Universities kasdi merbah ouargla
Faculté des nouvelles technologies de l'information et de la communication
Département de l'informatique et de technologie de l'information

Mémoire
MASTER ACADEMIQUE
Domaine : Mathématiques et Informatique Filière : Informatique
Option: Informatique industrielle Présenté par
LEMTENNCHE BOUBAKEUR

Thème

UN ALGORITHME DES FEUX D'ARTIFICE DISCRET PARALLELE POUR L'EXPANSION DES REQUETES

Devant le jury

M	MAHDJOUB M. Bachir	MAA	UKM Ouargla	Président
M	KHALDI Amine	MAA	UKM Ouargla	Examinateur
M	BEKKARI Fouad	MAA	UKM Ouargla	Encadreur

Année universitaire : 2020 /2021

Remerciements

Nous tenons à remercier le bondieu, le tout puissant de nous donner la patience, la santé et le courage pourfinir ce travail.

Nous remercions profondément notre

encadreur : Monsieur BEKKARI FOUAD qui nous a encouragé à faire le maximum d'efforts dans ce travail, sans ses encouragements cette mémoire n'aurait sans doute pas abouti.

Et à toutes les personnes qui mous'ont aidé de près et de loin.

Dédicace

C'est avec joie que je dédie ce modeste

travail : A ma grans famille : mon père, ma

mère

A ma petite famille : Ma femme

A mes enfants : aya, bensalem,

abdellah Mon grand frère :

abderrahmane

Pour leurs patiences et encouragements. Que dieu les protègent.

À mes amis et mes collègues pour les bons momentsque j'ai passé avec eux

LEMTENNECHE BOUBAKEUR

ملـــخـص :

من أكثر المجالات التي تشهد تطورا علميا سريعا مجال تحسين النتائج, و الذي يهتم بإيجاد أحسن الحلول و النتائج لمجموعة كبيرة جدا من الحلول المتاحة . وتعد خوارزميات الألعاب النارية من احدث هذه الطرق المستخدمة إذ أنها ظهرت أول مرة سنة 2010 , وهي خوارزمية تعتمد في طريقة بحثها عن أحسن نتيجة ممكنة على مبدأ انفجار و انتشار الألعاب النارية, في بحثنا هذا سنحاول إنشاء خوارزمية الألعاب النارية المتوازية وتطبيقها على الاستعلام المتوازي لإيجاد أحسن تركيبة من الكلمات المكونة للاستعلام.

كلمات مفتاحية

استرجاع المعلومات, تمديد الاستعلام, الخوارزميات المتوازية

Résumé :

L’optimisation combinatoire est une approche qui connaît un développement scientifique rapide, ce domaine consiste à trouver les meilleures solutions et les meilleurs résultats pour nombre très grands des solutions disponibles. Parmi les méthodes de résolution des ces problèmes, Les algorithmes de feux d'artifice est un nouvelle algorithme créer en 2010, qui est chercher d’un meilleur résultat possible du principe de l'explosion et de la propagation des feux d'artifice. Dans notre recherche, nous allons essayer de créer un algorithme de feux d'artifice parallèle discret et l'appliquer à l’expansion des requêtes parallèles pour trouver la meilleure combinaison de mots de requête.

Mots kles :
recuperation dinformation, extansion de requete, algorithms paralleles

Table des matières

Chapitre IV : l'implémentation de l'algorithme de feux d'artifice discret parallèle

Liste des figures

Introduction générale :

L'optimisation combinatoire occupe une place très importante en informatique. Cette importance justifie d'une part les difficultés des problèmes d'optimisation, d'autre part de nombreuses applications pratiques peuvent être formulées sous la forme d'un problème d'optimisation combinatoire. Les problèmes d'optimisation sont utilisés pour modéliser de nombreux problèmes dans les différents secteurs de l'industrie (télécommunications, électronique, mécanique, chimie, transport, …).

Parmi les méthodes de résolutions des problèmes d'optimisation combinatoire nous allons choisir dans notre mémoire une nouvelle approche appelé : algorithme de feux d'artifice, cet algorithme a été créée en 2010, nous asseyons d'ajouter la notion de discret et le parallélisme. Pour accepter notre bute nous devisons cette éducation sur quatre chapitres :

- Dans le premier chapitre, on commence par la définition des problèmes de l'optimisation combinatoire. Ensuite, les déférents méthodes de résolution (les méthodes exactes et approchées) en insistante sur les méthodes approchées.
- En deuxième chapitre : l'algorithme original de feux d'artifice est le premier type d'algorithme créé à travers l'observation du fait de l'explosion du feu d'artifice qui a été proposé par Ying Tan et Yuanchun Zhu en 2010. puis, ils l'assimilent dans un algorithme pour chercher une solution optimale. On vous résume les principaux processus dans l'algorithme des feux d'artifice :

(Le processus d'exploration : Le processus d'exploitation : Le processus de mutation) et ces paramètres (L'opérateur d'explosion, le nombre des étincelles, Génération des étincelles : L'amplitude de l'explosion) enfin, on va vous donner un vue sur les feux d'artifice discret.

- Le troisième chapitre consiste a défini les parallélisassions et leur raison d'étatisation pour implémenté notre algorithme parallèle dans le dernier chapitre.
- Le quatrième chapitre (chapitre d'implémentations), nous commençons par l'expansion des requêtes et appliquer l'algorithme que nous avons créé pour chercher la meilleur combinaison des mots puis formuler ce problème comme un problème d'optimisation combinatoire. nous avons conclu que le nombre des solutions est très grand, pour cela on choisit le parallélisme et les MPI pour la transaction des meilleurs solutions entre les processus.

En fin nous allons créer l'algorithme de notre approche et représenté la (programmer) par le langage python.

Chapitre I : L’optimisation Combinatoire

1. Introduction :

L’optimisation combinatoire est une branche de l'optimisation appliquée en mathématique et en informatique, également liée à la recherche opérationnelle, l'algorithmique et la théorie de la complexité.

L'optimisation combinatoire consiste à trouver la meilleure solution dans un ensemble des solutions réalisables. En général, cet ensemble fini, mais compte un très grand nombre.

Dans ce chapitre nous présentons en détaille l’optimisation combinatoire et la définition de ces problèmes et les déférents méthodes de résolutions.

2. Problème du L'optimisation combinatoire :

Définition :

L’objectif de d’optimisation combinatoires est de minimiser ou maximiser une fonction, souvent appelée la fonction de coût, d’une ou plusieurs des variables soumises à des contraintes. L’optimisation combinatoire joue un rôle important dans la recherche opérationnelle, les mathématiques discrètes et l' informatique . Son importance est justifiée, en partie, par la difficulté des problèmes d' optimisation et, en partie, par le grand nombre d' applications pratiques qui peuvent être exprimées comme un problème d' optimisation combinatoire [2].En dépit du fait que des problèmes optimisation combinatoires sont souvent simples à définir, ils sont souvent difficiles à résoudre. En effet, la plupart de ces problèmes appartiennent à la classe des problèmes NP-difficiles et ne possèdent donc pas à ce jour de solution algorithmique efficace valable pour toutes les données [1].

Résolution d’un problème d’optimisation combinatoire :

Pour La résolution d’un problème d’optimisation combinatoire nécessite :

- la définition de l’ensemble des solutions réalisables,
- l’expression de l’objectif à optimiser,
- le choix de la méthode d’optimisation à utiliser.

Les deux premiers points relèvent de la modélisation du problème, le troisième de sa résolution. Afin de définir l’ensemble des solutions réalisables, il est nécessaire d’exprimer l’ensemble des contraintes du problème. Ceci ne peut être fait qu’avec une bonne connaissance du problème sous étude et de son domaine d’application

Les méthodes d’optimisation combinatoire :

Les méthodes d'optimisation peuvent être réparties en deux grandes classes de méthodes pour la résolution des problèmes :

- Les méthodes exactes.
- Les méthodes approchées.

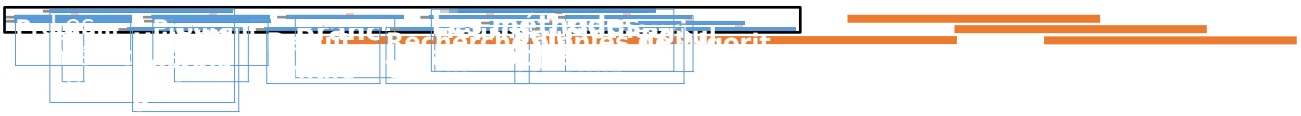

Figure 1:Classification des méthodes d'optimisation combinatoire

Les méthodes exactes :

Les algorithmes exacts sont utilisés pour trouver au moins une solution optimale d'un problème [3]. Les algorithmes exacts les plus réussis dans la littérature appartiennent aux paradigmes de trois grandes classes :

- ❖ La programmation dynamique,
- ❖ La programmation linéaire,
- ❖ Les méthodes de recherche arborescente (Branch & bound),

La méthode de branch and bound :

La méthode branch and bound (procédure par l'évaluation et progressive séparation) se compose de systématiquement énumérer ces solutions dans telle une manière qui, en utilisant certaines propriétés de la problème, cette méthode est en mesure d' éliminer partiellement des solutions qui ne pas conduisent à la solution souhaitée . En un résultat, il est commun de trouver la désirée solution dans une raisonnable quantité de temps. Dans le pire des cas , le scénario, on doit toujours compter sur l' explicite l' élimination de toutes les possibles solutions à la problème.

Pour faire cela, cette méthode comprend une fonction qui permet vous de placer une prise sur certaines solutions, soit pour éliminer les ou à garder les comme potentiels des solutions. Bien entendu,

l' efficacité d' une méthode branch and bound est déterminée, entre autres , par la qualité de la fonction (c'est-à-dire sa capacité à éliminer rapidement des solutions partielles).

- **Algorithme général :**

Début

Placer le nœud début de longueur *0* dans une liste.

Répéter

Si la première branche contient le nœud recherché **alors**
Fin avec succès.

Sinon

- Supprimer la branche de la liste et former des branches nouvelles en étendant la branche supprimée d'une étape.
- Calculer les coûts cumulés des branches et les ajouter dans la liste de telle sorte que la liste soit triée en ordre croissant.

Jusqu'à (liste vide ou nœud recherché trouvé)

Programmation Dynamique :

Richard Bellman a inventé le terme «programmation dynamique » en 1940 pour décrire une technique qui permet un d'aborder un problème dans une autre façon que l' un pourrait attendre au premier. Le concept fondamental est simple : une solution optimale est la somme de sous-problèmes résolus de manière optimale .En un résultat, diviser un problème en sous-problèmes et résoudre les un par un.

