994

[Lucarella & Morara, 1991] D. Lucarella & R. Morara : *FIRST Fuzzy Information Retrieval Systems.* Journal of Information Science, 81-91, 1991

[Lundquist & al, 1997] C. Lundquist, D. Grossman & O. Frieder : *Improving Relevance Feedback in the Vector Space Model.* In The Proceedings of the 6th ACM Annual Conference on Information Knowledge Management (CIKM'97)

[Mandala & al, 1999] R. Mandala, T.Tokunaga & H. Takana, *Combining Multiple Evidence from Different Types of Thesaurus for Query Expansion,* In proceedings of the 22th Annual International ACM SIGIR, Conference on Research and Development in Information Retrieval, August, 1999, Buckley USA

[Mansanne & al, 1999] F. Mansanne, F. Carrère, A. Ehinger & M. Schoenauer : *Evolutionary Algorithms as Fitness Debuggers,* In Proceedings of ISMIS'99, LMAI 1609 Springer Verlag, pp 639-647, 1999

[Menczer & Belew, 1999] F. Menczer , R.K. Belew : *Adaptive Retrieval Agents : Internalizing Local Context and Scaling up to the WEB,* Machine Learning, pp 1-45, Kluwer Academic Publishers, 1999

[Michalewicz, 1996] Z. Michalewicz : *Genetic Algorithms + Data Structures = Evolutionary Programs,* Springer Verlag, New York, 3rd Edition, 1996

[Mitra & al, 1998] M. Mitra, A. Singhal, C.Buckley : *Improving Automatic Query Expansion,* In proceedings of the 22th Annual International ACM SIGIR, Conference on Research and Development in Information Retrieval

[Montana & Davis, 1989] D.J Montana and L.Davis : *Training Feedforward Neural Networks Using Genetic Algorithms,* In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, IJCAI 89, pp 775- 780, Detroit, MI, August pp 20-25, 1989

[Morita & Shinoda, 1994] Masahiro Morita and Yoichi Shinoda *Information Filtering Based on Ananlysis and Best Match Text Retrieval,* In Proceedings of the seventh annual International ACM-SIGIR, Conference on Research and Development in Information Retrieval, W. Bruce Croft and C.J. van Rijsbergen, Eds July 1994, pp 272-281, Springer Verlag

[Mothe, 1994] J. Mothe : *Modèle Connexioniste pour la Recherche d'informations, Expansion Dirigée de Requêtes et Apprentissage,* Thèse de l'Université Paul Sabatier Toulouse, 1994

[Oard, 1996] Douglas William Oards , *Adaptive Vector Space Text Filtering for Monolingual & Cross-language applications* ,Dissertation for the requirement for the degree of Doctor of Philosophy, University of Maryland, 1996

[Oliver & al, 1987] I.M Oliver, D.J Smith , J.R C Holland : *A Study of Permutation Cross-Over Operators on the Travelling Salesman Problem ,* ICGA 2, 1987

[Peat & Willet, 1991] H. Peat & P. Willet : *The Limitations of Term Co-occurrence Data for Query Expansion in Document Retrieval Systems.* Journal of the American Society for Information Science, 378-383

[Ponte, 1998] J.M. Ponte, *A Language Modelling Approach to Information Retrieval,* Doctor of philosophy tgesis, University of Massachusetts, September 1998

[Preux, 1995] P.Preux : *Les Algorithmes Evolutifs* , Publication LIL 94-1. Laboratoire d'Informatique Fondamentale de Lille, Octobre 1995

[Puget , 1992] D. Puget, *Aspects Sémantiques dans les Systèmes de Recherche d'informations,* Thèse de Doctorat de l'université Paul Sabatier, Toulouse (France)

[Qiu & Frei, 1993] Y. Qiu & H.P. Frei : *Concept Based Query Expansion.* In Proceedings of the 16th ACM SIGIR Conference on Research and Development in Information Retrieval, 160-169, Pittsburg, USA 1993

[Radcliffe, 1991a] N.J Radcliffe : *Equivalence Class Analysis of Genetic Algorithms* Complex Systems, 5:183-20, 1991

