REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOHAMED BOUDIAF - M'SILA
FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE

DEPARTEMENT D'INFORMATIQUE

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Informatique Décisionnel et Optimisation

Par : ZROUG Rihab hibat-Errahmane

SUJET

Développement et implémentation d'un algorithme génétique pour la détection de communautés dans les réseaux sociaux

Soutenu publiquement le : 13/07/2019 devant le jury composé de :

Mr. LAKEHAL AYAT Raouf Ouanis	**Université de M'sila Président**
Dr. LAMICHE Chaabane	**Université de M'sila Rapporteur**
Mr. LOUCIF Hamza	**Université de M'sila Examinateur**

Promotion : 2018 /2019

Dédicace

Je dédie ce travail à mes parents qui m'ont toujours offert le bonheur. Je le dédie à ma sœur, mes frères, les petits enfants de la famille, et à ma grande famille, A Tous mes amis et mes collègues. A la promo 2018/2019 d'informatique. Enfin, à toutes celles et tous ceux qui ont contribué de près ou de loin à l'accomplissement de ce travail

Remerciements

Je tiens avant tout à remercier Dieu le tout puissant de m'avoir donné la force et la volonté pour achever ce modeste travail.

Je remercie Dr. LAMICHE Chaabane mon encadreur d'avoir bien dirigé ce travail, avec ses judicieux conseils dont il a fait preuve durant l'élaboration de notre étude.

Je tiens à remercier Dr. BENAZI Makhlouf qui m'a encadré tout au long de ma mémoire. Son aide a été déterminante dans l'aboutissement de mes travaux. Leur façon de penser, sa gentillesse, sa disponibilité et sa force de travail en font des exemples à suivre pour moi.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail Et de l'enrichir par leurs propositions.

Un grand merci à ma famille, en particulier mes parents, mes frères et ma sœur et tous mes amis pour leur soutien moral et leurs encouragements.

Table des Matières

Chapitre 2 : Revue sur les métaheuristique d'optimisation combinatoire

Chapitre 3 : Algorithm génétique appliqué a la detection de communautés dans les réseaux sociaux

Conclusion générale

Bibliographie

Liste des Figures

Liste des Algorithmes

Liste des Tableaux

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Les problèmes d'optimisation occupent actuellement une place importante dans la communauté des ingénieurs et scientifiques. En effet, ce type de problèmes intervient dans plusieurs domaines, telles que la conception des systèmes mécaniques, traitement d'images, l'électronique et la recherche opérationnelle.

Les communautés jouant un rôle crucial dans les réseaux, l'identification de leurs structures a suscité un vif intérêt dans les sociétés de physique et d'informatique. De nombreuses méthodes et algorithmes ont été proposés jusqu'à présent pour révéler la structure de la communauté sous-jacente dans des réseaux complexes. Le succès d'un algorithme de détection de communauté dans la recherche de communautés dépend de la façon dont il définit une communauté. Une définition quantitative populaire appelée modularité de réseau Q, proposée par Girvan et Newman, est largement utilisée comme mesure de qualité pour évaluer la répartition d'un réseau en communautés. La recherche de la valeur de modularité la plus élevée est un problème NP-complet, car l'espace de toutes les partitions possibles augmente plus rapidement que toute puissance de la taille du système. Pour cette raison, de nombreux algorithmes adoptent différentes stratégies heuristiques pour optimiser cette métrique. Cependant, ces algorithmes ont généralement de grandes complexités de calcul et ne conviennent pas aux réseaux à grande échelle. Ainsi, un algorithme hautement efficace est souhaité.

L'objectif de cette étude est de proposer une méthode approchée efficace capable à résoudre les problèmes de détection de communautés dans les réseaux sociaux. Et pour aboutir à notre but notre mémoire est organisée comme suit :

Dans le Chapitre 1 nous présenterons les aspects fondamentaux de la théorie des graphes pertinents pour notre étude.

Dans le Chapitre 2 on va définir les problèmes d'optimisation combinatoire, les heuristiques et les métaheuristiques, et la définition de la notion de complexité des algorithmes et des problèmes (classe NP, classe NP-complet, classe NP-difficile). On a présenté ensuite, quelques méthodes de la résolution exactes et approchées.

Partie pratique :

Dans le Chapitre 3 qui concerne la résolution du problème de notre sujet d'étude. Tout d'abord nous avons défini le problème de la détection de communautés. On a appliqué l'algorithme génétique pour résoudre notre problème et nous avons quelques résultats expérimentaux obtenus par cet algorithme.

Enfin, on termine par une conclusion générale.

CHAPITRE 1

NOTIONS SUR LA DETECTION DE COMMUNAUTES DANS LES RESEAUX SOCIAUX

CHAPITRE 1 NOTIONS SUR LA DETECTION DE COMMUNAUTES DANS LES RESEAUX SOCIAUX

1. Introduction

Beaucoup de systèmes complexes dans divers domaines comme la biologie, l'informatique, la linguistique, le commerce, etc., peuvent être représentés de manière abstraite par des réseaux. Une des caractéristiques communes que l'on retrouve dans de nombreux réseaux concerne l'existence de zones plus densément connectées que d'autres. Ces zones sont habituellement appelées communautés et correspondent intuitivement à des groupes de nœuds plus fortement connectés entre eux qu'avec les autres nœuds du réseau. La détection de ces zones dites communautaires est un outil important pour la compréhension des structures et des fonctionnements des grands réseaux.

De tels réseaux peuvent être modélisés en termes de graphes, où un nœud représente un membre individuel du système, et une arête représente un lien entre les nœuds selon une relation bien déterminée du système. Ainsi, le concept de graphe est généralement utilisé comme modèle de représentation dès que les données sont intrinsèquement liées. Cette structure de données permet en effet de modéliser efficacement les relations entre différents acteurs d'un système complexe. Ce chapitre tente de rapporter l'essentiel concernant la notion de communauté telle qu'elle a été abordée par différents auteurs et présentera le cadre formel de la détection de communautés. Mais avant cela, il convient d'introduire au préalable des cas de modélisation par des graphes, ainsi que certaines notions et définitions utiles de la théorie des graphes.

2. Définition des réseaux sociaux

Un réseau social est un réseau ayant été créé par des interactions sociales [13]. Il existe déférentes catégories de réseaux sociaux. Dans les graphes représentant ces réseaux, les individus correspondent aux nœuds et leurs interactions correspondent aux arêtes. On distingue les interactions bilatérales, nécessitant l'accord des deux individus pour être établie, des interactions unilatérales.

2.1 Les réseaux sociaux web

Permettent d'établir explicitement des relations entre les utilisateurs. Les relations dans certains de ces sites web sont bilatérales. C'est par exemple le cas de Facebook. Dans d'autres réseaux comme Twitter ou YouTube, la relation sociale est établie de manière unilatérale

parle follow (le fait de suivre les publications de quelqu'un). Deux individus sont voisins dans le graphe associé si une relation existe entre les deux.

2.2 Les réseaux de communication

Sont formés par des transmissions d'information entre individus. On retrouve la même distinction entre unilatéralité (comme pour les emails set les SMS) et bilatéralité (appels téléphonique set vidéo conférences). Deux individus sont voisins dans le graphe associé s'ils sont communiqués n utilisant ce réseau.

2.3 Les réseaux de collaboration

Correspondent à des individus ayant travaillé ensemble sur un sujet. On y trouve par exemple des réseaux d'acteur s ayant tourné des films ensembles ou des scientifiques ayant coécrit des articles. Les relations dans ce type de réseau sont bilatérales. Deux individus sont voisins dans le graphe associé s'ils sont collaborés.

3. Les graphes

3.1 Définition

Les définitions suivantes quelques notions manipulées par la théorie des graphes pour les réseaux sociaux :

- Sommet (nœud) est l'unité de base d'un réseau, il représente une ressource. Dans un réseau social on parle d'acteur.

- Arête est une connexion entre deux sommets. On parle également d'arc ou de lien.

3.2 Représentation d'un réseau

- Arête est orientée si elle ne s'utilise que dans une seule direction. Arête non orientée pour une arête qui s'utilise dans les deux directions.
- Graphe est défini par un ensemble de sommets et un ensemble d'arêtes.
- Degré d'un sommet est le nombre de ses arêtes adjacentes.
- Chemin est une séquence d'arêtes qui relie deux sommets.

Un réseau est traditionnellement défini par un graphe G= (V, E) dans lequel V est l'ensemble des nœuds et E l'ensemble des liaisons du réseau. E est un ensemble de couples

de nœuds tel que E⊆V×V. Ainsi, soient v_i et v_j deux nœuds du réseau, v_i, v_j∈V, si e=(v_i, v_j)∈E, alors il existe une liaison entre le nœud vi et le nœud v_j dans G. Les nœuds vi et v_j sont dits adjacents, ou encore connectés ou voisins. Le nombre total de nœuds dans le réseau est égal au cardinal de l'ensemble V(n=|V|). Le nombre de nœuds est souvent utilisé pour désigner la taille du réseau. Le nombre de liens (arêtes) est égal au cardinal de l'ensemble E(m=|E|). On peut classifier les graphes suivant plusieurs critères. Parmi les grandes familles des réseaux on distingue [23] :

- Les réseaux non-orientés

Dans un réseau non-orienté, l'ensemble des liens E est un ensemble de paires de nœuds (donc non ordonnées). Si les nœuds vi et v_j sont connectés, il existe également un lien entre v_j et v_i. Pour un réseau contenant n nœuds (n=|V|), le nombre de liens maximal est de (n×(n−1)) /2.

- Les réseaux orientés

Un réseau orienté est un réseau dont l'ensemble des liens E regroupe des couples de nœuds (donc ordonnés), appelés liens orientés. Dans un réseau orienté, la présence d'un lien e1=(v_i, v_j) entre les nœuds v_i et v_j n'implique pas nécessairement l'existence d'un lien e2=(v_i, v_j). Dans les représentations graphiques de tels réseaux, l'orientation du lien est généralement représentée par une flèche indiquant la direction du lien. Pour un réseau contenant n nœuds, le nombre de liens maximal est de n ×(n−1).

4. Représentation d'un réseau

Un certain nombre de représentations existent pour décrire un graphe. On distingue principalement la représentation par matrice d'adjacence et par liste d'adjacence [1].

4.1 Matrice d'adjacence

A tout graphe d'ordre non associe une matrice M de n lignes et n colonnes dont les éléments sont notés M_{ij}. Pour un graphe non orienté

G= (V, E), M (i, j) {1 si (i, j) ∈ E ; 0 sinon.

4.2 Liste d'adjacence

Pour un graphe G= (V, E), la liste d'adjacence associe à ce graphe permet de représenter pour chaque sommet, l'ensemble de ses successeurs. Tous les arcs émanant d'un même sommet, sont liés entre eux dans une liste. A chaque arc sont donc associes l'extrémité terminale et le pointeur au prochain sommet de la liste.

5. Structure communautaire

Intuitivement, sur un réseau social, on peut facilement être convaincu que les individus se regroupent naturellement en communautés, qui sont des groupes d'amis, de collègues de travail, de loisirs, familiaux, etc. Donner une définition formelle de ce qu'est une communauté est difficile et toutes les définitions existantes sont restrictives. Cependant, une définition largement acceptée, liée à la topologie du réseau, considère une communauté comme un sous-graphe dont les sommets sont plus liés entre eux qu'avec le reste du réseau.