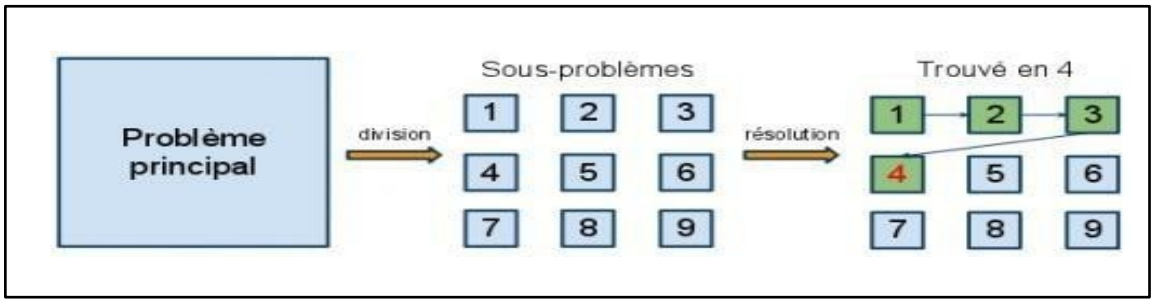

Figure 2: divisé en sous-problèmes

- **Algorithme général de programmation dynamique :**

La conception d'un algorithme de programmation dynamique peut être planifiée dans une séquence de quatre étapes.

1. Caractériser la structure d'une solution optimale.
2. Définir récursivement la valeur d'une solution optimale.
3. Calculer la valeur d'une solution optimale en remontant progressivement jusqu'à l'énoncé du problème initial.
4. Construire une solution optimale pour les informations calculées.

Les méthodes approchées ou heuristique :

Une méthode approchée ou heuristique est une méthode d'optimisation dont objectif est de trouver une solution possible à l'objectif fonction dans une raisonnable quantité de temps, mais avec aucune garantie de optimalité .Le principal avantage de ces méthodes est que ils peuvent être appliqués à tout le type de problème, ne importe comment simple**,** ou difficile. Dun autre côté les algorithmes d'optimisation tels que les algorithmes de recuit simulé, les algorithmes tabous et les algorithmes génétiques ont démontré leurs robustesses et efficacités face à plusieursproblèmes d'optimisation combinatoires [4].

Les méthodes approchées englobent deux classes :

- Les méthodes constructives,
- Les métaheuristiques,

Les méthodes constructives :

Les méthodes itératives sont celles qui construisent une solution pièce par pièce.A partir d' une partielle solution qui est d' abord vide, ils tentent d' étendre la partie solution de la précédente étape à chaque étape, et ce processus est répété jusqu'à ce que une complète solution est obtenue.

- **Utilisation :** Les méthodes constructives sont généralement utilisables quand la qualité de solution n'est pas un facteur primordial ou la taille de l'instance est raisonnable, en l'occurrence pour générer une solution initiale dans une métaheuristique, ces méthodes rapides et faciles d'implémentation.

L'Algorithme glouton :

La construction d' une pratique solution basée sur une série de décisions qui sont prises mieux chaque fois en fonction sur un locale critère, sans appel en question précédentes décisions. Dans la plupart des cas, la solution obtenue est acceptée**.**

- **Avantage** : algorithmes simples à implémenter,
- **Inconvénient** : solutions approchées obtenues plus ou moins bonnes, critère local.

Les Métaheuristiques :

Les métaheuristiques sont des heuristiques modernes inspirées de la nature, dédiées à la résolution de problèmes, en particulier des problèmes d'optimisation, dans le but d' atteindre un optimum global généralement enfoui dans une mer d' optimums locaux .Les métaheuristiques qui se subdivisent en deux sous-classes :

Les métaheuristiques sont divisées en deux sous-catégories :

- Les méthodes de voisinage.
- Les méthodes évolutives.

i. Les méthodes de voisinage :

Ces méthodes de départ avec une première solution (obtenue à une précision manière, ou par hasard échantillonnage) et peu à peu se déplacer loin de lui pour créer une trajectoire, ou un progressive chemin à travers l'espace de solutions. Les éléments suivants entrent dans cette catégorie :

- le recuit simulé.
- la méthode Tabou le terme de recherche locale est de plus en plus utilisée pour qualifier ces méthodes.

a) Le recuit simulé :

La méthode du circuit simulé est une généralisation de la méthode Monte-Carlo , dans le but de trouver la meilleure solution à un problème donné .Elle a été développée par trois IBM chercheurs, S. Kirkpatrick, CD Gelatt, et MP Vecchi, en 1983, et indépendamment par V. Cerny en 1985, basée sur le Metropolis algorithme, qui permet un de décrire l' évolution d' un thermodynamique système. [5].

La principale idée derrière la simulation recuit, comme proposé par Metropolis en 1953, est de simuler le comportement de la matière dans un recuit processus qui est largement utilisé dans la métallurgie. Le but est d'atteindre un équilibre thermodynamique ; cet équilibre (où l' énergie est minimale) représente la solution optimale à un problème dans la méthode de simulation . Le système de l' énergie sera être calculé en utilisant un coût fonction (ou fonction Objectif). Kirkpatrick, En gros le principe consiste à générer successivement des configurations à partir d'une solution initiale S_0 et d'une température initiale T_0 qui diminuera tout au long du processus jusqu'à atteindre une température finale ou un état d'équilibre (optimum global).

- **Algorithme général de recuit simulé :**

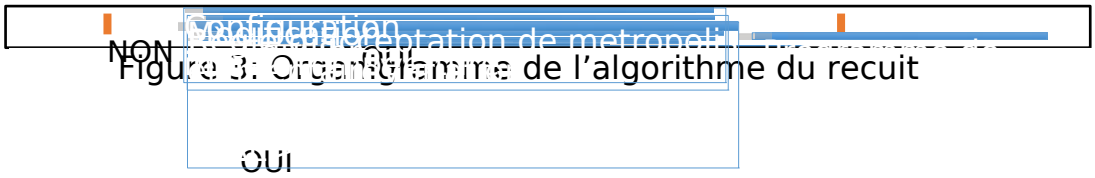

Figure 3: Organigramme de l'algorithme du recuit

b) La recherche tabou :

F. Glover [6] a formalisé la méthode tabou de recherche, ou simplement méthode tabou .Sa principale caractéristique distinctive est la mise en œuvre de mécanismes inspirés de la mémoire humaine .L' idée est de garder trace de votre voyage dans une mémoire et reportez - vous à ce quand vous cherchez pour quelque chose**.**

- **Notion de base :**

À chaque itération, l'algorithme tabou choisit le meilleur voisin non tabou, même si celui-ci dégrade la fonction de coût. Pour cette raison, on dit de la recherche avec tabou qu'elle est une méthode agressive.

- **La liste taboue (mémoire) :**

L’idée de base derrière la liste taboue est de mémoriser les configurations ou régions qui ont été visitées et de mettre en place des mécanismes qui empêchent la recherche de revenir trop rapidement à ces configurations.

- **Algorithme générale :**

Figure 4: Organigramme de l'algorithme tabou

ii. Les méthodes évolutives :

Ces méthodes sont distinguées à partir précédemment étudiés méthodes par le fait que ils opèrent sur une population de solutions; comme un résultat, ils sont souvent appelés à comme population méthodes. Certains d' entre eux sont basés sur les principes de la génétique et du comportement des insectes . La complexité de ces deux phénomènes biologiques a servi de modèle pour des algorithmes toujours plus sophistiqués ces vingt dernières années [7]. Nous avons retenu seulement par les algorithmes génétiques.

a) Les algorithmes génétiques :

Les algorithmes génétiques sont inspirés de la théorie de l'évolution et des processus biologiques qui permettent à des organismes de s'adapter à leur environnement. Ils ont été inventés dans le milieu des années 60 (Holland, 1962; Rechenberg, 1965; Fogel et al, 1966) [8].

La sélection naturelle, que Darwin appelle l'élément "propulseur" de l'évolution, favorise les individus d'une population qui sont le mieux adaptés à un environnement. La sélection est suivie de croisements et de mutations au niveau des individus, constitué d'un ensemble de gènes. Ainsi deux

individus "parents", qui se croisent, transmettent une partie de leur patrimoine génétique à leurs descendants. L'individu enfant fait que celui-ci est plus au. Au fur et à mesure des générations son sélectionne les individus les mieux adaptés, et l'augmentation du nombre des individus bien adaptés fait évoluer la population entière [9].

La mise en œuvre des algorithmes génétiques nécessite plusieurs étapes à détailler. La première est le codage d'un individu représenté par un chromosome. La seconde est le calcul de la qualité. La troisième est de définir les opérateurs de reproduction

- **Algorithme générale :**

Figure 5: Fonctionnement des algorithmes génétiques.

b) Codage des individus :

Un principe de codage de l'élément de population. Cette étape associe à chacun des points de l'espace d'état une structure de données. Elle se place généralement après une phase de modélisation mathématique du problème traité. La qualité du codage des données conditionne le succès des algorithmes génétiques. Les codages binaires ont été très utilisés à l'origine. Les codages réels sont désormais largement utilisés, notamment dans les domaines applicatifs pour l'optimisation de problèmes à variables réelles.

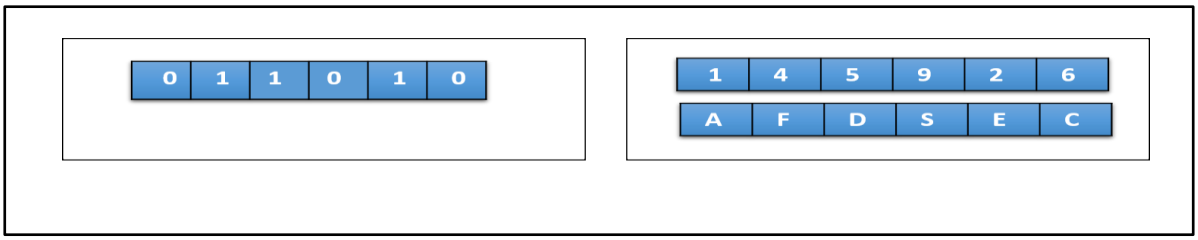

Figure 6: Codage des solutions

Les algorithmes génétiques utilisent trois opérateurs pour générer de nouvelles solutions :

1. L'opérateur de sélection qui permet de choisir des solutions parentes sur lesquelles la reproduction va être faite pour générer des nouvelles solutions.

2. L'opérateur de croisement qui permet de croiser les deux solutions parentes et créer de nouvelles solutions.

3. L'opérateur de mutation qui permet de diversifier les nouvelles solutions afin qu'elles ne ressemblent pas trop aux solutions parentes.

