typo lgocf@procnameProcédurelgocf@funcnameFonctionlanguagechoosenfrench

République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE Dr. TAHAR MOULAY SAIDA

FACULTE : TECHNOLOGIE

DEPARTEMENT : INFORMATIQUE

MEMOIRE DE MASTER

OPTION : Réseaux Informatique et Système Distribués (RISR)

Thème

Système de recommandation des livres à base des métaheuristiques

Présenté par :
MERAH Malika
BOUSAIDI Mahfoudh

Encadré par :
RAHMANI Mohamed El Hadi

Promotion : Juin 2020

Remerciements

En guise de reconnaissance, nous tenons à témoigner nos sincères remerciements à toutes les personnes qui ont, d'une quelconque manière, contribué à l'élaboration et la bonne réalisation de notre mémoire de Master .

Nous souhaitons avant tout remercier notre promoteur Dr. RAHMANI MOHAMED ELHADI pour avoir accepté de nous encadrer et nous proposer ce thème de recherche. Aussi, pour nous avoir guidé sur la globalité du travail .

Nous tenons à remercier tous les membres de notre comité de mémoire et tous les membres du jury pour avoir bien voulu donner de leur temps pour lire ce travail et faire partie des examinateurs.

Dans l'impossibilité de citer tous les noms, nos sincères remerciements vont à tous ceux et celles qui, de près ou de loin, ont permis, par leurs conseils, remarques et leurs compétences, la réalisation de cette thèse.

Enfin, nous n'oserons oublier de remercier tout le corps professionnel de l'Université de Dr. Moulay Tahar Saida pour le travail énorme qu'il effectue pour nous créer les conditions les plus favorables pour le déroulement de nos études.

Dédicaces

Je dédie ce travail a ceux qui m'ont encouragé et soutenu ;
A ma très chère maman d'amour ;
À la mémoire de mon père ;
A mon cher frère Mourad pour ses conseils,ses encouragements ;
A mes sœurs et frères ;
A mes neveux et nièces « Nadjib » et « nihel » et « serine » ;
A tous mes proches et ma belle soeur ;
A toute la famille Merah ;
A notre encadreur « Dr. Rahmani Mohamed Elhadi» ;
A tous le corps professoral et administratif de l'université Dr. Moulay Tahar de la wilaya de Saida ;
A tous nos amis et collègues qu'on a oublié de cité

Malika

Dédicaces

Toutes les lettres ne sauraient trouver les mots qu'il faut...
Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance...
C'est tout simplement que ce modeste travail soit dédié :
À mes chers parents ;
Vos conseils m'ont toujours guidé vers la réussite. Votre patience sans fin, votre compréhension et vos encouragements.
À ma famille ;
À mes proches ;
Qui n'ont cessé de me soutenir et de m'encourager durant toutes les années de mes études.
À tous mes chers amis et mes collègues de l'université Dr. Moulay Tahar ;
C'est une grande fierté d'être parmi vous.
A notre encadreur « Dr. Rahmani Mohamed Elhadi» ;
À tous ceux qui m'ont accompagné au long de mon parcour scolaire ;
Et à tous mes chers enseignants qui m'ont enseigné.

Mahfoudh

Table des matières

Table des figures

Liste des tableaux

Table des sigles et acronymes

RI	*Recherche d'Information*
IA	*L'Intelligence Artificielle*
FI	*Filtrage D'information*
ML	*Machine Learning*
MEV	*Modèle d'Espace Vectoriel*
TF	*Term Frequency*
IDF	*Inverse Document Frequency*
IMDb	*Internet Movie Database*
GRASP	*Greedy Randomized Adaptive Search Procedure*
NSGA	*non dominated sorting genetic algorithm*
SA	*Simulated Annealing*
SPEA	*Strength Pareto Evolutionary Algorithm*
MOO	*Multi-objective optimization*
EDA	*Estimation of Distribution Algorithm*
BOA	*Bayesian Optimization Algorithm*
ACS	*Adaptive Clonal Selection*
AINET	*Artificial Immune Network*
dDCA	*deterministic of Dendritic Cells Algorithm*
BAM	*Bidirectional Associative Memory*
LQV	*learning vector quantization*
CAM	*Content Addressable Memory*
ASSOM	*Adaptive-Subspace Self-Organizing Card*
ART	*Adaptive Resonance Theory*
EDI	*environnement de développement intégré*
XML	*Extensible Markup Language*
HTML	*Hypertext Markup Language*
IDE	*Integrated Development Environment*
JRE	*Java Runtime Environment*
JDK	*Java Development Kit*
BF	*Book Finder*

Système de recommandation des livres a base des métaheuristiques

Résumé *— Avec l'essor de web et l'arrivée des médias de masse nos sources d'informations ont été multiplié et diversifié. En conséquence, nous sommes obligés de traiter un grand nombre de données avant d'y arriver à notre but ou à l'information visé. Chaque jour, des livres sont publiés et rendus disponibles sur Internet. Toutefois, trouver le bon livre au bon moment peut devenir une tâche épuisante.*

Ce mémoire propose l'utilisation d'un système de recommandation de livres en utilisant la méthode de filtrage collaboratif basé sur les utilisateurs, nous avons conçu un système de recommandation de livres nommé BF (Book Finder). Le système vise à regrouper les utilisateurs similaires (qu'ils ont les mêmes intérêts) dans un même groupe. En se basant sur les notes données par ces utilisateurs, le système fait des prédictions des notes des livre qu'il ne sont pas encore visité par les nouveaux utilisateurs et il renvoie une liste de livres ordonnée selon les notes prévues.

L'objectif principal de BF est d'aider le chercheur à Choisir les livres qui correspondent le mieux à son style d'apprentissage et à ses champs d'intérêt. Ce problème amène à la définition formelle d'un problème d'optimisation qui est une variante d'un sous-problème du problème de sac à dos. Pour résoudre ce genre de problème, il est nécessaire d'utiliser une métaheuristique afin de tendre vers une bonne solution en un temps raisonnable. Nous présentons un algorithme basé sur le recuit simulé et un algorithme multi-objectif pour la résolution de ce problème (NSGA).

Mots clés : *Métaheuristique, Système de recommandation, filtrage collaborative, algorithme génétique, Récuit simulé .*

Book Recommender System Based on Metaheuristics

Abstract *— Nowadays, the rise of the web has brought us such a huge number of books, publications, and documents that students can hardly consider all of them. Finding the right book at the right time is an exhausting task, especially for new users who have diverse learning styles, needs, and interests. Moreover, the growing number of books in one subject can overwhelm Researchers trying to choose the right book.*
This thesis studies how recommendation systems can be applied to help overcome this challenge by ranking books using collaborative filtering methods based on users. we have designed and implemented a book recommendation system called Book Finder (BF). The system aims to group similar users (that they have the same interests) in the same group. Based on ratings given by users, the system predict ratings of books hasn't been visited yet by the new users and it returns a list of books ordered according to theirs predicted ratings.
BF aims to help learners select books which correspond to their fields of interest and which best match their learning styles. This problem leads to the formal definition of optimization problem which is a variant of the knapsack problem. To resolve this kind of problem, it is necessary to use a metaheuristic to tend toward a good solution in a reasonable time. We present an algorithm based on simulated annealing and a multi-objective algorithm (NSGA)to solve this problem.

Keywords : *metaheuristic, recommendation system, collaborative filtering, genetic algorithm, simulated annealing.*

Introduction

L'expansion technologique et l'arrivée des médias de masse ont diversifié et multiplié nos sources d'informations. En conséquence, nous sommes contraints de traiter un grand nombre de données avant d'y arriver à notre objectif ou à l'information recherchée. Par exemple, pour nous tenir au courant des événements qui nous entourent, il y a à notre disposition les nouvelles télévisées, les articles de journaux. Par ailleurs, lorsque nous avons besoin d'acheter un produit quelconque, nous nous retournons souvent vers les publicités télédiffusées, les panneaux publicitaires. Certains magasins mettent même à notre disposition des circulaires qui arrivent directement sur le pas de notre porte. Tous ces moyens de communication ont pour effet, entre autres, d'augmenter la quantité d'informations que nous avons à analyser avant de faire notre choix.

Pour pallier les problèmes de surcharge d'informations, la science informatique étudie les manières dont l'ordinateur peut assister l'homme pour l'aider à gérer le surdosage informationnel [LK92]. En particulier, ce domaine a développé des techniques de Recherche d'informations et de filtrage d 'Informations qui permettent la personnalisation de l'information selon les goûts, les besoins et les préférences de chaque individu. Plus récemment, les systèmes de recommandations [LK92],[Yam05],[LSY03],[Cha+02],ont été employés dans le but de ne proposer que l'information pouvant intéresser leurs utilisateurs, en omettant celles qui ne correspondent pas à leur profil.

Nous pensons que les systèmes de recommandations peuvent être très bénéfiques pour les chercheurs qui surfent le net à la recherche des informations pertinentes. Dans ce mémoire, nous proposons Book Finder un système de recommandations de livres à base des métaheuristiques.

Ce document est organisé comme suit :

Chapitre 1 nous passons en revue les systèmes de recommandations et leurs techniques, notamment le filtrage basé sur le contenu, le filtrage collaboratif et la méthode hybride, qui est basée sur l'application de ces deux techniques simultanément.

Chapitre 2 a pour objectif de définir le problème d'optimisation en utilisant des métaheuristiques. Ce chapitre présente une étude sur le domaine de l'heuristique et les métaheuristiques en donnant une définition de chaque algorithme.

Le chapitre 3 illustre, dans un premier temps l'architecture et les techniques appliquées dans BF. Nous commençons par une discussion du fonctionnement général du système pour ensuite voir en détail chacun de ses composants. Ensuite, nous analysons les résultats des tests effectués et la validation du système. Et en finira par l'implémentation.

Chapitre 1

Les systèmes de recommandation

1.1 Introduction

Avec L'essor du web et les évolutions technologiques, entre autres, La masse de données à exploiter ou à analyser est devenue très volumineuse. Il est devenu difficile de savoir quelles sont les données à rechercher et où les trouver.Des techniques informatiques ont été développées pour faciliter cette recherche ainsi que l'extraction des informations pertinentes.
Celle sur laquelle nous nous parlerons est la recommandation. Il s'agit de guider l'utilisateur dans son exploration des données afin qu'il trouve des informations pertinentes.
Un système de recommandation a pour objectif de fournir à un utilisateur des ressources pertinentes en fonction de ses préférences. Ce dernier voit ainsi réduit son temps de recherche, il peut être vus initialement comme une réponse donnée aux utilisateurs ayant des difficultés à prendre une décision dans le cadre d'utilisation d'un système de recherche d'information "classique".

1.2 Systèmes de recommandation

Sont des outils et techniques logiciels fournissant des suggestions d'items permettant de guider l'utilisateur vers des ressources intéressantes ou utiles au sein d'un espace de données important [Bur02][Bur+07].Les systèmes de recommandation sont essentiellement orientés vers les individus qui n'ont pas suffisamment d'expérience personnelle ou de compétences pour évaluer la quantité potentiellement immense d'informations qu'un site Web,par exemple, peut offrir [RV97].Le développement des systèmes de recommandation s'est initié à partir d'une observation assez simple : les individus s'appuient souvent sur les recommandations des autres pour la prise de décisions quotidiennes[MM09].Par exemple, il est commun de s'appuyer sur ce que nos semblables recommandent lors du choix d'un livre à lire ; pour la sélection de films à regarder, les individus tendent à lire et se fier aux critiques de film, etc. Aujourd'hui, le domaine d'application de ces systèmes de recommandation est très large. Ils font partie intégrante des sites de e-commerce. Un bon exemple de système de recommandation, est celui utilisé par le site Web Amazon. Le site a en effet accru significativement le nombre de ses visites et de ses ventes grâce à cet outil. Certains utilisateurs vont même maintenant sur ce site principalement dans le but de bénéficier de ses recommandations, même s'ils n'ont pas l'intention d'y acheter de produit. Et il en est de même pour des sites tels que Allociné pour les cinéphiles, Deezer pour la recommandation de musiques, Gameloft pour la recommandation de jeux vidéo, lasminute.com pour la recommandation de séjours touristiques ou encore Tripadvisor pour la recommandation d'hôtels et de restaurants.

Figure 1.1 montre un exemple d'un système de recommandation :

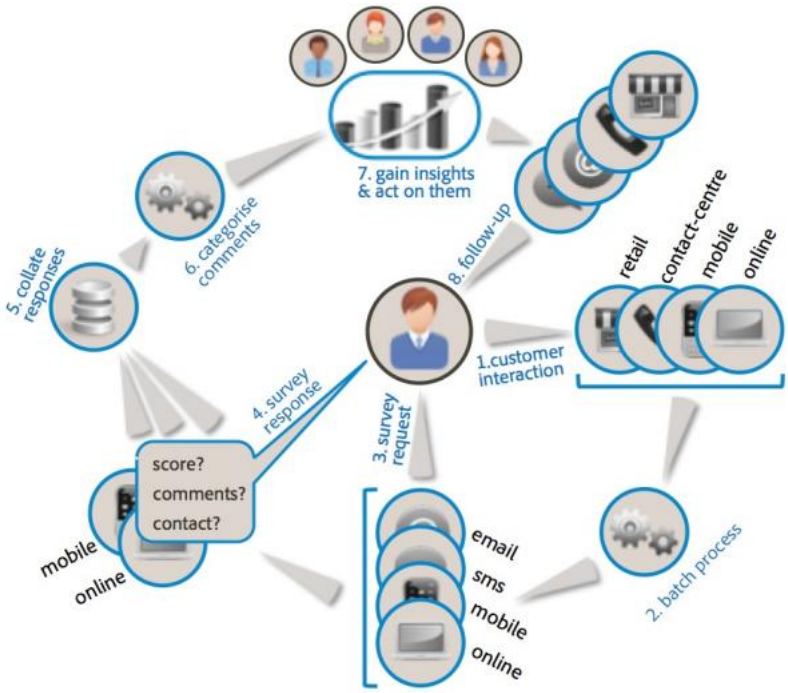

FIGURE 1.1 – exemple de système de recommandation

Les deux entités de base qui apparaissent dans tous les systèmes de recommandation sont l'item et l'utilisateur :
Item est le terme général utilisé pour dénoter ce que le système recommande aux utilisateurs. Afin de proposer des suggestions utiles et pertinentes pour un utilisateur en particulier, un système de recommandation va se focaliser sur un type spécifique d'item (par exemple, des livres, des images, etc.) et en conséquence, va personnaliser son modèle de navigation, son interface, ses techniques de recommandation, etc. Les utilisateurs doivent être modélisés dans le système de recommandation afin que le système puisse exploiter leurs profils et préférences. Par ailleurs, une description précise et claire des contenus est également nécessaire pour obtenir de bons résultats au moment des recommandations.

Les systèmes de recommandation ont prouvé ces dernières années qu'ils sont un bon moyen pour faire face au problème de surcharge cognitive. En effet pour résoudre ce problème, un système de recommandation met en avant des items inconnus qui peuvent être pertinents pour les utilisateurs. Ce niveau de pertinence est déterminé par le système en fonction de connaissances variées (profil de l'utilisateur, contexte de consommation, items disponibles, historique des transactions, feedbacks d'autres utilisateurs sur l'item, etc.). L'utilisateur peut alors parcourir les recommandations et peut fournir un feedback implicite ou explicite.
Par exemple, dans une plateforme e-learning un feedback peut être des notes ou des avis que les apprenants peuvent attribuer aux contenus. Toutes les actions et les feedbacks des utilisateurs peuvent être enregistrés dans la base de données du système de recommandations, et par la suite utilisés pour générer de nouvelles recommandations. Il est possible de classer les systèmes de recommandation de différentes manières.