6. Définition de la communauté

Les communautés sont également appelées groupes, clusters, sous-graphes, ou des modules dans des contextes différents. Elle est l'une des tâches fondamentales dans l'analyse des réseaux sociaux. Trouver une communauté dans un réseau social est d'identifier un ensemble des nœuds tels qu'ils interagissent les uns avec les autres plus fréquemment qu'avec les nœuds en dehors du groupe [23].

La détection de Communautés peut faciliter des tâches informatiques sociales, la détection de communautés est appliquée dans de nombreuses applications du monde réel. Par exemple, le regroupement des clients ayant des intérêts similaires dans les médias sociaux rend des recommandations simples et efficaces qui exposent les clients à une large gamme d'articles pour améliorer le taux de réussite de la transaction. Les communautés peuvent également être utilisées pour compresser un réseau énorme, résultant en un réseau plus petit. En d'autres termes, la résolution de problèmes est accomplie au niveau du groupe, au lieu du niveau de nœud. Dans le même esprit, un vaste réseau peut être visualisé à des résolutions différentes, offrant une solution intuitive pour l'analyse de réseau.

7. Définition formelle du problème

Le but de la détection de communautés dans un graphe G= (V, E) est de trouver une partition P=c_1, ..., c_k de l'ensemble V des nœuds. Les communautés c_i sont donc disjointes ($\forall$ i $\neq$ j, $c_i \cap c_j = \emptyset$) et recouvrent V($\bigcup_i c_i$=V). Notons que dans la plupart des cas nous ne connaissons a priori ni le nombre ni la taille des communautés qui composent la partition recherchée. Comme le montre cette définition, on n'a pas a priori de définition de la notion de communauté, alors on utilise des fonctions de qualité pour l'évaluation de la qualité de la partition. La fonction de qualité Q donne un score à toute partition qui mesure à quel point les communautés de cette partition vérifient des critères caractéristiques des communautés. En particulier, ces fonctions de qualité Q(P) tiennent généralement compte des densités de liens à l'intérieur et entre les communautés. Il existe plusieurs fonctions de qualité pour la détection de communautés, la plus communément utilisée étant la modularité. Finalement cette définition impose que les communautés ci sont disjointes, mais en pratique rien n'empêche l'existence de communautés qui se chevauchent.

8. La Modularité

La modularité est une mesure pour la qualité d'un partitionnement des nœuds d'un graphe, ou réseau, en communautés. Elle est principalement utilisée en analyse des réseaux sociaux. Elle a été introduite par M. Newman. C'est aussi une fonction d'optimisation pour certaines tâches de détection de communautés dans les graphes.

Le principe est qu'un bon partitionnement d'un graphe implique un nombre d'arêtes intra-communautaires important et un nombre d'arêtes intercommunautaires faible [28].

9. Algorithmes de détection de communauté

9.1 Stratégies de partitionnement

Newman et Girvan ont introduit la modularité qui inspira une série de méthodes basées sur celle-ci. Il a été prouvé par Brandes et al. Que trouver la partition optimale en modularité est un problème NP-Complet, ce qui rend le passage à l'échelle extrêmement coûteux. Des approches d'optimisation gloutonne ont étés développées pour apporter une alternative acceptable en temps de calcul, comme l'algorithme de Louvain. D'autres approches existent, comme la méthode du recuit simulé ou l'analyse spectrale [4].

La modularité réseau Q est une métrique de qualité pour l'évaluation du partitionnement D'un réseau en communautés. Elle a été largement utilisée dans les travaux sur les réseaux. Elle est fondée sur l'idée intuitive que les réseaux aléatoires ne renferment pas de structure communautaire. Par conséquent un sous-graphe forme une communauté si la distribution des liens entre ses nœuds n'est pas due au hasard. Elle se base alors sur la différence entre le nombre de liens à l'intérieur d'une communauté et le nombre de liens attendus à l'intérieur de cette communauté si les liens apparaissent aléatoirement dans le graphe tout en respectant la distribution des degrés des nœuds (« nul model »). Une communauté est d'autant mieux appréciée que sa proportion d'arêtes internes sera supérieure à sa proportion attendue d'arêtes. Ceci est synthétisé dans l'équation donnant la modularité, et que l'on cherchera toujours à maximiser [13] :

$$\mathrm{Q} = \sum_{Ci} e(Ci) \quad - a(Ci)^2$$

Où $e(Ci)$ représente la proportion d'arêtes ayant les deux extrémités dans Ci et $a(Ci)^2$ est la probabilité pour qu'une arête ait une extrémité dans Ci.

10. Méthodes de comparaison

Une méthode de comparaison (ou (extrinsic clustering evaluation metric)) est une fonction évaluant la proximité entre deux couvertures du même ensemble. La NMI (Normalized Mutual Information) est une méthode de comparaison basée sur la théorie de l'information. J'utilise la version introduite par Lancichinetti et al.

Les valeurs du NMI sont entre 0 et 1. Si deux partitions sont identiques alors NMI= 1.

Soit deux partitions A et B d'un réseau dans des communautés, soit C la matrice de confusion dont l'élément C_{ij} est le nombre de nœuds de la communauté i de la partition A qui font également partie de la communauté j de la partition B. L'information mutuelle normalisée I (A, B) est défini comme [6] :

$$I(A,B) = \frac{-2 \sum_{i=1}^{C_A} \sum_{j=1}^{C_B} C_{ij} \log(C_{ij} N / C_i C_j)}{\sum_{i=1}^{C_A} C_i \log(C_i/N) + \sum_{j=1}^{C_B} C_j \log(C_j/N)}$$

Où C_A (C_B) est le nombre de groupes dans la partition A (B), C_i. (C_j) est la somme des éléments de C dans la rangée i (colonne j) et N est le nombre de nœuds.

Si A = B, I (A, B) = 1. Si A et B sont complètement différents, I (A, B) = 0.

11. Conclusion

Nous avons consacré le premier volet de ce chapitre à l'introduction de certains concepts de base indispensables (issus de la théorie des graphes) afin de décrire avec aisance les notions utilisées dans les différents travaux rencontrés et l'approche que nous proposons. Nous nous sommes particulièrement penchés dans le deuxième volet de ce chapitre sur les différentes définitions relatives à la notion de communauté. Nous avons par ailleurs, décrit formellement le problème de la détection de communautés.

Nous avons discuté dans ce chapitre quelques concepts de la théorie des graphes. Cela devrait nous permettre de passer au prochain chapitre avec un arrière-plan acceptable sur les graphes en général.

CHAPITRE 2

REVUE SUR LES METAHEURISTIQUES D'OPTIMISATION COMBINATOIRE

1. Introduction

L 'optimisation est une discipline en plein essor qui entre en jeu dans beaucoup de domaines, comme dans la conception de circuits électroniques, la recherche opérationnelle, la biologie, mais aussi pour répondre aux besoins croissants des secteurs économiques et industriels (maximisation des performances, minimisation des coûts). Dans ce chapitre on va présenter les métaheuristiques pour la résolution de problèmes d'optimisation difficile.

2. Problème d'optimisation combinatoire

Un problème d'optimisation combinatoire est défini par un ensemble d'instances. A chaque instance du problème est associé un ensemble discret de solutions S, un sous-ensemble X de S représentant les solutions admissibles (réalisables) et une fonction de coût f (ou fonction objectif) qui assigne à chaque solution $s \in X$ le nombre réel (ou entier) f(s). Résoudre un tel problème (plus précisément une telle instance du problème) consiste à trouver une solution $s^* \in X$ optimisant la valeur de la fonction de coût f. Une telle solution s* s'appelle une solution optimale ou un optimum global. Nous avons donc la définition suivante :

Définition : Une instance I d'un problème de minimisation est un couple (X, f) où $X \subseteq S$ est un ensemble fini de solutions admissibles, et f une fonction de coût (ou objectif) à minimiser f : $X \rightarrow R$. Le problème est de trouver $s^* \in X$ tel que $f(s^*) \leq f(s)$ pour tout élément $s \in X$ [7].

Notons que d'une manière similaire, on peut également définir les problèmes de maximisation en remplaçant simplement ≤ par ≥. L'optimisation combinatoire trouve des applications dans des domaines aussi variés que la gestion, l'ingénierie, la conception, la production, les télécommunications, les transports, l'énergie, les sciences sociales et l'informatique elle-même [15].

3. Complexité des algorithmes

Un même problème peut généralement être résolu par plusieurs algorithmes, donc il faut comparer entre ces algorithmes, cette comparaison se base sur le temps de calcul et sur l'espace mémoire requis par l'algorithme. On appelle complexité en temps d'un algorithme dans le pire cas, la fonction F(n) qui donne une borne supérieure du nombre d'opérations élémentaires effectuées par l'algorithme lorsque la taille de l'entrée est n. Un algorithme dont le nombre d'opérations élémentaire nécessaires pour résoudre un exemple de taille n est une fonction polynomiale en n. Un algorithme est efficace (qu'il est<bon>) si le nombre des opérations

nécessaires pour résoudre un problème est borné par une fonction polynomiale d'un paramètre caractérisant la taille de ce problème. Il faut noter qu'un problème décision, est le problème pour lequel une solution est soit « oui » soit « non », alors qu'un problème d'optimisation, est le problème pour lequel on doit chercher à déterminer une solution qui optimise un critère. A chaque problème d'optimisation on peut associer un problème de décision [14].

Réduction polynomiale

Soient Π et Π' deux problèmes de décision. Π' se réduit polynomialement a Π et on note Π' α Π, si Π' est polynomial ou s'il existe un algorithme polynomial qui construit, à partir d'une donnée D'de Π', une donnée D de Π telle que, la réponse pour D'soit oui si est seulement si la réponse pour D est oui [9].

3.1 La classe NP

La classe NP est la classe des problèmes de décision qui peuvent être résolus par une machine de Turing non déterministe en temps polynomial. On distingue trois sous classes : La classe P qui contient les problèmes les plus faciles de NP, ils ont un algorithme polynomial pour les résoudre La classe des problèmes NP-complets contient les problèmes les plus difficiles de NP. Alors que la classe des problèmes ouverts englobe les problèmes de statut indéterminé [9].

3.2 Les problèmes NP-difficile

Un problème d'optimisation est dit NP-difficile sil le problème de décision associe est NP-complet [9].

4. Méthodes de résolution

Il existe deux grandes catégories des méthodes de résolution des problèmes d'optimisation combinatoire : les méthodes exactes et les méthodes approchées. Les méthodes exactes se basent sur l'énumération de l'ensemble des solutions potentielles et permettent d'obtenir la solution optimale. On a par exemple l'algorithme de séparation et évolution (Branche and Bound, programmation dynamique...), la taille du problème influe rationnellement sur l'efficacité (temps d'exécution) des méthodes de cette approche. Par contre les méthodes approchées, encore appelées heuristiques, permettent d'obtenir une bonne solution, qui n'est pas toujours la solution optimale, mais avec le temps raisonnable.

