الجمهوريـة الجزائـريـة الديمقراطيـة الشعبيـة
République Algérienne Démocratique et Populaire
وزارة التعليـم العالـي والبحـث العلمـي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N° Réf :……

Centre Universitaire
Abd elhafid boussouf Mila

Institut des sciences et de la technologie **Département de Mathématiques et Informatique**

Mémoire préparé En vue de l'obtention du diplôme de Master

En :Filière informatique

Spécialité sciences et technologies de l'information et de la communication stic

Apprentissage Automatique et Optimisation pour la Résolution de Problèmes

Préparé par :

Zaimeche Mourad

Soutenue devant le jury

Président :	Mme Zekiouk Mounira	**Grade : MAA**
Examinateur :	Mlle Bouchekouf Asma.	**Grade : MAB**
Promoteur :	Mme Afri Faiza.	**Grade : MAB**

Année universitaire :2014/2015

الجمهوريـة الجزائـريـة الديمقراطيـة الشعبيـة
République Algérienne Démocratique et Populaire
وزارة التعليـم العالـي والبحـث العلمـي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N° Réf :……

Centre Universitaire
Abd elhafid boussouf Mila

Institut des sciences et de la technologie **Département de Mathématiques et Informatique**

Mémoire préparé En vue de l'obtention du diplôme de Master

En :Filière informatique

Spécialité sciences et technologies de l'information et de la communication stic

Apprentissage Automatique et Optimisation pour la Résolution de Problèmes : Découverte de Motifs par algorithme hybride (GA-SVM)

Préparé par :

Zaimeche Mourad

Soutenue devant le jury

Président :	Mme Zekiouk Mounira	**Grade : MAA**
Examinateur :	Mlle Bouchekouf Asma.	**Grade : MAB**
Promoteur :	Mme Afri Faiza.	**Grade : MAB**

Année universitaire :2014/2015

Remerciement

Je Remercie En tout premier lieu ALLAH le tout puissant, qui m'a donné la force, la volonté et le courage pour accomplir ce modeste travail.

je tiens ici à remercier Mme: Afri Faiza , mon encadreur pour son aide et sa grande patience qu'elle a apporté tout au long la préparation de ce mémoire, ses conseils, ses orientations et encouragements qui ont contribué notablement à la réussite du travail à ce niveau, que dieu la protège.

Mes vifs remerciements sont également aux membres du jury pour l'intérêt qui 'ils ont porté à mon travail en acceptant d'examiner mon travail et de l'enrichir par leurs propositions et commentaires.

Je n'oublie pas mes chers enseignants qui M'ont encadré et guidé tout au long du cycle d'étude au centre universitaire de Mila, et d'avoir transmettre leur savoir.

Je tiens enfin à remercier tous ceux qui ont collaborés de prés ou de loin à l'élaboration de ce

travail. Qu'ils acceptent mes humbles remerciements.

Dédicace

Je dédie ce modeste travail à :
A mes très chers parents qui m'ont tant soutenu et encouragé dans tous les domaines et surtout pour réaliser ce
mémoire :
mon père ABDERRACHID qui ma donné tout ces pouvoir dans
mon parcours d'étude, il ma donné le courage, la volonté, la
confiance en moi-même, que dieu le protège.
ma bien-aimée chère mère ZEBIDA la plus précieuse personne pour
moi, la meilleure femme dans le monde, que dieu la protège.
A mes frères Abdelouahab, et Oussama
A mes soeurs Asmaa et Iness
À Toute ma famille
À Tous mes amis
et mes collègues du centre universitaire, surtout de la filière
Informatique.
A tous mes enseignants.
À Tous ceux qui me sont chers.
A toute Personne qui me connaît.

MOUARD

بسم الله الرحمن الرحيم

" *الْحَمْدُ لِلَّهِ الَّذِي هَدَانَا لِهَذَا وَمَا كُنَّا لِنَهْتَدِيَ* لَوْلا أَنْ هَدَانَا اللَّهُ "..

الحمد لله أولا و آخرا على فضله ونعمه ، أنعم علي لانجاز هذا العمل المتواضع وإن أصبت فيه فمن الله وان أخطأت فمن نفسي ومن الشيطان

مراد

Résumé

De nos jours, nous constatons une prolifération des approches hybrides qui puisent, des métaphores capables d'apporter des solutions innovantes aux problèmes complexes. Notre objectif est l'élaboration d'une méthode de résolution qui sera à la fois efficace, plus, flexible et aussi robuste face aux imprévus.

L'hybridation consiste à exploiter les avantages respectifs de plusieurs méthodes en combinant leurs algorithmes suivant une approche synergétique, ce qui permet d'avoir un équilibre entre l'exploration et l'exploitation de l'espace de recherche et surmonter ainsi le problème de la convergence prématurée. Nous avons utilisé l'hybridation de méthodes à différents niveaux.

Dans cette thèse, nous proposons une hybridation entre deux familles d'algorithmes : les algorithmes génétique (optimisation) et machine à vecteur de support (apprentissage automatique) pour la résolution d'un problème lié à la bioinformatique à savoir : la découverte de motifs dans des séquences biologiques. Notre objectif principal est de montrer l'utilité de l'hybridation en parvenant à améliorer les performances : fiabilité, robustesse et accélération de la convergence.

Mots-clés : Hybridation, optimisation, apprentissage automatique, algorithme génétique, machine à vecteur de support, découvert des motifs, bioinformatique.

.Abstract

Today we are seeing a proliferation of hybrid approaches that are able to provide innovative solutions to complex problems. Our goal is to develop a method of resolution that will be effective, simple, more flexible and robust.

Hybridization consists of exploiting the advantages of several methods by combining their algorithms in a synergistic approach, which allows a balance between exploration and exploitation of space research and thus overcome the problem of premature convergence. We used the hybridization methods at different levels.

In this thesis, we propose the hybridization between two methods: genetic algorithms (optimization algorithm) and Support Vector Machine (machine learning) for the problem

pattern discovery in biological sequences in bioinformatics field. Our main objective is to show the usefulness of hybridization achieving better performance: reliability, robustness and faster convergence.
Keywords: hybridization, optimization, machine learning, genetic algorithm, Support Vector Machine, pattern discovery, bioinformatics.

ملخص

تعرف اليوم طرق التهجين انتشارا واسعا، وذلك لقدرتها على تقديم حلول مبتكرة للمشاكل المعقدة. هدفنا في هذا البحث هو تطوير طريقة للحل تكون فعالة (الحصول على الحلول المثلى في زمن معقول) ، بسيطة (سهلة الفهم و التنفيذ) ، مرنة (سهلة التعميم إلى مشاكل أكثر تعقيدا) ومتينة.

لذلك نستخدم طرق التهجين على مختلف المستويات. التهجين يتمثل أساسا في استغلال مزايا عدة طرق للحساب ، بالجمع بين خوارزمياتها وفقا لمقاربة تعاونية مما يسمح بتحقيق توازن أكبر بين استكشاف واستغلال فضاء البحث والتغلب بالتالي على مشكلة التقارب السابق لأوانه. في هذه الأطروحة، نقترح طريقة تهجين الخوارزمية الجينية مع آلة الأشعة الحاملة لاستعمالها في التعرف على الأنماط الموجودة في السلسلة البيولوجية .

كلمات مفتاحيه : التهجين ، الخوارزمية الجينية ، آلة الأشعة الحاملة ، السلسلة الجينية

Table des matières

Liste des Figures :

Liste des tables

Liste des équations

Introduction Générale

Contexte de l’étude :

La bioinformatique est l'étude de l'information biologique. Ce n'est pas simplement l'application à la biologie de l'informatique ; c'est une branche à part entière de la biologie. La bioinformatique actuelle se concentre surtout sur l'étude des séquences d'ADN et sur le repliement des protéines, donc travaille surtout au niveau moléculaire. De nombreux bioinformaticiens travaillent également à l'élaboration d'outils biologiques permettant de résoudre des problèmes de l'informatique classique.

En bioinformatique, la découverte de motifs devient très importante parce qu'ils représentent des séquences conservées qui peuvent être biologiquement significative. Il pourrait être essentiel pour l'analyse et la compréhension des données biologiques.
La découverte de motifs dans l'ADN, l'ARN et les séquences de protéines a conduit à la solution de nombreux problèmes biologiques vitaux.

Le développement des techniques d’hybridation a permis de résoudre beaucoup de problèmes NP-complets, et l'augmentation de la puissance des appareils informatiques a autorisé la mise en œuvre de ces techniques et leur utilisation. La bioinformatique permet d'analyser des quantités de données très importantes, ce que l'analyse "à la main" ne permet pas.

Le problème de découverte de motifs dans des séquences biologiques est l’un des problèmes les plus connus. Découverte de motifs en séquences biologique peut conduire à la détermination de la fonction et de l'élucidation des relations évolutives entre les séquences.

Problématique et motivation :

La quantité de données biologiques générées par la communauté scientifique a connu une croissance exponentielle, et un grand nombre de séquences d'ADN sont maintenant disponibles dans plusieurs bases de données. Ainsi, l'objet de nombreux projets de recherche est déplacé à partir de la génération des données pour leur analyse. Trouver des motifs consiste à déterminer des courtes séquences significatives qui peuvent être répétées sur plusieurs séquences de différentes espèces.

Dans ce mémoire, nous nous concentrons sur le problème de découverte de motifs dans les séquences biologiques, comme cela est peut-être l'un des domaines d'application les plus importants ayant des implications à la biologie moléculaire et de la médecine.

Pour y arriver, nous avons proposé un algorithme hybride GA-SVM basé sur les Algorithmes Génétiques et l’algorithme Machines à Vecteurs Supports.

Structure du mémoire

Le présent mémoire est divisé en trois parties. La première, appelée état de l'art, cette partie fournit des connaissances préliminaires et nécessaire pour une bonne compréhension de ce travail. La seconde partie, nommée Combinaison des algorithmes génétiques (l'optimisation) et les Machines à Vecteurs Supports (l'apprentissage automatique), Cette partie présente les concepts de base de l'algorithme utilisé dans ce travail, la troisième, appelée étude de cas, Présente les résultats finaux obtenus après l'exécution du programme.

La première partie est composée de deux chapitres. Le premier aborde le problème de l'optimisation et l'apprentissage automatique. Le Deuxième présente les notions de base de la biologie moléculaire et introduit les concepts de la bioinformatique, ensuite le problème de découverte de motifs est expliqué.

La seconde partie est composée de deux chapitres, Le premier pour définir les méthodes hybrides et expliquer les différents travaux d'hybridation qui ont vu le jour. Le Deuxième pour expliquer les algorithmes génétiques et les machines à vecteur de support, aussi expliqué la combinaison entre GA et SVM est détaillée.

Quant à la troisième partie présente l'étude de cas. En premier lieu, les outils utilisés pour l'implémentation de l'algorithme proposé sont évoqués, ensuite quelques résultats sont présentés comme exemples explicatifs. Nous terminons par une conclusion générale et quelques perspectives.

Chapitre I

Apprentissage Automatique et Optimisation

I. L'apprentissage automatique

1. Introduction

L'**apprentissage automatique (ou artificiel)** (*machine-learning* en anglais) est un des champs d'étude de l'intelligence artificielle. Commençons par la définition de l'AAAI[1] et celle fournie dans l'avant-propos de (Cornuéjols *et al.*,2002) [1].

L'apprentissage artificiel fait référence à la capacité d'un système à acquérir et intégrer de façon autonome des connaissances[2].

Cette notion englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle [1].

L'apprentissage automatique fait référence au développement, l'analyse et l'implémentation de méthodes qui permettent à une machine (au sens large) d'évoluer et de remplir des tâches associées à une intelligence artificielle grâce à un processus d'apprentissage. Cet apprentissage permet d'avoir un système qui s'optimise en fonction de l'environnement, les expériences et les résultats observés [1].