1.3 Comment classer les déférents systèmes de recommandation ?

Plusieurs facteurs entrent en considération afin de catégoriser les systèmes de recommandation :

- *La connaissance de l'utilisateur (c-à-d. son profil en fonction de ses goûts).*
- *Le positionnement d'un utilisateur par rapport aux autres (la notion de classes ou réseaux d'utilisateurs).*
- *La connaissance des items à recommander.*
- *La connaissance des déférentes classes d'items à recommander.*

A partir de ces facteurs divers type de recommandations ont été produis dont les plus utilisées dans la littérature sont le filtrage basé sur le contenu et le filtrage collaboratif.

1.3.1 La recommandation basée sur le contenu :

Le système recommande des items qui sont similaires à ceux que l'utilisateur a aimés dans le passé. La similarité des items est calculée en se basant sur les caractéristiques associées aux items comparés. Par exemple, si l'utilisateur a noté positivement un film qui appartient au genre « action », alors le système peut fournir des recommandations de films de ce genre.

1.3.2 La recommandation par filtrage collaboratif :

Le filtrage collaboratif est la technique la plus populaire et la plus répandue dans les systèmes de recommandation. Le système demande aux utilisateurs d'évaluer des ressources, de sorte qu'il sache ce qu'ils aiment le plus. Puis, quand une recommandation est demandée pour l'utilisateur courant, lui seront alors proposées des ressources que des utilisateurs semblables a lui ont aimé.

1.3.3 Systèmes de recommandation basés sur le contenu

Les systèmes de recommandation ont pour objectif de fournir aux utilisateurs des ressources pertinentes issus d'un large espace d'options possibles en fonction de leurs préférences. Les systèmes de recommandation basés sur le contenu essayent de recommander des items similaires à ceux aimés par l'utilisateur dans le passé. En effet, le processus principal réalisé par un système de recommandation basé sur le contenu consiste à déterminer quel item coïncide le mieux avec les préférences de l'utilisateur en faisant correspondre les attributs d'un profil utilisateur avec les attributs des items. Pour un livre par exemple, on peut utiliser le genre, le nom des auteurs, l'éditeur ou toute autre information relative au livre, puis stocker ces caractéristiques (dans une base de données par exemple). Le profil de l'utilisateur est exprimé sous forme d'une liste d'intérêts basée sur les mêmes caractéristiques. La recherche sur les systèmes

de recommandation basés sur le contenu prend place à l'intersection de différents domaines de recherche en informatique, notamment la Recherche d'Information (RI) et l'Intelligence Artificielle (IA) [BYRN+99].

Figure 1.2 présente l'rchitecture d'un système de recommandation à base du contenu :

FIGURE 1.2 – Architecture d'un système de recommandation à base du contenu

En Recherche d'Information, l'utilisateur exprime un bsoin ponctuel en donnant une requête (habituellement une liste de mots-clefs). Dans les systèmes de filtrage d'information (FI), le besoin est représenté par le profil de l'utilisateur. Les items à recommander peuvent être très différents, en fonction du nombre et du type des attributs utilisés pour les décrire. En intelligence artificielle, la tâche de recommandation exploite la connaissance passée des utilisateurs. De manière simple, les profils des utilisateurs sont sous la forme de vecteurs de mots-clefs et reflètent les intérêts à long terme de l'utilisateur. Souvent, il est préférable que le système apprenne lui-même le profil de l'utilisateur plutôt que d'imposer à ce dernier de le fournir. Cela implique généralement l'application de techniques de Machine Learning (ML). Leur but est d'apprendre à catégoriser de nouvelles informations en se basant sur les informations précédemment vues, et qui ont été libellées implicitement ou explicitement comme intéressant ou non par l'utilisateur. Avec ces libellés, les méthodes de Machine Learning sont capables de générer un modèle prédictif qui, étant donné un nouvel item, va aider à décider le degré d'intérêt que peut porter l'utilisateur pour l'item.

Techniques de représentation et de recommandation :

recommandations basées sur les vecteurs de mots-clefs

La plupart des systèmes de recommandation utilisent de simples modèles de recherche, comme la correspondance de mots-clefs ou le Modèle d'Espace Vectoriel (MEV) avec la pondération basique TF-IDF (Term Frequency-Inverse Document Frequency). Dans ce modèle (MEV), chaque document est représenté par un vecteur de dimension n, où chaque dimension correspond à un terme de l'ensemble du vocabulaire d'une collection de documents.
Formellement, tout document est représenté par un vecteur poids sur des termes, où chaque poids indique le degré d'association entre le document et le terme.
Soit $D = (d_1, d_2, ...d_n)$ dénotant un ensemble de documents ou corpus, et $T = (t_1, t_2, ...t_n)$ le dictionnaire, ou l'ensemble des mots du corpus . T est obtenu en appliquant des opérations de traitement du langage naturel, comme l'atomisation (tokenization), l'élimination des mots vides de sens, et la troncature (stemming) [BYRN+99]. Chaque document d_j est représenté

par un vecteur dans un espace vectoriel à n dimensions,
tel que, $d_j = (w_{1j}, w_{2j}, ...w_{nj})$ *où* w_{kj} *le poids du terme* t_k *dans le document* d_j*. La représentation de documents dans le MEV fait apparaître deux difficultés : la pondération des termes et la mesure de similarité des vecteurs caractéristiques. Le schéma de pondération de terme le plus communément utilisé est la pondération TF-IDF (Term Frequency-Inverse Document Frequency) basée sur des observations empiriques sur le texte [Sal89] :*

- *Les termes rares ne sont pas moins pertinents que les termes fréquents (IDF).*
- *Des occurrences multiples d'un terme dans un document ne sont pas moins pertinentes que de simples occurrences (TF).*
- *Des documents longs ne sont pas préférables à des documents courts (normalisation).*

En d'autres mots, les termes qui apparaissent fréquemment dans un document, mais rarement dans le reste du corpus ont plus de chance de représenter le sujet du document. De plus, la normalisation des vecteurs résultats empêche les documents trop longs d'avoir plus de chance d'être retrouvés. Ces conjectures sont bien illustrées par la fonction TF-IDF :
ÉQUATION 1 :

$$TFIDF(t_k, d_j) = TF(t_k, d_j).\log\frac{N}{n_k}$$

où N dénote le nombre de documents dans le corpus, et n_k *représente le nombre de documents de la collection dans lesquels le terme* t_k *apparait au moins une fois, avec :*
ÉQUATION 2 :

$$TF(t_k, d_j) = \frac{f_k, j}{max_z f_{z,j}}$$

Où le maximum est calculé sur la fréquence f_{zj} *de tous les termes* t_z *qui apparaissent dans le document* d_j *. Pour que les poids soient dans l'intervalle [0,1] et que les documents soient représentés par des vecteurs de même longueur, les poids obtenus par la fonction TFIDF sont généralement normalisés par la normalisation cosinus (Équation 3).*

ÉQUATION 3 :

$$W_k, j = \frac{TFIDF(t_k, d_j)}{\sqrt{\sum_{S=1}^{|T|} TFIDF(t_k, d_j)^2}}$$

Après la pondération des termes, il faut définir une mesure de similarité des vecteurs caractéristiques. Cette mesure de similarité est requise pour déterminer la proximité entre deux documents. Il existe de nombreuses mesures de similarité, mais la plus largement utilisée est la similarité cosinus :

ÉQUATION 4 :

$$Sim(d_i, d_j) = \frac{\sum_k w_{ki}.w_{Kj}}{\sqrt{\sum_k w_{ki}^2 . \sqrt{\sum_k w_{ki}^2}}}$$

Dans les systèmes de recommandation basés sur le contenu s'appuyant sur le modèle d'espace vectoriel, les profils des utilisateurs et les items sont représentés comme des vecteurs de termes pondérés. La prédiction de l'intérêt d'un utilisateur sur un item donné peut être effectuée par calcul de similarité cosinus entre le vecteur de profil et le vecteur de l'item.
De nombreux systèmes de recommandation basés sur les mots-clefs ont été développés en très

peu de temps dans de multiples domaines d'applications, comme la musique, les films, etc. Des exemples des systèmes populaires Dans le domaine des systèmes de recommandation Web sont Letizia et Personal WebWatcher. Letizia [Lie95] est implémenté comme une extension de navigateur Web traquant le comportement de l'utilisateur et construit un modèle personnalisé constitué des mots-clefs liés aux intérêts de l'utilisateur. Il s'appuie sur un feedback implicite pour inférer des préférences de l'utilisateur. Par exemple, ajouter une page en favoris est interprété comme une preuve évidente de l'intérêt de l'utilisateur pour cette page. De manière similaire, Personal WebWatcher [MG99] apprend les intérêts des utilisateurs à partir des pages Web qu'ils visitent, et à partir des documents qui ont un lien hypermédia avec les pages visitées. Il traite les documents visités comme des exemples positifs d'intérêts utilisateurs, et les documents non visités comme des exemples négatifs. Dans le domaine du filtrage des actualités, des systèmes de recommandation notables sont NewT et NewsDude. NewT (NewsTailor) [SM93] permet aux utilisateurs de fournir un feedback positif ou négatif sur des articles, des parties d'articles, des auteurs ou des sources. Plusieurs agents de filtrage sont formés pour différents types d'information, par exemple, un pour les actualités politiques, un pour le sport, etc. Apprendre des profils à long terme et des profils à court terme est typique des systèmes de filtrage d'actualités. NewsDude [BP99] construit un modèle utilisateur à court terme basé sur TF-IDF, et un modèle à long terme basé sur un classificateur Bayésien naïf en s'appuyant sur un ensemble initial d'articles intéressants fournis par l'utilisateur. La source des actualités utilisée est Yahoo ! News.

Une variété de systèmes basés sur le contenu existe dans d'autres domaines d'application. INTIMATE [MKP03] recommande des films en utilisant des techniques de catégorisation de texte sur les synopsis de film obtenus sur IMDb (Internet Movie Database). Pour avoir des recommandations, l'utilisateur doit noter un certain nombre de films dans six catégories : horrible, mauvais, au-dessous de la moyenne, au-dessus de la moyenne, bon, excellent. De la même façon, Movies2GO [Muk+01] apprend les préférences des utilisateurs à partir des synopsis des films notés par l'utilisateur. Ce système intègre des schémas de votes [SJ80], conçus à l'origine pour permettre à des individus ayant des préférences conflictuelles d'arriver à un compromis acceptable. Ces schémas de votes sont adaptés ici pour gérer les préférences conflictuelles au sein d'un même profil utilisateur. De l'analyse des principaux systèmes développés ces 15 dernières années, le plus important à retenir est que la représentation par mots-clefs à la fois pour les items et pour les profils peut donner des résultats précis. Toutefois, pour cela, il est nécessaire qu'un nombre suffisant de preuves d'intérêts des utilisateurs soit disponible. La plupart des systèmes basés sur le contenu sont conçus comme des classificateurs de textes construits à partir d'un ensemble de documents d'apprentissage qui sont soit des exemples positifs, soit des exemples négatifs des intérêts de l'utilisateur. Le problème avec cette approche est le « manque d'intelligence ». Lorsque des caractéristiques plus complexes sont nécessaires, les approches à base de mots-clefs montrent leurs limites. Si l'utilisateur, par exemple, aime « l'impressionnisme français », les approches à base de mots-clefs chercheront seulement des documents dans lesquels les mots « français » et « impressionnisme » apparaissent. Des documents concernant Claude Monet ou Renoir n'apparaîtront pas dans l'ensemble des recommandations, même s'ils sont susceptibles d'être pertinents pour l'utilisateur. Des stratégies de représentation plus avancées sont nécessaires pour que les systèmes de recommandation basés sur le contenu prennent en compte la sémantique associée aux mots.

recommandations basées sur la sémantique

Les systèmes basés sur la sémantique sont un cas particulier des systèmes basés sur le contenu. Ils tendent à intégrer les nouvelles technologies du Web sémantique, afin de remédier à certains manques des systèmes basés sur le contenu classique : SiteIF a été le premier système à adopter une représentation basée sur le sens des documents pour construire un modèle des intérêts de l'utilisateur [MS01]. SiteIF est un agent personnel pour un site Web de nouvelles multilingues. La source externe de connaissance impliquée dans le processus de représentation est MultiWordNet, une base de données lexicale multilingue. Chaque news est automatiquement associé à une liste d'ensembles de synonymes (dits synsets) de MultiWordNet en utilisant Word Domain Disambiguation [MS00]. Le profil utilisateur est un réseau sémantique où les noeuds représentent les synsets trouvés dans les documents lus par l'utilisateur. Durant la phase de correspondance, le système reçoit en entrée la représentation sous forme de synsets d'un document et le modèle utilisateur courant, et il produit en sortie une estimation de la pertinence du document en utilisant la technique Semantic Network Value Technique [SS98].

1.3.4 Systèmes de recommandation basés sur une approche collaborative

Les méthodes de filtrage collaboratif produisent des recommandations en calculant la similarité entre les préférences d'un utilisateur et celles d'autres utilisateurs, De tels systèmes ne tentent pas d'analyser ou de comprendre le contenu des items à recommander. À la différence des approches basées sur le contenu, qui utilisent les items précédemment notés par un seul utilisateur u, les approches de filtrage collaboratif s'appuient sur les notes de tous les utilisateurs du système. L'idée clef est que la note de u pour un nouvel item i est susceptible d'être similaire à celle donnée par un autre utilisateur v, si u et v ont noté d'autres items d'une manière similaire. De même, u est susceptible de noter deux items i et j de la même façon, si d'autres utilisateurs ont donné des notes similaires à ces deux items. Les approches collaboratives dépassent certaines limitations des approches basées sur le contenu. Par exemple, des items dont le contenu n'est pas défini, ou difficilement définissable peuvent quand même être recommandés aux utilisateurs grâce aux feedbacks des autres utilisateurs. De plus, les recommandations collaboratives sont basées sur la qualité des items évaluée par les utilisateurs, au lieu de s'appuyer sur le contenu qui peut être un mauvais indicateur de qualité. Enfin, au contraire des systèmes basés sur le contenu, le filtrage collaboratif peut recommander des items avec des contenus différents, tant que les autres utilisateurs manifestent leurs intérêts pour ces différents items. Selon [BKV07] [Aci+07] [DG+08] [CBC08], les méthodes collaboratives peuvent être groupées en deux classes générales : les méthodes de voisinages et les méthodes basées sur un modèle.