L'opérateur de sélection :

La sélection consiste à choisir des individus qui permettront de générer de nouveaux individus. Plusieurs méthodes existent pour sélectionner des individus destinés à la reproduction. On citera les deux méthodes classiques les plus utilisées.

La sélection par la roulette :

La sélection des individus par le système de la roulette s'inspire des roues de loterie [10]. À chacun des individus de la population est associé un secteur d'une roue. L'angle du secteur étant proportionnel à la qualité de l'individu qu'il représente. Vous tournez la roue et vous obtenez un individu. Les tirages des individus sont ainsi pondérés par leur qualité. Et presque logiquement, les meilleurs individus ont plus de chance d'être croisés et de participer à l'amélioration de notre population. illustre une population de 5 individus dont les performances sont représentées en roulette.

La sélection par tournoi : Le principe de la sélection par tournoi augmente les chances pour les individus de piètre qualité de participer à l'amélioration de la population. Le principe est très rapide à implémenter. Un tournoi consiste en une rencontre entre plusieurs individus pris au hasard dans la population. Le vainqueur du tournoi est l'individu de meilleure qualité. Cette méthode est en général satisfaisante.

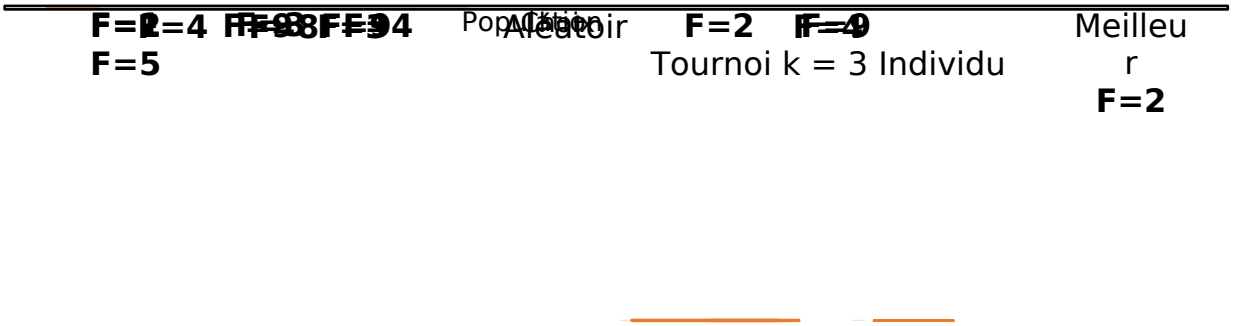

Figure 7: Sélection par tournoi.

L’opérateur de croisement:

Les croisements permettent de simuler des reproductions d'individus dans le but d'en créer des nouveaux. Il est tout à fait possible de faire des croisements aléatoires. Toutefois, une solution largement utilisée est d'effectuer des croisements multi-points.

Le croisement à un point : Il a été initialement défini pour le codage binaire. Le principe consiste à tirer aléatoire une position pour chaque parent et à échanger les sous- chaines des parents à partir des positions tirées. Ce qui donne naissance à deux nouveaux individus ind 1 et ind 2 (Figure 8).

Figure 8: Croisement à un point.

Les croisements multi-points : Elle reprend le mécanisme de la méthode de croisement à un point en généralisant l’échange à 3 ou 4 sous chaines.

Figure 9 : Croisement à deux points.

Nous avons présenté au cours de ce chapitre l'optimisation combinatoire de façon détaillé ; Nous essayons de répondre, Qu'est-ce qu'un problème d'optimisation et quelle est les différents méthodes de résolutions. Nous expliquons la notion des problèmes d'optimisation combinatoire et c'est-à-dire que les méthodes de résolution peuvent être réparties en deux grandes classes : Les méthodes exactes, Les méthodes approchées et chacun de ces méthodes diviser en plusieurs classes.

L'opérateur de Mutation :

L'opération de mutation protège les algorithmes génétiques des pertes prématurées d'informations pertinentes. Elle permet d'introduire une certaine information dans la population, qui a pu être perdue lors de l'opération de croisement. Ainsi elle participe au maintien de la diversité, utile à une bonne exploration du domaine de recherche (Figure 10).

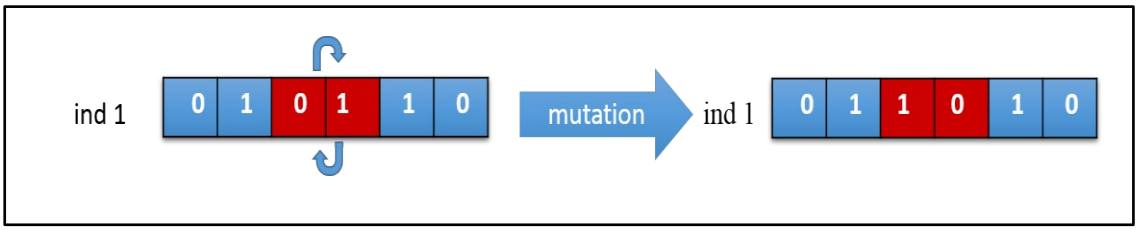

Figure 10: Une mutation.

L'opérateur de remplacement :

Cet opérateur est consisté à réintroduire les descendants obtenus par application successive des opérateurs de sélection, de croisement et de mutation (la population P') dans la population de leurs parents (la population P). On trouve des méthodes de remplacement différentes par exemple méthode de remplacement stationnaire **[11]**.

Le remplacement stationnaire : dans ce cas, les enfants remplacent automatiquement les parents sans tenir compte de leurs performances respectives, et le nombre d'individus de la population ne varie pas tout au long du cycle d'évolution simulé, ce qui implique donc d'initialiser la population initiale avec un nombre suffisant d'individus. Cette méthode peutêtre mise en œuvre de 2 façons différentes :

Nous avons présenté au cours de ce chapitre l’optimisation combinatoire de façon détaillé ; Nous essayons de répondre, Qu'est-ce qu'un problème d'optimisation et quelle est les différents méthodes de résolutions. Nous expliquons la notion des problèmes d’optimisation combinatoire et c’est-à-dire que les méthodes de résolution peuvent être réparties en deux grandes classes : Les méthodes exactes, Les méthodes approchées et chacun de ces méthodes diviser en plusieurs classes.

4. La première se contente de remplacer la totalité de la population P par la population P', cette méthode est connue sous le nom de remplacement générationnel**.**

5. La deuxième méthode consiste on remplace les mauvaise individus trouvées dans la population P par les meilleurs de la population P'.

3. Conclusion :

Chapitre II: Algorithme Des feux d'artifice

Chapitre II: Algorithme Des Feux D'artifice

1. Introduction :

L’utilisation des méthodes exactes pour la résolution des problèmes d’optimisation combinatoire est très coûteuse de temps, certain cas impossible dû à leur temps d’exécution exponentiel. Les recherches sont orientées vers l’utilisation des métaheuristiques permettent des gagner du temps et de l’espace.

Dans ce chapitre nous essayons présenter l'algorithme des feux d'artifice comme une méthode d’optimisation par essaim. La définition, les étapes de l’algorithme et les concepts fondamentaux.

En fin, nous présentons l’algorithme des feux d’artifice discret pour l’utilisation dans le dernier chapitre comme un des objectifs de notre thème.

2. Algorithme des feux d'artifice :

L'algorithme des feux d'artifice forme une nouvelle classe des métaheuristiques récemment proposée pour résoudre les problèmes d'optimisation difficile. Cet algorithme s'inspire par l'observation des comportements des explosions des feux d'artifice dans le ciel à la nuit. Souvent, les algorithmes des feux d'artifice sont classés comme étant une nouvelle branche de l'intelligence en essaime (Swarn Intelligence).

3. .Métaphore d'inspiration:

Le premier algorithme de ce type a été proposé par Ying Tan et Yuanchun Zhu en 2010 [3], à travers l'observation du fait que l'explosion d'un feu d'artifice est similaire à la façon dont un individu cherche une solution optimale dans un algorithme issue de l'intelligenceen essaim.

4. .Motivation de l'algorithme des feux d'artifice :

L'algorithme des feux d'artifice (Fireworks Algorithm) est un nouvel algorithme récemment développé de nature de l'intelligence en essaim. Il prend sa base théorique par la simulation du processus d'explosion des feux d'artifice. En analogie avec de véritables feux d'artifice explosant et illuminant le ciel nocturne, les feux d'artifice (c'est-à-dire les individus) dans cet algorithme sont laissés à l'espace de recherche potentiel. Pour chaque feu d'artifice, un processus d'explosion est initié et une pluie d'étincelles remplit l'espace local qui l'entoure. Les feux d'artifice ainsi que les nouvelles étincelles générées représentent des solutions potentielles dans l'espace de recherche.

Par analogie à d'autres algorithmes d'optimisation, l'objectif est de trouver une bonne solution (idéalement l'optimale) d'un problème d'optimisation combinatoire, qui est généralement définie par une fonction objective.

5. .Framework général de l'algorithme des feux d'artifices :

Quand un feu d'artifice est déclenché, une pluie d'étincelles remplira l'espace local autour du feu d'artifice. Selon les auteurs de cet algorithme le processus d'explosion d'un feu d'artifice est considéré comme étant une recherche locale dans l'espace autours d'un point bien déterminé où le feu d'artifice est déclenché par les étincelles générées dans l'explosion [3].

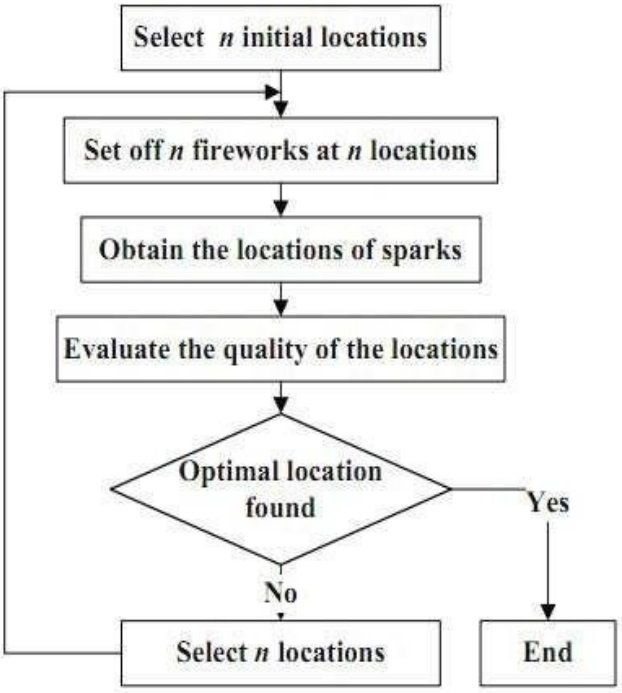

Figure 11: Framework standard de l'algorithme des feux d'artifice

1. Au début, N feux d'artifice (Fireworks) sont initialisés aléatoirement, et leur qualité (c.-à-d. Leur condition physique) est évaluée afin de déterminer l'amplitude de l'explosion et le nombre d'étincelles pour chaque feu d'artifice.