Voyons quelques exemples. La capacité d'apprentissage est une caractéristique des êtres vivants. De la naissance à l'âge adulte, les êtres vivants acquièrent de nombreuses capacités qui leur permettent de survivre dans leur environnement. L'apprentissage d'un langage, de l'écriture et de la lecture sont de bons exemples des capacités humaines, et des phénomènes mis en jeu : apprentissage par cœur, apprentissage supervisé par d'autres êtres humains, apprentissage par généralisation.

Une application classique de l'apprentissage artificiel est la reconnaissance de caractères manuscrits, tels qu'ils apparaissent sur une enveloppe. La difficulté tient au fait que la variété des formes rencontrée est infinie. L'apprentissage par cœur n'est pas possible, et il faut donc être capable de généraliser à partir d'un ensemble d'exemples de caractères.

[1] American Association for Artificial Intelligence, http://www.aaai.org.

[2] Machine learning refers to a system capable of the autonomous acquisition and integration of knowledge.

2. Quelques mots d'Histoire

L'apprentissage artificiel est une discipline jeune, à l'instar de l'Informatique et de l'Intelligence artificielle. Il se situe au carrefour d'autres disciplines: philosophie, psychologie, biologie, logique, mathématique [1].les premières études remontent à des travaux de statistique dans les années 1920. C'est après la seconde guerre mondiale que les premières expériences deviennent possibles. Se développent ensuite dans les années 1960 les approches connexionnistes avec des perceptrons, et la reconnaissance des formes. La mise en évidence des limites du perceptron simple arrête toutes les recherches dans ce domaine jusqu'à la renaissance dans les années 1980. Les années 1970 sont dominées par des systèmes mettant l'accent sur les connaissances, les systèmes experts, Les limites de tels systèmes se font sentir dans les années 1980, pendant lesquelles a lieu le retour du connexionnisme avec un nouvel algorithme d'apprentissage [1].

Les mathématiciens commencèrent à s'éloigner du cadre cognitif de l'apprentissage pour envisager le problème sous l'angle de l'optimisation, pendant qu'apparaissaient de nouvelles méthodes comme les arbres de décision ou l'induction de programmes logiques. L'influence de la théorie statistique de l'apprentissage s'est réellement fait sentir dans les années 1990, avec l'ouvrage de Vapnik [2].

3. Concepts et Sources de l'apprentissage automatique

L'apprentissage de l'être humain se compose de plusieurs processus qu'il est difficile précisément à décrire. Les facultés d'apprentissage chez l'humain lui ont conféré un avantage évolutif déterminant pour son développement [1].

Par " faculté d'apprendre " on entend un ensemble d'aptitudes comme :

- L'obtention de la capacité de parler en observant les autres.
- L'obtention de la capacité de lire, d'écrire, d'effectuer des opérations arithmétiques et logiques avec l'aide d'un tuteur.
- L'obtention d'habilités motrices et sportives en s'exerçant.

4. Qu'est-ce que l'apprentissage automatique

La faculté d'apprendre de ses expériences passées et de s'adapter est une caractéristique essentielle des formes de vies supérieures. Elle est essentielle à l'être humain dans les premières étapes de la vie pour apprendre des choses aussi fondamentales que

reconnaître une voix, un visage familier, apprendre à comprendre ce qui est dit, à marcher et à parler [3].

L'apprentissage automatique est une tentative de comprendre et reproduire cette faculté d'apprentissage dans des systèmes artificiels. Il s'agit, très schématiquement, de concevoir des algorithmes capables, à partir d'un nombre important d'exemples (les *données* correspondant à « l'expérience passée »), d'en assimiler la nature afin de pouvoir appliquer ce qu'ils ont ainsi appris aux cas futurs [3].

- **Définition**

Un programme d'ordinateur est capable d'apprendre à partir d'une expérience E et par rapport à un ensemble T de tâches et selon une mesure de performance P, si sa performance à effectuer une tâche de T, mesurée par P, s'améliore avec l'expérience E[1].

- **Modélisation**

L'apprentissage automatique d'une machine toujours concerne un ensemble de tâches concrètes - T. Pour déterminer la performance de la machine, on utilise une mesure de la performance P. La machine peut avoir à l'avance un ensemble d'expérience E ou elle va enrichir cet ensemble plus tard.

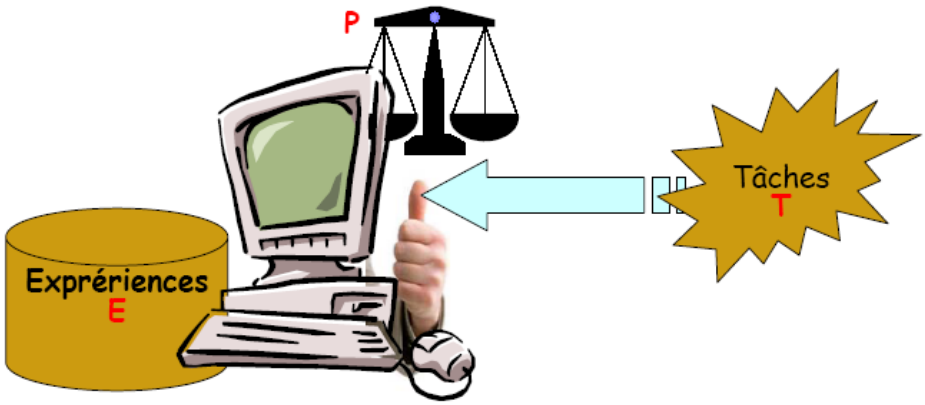


Figure 1: Schéma de modélisation d'une machine d'apprentissage

Donc, l'apprentissage automatique pour la machine est qu'avec l'ensemble de tâches T que la machine doit réaliser, elle utilise l'ensemble d'expériences E telle que sa performance sur T est améliorée.

- **Domaines d'applications de l'apprentissage automatique:**

L'apprentissage automatique s'applique à un grand nombre d'activités humaines et convient en particulier au problème de la prise de décision automatisée. Il s'agira, par exemple:

- D'établir un diagnostic médical à partir de la description clinique d'un patient;
- De donner une réponse à la demande de prêt bancaire de la part d'un client sur la base de sa situation personnelle;
- De déclencher un processus d'alerte en fonction de signaux reçus par des capteurs ;
- De la reconnaissance des formes;
- De la reconnaissance de la parole et du texte écrit;
- De contrôler un processus et de diagnostiquer des pannes;

- **Mémoriser n'est pas généraliser**

Le terme **apprentissage** dans la langue courante est ambigu. Il désigne aussi bien l'apprentissage "par cœur" d'une poésie, que l'apprentissage d'une tâche complexe telle que la lecture. Clarifions la distinction [3] :

– Le premier type d'apprentissage correspond à une simple mémorisation. Ou les ordinateurs contemporains, avec leurs mémoires de masse colossales, n'ont aucune difficulté à mémoriser une encyclopédie entière, sons et images inclus.

– Le second type d'apprentissage se distingue fondamentalement du premier en cela qu'il fait largement appel à notre faculté de généraliser. Ainsi pour apprendre à lire, on doit être capable d'identifier un mot écrit d'une manière que l'on n'a encore jamais vue auparavant.

5. Types d'apprentissage

Les algorithmes d'apprentissage peuvent se catégoriser selon le mode d'apprentissage qu'ils emploient :

- **L'apprentissage supervisé**

Si les classes sont prédéterminées et les exemples connus, le système apprend à classer selon un modèle de classement ; on parle alors d'apprentissage supervisé (ou d'analyse discriminante).

Un expert (ou oracle) doit préalablement correctement étiqueter des exemples. L'apprenant peut alors trouver ou approximer la fonction qui permet d'affecter la bonne « étiquette » à ces exemples. Parfois il est préférable d'associer une donnée non pas à une classe unique, mais une probabilité d'appartenance à chacune des classes prédéterminées (on parle alors d'apprentissage supervisé probabiliste). L'analyse discriminante linéaire ou les SVM sont des exemples typiques. Autre exemple : en fonction de *points communs* détectés avec les symptômes d'autres patients connus (les « exemples »), le système peut catégoriser de nouveaux patients au vu de leurs analyses médicales en risque estimé (probabilité) de développer telle ou telle maladie.

- **L'apprentissage non-supervisé**

Quand le système ou l'opérateur ne dispose que d'exemples, mais non d'étiquettes, et que le nombre de classes et leur nature n'ont pas été prédéterminés, on parle d'apprentissage non supervisé (ou clustering).

Aucun expert n'est disponible ni requis. L'algorithme doit découvrir par lui-même la structure plus ou moins cachée des données. Le système doit ici dans l'espace de description (la somme des données) cibler les données selon leurs attributs disponibles, pour les classer en groupe homogènes d'exemples. La similarité est généralement calculée selon la fonction de distance entre paires d'exemples. C'est ensuite à l'opérateur d'associer ou déduire du sens pour chaque groupe. Divers outils mathématiques et logiciels peuvent l'aider. On parle aussi d'analyse des données en régression. Si l'approche est probabiliste (c'est à dire que chaque exemple au lieu d'être classé dans une seule classe est associé aux probabilités d'appartenir à chacune des classes), on parle alors de *« soft clustering »* (par opposition au *« hard clustering* ») [1].

Exemple : Un épidémiologiste pourrait par exemple dans un ensemble assez large de victimes de cancers du foie tenter de faire émerger des hypothèses explicatives, l'ordinateur pourrait différencier différents groupes, qu'on pourrait ensuite associer par exemple à leur

provenance géographique, génétique, à l'alcoolisme ou à l'exposition à un métal lourd ou à une toxine telle que l'aflatoxine.

- **L'apprentissage semi-supervisé**

Effectué de manière probabiliste ou non, il vise à faire apparaitre la distribution sous-jacente des *« exemples »* dans leur espace de description. Il est mis en œuvre quand des données (ou *« étiquettes »*) manquent... Le modèle doit utiliser des exemples *non-étiquetés* pouvant néanmoins renseigner.

Exemple : En médecine, il peut constituer une aide au diagnostic ou au choix des moyens les moins onéreux de tests de diagnostics.

- **L'apprentissage par renforcement**

L'algorithme apprend un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit une valeur de retour qui guide l'algorithme.

6. Les Algorithmes utilisés

➢ Les algorithmes de L'apprentissage supervisé

- **Les machines à vecteurs support (SVM) :**

Cette technique initiée par Vapnik tente de séparer linéairement les exemples positifs des exemples négatifs dans l'ensemble des exemples. Chaque exemple doit être représenté par un vecteur de dimension n.

La méthode cherche alors l'hyperplan qui sépare les exemples positifs des exemples négatifs, en garantissant que la marge entre le plus proche des positifs et des négatifs soit maximale.

Intuitivement, cela garantit un bon niveau de généralisation car de nouveaux exemples pourront ne pas être trop similaires à ceux utilisés pour trouver l'hyperplan mais être tout de même situés franchement d'un côté ou l'autre de la frontière.

L'intérêt des SVM est la sélection de Vecteurs Supports qui représentent les vecteurs discriminant grâce auxquels est déterminé l'hyperplan. Les exemples utilisés lors de la recherche de l'hyperplan ne sont alors plus utiles et seuls ces vecteurs supports sont utilisés pour classer un nouveau cas. [4]

- **Naïve Bayes :**

Il s'agit cette fois d'un modèle représentant des variables aléatoires, fondé sur le théorème des probabilités conditionnelles de Bayes. Adapté à notre problématique, cet algorithme permet de calculer la probabilité pour une instance d'appartenir à la classe des positifs, et celle d'appartenir à celle des négatifs, en connaissant ses attributs.