[Radcliffe, 1991b] N.J Radcliffe : *Set Recombination and Its Application To Neural Network Topology Optimisation,* Technical Report EPCC-TR-91-21, Edinburgh Parallel Computing Center, 1991

[Raghavan & Doegun, 1986] Raraghavan & Doegun in [Gordon, 1991]

[Razouk, 1990] R. Razouk : *Bases d'Information Généralisées : Hypermedia et Classification,* Thèse de Doctorat de L'université de Paul Sbatier, Toulouse (France), 1990

[Rijsbergen, 1979] C.J. Van Rijsbergen, *Information Retrieval,* 2ème Edition, Butterworths, Londres(UK), 1979

[Robertson & al, 1995] S.E Robertson, S.E. Walker & M.M Hnacock-Beaulieu : *Large Test Collection Experiments on an Operational Interactive System : Okapi at TREC,* in IP&M, pp 260-345, 1995

[Robertson & al, 1999] S.E. Robertson, S. Walker, M. Beaulieu : *OKAPI at TREC 7 : Automatic Adhoc Filtering, VLC and Interactive Track,* In Proceedings of the 7th Text Retrieval Conference TREC7, July 1999

[Robertson & Sparck Jones, 1976] S.E Robertson & K. Sparch Jones : *Relevance Weighting for Search Terms,* Journal of The American Society for Information Science, Vol 27, N°3, pp 129-146, 1976

[Robertson & Walker, 1994] Robertson S., Walker S. : *Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval,* In Proceedings of the seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 232-241, 1994

[Robertson & Walker, 1997] Robertson S., Walker S. : *On Relevance Weights with Little Relevance Information,* In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development, pp16-24, 1997

[Rocchio, 1971] J. J Rocchio : *Relevance Feedback in Information Retrieval,* in The Smart System Experiments in Automatic Document Processing, G.Salton, Editor, Prentice-Hall, Inc., Englewood Cliffs, NJ, pp 313-23, 1971

[Roget, 1988] P. Roget : *Roget's II the New Thesaurus,* Houghton Mifflin Company, Boston, USA, 1988

[Rosenberg, 1967] Rosenberg R.S : *Simulation of Genetic Populations with Biochemical Properties,* Doctoral Dissertation, University of Michigan. Dissertation Abstracts International, 28(7), 2732B, University Microfilms N° 67-17,836

[Salton, 1968] G. Salton : *Automatic Information and Retrieval,* Mcgrawhill Book Company, N. Y., 1968

[Salton, 1971] G. Salton : *The Smart Retrieval System : Experiments in Automatic Document Processing,* G. Salton Editor, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1971

[Salton & MacGill, 1983] : G. Salton & M.J Macgill : *Extended Booleen Information Retrieval,* Communications of The ACM, Vol. 26, N°12, pp 1022-1036, 1983

[Salton, 1989] G. Salton : *Automatic Text Processing.* The Transformation Analysis and Retrieval of Information by Computer. Addison Wesley, Reading 1989

[Salton & al, 1993] G. Salton, J.Allan & C. Buckley : *Approaches to Passage Retrieval in Full Text Information Systems,* Association for Computing Machinery, New York, June 1993

[Salton & Allan, 1994] G. Salton & J. Allan : *Automatic Text Decomposition and Structuring,* Actes du Congrès RIAO'94, Intelligent Multimedia Information on Retrieval Systems and Management, New York (USA) pp 6-20, 1994

[Salton & Buckley, 1988] G.Salton & C.Buckley : *Term Weighting Approaches in Automatic Text Retrival,* Information Processing and Management, pp 513-523, 1988

[Salton & Buckley, 1990] G.Salton & C.Buckley : *Improving Retrieval Performance By Relevance Feedback,* Journal of The American Society for Information Science, Vol. 41, N°4, pp 288-297, 1990

[Savoy & Desbois, 1991] J. Savoy & D. Desbois : *Bayesian Inference Networks in Hypertext,* Intelligent Multimedia Information Systems and Management (RIAO), pp 662-681, 1991