2. Par la suite, les feux d'artifice explosent et génèrent différents types d'étincelles (Sparks) dans leur espace local.

3. Enfin, N feux d'artifice candidat sont sélectionnés parmi l'ensemble des candidats (après l'évaluation de leurs qualités), ce qui inclut les étincelles nouvellement générées ainsi que les N feux d'artifice originaux.

Afin d'assurer la diversité et d'équilibrer la recherche globale et locale, l'amplitude de l'explosion et la population des étincelles d'explosion nouvellement générées diffèrentparmi les feux d'artifice.

6 .Définition d'un feu d'artifice (Firework) :

Un bon feu d’artifice (Good Firework) :

Un feu d’artifice avec une meilleure forme physique peut générer une plus grande population d’étincelles d’explosion dans une gamme plus petite, c’est-à-dire avec une petite amplitude d’explosion (voir Figure 12-A).

6.2 .Un mauvais feu d'artifice (Bad Firework) :

Au contraire, les feux d'artifice ayant une forme physique inférieure ne peuvent générer qu'une population plus petite dans une plage plus large, c'est-à-dire avec une amplitude d'explosion plus élevée (voir Figure 12-B).

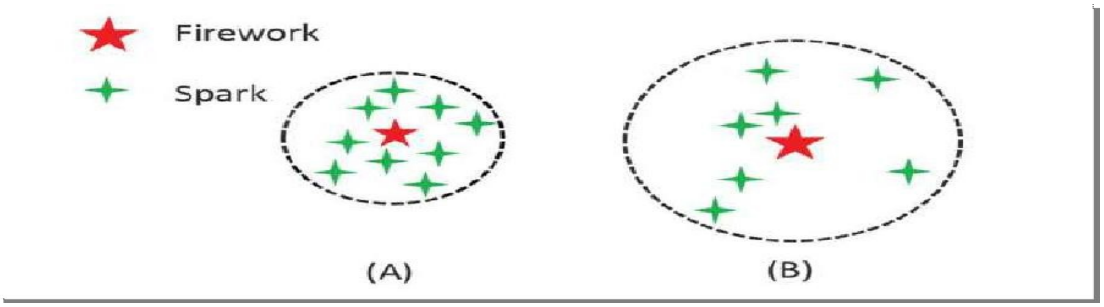

Figure 12: Les bonnes et les mauvaises explosions d'un feu d'artifice A. une bonne explosion. B une mauvaise explosion

Cette technique permet d'équilibrer les capacités d'exploration et d'exploitation de l'algorithme. Alors que l'exploration fait référence à la capacité de l'algorithme à explorer différentes régions de l'espace de recherche afin de localiser de bonnes solutions prometteuses, l'exploitation fait référence à la possibilité de mener une recherche approfondie dans une zone plus petite reconnue comme prometteuse [3].

7 .Les principaux processus dans l'algorithme des feux d'artifice :

7 .1 .Le processus d'exploration :

L'exploration est réalisée par les bons feux d'artifice qui ont une grande amplitude d'explosion, puisqu'ils ont la capacité d'échapper aux minima locaux (voir Figure 12-A)

7.2 .Le processus d'exploitation :

L'exploitation est réalisée par les mauvais feux d'artifice qui ont une faible amplitude d'explosion, puisqu'ils renforcent la capacité de recherche locale dans des zones prometteuses (voir Figure 12- B).

7.3. Le processus de mutation :

Après l'explosion, un autre type d'étincelles est généré sur la base d'une mutation gaussienne des feux d'artifice choisis au hasard. L'idée derrière ceci est d'assurer davantagela diversité de l'essaim.

Afin d'améliorer la lisibilité, les fondateurs de l'algorithme assignent de nouvelles notations aux deux types d'étincelles distinctes: les étincelles d'explosion sont générées par le processus d'explosion, et les étincelles gaussiennes sont générées par une mutation gaussienne [12].

8 .Les paramètres principaux de l'algorithme des feux d'artifice :

Comme toute méthode d'optimisation, l'efficacité de l'algorithme des feux d'artifice dépend de certains paramètres qui sont:

L'opérateur d'explosion :

Lors de l'initialisation, l'algorithme génère N feux d'artifice de manière aléatoire. Ensuite, les N feux d'artifice génèrent des étincelles par les opérations d'explosion. L'opérateur d'explosion est un facteur clé dans l'algorithme et joue un rôle très important. En effet, ce dernier inclut la force d'explosion, l'amplitude de l'explosion et l'opération de déplacement [TAN 13].

La force de l'explosion (le nombre des étincelles) :

Supposons que l'algorithme est désigné pour le problème d'optimisation suivant :

$$\text{Min } f(x) \in R, x_{min} \leq x \leq x_{max}$$

Tel que $x = x_1, x_2, \ldots, x_n$ indique un emplacement dans l'espace potentiel de recherche,f(x) est une fonction objective, et x_{min} et x_{max} désignent les limites de l'espace potentiel. Le nombre des étincelles générés par chaque feu d'artifice x_i est défini par l'équation suivante :

$$s_i = m \cdot \frac{y_{max} - f(x_i) + \xi}{\sum_{i=1}^{n} (y_{max} - f(x_i)) + \xi}$$

Où **m** est un paramètre qui contrôle le nombre total d'étincelles générées par les N feux d'artifice,

Ymax = max (f (x_i)) (i = 1, 2, …, n) est le maximum (le pire) valeur de la fonction objectif parmi les N feux d'artifice, et **ξ**, qui dénote la plus petite constante dans l'ordinateur, est utilisée pour éviter l'erreur de division par zéro. Pour éviter les effets accablants des feux d'artifice splendides, des limites sont définies pour si, comme montré dans l'équation suivante :

$$\hat{si} = \begin{cases} round\ (a.m)\ si\ Si < am \\ round\ (b.m) si\ Si > bm,\ \ a < b < 1 \\ round\ (Si) Sinon \end{cases}$$

Avec a et b sont des paramètres constants.

L'amplitude de l'explosion :

Contrairement à la conception du nombre d'étincelles, l'amplitude d'un bon feu d'artifice est plus faible que celle d'un mauvais feu d'artifice. L'amplitude d'explosion pour chaque feu d'artifice est définie par l'équation suivante:

$$A_i = \hat{A} \cdot \frac{f(\boldsymbol{x_i}) - y_{\min} + \xi}{\sum_{i=1}^{n} (f(\boldsymbol{x_i}) - y_{min}) + \xi}$$

Où **Â** désigne l'amplitude maximale de l'explosion, et y_{min}=min (f (x_i)) (i=1, 2, . . ., n) est la valeur minimale (la meilleure) de la fonction objectifparmi les N feux d'artifice.

Génération des étincelles:

Dans le processus d'explosion, des étincelles peuvent subir les effets de l'explosion à partir de z directions aléatoires (z dimensions). Dans l'algorithme des feux d'artifice, nous obtenons le numéro de directions affectées au hasard par la formule suivante:

z = round (d * rand (0, 1)),

Où **d** est la dimensionnalité de l'emplacement x, et rand (0, 1) est une distribution uniformesur [0,1].

L'emplacement d'une étincelle du feu d'artifice x_i est obtenu à l'aide de l'algorithme 1.

Algorithm 1. Obtain the location of a spark

Initialize the location of the spark: $\tilde{\boldsymbol{x}}_j = \boldsymbol{x}_i$;
z=$round(d \cdot rand(0,1))$;
Randomly select z dimensions of $\tilde{\boldsymbol{x}}_j$;
Calculate the displacement: $h = A_i \cdot rand(-1,1)$;
for each dimension $\tilde{x}_k^j \in$ {pre-selected z dimensions of $\tilde{\boldsymbol{x}}_j$} **do**
  $\tilde{x}_k^j = \tilde{x}_k^j + h$;
  if $\tilde{x}_k^j < x_k^{\min}$ or $\tilde{x}_k^j > x_k^{\max}$ **then**
    map $\tilde{x}_k^j$ to the potential space: $\tilde{x}_k^j = x_k^{\min} + \mid \tilde{x}_k^j \mid \%(x_k^{\max} - x_k^{\min})$;
  end if
end for

Explosion gaussienne (mutation gaussienne) :

Pour garder la diversité des étincelles, les auteurs de l'algorithme utilisent une autre façon de générer des étincelles selon l'algorithme 2.

La fonction Gaussienne (1, 1) dénote une distribution gaussienne avec la moyenne 1 et l'écart-type 1, est utilisé pour définir le coefficient de l'explosion.

D'après les expériences des auteurs, mˆ étincelles de ce type sont générés dans chaque génération d'explosion.

Algorithm 2. Obtain the location of a specific spark

Initialize the location of the spark: $\hat{\boldsymbol{x}}_j = \boldsymbol{x}_i$;
z=$round(d \cdot rand(0,1))$;
Randomly select z dimensions of $\hat{\boldsymbol{x}}_j$;
Calculate the coefficient of Gaussian explosion: $g = Gaussian(1,1)$;
for each dimension $\hat{x}_k^j \in$ {pre-selected z dimensions of $\hat{\boldsymbol{x}}_j$} **do**
  $\hat{x}_k^j = \hat{x}_k^j \cdot g$;
  if $\hat{x}_k^j < x_k^{\min}$ or $\hat{x}_k^j > x_k^{\max}$ **then**
    map $\hat{x}_k^j$ to the potential space: $\hat{x}_k^j = x_k^{\min} + \mid \hat{x}_k^j \mid \%(x_k^{\max} - x_k^{\min})$;
  end if
end for

L'opération de déplacement :

Au début de chaque génération d'explosion, N emplacements doivent être sélectionnés pour l'explosion des feux d'artifice. Dans l'algorithme de feux d'artifice, le meilleur emplacement actuel *x**, sur lequel la fonction objectif *f (x*)* est optimale parmi les emplacements actuels, est toujours conservée pour la prochaine génération d'explosion. Après cela, N - 1 emplacements sont sélectionnés en fonction de leur distance à d'autres endroits afin de garder la diversité des étincelles.