Chacune de ces probabilités est calculée par approximation naïve, et la plus élevée des deux déterminera la classe de l'instance considérée. Nous intégrons ce nouvel algorithme à nos expérimentations pour tester son efficacité sur nos données, de la même manière que dans les travaux de (Kobdani and Schütze, 2010) [5].

- **Réseaux de neurones :**

Les réseaux de neurones sont utilisés pour leur capacité à apprendre à partir d'exemples bruités comme les caméras ou les micros (reconnaissance de forme ou de son). Mais ils sont aussi utilisables pour des problèmes où les méthodes symboliques (arbres de décisions) sont souvent utilisées. Leur performance est alors équivalente.

Les réseaux de neurones sont appropriés lorsque le temps d'apprentissage n'est pas essentiel : ce temps est en effet souvent très supérieur à d'autres méthodes comme les arbres de décision. Par contre, la classification d'un nouveau cas (par exemple un document) est très rapide.

Enfin, les réseaux de neurones sont appropriés si la compréhension de la fonction apprise par le réseau n'est pas essentielle. Avec un arbre de décision, l'opérateur humain peut toujours visualiser l'arbre et « comprendre » comment la machine décide. Avec un réseau de neurone, des techniques de visualisation existent, mais elles demandent généralement plus d'expertise que l'analyse d'un arbre de décision (qui peut être visualisé sous forme de règles). [6]

- **Les arbres de décision :**

Cet algorithme apprend à partir de données d'apprentissage un « arbre » dont les nœuds représentent des critères, portant chacun sur un unique attribut de l'ensemble, et où chaque feuille correspond à une classe. L'initialisation de l'algorithme se fait sur un arbre vide (dont l'unique nœud est la racine), puis une itération débute, qui vérifie si le nœud courant est une feuille ou non : si c'est le cas, une classe lui est associée ; sinon, le programme sélectionne un test et crée un sous-arbre dont les branches sont les différentes réponses de ce test. L'algorithme est répété sur le même schéma, et ce jusqu'à ce que toutes les feuilles soient des classes. La sélection du nœud racine se fait par comparaison des mesures d'entropie des classes pour chacun des attributs : on recherche celui dont l'entropie est la plus faible pour débuter l'arbre (i.e. celui qui répartit le mieux les classes). [7]

- **La méthode des k plus proches voisins pour un apprentissage supervisé :**

Plus connus en anglais sous le nom K-nearest neighbor (K-NN). Cette méthode diffère des traditionnelles méthodes d'apprentissage car aucun modèle n'est induit à partir des exemples. Les données restent telles quelles : elles sont simplement stockées en mémoire [6].

L'algorithme des k plus proches voisins sert dans plusieurs problèmes informatiques incluant la reconnaissance des formes, la recherche dans les données multimédia, la compression vectorielle, les statistiques informatiques et l'extraction des données.

Pour prédire la classe d'un nouveau cas, l'algorithme cherche les K plus proches voisins de ce nouveau cas et prédit (s'il faut choisir) la réponse la plus fréquente de ces K plus proches voisins [6].

La méthode utilise donc deux paramètres : le nombre K et la fonction de similarité pour comparer le nouveau cas aux cas déjà classés.

- **L'analyse discriminante linéaire :**

L'analyse discriminante linéaire fait partie des techniques d'analyse discriminante prédictive. Il s'agit d'expliquer et de prédire l'appartenance d'un individu à une classe

(groupe) prédéfinie à partir de ses caractéristiques mesurées à l'aide de variables prédictives.

L'analyse discriminante linéaire peut être comparée aux méthodes supervisées développées en apprentissage automatique et à la régression logistique développée en statistique.

- **Le boosting :**

Le boosting est un domaine de l'apprentissage automatique (branche de l'intelligence artificielle). C'est un principe qui regroupe de nombreux algorithmes qui s'appuient sur des ensembles de classifieurs binaires : le boosting optimise leurs performances. Le principe est issu de la combinaison de classifieurs (appelés également hypothèses). Par itérations successives, la connaissance d'un classifieur faible - *weak classifier* - est ajoutée au classifieur final - *strong classifier*. On appelle **apprenant faible** un algorithme qui fournit des classifieurs faibles, capables de reconnaitre 2 classes au moins aussi bien que le hasard ne le ferait (c'est-à-dire qu'il ne se trompe pas plus d'une fois sur deux en moyenne). Le classifieur fourni est pondéré par la qualité de sa classification : plus il classe bien, plus il sera important. Les exemples mal classés sont *boostés* pour qu'ils aient davantage d'importance vis à vis de l'apprenant faible au prochain tour, afin qu'il pallie le manque. Un des algorithmes les plus utilisés en **boosting** s'appelle AdaBoost, qui signifie *adaptative boosting*. Le boosting est basé sur le Cadre PAC (en anglais,Probably Approximately Correct).

- **La régression logistique :**

La régression logistique ou modèle logit est un modèle de régression binomiale. Comme pour tous les modèles de régression binomiale, il s'agit de modéliser l'effet d'un vecteur de variables aléatoires $(x_1, \ldots, x_K)$sur une variable aléatoire binomiale génériquement notée y. La régression logistique est un cas particulier du modèle linéaire généralisé.

- **Les algorithmes de L'apprentissage non supervisé**

Parmi les algorithmes non supervisées les plus connues, on peut citer par exemple [6]:

- **La logique floue :**

La logique floue s'affirme comme une technique opérationnelle. Utilisée à côté d'autres techniques de contrôle avancé, elle fait une entrée discrète mais appréciée dans les automatismes de contrôle industriel.

Les bases théoriques de la logique floue (fuzzy logic , en anglais) ont été établies au début des années 1965 par le professeur Zadeh de l'université de Californie de Berkeley. Cette technique associe les notions de « sous-ensemble flou » et de « théorie des possibilités ».

Il s'agit d'une approche calquée sur le raisonnement humain plutôt que sur des calculs rigides; pour des problèmes mal définis, l'être humain est irremplaçable.

En effet, le mode de raisonnement en logique floue est plus intuitif que la logique classique. Il permet aux concepteurs de mieux appréhender les phénomènes naturels, imprécis et difficilement modélisables en s'appuyant sur la définition de règles et de fonctions d'appartenance à des ensembles dits « ensembles flous ».

La logique floue est utilisée dans des domaines aussi variés que l'automatisme (freins ABS, conduite de processus), la robotique (reconnaissance de formes), la gestion de la circulation routière (feux rouges), le contrôle aérien (gestion du trafic aérien), l'environnement (météorologie, climatologie, sismologie, analyse du cycle de vie), la médecine (aide au diagnostic), l'assurance (sélection et prévention des risques) et bien d'autres.

- Il existe d'autres algorithmes de l'apprentissage non supervisé :

- K-moyen
- Single-pass
- Suffix tree clustering
- Hierarchical Agglomerative Clustering
- Les cartes auto organisatrices de Kohonen
- Les méthodes statistiques comme par exemple le modèle de mixture gaussienne.

Ces méthodes sont souvent combinées pour obtenir diverses variantes d'apprentissage. L'utilisation de tel ou tel algorithme dépend fortement de la tâche à résoudre (classification, estimation de valeurs, etc.).

7. Facteurs de pertinence et d'efficacité

La qualité de l'apprentissage et de l'analyse dépendent du besoin en amont et *a priori* compétence de l'opérateur pour préparer l'analyse. Elle dépend aussi de la complexité du modèle (spécifique ou généraliste) et de son adaptation au sujet à traiter. Enfin, la qualité du travail dépendra aussi du mode (de mise en évidence visuelle) des résultats pour l'utilisateur final (un résultat pertinent pourrait être caché dans un schéma trop complexe, ou mal mis en évidence par une représentation graphique inappropriée). Avant cela, la qualité du travail dépendra de facteurs initiaux contraignants, liées à la base de données :

- **Nombre d'exemples *:*** moins il y en a plus l'analyse est difficile, mais plus il y en a plus le besoin de mémoire informatique est élevé et plus longue est l'analyse.
- **Nombre et qualité des attributs :** décrivant ces exemples (La distance entre deux "exemples" numériques (prix, taille, poids, intensité lumineuse, intensité de bruit, etc) est facile à établir, celle entre deux attributs catégoriels (couleur, utilité, est plus délicate)
- **Pourcentage de données renseignées** et manquantes
- **« Bruit »** ; Le nombre et la « *localisation* » des valeurs douteuses (erreurs) ou naturellement non conformes au modèle de distribution générale des « *exemples* » sur leur espace de distribution.

II. L'optimisation

1. Introduction :

Les ingénieurs se heurtent quotidiennement à des problèmes technologiques de complexité grandissante, qui surgissent dans des secteurs très divers, comme dans le traitement des images, la conception des systèmes mécaniques, la recherche opérationnelle, ...

Le problème à résoudre peut fréquemment être exprimé sous la forme générale d'un problème d'optimisation, dans lequel on définit une fonction objective, ou fonction de coût (voire plusieurs), que l'on cherche à minimiser par rapport à tous les paramètres concernés. Par exemple, dans le célèbre problème du voyageur de commerce, on cherche à minimiser la longueur de la tournée d'un « voyageur de commerce », qui doit visiter un certain nombre de villes, avant de retourner à la ville de départ. La définition du problème d'optimisation est souvent complétée par la donnée de contraintes : tous les paramètres (ou variables de

décision) de la solution proposée doivent respecter ces contraintes, faute de quoi la solution n'est pas réalisable.

Il existe de nombreuses méthodes d'optimisation « classiques » pour résoudre de tels problèmes, applicables lorsque certaines conditions mathématiques sont satisfaites, ainsi, la programmation linéaire traite efficacement le cas où la fonction objective, ainsi les contraintes, s'expriment linéairement en fonction des variables de décision.

Malheureusement, les situations rencontrées en pratique comportent souvent une ou plusieurs complications, qui mettent en défaut ces méthodes : par exemple, la fonction objective peut être non linéaire, ou même ne peut pas s'exprimer analytiquement en fonction des paramètres; ou encore, le problème peut exiger la considération simultanée de plusieurs objectifs contradictoires. [9]

L'arrivée d'une nouvelle classe de méthodes, nommées métaheuristiques, marque une réconciliation des deux domaines : en effet, celles-ci s'appliquent à toutes sortes de problèmes combinatoires, et elles peuvent également s'adapter aux problèmes continus.
Ces méthodes, qui comprennent notamment la méthode du recuit simulé, les algorithmes génétiques, la méthode de recherche tabou, les algorithmes de colonies de fourmis, etc. sont apparues, à partir des années 1980, avec une ambition commune : résoudre au mieux les problèmes d'optimisation difficiles.

2. Définitions :

2.1 Un problème :

Un problème dans le point de vue informatique (Computational Problem) veut dire comment faire relier un ensemble de données par un ensemble de résultats, où les données vont subir un ensemble d'opérations dont on les appelle le traitement. Donc il est conçu comme une relation $\pi \subseteq I \times S$ entre les entrées ou instances, et les sorties ou solutions. Pour qu'une instance d'un problème soit compréhensible par un ordinateur numérique, elle doit être décrite comme une séquence finie de symboles d'un ensemble fini arbitraire appelé alphabet. De même, la solution d'une instance d'un tel problème est émise dans ce format. Généralement, I est infini alors que S peut être fini [10].