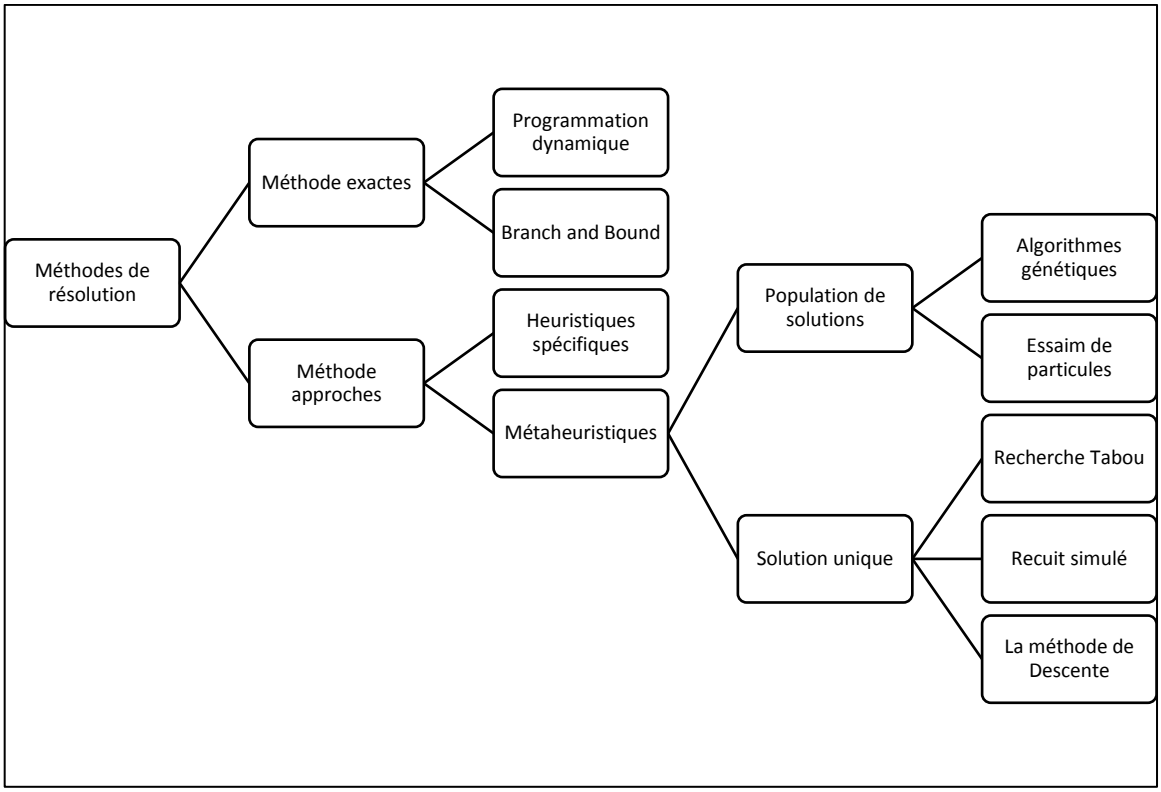


Figure 2.1 Classification des méthodes de résolution le problème d'optimisation

5. Algorithmes approchés d'optimisation

5.1. Heuristiques

L'utilisation de méthodes exactes n'est pas toujours possible pour un problème donné à cause d'un certain nombre de contraintes, tel que le temps de calcul souvent important ou bien la difficulté, voire l'impossibilité dans certains cas. Une heuristique est un algorithme qui fournit rapidement (en un temps polynomial) une solution approchée et réalisable, pas nécessairement optimale, pour un problème d'optimisation difficile. Cette méthode approximative est le contraire d'un algorithme exact qui donne une solution optimale pour un problème donné [2].

5.2. Métaheuristiques

Le terme métaheuristique a été inventé par Fred Glover en 1986, lors de la conception de la recherche Tabou. Une métaheuristique est un processus itératif qui subordonne et qui guide une heuristique, en combinant intelligemment plusieurs concepts pour explorer et exploiter tout l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions optimales, ou presque optimales [12].

Les métaheuristique sont utilisables pour résoudre différents problèmes d'optimisation, et ne nécessitant que peu de modifications pour qu'elle puisse s'adapter à un problème particulier. Elles ont donc pour objectif de pouvoir être programmées et testées rapidement sur un problème [3].

6. Présentation des principales métaheuristique

Il existe un grand nombre de métaheuristiques d'optimisations. Nous allons nous appuyer sur la classification qui distingue les méthodes de trajectoires (qui exploitent séquentiellement un seul voisinage) des méthodes basées sur des populations de solutions (qui exploitent plusieurs à la fois).

6.1. Méthode de trajectoire

Les méthodes à solution unique, plus connues sous le nom de méthodes de recherche locale ou encore méthodes de trajectoire, sont basées sur l'évolution d'une seule solution dans l'espace de recherche. Nous présentons les méthodes les plus utilisées, notamment la méthode de Descente, le Recuit Simule et la recherche Tabou.

6.1.1. Méthode de Descente (Hill Climbing)

Les Méthodes de Descente sont assez anciennes. Leur succès revient à leur simplicité et à leur rapidité [8]. Le principe consiste à choisir une solution s'dans le voisinage d'une solution s en améliorant la recherche tel que f (s') < f(s)[4].

On peut décider soit d'examiner toutes les solutions de voisinage et prendre la meilleure de toutes (ou prendre la meilleur trouvée), soit d'examiner un sous ensemble de voisinage. La méthode de Descente peut être décrite comme suit :

Algorithme 2.1 Méthode de descente

Procédure Descente Simple (solution initiale s)
Répéter :
Choisir s'dans N(s)
  Si f(s') <f(s) alors s→ s'
Jusqu'à ce que f(s') >= f(s), ∀s'∈S

6.1.2. Recuit Simulé (Simulated Annealing)

Cette méthode est issue d'une analogie entre le phénomène physique de refroidissement lent d'un corps en fusion, qui le conduit à un état solide, de basé énergie.

Il faut baisser lentement la température, en marquant des paliers suffisamment longs pour que le corps atteigne l'équilibre thermodynamique à chaque palier de température [21]. Pour les matériaux, cette basse énergie se manifeste par l'obtention d'une structure régulière, comme dans les cristaux et l'acier.

L'analogie exploitée par le recuit simulé consiste à considérer une fonction f à minimiser comme fonction d'énergie, et une solution s peut être considérée comme un état donné de la matière dont f(s) est l'énergie. Le recuit simulé se démarre par une solution initiale s qui peut être prise d'une manière aléatoire dans l'espace de recherche. A cette solution correspond une énergie initiale f(s). Une température initiale T élevée est également choisie. A chaque itération de l'algorithme une modification de la solution est effectuée. Cette modification entraine une variation Δ de l'énergie du système. Si cette variation est négative (c'est-à-dire qu'elle fait baisser l'énergie du système), elle est appliquée à la solution courante. Sinon, elle est acceptée avec une probabilité $e^{\Delta/T}$. Ce choix de l'exponentielle pour la probabilité s'appelle règle de Métropolis. On itère ensuite selon ce procédé en gardant la température constante [18]. L'algorithme Recuit Simulé est présenté ci-dessous :

Algorithme 2.2 Recuit Simulé

Choisir solution initiale s∈ S et température initiale T ;
Répéter
Choisir aléatoirement s'∈V(s), Δ= f(s') –f(s)
Si(Δ<0) alors
s : =s'
Sinon Choisir u un nombre aléatoire de [0,1] ;
Si (u < $e^{\Delta/T}$) alors
s : =s'
Fin si
Fin si
Jusqu'à condition d'arête satisfaite

6.1.3. Recherche Tabou (tabou search)

La méthode Tabou est une technique de recherche dont les principes ont été proposés pour la première fois par Fred Glover dans les années 80. A l'inverse du recuit simulé qui génère de manière aléatoire une seule solution voisine s'∈N(s) à chaque itération, tabou examine un échantillonnage de solutions de N(s) et retient la meilleure s' même si s' est plus mauvaise que s. La recherche Tabou ne s'arrête donc pas au premier optimum trouvé. Cependant, cette stratégie peut entraîner des cycles, par exemple un cycle de longueur 2 : s → s'→ s → s'... Pour empêcher ce type de cycle, on mémorise les k dernières configurations visitées dans une mémoire à court terme et on interdit tout mouvement qui conduit à une de ces configurations. Cette mémoire est appelée la liste taboue, une des composantes essentielles de cette méthode. Elle permet d'éviter tous les cycles de longueur inférieure ou égale à k. La valeur de k dépend du problème à résoudre et peut éventuellement évoluer au cours de la recherche [21].

Algorithme 2.3 Recherche Tabou
1: initialization s0 une solution initial s← s0, s*←s0, f*← f(s0) T = ∅ 2 : générer un sous-ensemble de solutions au voisinage de s s'∈V(s) tel que ∀x∈ V(s), f(x)≥ f (s') Si f (s') < f* alors s* ← s' et f*← f (s') Mise-à-jour de T 3 : Si la condition d'arrêt n'est pas satisfaite retour à l'étape2.

6.2. Méthode basée sur les populations de solution

Les métaheuristiques basées sur la manipulation d'une population des solutions essayent d'augmenter la qualité moyenne d'un ensemble des solutions via un mécanisme défini.

On distingue deux approches utilisées par les méthodes basées sur une population de solutions : la première est l'utilisation des mécanismes d'évolution naturels, cette approche est représentée par les algorithmes génétiques et autres. La deuxième est l'utilisation de

l'intelligence collective, cette approche est représentée par des colonies fourmis, essaimes particulaires et autres.

6.2.1. Algorithme génétique (Genetic Algorithm)

Les algorithmes génétiques (AG) sont des techniques de recherche et d'optimisation stochastique dérivées de la génétique et des mécanismes de la sélection naturelle et de l'évolution. Un algorithme génétique est un algorithme itératif de recherche globale dont le but est d'optimiser une fonction définie par l'utilisateur. Pour cela, les AGs fonctionnent avec une population regroupant un ensemble d'individus appelés chromosomes distribués sur l'entièreté de l'espace de recherche. Chaque chromosome est constitué d'un ensemble de gènes. Pour chaque individu on attribue une valeur calculée par la fonction d'adaptation (fitness). En pratique, à partir d'une population, des chromosomes sont générés d'une façon aléatoire lors de l'initialisation, dans chaque cycle d'opérations génétiques, une nouvelle population appelée génération est créée à partir des chromosomes de la population courante. Pour cela certains chromosomes appelés 'parents' sont sélectionnés afin d'élaborer les opérations génétiques. Les gènes de ces parents sont mixés et recombinés pour la production d'autres chromosomes appelés (enfants)constituant la nouvelle génération possédante, dans son ensemble, de meilleures solutions que la précédente.

6.2.1.1. Codage (représentation) d'un individu

Premièrement, il faut représenter les différents états possibles de la variable dont on cherche la valeur optimale sous une forme utilisable par un AE, c'est-à-dire par le codage approprié des solutions sous forme de chromosomes ou génotypes. Cela permet d'établir une connexion entre les valeurs de la variable et les individus de la population, de manière à imiter la connexion qui existe en biologie entre le génotype et le phénotype. Il existe principalement deux types de codage : le codage binaire qui est une représentation sous forme de chaîne binaire et le codage réel qui est une représentation directe des valeurs réelles de la variable [24].

6.2.1.2. Génération de la population initiale

L'AG démarre avec une population composée de N individus dans le codage retenu. Le choix des individus conditionne fortement la rapidité de l'algorithme. Si la position de l'optimum dans l'espace de recherche est totalement inconnue, il est intéressant

Que la population soit répartie sur tout l'espace de recherche. Si par contre des informations à priori sur le problème sont disponibles, il paraît évident de générer les individus dans un espace particulier afin d'accélérer la convergence. Disposant d'une population initiale souvent non homogène, la diversité de la population doit être entre tenue aux cours des générations afin d'explorer le plus largement possible l'espace de recherche. C'est le rôle des opérateurs de croisement et de mutation [26].