Figure 1.3 présente l'rchitecture d'un système de recommandation par approche collaborative :

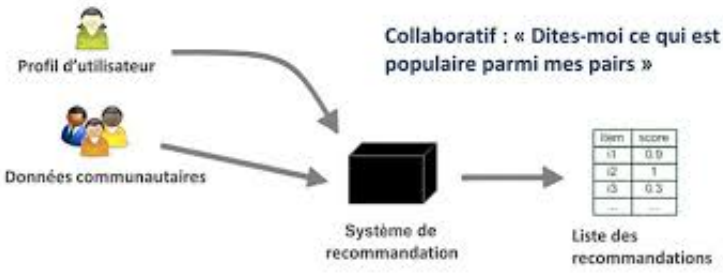

FIGURE 1.3 – Architecture d'un système de recommandation par approche collaborative

recommandations basées sur voisinage

Les systèmes de recommandation basés sur le voisinage automatisent le principe de bouche-à-oreille, où l'on se fonde sur l'avis de personnes partageant les mêmes idées ou d'autres sources fiables pour évaluer la valeur d'un item (film, livre, article, album, etc.), selon nos propres préférences. Ainsi, dans le filtrage basé sur le voisinage, les notes des utilisateurs stockées par le système sont directement utilisées pour prédire les notes pour de nouveaux items. Cela peut être fait de deux manières connues sous le terme de recommandations basées sur les utilisateurs ou recommandations basées sur les items.

recommandations basées sur le voisinage utilisateur

Les systèmes basés sur le voisinage utilisateur, comme GroupLens [Kon+97], Bellcore video [Hil+95], et Ringo [SM95], évaluent l'intérêt d'un utilisateur u pour un item i en utilisant les notes de cet item. Ces notes sont données par d'autres utilisateurs, appelés voisins, qui ont des habitudes de notation similaires. Les voisins d'un utilisateur u sont typiquement les utilisateurs v dont les notes sur les items sont les plus proches de celles de u sur ces items. Les plus proches voisins sont les utilisateurs les plus similaires à u dans leur notation. Si on suppose que l'on a une mesure de similarité w_{uv}, pour tout utilisateur $v \neq u$, les K plus proches voisins de u, notés $N(u)$, sont les K utilisateurs v avec la plus grande mesure de similarité w_{uv} par rapport à u. Seuls les utilisateurs ayant noté l'item i peuvent être utilisés pour la prédiction. L'ensemble de ces voisins est noté $N_i(u)$. La note $\acute{r}_{ui}$ de l'utilisateur u sur l'item i peut être prédite par la moyenne des notes r_{vi} de ces voisins :

ÉQUATION 5 :

$$\acute{r}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} r_{vi}$$

Un problème avec cette méthode par moyenne est qu'elle ne prend pas en compte le fait que les voisins peuvent avoir des niveaux différents de similarité. En effet, on peut prédire que la note de l'utilisateur u est plus à même de se rapprocher de la note des voisins avec une plus grande similarité. Ainsi, une solution à ce problème est de pondérer la contribution de chaque voisin par leur similarité à u . Cependant, si la somme de ces poids ne fait pas 1, les notes prédites peuvent être en dehors des valeurs autorisées. De ce fait, il est courant de normaliser

ces poids de telle sorte que la note prédite devienne :
ÉQUATION 6 :

$$\acute{r}_{ui} = \frac{\sum_{v \in N_i(u)} w_{uv} r_{vi}}{\sum_{v \in N_i(u)} |w_{uv}|}$$

Dans [BH98], W_{uv} est remplacé par $w_{uv}\alpha$, où $\alpha > 0$ est un coefficient amplificateur. Quand $\alpha > 0$, une plus grande importance est donnée aux voisins qui sont les plus proches de u . Un important défaut de cette équation est qu'elle ne considère pas le fait que des utilisateurs peuvent utiliser des notes différentes pour quantifier le même niveau d'appréciation sur un item. Ce problème est habituellement résolu en convertissant les notes des utilisateurs r_{vi} en notes normalisées $h(r_{vi})$ [BH98] [Res+] :
ÉQUATION 7 :

$$\acute{r}_{ui} = h^{-1}\left(\frac{\sum_{v \in N_i(u)} w_{uv} h(r_{vi})}{\sum_{v \in N_i(u)} |w_{uv}|}\right)$$

Ces méthodes de prédiction basées sur une moyenne de notes du voisinage résolvent essentiellement un problème de régression. Une autre orientation est la classification. Cela consiste à rechercher la note la plus probable que donnerait un utilisateur u à un item i, en prenant la valeur donnée le plus souvent par les plus proches voisins de u sur cet item et en considérant leur similarité avec u. Le vote v_{ir} donné par les k plus proches voisins de u ayant donné une note $r \in s$ peut être obtenu par la somme des valeurs de similarité des voisins qui ont donné cette note à i :

ÉQUATION 8 :

$$v_{ir} = \sum_{v \in N_i(u)} \delta(r_{vi} = r) w_{uv}$$

où $\delta(r_{vi} = r)$ est 1 si $r_{vi} = r$, et 0 sinon. Une fois que cela a été calculé pour chaque note possible, la note prédite est simplement la valeur r pour laquelle v_{ir} est le plus grand. Une méthode de classification qui considère des notes normalisées peut aussi être définie. Soit, ś *l'ensemble des valeurs normalisées possibles, la note prédite est :*

ÉQUATION 9 :

$$\acute{r}_{ui} = h^{-1}\Big(\arg\max_{r \in \acute{s}} \sum_{v \in N_i(u)} \delta(h(r_{vi}) = r) w_{uv}\Big)$$

Recommendation basée sur le voisinage item

Alors que les méthodes basées sur le voisinage utilisateur s'appuient sur l'avis d'utilisateurs partageant les mêmes idées pour prédire une note, les approches basées sur les items [LSY03][DK04] [LB10] prédisent la note d'un utilisateur u pour un item i en se basant sur les notes de u pour des items similaires à i. Dans de telles approches, deux items sont similaires si plusieurs utilisateurs du système les ont notés d'une manière similaire. Cette idée peut être formalisée comme suit. Soit $N_u(i)$ les items notés par un utilisateur u qui sont similaires à un item i. La note prédite de u pour i peut être obtenue par la moyenne pondérée des notes données par u aux items de $N_u(i)$:

ÉQUATION 10 :

$$\acute{\mathrm{r}}_{ui} = \frac{\sum_{j \in N_u(i)} w_{ij} r_{ui}}{\sum_{j \in N_u(i)} |w_{ij}|}$$

Comme pour la recommandation basée sur le voisinage utilisateur, les différences d'échelle de notation des utilisateurs peuvent être considérées en normalisant les notes avec une fonction h :
ÉQUATION 11 :

$$\acute{\mathrm{r}}_{ui} = h^{-1}\left(\frac{\sum_{j \in N_u(i)} w_{ij} h(r_{uj})}{\sum_{j \in N_u(i)} |w_{ij}|}\right)$$

De plus, il est aussi possible d'utiliser une approche par classification. Dans ce cas, les items j notés par l'utilisateur u votent pour la note à donner à un nouvel item i, et ces votes sont pondérés par la similarité entre j et i . La version normalisée de cette approche peut être exprimée comme suit :

ÉQUATION 12 :

$$\acute{\mathrm{r}}_{ui} = h^{-1}\left(\arg\max_{r \in \acute{\mathrm{s}}} \sum_{j \in N_u(i)} \delta(h(r_{uj}) = r) w_{ij}\right)$$

recommandations basées sur un modèle prédictif

A la différence des systèmes basés sur le voisinage qui utilisent les notes stockées pour les prédictions, les approches basées sur un modèle utilisent ces notes pour construire un modèle prédictif par apprentissage. L'idée générale est de modéliser les interactions utilisateur-item avec des facteurs représentant des caractéristiques latentes des utilisateurs et items dans le système, comme des classes d'utilisateurs et d'items. Ce modèle est ensuite qualifié à partir des données disponibles, et utilisé plus tard pour prédire les notes des utilisateurs pour de nouveaux items. Les approches basées sur un modèle sont nombreuses, elles incluent le clustering bayésien [BH98], l'analyse de la sémantique latente [ATH03], l'allocation de Dirichlet latente [BNJ03], l'entropie maximum [ZK04], les machines de Boltzmann [SMH07], les machines à support vectoriel [Grč+06], et la décomposition en Valeur singulière[BKV07] [Kor08][Pat07].Ces systèmes transforment à la fois les items et les utilisateurs dans un espace de facteurs latent, permettant de les comparer directement. L'espace latent cherche à expliquer les notes en caractérisant les produits et les utilisateurs sur des facteurs automatiquement déduits des feedbacks utilisateurs. Par exemple, pour des films, les facteurs peuvent mesurer des dimensions évidentes comme la quantité d'action, ou l'orientation pour un certain public ; des dimensions moins bien définies telles que la profondeur du développement des personnages ; ou des dimensions complètement ininterprétables. Pour les utilisateurs, chaque valeur de facteur mesure à quel point l'utilisateur aime les films sur ce facteur donné. Les systèmes de recommandation basés sur des modèles de facteurs latents offrent des capacités expressives hautes pour décrire les divers aspects des données. Ainsi, ils tendent à fournir des résultats plus précis que les systèmes basés sur le voisinage. Cependant, la grande partie de la littérature et des systèmes commerciaux (par exemple, Amazon [LSY03], TiVo [AVS04] et Netflix sont basés sur le voisinage. Actuellement, il existe beaucoup plus de systèmes de recommandation basés sur le voisinage, car ils sont considérés comme plus faciles et intuitifs

à manipuler. Tout d'abord, ils fournissent naturellement des explications plus intuitives du raisonnement derrière les recommandations, ce qui améliore l'expérience utilisateur. Enfin, ils peuvent immédiatement délivrer des recommandations à l'utilisateur en se basant sur les feedbacks qu'il vient juste de fournir.

1.4 Avantages et inconvénients des approches de recommandation

1.4.1 Cross-genre niches

Le filtrage collaboratif a comme capacité de recommander à un utilisateur ce qui est hors du familier. Par exemple, un utilisateur qui a des voisins similaires du point de vue des sports peut se voir recommander des recettes de cuisine si ces voisins aiment la cuisine et les recettes, même si cet utilisateur n'a jamais exprimé ce genre de favoris.

1.4.2 Connaissance du domaine

La connaissance du domaine n'est pas requise, le processus de recommandation se base uniquement sur les évaluations des items.

1.4.3 Adaptabilité

Au fur et à mesure que la base de données des évaluations augmente, la recommandation devient plus précise.

1.4.4 Problème de démarrage à froid, cas du nouvel utilisateur

Un nouvel utilisateur qui n'a pas encore accumuler suffisamment d'évaluations ne peut pas avoir de recommandations pertinentes.

1.4.5 Problème de démarrage à froid, cas du nouvel item

C'est un problème qui concerne le filtrage collaboratif. L'item doit avoir suffisamment d'évaluations pour qu'il soit pris en considération dans le processus de recommandation.

1.4.6 Problème de démarrage à froid, cas du système débutant

Le cas du système débutant provient lors du lancement d'un nouveau service de recommandation. Le système ne possède alors aucune information sur les utilisateurs et sur les items. Les méthodes de filtrage collaboratif ne peuvent pas fonctionner sur une matrice vide. La solution consiste en général à trouver des informations descriptives des items afin d'organiser le catalogue et inciter les utilisateurs a le parcourir jusqu'à ce que la matrice soit assez remplie et permette de passer en mode collaboratif

1.4.7 Le gray Sheep

Quand des utilisateurs ont des gouts atypiques ils n'auront pas beaucoup d'utilisateurs en tant que voisins. Cela mènera à des recommandations pauvres. Ce problème est également connu comme gray sheep.

1.4.8 Le shilling

C'est l'action malveillante d'influencer la recommandation, en créant de faux profils pour voter et favoriser/défavoriser certains items.

Le tableau 1.1, résume les avantages et les inconvénients des méthodes traditionnelles de recommandations.

TABLE 1.1 – Les avantages et les inconvénients des méthodes traditionnelles de recommandations

Techniques de recommendation	Avantages	Inconvénients
Recommandation basée sur le contenu	— Connaissance du domaine. — Adaptabilité.	— Problème de démarrage à froid, cas du nouvel utilisateur. — Le gray Sheep. — Le shilling.
Recommendation par filtrage collaboratif	— Cross-genre niches. — Connaissance du domaine. — Adaptabilité.	— Problème de démarrage à froid, cas du nouvel item. — Problème de démarrage à froid, cas du système débutant. — Le gray Sheep. — Le shilling.

1.5 Approches hybrides

Le tableau comparatif ci-dessus nous montre que chaque technique de recommandation a des apports mais aussi des limites. Afin de combler les faiblesses de ces techniques, plusieurs travaux de recherche ont proposé de combiner ou d'hybrider des techniques de recommandation.

1.6 Conclusion

Les systèmes de recommandation proposent une solution au problème de surcharge d'information par proposition de recommandations d'items. Le problème de recommandation d'items à un utilisateur a été formalisée et les différents types de systèmes de recommandation et leurs méthodes ont été exposées. Nous avons identifié deux catégories de systèmes de recommandation : les systèmes de recommandation basés sur le contenu et les systèmes de recommandation basés sur le filtrage collaboratif. Ces deux approches présentent néanmoins des caractéristiques complémentaires. Par conséquent les nouveaux travaux s'intéressent à différentes techniques d'hybridation, qui en plus de profiter des avantages respectifs de ces approches, s'avèrent fournir des recommandations plus précises. Dans ce memoire, nous avons focalisé sur le système de recommandation basés sur le filtrage collaboratif afin d'adapter des techniques de méta heuristiques pour créer un système de recommandation éfficace. Dans le chapitre suivant, nous détaillerons les métaheuristiques.

Chapitre 2

Métaheuristique

2.1 Introduction

Deux catégories de méthodes se présentent à nous pour la résolution des problèmes d'optimisation combinatoire : les méthodes dites exactes et les méthodes approchées. Malgré leur efficacité, les premières coûtent cher en temps et espace mémoire des ordinateurs. Alors, que si les secondes sont plus économiques, elles sont moins efficaces que les premières. Les métaheuristiques sont un moyen médian pour équilibrer entre le temps nécessaire à l'exécution du programme concerné et la qualité de la solution obtenue. Dans Ce chapitre on fournit une vue sur l'ensemble du domaine des métaheuristiques. En premier lieu nous introduisons brièvement quelques notions des métaheuristiques. Ensuite nous présentons sept grandes familles d'approches heuristiques : Les algorithmes stochastiques, évolutionnistes, physiques, probabilistes, Swarm, immunitaires et les Algorithmes neuronaux, tout en citant Les avantages et les inconvénients de chacun d'eux.

2.2 Qu'est-ce que l'IA ?

L'intelligence artificielle est avant tout un programme informatique visant à effectuer, au moins aussi bien que des humains, des tâches nécessitant un certain niveau d'intelligence. Les promesses non tenues des débuts de l'IA ont amené à distinguer d'une part les machines qui non seulement mettraient en oeuvre des raisonnements semblables aux raisonnements humains, mais auraient également une réelle conscience d'elles mêmes : c'est ce qu'on appelle l'intelligence artificielle forte ; d'autre part les machines qui rendent de nombreux services aux humains en simulant l'intelligence humaine : c'est l'intelligence artificielle faible.

2.3 Métaheuristique

L'heuristique est une méthode d'optimisation qui a pour but de trouver une « bonne » solution à un problème en un temps raisonnable, mais sans garantie sur la validité ou l'optimalité de la solution ainsi fournie [BS+12]. Parmi les heuristiques certaines sont adaptables à un grand nombre de problèmes différents sans changements majeurs dans l'algorithme, on parle alors de méta-heuristiques. Le qualificatif "méta » fait référence à une combinaison de un ou plusieurs heuristiques. Avant de pouvoir être appliquée à la résolution d'un problème particulier, quelques transformations (mineures en général) sont nécessaires, Pour lesquels elles s'adapteront avec plus ou moins de facilité à chaque problème[BS+12]. Les Métaheuristiques contient un composant d'exploration (aussi appelée « diversification ») qui permet de rechercher de nouvelles solutions dans l'espace de recherche, et un composant d'exploitation

(également appelée "intensification") qui utilise les résultats obtenus lors de la phase d'exploration, afin de sélectionner le sous-espace de recherche le plus prometteur, et de "plonger" vers l'optimum local le plus proche [MBF11]. Les métaheuristiques sont apparues dans les années 1980 et forment une famille d'algorithmes d'optimisation visant à résoudre des problèmes d'optimisation difficile. Durant les vingt dernières années, les métaheuristiques ont reçu un intérêt grandissant et ont montré leur efficacité dans de vastes domaines d'application en résolvant de nombreux problèmes d'optimisation, ce sont généralement des problèmes des données incomplètes, ou capacité de calcul limitée. Plusieurs d'entre elles sont souvent inspirées par des systèmes naturels dans de nombreux domaines tels que : la biologie (algorithmes évolutionnaires et génétiques), *l'éthologie*(algorithmes de colonies de fourmis) *et aussi la physique* (recuit simulé)*[Tal09].*

2.4 Les algorithmes stochastiques

Les algorithmes stochastiques sont des techniques de simulation numériques de chaines de Markov, visant à résoudre des problèmes d'optimisation ou d'estimation complexes. A la déférence de leurs homologues déterministes, ces méthodes de recherche aléatoire permettent d'explorer des espaces de grandes dimensions, tout en évitant certains pièges, tels des puits de minima locaux en optimisation globale. Parmi les algorithmes stochastiques on cite :

2.4.1 Recherche aléatoire

Recherche aléatoire appartient aux domaines de l'optimisation stochastique et de l'optimisation globale, c'est une méthode de recherche directe car elle ne nécessite pas de dérivés pour rechercher un domaine continu. le principe de base de cet algorithme est d'échantillonner des solutions de l'ensemble de l'espace de recherche en utilisant une distribution de probabilité uniforme. Chaque échantillon futur est indépendant des échantillons qui le précèdent [Kar63].