- **Types de problèmes :**

Les problèmes peuvent être classés selon les propriétés de l'ensemble des solutions :

- **Un problème de décision :** c'est un problème dont la réponse est tout simplement OUI ou NON.
- **Un problème polynomial réductible** : on a L1 et L2 deux problèmes de décision, on dit L1 est polynomial réductible à L2 s'il existe un algorithme polynomial qui convertit chaque instance d'entrée de L1 à une autre instance de L2.
- **Un problème de la classe P** : un problème est dit polynomial s'il existe un algorithme de complexité polynomiale permettant de répondre à la question posée dans ce problème, quelque soit la donnée de celui-ci. La classe P est l'ensemble de tous les problèmes de reconnaissances polynomiaux.
- **Un problème de la classe NP :** un problème de reconnaissance est dans la classe NP si, pour toute instance de ce problème, on peut vérifier, en un temps polynomial par rapport à la taille de l'instance, qu'une solution proposée ou devinée permet d'affirmer que la présence est « oui » pour cette instance. [9]
- **Un problème de la classe NP-hard :** on dit qu'un problème est dans la classe NP-hard si chaque autre problème dans NP est réductible d'une manière polynomiale dans ce dernier.
- **Un problème de la classe NP-complet :** s'il appartient à NP et chaque autre problème dans NP est réductible d'une manière polynomiale dans ce dernier. C'est un problème de décision et NP-hard qui cherche à trouver l'optimum. [11]

2.2 Un problème combinatoire :

Un problème combinatoire est toute situation dont on cherche d'avoir une solution tout en respectant la présence d'un ensemble de contraintes. La solution c'est un résultat de faire combiner ces contraintes ensemble d'une manière qu'on maximise quelques uns et on minimise les autres, ces contraintes ont une caractéristique primordiale, c'est que chaque contrainte influe sur les autres soit quand on minimise sa valeur ou on la maximise, dans un autre terme on dit que les contraintes sont conflictuelles.

Par exemple, le schéma suivant présente une situation de problème combinatoire: où on veut acheter une voiture dans la mode et en même temps avec un prix raisonnable qui ne peut pas dépasser certaine limite. Si on maximise la première contrainte (une bonne voiture) on va avoir un prix maximale, dans le contraire on va aboutir à une mauvaise voiture mais avec un prix minimale dans les limites; on constate dans cet exemple que c'est difficile d'arranger ces deux contraintes dans nos besoins.

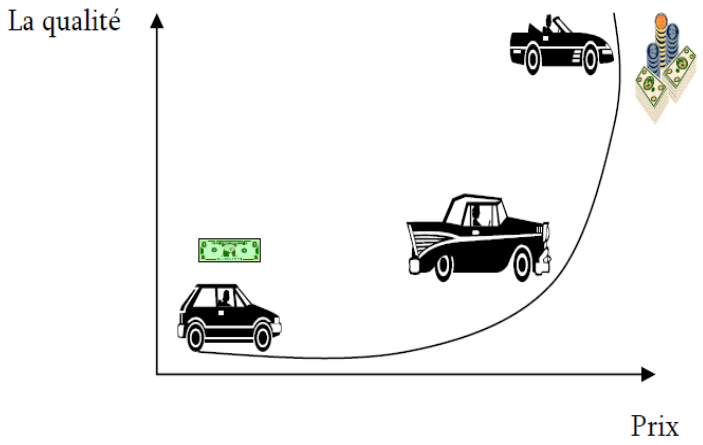


Figure 2: une figure illustrant un problème combinatoire.

2.3 Un problème d'optimisation :

- **Définition :**

Les problèmes d'optimisation ont été définis pour représenter les problèmes décrits dans le paragraphe précédent. Ces problèmes occupent actuellement une place de choix dans la communauté scientifique. Non pas qu'ils aient été un jour considérés comme secondaires mais l'évolution des techniques informatiques a permis de dynamiser les recherches dans ce domaine.

Le monde réel offre un ensemble très divers de problèmes d'optimisation :

- Problème combinatoire ou à variables continues.
- Problèmes à un ou plusieurs objectifs.
- Problèmes statistiques ou dynamiques.
- Problème dans l'incertain. [12]

Un tel problème est caractérisé par :

- Un problème d'optimisation est défini par un **espace d'état**, **une ou plusieurs fonction(s) objective(s)** et un **ensemble de contraintes**.
- **L'espace d'état** est défini par l'ensemble de domaines de définition des variables du problème.
- **Les variables** du problème peuvent être de nature diverse (réelle, entière, booléenne, etc.) et sont exprimées de données qualitatives ou quantitatives.

- **Une fonction objective** représente le but à atteindre pour le décideur (minimisation de coût, de durée, d'erreur, ...). Elle définit un espace de solutions potentielles au problème.
- **L'ensemble de contraintes** définit des conditions sur l'espace d'état que les variables doivent satisfaire. Ces contraintes sont souvent des contraintes d'inégalité ou d'égalité et permettent en général de limiter l'espace de recherche.
- **Une méthode d'optimisation** cherche un point ou un ensemble de points de l'espace des états possibles qui satisfait au mieux un ou plusieurs critères. Le résultat est appelé valeur optimale ou optimum. [12]

On peut dire qu'un problème d'optimisation se définit comme la recherche du minimum ou du maximum (de l'optimum donc) d'une fonction donnée. On peut aussi trouver des problèmes d'optimisation pour lesquels les variables de la fonction à optimiser sont des contraintes à évoluer dans une certaine partie de l'espace de recherche.
Ce besoin d'optimisation vient de la nécessité de l'ingénieur de fournir à l'utilisateur un système qui répond au mieux au cahier des charges. Ce système devra être calibré de manière à:

- Occuper le volume nécessaire à son bon fonctionnement (coût des matières premières),
- Consommer le minimum d'énergie (coût de fonctionnement),
- Répondre à la demande de l'utilisateur (cahier des charges).

Dans ce sens, nous pouvons définir :

- **Fonction objective :**

C'est le nom donné à la fonction ***f*** (on l'appelle encore fonction de coût ou critère d'optimisation). C'est cette fonction que l'algorithme d'optimisation va devoir « optimiser » (trouver un optimum).

- **Variables de décision :**

Elles sont regroupées dans le vecteur $\vec{x}$. C'est en faisant varier ce vecteur que l'on cherche un optimum de la fonction ***f***.

- **Minimum global :**

Un « point » $\vec{x}*$ est un minimum global de la fonction ***f*** si on a :

$f(\vec{x}*) < f(\vec{x})$ Quel que soit $\vec{x}$ tel que $\vec{x}* \neq \vec{x}$

Exemple : M3 dans la figure 3

- **Minimum local fort :**

Un « point » $\vec{x}^*$ est un minimum local fort de la fonction ***f*** si on a :

$f(\vec{x}^*) < f(\vec{x})$ Quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$, où $V(\vec{x}^*)$ définit un « voisinage » de $\vec{x}^*$.

Exemple **:** M2 et M4 dans la figure 3

- **Minimum local faible :**

Un « point » $\vec{x}^*$ est un minimum local faible de la fonction ***f*** si on a :

$f(\vec{x}^*) \leq f(\vec{x})$ Quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$, où $V(\vec{x}^*)$ définit un « voisinage » de $\vec{x}^*$ [21].

Exemple : M1 dans la figure 3

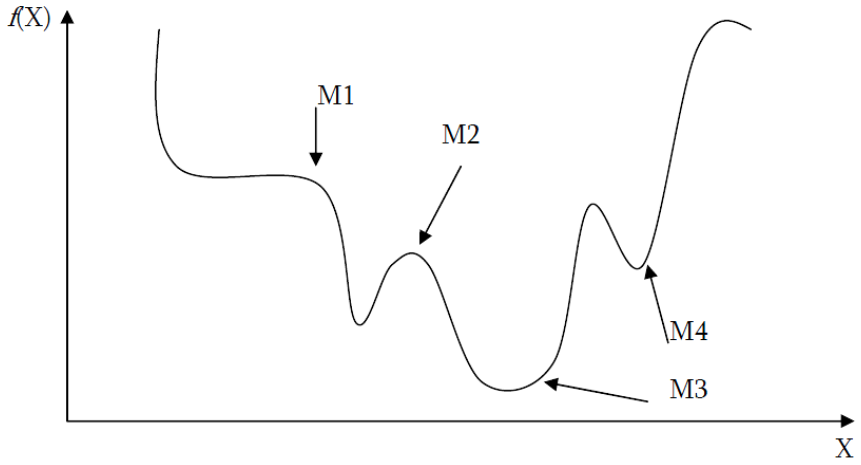


Figure 3: différents Minima.[9]

- ## Classification des problèmes d'optimisation :

On peut classer les différents problèmes d'optimisation que l'on rencontre dans la vie courante en fonction de leurs caractéristiques :

- **Le domaine des variables de décision :** soit Continu et on parle alors de **problème continu**, soit discret et on parle donc de **problème combinatoire ;**
- **La nature de la fonction objectif à optimiser :** soit linéaire et on parle alors de **problème linéaire**, soit non linéaire et on parle donc de **problème non linéaire** ;
- **Le nombre de fonctions objectifs à optimiser :** soit une fonction scalaire et on parle alors de **problème mono-objectif**, soit une fonction vectorielle et on parle donc de **problème multiobjectif ;**

- **La présence ou non des contraintes :** on parle de **problème sans contrainte** ou **avec contrainte**
- **Sa taille : problème de petite** ou **de grande taille ;**
- **L'environnement : problème dynamique** (la fonction objectif change dans le temps).

- Les problèmes d'optimisation mono-objectifs

Lorsqu'un seul objectif (critère) est donné, le problème d'optimisation est mono-objectif. Dans ce cas la solution optimale est clairement définie, c'est celle qui a le coût optimal (minimal, maximal). De manière formelle, à chaque instance d'un tel problème est associé un ensemble **W** des solutions potentielles respectant certaines contraintes et une fonction d'objectif $f : \mathbf{W} \rightarrow \mathbf{Y}$ qui associe à chaque solution admissible $s \in \mathbf{W}$ une valeur $f(s)$. Résoudre l'instance (**W**, f) du problème d'optimisation consiste à trouver la solution optimale $s^* \in \mathbf{W}$ qui optimise (minimise ou maximise) la valeur de la fonction objectif f.

- Optimisation combinatoire

L'Optimisation combinatoire consiste à trouver le meilleur entre un nombre fini de choix. Autrement dit, minimiser une fonction, avec contraintes, sur un ensemble fini. Quand le nombre de choix est exponentiel (par rapport à la taille du problème), mais que les choix et les contraintes sont bien structurés, des méthodes mathématiques doivent et peuvent intervenir. L'Optimisation combinatoire se situe au carrefour de la *Théorie des graphes*, de la *Programmation mathématique*, de *l'Informatique théorique* (algorithmique et théorie de la complexité) et de la *Recherche opérationnelle.*

3. Les méthodes d'optimisation

Un très grand nombre de méthodes d'optimisation existent dans la littérature pour l'optimisation combinatoire. La **figure** 4 met en parallèle les différentes méthodes d'optimisation.

Ces méthodes font partie de deux groupes de nature différente :

- **Le premier groupe**, comprend les méthodes exactes d'arborescence qui garantissent la complétude de la résolution : c'est le cas de A*, Branch and Bound et programmation dynamique. Le temps de calcul nécessaire d'une telle méthode augmente en général exponentiellement avec la taille du problème à résoudre (dans le pire des cas).

- **Le second groupe**, comprend les méthodes approchées dont le but est de trouver une solution de bonne qualité en un temps de calcul raisonnable sans garantir l'optimalité de la solution obtenue. Dans ce cas, les méthodes approchées sont fondées principalement sur diverses heuristiques, souvent spécifiques à un type de problème.