6.2.1.3. La fonction d'évaluation

L'évaluation permet de s'assurer que les individus performants seront conservés, alors que les individus peu adaptés seront progressivement éliminés de la population. L'évaluation d'un individu ne dépendant pas de celle des autres individus, pour calculer le coût d'un point de l'espace de recherche, on utilise une fonction d'évaluation f, cette fonction est souvent une transformation g de la fonction objectif $f(x) = g(\Phi(x))$, appelée aussi la fonction fitness, le résultat fourni par cette fonction va permettre de sélectionner ou de refuser un individu pour ne garder que les individus ayant le meilleur coût en fonction de la population courante [30].

6.2.1.4. Operateurs de sélection

Pour sélectionner les meilleurs individus (chromosomes) dans une population, il existe plusieurs types de sélection, chacune étant adaptée à un type particulier de problèmes. On en cite les plus classiques qui sont : la sélection roulette biaisée (Roulette wheel selection), la sélection rang (selection by rank), la sélection par tournoi (tournament selection) et la sélection uniforme (uniform selection).

- **Sélection par roulette**

La sélection par roulette peut être vue comme une sorte de roulette de casino sur laquelle sont mis tous les chromosomes de la population. La place accordée à chaque chromosome étant en relation avec sa valeur d'adaptation. Un exemple de roulette de sélection est représenté par la (figure 2.2).

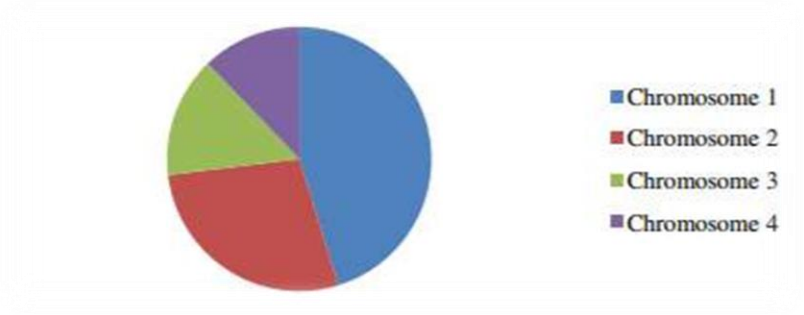


Figure 2.2 Schéma d'une roulette de sélection.

- **Sélection par rang**

La sélection par rang effectue d'abord un tri sur la population par rapport au fitness. Ensuite, à chaque chromosome est associe un rang en fonction de sa position. En conséquence, pour une population de N chromosomes, le plus mauvais aura le rang 1, et ainsi de suite jusqu'au meilleur chromosome qui aura le rang N. la sélection par rang et la sélection par roulette sont presque identiques, sauf que, avec la sélection par rang les populations sont en rapport avec le rang plutôt qu'avec la valeur de L'évolution. Ainsi, les probabilités finales seront calculées avec la formule : Rang / ∑ (des rangs) [1].

Tableau 2.1 Example de sélection par range pour 6 chromosomes.

Chromosomes	1	2	3	4	5	6	**Total**
Probabilités initiales	89%	5%	1%	4%	3%	2%	100%
Rang	6	5	1	4	3	2	21
Probabilités finales	29%	24%	5%	19%	14%	9%	100%

- **Sélection par tournoi**

On forme m paires de chromosomes dans une population de m chromosomes. Dans les paramètres de l'AG, on décide d'une probabilité de victoire du plus fort. Cette probabilité représente la chance que possède le meilleur chromosome de chaque paire d'être sélectionné. Cette probabilité doit être grande (entre 70% et 100%). A partir des m paires, on détermine ainsi m individus pour la reproduction [1].

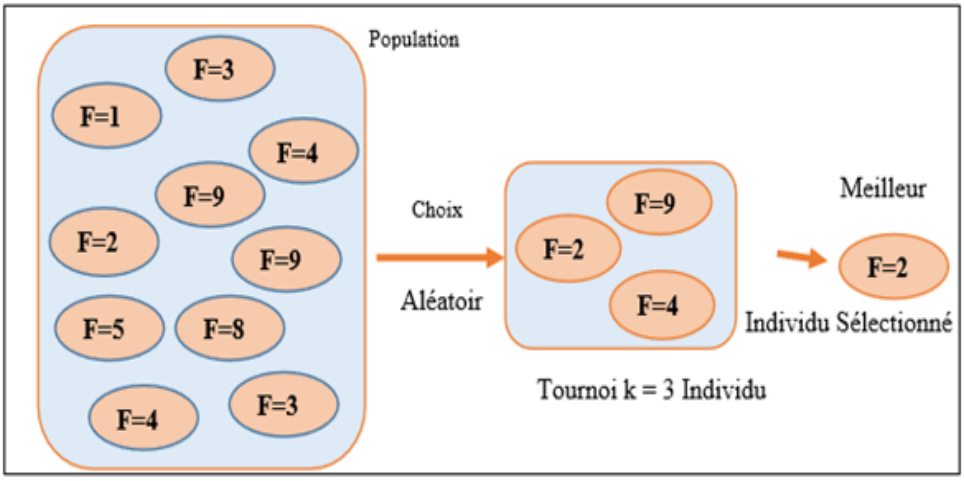


Figure 2.3 Représentions de la sélection par tournois

- **Sélection uniforme**

C'est une technique très simple qui consiste à sélectionner un individu C_i aléatoirement de la population P. La probabilité pi pour qu'un individu soit sélectionné est définie par :

Pi = 1 / taille-pop

6.2.1.5. Opérateurs de croisement

Le croisement est l'opérateur de recherche essentiel d'un AG. Il a lieu après la sélection. Il permet l'échange d'informations génétiques entre les individus après sélection, de façon aléatoire en une (ou plusieurs) position de césure (locus). Dans ce cas, deux des individus retenus de la sélection (parents) échangent une ou plusieurs parties de leurs génotypes, selon une probabilité Pc (souvent supérieure à 60%). Pour former deux individus différents de ceux d'origine.

Il existe différentes manières pour croiser : le croisement « à un point », « à deux points », « multipoint », « uniforme ».

- **Croisement a «1point »**

Le croisement "à un point" peut être illustré comme dans la figure. Il consiste à sélectionner aléatoirement un emplacement (une position) de césure et de permuter les parties droites des deux parents.

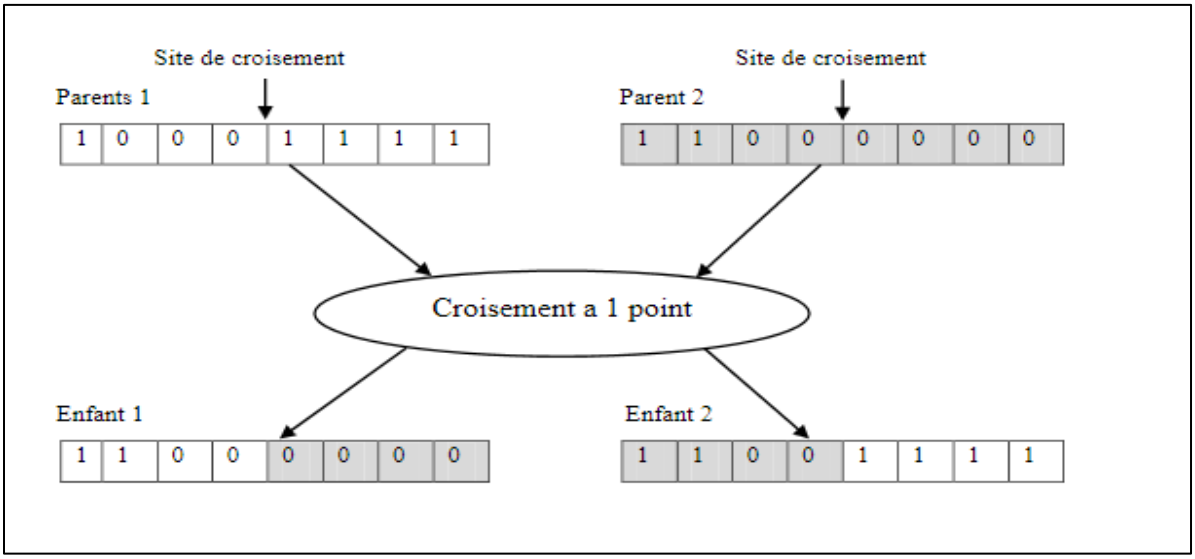


Figure 2.4 Croisements en un point

- **Croisement a «2 point »**

Ce croisement diffère du premier, tout simplement, du fait qu'il s'effectue en deux points de coupure, comme le montre la figure 2.5.

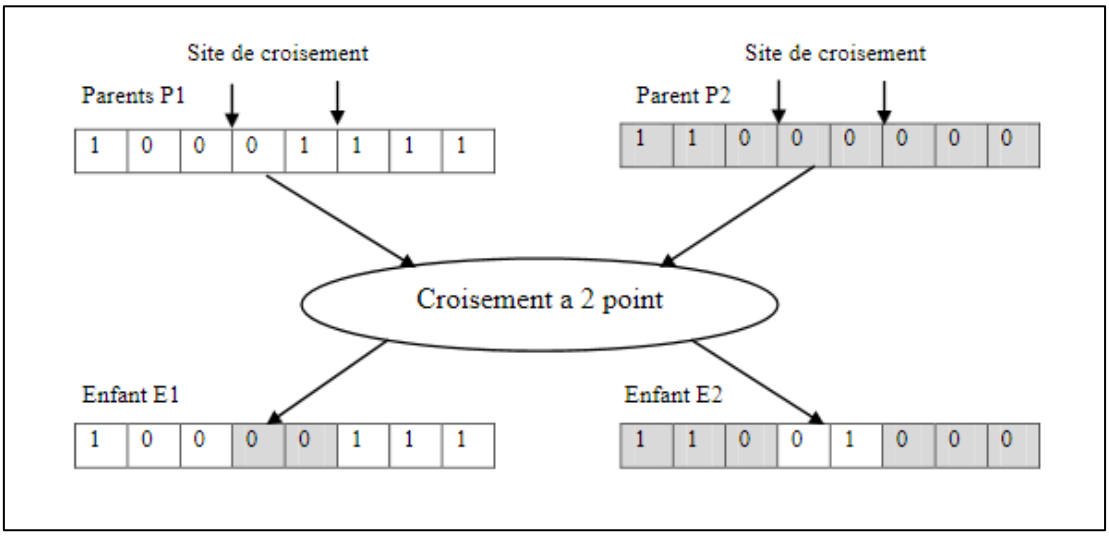


Figure 2.5 Croisements en deux points.

- **Croisement multipoints**

Dans ce cas, plusieurs points de croisement sont sélectionnés et il y a un échange des différentes parties d'allèles cernées par ces points, entre les parents, (Figure 2.6).