[Schoenauer & Michalewitz, 1997] M. Schoenauer, Z. Micahlewicz : *Evolutionary Computation* Control and Cybernetics, Special issue on Evolutionary computation , pp307-338, 1997

[Sebag & Schoenauer, 1996] M. Sebag & M. Schoenauer : *Contrôle d'un Algorithme Génétique,* Revue D'intelligence Artificielle, Volume N° 2-3, pp 389-428, 1996

[Shaw & al, 1997] W.M. Shaw, R. Burgin & P. Howell : *Performances Standards an Evaluation in IR test Collections : Cluster based Retrieval Models.* Information Processing and Management, 1-14, 1997

[Shutze & Pedersen, 1997] Schutze H., Pedersen J. : *A Cooccurrence- Based Thesaurus and two Applications to Information Retrival,* Information Processing & Management, 33(3) : pp 307-318, 1997

[Shutze & Silverstein, 1997] H. Schutze & J. C. Silverstein : *Projections for Efficient Document Clustering.* In the Proceedings of the 20th Annual Internation ACM SIGIR Conference on Research and Development in Information Retrieval, 74-81

[Singhal & al, 1994] A. Singhal, G. Salton , M. Mitra, C. Buckley : *Document Lenght Normalisation,* Rapport de recherche, 1995

[Sparck Jones & Needham, 1972] K.Sparck Jones & R. M Needham : *Automatic Theme Classification and Retrieval,* Information Processing and Management , Vol 4, pp 91-100, 1972

[Suy & Lang, 1994] I. Syu & S. D Lang : *A Competition-Based Connexionist Model for Information Retrieval,* Intelligent Multimedia Information Systems and Management (RIAO), New York, Vol 1. pp 248-265,1994

[Talbi, 1995] E.G. Talbi : *Régulation de Charge dans les Systèmes Distribués et Parallèles,* Journées de Recherche sur le placement dynamique et la répartition de charge : Application aux systèmes parallèles et distribués, p17-20, Paris, France, 1995

[Talbi, 1999] E.G. Talbi: *A taxonomy of Hybrid Metaheuristics* Journal of Combinatorial Optimisation pp 1-45, Kluwer Academic Publishers, 1999

[Tamine, 1997] L. Tamine : *Reformulation Automatique de Requête basée sur l'Algorithmique Génétique,* Actes du Congrès Inforsid, pp 643-662, Toulouse Juin 1997

[Tamine & Boughanem, 2000] L. Tamine, M. Boughanem : *Query Optimisation Using an Improved GA,* Conference on Information Knowledge and Management CIKM, Washington, November 2000

[Turtle & Croft, 1991] H. Turtle & W.B Croft : *Evaluation of an Inference Network Based Retrieval Model. ACM Transactions on Information Systems July 1991*

[Venturini, 1996] Venturini G. : *Algorithme Génétique et Apprentissage,* Revue d'Intelligence Artificielle, Volume 10, N° 2-3, pp 345-387, 1996

[Voorhees, 1998] E.M. Voorhees, *Variations in Relevance Judgements and the Measurement of Retrieval Effectiveness,* In the Proceedings of the 21th Annual International Conference on Research and Development in Information Retrieval (SIGIR), 1998

[Voorhees, 1999] E.M. Voorhees, *TREC Overview,* In Proceedings of the seventh Text Retrieval Conference TREC7, 1999

[Wang & al, 1985] Wang Y., Vandendorpe J,Evens, M., : *Relational Thsauri in Information Retrieval,* Journal of the Americn Society for Information Science, 36(1) : pp15-27, 1985

[Wilkinson, 1994] Wilkinson R. : *Effective Retrival of Structures Documents,* In Proceedings of the seventeeth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 311-317, 1994

[Wilkinson & Hingston, 1991] R. Wilkinson & P. Hingston : *Using The Cosine Measure in A Neural Network for Document Retrieval,* Conference on Research and Development in Information Retrieval (SIGIR), pp 202-210, Chicago (USA), 1991