2.4.2 Recherche aléatoire adaptive

Recherche aléatoire adaptive appartient à l'ensemble général d'approches connues sous le nom d'optimisation stochastique et d'optimisation globale. Recherche aléatoire adaptive appartient a été conçu pour remédier aux limites de taille pas fixe dans l'algorithme de recherche aléatoire localisée [KW71]. L'idee generale de cet algorithme est de Tester une étape plus grande à chaque itération et adopter l'étape plus grande si elle aboutit à un résultat amélioré. De très grandes tailles sont testées de la même manière mais avec une fréquence beaucoup plus faible.

2.4.3 Algorithme de descente

Algorithme de descente (Hill Climbing) Est un algorithme d'optimisation stochastique et d'optimisation locale (contrairement à l'optimisation globale). Il s'agit d'une technique de recherche directe, car elle ne nécessite pas de dérivées de l'espace de recherche. La méthode est une généralisation de la méthode de la descente de gradient elle consiste à partir d'une solution S à choisir une solution $\acute{S}$ dans un voisinage de S . La nouvelle solution choisie est

meilleure que la précédente sous la fonction objective. Cela nous permet d'explorer l'espace d'une manière itérative jusqu'à convergence à un optimum local[WJ94].

2.4.4 Recherche local itérée

Recherche local itérée algorithme est une technique de métaheuristique et d'optimisation globale. Elle explore une séquence de solutions créées sous forme de perturbations de la meilleure solution actuelle, dont le résultat est affiné à l'aide d'une heuristique intégrée [Lou01].

2.4.5 Recherche local guidée

Recherche local guidée est un algorithme métaheuristiques dont le but est d'aider la recherche locale pour échapper à des optimums locaux. Un algorithme de recherche locale est exécuté jusqu'à ce qu'il se coince dans un optima local. Les caractéristiques des optima locaux sont évaluées et pénalisées, dont les résultats sont utilisés dans une fonction de coût augmenté utilisée par la procédure de recherche locale. La recherche locale est répétée plusieurs fois en utilisant le dernier optima local découvert et la fonction de coût augmenté qui guide l'exploration loin des solutions avec des fonctionnalités présentes dans les optima locaux découverts[TV97].

2.4.6 La recherche à voisinages variables

La recherche à voisinages variables est une métaheuristique récente pour la résolution de problèmes d'optimisation combinatoire et globale, dont l'idée de base est le changement systématique de voisinage au sein d'une recherche locale [HM01]. L'heuristique RVV utilise les constats suivants :

- *Un minimum local par rapport à un voisinage n'en est pas nécessairement un par rapport à un autre.*
- *Un minimum global est un minimum local par rapport à tous les voisinages possibles Pour de nombreux problèmes, les minimaux locaux par rapport à un ou à plusieurs voisinages sont relativement proches les uns des autres.*

2.4.7 Procédure de recherche gloutonne aléatoire adaptative

Procédure de recherche gloutonne aléatoire adaptative (GRASP)est un algorithme d'optimisation métaheuristique et globale, initialement proposé aux praticiens de la recherche opérationnelle [HS87].GRASP est un algorithme itératif, ou chaque itération se passe sur 2 étapes :

- Étape de Construction *: Une solution réalisable est construite selon une heuristique semi-glouton.*
- Étape de Recherche Local *: La solution construite est améliorée par une recherche locale sur son voisinage. La solution finale sera la meilleure solution obtenue parmi toutes les solutions de chaque itération.*
  On note l'indépendance de itérations

2.4.8 Recherche dispersée

Recherche dispersée est un algorithme de métaheuristique et d'optimisation globale, une méthode d'évolution qui a été proposée par Glover en 1977 basée sur la population. Elle construit des solutions par la combinaison d'autres solutions en se basant sur des principes qui capturent l'information qui n'existe pas dans les solutions originales [Glo99]. L'algorithme général de la recherche dispersée est constitué de deux phases : phase d'initialisation et phase d'évolution.

2.4.9 Recherche taboue

Développée dans un cadre particulier par Glover en 1986 (et indépendamment par Hansen en 1986), c'est une méthode heuristique de recherche locale utilisée pour résoudre des problèmes complexes et/ou de très grande taille [CK95].son principe de base consiste à poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et ce :

- *en permettant des déplacements qui n'améliorent pas la solution.*
- *en utilisant le principe de mémoire (une liste taboue qui contient des mouvements ou des solutions qui sont temporairement interdits) pour éviter les retours en arrière (mouvements cycliques).*

2.4.10 Recherche taboue réactive

C'est une extension de recherche taboue dont L'objectif est d'éviter les cycles lors de l'application d'une technique de recherche locale.Recherche taboue réactive utilise une mémoire à long terme pour diversifier la recherche après qu'un seuil de répétitions de cycle a été atteint [BT94].

2.4.11 Principaux avantages des algorithmes stochastiques

- *Grace à leur caractère stochastique, il s'agit de méthodes d'optimisation globale.*
- *Grace à l'utilisation d'une population, il s'agit de méthodes facilement parallélisables.*
- *Toutes ces méthodes permettent de gérer les contraintes de manière relativement simple et efficace.*

2.4.12 Principaux inconvénients des algorithmes stochastiques

- *Cout de calcul important.*
- *La vitesse de convergence est également lente.*
- *Il existe très peu de résultats de convergence théorique de ces méthodes.*

2.5 Les algorithmes évolutionnistes

Les algorithmes évolutionnistes ou algorithmes évolutionnaires (evolutionary algorithms en anglais), sont une famille d'algorithmes dont le principe s'inspire de la théorie de l'évolution pour résoudre des problèmes divers. L'idée est de faire évoluer un ensemble de solutions à un problème donné, dans l'optique de trouver les meilleurs résultats[FC95].
Parmi les algorithmes évolutionnistes on cite :

2.5.1 Algorithmes génétiques

Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnistes. Ils sont utilisés pour effectuer des optimisations sur des problèmes complexes afin d'obtenir une solution proche de l'optimal [Gol94].Elle repose sur trois principes : le principe de variation, le principe d'adaptation et le principe d'hérédité. Il y a trois opérateurs d'évolution dans les algorithmes génétiques : La sélection, le croisement et la mutation

2.5.2 Programmation génétiques

La programmation génétique fait partie des algorithmes évolutionnaires. L'idée commune à tous ces algorithmes est d'appliquer le principe de la sélection naturelle à des objets informatiques, par exemple des programmes. L'algorithme consiste à faire évoluer une population constituée d'un grand nombre de programmes. Au départ, la population est constituée de programmes créés aléatoirement. Chaque programme est évalué selon une méthode propre au problème posé. A chaque itération (génération) on classe les programmes en fonction des notes qu'ils ont obtenues, et on crée une nouvelle population, où les meilleurs programmes auront une plus grande chance de survivre ou d'avoir des enfants que les autres [Ban+98].

2.5.3 Stratégies d'évolution

Stratégies d'évolution est un algorithme d'optimisation globale et est une instance d'un algorithme évolutionnaire du domaine du calcul évolutionnaire. L'objectif de l'algorithme stratégies d'évolution est de maximiser la collecte de solutions candidates. L'objectif a été atteint de façon classique grâce à l'adoption de la variation dynamique, un substitut pour la descente avec modification, où la quantité de variation a été adaptée dynamiquement avec des heuristiques basées sur les performances [Ven75].

2.5.4 Algorithmes à évolution différentielle

algorithmes à évolution différentielle est un algorithme de recherche directe et d'optimisation globale stochastique, et est une instance d'un algorithme évolutionnaire du domaine du calcul évolutif. Il consiste à maintenir une population de solutions candidates soumises à des itérations de recombinaison, d'évaluation et de sélection [Cor+99].

2.5.5 Programmation évolutionnaire

Programmation évolutionnaire est un algorithme d'optimisation globale et est une instance d'un algorithme évolutionnaire du domaine du calcul évolutif. L'objectif est de maximiser l'adéquation d'une collection de solutions candidates dans le contexte d'une fonction objective du domaine. Cet objectif est poursuivi en utilisant un modèle adaptatif avec substituts pour les processus d'évolution, spécifiquement héréditaires (reproduction avec variation) en compétition [FC95].

2.5.6 Evolution grammaticale

Evolution grammaticale est une technique d'optimisation globale et une instance d'un algorithme évolutionnaire du domaine du calcul évolutionnaire. Il peut également être considéré comme un algorithme de programmation automatique.

L'objectif est d'adapter un programme exécutable à une fonction objectif spécifique au problème. Ceci est réalisé grâce à un processus itératif avec des substituts de mécanismes évolutifs tels que la descente avec variation, la mutation génétique et la recombinaison, la transcription génétique et l'expression des gènes [OR01].

2.5.7 Programmation de l'expression génique

Programmation de l'expression génique est un algorithme d'optimisation globale et une technique de programmation automatique, et c'est une instance d'un algorithme évolutionnaire du domaine du calcul évolutif. L'idée est D'utiliser un processus évolutif qui fonctionne sur une représentation sous-symbolique des solutions candidates en utilisant des substituts pour les processus et les mécanismes (recombinaison génétique, mutation, inversion, transposition et expression des gènes) de l'évolution [Fer06].

2.5.8 Système d'apprentissage du classificateur

Système d'apprentissage du classificateur est à la fois une instance d'un algorithme évolutionnaire du domaine du calcul évolutif et une instance d'un algorithme d'apprentissage par renforcement de l'apprentissage machine. L'objectif de l'algorithme est d'optimiser les gains en fonction de l'exposition aux stimuli d'un environnement spécifique au problème. Ceci est réalisé en gérant l'attribution de crédit pour les règles qui s'avèrent utiles et en recherchant de nouvelles règles et de nouvelles variations sur les règles existantes à l'aide d'un processus évolutif [Bul05].

2.5.9 La méthode NSGA

La méthode NSGA (non dominated sorting genetic algorithm)est un algorithme d'optimisation à objectifs multiples (MOO) et est une instance d'un algorithme évolutionnaire du domaine du calcul évolutionnaire.
Dans cette méthode, avant que la sélection soit entamée, les solutions sont classifiées à base de non-dominance. Tous les individus non-dominés sont classés dans une catégorie avec une valeur de fitness factice proportionnelle à la taille de la population afin de fournir une possibilité de reproduction égale pour tous les individus [Deb+00].

2.5.10 La méthode SPEA

La méthode SPEA (Strength Pareto Evolutionary Algorithm) est un algorithme d'optimisation à objectifs multiples (MOO) et un algorithme évolutionnaire du domaine du calcul évolutif. L'objectif de l'algorithme est de localiser et de maintenir un front de solutions non dominées, idéalement un ensemble de solutions optimales de Pareto. Ceci est réalisé en utilisant un processus évolutif et un processus de sélection et d'une estimation de la densité du front de Pareto [ZT99].

2.5.11 Principaux Avantages des algorithmes évolutionnaires

— Un large domaine d'application : *les algorithmes évolutionnaires ont été utilisés avec succès dans une large gamme de problèmes.*
— Faciles à paralléliser *: le concept de population et la faiblesse, ou voir la non existence, de dépendances entre les individus rend la parallélisation très facile et permet ainsi de réduire considérablement le temps d'exécution.*
— *Adaptés aux espaces de recherche complexes.*

2.5.12 Principaux inconvénients des algorithmes évolutionnaires

— Complexité *: la simplicité de l'idée de base des algorithmes évolutionnaires sera payée par une grande consommation en termes de ressources.*
— Difficulté d'ajustement de paramètres *: un seul changement dans le problème peut mener à un comportement tout à fait différent.*
— Nature heuristique *: ils constituent des méthodes d'approximation qui ne donnent aucune garantie d'exactitude ou d'optimalité.*

2.6 Les algorithmes physiques

Les algorithmes Physiques c'est une technique qui repose sur l'expérimentation plutôt que sur une analyse théorique. Ils ont été développés en s'inspirant de processus qu'on retrouve dans la nature (en physique).
Parmi les algorithmes physiques on cite :

2.6.1 Recuit Simulé

Recuit simulé est un algorithme d'optimisation global qui appartient au domaine de l'optimisation stochastique et de la métaheuristique. Le recuit simulé est une adaptation de l'algorithme Monte Carlo de Metropolis-Hastings .L'idée est d'effectuer un mouvement selon une distribution de probabilité qui dépend de la qualité des différents voisins on utilisant un paramètre, appelé la température notée T et on se basant sur l'algorithme Metropolis-Hastings [DC05].

2.6.2 Optimisation extrêmale

Optimisation extrêmale est une technique de recherche stochastique qui a les propriétés d'être une méthode de recherche locale et globale. L'idée est d'Identifier de manière itérative les composants les moins performants d'une solution donnée et de les remplacer ou de les échanger avec d'autres composants. Ceci est réalisé grâce à l'allocation des coûts aux composants de la solution en fonction de leur contribution au coût global de la solution dans le domaine problématique.

2.6.3 Recherche d'harmonie

Recherche d'harmonie appartient aux domaines de l'intelligence informatique et de la métaheuristique. dont le principe est d'Utiliser les bonnes solutions candidates déjà découvertes

pour influencer la création de nouvelles solutions candidates vers la localisation des problèmes optima en créant stochastiquement des solutions candidates par étapes, où chaque composant est soit tiré au hasard d'une mémoire de solutions de haute qualité, ajusté à partir de la mémoire de solutions de haute qualité, soit assigné au hasard dans les limites du problème [Gee09].

2.6.4 Algorithme Culturel

Algorithme Culturel est une extension du domaine du calcul évolutif et peut être considéré comme un algorithme méta-évolutionnaire. Il appartient plus largement au domaine de l'Intelligence Computationnelle et de la Métaheuristique. L'idée est d'Améliorer l'apprentissage ou la convergence d'une technique de recherche intégrée en utilisant une évolution culturelle d'ordre supérieur. L'algorithme fonctionne à deux niveaux : un niveau de population où les individus représentent des solutions candidates et un niveau culturel où sont stockées les informations acquises par les générations et accessibles à la génération actuelle. Un protocole de communication est utilisé pour permettre aux deux espaces d'interagir [Rey94].

2.6.5 Algorithme mémétique

Les algorithmes mémétiques sont des éléments de métaheuristique et d'intelligence computationnelle. Bien qu'ils aient des principes d'algorithmes évolutionnaires, ils ne peuvent pas être strictement considérés comme une technique évolutionniste.L'objectif est d'xploiter une technique de recherche globale basée sur la population pour localiser largement les bonnes zones de l'espace de recherche, combinée à l'utilisation répétée d'une heuristique de recherche locale par des solutions individuelles pour localiser l'optimum local [KS05].

2.6.6 Les principaux avantages des algorithmes physiques

- *Temps de calcul raisonnable.*
- *Ils Permettent d'obtenir un état solide « bien ordonné ».*
- *L'utilisation des paramètres simple.*

2.6.7 Les principaux inconvénients des algorithmes physiques

- *L'incertitude de trouver la solution optimale .*
- *Leur difficulté .*

2.7 Les Algorithmes probabilistes

Les algorithmes probabilistes ont la particularité que, lorsqu'ils sont confrontés à un choix, plutôt que « perdre du temps » à essayer de déterminer l'alternative qui semble la meilleure, ils effectuent plutôt un choix aléatoire, en espérant que le temps ainsi sauvé permettra, en moyenne, d'obtenir quand même un bon résultat, et ce en temps raisonnable [HLG99].
Parmi les algorithmes probabilistes on cite :

2.7.1 La méthode PBIL

La méthode PBIL (population based incremental learning) est une estimation de l'algorithme de distribution (EDA), une extension du domaine du calcul évolutif dont le principe est de réduire la mémoire requise par l'algorithme génétique. Cela se fait en réduisant la population d'une solution candidate à un seul prototype vecteur d'attributs à partir duquel des solutions candidates peuvent être générées et évaluées [Gre96].