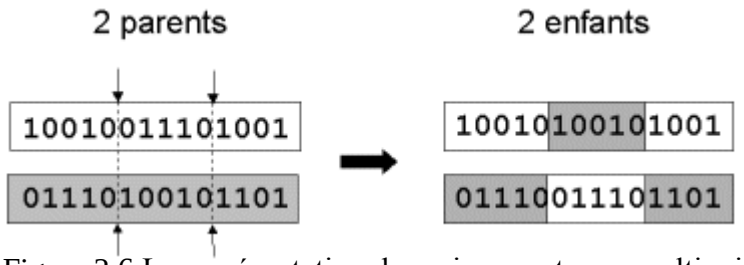


Figure 2.6 La représentation de croissement a un multipoints

- **Croisement uniforme**

Il opère à l'aide d'un masque qui représente les tirages aléatoires, pour décider de la transmission de la valeur de l'allèle à l'un ou l'autre des descendants. Si, à la même position que l'allèle, la valeur du masque est égale à 1, l'allèle du parent 1 passe à celui de l'enfant 1 et l'allèle du parent 2 passe à l'enfant 2. Sinon, c'est l'inverse qui se produit. D'autres types de croisement, plus spécifiques au problème traité, peuvent bien entendu être utilisés dans le cadre d'un algorithme génétique. L'efficacité du croisement dépend souvent de son adaptation au problème (voir la figure 2.7).

Parent 1	1	0	0	0	1	1	1	1	1	0
Parent 2	1	1	1	1	0	0	0	0	0	1
Masque	1	0	0	1	1	1	0	0	1	0
Enfant 1	1	1	1	0	1	1	0	0	1	1
Enfant 2	1	0	0	1	0	0	1	1	0	0

Figure 2.7 La représentation de croissement uniforme

6.2.1.6. Opération de mutation

La mutation est un changement aléatoire selon une certaine règle probabiliste qui doit faire sur les génotypes, avec une faible probabilité Pm (fixée par l'utilisateur) de la valeur d'un ou plusieurs allèles d'un chromosome. En général, la mutation ne permet pas l'obtention de meilleures solutions, mais elle permet de garder une diversité dans l'évolution des individus et d'éviter les optimums locaux, et se protège contre une perte irrécouvrable dans les

caractéristiques des individus. La mutation classique consiste à transformer dans un chromosome binaire un 1 en un 0 ou le contraire. (Figure 2.8).

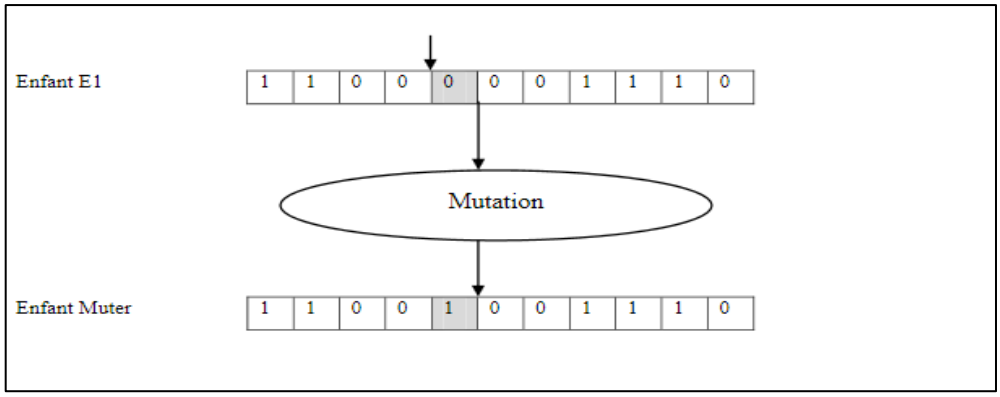


Figure 2.8 Représentation schématique de la mutation.

6.2.2. L'algorithme d'optimisation par essaim de particules (PSO)

L'optimisation par Essaim de particule (PSO) est une technique utilisée pour explorer l'espace de recherche d'un problème quelconque pour trouver l'ensemble des paramètres qui maximise/minimise un objectif particulier. Kennedy et Eberhart [31] ont cherché à simuler la capacité des oiseaux à voler de façon synchrone, et leur aptitude à changer brusquement de direction tout en restant en une formation optimale.

6.2.2.1. Principe de fonctionnement

Dans un système PSO, un essaim d'individus (particules) volent dans l'espace de recherche. Chaque particule représente une solution potentielle au problème d'optimisation. La position d'une particule est influencée par la meilleure position visitée par elle-même (c.-à-d. ses propres expériences) et la position de la meilleure particule dans son voisinage (c.-à-d. l'expérience des particules voisines). Quand le voisinage d'une particule est l'essaim entier, la meilleure position dans le voisinage exprime la meilleure particule globale, et l'algorithme résultant désigné sous le nom d'un PSO gbest. Si le voisinage est petit l'algorithme est généralement connu sous le nom d'un PSO lbest. La performance de chaque particule (c.-à-d. La convergence de la particule vers l'optimum global) est mesurée en utilisant une fonction de forme physique qui change selon le problème d'optimisation.

Chaque particule dans l'essaim est représentée par les caractéristiques suivantes :

x_i: La position actuelle de la particule i.

v_i: La vitesse courante de la particule i.

y_i: La meilleure position personnelle de la particule i.

$\hat{y}i$: La meilleure position de voisinage de la particule i

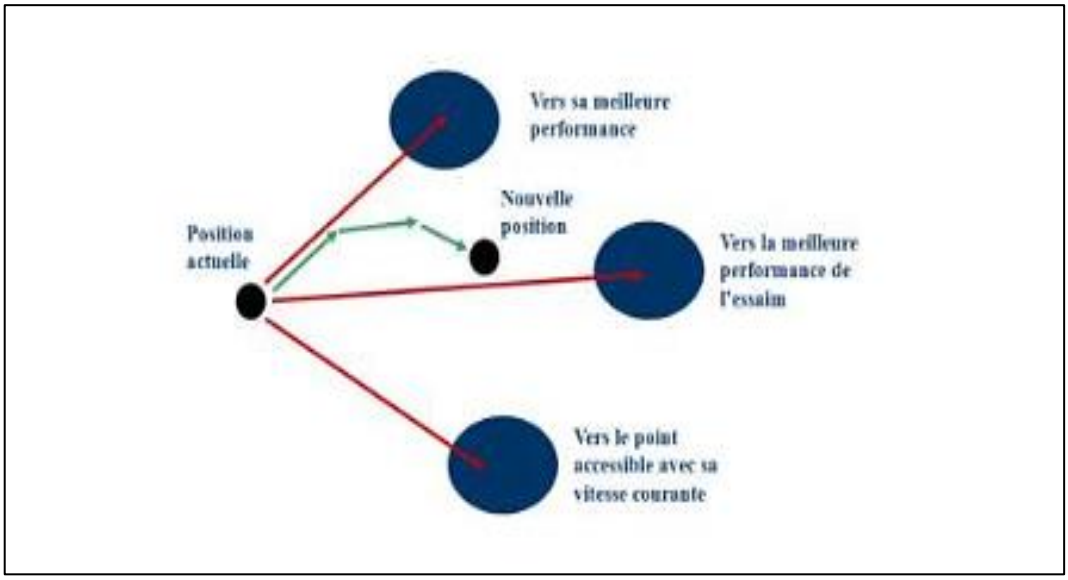


Figure 2.9 Déplacements d'une particule.

La meilleure position personnelle de la particule i est la meilleure position (c-à-d, celle ayant pour résultat la meilleure valeur de forme physique) visitée par la particule i jusqu'ici. La mise à jour de la meilleure position de la particule est comme suit :

$$yi(t+1)\begin{cases} yi(t) si\ f(xi(t+1)) > f(yi(t)) \\ xi(t+1) si\ f(xi(t+1)) < f(yi(t)) \end{cases}$$

Pour le modèle gbest, la meilleure particule est déterminée à partir de l'essaim entier par le choix de la meilleure position personnelle. Si la position de la meilleure particule est notée parle vecteur $\hat{y}i$ alors :

$$\hat{y}i(t) \in \{y0, y1, \dots, ys\} = \min\{f(y0(t)), f(y1(t)), \dots, f(ys(t))\}$$

Où s indique la taille de l'essaim. L'étape de la mise à jour de vitesse est indiquée pour chaque dimension j∈1, ..., Nd Par conséquent, $v_{i,j}$ représente le $j^{éme}$ élément du vecteur de vitesse de la $i^{éme}$ particule.

Ainsi la vitesse de la particule i est mise à jour en utilisent l'équation suivante :

Où :

W est le facteur d'inertie. C_1 et C_2 sont les constantes d'accélérations. r_1 et r_2: suivant une loi uniforme.

La position de la particule i, xi est mise à jour par l'équation suivante :

$$xi(t+1) = xi(t) + vi(t+1)$$

C'est la vectrice vitesse qui dirige le processus de recherche et reflète la "sociabilité" des particules. Si on considère N particules et que chaque particule compare sa nouvelle position à sa meilleure position obtenue, c'est-à-dire qu'on ne prend pas compte du voisinage puisqu'on utilise uniquement l'amélioration obtenue sur la particule elle-même, cela donne l'algorithme.

7. Conclusion

Dans ce chapitre nous avons passé en revue les problèmes d'optimisation et ainsi leurs complexités. Ensuite, nous avons présenté les principales métaheuristiques couramment utilisées pour la résolution des problèmes d'optimisation avec une significative concentration sur les algorithmes génétiques que nous allons utiliser dans notre travail.

CHAPITRE 3

ALGORITHME GENETIQUE APPLIQUE A LA DETECTION DE COMMUNAUTES DANS LES RESEAUX SOCIAUX

1. Introduction

Dans ce chapitre, nous allons présenter le problème de communautés dans les réseaux sociaux. Ensuite on va détailler l'adaptation de l'algorithme génétique pour résoudre ce problème. Une grande partie sera consacrée aux différents composants de l'algorithme ainsi que son principe de fonctionnement.

2. Adapter l'algorithme AG au problème de la détection de communauté

Dans cette partie on va détailler les différents composants de l'algorithme génétique adapté à la résolution du problème de la détection de communauté.

2.1 Codage

Notre algorithme de classification utilise la représentation d'adjacence basée sur le locus proposé dans [25] et utilisé par [16,19] pour la classification multi objective. Dans cette représentation graphique, un individu de la population est composé de N gènes et chaque gène peut assumer des valeurs d'allèle j comprises dans l'intervalle {1, ..., N}. Les gènes et les allèles représentent les nœuds du graphe G = (V, E) modélisant un réseau social, et une valeur j attribuée au gène $i^{éme}$ est interprétée comme un lien entre les nœuds i et j de V. Cela signifie que, dans la solution de regroupement trouvée i et j sera dans le même cluster. Une étape de décodage est toutefois nécessaire pour identifier toutes les composantes du graphe correspondant. Les nœuds participant au même composant sont affectés à un cluster. Comme observé dans [16]. Un avantage majeur de cette représentation est que le nombre k de grappes est automatiquement déterminé par le nombre de composants contenus dans un individu et déterminé par l'étape de décodage. Supposons que le réseau illustré à la figure 3.1. Il est composé de onze nœuds numérotés de 1 à 11. Le réseau peut être partitionné en trois groupes. Parmi les nombreux génotypes possibles, ceux présentés dans le tableau 3.1, correspondant à la solution optimale, sont traduits dans la structure de graphique présentée à la figure 3.2. Chaque composant connecté fournit un groupe de nœuds correspondant au partitionnement du réseau illustré à la figure 3.1.