La distance entre un emplacement x_i et d'autres emplacements est définie comme suit.

$$R(\boldsymbol{x}_i) = \sum_{j \in K} d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{j \in K} \parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel$$

Où K est l'ensemble des emplacements actuels des feux d'artifice et des étincelles.
Ensuite, la probabilité de sélection d'un emplacement x_i est définie comme suit.

$$p(\boldsymbol{x_i}) = \frac{R(\boldsymbol{x_i})}{\sum_{j \in K} R(\boldsymbol{x_j})} \quad .$$

Pour le calcul de la distance, plusieurs mesures de distance peuvent être utilisées, comme la distance de Manhattan et la distance euclidienne.

9 .Algorithme de feu d'artifice discret :

Les d'optimisation discrets sont caractérisés par un manque de continuité. En conséquence, un changement mineur dans la solution peut avoir un impact significatif sur le résultat. En conséquence, nous devons considérer comment utiliser les caractéristiques uniques du problème d'optimisation plutôt que d'appliquer simplement le cadre FWA. Nous utilisons 2-opt et 3-opt comme opérations de base pour générer des étincelles dans l'explosion de feux d'artifice en raison des performances exceptionnelles de l'échange des mots pour les requêtes. De plus, une méthode de mutation spécifique basée sur l'insertion est utilisée pour générer différents types d'étincelles à partir de l'explosion. Nous utilisons également une stratégie adaptative pour contrôler l'amplitude de l'explosion, qui est déterminée par la progression précédente de la meilleure composition des mots plutôt que par un simple calcul basé sur la qualité et les emplacements actuels des feux d'artifice. Une méthode de sélection spéciale basée sur la qualité des feux d'artifice (étincelles) est utilisée. Sur une variété de données, nous montrons les résultats des expériences numériques du DFWA, de l'algorithme génétique et de tout algorithme de système de colonie.

Le principe FWA est suivi par notre DFWA. Il peut également être utilisé pour rechercher la meilleure solution d'enchainement des mots des requêtes en simulant des feux d'artifice. Il est composé de quatre composants majeurs en général:

1) Initialisation des feux d'artifice Cette étape consiste à sélectionner les emplacements initiaux des feux d'artifice, ce qui implique de sélectionner un nombre aléatoire des mots de requête.

2) L'explosion. Les feux d'artifice explosent à cette étape, provoquant des étincelles d'explosion. Premièrement, la force et l'amplitude de l'explosion doivent être Déterminées. la ténacité représente le nombre d'étincelles produites par ce feu d'artifice. Plus la force est grande, mieux c'est. Plus il produit de feux d'artifice, mieux c'est. La distance possible (différence) entre un feu d'artifice et les étincelles qu'il produit est appelée amplitude. Plus l'amplitude est grande, plus la différence de potentiel est grande. être, et plus l'étude de la région des possibles devient plus vaste. L'explosion commence après le calcul de ces deux paramètres.

3) La mutation est un terme utilisé pour décrire le processus En complément de l'échange de bords, nous utilisons la méthode d'insertion pour générer plus d'étincelles dans cette étape. Chaque feu d'artifice produira un certain nombre d'étincelles de mutation.

4) Choisir Dans cette étape, nous utilisons les feux d'artifice et les étincelles actuels pour déterminer les emplacements des feux d'artifice lors de la prochaine itération. Tous les feux d'artifice et étincelles seront pris en compte. La probabilité d'en ramasser d'autres sera alors calculée. Selon cette probabilité, l'algorithme en choisit un au hasard. Après cela, le critère de terminaison est vérifié pour voir s'il est satisfait ou non. Si la réponse est oui, l'algorithme se termine et le meilleur tour est retourné. Sinon, les feux d'artifice et les étincelles sélectionnés dans cette étape seront utilisés pour créer des feux d'artifice lors de la prochaine itération, et l'algorithme passera à l'étape 2 Explosion.

10 .Conclusion :

Parmi les méthodes approchées de résolution des problèmes d'optimisation combinatoire nous avons choisis en cour de ce chapitre, un nouveau algorithme nommé l'algorithme des feux d'artifice, celui-ci été créer en 2010 par Ying Tan et Yuanchun Zhu à travers l'observation de l'explosion d'un feu d'artifice.

Dans ce chapitre nous avons définit l'algorithme originale et ces processus puis, sa mise en œuvre et la notion d'algorithme de feu d'artifice discret.

Chapitre III: Métaheuristiques parallèles

1. Introduction :

Avec la prolifération des ordinateurs parallèles, des stations de travail puissantes et des réseaux de communication rapides, les implémentations parallèles de métaheuristiques apparaissent tout naturellement comme une alternative à l'accélération de la recherche de solutions approximatives, elles permettent également de trouver des solutions améliorées par rapport à leurs homologues séquentielles, en raison de la partition de l'espace de recherche et de plus de possibilités d'intensification et de diversification de recherche.

Le parallélisme est un moyen non seulement de réduire le temps de fonctionnement des métaheuristiques, mais aussi d'améliorer leur efficacité et leur robustesse. Ces derniers sont susceptibles d'être la contribution la plus importante du parallélisme aux métaheuristiques.

Dans ce chapitre nous avons concentré l'étude sur la notion de parallélisassions des métaheuristiques et les raison, puis La modélisation parallèle des métaheuristiques et Les architectures parallèles, et en fin Les schémas d'exécution des algorithmes évolutionnaire a base parallèle.

2. Les raisons de parallélisassions des métaheuristiques :

Le calcul parallèle et distribué peut être utilisé dans la conception et l'implémentation des métaheuristiques pour les raisons suivantes [13] :

A. Accélérer la recherche : L'un des objectifs principaux de la parallélisassions d'une métaheuristique est de réduire le temps de recherche. Cela permet de concevoir des méthodes d'optimisation interactives et en temps réel. Ceci est un aspect très important pour une classe de problèmes où il y a des exigences strictes sur le temps de recherche, comme dans les problèmes d'optimisation dynamiques et les problèmes de contrôle de temps critique tels que la planification "en temps réel".

B. Améliorer la qualité des solutions obtenues : Certains modèles parallèles de métaheuristiques permettent d'améliorer la qualité de la solution trouvée. En effet, l'échange d'informations entre les métaheuristiques coopératives va modifier leur comportement en termes de recherche dans l'espace associé au problème. L'objectif principal d'une coopération parallèle entre les métaheuristiques est d'améliorer la qualité des solutions. Une meilleure convergence et un temps de recherche réduit peuvent se produire. Notons qu'un modèle parallèle pour les métaheuristiques peut être plus efficace qu'une métaheuristique séquentielle même sur un seul processeur.

C. Améliorer la robustesse : Une métaheuristique parallèle peut être plus robuste en termes de résoudre de manière efficace différents problèmes d'optimisation et différentes instances d'un problème donné. La robustesse peut également être mesurée en termes de sensibilité de la métaheuristique à ses paramètres.

D. Résoudre des problèmes à grande échelle :

Les métaheuristiques parallèles permettent de résoudre les instances à grande échelle de problèmes d'optimisation complexes. Un défi ici est de résoudre très grandes instances qui ne peuvent pas être résolues par une machine séquentielle. Un autre défi similaire consiste à résoudre des modèles mathématiques plus précis associés à différents problèmes d'optimisation. L'amélioration de la précision des modèles mathématiques augmente, en général, la taille des problèmes associés à résoudre. De plus, Certains problèmes d'optimisation nécessitent la manipulation des bases de données énormes telles que des problèmes d'exploration de données.

3. La modélisation parallèle des métaheuristiques :

La raison qui pousse les chercheurs à utiliser les modèles parallèles dans les métaheuristiques est l'importante demande en puissance de calcul pour les problèmes à résoudre. En termes de conception de métaheuristiques parallèles, trois grands modèles parallèles sont identifiés [13] .

Le modèle parallèle au niveau de l'algorithme :

Ce modèle parallèle consiste à paralléliser des algorithmes (parallélisassions inter-algorithme). Il ne dépend pas du problème traité. Si les algorithmes sont indépendants les uns des autres, leur parallélisassions n'apporte qu'une accélération de leur exécution, Il n'y a pas d'amélioration de la qualité des solutions trouvées par rapport au modèle séquentiel. En revanche, si les algorithmes sont coopératifs, leur parallélisassions peut non seulement réduire leur temps d'exécution, mais aussi améliorer les solutions trouvées par rapport au modèle séquentiel.

Le modèle parallèle au niveau de l'itération :

Dans ce modèle, chaque itération d'une métaheuristique est parallélisée (parallélisassions intra-algorithme), indépendamment du problème traité. Le comportement de la métaheuristique n'est pas modifié. L'objectif principal est d'améliorer l'algorithme en réduisant le temps de recherche, et non pas les solutions trouvées par rapport au modèle séquentiel. En effet, le cycle d'itération des métaheuristiques sur les grands voisinages pour les S- métaheuristiques ou les grandes populations pour les P-métaheuristiques nécessite une grande quantité de ressources de calcul, en particulier pour les problèmes du monde réel.

Le modèle parallèle au niveau de la solution :

Dans ce modèle, le processus de parallélisassions gère une seule solution de l'espace de recherche (parallélisassions intra-algorithme) dépendant du problème. En général, l'évaluation de la ou des fonction (s) objective (s) ou des contraintes pour une solution générée est souvent l'opération la plus coûteuse en

métaheuristiques. Dans ce modèle, le comportement de la métaheuristique n'est pas modifié. L'objectif est principalement l'accélération de la recherche.

4. Les architectures parallèles :

Un modèle d'architecture parallèle est une description abstraite d'un ordinateur parallèle dont le but est de représenter les particularités les plus importantes de la machine [14].

Plusieurs classifications des ordinateurs parallèles sont proposées dans la littérature. La classification la plus connue est celle de Flynn [94]. Elle est fondée sur l'organisation des flots d'instructions (séquences d'instructions transmises à partir d'une unité de contrôle vers un ou plusieurs processeurs) et des flots de données (séquences de données provenant de la mémoire et se dirigeant vers un processeur ou provenant d'un processeur et se dirigeant vers la mémoire). Selon la classification de Flynn, quatre architectures sont proposées pour gérer les modèles de parallélisassions Flynn développe quatre types d'organisation pour une machine de donnée. On distingue alors les architectures suivantes:

5. Les schémas d'exécution des algorithmes évolutionnaire a base parallèle :

Un des nombreux avantages des algorithmes évolutionnaires est qu'ils sont faciles à paralléliser. Le processus d'évolution artificielle peut être implémenté de différentes manières sur des machines parallèles. Il est possible de paralléliser des opérations spécifiques ou de paralléliser le processus évolutif lui-même. Cette dernière approche a conduit à une variété d'algorithmes de recherche [16].