2.7.2 Algorithme à distribution marginale univariée

Algorithme à distribution marginale univariée appartient au domaine de l'estimation des algorithmes de distribution (EDA), une extension du domaine du calcul évolutif. L'idée est d'tiliser la fréquence des composants dans une population de solutions candidates dans la construction de nouvelles solutions candidates. Ceci est réalisé en mesurant d'abord la fréquence de chaque composant dans la population (la probabilité marginale univariée) et en utilisant les probabilités pour influencer la sélection probabiliste des composants dans la construction par composant de nouvelles solutions candidates [PM98].

2.7.3 Algorithme génétique compact

Algorithme génétique compact est un algorithme d'estimation de la distribution (EDA), une extension du domaine du calcul évolutif. Il a pour objectif de Simuler le comportement d'un algorithme génétique avec une empreinte mémoire beaucoup plus petite. Ceci est réalisé en maintenant un vecteur qui spécifie la probabilité d'inclure chaque composant dans une solution dans de nouvelles solutions candidates. Les solutions candidates sont générées de manière probabiliste à partir du vecteur et les composants de la meilleure solution sont utilisés pour apporter de petites modifications aux probabilités dans le vecteur [HLG99].

2.7.4 Optimisation bayésienne

Optimisation bayésienne Algorithme appartient au domaine des algorithmes d'estimation des distributions, une extension du domaine du calcul évolutif. Plus largement, BOA appartient au domaine de l'Intelligence Computationnelle. L'idée est de Construire un modèle probabiliste qui décrit les relations entre les composants des solutions d'ajustement dans l'espace du problème. Ceci est réalisé en répétant le processus de création et d'échantillonnage à partir d'un réseau bayésien qui contient les dépendances conditionnelles, les indépendances et les probabilités conditionnelles entre les composants d'une solution [PG02].

2.7.5 L'entropie croisée

L'entropie croisée est une optimisation probabiliste appartenant au domaine de l'optimisation stochastique. Son principe consiste à échantillonner l'espace du problème et approximer la distribution des bonnes solutions. Ceci est réalisé en supposant une distribution de l'espace du problème (tel que gaussien), en échantillonnant le domaine du problème, en générant des solutions candidates en utilisant la distribution et en mettant à jour la distribution en fonction des meilleures solutions candidates découvertes [RK04].

2.7.6 Principaux avantages des algorithmes probabilistes

— *Peuvent produire, de façon certaine, des solutions exactes*

— *Peuvent être considérés comme des algorithmes non-déterministes*

2.7.7 Principaux inconvénients des algorithmes probabilistes

— *Difficile de choisir les différents paramètres*
— *Temps d'exécution non raisonnable*

2.8 Les algorithmes swarm

La Swarm Intelligence est un concept reposant sur la coordination entre un nombre massif d'intelligences individuelles. Dans la nature, de nombreuses créatures sont en mesure de coordonner leurs intelligences individuelles pour effectuer des tâches complexes. Elles font preuve d'efficacité ou même faire preuve de créativité. Prenons l'exemple des fourmis. Une fourmi seule n'est capable d'effectuer qu'un nombre limité d'actions. Toutefois, une colonie de fourmis est en mesure de construire des ponts, des autoroutes de nourriture et d'informations. Elle peut aussi partir en guerre, ou même réduire d'autres espèces de fourmis en esclavage. Les informations sont généralement stockées dans tous les agents homogènes participants, ou sont stockées ou communiquées dans l'environnement elle-même, par exemple grâce à l'utilisation de phéromones chez les fourmis [EK95].

Parmi les algorithmes swarm on cite :

2.8.1 Optimisation par essaims particulaires

Optimisation par essaims particulaires appartient au domaine de l'intelligence collective et est un sous-domaine de l'intelligence informatique. Il est lié à d'autres algorithmes Swarm Intelligence tels que Ant Colony Optimization. Le but de l'algorithme est de faire en sorte que toutes les particules localisent les optima dans un hyper-volume multidimensionnel en attribuant des positions initialement aléatoires à toutes les particules dans l'espace et de petites vitesses aléatoires initiales. L' objectif est d'échantillonnée après chaque mise à jour de position. Au fil du temps, grâce à une combinaison d'exploration et d'exploitation de bonnes positions connues dans l'espace de recherche, les particules se regroupent ou convergent autour d'un optima, ou de plusieurs optima.

2.8.2 Système fourmi

Système fourmi était à l'origine le terme utilisé pour désigner une gamme d'algorithmes basés sur Ant. L'algorithme Ant System est la méthode de base pour l'optimisation des colonies de fourmis pour les extensions populaires telles que Elite Ant System, Rank Ant-System, Max-Min Ant System et Ant Colony System. L'objectif est d'exploiter les informations historiques et heuristiques pour construire des solutions candidates qui sont construisent une pièce discrète de manière probabiliste par étapes. La probabilité de sélection d'un composant est déterminée

par la contribution heuristique du composant au coût global de la solution. L'historique est mis à jour proportionnellement à la qualité de la solution la plus connue [DMC96].

2.8.3 Système de colonie de fourmis

L'algorithme Système de colonie de fourmis est un exemple de méthode d'optimisation Ant Colony dans le domaine de Swarm Intelligence, Metaheuristics et Computational Intelligence. Il s'agit d'une extension de l'algorithme Ant System et est liée à d'autres méthodes d'optimisation de la colonie de fourmis telles que Elite Ant System et Rank-based Ant System [DMC96].

2.8.4 Algorithme des abeilles

L'algorithme des abeilles est lié à d'autres algorithmes inspirés des abeilles, tels que l'optimisation des colonies d'abeilles, et à d'autres algorithmes Swarm Intelligence tels que l'optimisation des colonies de fourmis et l'optimisation des essaims de particules. L'objectif de l'algorithme est de localiser et d'explorer de bons sites dans un espace de recherche de problèmes. Les bons sites sont exploités via l'application d'une recherche locale. Les bons sites sont continuellement exploités, bien que de nombreux éclaireurs soient envoyés à chaque itération toujours à la recherche de bons sites supplémentaires [Pha+05].

2.8.5 Optimisation de nourriture bactérienne

Optimisation de nourriture bactérienne est lié à l'algorithme de chimiotaxie des bactéries et à d'autres algorithmes de Swarm Intelligence. Il y a eu de nombreuses extensions de l'approche qui tentent d'hybrider l'algorithme avec d'autres algorithmes d'intelligence informatique et métaheuristiques. L'algorithme permet aux cellules de pulluler stochastiquement et collectivement vers les optima. Ceci est réalisé par : chimiotaxie, Reproduction, Élimination-dispersion [Mul+02].

2.8.6 Principaux avantages des algorithmes Swarm

- *Simple à mettre en oeuvre.*
- *Convergence rapide et robuste*
- *probabilité et une efficacité plus élevée pour trouver les optima globaux.*

2.8.7 Principaux inconvénients des algorithmes Swarm

- *Il peut être difficile de définir les paramètres de conception initiaux.*
- *Peut converger prématurément et être piégé dans un minimum local surtout avec des problèmes complexes.*

2.9 Les algorithmes immunitaires

L'AI est un ensemble de systèmes informatiques inspirés de l'immunologie théorique, des fonctions immunitaires observées, des principes et des mécanismes, et a été appliqué pour

résoudre divers problèmes d'optimisation complexes [DCT03]. Les objectifs du système immunitaire sont de protéger le corps contre les agents pathogènes (pathogènes) et d'éliminer les cellules défectueuses.
Parmi les algorithmes immunitaires on cite :

2.9.1 Algorithme de sélection clonale

L'algorithme de sélection clonale appartient au domaine des systèmes immunitaires artificiels. Il existe de nombreuses extensions à cet algorithme, y compris des ajustements tels que les approches CLONALG1 et CLONALG2, une version pour la classification appelée CLONCLAS et une version adaptative appelée Adaptive Clonal Selection (ACS). La stratégie utilisée par l'algorithme implique une population d'unités d'information adaptatives soumises à des processus de sélection compétitifs qui, avec la duplication et la variation qui en résultent, améliorent l'ajustement adaptatif des unités d'information à leur environnement [Bur+59].

2.9.2 Algorithme de sélection négative

L'algorithme de sélection négative appartient au domaine des systèmes immunitaires artificiels. L'algorithme est lié à d'autres systèmes immunitaires artificiels tels que l'algorithme du réseau immunitaire. Le principe est atteint en construisant un modèle de changements, ou des données inconnues en générant des modèles qui ne correspondent pas à un corpus existant de modèles disponibles. Le modèle non normal préparé est ensuite utilisé pour surveiller les données normales existantes de nouvelles données en recherchant des correspondances aux modèles non normaux [JD07].

2.9.3 Systèmes immunitaires artificiels algorithme du réseau immunitaire

Systèmes immunitaires artificiels algorithme du réseau immunitaire, il fournit la base d'extensions telles que le système de reconnaissance immunitaire artificielle parallèle. Il est lié à d'autres algorithmes du système immunitaire artificiel tels que l'algorithme des cellules dendritiques. L'objectif de la technique est de préparer un ensemble de vecteurs à valeur réelle pour classer les modèles. Le système conserve un pool de cellules mémoire qui sont préparées en exposant le système à une seule itération des données d'apprentissage. Les cellules mémoire candidates sont préparées lorsque les cellules mémoire sont insuffisamment stimulées pour un modèle d'entrée donné. Un processus de clonage et de mutation des cellules se produit pour la cellule mémoire la plus stimulée[MB02].

2.9.4 Algorithme de réseau immunitaire

L'algorithme de réseau immunitaire artificiel (AINET) est un algorithme de réseau immunitaire du domaine des systèmes immunitaires artificiels. l'algorithme comprend la version de base et l'extension pour les problèmes d'optimisation appelés l'algorithme Optimization Artificial Immune Network (opt-aiNet). L'objectif est de préparer un répertoire de détecteurs de motifs discrets pour un domaine de problème donné. Ce principe est atteint par un processus interactif d'exposition de la population à des informations externes auxquelles elle répond à

la fois par une réponse de sélection clonale et une méta-dynamique interne de réponses de la population.

2.9.5 Algorithme de cellules dendritiques

L'algorithme des cellules dendritiques appartient au domaine de l'immunité artificielle Systèmes L'algorithme est la base d'extensions telles que l'algorithme déterministe des cellules dendritiques (dDCA). Il est généralement lié à d'autres algorithmes du système immunitaire artificiel tels que l'algorithme de sélection clonale. L'objectif de l'algorithme est de préparer un ensemble de cellules dendritiques matures qui fournissent des informations spécifiques au contexte sur la façon de classer les modèles d'entrée normaux et anormaux. Réalisé comme un système de trois processus asynchrones de :

1. *migrer des cellules immatures suffisamment stimulées.*
2. *promouvoir les cellules migrées vers un état semi-mature en fonction de leur réponse accumulée.*
3. *étiqueter les profils observés.*

2.9.6 Principaux avantages des algorithmes immunitaires

— *Simple*
— *Naturellement adapté aux tâches de reconnaissance de formes et de classification.*
— *peut gérer un problème d'optimisation multi objectif.*

2.9.7 Principaux inconvénients des algorithmes immunitaires

— *Ils nécessitent des paramètres supplémentaires.*
— *Faible convergence.*

2.10 Les algorithmes neuronaux

Algorithmes modélisés librement d'après le cerveau humain et conçus pour reconnaître les modèles. Ils interprètent les données sensorielles à travers une sorte de perception machine, d'étiquetage ou de regroupement des entrées brutes. Les motifs qu'ils reconnaissent sont numériques, contenus dans des vecteurs, dans lesquels toutes les données du monde réel, que ce soit des images, du son, du texte ou des séries temporelles, doivent être traduites.
Parmi les algorithmes neuronaux on cite :

2.10.1 Perceptron

L'algorithme Perceptron appartient au domaine des réseaux de neurones artificiels. Il s'agit d'un réseau neuronal à action directe à une seule couche qui a inspiré de nombreuses extensions, non limitées à ADALINE et aux règles d'apprentissage de Widrow-Hoff. L'objectif de la technique est de modéliser une fonction donnée en modifiant les pondérations internes des signaux d'entrée pour produire un signal de sortie attendu. En utilisant une méthode d'apprentissage supervisé, où l'erreur entre la sortie du système et une sortie attendue connue est présentée au système et utilisée pour modifier son état interne [Min69].

2.10.2 Rétropropagation

L'algorithme de rétropropagation est une méthode d'apprentissage supervisé pour les réseaux multicouches à action directe. Le nom fait référence à la propagation en arrière de l'erreur lors de la formation du réseau. la même stratégie que l'algorithme de préception. mais chaque couche du réseau fournit une abstraction du traitement de l'information de la couche précédente, permettant la combinaison de sous-fonctions et d'une modélisation d'ordre supérieur [Rum+95].

2.10.3 Réseau de Hopfield

Le réseau Hopfield est un réseau neuronal récurrent et est lié à d'autres réseaux récurrents tels que la mémoire associative bidirectionnelle (BAM). Il est généralement lié aux réseaux de neurones artificiels à action directe tels que l'algorithme de rétropropagation

L'objectif du système est d'associer les composants d'un modèle d'entrée à une représentation globale du modèle appelé Mémoire adressable par le contenu (CAM). Cela signifie qu'une fois formé, le système rappellera des modèles entiers, étant donné une partie ou une version bruyante du modèle d'entrée [Wei+92].

2.10.4 Apprentissage de la quantification vectorielle

L'algorithme de quantification vectorielle d'apprentissage est un réseau neuronal supervisé qui utilise une stratégie d'apprentissage compétitive (gagnant-tout). De plus, LVQ est une technique de base qui a été définie avec quelques variantes LVQ1, LVQ2, LVQ2.1, LVQ3, OLVQ1 et OLVQ3 ainsi que de nombreuses extensions et raffinements tiers trop nombreux pour être répertoriés. L'objectif de l'algorithme est de préparer un ensemble de vecteurs de livre de codes dans le domaine des échantillons de données d'entrée observés et d'utiliser ces vecteurs pour classer des exemples invisibles. Un pool de vecteurs initialement aléatoire est préparé qui est ensuite exposé à des échantillons d'apprentissage [Wei+92].

2.10.5 Carte auto-organisatrice

La carte auto-organisatrice est un réseau neuronal non supervisé qui utilise une stratégie d'apprentissage compétitive (gagnant-à-tout). Elle est liée à d'autres réseaux de neurones non supervisés tels que la méthode de la théorie de la résonance adaptative (ART). SOM est une technique de base qui a inspiré de nombreuses variantes et qui ne se limite pas à la carte auto-organisatrice Adaptive-Subspace (ASSOM) L'objectif de l'algorithme est de placer de manière optimale une topologie de vecteurs de livre de codes dans le domaine des échantillons de données d'entrée observés. [Koh90].

2.10.6 Principaux avantages des algorithmes neuronaux

— *Facile à conceptualiser.*
— *Capable de détecter des relations complexes.*
— *Calcul à grande vitesse.*

2.10.7 Principaux inconvénients des algorithmes neuronaux

— *Pas exact.*
— *Grande complexité de la st ructure du réseau.*

2.11 Conclusion

Dans ce chapitre nous avons dressé un état de l'art des métaheuristiques. Nous avons présenté les principales métaheuristiques utilisées pour résoudre des problèmes d'optimisation. Elles ont été regroupées en sept grandes familles, Les algorithmes stochastiques, évolutionnistes, physiques, probabilistes, Swarm, immunitaires et les Algorithmes neuronaux.