[Wong & al, 1985] S.K.M. Wong, W. Ziarko & P.C. N. Wong : *Generalized Vector Space Model in Information Retrieval.* In Proc of the 8th ACM SIGIR Conference on Research and Development, New-York USA, 1985

[Xu & Croft, 1996] J. Xu & W.B. Croft : *Query Expansion Using Local and Global Document Analysis.* In Proc. ACM SIGIR Annual Conference on Research and Development, Zurich, 1996

[Yang & Korfhage, 1993] [Yang & Korfhage, 1993] J. J Yang & R. R Korfhage : *Query Optimisation in Information Retrieval Using Genetic Algorithms, ICGA'93*

[Yates & Neto, 1999] B. Yates & R. Neto : *Modern Information Retrieval,* ACM Press NewYork, Addison Wesley 1999

[Yu & Chen, 1985] C.T. Yu & C.H. Chen : *Adaptive Document Clustering*, Conference on Research and Development in Information Retrieval (SIGIR), pp 197-203, 1985

[Zadeh, 1965] L.A. Zadeh : *Fuzzy Sets*, Information Control, 8 : p 338-353, 1965

[Zobel & al, 1995] J. Zobel , A.Moffat & K. Ramamohanarao : *Inverted Files vs Signature Files for Text Indexing.* Technical Report CITRI/TR-95-5, Collaborative Information Technology Research Institute, Departement of Computer Science, Royal Melbourne Institute o technology, Australia, July 1995

MERCURE [Boughanem & Soule-Dupuy, 1997] (ModèlE de Réseau Connexioniste poUr la REcherche d'informations) est un SRI modélisé par un réseau connexioniste à trois couches interconnectées. Les requêtes, documents et termes sont représentés par des nœuds reliés entre eux. L'appariement requête-document est effectué par un processus de propagation de signaux.

1. Le modèle de base

Le modèle de base est décrit sur la figure 1. On y distingue trois couches : la couche d'entrée, la couche termes et la couches documents.

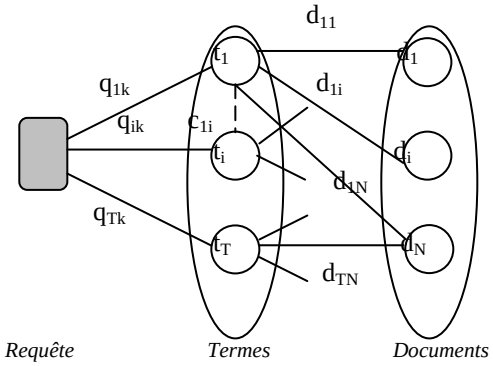


Figure1: Le modèle Mercure

- La couche d'entrée

Représente le besoin en informations d'un utilisateur. Ce besoin est exprimé au système à l'aide d'une requête en langage naturel, analysée par une procédure automatique qui consiste essentiellement à extraire les mots isolés, éliminer les mots vides par référence à un antidictionnaire puis enfin constituer la racine du mot par application de l'algorithme de Porter pour les textes en anglais et d'une troncature pour les textes en français, anglais et italien. Ceci conduit à la construction d'une liste de mots significatifs qui constitue la requête indexée.

- La couche termes

Cette couche représente l'ensemble des termes d'indexation utilisés pendant la phase de recherche.

- La couche documents

Cette couche représente l'ensemble des neurones documents gérés par le SRI. Le neurone document représente l'image en mémoire d'un descripteur de document.

Les différents nœuds sont reliés par des liens pondérés. Le poids d'un lien entre deux nœuds représente l'importance de l'interaction entre ces nœuds. On distingue trois types de liens : liens terme – document, terme – requête, terme – terme.