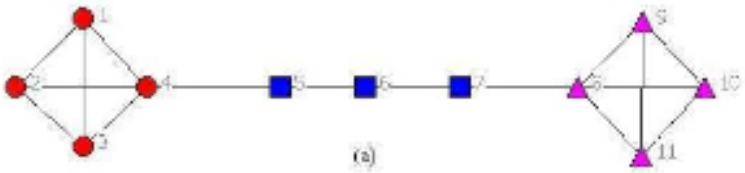


Figure 3.1 Un réseau modélisé sous forme de graphe.

Tableau 3.1 La représentation d'un génotype basée sur le locus.

Position	1	2	3	4	5	6	7	8	9	10	11
Génotype	2	1	2	3	6	5	6	10	8	11	8

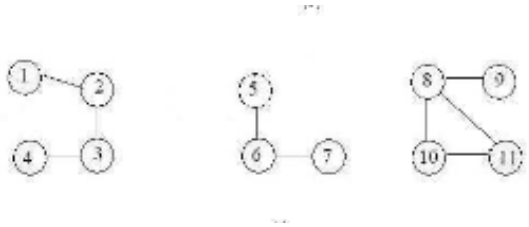


Figure 3.2 La structure graphique du génotype.

2.2 Initialisation

Notre processus d'initialisation prend en compte les connexions effectives des nœuds du réseau social. Une génération aléatoire d'individus pourrait générer des composants qui, dans le graphe d'origine, sont déconnectés. En fait, un individu d généré aléatoirement pourrait contenir une valeur d'allèle j dans la $i^{éme}$position, mais aucune connexion n'existe entre les deux nœuds i et j, c'est-à-dire que l'arête (i, j) n'est pas présente. Dans un tel cas, il est évident que le regroupement dans le même groupe des deux nœuds i et j est un mauvais choix. Pour remédier à cet inconvénient, une fois qu'un individu est généré, il est réparé, c'est-à-dire qu'un contrôle est exécuté pour vérifier Un lien efficace existe entre un gène en position i et la valeur d'allèle j. Cette valeur n'est conservée que si le bord (i, j) existe. Autrement, j est remplacé par l'un des voisins de i. Cette initialisation guidée oriente l'algorithme vers une décomposition

du réseau en groupes de nœuds connectés. Nous appelons un individu générant ce type de partitionnement sécurisé, car il évite les divisions sans intérêt contenant des nœuds non connectés. Les individus sécurisés améliorent la convergence de la méthode car l'espace des solutions possibles est restreint [6].

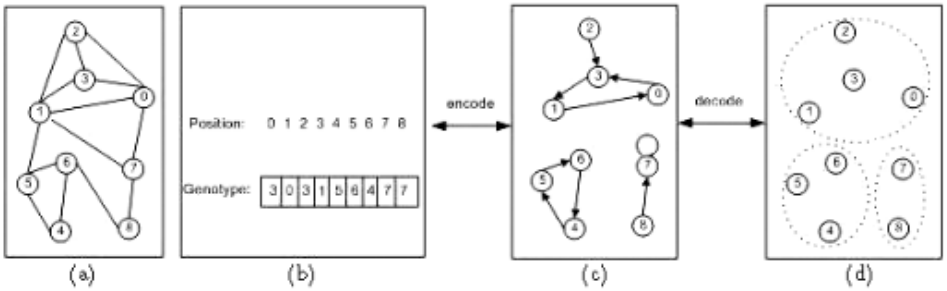


Figure 3.3 Illustration de la représentation de la contiguïté basée sur le locus. (a) montre la topologie d'un réseau complexe. (b) montre un génotype possible. (c) montre comment le génotype de (b) est traduit en une structure de graphe, par exemple le nœud 0 est lié au nœud 3, car la valeur du gène g0 est 3. (d) montre le résultat de la partition.

2.3 Les stratégies de sélection

La sélection permet d'identifier statistiquement les meilleurs individus d'une population, c'est-à-dire de favoriser leur présence dans la génération suivante, et d'éliminer les mauvais. A chaque itération, on sélectionne un ensemble d'individus pour la préparation à la reproduction, afin de reproduire une nouvelle génération. Dans notre problème sujet d'étude on a choisi d'utiliser la sélection par roulette.

2.4 Fonction objective

Chaque chromosome apporte une solution potentielle au problème. Néanmoins, ces solutions n'ont pas toutes le même degré de pertinence. C'est à la fonction de performance de mesurer cette efficacité, pour permettre à l'AG de faire évoluer la population dans un sens bénéfique pour la recherche de la meilleure solution.

Le choix de la fonction fitness est une autre étape cruciale pour obtenir de bonnes solutions. Dans le contexte de la détection de communauté, la fonction la plus répandue est la modularité, introduite à l'origine par Newman et Girvan pour évaluer les résultats du regroupement, puis utilisée comme critère d'optimisation. De manière plus formelle, la modularité est définie comme suit [14] :

$$Q = \sum_{s=1}^{Nm} \left(\frac{ls}{L} - \left(\frac{ds}{2L} \right)^2 \right)$$

Q : Newman-Girvan modularité.

Nm : nombre de modules.

ls : nombre de liens entre les nœuds dans le module s.

L : nombre de liens dans le réseau.

ds : somme des degrés des nœuds dans le module s.

2.5 Operateur de croissement

On applique cet opérateur avec une probabilité Pc qui appartient à [0,1]. Dans notre problème nous avons utilisé un croisement uniforme car il garantit le maintien des connexions effectives des nœuds du réseau social chez l'enfant. En fait, en raison de l'initialisation biaisée, chaque individu de la population est en sécurité, c'est-à-dire qu'il a la propriété que si un gène contient une valeur j, alors le bord (i, j) existe. Ainsi, à deux parents sûrs, un 1 vecteur binaire aléatoire est créé. Un croisement uniforme sélectionne ensuite les gènes où le vecteur est un 1 du premier parent et les gènes où le vecteur est un 0 du second parent et combine les gènes pour former l'enfant. L'enfant à chaque position i contient une valeur j venant de l'un des deux parents. Ainsi, le bord (i, j) existe. Cela implique que deux parents en sécurité génèrent un enfant en toute sécurité [5].

Tableau 3.2 Croisement uniforme pour la représentation basée sur le locus.

Position	1	2	3	4	5	6	7	8	9	10	11	12
Parent1	3	1	4	7	4	12	4	7	8	9	12	6
Parent2	2	9	2	5	6	7	8	10	2	11	10	8
Mask	1	1	0	0	1	1	0	0	0	1	1	1
Fils	2	9	4	7	6	7	4	7	8	11	10	8

2.6 Operateur de mutation

La mutation des individus correspond à des changements aléatoires de passage par un nœud d'une solution. On applique cet opérateur avec une probabilité Pm qui appartient à [0,1]. L'opérateur de mutation qui modifie de manière aléatoire la valeur j d'un $i^{éme}$gène provoque une exploration inutile de l'espace de recherche. Ainsi, les valeurs possibles qu'un allèle peut prendre sont limitées aux voisins du gène i. Cette mutation réparée garantit la génération d'un enfant muté sûr dans lequel chaque nœud n'est lié qu'à l'un de ses voisins. Chaque individu génère une structure de graphe dans laquelle chaque composant est un sous-graphe connecté de G. Pour un nombre fixe de générations, L'algorithme calcule la fonction de fitness de chaque membre de la solution et applique les opérateurs de variation spécialisés pour produire la nouvelle population [5].

3. Décodage du génotype

Algorithme 3.1 Décodage du génotype [15]

```
1:  Procédure Décodage
2:    current_cluster = 1
3:    Pour chaque  i dans 1 à N alors
4:        cluster_assign = −1
5:   Fin pour
6:   Pour chaque  i dans 1 à N alors
7:      ctr = 1
8:      Si cluster_assign i = −1 alors
9:         cluster_assign i = current_cluster
10:        voisin = gi
11:       précédent_ctr = i
12:        ctr = ctr + 1
13:              tant que cluster_assign_ voisin == −1 alors
14:                    précédent _ctr = voisin
15:                    cluster assign_ voisin = curent_cluster
16:                    voisin = g_ voisin
17:                     ctr = ctr + 1
18:               Fin tant que
19:               si cluster_assign_ voisin = curent_cluster alors
```

```
20:                    ctr = ctr – 1
21:                        tant que ctr >= 1 alors
22:                    cluster_assign_ précédent _ctr = cluster_assign_ voisin
23:                    ctr = ctr – 1
24:                Fin tant que
25:            Sinon
26:              curent_cluster=curent_cluster + 1
27:            Fin Si
28:        Fin Si
29:    Fin pour
30:    numéro de clusters=curent cluster
31: Fin procédure
```

4. Remplacement

C'est la phase pour construire la nouvelle génération. Pour assurer cette étape, les chromosomes sont ordonnés par ordre décroissant de leurs fitness et les meilleurs individus seront gardés ce qui reflète les contraintes du problème.

5. Critère d'arrêt

Dans notre travail, nous avons fixé un nombre d'itérations pour le déroulement de l'exécution de l'algorithme.

6. Réalisation et Expérimentations

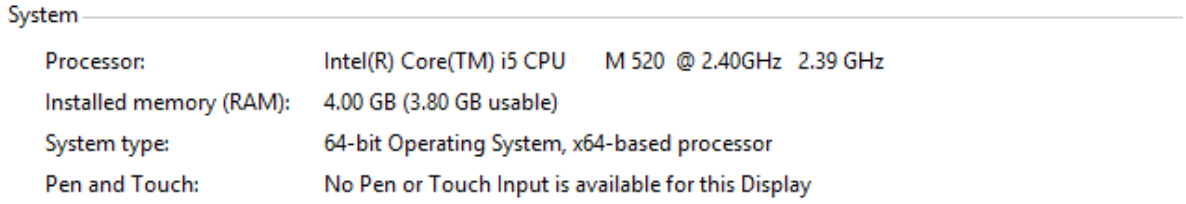
L'implémentation d'un logiciel vient après un enchainement de plusieurs étapes dans le processus de développement, son but principal est de réaliser un produit capable de résoudre les problèmes posés en utilisant des outils et des algorithmes.

Dans cette section, nous donnée quelques résultats expérimentaux obtenus par cet algorithme.

7. Environnement de développement

Pour la réalisation de ce travail, nous avons eu recours aux environnements suivants :

7.1 Environnement matériel



7.2 Environnement logiciel

- Windows 8.1 comme système d'exploitation.
- Langage java comme langage de programmation.

7.2.1 Le langage de programmation java

Le langage Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy, cofondateur de Sun Microsystems. Java a été officiellement présentée le 23 mai 1995 au SunWorld. La société Oracle racheta alors la société Sun en 2009, ce qui explique pourquoi ce langage appartient désormais à Oracle. La particularité et l'intérêt de Java réside dans sa portabilité entre les différents systèmes d'exploitation tels que Unix, Windows, ou MacOS. Un programme développé en langage Java, peut ainsi s'exécuter sur toutes les plateformes, grâce à ses Framework associés visant à garantir cette portabilité [29].

- C'est l'un des langages de programmation les plus populaires au monde.
- Il est facile à apprendre et simple à utiliser.
- C'est open-source et gratuit.
- C'est sécurisé, rapide et puissant.
- Il a un énorme soutien de la communauté (des dizaines de millions de développeurs).