Chapitre 3

Implémentation et Expérimentation

3.1 Introduction

Ce chapitre est consacré à la partie conception et réalisation de notre application, qui consiste à recommander des livres à un utilisateur en se basant sur deux méthodes métaheuristiques : le NSGAII et recuit simulé. Dont nous préciserons les paramètres (nombre d'itérations, le critère de similarité). Ensuit nous effectuerons des tests pratiques pour mesurer l'efficacité de notre travail et un bilan comparatif de ces deux algorithmes. Enfin nous décrivons les technologies utilisées pour l'implémentation et nous présentons quelques captures d'écran de l'interface.

3.2 Description de notre approche

Notre système est composé de 4 étapes :

1. *Exploration et traitement des données.*
2. *regroupement des utilisateurs similaires avec les techniques métaheuristiques.*
3. *Prédiction des notes des livres non pas encore visités par les nouveaux utilisateurs.*
4. *recommandation des livres qui ont eu des notes supérieures à 3.*

3.2.1 Exploration et traitement des données

Le projet utilise un ensemble de données obtenu à partir de fastml.com qui à son tour est basé sur le bien-connu site Web goodreads.com . Les données ont été importées des fichiers .txt en trois trames de données pour l'évaluation et le traitement des données avant leur utilisation dans la recommandation.

Evaluations des utilisateurs

Figure 3.1 présente l'evaluations des utilisateurs :

```
"User-ID";"ISBN";"Book-Rating"
"276725";"034545104X";"0"
"276726";"0155061224";"5"
"276727";"0446520802";"0"
"276729";"052165615X";"3"
```

Figure 3.1 – l'evaluations des utilisateurs :

Les livres

Chaque livre est définie par un Identifiant, Titre , Auteur , Date de publication et edéteur. Figure 3.2 illustre la représentation des livres :

```
"ISBN";"Book-Title";"Book-Author";"Year-Of-Publication";"Publisher";"Image-URL";
"0590353403";"Harry Potter and the Sorcerer's Stone (Book 1)";"J. K. Rowling";"1998";"Scholastic";"http://images.amazon.com/images/P/0590353403.01.THUMBZZZ.jpg";
"0553243055";"Firefox Down";"Craig Thomas";"1984";"Bantam Doubleday Dell";"http://images.amazon.com/images/P/0553243055.01.THUMBZZZ.jpg";
"0064401774";"More Scary Stories To Tell In The Dark";"Alvin Schwartz";"1986";"HarperTrophy";"http://images.amazon.com/images/P/0064401774.01.THUMBZZZ.jpg";
```

FIGURE 3.2 – la représentation des livres

Les utilisateurs

Chaque utilisateur est définier par un Identifiant, Nom et Prénom. Figure 3.3 illustre la représentation des utilisateurs :

```
"User-ID";"Nom";"Prénom"
"1";"djes";"gen"
"2";"scott";"arto"
"3";"parter";"bent"
"4";"wad";"ranger"
```

FIGURE 3.3 – la représentation des utilisaterus

3.2.2 Regroupement des utilisateurs similaires avec les techniques méta-heuristiques

Les titres des livres ont été comparés aux notes des lecteurs et un vecteur X est créée pour chaque lecteur et livre « 1 » si le lecteur avait noté un livre « 0 » sinon . un vecteur Y pour les notes données par ses lecteurs aux livres .

Distribution aléatoire des utilisateurs

Après avoir attribuer chaque utilisateur à un groupe aléatoirement nous calculons le centre de chaque groupe, Ensuite nous calculons la distance entre les centre et les utilisateurs puis nous plaçons les utilisateurs dans les groupes des centres les plus proche (Mutation).

Fonction Objective

la somme des nombres des livres en commun et la somme des distances euclidiennes entre tous les utilisateur pour tous les groupes.

Contribution à la recommandation des livres avec les méthodes métaheuristiques

Contribution à la recommandation des livres par Recuit simulé

Algorithme 1: Recuit Simulé

Entrée *: Les utilisateurs, Les livres, nbClustre;*
pour *chaque utilisateur* **faire**
    Calculer vecteur X ;
    Calculer vecteur Y ;
fin
$T = T_{int}$; T_{int} *est la température initialisée* à *une grande valeur*
$S_0 = Distribution\ aléatoire(utilisateurs, nbclustre, livre)$
$S \leftarrow S_0$; S_0 *unesolution initiale*
$Best \leftarrow S$;
tant que $T > 0$ **faire**
    R = *Distribution aléatoire (utilisateurs, nbclustre, livre);*
    Générer un r aléatoire entre 0 et 1
    si $R < S_0$ ***ou*** $r < e - \frac{R-S_0}{T}$ **alors**
        $S \leftarrow R$;
    fin
    si $S <$ *best* **alors**
        $Best \leftarrow S$;
        Décrémenter t ;
    fin
fin

Algorithme 2: Distribution aléatoire (utilisateurs, nbclustre, livre)

Générer aléatoirement une distribution des utilisateurs Pour chaque clustre Calculer Centroid;
Mutation ;
pour *chaque groupe* **faire**
    Calculer Centroid;
    Calculer somme des distances entre les utilisateurs;
    Nombre des livres communs;
fin
Calculer la fonction objective ;
Retourner *Fonction objective ;*

3.2.3 Contribution à la recommandation des livres par NSGAII

Initialisation Nous attribuons chaque utilisateur à un clustre aléatoirement, cette opération sera répétée selon le nombre de population tout en construisant une matrice des chromosomes.

Algorithme 3: NSGA

Entrée *: Les utilisateurs, Les livres, nbClustre, nbPopulation, NbGénération;*
pour *chaque utilisateur* **faire**
    Calculer vecteur X ;
    Calculer vecteur Y ;
fin
Initialisation*;*
pour chaque chromosome Calculer Fonction objective ;
$T = 0$ *;*
tant que $T >$ *NbGénération* **faire**
    Crossover*;*
    $P_t \leftarrow t_{ri}$ *selon non dominance ;*
    $Q_t \leftarrow$ *Génération partiale*;
    $R \leftarrow P_t \cup Q_t$ *; Créer nouvelle population*
    Mutation *;*
    $T = T++$ *;*
fin

Figure 3.4 présente le code de NSGAII :

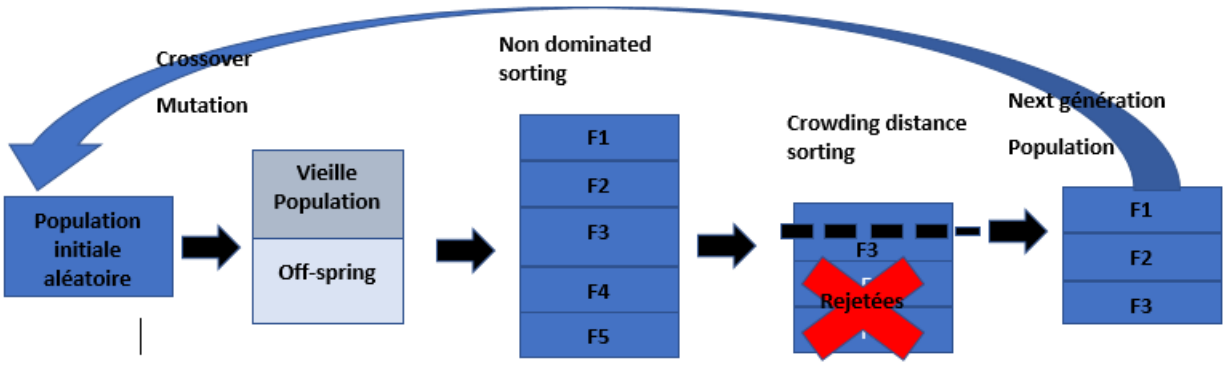

FIGURE 3.4 – code de NSGAII

3.2.4 Prédiction des notes des livres non pas encore visités par les nouveaux utilisateurs

Après avoir regrouper les lecteurs similaires dans un même groupe les voisins de nouveau utilisateur u, notés $N(u)$, sont les K utilisateurs v avec la plus grande mesure de similarité w_{uv} par rapport à u. Seuls les utilisateurs ayant noté l'item i peuvent être utilisés pour la prédiction. L'ensemble de ces voisins est noté $N_i(u)$. La note r_{ui} de l'utilisateur u sur l'item i peut être prédite par la moyenne des notes r_{vi} de ces voisins :

$$\acute{r}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} r_{vi}$$

3.2.5 Recommandation

Recommandation des livres qui ont eu après la prédiction des notes supérieures à 3.

3.3 Evaluation

Évaluer un système de recommandation comprend la résolution d'un problème, et l'application d'une méthode d'évaluation pour vérifier si le problème est résolu ou pas. Pour que les systèmes de recommandation soient utiles Ils doivent apporter une solution à un problème. Le problème doit être bien défini afin qu'il soit mesurable même si le problème a été résolu ou non. Il existe plusieurs façons différentes d'expérimenter, et valider l'efficacité d'un système de recommandation. L'approche d'évaluation que nous utiliserons pour notre système est : la méthode d'évaluation hors ligne.

3.4 L'approche d'évaluation hors ligne

La méthode d'évaluation hors ligne est très attractive pour notre approche car cela nous permet de tester nos algorithmes sans interaction avec l'utilisateur. Ceci est idéal dans les situations où nous avons déjà des données sur les utilisateurs, les items et les notes. Cette méthode prend comme entrées le data set pour simuler les utilisateurs réels et tester les algorithmes. Un petit nombre des utilisateurs est sélectionné au début pour représenter l'ensemble des utilisateurs pour tester les algorithmes. Ils sont ensuite introduits au système en tant que nouveaux utilisateurs. L'ensemble des résultats produits par notre système de recommandation est comparé aux valeurs réelles pour voir nos performances. Pour évaluer nos algorithmes, nous faisons une recommandation pour chaque utilisateur qui seront après comparer avec les notes données par ces utilisateurs dans le data set. Plus de détails sur nos expériences, nous sélectionnerons 10 utilisateurs aléatoirement. Nous sélectionnerons ensuite un utilisateur aléatoire à partir de cette liste d'échantillons. Nous présenterons cet utilisateur au système en tant que nouvel utilisateur. Le data set aura toutes les notes de cet utilisateur supprimées. Ensuite, nous sélectionnerons au hasard un item que l'utilisateur a réellement évalué, pour représenter l'élément initial que l'utilisateur a évalué. Ensuite, nous exécuterons nos algorithmes pour

générer des recommandations. Nous le ferons pour tous les utilisateurs de l'échantillon. Pour bien comprendre, supposons que l'utilisateur ait des notes comme indiqué dans la figure 3.3.

Figure 3.5 présente exemples de notes pour un utilisateur :

#	TITRE	LA_NOTE
1	A Soldier of the Great War	2.0
2	A Game of Thrones (A Song of Ice and Fire, Book 1)	5.0
3	The COLDEST WAR: A MEMOIR OF KOREA	5.0
4	Maps and Mapmakers of the Civil War	5.0
5	War and Peace	4.0

FIGURE 3.5 – Exemples de notes pour un utilisateur

La table 3.1 a des items générés à l'utilisateur après la recommandation.

TABLE 3.1 – Exemples de recommandations générées pour l'utilisateur

Recommandation
The COLDEST WAR :A MEMOIR OF KOREA
Maps and Mapmakers of the Civil War
War and Peace
The Vikings

On calcule la moyenne de l'utilisateur 4.20, à partir des items notés comme le montre l'équation (a). Après nous calculons la moyenne des livres qui ont été générés à l'utilisateur comme le montre l'équation (b) , Ensuit nous calculons les deux résultats obtenus.

La moyenne des items recommandées est supérieur à la moyenne réelle de l'utilisateur, par conséquent On peut dire que la qualité de la recommandation est favorable. Une chose clé à noter est que la recommandation inclure un livre «The Vikings » que l'utilisateur n'a pas vu ni noté ,nous ne pouvons pas utiliser de manière fiable ce livre dans notre prévision de notation si cela peut affecter notre résultat positivement ou négativement. Nous éliminant toute incertitude de nos résultats.

$$\begin{aligned} AVG &= \tfrac{2+5+5+5+4}{5} \\ &= \tfrac{21}{5} \\ &= 4.20 \end{aligned} \qquad (a)$$

$$\begin{aligned} AVG &= \tfrac{5+5+4}{3} \\ &= \tfrac{14}{3} \\ &= 4.66 \end{aligned} \qquad (b)$$

Les limites de la méthode d'évaluation hors ligne sont que nous ne pouvons pas vérifier les livres auxquels l'utilisateur n'a pas attribué de note. Cela nous limite le nombre d'items que nous pouvons utiliser.la méthode hors ligne n'est pas idéal dans les cas des nouveaux items à recommander qui essaient de résoudre les problèmes de rareté. Ce n'est pas le cas avec notre approche vu que notre objectif est de faire des recommandations de toute sorte à l'utilisateur.

3.5 Résultats généraux

Dans la table 3.2 nous avons calculé les notes réelles et les notes des items générés de 10 utilisateurs pour les deux algorithmes. Dans la figure 3.6, nous avons un graphique des notes réelles de l'utilisateur comparées à celles des items génères avec l'algorithme de Récuit Simulé et dans la figure 3.7 avec le NSGA. Les notes réelles faire référence aux notes données par les utilisateurs aux items dans le data set. Les graphiques montrent que les notes sont plus ou moins comparées les unes aux autres. Les notes réelles sont supérieures pour certains utilisateurs. Tandis que celles des items génères sont supérieurs pour les autres. Regarder de plus près, on voit qu'en moyenne, les notes des items générées sont supérieurs des notes réelles. C'est une indication que nos algorithmes sont bien adaptés pour générer des livres de qualité à recommander en l'absence d'informations à propos de l'utilisateur.

TABLE 3.2 – Résultats de l'exécution du Recuit simulé et NSGA dix fois

Recuit simulé		**NSGA**	
Notes réelles	**Notes recommandées**	**Notes réelles**	**Notes recommandées**
2.1	3	3.9	4
3.1	4	3.7	3.9
2.5	2.4	3.5	3.5
2.7	3	3.7	3.6
4.4	3.2	3.2	2.9
4.1	3.8	3.9	4
4.2	4.3	4.2	4.6
2.9	3.6	4.2	4.5
3.9	4.2	3.9	4.2
4	5	4	4

Figure 3.6 présente une Aperçu des évaluations de 10 utilisateurs Avec le Recuit simulé :

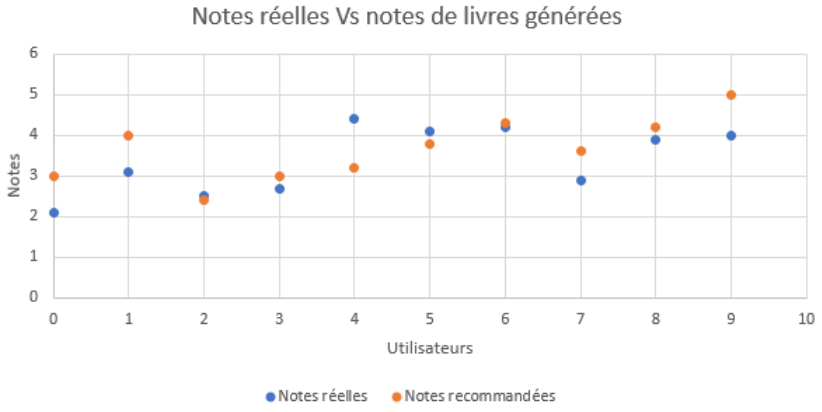

FIGURE 3.6 – Aperçu des évaluations de 10 utilisateurs Avec le Recuit simulé

Figure 3.7 présente Aperçu des évaluations de 10 utilisateurs Avec NSGA :

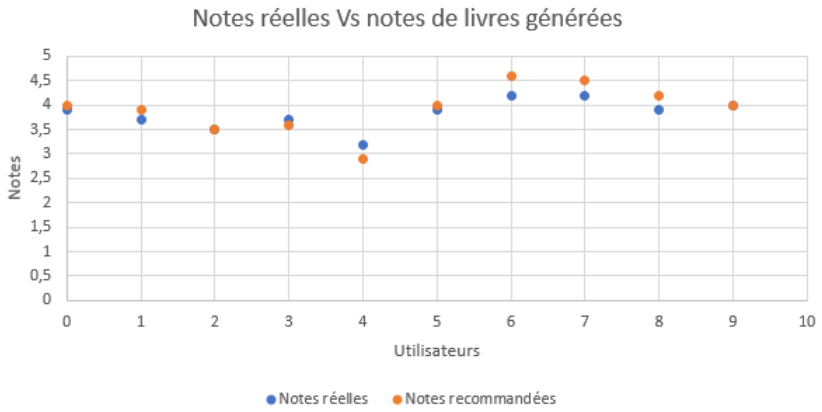

FIGURE 3.7 – Aperçu des évaluations de 10 utilisateurs Avec NSGA

3.6 Discussion des résultats

Nous avons implémenté nos deux algorithmes pour faire des recommandations en utilisant un minimum d'informations. Nous avons constaté que nos deux algorithmes remplissent les critères pour lesquels ils sont conçus. Le livre que l'utilisateur à noter a été utilisé pour trouver des livres similaires. Les livres ont été ensuite utilisés pour former un voisinage des utilisateurs qui serons utilisés pour faire des prédictions. L'utilisation d'un minimum d'information est un cas très difficile et peu probable. Cependant, en utilisant un algorithme évolutionniste multi-objectif nous avons pu faire des recommandations. L'algorithme NSGA s'est mieux comporté que le Récuit simulé au prix de la vitesse.