- Lien terme- terme

Le lien entre deux neurones termes représente une relation de cooccurrence. Le poids de ce lien est déterminé à partir de la distribution des termes dans les documents ; il est calculé selon la formule de Jaccard comme suit :

$$C_{ij}=\alpha*\frac{\sum_{k=1}^{N}(d_{ik}*d_{jk})}{\sqrt{\sum_{k=1}^{N}d_{ik}^{2}+\sum_{k=1}^{N}d_{jk}^{2}-\sum_{k=1}^{N}(d_{ik}*d_{jk})}}$$

Où :

d_{ik} : Poids du lien entre le terme t_i et document d_k

N : Nombre de documents dans la collection

- Lien terme - document

La performance d'un SRI est directement liée à la formulation utilisée pour mesurer l'importance des termes dans les documents. La formulation testée et retenue dans Mercure est la suivante :

$$d_{ij}=\frac{tf_{ij}*(h_1+h_2*\log(\frac{N}{n_i})}{h_3+h_4*\frac{dl_j}{\Delta d}+tf_{ij}}$$

Où :

h_i : Paramètre dépendant de la collection de documents

tf_{ij} : Fréquence du terme t_i dans le document d_j

N : Nombre de documents dans la collection

dl_j : Taille du document d_j en nombre de termes, sans les mots vides

Δd : Taille moyenne des documents de la collection

- Lien requête - terme

La formulation retenue pour la pondération de ce type de lien est la suivante :

$$q_{ui}^{(s)}=\begin{cases}\frac{nq*qtf}{nq-qtf} & si\ (nq>qtf)\\ qtf & \text{sinon}\end{cases}$$

Où :

qtf : Fréquence d'un terme dans une requête

nq : Nombre de termes dans la requête

$q_{ui}^{(s)}$: Poids du terme i dans la requête u à l'étape 0 de la recherche, s=0 car cette formule intervient à l'initialisation du proceesus d'évaluation de requête

2. Fonctionnalités

Mercure assure principalement les fonctions d'évaluations de requête et reformulation automatique de requête.

2.1. Evaluation de requête

La fonction d'évaluation mesure la similitude entre la requête à l'entrée du réseau et documents. Elle est mise en œuvre grâce à un processus de transfert des activations de la couche requête vers la couche documents. A la fin du processus, les nœuds sont classés par ordre décroissant de leurs valeurs d'activation et les documents associés sont présentés à l'utilisateur. Le processus est réalisé selon les étapes suivantes :

1. Indexation de la requête et représentation sous la forme :

$$Q_u^{(s)} = (q_{u1}^{(s)}, q_{u2}^{(s)}, \ldots, q_{uT}^{(s)})$$

Les poids des termes dans la requête sont affectés aux liens requête – termes

2. Déclenchement de l'évaluation à partir du nœud requête, en envoyant un signal de valeur *1* à travers les liens requête-termes.

3. Calcul d'une valeur d'entrée et valeur de sortie à chaque nœud :

$$In(t_i)=q_{ui}^{(s)} \qquad Out(t_i)=g(In(t_i))$$

4. Transmission des signaux vers la couche documents. Chaque nœud document calcule une entrée selon la formule :

$$In(d_j)=\sum_{i=1}^{T} Out(t_i)*d_{ij})$$

puis une valeur d'activation selon la formule :

$$Out(d_j)=g(In(d_j))$$

5. Tri des documents répondant à la requête selon l'ordre décroissant de leur valeur d'activation

2.2. Reformulation automatique de requête

Deux modes de reformulation automatique de requête sont proposés dans le système Mercure : reformulation directe et reformulation indirecte.

- Reformulation directe

Consiste à ajouter, en utilisant les liens de cooccurrence, de nouveaux termes à la requête initiale. Plus précisément, on ajoute les termes actifs, atteints par transfert d'activation à partir des termes de la requête, et ce à la première itération.

- Reformulation indirecte

La reformulation est dans ce cas basée sur les résultats de la recherche. Elle consiste essentiellement à injecter la pertinence dans le processus de recherche afin d'apprendre les liens requêtes- termes. A cet effet, deux stratégies sont mises en œuvre. La première est la rétropagation de la pertinence ; elle consiste essentiellement à rétropager les activations à partir d'une sortie désirée déterminée à partir du jugement de pertinence de l'utilisateur. Le processus de propagation inverse , de la couche documents vers la couche d'entrée, permet de corriger les poids des liens requêtes- termes. La seconde consiste en l'adaptation de l'algorithme de rétro-propagation du gradient.