Modèle maître-esclave (Master-Slave) :

Il existe plusieurs façons d'utiliser des machines parallèles. Un moyen simple d'utiliser la parallélisassions consiste à exécuter des opérations sur des processeurs séparés. Cela peut concerner les opérateurs de variation comme la mutation et la recombinaison ainsi que les évaluations de fonctions. En fait, cela a plus de sens pour les évaluations de fonctions car ces opérations peuvent être effectuées indépendamment et elles sont souvent parmi les opérations les plus coûteuses. Dans ce modèle, une machine représente le maître et distribue la charge de travail pour l'exécution des opérations sur plusieurs autres machines appelées esclaves. Il est bien adapté à la création des descendants des populations car il est possible de créer et d'évaluer ces derniers, une fois que les parents appropriés ont été sélectionnés.

Le système est généralement synchronisé, où le maître attend que tous les esclaves aient terminé leurs opérations avant de poursuivre. Cependant, il est possible d'utiliser des systèmes asynchrones où le maître n'attend pas les esclaves trop longs. Le comportement des modèles maître-esclave synchronisés n'est pas différent de leurs homologues séquentiels. L'implémentation est différente, mais l'algorithme est le même.

Modèle d’exécutions indépendantes (IndependentRuns) :

Des machines parallèles peuvent également être utilisées pour simuler différentes exécutions indépendantes du même algorithme en parallèle. Un tel système est très facile à mettre en place car aucune communication pendant le temps d'exécution n'est nécessaire. Une fois que toutes les analyses ont été arrêtées, les résultats doivent être collectés et la meilleure solution (ou une sélection de différentes solutions de haute qualité) est produite. Alternativement, toutes les machines peuvent communiquer périodiquement leurs meilleures solutions actuelles afin que le système puisse être arrêté dès qu'une solution satisfaisante a été trouvée. En ce qui concerne les modèles maître-esclave, cela nous évite d'attendre que la plus longue exécution se termine. Malgré sa simplicité, les exécutions indépendantes peuvent être assez efficaces. Considérons un paramètre où une seule exécution d'un algorithme a une probabilité de succès particulière, c.-à-d., une probabilité d'obtenir une solution satisfaisante dans un délai donné. Que cette probabilité soit notée p. En utilisant plusieurs exécutions indépendantes, cette probabilité de succès peut être augmentée de façon significative. Cette approche est communément appelée amplification de probabilité.

La probabilité que dans λ exécutions indépendants, aucune exécution est réussie est $(1-p)^{\lambda}$. La probabilité qu'il y ait au moins une exécution réussie parmi ceux-ci est donc $1-(1-p)^{\lambda}$. Nous pouvons voir que pour un petit nombre de processeurs, la probabilité de succès augmente presque linéairement. Si le nombre de processeurs est important, un effet de saturation se produit. Le bénéfice d'utiliser toujours plus de processeurs diminue avec le nombre de processeurs utilisés. Le point où la saturation se produit dépend crucialement de p: pour de plus petites probabilités de succès, la saturation ne se produit qu'avec un nombre assez important de processeurs.

En outre, des exécutions indépendantes peuvent être mises en place avec différentes conditions initiales ou différents paramètres. Ceci est utile pour explorer efficacement l'espace de recherche et pour trouver de bons paramètres dans un temps plus court.

Modèle d'ilot (island model) :

Dans les modèles d'îlot, également appelés Algorithmes évolutionnaires distribués (distributed EAs,), modèle à grains grossiers (coarse-grained model) ou modèle multi-dèmes (multi-deme model), la population de chaque exécution est considérée comme un îlot. On parle souvent d'îlots en tant que sous-populations qui forment ensemble la population globale du modèle d'îlot. Les îlots évoluent indépendamment comme dans le modèle des exécutions indépendantes, la plupart du temps. Mais des solutions sont périodiquement échangées entre les îlots dans un processus appelé migration.

L'idée est d'avoir une topologie de migration, un graphe orienté avec des îlots comme nœuds et des arcs orientés reliant deux îlots. À certains moments, des individus sélectionnés de chaque îlot sont envoyés dans les îlots voisines, i. e., les îlots qui peuvent être atteintes par un arc orienté dans la

topologie. Ces individus sont appelés migrants et ils sont inclus dans l'îlot cible après un processus de sélection supplémentaire. De cette façon, les îlots peuvent communiquer et se faire concurrence. Un exemple de modèle d'îlot est donné à la Figure 13.

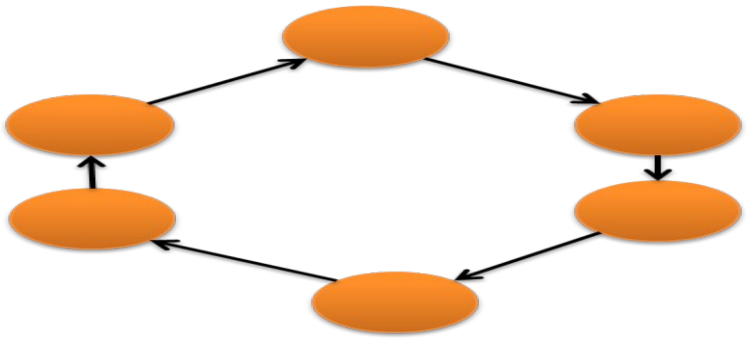

Figure 13 Exemple de topologie du modèle d'îlot.

Si tous les îlots utilisent le même algorithme dans des conditions identiques, on parle d'un modèle d'îlot homogène. Tandis que, les modèles d'îlots hétérogènes contiennent des îlots ayant des caractéristiques différentes, des représentations différentes, des fonctions objectives ou des paramètres différents, et même desalgorithmes différents peuvent être utilisés.

Modèle cellulaire des algorithmes évolutionnaires(Cellular EAs) :

Le modèle cellulaire représente un cas particulier de modèle d'îlot avec une forme de parallélisassions plus fine. Comme dans le modèle d'îlot, nous avons des îlots reliés par une topologie fixe. Les anneaux et les graphiques de tore bidimensionnels sont le choix le plus commun. La caractéristique la plus forte est que chaque îlot ne contient qu'un seul individu. Les îlots sont souvent appelés cellules dans ce contexte, ce qui explique le terme cellulaire. Chaque individu est seulement autorisé à s'accoupler avec ses voisins dans la topologie. Ce type d'interaction se produit à chaque génération.

Les modèles cellulaires donnent un système beaucoup plus fin; ils ont donc été appelés modèles à grain fin (fine-grained models), modèle de voisinage (neighborhood model) ou modèle de diffusion. La différence par rapport aux modèles d'îlot est qu'aucune évolution n'a lieu sur la cellule elle-même, i. e., il n'y a pas d'évolution intra- îlot. Les améliorations ne peuvent être obtenues que par une interaction entre les cellules. Il est cependant possible qu'un îlot puisse interagir avec elle-même.

Modèles hybrides :

Il est également possible de combiner plusieurs modèles. Par exemple, on peut imaginer un modèle d'îlot où chaque îlot exécute un algorithme évolutionnaire cellulaire pour promouvoir la diversité. Ou on peut penser à des modèles d‟îlot hiérarchiques où les îlots sont elles-mêmes des modèles d'îlot. Les modèles d'îlots et les modèles cellulaires peuvent également être implémentés en tant que modèles maître-esclave pourobtenir une meilleure accélération.

6. Métaheuristiques parallèles sur GPU :

La plupart des ordinateurs personnels, qu'ils soient fixes ou portables, comportent un GPU (Graphic Processing Unit). Ce processeur graphique a la particularité d'être un composant hautement parallèle avec une puissance de calcul très élevée. Pendant des années, l'utilisation de processeurs graphiques a été dédiée aux applications graphiques. Motivés par la demande de graphiques 3D haute définition sur les ordinateurs personnels, les GPU ont évolué vers un environnement hautement parallèle, multithread et multicœur. En effet, cette architecture fournit une puissance de calcul énorme et une bande passante mémoire très élevée par rapport aux processeurs traditionnels. Le potentiel énorme de ce composant est exploité dans les applications non graphiques, comme l'implémentation des métaheuristiques parallèles. Dans cette section, on met l'accent sur la description de calcul par les GPUs. Une compréhension claire des caractéristiques GPU est nécessaire pour fournir une implémentation efficace des métaheuristiques parallèles.

7. Conclusion :

Dans ce chapitre, nous présentons les systèmes parallèles. on commence par Les raisons de parallélisassions des métaheuristiques, les modèles parallèles des métaheuristiques, la classification des architectures parallèles et les déférents modèle parallèle suivie des concepts fondamentaux liés aux schémas d'exécution des algorithmes évolutionnaire. Enfin, nous parlons sur les métaheuristiques parallèles GPU.

Tout cala nous aide à trouver des solutions efficaces des problèmes de communication entre les processus qui vont exécuter notre algorithme proposé dans le dernier chapitre

Chapitre IV : L’implémentation de l’algorithme Algorithme Des feux d'artifice Discret Parallèle

1) **Introduction :**

Dans ce chapitre nous présentons l'expansion de requête, nous définirons l'expansion de requête et les déférentes approches et ses étapes, puis nous allons formuler l'expansion de requête comme un problème d'optimisation combinatoire et en fin proposer les MPI comme une solution.