3.7 Implémentation

3.7.1 Outils de développement

NetBeans IDE

NetBeans IDE est un environnement de développement intégré (EDI), permet également de supporter différents autres langages, comme java, C, C++, JavaScript, XML, Groovy, PHP et HTML de faon native ainsi que bien d'autres (comme Python ou Ruby) par l'ajout de greffons. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refacturant, éditeur graphique d'interfaces et de pages Web). NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plateforme.
Figure 3.8 présente fenêtre de programmation Sur Netbeans :

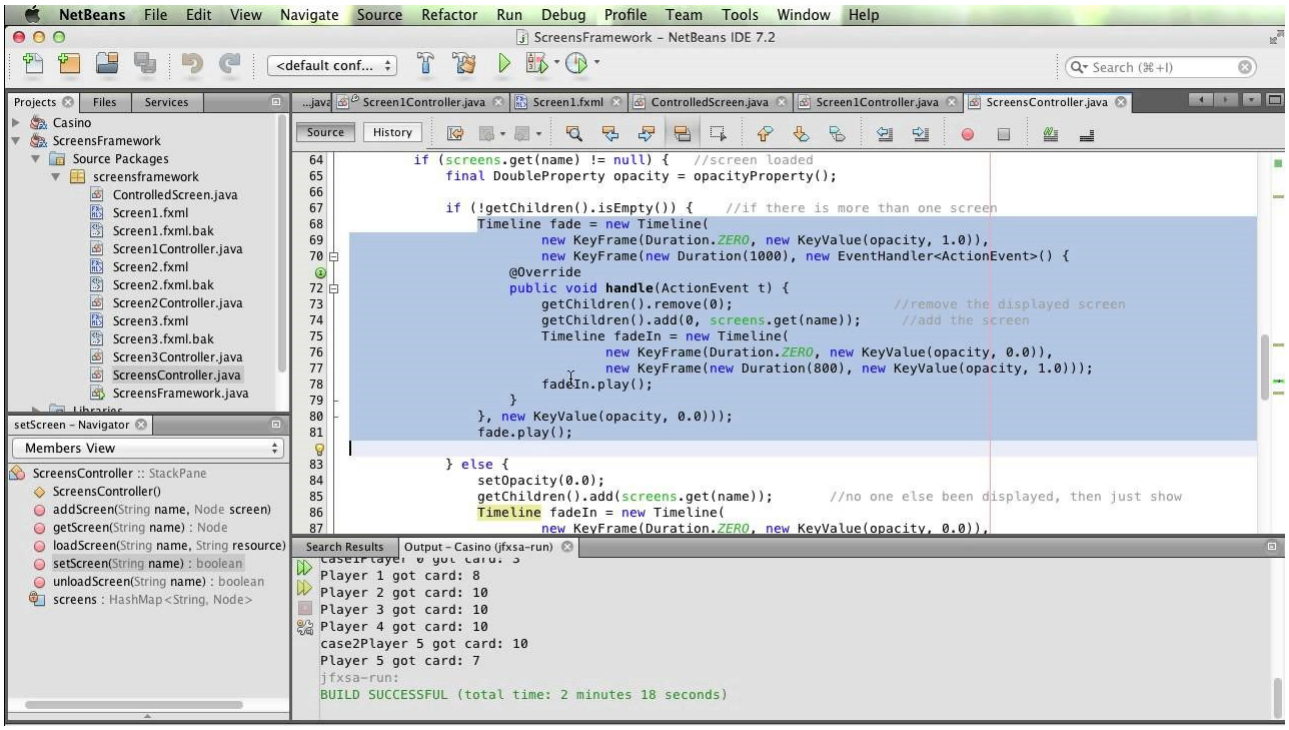

FIGURE 3.8 – fenêtre de programmation Sur Netbeans

3.7.2 Langage de programmation (Java)

Java est un langage de programmation et une plateforme informatique qui ont été créés par Sun Microsystems en 1995. Beaucoup d'applications et des sites Web ne fonctionnent pas si Java n'est pas installé et leur nombre ne cesse de croître chaque jour. Java est rapide, sécurisé et fiable. Des ordinateurs portables aux centres de données, des consoles de jeux aux super ordinateurs scientifiques, des téléphones portables à Internet, la technologie Java est présente sur tous les fronts.
C'est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates. Java est notamment largement utilisée pour le développement d'applications d'entreprises et mobiles.

Environnement Java

Java est un langage interprété, ce qui signifie qu'un programme compilé n'est pas directement exécutable par le système d'exploitation mais il doit être interprété par un autre programme, qu'on appelle interpréteur.
Figure 3.9 présente l'architecture exécutable Code java :

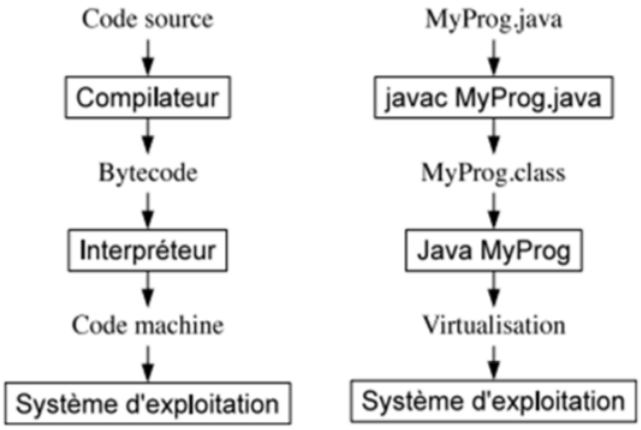

FIGURE 3.9 – architecture exécutable Code java

3.7.3 JavaFX

JavaFX Intégration

JavaFX est une bibliothèque graphique intégrée dans le JRE et le JDK de Java. Oracle la décrit comme « The Rich Client Platform », c'est-à-dire qu'elle permet de réaliser des interfaces graphiques évoluées et modernes grâce à de nombreuses fonctionnalités, telles que les animations, les effets, la 3D, l'audio, la vidéo, etc. Elle a de plus l'avantage d'être dans le langage Java, qui permet de réaliser des architectures avec des paradigmes objet, et aussi de pouvoir utiliser le typage statique. Dans ce premier tutoriel, nous allons voir ensemble un rapide historique de la bibliothèque pour ensuite découvrir les fondamentaux qui sont les classes « Stage », « Scene », « Application » et le « threading » associé, pour finir nous verrons les « Node » avec un exemple d'utilisation du « scene graphe ». Cette présentation ne fait pas dans le bling-bling, même si JavaFX est doué pour cela, en préférant se focaliser sur les concepts primordiaux d'une telle bibliothèque. Bien comprendre ces basiques vous aidera bien à commencer pour ensuite pouvoir faire des interfaces de qualité et peut-être spectaculaires

Figure 3.10 Illustre un projet Java FX Main :

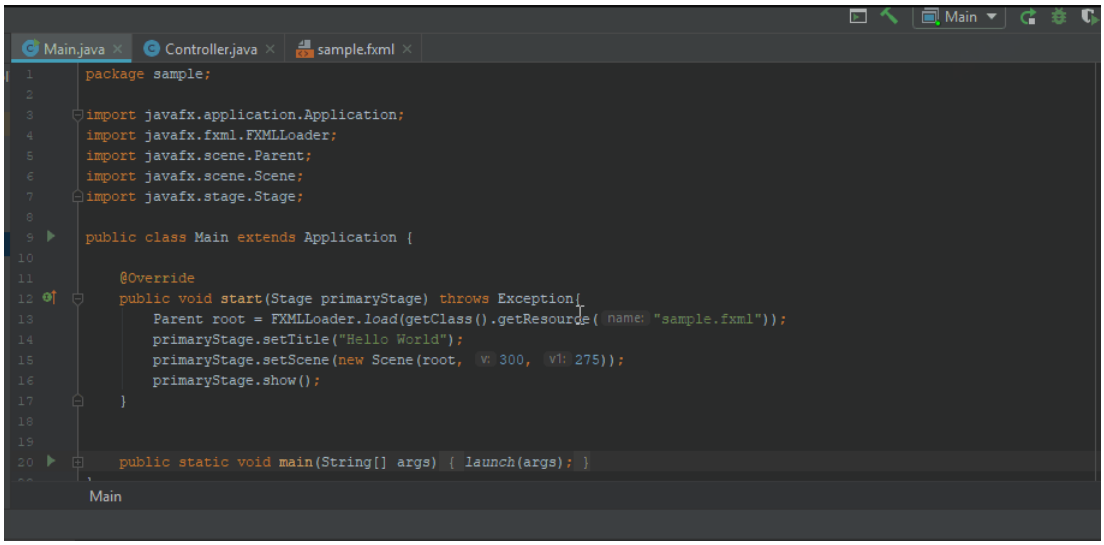

FIGURE 3.10 – projet Java FX Main

3.7.4 Scene Builder

JavaFXSceneBuilder (Scene Builder) vous permet de concevoir rapidement des interfaces utilisateur d'application JavaFX en faisant glisser un composant de l'interface utilisateur d'une bibliothèque de composants de l'interface utilisateur et en le déposant dans une zone d'affichage du contenu. Le code FXML de la mise en page de l'interface utilisateur que vous créez dans l'outil est automatiquement généré en arrièreplan.
Scene Builder peut être utilisé comme un outil de conception autonome, mais il peut également être utilisé avec des IDE Java pour que vous puissiez utiliser l'IDE pour écrire, construire et exécuter le code source du contrôleur que vous utilisez avec l'interface utilisateur de votre application. Bien que SceneBuilder soit plus étroitement intégré à l'EDINetBeans, il est également intégré aux autres EDI Java décrits dans ce document. L'intégration vous permet d'ouvrir un document FXML à l'aide de Scene Builder, d'exécuter les exemples Scene Builder et de générer un modèle pour le fichier source du contrôleur.

Figure 3.11 présente l'utilisation Java FX Scene Builder :

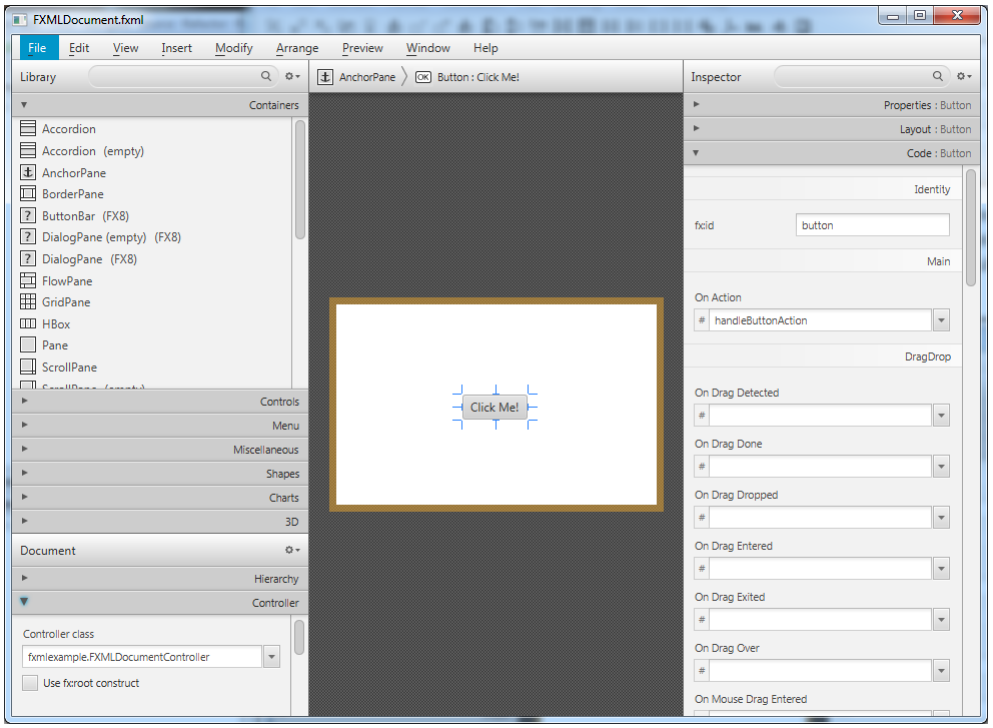

FIGURE 3.11 – Utilisation Java FX Scene Builder

3.7.5 BASE DE DONNEE

Derby Apache

Derby Apache est une base de données relationnelle open-source entièrement développée en Java par la fondation Apache. Derby a la particularité de pouvoir être utilisé comme gestionnaire de base de données embarqué dans une application Java. Ce qui rend inutile l'installation et la maintenance d'un serveur de base de données autonome. A l'inverse Derby supporte aussi un mode de fonctionnement client-serveur.

3.8 Conclusion

Ce chapitre a abordé l'aspect de l'implémentation de Book Finder. Nous avons discuté des technologies employées pour ensuite nous intéresser à l'évaluation du projet en général. Cette évaluation a abouti à des résultats prometteurs. Globalement, elle a montré qu'il existe une certaine concordance entre les deux algorithmes, que ce projet a atteint presque tous ses objectifs et que BF à réussit à générer des recommandations de livres adéquats.

Conclusion et perspectives

Dans le cadre de ce mémoire, nous avons présenté BF, un système de recommandation de livres, dont le but est de répondre au problème de surcharge d'informations. Nous avons commencé par présenter plusieurs techniques de filtrage étudiées dans la littérature, pour ensuite aborder spécifiquement la recommandation de livres.
Le travail présenté dans ce mémoire rentre dans le cadre du contexte de la recommandation de livres. Nous avons donné une vue générale sur ce domaine en introduisant la notion de recommandation sociale basant sur le filtrage collaboratif pour regrouper les utilisateurs qui partagent les mêmes intérêts dans un même groupe, En Calculant les similarités entre eux pour la prédiction des notes manquantes pour faire des recommandations des meilleures notes prédites. Dans le présent travail nous avons réalisé une application qui retourne une liste ordonnée de livres qui ont eu des notes supérieures à 3 à un utilisateur cible, et ceux en se basant sur deux algorithmes : le recuit simulé et NSGA2 pour le clustreing. Notre objectif été donc, de construire un système de recommandation de livres fiable qui donne des résultats pertinents dans un temps fini à l'aide des métaheuristiques.