Les métaheuristiques constituent une autre partie importante des méthodes approchées et ouvrent des voies très intéressantes en matière de conception de méthodes heuristiques pour l'optimisation combinatoire. [13]

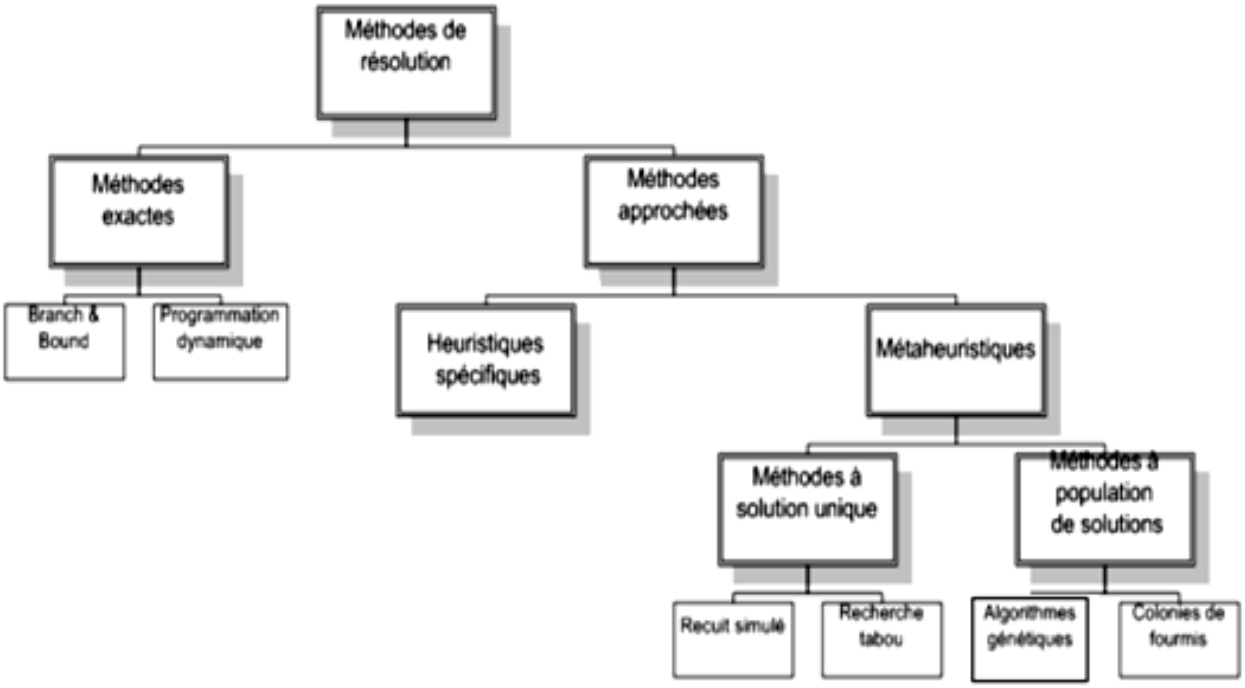


Figure 4: Les méthodes d'optimisation.

3.1 Les méthodes exactes :

Les méthodes exactes reposent sur l'utilisation d'algorithmes qui mènent de façon sûre vers la solution optimale. Le principe essentiel de ces méthodes est d'énumérer de manière implicite l'ensemble des solutions de l'espace de recherche. Malgré le temps de calcul important que nécessitent, généralement, ces approches, plusieurs méthodes ont été développées. [13]

- **Méthode "Branch and Bound"**

Un algorithme par séparation et évaluation est une méthode générique de résolution de problèmes d'optimisation, et plus particulièrement d'optimisation combinatoire ou discrète. C'est une méthode d'énumération implicite : toutes les solutions possibles du problème peuvent être énumérées mais, l'analyse des propriétés du problème permet d'éviter l'énumération de larges classes de mauvaises solutions. Dans un bon algorithme par séparation et évaluation, seules les solutions potentiellement bonnes sont donc énumérées.

Le principe de la méthode de séparation & évaluation B & B est de découper l'espace de recherche en domaines plus restreints afin de séparer l'optimum global.

L'algorithme de recherche forme donc un arbre dont chaque nœud représente une partie de l'espace. Ensuite chaque nœud est évalué de façon à déterminer sa borne (bound) inférieure [3] en fonction d'un critère d'évaluation. Si cette borne n'est pas meilleure que la solution courante alors la recherche est stoppée, sinon la séparation continue. [13]

- **Programmation dynamique**

La programmation dynamique est une technique algorithmique pour optimiser des sommes de fonctions monotones croissantes sous contrainte.

Elle a été désignée par ce terme pour la première fois dans les années 1940 par Richard Bellman.

Elle s'applique à des problèmes d'optimisation dont la fonction objectif se décrit comme « la somme de fonctions monotones croissantes des ressources ».

Elle a d'emblée connu un grand succès, car la plupart des fonctions économiques de l'industrie étaient de ce type : maximisation du tonnage de charbon (ou de barils de pétrole) produit à partir de plusieurs puits à budget donné, par exemple.

La programmation dynamique s'appuie sur un principe simple : toute solution optimale s'appuie elle même sur des sous problèmes résolus localement de façon optimale. Concrètement, cela signifie que l'on va pouvoir déduire la solution optimale d'un problème en combinant des solutions optimales d'une série de sous problèmes.

Les solutions des problèmes sont étudiées 'de bas en haut', c'est-à-dire qu'on calcule les solutions des sous-problèmes les plus petits pour ensuite déduire petit à petit les solutions de l'ensemble. [13]

[3] Inférieure dans le cas d'une minimisation, Supérieure dans le cas d'une maximisation.

3.2 Les méthodes approchées :

Le but d'une méthode approchée est de trouver une solution réalisable, tenant compte de la fonction objectif mais sans garantie d'optimalité. Son utilisation offre de multiples avantages par rapport à une méthode exacte, citons :

- Elles sont plus simples et plus rapides à mettre en œuvre quand la qualité de la solution n'est pas trop importante.
- Elles sont plus souples dans la résolution des problèmes réels. En pratique, il arrive souvent que l'on doive prendre en considération de nouvelles contraintes qu'on ne pouvait pas formuler dès le départ. Ceci peut être fatal pour une méthode exacte si les nouvelles contraintes changent les propriétés sur lesquelles s'appuyait la méthode.
- Elles fournissent des solutions et des bornes qui peuvent être utiles dans la conception de méthodes exactes. [14]

- **Les heuristiques**

Une heuristique est une technique qui améliore l'efficacité d'un processus de recherche, en sacrifiant éventuellement l'exactitude ou l'optimalité de la solution. Pour des problèmes d'optimisation (NP complets) où la recherche d'une solution exacte (optimale) est difficile (coût exponentiel), on peut se contenter d'une solution satisfaisante donnée par une heuristique avec un coût plus faible. Méthode de recherche guidée par des "astuces" qui dépendent du problème traité.

- **Les métaheuristiques**

Les métaheuristiques sont un ensemble d'algorithmes d'optimisation visant à résoudre les problèmes d'optimisation difficiles. Elles sont souvent inspirées par des systèmes naturels, qu'ils soient pris en physique (cas du recuit simulé), en biologie de l'évolution (cas des algorithmes Génétiques) ou encore en éthologie (cas des algorithmes de colonies de fourmis ou de l'optimisation par essaims particulaires). Ces techniques métaheuristiques peuvent être classées en deux groupes : les méthodes à population de solutions connues sous le nom d'algorithmes évolutionnaires comme les algorithmes génétiques…etc., ainsi que les méthodes à solution unique comme le recuit simulé. Les méthodes métaheuristiques ont prouvé leurs efficacités dans le domaine de l'optimisation mono-objectif. Actuellement les recherches qui utilisent ces algorithmes sont développées pour la résolution des problèmes d'optimisation multi objectif, en tenant

compte de plusieurs contraintes et de nouvelles configurations des réseaux électriques surtout à l'associations de sources des énergies renouvelables où la résolution de ce system complexe est un défi .[15]

- **Métaheuristique à base de solution unique**

Les métaheuristiques à base de solution unique débutent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante. En fait, la solution initiale subit une série de modifications en fonction de son voisinage. Le but de ces modifications locales est d'explorer le voisinage de la solution actuelle afin d'améliorer progressivement sa qualité au cours des différentes itérations. [14]

- **Le recuit simulé :**

Une nouvelle technique de résolution des problème d'optimisation est nommée recuit(RS) simulé, proposée en 1983 par Kirkpatrick, C.Daniel Gelatt et Mario P Vecchi [9]. Elle est testée sur plusieurs problèmes d'optimisation et prouve qu'elle possède une grande capacité pour éviter le minimum local. (RS) est une méthode basée sur la recherche locale dans laquelle chaque mouvement est accepté s'il améliore la fonction objective. Autres solutions possibles sont également acceptées selon un critère de probabilité. Cette méthode est inspirée du processus de recuit utilisé en métallurgie pour améliorer la qualité d'un solide en cherchant un état d'énergie minimum. La méthode du recuit simulé, appliquée aux problèmes d'optimisation, considère une solution initiale et cherche dans son voisinage une autre solution de façon aléatoire [16].

- **La recherche Tabou :**

La méthode taboue qui fait partie des méthodes de voisinage, a été proposée par F.Glover durant les années 1980 [16]. Elle utilise la notion de mémoire pour éviter un optimum local. Le principe de l'algorithme est le suivant; à chaque itération, le voisinage de la solution est sélectionné en appliquant le principe de voisinage. La méthode autorise de remonter vers des solutions qui semblent moins intéressantes mais qui ont peut être un meilleur voisinage. Des fois, ce principe engendre des phénomènes de cyclage entre deux solutions, tandis que la méthode taboue a l'interdiction de visiter une solution récemment visitée. Pour cela, une liste taboue contenant les attributs des dernières solutions

considérées est tenue à jour. Chaque nouvelle solution considérée enlève de cette liste la solution la plus anciennement visitée. Ainsi, la recherche de la solution suivante se fait dans le voisinage de la solution actuelle sans considérer les solutions appartenant à la liste taboue.

- **Métaheuristiques à base de population de solutions**

Les métaheuristiques à base de population de solutions débutent la recherche avec une panoplie de solutions. Elles s'appliquent sur un ensemble de solutions afin d'en extraire la meilleure (l'optimum global) qui représentera la solution du problème traité. L'idée d'utiliser un ensemble de solutions au lieu d'une seule solution renforce la diversité de la recherche et augmente la possibilité d'émergence de solutions de bonne qualité. [14]

- **Colonies de fourmis**

Les algorithmes de colonies de fourmis ont été proposés par Colorni, Dorigo et Maniezzo en 1992 [18] et appliquées la première fois au problème du voyageur de commerce. Ce sont des algorithmes itératifs à population où tous les individus partagent un savoir commun qui leur permet d'orienter leurs futurs choix et d'indiquer aux autres individus des choix à suivre ou à éviter. Le principe de cette métaheuristique repose sur le comportement particulier des fourmis, elles utilisent pour communiquer une substance chimique volatile particulière appelée phéromone grâce à une glande située dans leur abdomen. [19]

- **Les algorithmes évolutionnaires**

Les algorithmes évolutionnaires s'inspire de l'évolution naturelle des êtres vivants. Ils adoptent une sorte d'évolution artificielle pour améliorer la qualité des individus de la population. En fait, ils font évoluer itérativement une population d'individus. Ces derniers représentent des solutions du problème traité. Les qualités des individus sont mesurées à chaque itération du processus de d'évolution. En fonction de leurs qualités, les meilleurs individus seront sélectionnés pour subir des combinaisons qui permettent la production d'une nouvelle population (dite: population d'enfants). Les individus (ou une partie des individus) de la nouvelle population vont remplacer les individus de la population courante (dite: population de parents) pour construire la

nouvelle génération d'individus. Le processus d'évolution d'un algorithme évolutionnaire est basé sur trois opérations principales: la sélection, la reproduction et l'évaluation. [14]

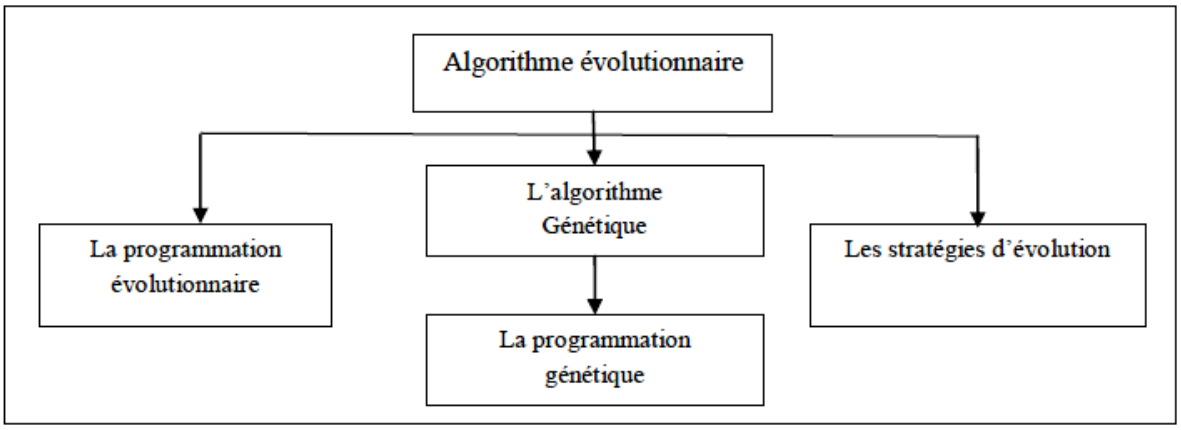


Figure 5: Les types des algorithmes évolutionnaires.