2) **Expansion de requêtes :**

Définition :

L'expansion de la requête est la procédure qui modifie une requête par l'ajout de nouveaux termes afin d'améliorer l'efficacité de la recherche d'information **[17].**

3) **les approches de l'expansion de requête :**

Les approches d'expansion de requêtes classifiées selon les sources de données utilisées en trois catégories: approches statistiques, approche à partir de logs et approches linguistiques. **[19]**

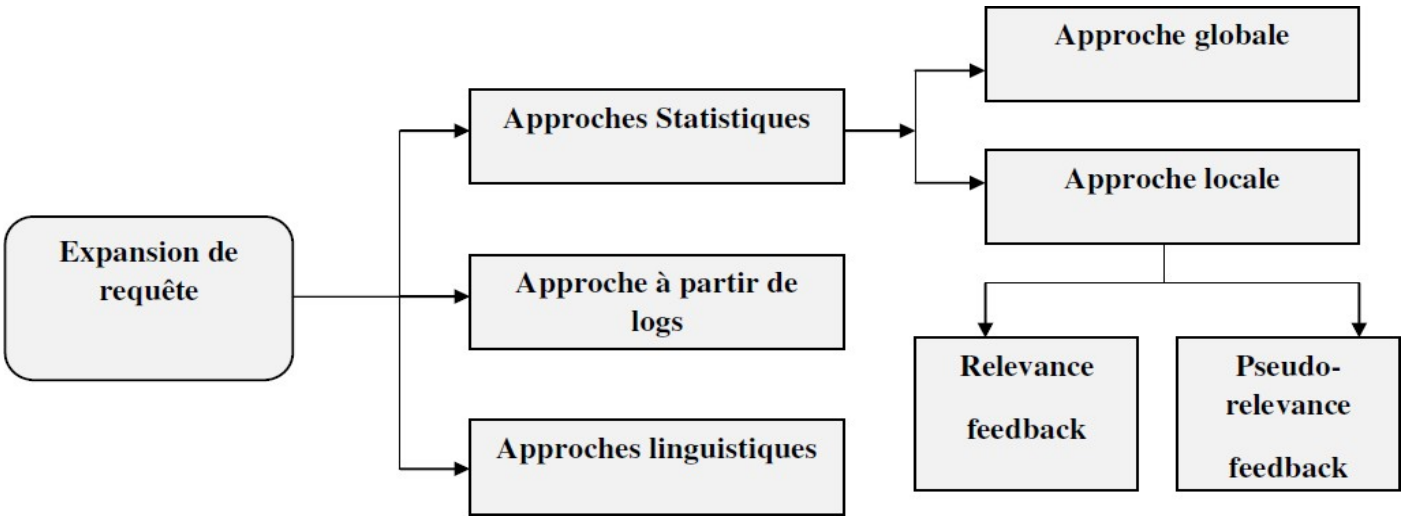

Figure 14 : Taxonomie des approches d'expansion de requêtes

3.1 Approches statistiques :

Cette approche est une relation entre les mots basée sur leurs cooccurrences, elle se décompose en deux catégories.

3.2 Approches locale :

L'approche locale inclut des techniques d'expansion de requête qui sélectionnent les termes d'expansion à partir de l'ensemble documents récupérés en réponse à la requête initiale. Il existe deux façons d'étendre la requête initiale de l'utilisateur en utilisant cette approche :

Relevance feedback et Pseudo-relevance feedback.

Relevance feedback :

L’utilisateur fait une première requête, et indique parmi les documents retournés lesquels sont les plus pertinents; à partir de ceux-ci des mots clés sont extraits afin de mener une deuxième requête "la requête étendue".

Pseudo- relevance feedback :

Le principe reste le même avec l’approche relevance feedback, mais la pertinence est basée sur le classement du moteur de recherche en utilisant directement les documents les mieux classés, récupérés en réponse à la requête initiale (d’où le terme "aveugle").

Approches globale :

Dans l’approche globale, les techniques d’expansion sélectionnent les termes d’expansion de requête à partir de l'ensemble des documents indexés pour reformuler et développer la requête initiale.

Approches à partir de logs :

Cette approche utilise plusieurs méthodes:

- Calcul de similarité entre la requête courante et les anciennes requêtes, puis approche statistique
- Association entre les requêtes et les documents retournés.
- Considérer les requêtes précédentes comme des documents, puis techniques statistiques de relevance feedback ou d'expansion aveugle

Approches linguistiques :

Les approches de cette catégorie analysent les caractéristiques d’expansion de requête à partir des ressources sémantiques (WordNet, ConceptNet) et lexico-syntaxiques (lemmes) afin de reformuler et développer la requête initiale. Pour se faire, Cette technique est elle-même devisée en sous-types : Word stemming, analyse sémantique et l’analyse syntaxique.

- Le Word stemming est l’une des premières approches les plus connues d’expansion de requête basée sur la linguistique. Cet algorithme peut être utilisé soit au moment de l’extraction, ou de l’indexation.

- L’analyse sémantique et contextuelle est une autre méthode d’expansion de requêtes liée à la sémantique. Il comprend des sources de données telles que les ontologies, le Cloud, les dictionnaires et les thésaurus.

- L’analyse syntaxique est la troisième approche qui fournit des informations linguistiques supplémentaires à la requête initiale. L'objectif est d'extraire des relations entre les termes de la requête,

qui peuvent ensuite être utilisés pour identifier les termes d'expansion qui apparaissent dans les relations associées**.**

4) **Les étapes de l’expansion de requête :**

Le processus de génération d’extension de requête comprend principalement quatre étapes illustrées dans la figure 15 qui sont : le prétraitement de la source de données, pondération et classement des termes, sélection des termes et reformulation des requêtes [19] [17]. Chaque étape est abordée dans les sections suivantes.

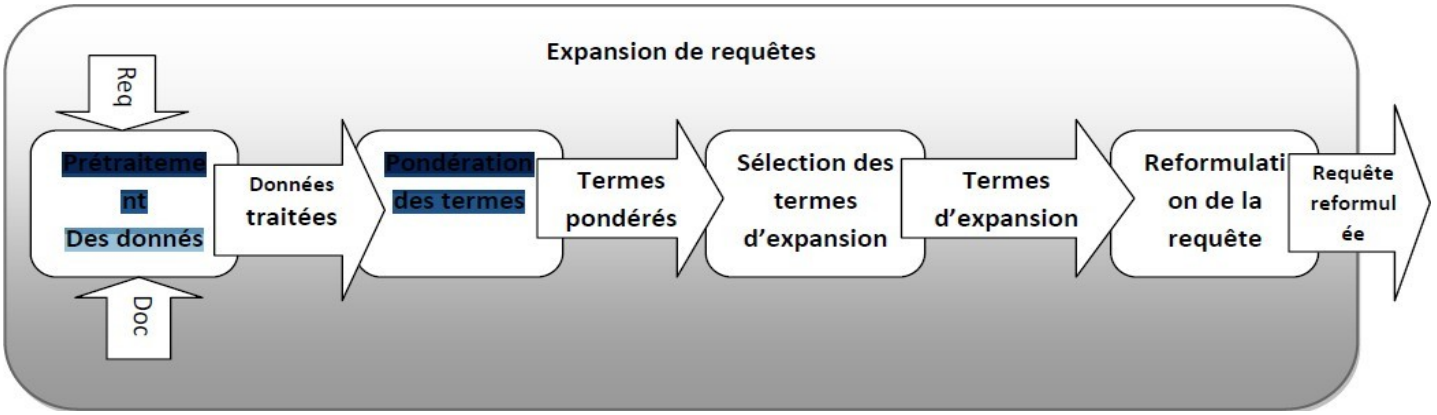

Figure 15:Les étapes de l’expansion automatique des requêtes.

5) **Prétraitement de la source de données :**

Cette étape dépend des sources de données et des approches utilisées pour l’expansion de la requête, au lieu de la requête de l'utilisateur. L'objectif principal est d’extraire un ensemble de termes provenant de sources de données qui augmentent de manière significative la requête initiale de l'utilisateur. Il se compose des cinq sous-étapes suivantes:

- Extraction de texte à partir de sources de données spécifiques utilisées pour l'expansion de la requête (des documents tels que HTML et PDF).
- Tokenization (extraction de mots individuels, en ignorant la ponctuation et la casse).
- suppression des mots fréquemment utilisés, par exemple : articles, adjectif, prépositions
- L’origine des mots (réduction des mots dérivés en leur forme racine. Un grand nombre de sources de données ont été utilisées pour l’expansion de requête dans la littérature

6) **Pondération et classement des termes d'extension de requêtes :**

Dans la deuxième étape de l’approche d’expansion de la requête, le système classe les termes d’extension par l’attribution des poids et des rangs, ce classement est très important car la plupart des

méthodes d’extension de requête ne choisiront qu'une petite proportion des termes d'expansion possibles à ajouter à la requête initiale, pour se faire , le système utilise la requête de l’utilisateur ainsi que les données traitées dans la première étape.

Les techniques utilisées pour réaliser la pondération et le classement des termes d’expansions générées et les termes sont classées en quatre catégories **[19]** :

7) **Distribution des fonctions des documents les mieux classés :**

Les techniques décrites dans cette section sont entièrement distincts des approches décrites dans les sections précédentes, car elles n'essaient pas de rechercher des caractéristiques directement associées aux termes de la requête, qu'elles soient simples ou multiples.

L'idée est d'utiliser les premiers documents récupérés en réponse à la requête initiale en tant que description plus détaillée du sujet de la requête, à partir duquel on peut extraire les termes les plus importants à utiliser comme termes d'expansion. En un sens, ces derniers sont liés à la signification complète de la requête car les termes extraits sont ceux qui caractérisent le mieux les documents pseudo-pertinents dans leur ensemble, mais leur association avec les termes de la requête n'est pas analysée explicitement.

De telles approches d’expansion de requêtes présentent de meilleurs résultats par rapport aux approches citées auparavant. Elles peuvent être subdivisées en deux catégories:

- Extension de la requête via le retour de pertinence (Relevance feedback). Les termes d’expansion de la requête sont extraits des documents récupérés en réponse à la requête initiale et l'utilisateur décide de la pertinence des résultats.

- Extension de la requête par Pseudo-relevance feedback. Les termes d'extension de la requête sont extraits des documents les mieux classés en réponse à la requête initiale.

8) **Sélection des termes d'extension**

Dans cette étape, seul un nombre limité de termes sont sélectionnés pour l’expansion, parce que la requête résultante peut être traitée plus rapidement, vu que l’efficacité de la recherche d’un petit ensemble de termes valables n’est pas nécessairement moins efficace que l’ajout de tous les termes [19]. Quelques travaux de recherche ont été menés déterminer le nombre optimal de termes d’extension à inclure dans la requête initiale. Toutefois, cet optimum suggéré peut varier de cinq à dix termes à quelques centaines de termes [16]. Par contre, d’autres travaux montrent que le nombre de termes utilisés pour l’expansion des requêtes est moins important que la qualité des termes choisis. Il a été généralement démontré que l’efficacité de l’extension de la requête diminue de manière minime avec le nombre non optimal de termes d’expansion [19].

La plupart des études expérimentales s'accordent pour dire que le nombre de termes d'expansion est peu pertinent et qu'il varie d'une requête à une autre. Il a été observé que l'efficacité du développement des requêtes (mesurée en tant que précision moyenne) diminue lorsque nous considérons moins de 20 termes de développement donc 20 à 40 termes constituent le meilleur choix pour l'extension de la requête. Lorsque les scores des caractéristiques peuvent être interprétés comme des probabilités, on ne peut sélectionner que les termes ayant une probabilité supérieure à un certain seuil, par exemple : p = 0,001 comme dans Zhai et Lafferty [19].