Bibliographie

[Aci+07] Silvana Aciar et al. « Informed recommender : Basing recommendations on consumer product reviews ». In : *IEEE Intelligent systems* 22.3 (2007), p. 39-47 (cf. p. 9).

[ATH03] Stuart Andrews, Ioannis Tsochantaridis et Thomas Hofmann. « Support vector machines for multiple-instance learning ». In : *Advances in neural information processing systems*. 2003, p. 577-584 (cf. p. 12).

[AVS04] Kamal Ali et Wijnand Van Stam. « TiVo : making show recommendations using a distributed collaborative filtering architecture ». In : *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, p. 394-401 (cf. p. 12).

[Ban+98] Wolfgang Banzhaf et al. *Genetic programming*. Springer, 1998 (cf. p. 21).

[BH98] Heckermen Breese et D Heckerman. « Kadie. Empirical analysis of predictive algorithms for collaborative filtering ». In : *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence,(Madison, WI, 1998)*. 1998, p. 43-52 (cf. p. 11, 12).

[BKV07] Robert Bell, Yehuda Koren et Chris Volinsky. « Modeling relationships at multiple scales to improve accuracy of large recommender systems ». In : *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007, p. 95-104 (cf. p. 9, 12).

[BNJ03] David M Blei, Andrew Y Ng et Michael I Jordan. « Latent dirichlet allocation ». In : *Journal of machine Learning research* 3.Jan (2003), p. 993-1022 (cf. p. 12).

[BP99] Daniel Billsus et Michael J Pazzani. « A hybrid user model for news story classification ». In : *Um99 user modeling*. Springer, 1999, p. 99-108 (cf. p. 8).

[BS+12] S Binitha, S Siva Sathya et al. « A survey of bio inspired optimization algorithms ». In : *International journal of soft computing and engineering* 2.2 (2012), p. 137-151 (cf. p. 17).

[BT94] Roberto Battiti et Giampietro Tecchiolli. « The reactive tabu search ». In : *ORSA journal on computing* 6.2 (1994), p. 126-140 (cf. p. 20).

[Bul05] Larry Bull. *Foundations of learning classifier systems*. T. 183. Springer Science & Business Media, 2005 (cf. p. 22).

[Bur02] Robin Burke. « Hybrid recommender systems : Survey and experiments ». In : *User modeling and user-adapted interaction* 12.4 (2002), p. 331-370 (cf. p. 3).

[Bur+07] Edmund K Burke et al. « A graph-based hyper-heuristic for educational timetabling problems ». In : *European Journal of Operational Research* 176.1 (2007), p. 177-192 (cf. p. 3).

[Bur+59] Sir Frank Macfarlane Burnet et al. « The clonal selection theory of acquired immunity ». In : (1959) (cf. p. 28).

[BYRN+99] Ricardo Baeza-Yates, Berthier Ribeiro-Neto et al. *Modern information retrieval*. T. 463. ACM press New York, 1999 (cf. p. 6).

[CBC08] Iván Cantador, Alejandro Bellogín et Pablo Castells. « Ontology-based personalised and context-aware recommendations of news items ». In : *2008 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology*. T. 1. IEEE. 2008, p. 562-565 (cf. p. 9).

[Cha+02] Joyce Chai et al. « Natural language assistant : A dialog system for online product recommendation ». In : *AI Magazine* 23.2 (2002), p. 63-63 (cf. p. 1).

[CK95] Djurdje Cvijović et Jacek Klinowski. « Taboo search : an approach to the multiple minima problem ». In : *Science* 267.5198 (1995), p. 664-666 (cf. p. 20).

[Cor+99] David Corne et al. *New ideas in optimization*. McGraw-Hill Ltd., UK, 1999 (cf. p. 21).

[DC05] Arnab Das et Bikas K Chakrabarti. *Quantum annealing and related optimization methods*. T. 679. Springer Science & Business Media, 2005 (cf. p. 23).

[DCT03] Leandro Nunes De Castro et Jon I Timmis. « Artificial immune systems as a novel soft computing paradigm ». In : *Soft computing* 7.8 (2003), p. 526-544 (cf. p. 28).

[Deb+00] Kalyanmoy Deb et al. « A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization : NSGA-II ». In : *International conference on parallel problem solving from nature*. Springer. 2000, p. 849-858 (cf. p. 22).

[DG+08] Marco De Gemmis et al. « Integrating tags in a semantic content-based recommender ». In : *Proceedings of the 2008 ACM conference on Recommender systems*. 2008, p. 163-170 (cf. p. 9).

[DK04] Mukund Deshpande et George Karypis. « Item-based top-n recommendation algorithms ». In : *ACM Transactions on Information Systems (TOIS)* 22.1 (2004), p. 143-177 (cf. p. 11).

[DMC96] Marco Dorigo, Vittorio Maniezzo et Alberto Colorni. « Ant system : optimization by a colony of cooperating agents ». In : *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 26.1 (1996), p. 29-41 (cf. p. 27).

[EK95] Russell Eberhart et James Kennedy. « Particle swarm optimization ». In : *Proceedings of the IEEE international conference on neural networks*. T. 4. Citeseer. 1995, p. 1942-1948 (cf. p. 26).

[FC95] David B Fogel et Evolutionary Computation. « Toward a New Philosophy of Machine Intelligence ». In : *IEEE Evolutionary Computation* (1995) (cf. p. 20, 21).

[Fer06] Cândida Ferreira. *Gene expression programming : mathematical modeling by an artificial intelligence*. T. 21. Springer, 2006 (cf. p. 22).

[Gee09] Zong Woo Geem. *Harmony search algorithms for structural design optimization*. T. 239. Springer, 2009 (cf. p. 24).

[Glo99] Fred Glover. « Scatter search and path relinking ». In : *New ideas in optimization* 297316 (1999) (cf. p. 20).

[Gol94] David E Goldberg. « Genetic and evolutionary algorithms come of age ». In : *Communications of the ACM* 37.3 (1994), p. 113-120 (cf. p. 21).

[Grč+06] Miha Grčar et al. « kNN versus SVM in the collaborative filtering framework ». In : *Data Science and Classification*. Springer, 2006, p. 251-260 (cf. p. 12).

[Gre96] JR Greene. « Population-based incremental learning as a simple, versatile tool for engineering optimization ». In : *Proceedings of the first international conf. on ec and applications*. 1996, p. 258-269 (cf. p. 25).

[Hil+95] Will Hill et al. « Recommending and evaluating choices in a virtual community of use ». In : *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1995, p. 194-201 (cf. p. 10).

[HLG99] Georges R Harik, Fernando G Lobo et David E Goldberg. « The compact genetic algorithm ». In : *IEEE transactions on evolutionary computation* 3.4 (1999), p. 287-297 (cf. p. 24, 25).

[HM01] Pierre Hansen et Nenad Mladenović. « Variable neighborhood search : Principles and applications ». In : *European journal of operational research* 130.3 (2001), p. 449-467 (cf. p. 19).

[HS87] J Pirie Hart et Andrew W Shogan. « Semi-greedy heuristics : An empirical study ». In : *Operations Research Letters* 6.3 (1987), p. 107-114 (cf. p. 19).

[JD07] Zhou Ji et Dipankar Dasgupta. « Revisiting negative selection algorithms ». In : *Evolutionary Computation* 15.2 (2007), p. 223-251 (cf. p. 28).

[Kar63] Dean C Karnopp. « Random search techniques for optimization problems ». In : *Automatica* 1.2-3 (1963), p. 111-121 (cf. p. 18).

[Koh90] Teuvo Kohonen. « The self-organizing map ». In : *Proceedings of the IEEE* 78.9 (1990), p. 1464-1480 (cf. p. 30).

[Kon+97] Joseph A Konstan et al. « GroupLens : applying collaborative filtering to Usenet news ». In : *Communications of the ACM* 40.3 (1997), p. 77-87 (cf. p. 10).

[Kor08] Yehuda Koren. « Factorization meets the neighborhood : a multifaceted collaborative filtering model ». In : *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2008, p. 426-434 (cf. p. 12).

[KS05] Natalio Krasnogor et James Smith. « A tutorial for competent memetic algorithms : model, taxonomy, and design issues ». In : *IEEE Transactions on Evolutionary Computation* 9.5 (2005), p. 474-488 (cf. p. 24).

[KW71] J Kregting et RC White. *Adaptive random search*. University of Technology. Department of Electrical Engineering. Group . . ., 1971 (cf. p. 18).

[LB10] Mark Levy et Klaas Bosteels. « Music recommendation and the long tail ». In : *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys, 2010, Barcelona, Spain*. Citeseer. 2010 (cf. p. 11).

[Lie95] Henry Lieberman. « Letizia ». In : *An agent that assists web browsing* (1995), p. 924 (cf. p. 8).

[LK92] XZ Li et KH Kuo. « The structural model of Al-Mn decagonal quasicrystal based on a new AI-Mn approximant ». In : *Philosophical Magazine B* 65.3 (1992), p. 525-533 (cf. p. 1).

[Lou01] HR Loureno. « A beginner's introduction to iterated local search ». In : (2001) (cf. p. 19).

[LSY03] Greg Linden, Brent Smith et Jeremy York. « Amazon. com recommendations : Item-to-item collaborative filtering ». In : *IEEE Internet computing* 7.1 (2003), p. 76-80 (cf. p. 1, 11, 12).

[MB02] Gaurav Marwah et Lois Boggess. « Artificial immune systems for classification : Some issues ». In : *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*. T. 1. Citeseer. 2002, p. 149-153 (cf. p. 28).

[MBF11] David Martens, Bart Baesens et Tom Fawcett. « Editorial survey : swarm intelligence for data mining ». In : *Machine Learning* 82.1 (2011), p. 1-42 (cf. p. 18).

[MG99] Dunja Mladenic et Marko Grobelnik. « Feature selection for unbalanced class distribution and naive bayes ». In : *ICML*. T. 99. 1999, p. 258-267 (cf. p. 8).

[Min69] Marvin Minsky. *S. papert. Perceptrons : An Introduction to Computational Geometry*. 1969 (cf. p. 29).

[MKP03] Harry Mak, Irena Koprinska et Josiah Poon. « Intimate : A web-based movie recommender using text categorization ». In : *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*. IEEE. 2003, p. 602-605 (cf. p. 8).

[MM09] Frank McSherry et Ilya Mironov. « Differentially private recommender systems : Building privacy into the netflix prize contenders ». In : *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2009, p. 627-636 (cf. p. 3).

[MS00] Bernardo Magnini et Carlo Strapparava. « Experiments in word domain disambiguation for parallel texts ». In : *Proceedings of the ACL-2000 workshop on Word senses and multi-linguality-Volume 8*. Association for Computational Linguistics. 2000, p. 27-33 (cf. p. 9).

[MS01] Bernardo Magnini et Carlo Strapparava. « Improving user modelling with content-based techniques ». In : *International Conference on User Modeling*. Springer. 2001, p. 74-83 (cf. p. 9).

[Muk+01] Rajatish Mukherjee et al. « Movies2go : an online voting based movie recommender system ». In : *Proceedings of the fifth international conference on Autonomous agents*. 2001, p. 114-115 (cf. p. 8).

[Mul+02] Sibylle D Muller et al. « Optimization based on bacterial chemotaxis ». In : *IEEE transactions on Evolutionary Computation* 6.1 (2002), p. 16-29 (cf. p. 27).

[OR01] Michael O'Neill et Conor Ryan. « Grammatical evolution ». In : *IEEE Transactions on Evolutionary Computation* 5.4 (2001), p. 349-358 (cf. p. 22).

[Pat07] Arkadiusz Paterek. « Improving regularized singular value decomposition for collaborative filtering ». In : *Proceedings of KDD cup and workshop*. T. 2007. 2007, p. 5-8 (cf. p. 12).

[PG02] Martin Pelikan et David E Goldberg. *Bayesian optimization algorithm : From single level to hierarchy*. University of Illinois at Urbana-Champaign Urbana, IL, 2002 (cf. p. 25).

[Pha+05] DT Pham et al. « The bees algorithm ». In : *Technical Note, Manufacturing Engineering Centre, Cardiff University, UK* (2005) (cf. p. 27).

[PM98] Martin Pelikan et Heinz Mühlenbein. « Marginal distributions in evolutionary algorithms ». In : *Proceedings of the International Conference on Genetic Algorithms Mendel*. T. 98. Citeseer. 1998, p. 90-95 (cf. p. 25).

[Res+] P Resnick et al. « GroupLens 1994 :"An open architecture for collaborative filtering of netnews" ». In : *Proceedings of the 1994 Computer Supported Cooperative Work Conference* (cf. p. 11).

[Rey94] Robert G Reynolds. « An introduction to cultural algorithms ». In : *Proceedings of the third annual conference on evolutionary programming*. World Scientific. 1994, p. 131-139 (cf. p. 24).

[RK04] RY Rubinstein et DP Kroese. « The Cross-Entropy Method : A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning. Springer Science & Business Media ». In : (2004) (cf. p. 25).

[Rum+95] David E Rumelhart et al. « Backpropagation : The basic theory ». In : *Backpropagation : Theory, architectures and applications* (1995), p. 1-34 (cf. p. 30).

[RV97] Paul Resnick et Hal R Varian. « Recommender systems ». In : *Communications of the ACM* 40.3 (1997), p. 56-58 (cf. p. 3).

[Sal89] Gerard Salton. « Automatic text processing. 1989 ». In : *ed : Addison Wesley* (1989) (cf. p. 7).

[SJ80] Philip D Straffin Jr. *Topics in the Theory of Voting*. ERIC, 1980 (cf. p. 8).

[SM93] Beerud Sheth et Pattie Maes. « Evolving agents for personalized information filtering ». In : *Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications*. IEEE. 1993, p. 345-352 (cf. p. 8).

[SM95] Upendra Shardanand et Pattie Maes. « Social information filtering : algorithms for automating "word of mouth" ». In : *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1995, p. 210-217 (cf. p. 10).

[SMH07] Ruslan Salakhutdinov, Andriy Mnih et Geoffrey Hinton. « Restricted Boltzmann machines for collaborative filtering ». In : *Proceedings of the 24th international conference on Machine learning*. 2007, p. 791-798 (cf. p. 12).

[SS98] Anna Stefani et C Strappavara. « Personalizing access to web sites : The siteif project ». In : *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT*. T. 98. 1998, p. 20-24 (cf. p. 9).

[Tal09] El-Ghazali Talbi. *Metaheuristics : from design to implementation*. T. 74. John Wiley & Sons, 2009 (cf. p. 18).

[TV97] Edward Tsang et Chris Voudouris. « Fast local search and guided local search and their application to British Telecom's workforce scheduling problem ». In : *Operations Research Letters* 20.3 (1997), p. 119-127 (cf. p. 19).

[Ven75] W Vent. « Rechenberg, Ingo, Evolutionsstrategie—Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. 170 S. mit 36 Abb. Frommann-Holzboog-Verlag. Stuttgart 1973. Broschiert ». In : *Feddes Repertorium* 86.5 (1975), p. 337-337 (cf. p. 21).

[Wei+92] Andreas S Weigend et al. « Introduction to the theory of neural computation ». In : *Artificial Intel.* 62.XEROX-0004-3702-93 (1992), p. 93-111 (cf. p. 30).

[WJ94] Marty Wattenberg et Ari Juels. « Stochastic hillclimbing as a baseline method for ». In : (1994) (cf. p. 19).

[Yam05] Kamal Yammine. « DIA : un système de recommandation de livres dans un contexte pédagogique ». In : (2005) (cf. p. 1).

[ZK04] C Lawrence Zitnick et Takeo Kanade. « Maximum entropy for collaborative filtering ». In : *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press. 2004, p. 636-643 (cf. p. 12).

[ZT99] Eckart Zitzler et Lothar Thiele. « Multiobjective evolutionary algorithms : a comparative case study and the strength Pareto approach ». In : *IEEE transactions on Evolutionary Computation* 3.4 (1999), p. 257-271 (cf. p. 22).