Les algorithmes évolutionnaires forment une classe principale de trois sous classes d'algorithmes (**figure 5**): les stratégies d'évolution, la programmation évolutionnaire et les algorithmes génétiques.

- **Les stratégies d'évolution :** Ont été conçues pour la résolution des problèmes d'optimisation continus.

- **La programmation évolutionnaire :** Les algorithmes de la classe de la programmation évolutionnaire sont conçus pour faire évoluer des structures d'automates à état fini.

- **L'algorithme génétique :** Il sera bien détaillé dans le chapitre suivant. Il est à noter qu'il existe une sous classe de la classe des algorithmes génétiques appelée « La programmation génétique». Cette dernière utilise des structures arborescentes pour représenter les individus de la population. [14]

➢ Autres méthodes à base de population :

D'autres méthodes à base de population existent :

- **La programmation génétique :**

Qui est apparue initialement comme sous-domaine des algorithmes génétiques, mais actuellement elle tend à être considérée comme une branche à part entière.

- **Optimisation par essaim de particules**

L'optimisation par particule d'essaim (PSO) c'est un algorithme d'intelligence d'essaim, inspiré par le social dynamique et le comportement émergeant qui surgissent dans les colonies socialement organisées, PSO est un algorithme basé sur la population, c'est-à-dire il exploite une population d'individus pour explorer les régions prometteuses de l'espace de recherche. Dans ce contexte, la population est nommée les essaims et les individus (c'est-à-dire les point de recherche) sont nommés les particules. Chaque particule meut avec une vitesse adaptable avec l'espace de recherche, et garde une mémoire de la meilleure position dont elle rencontre. Dans les variant global de PSO, la meilleure position qui est retenue par les individus de l'essaim est communiquée aux autres particules. Dans le variant local, chaque particule est assignée à un voisinage topologique consisté d'un nombre pré-spécifié de particules. Dans ce cas, la meilleure position retenue par les particules qui comprend le voisinage est communiquée entre eux. [20]

4. Conclusion

Dans le présent chapitre, nous avons présenté la catégorie de l'apprentissage automatique et les différents domaines où elle peut intervenir, et l'optimisation et les méthodes d'optimisation les plus utilisées.

Dans le chapitre suivant, nous avons présenté la biologie moléculaire, aussi nous avons présenté les éléments de base de la biologie moléculaire, et le domaine de la bioinformatique, ensuite les différentes représentations de motifs biologiques pour le problème de découverte de motifs.

Chapitre II

La Biologie Moléculaire et la bioinformatique et découvert de motif

I. Biologie moléculaire:

1. Introduction :

Au milieu du XXe siècle **(le vingtième siècle**), la découverte de la structure d'ADN représente un tournant majeur dans l'histoire de la biologie, c'est la naissance d'une nouvelle discipline, la biologie moléculaire. La fin du XXe siècle est également marquée par le développement exponentiel de la génétique.

La biologie moléculaire est essentiellement l'étude des molécules qui constituent les êtres vivants et des processus moléculaires qui assurent leur fonctionnement. Dans cette partie de mémoire on va voir une liste non exhaustive des molécules et quelques processus moléculaires indispensables pour la compréhension des chapitres suivants.

2. Définition:

La biologie moléculaire a contribué de manière fondamentale au développement de la biologie moderne. On connaît maintenant la structure et le mode d'action des acides nucléiques (**ADN**, **ARN**) et des protéines (**A, C, G ou T**), les molécules de base de la matière vivante. La découverte des mécanismes biochimiques de l'hérédité fut un énorme progrès. Un autre grand pas fut réalisé lorsque l'on comprit le rôle des molécules dans le métabolisme, mécanisme produisant l'énergie nécessaire à la vie [24].

3. La cellule :

La cellule (du latin cellula, petite chambre) est l'unité structurale, fonctionnelle et reproductrice constituant tout ou partie d'un être vivant. Chaque cellule est une entité vivante qui, dans le cas d'organismes multicellulaires, fonctionne de manière autonome mais coordonnée avec les autres.
La cellule est donc une enceinte séparée de l'extérieur par une **membrane** capable de filtrer sélectivement les échanges.

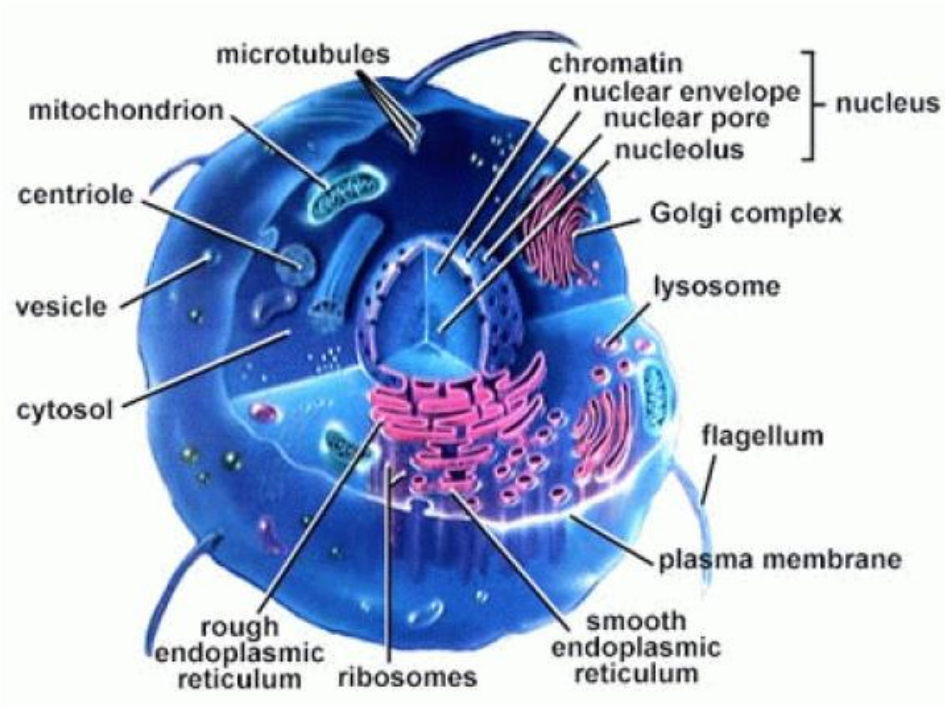


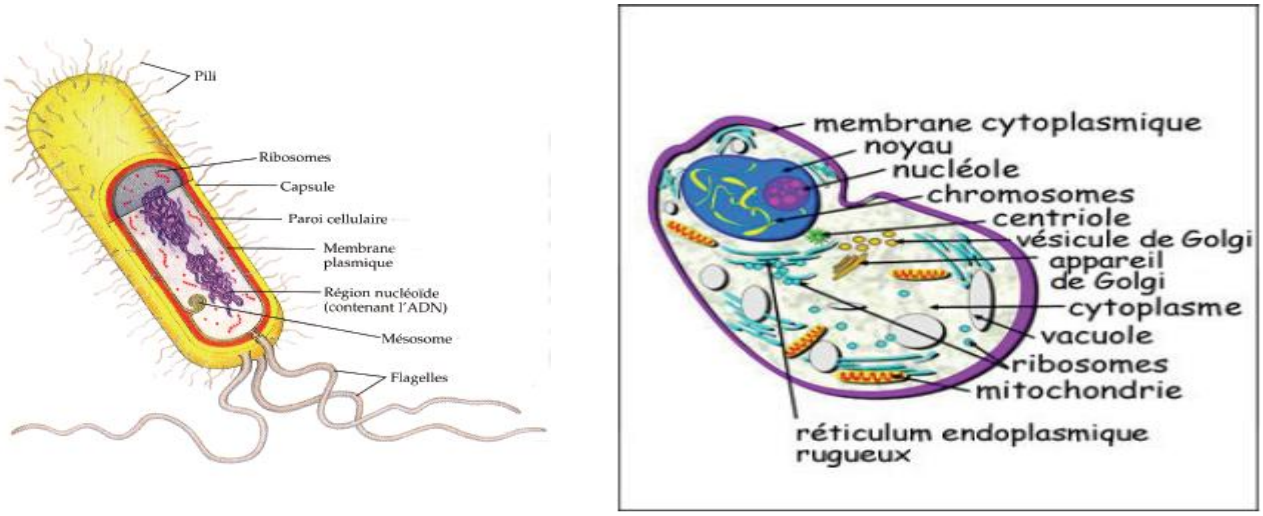
Figure 6: la structure d'une cellule.

Il existe deux types de cellules: Procaryote et Eucaryotes :

- **Les Procaryotes** (du grec **pro**, avant et **karyon**, noyau) sont les bactéries. Elles vivent sous forme unicellulaire, même si elles savent s'organiser sous formes de communauté de citoyens libres. Elles ont une membrane (une peau) rigide et ne possèdent pas de noyaux.

- **Les Eucaryotes** sont des cellules à enveloppe extérieure souple (pour ne pas dire molle) et possèdent un noyau où l'information génétique est entreposée. constitués de plusieurs brins linéaires d'ADN enroulé en double hélice sur des protéines les histones.

Chaque cellule contient la même information mais l'exprime spécifiquement selon sa fonction et son rôle.



(a) cellule procaryote **(b) cellule eucaryote**

Figure 7: Les constituants des cellules.

4. Les acides nucléiques :

Les acides nucléiques ont été isolés initialement des noyaux des cellules. On peut en distinguer deux grands types:

- les acides désoxyribonucléiques (ADN): essentiellement localisés dans le noyau des cellules.
- les acides ribonucléiques (ARN): essentiellement localisés dans le cytoplasme cellulaire.

Ces molécules biologiques (les acides nucléiques) contiennent l'information génétique. Les acides nucléiques (ADN et ARN) sont des macromolécules composés de molécules simples et comportent des sous-unités appelées nucléotides [23].

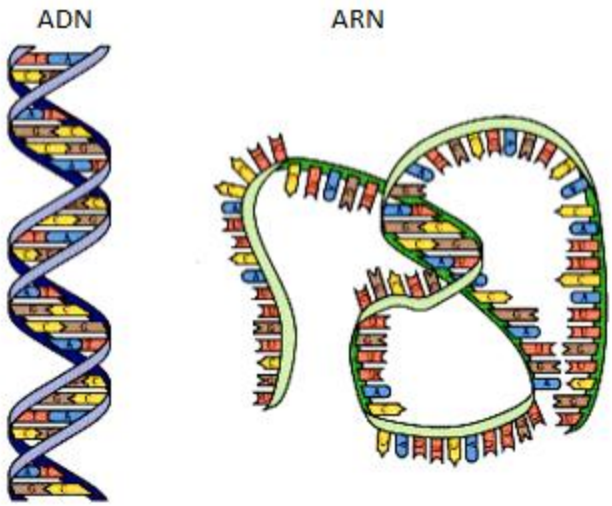


Figure 8: Acide nucléique.