7.2.2 L'environnement NetBeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL (Common Développent and Distribution License) et GPLv2. En plus de Java. NetBeans permet la prise en charge native de divers langages tels le C, le C, le JavaScript, le XML, le Groovy, le PHP et le HTML, ou d'autres (dont Python et Ruby) par l'ajout de greffons. Il offre toutes les facilités d'un IDE moderne (éditeur avec coloration syntaxique, projets multi-langage, re factoring, éditeur graphique d'interfaces et de pages Web). NetBeans est disponible sous Windows, Linux, Solaris, Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)).

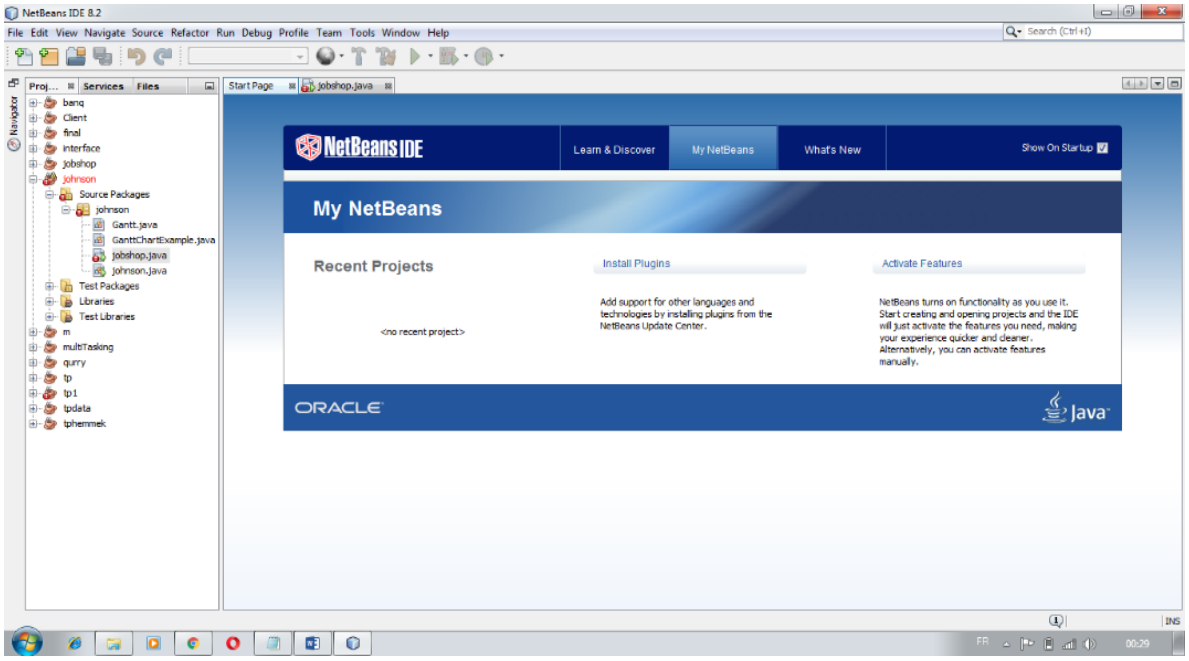


Figure 3.4 Page de démarrage de NetBeans

8. Résultats expérimentaux

Cette section présente une évaluation empirique de notre méthode proposée sur des réseaux réels. La performance de notre méthode est comparée à avec ceux rapportés par Girvan et Newman sur des réseaux du monde réel pour lesquels le partitionnement en communautés est connu.

L'algorithme a été écrit en Java. Nous avons utilisé des paramètres standard pour l'algorithme génétique, taux de croisement 0.8, taux de mutation 0.2, fonction de sélection de la roulette. La taille de la population était de 30, le nombre de générations 100.

On a besoin d'une métrique d'évaluation pour évaluer les résultats trouvés. La métrique la plus connu pour comparais deux partitions est l'information mutuelle normalisée (normalized mutual information NMI). Les valeurs du NMI sont entre 0 et 1. Si deux partitions sont identiques alors NMI= 1.

Soit deux partitions A et B d'un réseau dans des communautés, soit C la matrice de confusion dont l'élément C_{ij} est le nombre de nœuds de la communauté i de la partition A qui font également partie de la communauté j de la partition B. L'information mutuelle normalisée I (A, B) est défini comme :

$$I(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij}\log(C_{ij}N/C_iC_j)}{\sum_{i=1}^{C_A} C_i\log(C_i/N)+\sum_{j=1}^{C_B} C_j\log(C_j/N)}$$

Où C_A (C_B) est le nombre de groupes dans la partition A (B), C_i. (C_j) est la somme des éléments de C dans la rangée i (colonne j) et N est le nombre de nœuds.

Si A = B, I (A, B) = 1. Si A et B sont complètement différents, I (A, B) = 0.

Les réseaux réels que nous avons utilisés sont Club de karaté de Zachary [26], les Dauphins [10], Football américain [17] :

— **Club de karaté de Zachary :** C'est un réseau très populaire et très utilisé par plusieurs algorithmes afin de tester leurs performances puisque sa structure de communauté est connue à l'avance. Club de karaté de Zachary est construit à partir des relations entre 34 membres d'un club de karaté dans une université aux États-Unis avec 78 liens.

— **Les dauphins :** ce réseau contient 62 nœuds et 59 liens. Ce réseau modélise les associations fréquentes entre les dauphins dans une collectivité vivant à Doubtful Sound, New zealand.

— **Football américain :** représente le calendrier des matchs entre des équipes américaines de football durant l'année 2000 (American College football). Ce réseau constitué de 115

nœuds (équipes) et 613 liens. Les liens représentent les matchs de saison régulière entre les deux équipes qui ont joué ensemble.

Les résultats obtenus avec une taille de la population égal 30 et un nombre de générations fixé à 100 sont résumés dans le tableau 3 ci-dessous :

Tableau 3.3 Résultats de performance sur des réseaux réel.

Méthode	Karaté			Dauphins			Football		
	C	**NMI**	**Q**	**C**	**NMI**	**Q**	**C**	**NMI**	**Q**
GN	2	0.84	0.419	4	0.84	0.528	12	0.93	0.604
Notre méthode	4	0.595	0.406	6	0.49	0.471	11	0.696	0.424

9. Analyse des résultats

Le tableau 3.3 illustre les résultats sur les réseaux réels.

Pour chaque réseau, nous avons calculé les informations mutuelles normalisées moyennes sur ces 10 exécutions.

Les résultats sont reportés dans le tableau 3.3. Le tableau montre clairement la performance de notre méthode en ce qui concerne la méthode de Girvan et Newman.

Sur le réseau de Karaté, obtenu une information mutuelle normalisée moyenne de 0.595 sur 10, la pire valeur étant de 0.53 et la meilleure valeur de 0.687 avec Q moyenne 0.406.

Comme pour le Dauphin a obtenu une information mutuelle normalisée moyenne de 0.49 sur 10, la pire valeur étant de 0.422 et la meilleure valeur de 0.552 avec Q moyenne 0.471.

Comme pour le Football a obtenu une information mutuelle normalisée moyenne de 0.696 sur 10, la pire valeur étant de 0.649 et la meilleure valeur de 0.783 avec Q moyenne 0.471.

Notre méthode permet de classer correctement le réseau Football, notre méthode a regroupé certains des nœuds dans des communautés correctes. Les résultats obtenus montrent la

capacité des algorithmes génétiques à traiter efficacement l'identification des communautés dans les réseaux.