9) **L'expansion de requête comme un problème d'optimisation combinatoire :**

D'habitude l'utilisation de l'expansion de requête est de choisir les meilleurs mots disponibles à partir de la ressource utiliser pour l'extrait des mots. Mais a notre approcher on utilise le meilleur combinatoire de mots, c'est-à-dire on profite de la meilleure combinaison des *K* mots à partir d'un ensemble des *N* mots. Évidemment le nombre des cas est égal : $\frac{N!}{K!*(N-K)!}$

Mais ce nombre est énorme on l'additionnant à *N* ; pour cela nous avons utilisé l'algorithme des feux d'artifices pour ce problème ; sous sa forme parallèle à l'accélération de l'expansion de requête.

on utilise la bibliothèque **MPI** pour l'exécution d'un version parallèle de l'algorithme des feux d'artifice.

10) **Definition MPI (M essage P asing I nterface):**

MPI est une spécification pour les développeurs et les utilisateurs de bibliothèques de transmission de messages. En soi, ce n'est PAS une bibliothèque - mais plutôt la spécification de ce qu'une telle bibliothèque devrait être.

MPI *aborde* principalement *le modèle de programmation parallèle de transmission de messages* : les données sont déplacées de l'espace d'adressage d'un processus vers celui d'un autre processus via des opérations coopératives sur chaque processus.

En termes simples, l'objectif de l'interface de transmission de messages est de fournir une norme largement utilisée pour l'écriture de programmes de transmission de messages. L'interface tente d'être :

- Pratique
- Portable
- Efficace
- Souple

La norme MPI a subi un certain nombre de révisions, la version la plus récente étant MPI-3.x

Des spécifications d'interface ont été définies pour les liaisons de langage C et Fortran90 :

- Les liaisons C++ de MPI-1 sont supprimées dans MPI-3
- MPI-3 prend également en charge les fonctionnalités Fortran 2003 et 2008

Les implémentations réelles de la bibliothèque MPI diffèrent par la version et les fonctionnalités de la norme MPI qu'elles prennent en charge. Les développeurs/utilisateurs devront en être conscients.

11) **Raisons d'utiliser MPI**

- **Standardisation** - MPI est la seule bibliothèque de transmission de messages pouvant être considérée comme une norme. Il est pris en charge sur pratiquement toutes les plates-formes HPC. Pratiquement, il a remplacé toutes les bibliothèques de transmission de messages précédentes.
- **Portabilité** - Il est peu ou pas nécessaire de modifier votre code source lorsque vous portez votre application sur une plate-forme différente qui prend en charge (et est conforme à) la norme MPI.
- **Opportunités de performances** - Les implémentations des fournisseurs doivent pouvoir exploiter les fonctionnalités matérielles natives pour optimiser les performances. Toute implémentation est libre de développer des algorithmes optimisés.
- **Fonctionnalité** - Il existe plus de 430 routines définies dans MPI-3, qui incluent la majorité de celles dans MPI-2 et MPI-1.
- REMARQUE : La plupart des programmes MPI peuvent être écrits en utilisant une douzaine de routines ou moins
- **Disponibilité** - Une variété d'implémentations sont disponibles, à la fois fournisseur et domaine public.

12) **Conception de MPI pour le modèle de transmission des messages**

La base de la communication repose sur les opérations d'envoi et de réception entre les processus. Un processus peut envoyer un message à un autre processus en fournissant le rang du processus et une *étiquette* unique pour identifier le message. Le destinataire peut alors publier une réception pour un message avec une étiquette donnée (ou il peut même ne pas se soucier de l'étiquette), puis gérer les données en conséquence. De telles communications qui impliquent un expéditeur et un destinataire sont appelées communications *point à point*.

Il existe de nombreux cas où les processus peuvent avoir besoin de communiquer avec tout le monde. Par exemple, lorsqu'un processus gestionnaire doit diffuser des informations à tous ses processus de travail. Dans ce cas, il serait fastidieux d'écrire du code qui effectue tous les envois et les réceptions. En fait, il n'utiliserait souvent pas le réseau de manière optimale. MPI peut gérer une grande variété de ces types de communications *collectives* qui impliquent tous les processus.

Des mélanges de communications point à point et collectives peuvent être utilisés pour créer des programmes parallèles très complexes. En fait, cette fonctionnalité est si puissante qu'il n'est même pas nécessaire de commencer à décrire les mécanismes avancés de MPI.

Nous avons tous les meilleurs résultats de chaque processus, on choisit les bons résultats (les meilleurs étincelles) .la prochaine exécution commencera a partir les bons résultats obtenus. Si on essaye plusieurs fois avec les meilleurs résultats et on n'arrive pas à nouveau bon résultats on le considère comme solution optimale.

13) **Approches proposé** :

Dans notre mémoire nous avons appliqué la notion du parallélisme sur l'algorithme de feux d'artifice sur l'expansion de requête pour chercher une meilleur combinaison des mots, pour évitera les problèmes de communications entre les processus nous proposons la notion de MPI pour l'envoi des messages.

14) **. L'algorithme Proposé :**

1. **initialiser n feux d'artifice**
2. **évalué n feux d'artifice**
3. **tan que condition d'arrête n'est pas Vérifier faire :**

a) **Calculer l'amplitude et le numéro des étincelles pour chaque feu d'artifice**

4. **Générer les étincelles explosées pour tous les feux d'artifices**
5. **Evaluer tout les étincelles**
6. **Send (next générations par tout processus)**
7. **Recever (les générations envoyer par tout processus)**
8. **Evaluer touts les étincelles**
9. **Construire les générations suivantes**

Fin tan que.

On commence cet algorithme par une initialisation aléatoire de n feux d'artifices, puis l'évaluation des n feux d'artifices, si la condition d'exécution de l'algorithme est réalisé. On Calcule l'amplitude et le nombre des étincelles pour chaque feu d'artifice. On générera et on évaluera tout les étincelles pour choisir les meilleurs solutions (les meilleurs étincelles) puis on envoie les meilleurs étincèles aux processus et on reçoit les meilleurs étincelles des autres paresseuses, cette opération d'envoi et de réception s'organise par la bibliothèque MPI.

(combinaison des flux) :

Conclusion générale :

Dans cette mémoire nous allons commencer cette étude par la définition des problèmes combinatoire de façon générales et les déférentes méthodes de résolution des problèmes d'optimisation combinatoire,

En Deuxième partie nous définit la conception de l'algorithme de feux d'artifice et on dit que l'algorithme des feux d'artifice(FireworksAlgorithm)est un nouvel algorithme récemment développé de nature de l'intelligence en essaim. Il prend a base théorique parla simulation du processus d'explosion des feux d'artifice. En analogie avec de véritables feux d'artifice explosant et illuminant le ciel nocturne, les feux d'artifice (c'est-à-dire les individus) dans cet algorithme sont laissés à l'espace de recherche potentiel. Pour chaque feu d'artifice, un processus d'explosion est initié et une pluie d'étincelles remplit l'espacelocalquil'entoure.Lesfeuxd'artificeainsiquelesnouvellesétincellesgénéréesreprésententdes solutions potentielles dans l'espacederecherche.et on étudier la notion du DFAW l'algorithme d feux d'artifice discret.

Dans troisième partie on définit la parallélisassions des métaheuristiques et la modélisation parallèle des métaheuristiques et Les schémas d'exécution des algorithmes évolutionnaire à base parallèle.

Finalement pour l'implémentation nous avons proposé les MPI pour évitera le problème de communication entre les processus (l'envoi des messages entre les processus) et on implémenté l'algorithme de feux d'artifice parallèle à l'utilisation des MPI, et programmer avec le langage python et nous avons choisissons PyCharmCommunity Edition 2020.3.1 x64 comme un environnement de développement.

Bibliographie

[1] Abdesslem LAYEB, Utilisation des Approches d'Optimisation Combinatoire pour la Vérification des Applications Temps Réel. Thèse de Doctorat, Université Mentouri de Constantine 2010

[2] Mostepha, R : Résolution de problèmes d'optimisation combinatoire par systèmes artificiles auto-organisés. Thèse de magister, Université Mentouri de Constantine ,2008.

[3] Palpant M. : Recherche exacte et approchée en optimisation combinatoire : schémas d‟intégration et applications. Thèse de Doctorat, Université d‟Avignon, 2005.

[4] Reeves, C.: Modern Heuristic Techniques for Combinatorial Problems. Advances topics in computer science. Mc Graw-Hill, 1995.

[5] Omessaad, H : Contribution au développement de méthodes d'optimisation Stochastiques application à la conception des dispositifs électrotechniques, Thèse de Doctorat, Université De Lille France 2003.

[6] Lambert veller sylvain , lechevalier david,quirico tommy «Problème de ramassage dans une ville virtuelle - Algorithme Tabu Search »Université de Bourgogne 2010-2011

[7] Aissa Boulmerka, « Adaptation des métaheuristiques à l'ordonnancement hors-ligne d tâches temps réel à contraintes strictes en environnement monoprocesseur », Mémoire de magister en informatique, 20 Avril 2009.

[8] Alain Hertz, 'L'optimisation Combinatoire', École Polytechnique, Canada, 2006

[9] Rachid Chelouah, 'L'optimisation combinatoire', INA Institut d'informatique appliquée, Suisse, 2003.

[10] http://khayyam.developpez.com/articles/algo/genetic/

[11] Souquet Amédée « Algorithme génétique » *Radet Francois-Gérard*

[12] Y. Tan and Y. Zhu, "Fireworks Algorithm for Optimization," in *Advances in SwarmIntelligence*, vol. 6145, Y. Tan, Y. Shi, and K. C. Tan, Eds. Berlin, Heidelberg:

Springer Berlin Heidelberg, 2010, pp. 355–364.

[13] E.-G. Talbi. Metaheuristics: from design to implementation, volume 74. John Wiley & Sons, 2009.

[14] Leopold, C. Parallel and Distributed Computing: A Survey of Models, Paradigms and Approaches, Wiley-Interscience, 2001.

[15] [94]

[16] Sudholt, D. .Parallel evolutionary algorithms. In Springer Handbook of Computational Intelligence. Springer Berlin Heidelberg. pp. 929-959, 2015.

[17] : B Audeh. « Reformulation Sémantique des requêtes par la recherche d'information adhoc sur le web ». Thèse de doctorat à l'Ecole nationale supérieur des Mines de Saint-Etienne.6 Mars 2015.

[18] Bouraoui J, Guimier de Neef E, Gaillard B, Boualem M, Collin O. « Expansion sémantique de requêtes ». Cours à l'Université Catholique de LouvainMercredi 31 mars 2010.

[19] Hiteshwar K A, Akshay De. «Query expansion techniques for Information Retrieval: Asurvey ».Article (Information Processing & Management). 01 Août 2015.