5. Acide désoxyribonucléique (ADN) :

L'acide désoxyribonucléique (ADN ou DNA) est constitué de deux chaînes de nucléosides monophosphates liés chacun par une liaison ester entre son carbone 3' (alcool secondaire) et le carbone 5' (alcool primaire) du nucléotide suivant. Ces deux chaînes de nucléotides sont unies entre elles par des liaisons hydrogènes pour former un hybride en forme de double hélice (modèle de J.D. Watson et F.H.C. Crick, 1953) dont les deux brins sont :

- Antiparallèles : l'un est constitué d'un enchaînement commençant à gauche et se poursuivant vers la droite, l'autre commençant à droite et se poursuivant vers la gauche.
- Complémentaires : chaque **adénine (A)** d'un des deux brins est liée par deux liaisons hydrogène avec une **thymine (T)** de l'autre brin, et chaque **guanine (G)** d'un brin est liée par trois liaisons hydrogène avec une **cytosine (C)** de l'autre brin.

De sorte que si on désigne les nucléotides par les lettres A, C, G et T en fonction des bases azotées qu'ils contiennent, par exemple la **figure 9** [22]:

...AGAGTCGTCTCGAGTCA...
...TCTCAGCAGAGCTCAGT...

Figure 9: Fragment d'un gène.

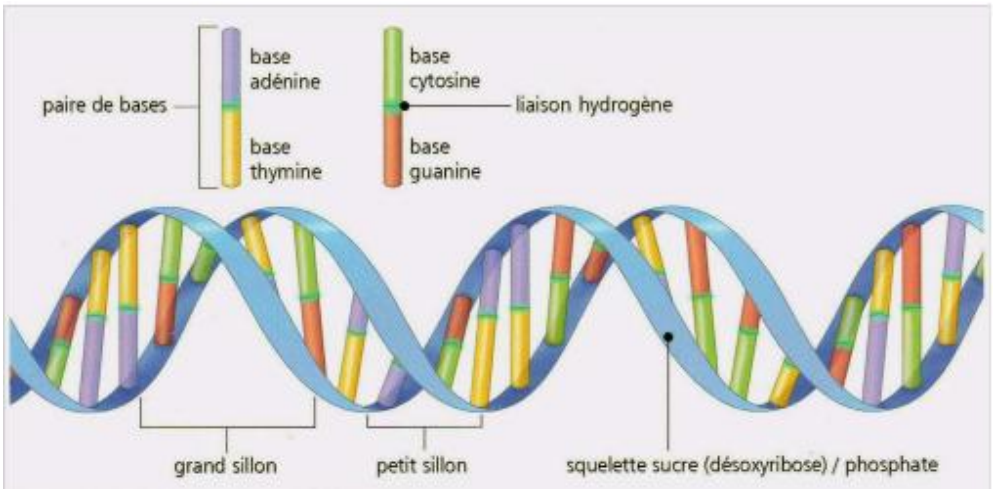


Figure 10: Schéma de la molécule d'ADN.

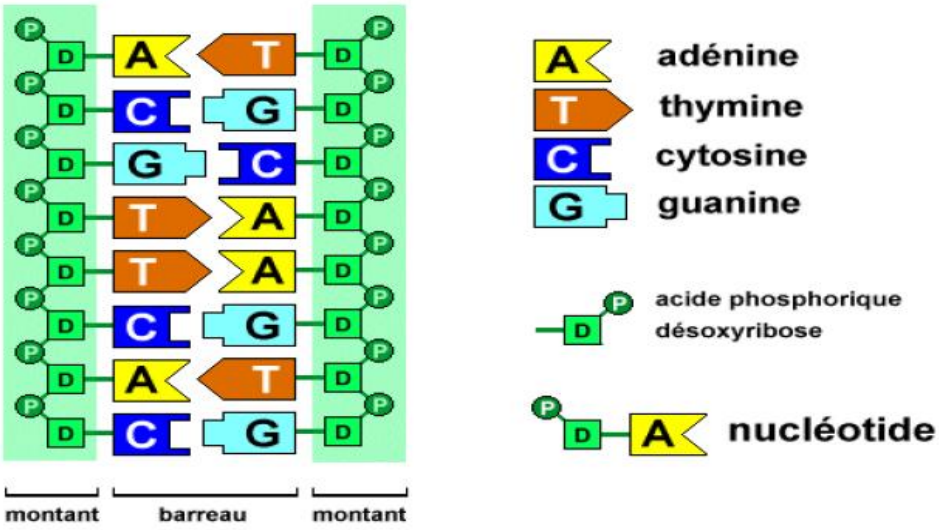


Figure 11: Structure d'ADN.

6. Acide ribonucléique (ARN) :

L'acide ribonucléique, ou ARN ressemble beaucoup à l'ADN sauf que le sucre de l'ADN (désoxyribose) est remplacé par un autre sucre dans l'ARN (ribose). La thymine (T) de l'ADN est remplacée par l'uracile (U) dans l'ARN.

L'ARN peut s'apparier avec un ARN complémentaire, mais les ARNs sont généralement simple brin et sont donc le siège d'appariements intramoléculaires. En effet, les ARNs sont issus de la transcription de l'ADN par une enzyme (l'ARN Polymérase) qui recopie en quelque sorte la séquence comme illustré en **Figure12**.

On connait depuis longtemps trois types d'ARN:

- L'ARN messager (ARNm) : est une copie de l'ADN utilisée comme intermédiaire dans la synthèse des protéines entre le noyau et les ribosomes.
- L'ARN de transfert (ARNt) : qui fait reconnaître les acides aminés à l'ARNm pour synthétiser les protéines.
- L'ARN ribosomal (ARNr) : qui rentre dans la structure du ribosome qui forme la machine de synthèse protéique.

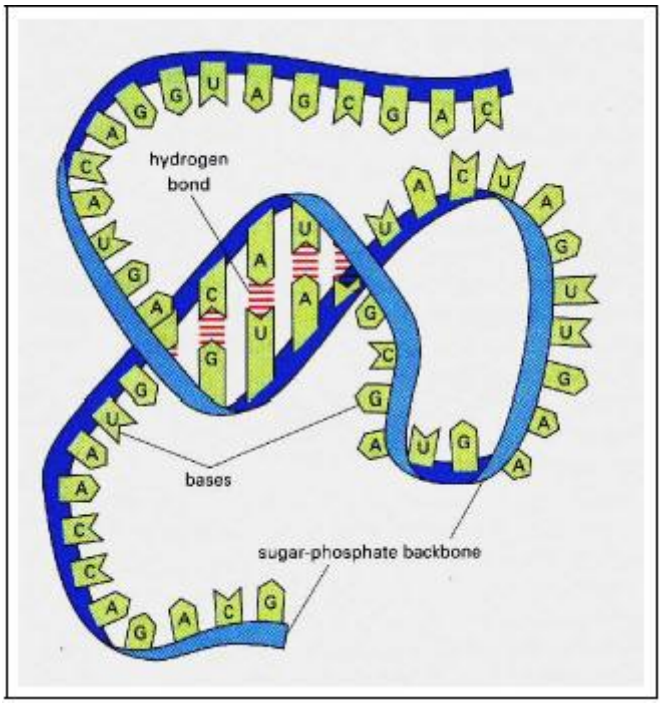


Figure 12: La structure d'ARN.

Le RNA diffère du DNA par plusieurs caractères :

- Il est plus court (70 à 10 000 nucléotides)
- Le squelette de pentoses et de phosphates contient du ribose à la place du désoxyribose
- Parmi les bases azotées l'uracile (U) remplace la thymine (T)
- Les RNA sont simple brin mais certaines régions sont appariées sur une courte distance par leurs bases complémentaires selon un ajustement au hasard (épingles à cheveux) [23].

7. Protéine :

La protéine est composée d'une séquence d'acides aminés. Il en existe 20 différents. Des protéines agissent tout au long de la synthèse des protéines. De même que pour la transcription, la traduction consiste à fabriquer la séquence d'acides aminés à partir d'acides aminés isolés dans la cellule, en prenant comme modèle l'ARN messager. Cette fois-ci, à un codon (3 bases à la suite) correspond un acide.

Les protéines qui traduisent reconnaissent un codon **Start** et un codon **Stop** sur l'ARN pour commencer et terminer le travail.

- Un codon initiateur (ATG) à partir duquel commence la traduction (signal d'initiation de la traduction) et qui code aussi pour un acide aminé, la méthionine (appelée encore méthionine initiatrice quand il s'agit du codon correspondant au signal de début de la traduction) ;
- Trois codons particuliers, les « codons-stop» (TAA, TAG, TGA), signaux de fin de traduction.

8. Génome et Gène

• Génome

Le génome contient l'information nécessaire pour construire et maintenir le fonctionnement d'une cellule, unité de base du vivant. Le génome est composé d'un ou de plusieurs chromosomes. Cet ensemble de chromosomes peut être vu comme un support d'information ; le métabolisme, c'est-à-dire le fonctionnement des cellules, est quant à lui assuré en majorité par des protéines issues de l'information du génome. Dans les génomes eucaryotes, l'information est portée par plusieurs chromosomes. Dans les bactéries, le génome est en général constitué d'un seul chromosome circulaire [21].

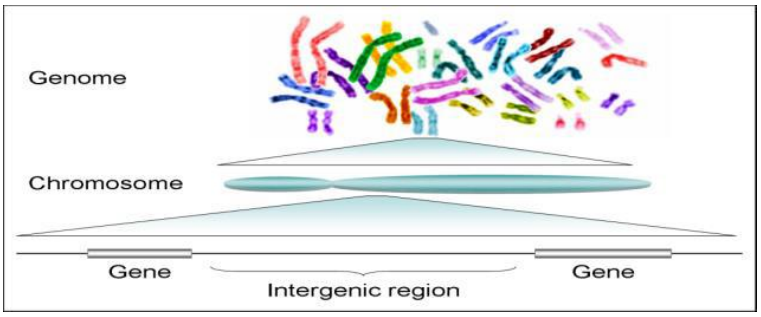


Figure 13: caryotype.

- **Gène**

Un gène est une portion d'ADN qui contient toute la recette d'assemblage d'une protéine.

Ce sont les protéines qui font l'essentiel du travail dans l'organisme et les gènes permettent leur reproduction de génération en génération.

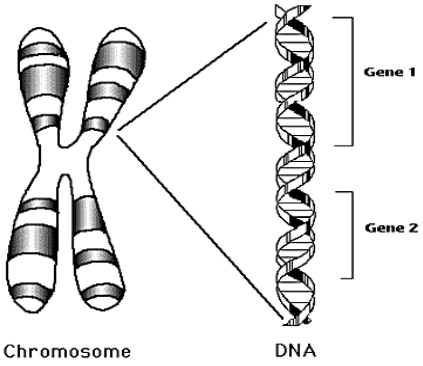


Figure 14: l'emplacement d'un gène dans un chromosome.

9. Information génétique :

Le dogme de Crick (qui a découvert avec Watson la structure en double hélice de l'ADN) dit que l'information transmise par un être à sa progéniture ne repose que sur l'ADN. Certaines parties de l'ADN sont transcrites en molécules d'ARN. Ces molécules d'ARN messager sont ensuite traduites en chaînes polypeptidiques [21].