Tableau 3.4 Comparaison NMI entre GN et notre méthode

NMI	Karaté	Dauphins	Football
Notre méthode	0.461	0.307	0.458
GN	0.83	0.64	0.882

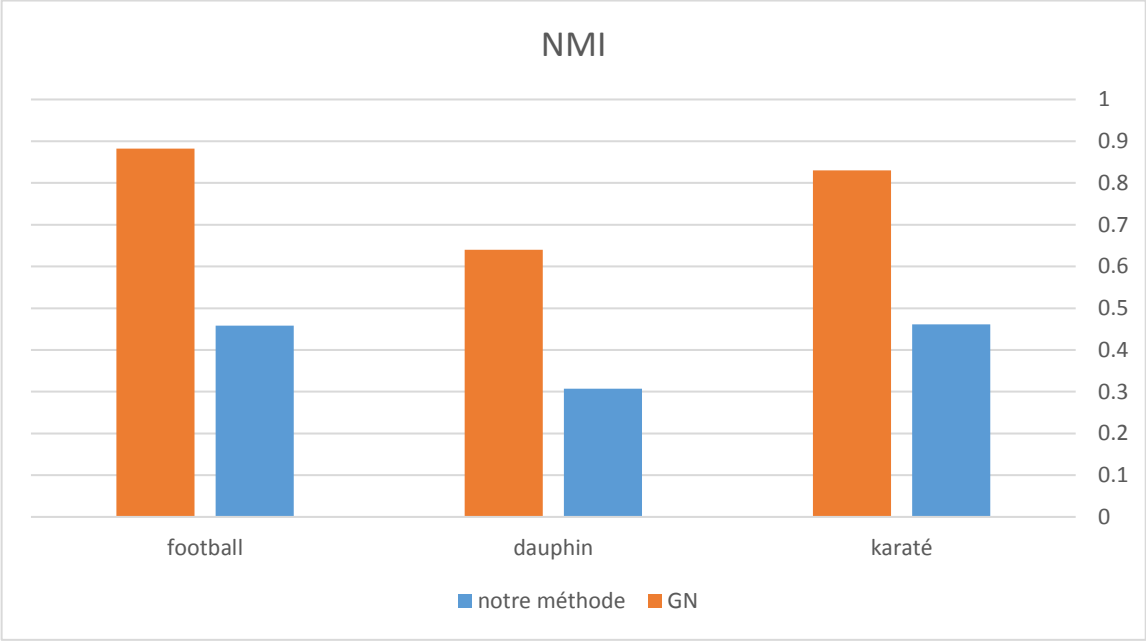


Figure 3.5 Comparaison NMI entre GN et notre méthode

Tableau 3.5 Comparaison Q Modularité entre GN et notre méthode

Q	Karaté	Dauphins	Football
Notre méthode	0.406	0.471	0.424
GN	0.419	0.528	0.604

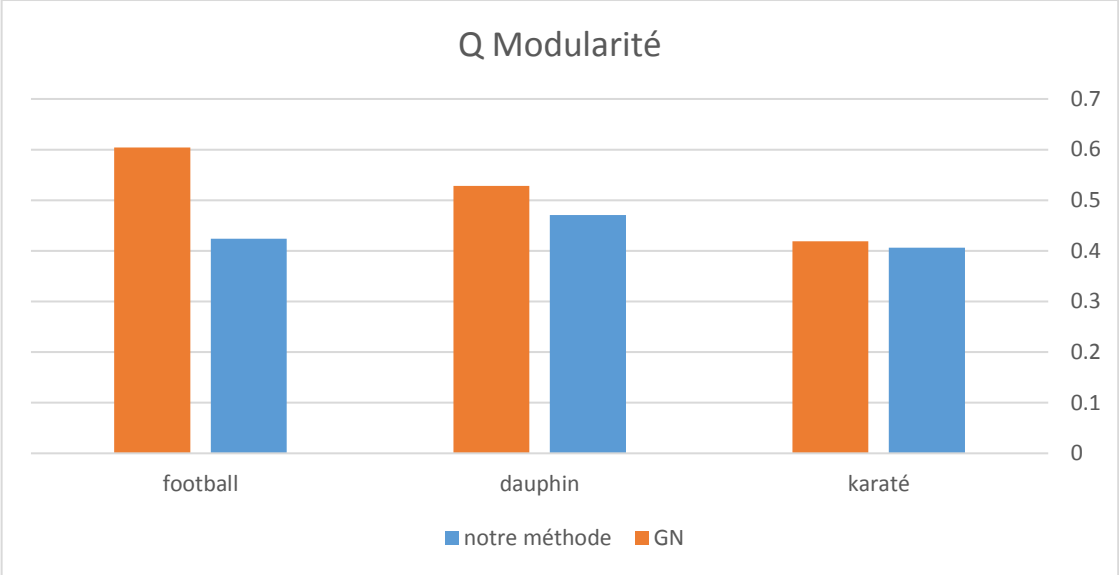


Figure 3.6 Comparaison Q Modularité entre GN et notre méthode

Tableau 3.6 Comparaison C nombre de communautés entre GN et notre méthode

C	Karaté	Dauphins	Football
Notre méthode	4	6	11
GN	2	4	12

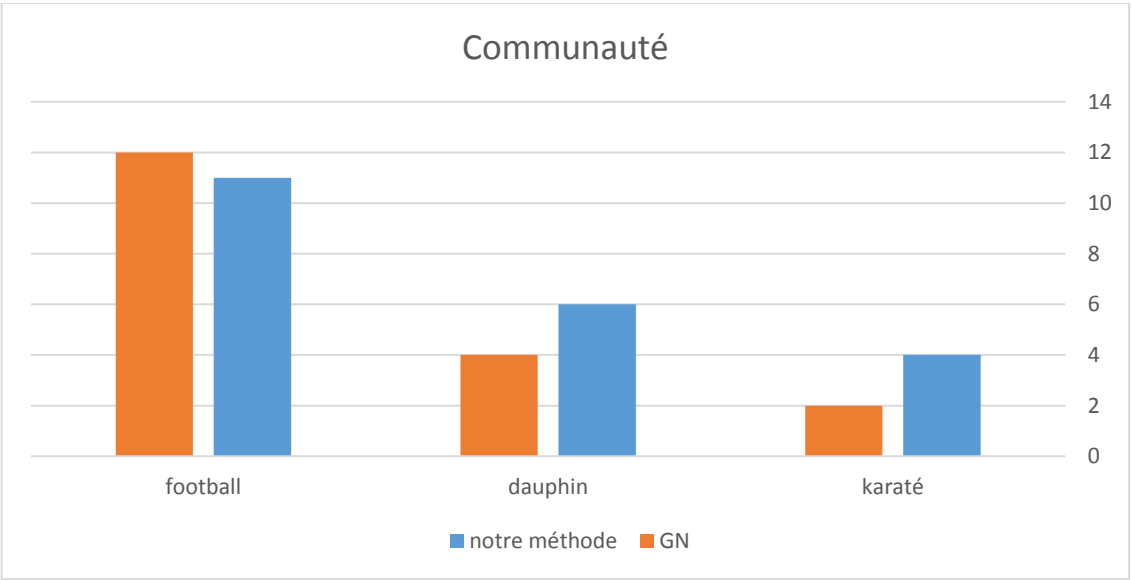


Figure 3.7 Comparaison C nombre de communautés entre GN et notre méthode

10. Conclusion

Dans ce chapitre, nous avons présenté le problème de détection de communautés dans les réseaux sociaux, ensuite nous avons décrit les composants essentiels de l'algorithme génétique choisi pour la résolution du problème posé. En deuxième temps, nous avons présenté les résultats numériques obtenus par notre algorithme sur un ensemble de réseaux sociaux avec une comparaison avec la méthode de Newman.

CONCLUSION GENERALE

CONCLUSION GENERALE

Le besoin d'utiliser l'optimisation combinatoire est très grand. Le problème de détection de communautés l'un des problèmes les plus complexes qui a ses applications réelles partout.

Dans ce travail, nous avons étudié le problème de la détection de communautés. Nous avons commencé par la définition générale sur la théorie des graphes. En deuxième temps, nous avons passé en revue les métaheuristiques d'optimisation et la définition de la notion de complexité des algorithmes et des problèmes. Ensuite nous avons défini le problème de détection de communautés.

Nous avons appliqué l'algorithme génétique pour résoudre notre problème. Enfin, nous avons résultats expérimentaux obtenus par cet algorithme.

Le concept de détection communautaire, bien que simple, s'est révélé très efficace. En fait, des expériences sur des réseaux réels ont montré la capacité de l'approche génétique à détecter correctement les communautés avec des résultats comparables avec la méthode de Newman. Comme perspectives, les travaux futures viseront à appliquer d'autres métaheuristiques d'optimisation dans le contexte mono ou multi-objective pour améliorer la qualité des résultats obtenus ou de proposer des méthodes hybrides basées sur les métaheuristiques pour résoudre le problème de la détection de communautés dans les réseaux sociaux en utilisant d'autres benchmarks de la littérature.

BIBLIOGRAPHIE

[1] ABDELOUAHAB Khalid Algorithme génétique appliqué au problème de k plus courts chemins dans un graphe : Etude& évaluation

[2] Abbas EL DOR. Perfectionnement des algorithmes d'Optimisation par Essaim Particulaire. Applications en segmentation d'images et en électronique.THÈSE DE DOCTORATEN INFORMATIQUE.UNIVERSITÉ PARIS-EST.2012

[3] A. BAPTISTE, Les métaheuristiques en optimisation combinatoire, conservatoire national des arts et métiers paris. 2006.

[4] Alaoui Abdiya. Application des techniques des métaheuristiques pour l'optimisation de la tâche de la classification de la fouille de données. Magister en informatique.UNIVERSITE D'ORAN Mohamed Boudiaf. 2012.

[5] Clara Pizzuti GA-Net: A Genetic Algorithm for Community Detection in Social Networks

[6] Clara Pizzuti Evolutionary Computation for Community Detection in Networks: A Review

[7] C. H. Papadimitriou: The complexity of combinatorial optimization problems. PhD, 1976.

[8] C.H. PAPADIMITRIOU, K. STEIGLITZ, Combinatorial optimization –algorithms and complexity, Prentice Hall, 1982.

[9] Carfier J. et Chertienne P.1988. Problèmes d'ordonnancement. Modélisation/ complexité/ algorithme.1988.

[10] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten and S.M. Dawson: The bottle nose dolphin community of Doubtful Sound features a large proportion of long-lasting associations, Behavioral Ecology and Sociobiology 54, 396-405, 2003.

[11] F. Glover. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, Vol. 13, pp. 533–549, 1986

[12] I. H. Osman ET G. La porte : Metaheuristics: A bibliography. Annals of Operations Research.63:513.623, 1996.

[13] Jean Creuse fond. Caractériser et détecter les communautés dans les réseaux sociaux. Réseaux sociaux et d'information [cs.SI]. Normandie Université, 2017. Français. <NNT : 2017NORMC203>. <Tel-01497593>

[14] Jean-Loup Guillaume COURS SYRRES DÉTECTION DE COMMUNAUTÉS page 16

[15] Jin-Kao Hao, Philippe Galinier, Michel Habib, Métaheuristique pour l'optimisation combinatoire et l'affectation sous contraintes, Revue d'Intelligence Artificielle Vol : No. 1999.

[16] Handle, J., Knowles, J.: An evolutionary approach to multi objective clustering. IEEE trans-actions on Evolutionary Computation 11(1), 56–76 (2007)

[17] M. Girvan and M.E.J. Newman: Community structure in social and biological net-works, Proceedings of the National Academy of Sciences, vol. 99, no. 12, page 7821-7826, 2002.

[18] Mehdi Rouan Serik. Implémentation de méthodes de recherches locales sur les architectures multi et many cœurs. Diplôme de magister. Université d'Oran.2012.

[19] Makate, N., Miki, M., Hiroyasu, T., Senda, T. : Multi objective clustering with automatic k-determination fot large-scale dara. In: Proc. of the Int. Genetic and Evolutionary Computation Conference (GECCO 2007), pp. 861–868 (2007)

[20] Mokeddemm Diab, Contrôle Flou des Processus Biotechnologiques à Base d'Algorithmes Génétiques, Présentée pour l'Obtention du Diplôme de Doctorat En Sciences en Electronique. Université FERHAT ABBASDE SETIF,2010.

[21] R. Battiti & G. Tecchiolli. The reactive tabu search. ORSA Journal on Computing, 6(2): 126–140, 1994.

[22] S. kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. Science, New Series, 220(4598):671-680, 1983.

[23] SAOUD Bilal Apprentissage incrémental dynamique dans les réseaux sociaux : application à la détection de communautés

[24] T. Vallée, M. Yildiz oglu, Présentation des algorithmes génétiques et de leurs applications en économie.

[25] Park, Y.J., Song, M.S.: A genetic algorithm for clustering problems. In: Proc. of 3rd Annual Conference on Genetic Algorithms, pp. 2–9 (1989)

[26] W. Zachary: An information flow model for conflict and fission in small groups, Journal of Anthropological Research, vol. 33, no.4, pp. 452-473, 1977.

[27] ZHENYU YAN, CHUAN SHI, YI WANG, YANAN CAI and BIN WUA GENETIC ALGORITHM FOR DETECTING COMMUNITIES IN LARGE SCALE COMPLEX NETWORKS page 9

Site web:

[28] https://fr.wikipedia.org/wiki/Java_(langage)

[29] https://fr.wikipedia.org/wiki/Modularit%C3%A9_(r%C3%A9seaux)

[30] http://sis.univ-tln.fr/~tollari/TER/AlgoGen1/node5.html,consultele 20/05/2017,14:53.

ملخص

مشكلة اكتشاف المجتمعات في الشبكات الاجتماعية تمت دراستها على نطاق واسع في السنوات الأخيرة. هذه المشكلة لها العديد من التطبيقات المهمة والمتنوعة في مجالات مختلفة. في هذه المذكرة ، اقترحنا طريقة وراثية لاكتشاف المجتمعات في الشبكات الاجتماعية. تستخدم الخوارزمية مقياسًا يُطلق عليه الوحدة النمطية Q. تم اختبار كفاءة هذه الخوارزمية في بعض الأمثلة من خلال عمل مقارنة مع خوارزمية نيومان.

الكلمات المفتاحية: اكتشاف المجتمعات ، الشبكات الاجتماعية ، الوحدة النمطية Q ، الخوارزمية الجينية.

Résumé

Le problème de la détection de la structure de la communauté dans les réseaux sociaux a été étudié de manière approfondie ces dernières années. Ce problème a de nombreuses applications importantes et diverses dans différents domaines. Dans ce mémoire, nous avons proposé une approche génétique pour découvrir les communautés dans les réseaux sociaux. L'algorithme utilise une métrique, appelée modularité Q, comme fonction de fitness. L'efficacité de cet algorithme a été testée sur quelques exemples avec une étude comparative avec la méthode de Newman.

Mots clés : détection de communauté, réseaux sociaux, modularité, algorithme génétique.

Abstract

The problem of detecting the structure of the community in social networks has been extensively studied in recent years. This problem has many important and diverse applications in different areas. In this dissertation, we proposed a genetic approach to discover communities in social networks. The algorithm uses a metric, called modularity Q, as a fitness function. The effectiveness of this algorithm has been tested on some examples with a comparative study with Newman's method.

Key words: community detection, social networks, modularity, genetic algorithm.