10. Transcription

La transcription transforme des parties spécifiques de l'ADN en ARN simple brin appelé ARN messager (noté ARNm dans l'illustration donnée à la figure 15). Cette transformation est faite à l'aide de protéines appelées facteurs de transcription, et d'une autre protéine, l'ARN polymérase. L'ARN produit est transformé chez les eucaryotes. Il s'agit de l'épissage : des sous séquences appelées introns (les introns sont des segments d'ADN localisés au sein du gène, transcrit en ARN) sont retirées de l'ARN, laissant une chaîne formée par la concaténation des parties restantes appelées exons (les exons sont les régions de gène qui contiennent l'information codante) [21].

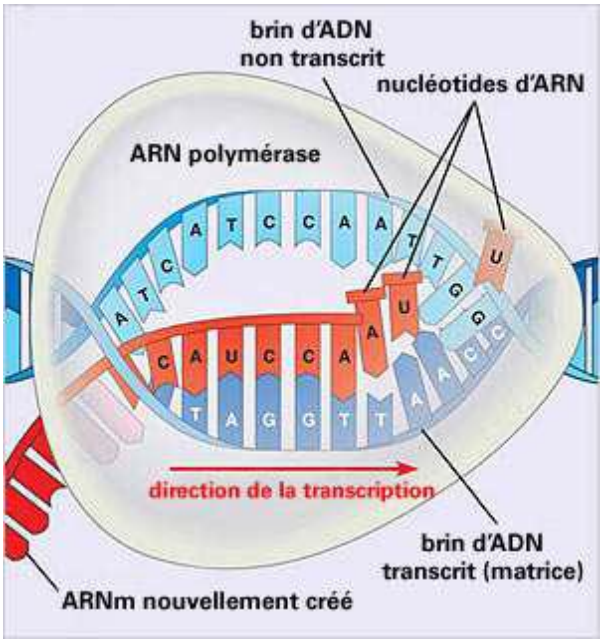


Figure 16: Transcription.

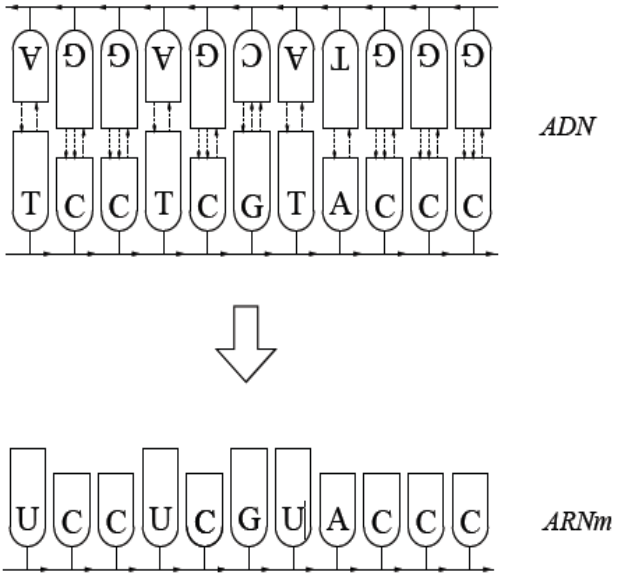


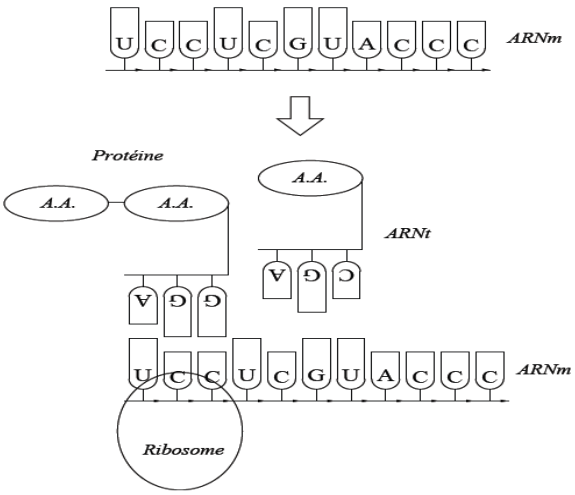
Figure 15: Le mécanisme de la transcription

11. Traduction

La traduction est l'étape qui transforme un ARN messager en protéine. Cette étape fait intervenir un complexe nommé ribosome composé de deux ARN et de protéines, ainsi que des ARN de transfert (notés ARNt dans l'illustration donnée à la figure 18).

Une protéine est composée par une chaîne d'acides aminés (il existe 20 types d'acides aminés différents). Chaque acide aminé est encodé par un triplet de nucléotides appelé codon : comme il existe 20 types acides aminés et que le codage utilisé autorise 64 combinaisons, le code utilisé est donc redondant (voir la figure 19). Un codon particulier AUG sert d'amorce (Start) au cadre de lecture, l'ARN est ensuite lu par triplet jusqu'à ce qu'un codon stop (défini par UGA, UAA, UAG) soit trouvé.

Le code génétique établissant la correspondance entre codons et acides aminés est identique pour la plupart des organismes : il s'agit du code génétique universel. Seuls certains protozoaires, et beaucoup de mitochondries ont un code légèrement différent de celui donné dans la figure 19 [21].



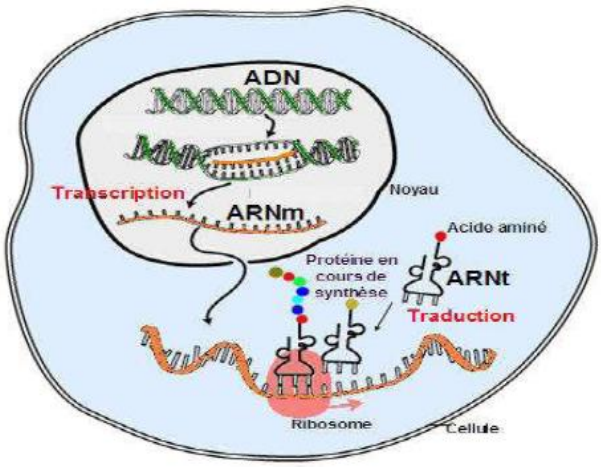


Figure 17: Récapitulatif ADN →ARNm→Protéine

Figure 18: Traduction.

1er nucléotide	2ème nucléotide U	C	A	G	3ème nucléotide
U	Phe	Ser	Tyr	Cys	U
	Phe	Ser	Tyr	Cys	C
	Leu	Ser	Stop	Stop	A
	Leu	Ser	Stop	Trp	G
C	Leu	Pro	His	Arg	U
	Leu	Pro	His	Arg	C
	Leu	Pro	Gln	Arg	A
	Leu	Pro	Gln	Arg	G
A	Ile	Thr	Asn	Ser	U
	Ile	Thr	Asn	Ser	C
	Ile	Thr	Lys	Arg	A
	Met	Thr	Lys	Arg	G
G	Val	Ala	Asp	Gly	U
	Val	Ala	Asp	Gly	C
	Val	Ala	Glu	Gly	A
	Val	Ala	Glu	Gly	G

Figure 19: code génétique.

I. La bioinformatique

1. Introduction

La bioinformatique est une discipline relativement récente, le terme ayant été créé dans les années 80. Cette notion englobe l'ensemble des applications de l'informatique aux sciences de la vie, domaine très vaste qui recouvre tous les axes de recherche, allant des applications en robotique aux techniques les plus avancées en intelligence artificielle. Pour la plupart des membres de la communauté scientifique, cette notion semble dans la pratique s'adapter plus particulièrement aux outils informatiques qui permettent de stocker, d'analyser et de visualiser les informations contenues dans les séquences des gènes et des protéines des êtres vivants. L'histoire de la bioinformatique est donc étroitement liée à celle de la biologie moléculaire, l'étude des molécules du vivant.

Il est intéressant de constater que l'essor des connaissances en biologie moléculaire progresse parallèlement à celle de l'informatique. En ce qui concerne la biologie moléculaire, un tournant important a été impulsé par la mise au point de techniques de séquençage de l'ADN en 1977. Il faudra attendre le milieu des années 1980 pour voir apparaître le développement des premiers robots séquenceurs. Dans les mêmes années l'informatique connaîtra de grandes avancées : avènement des micro-ordinateurs personnels et création de langages de programmation évolués (comme le langage C).

Les biologistes s'apercevront rapidement du bénéfice qu'ils pourront tirer de tels outils. Il faut en effet se rappeler que les premières recherches en biologie moléculaires ont été menées avec des moyens très limités : en 1962, la résolution de la structure de la myoglobine a demandé à Max Perutz le traçage manuel de plus de deux mille cercles. La visualisation tridimensionnelle des molécules nécessitait par ailleurs la construction fastidieuse d'imposantes structures à base de tiges métalliques qui encombraient les bureaux des chercheurs.

Un des pionniers de la bioinformatique est très certainement Rodger Staden qui a très vite ressenti l'intérêt de développer des programmes pour analyser les séquences. Dès 1977, il propose ainsi des outils informatiques qui servent encore aujourd'hui (un package très utilisé porte son nom). Le résumé d'un de ces articles, intitulé " **Sequence data handling by computer "** [Staden 1977], est très explicite quant à ces motivations :

The speed of the new DNA sequencing techniques has created a need for computer programs to handle the data produced. This paper describes simple programs designed

specifically for use by people with little or no computer experience. The programs are for use on small computers and provide facilities for storage, editing and analysis of both DNA and amino acid sequences.

2. QUELQUES MOTS SUR LA BIOINFORMATIQUE

- **Définition de la bioinformatique :**

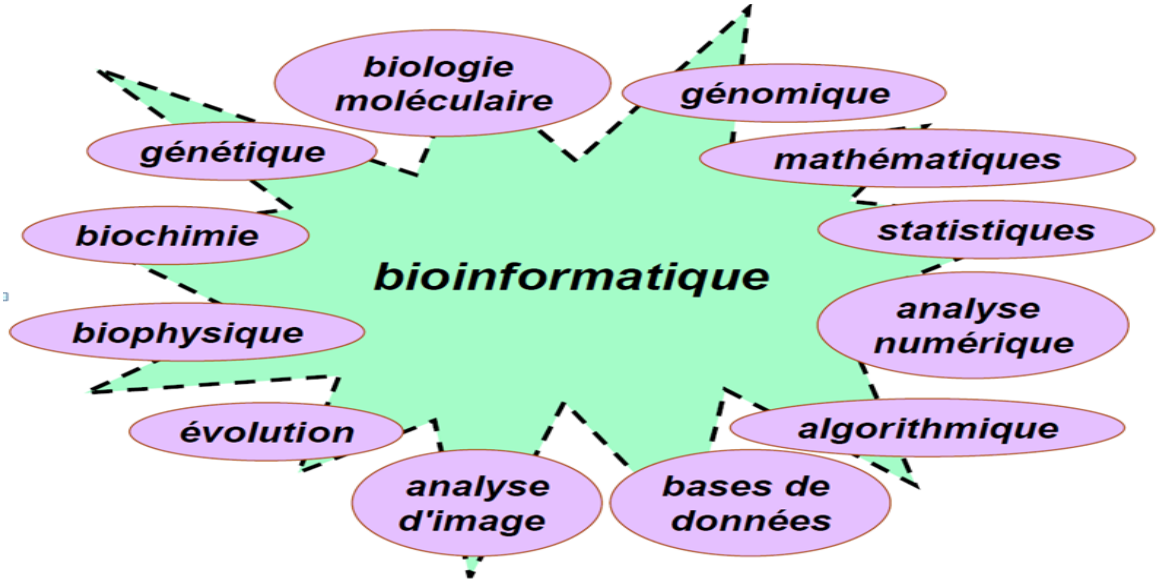


Figure 20: La bioinformatique et les autres domaines.

La bioinformatique est une discipline à l'interface de la biologie et de l'informatique, qui recouvre l'ensemble des technologies et des méthodes permettant de collecter, de stocker, d'analyser et d'interpréter les données biologiques. Elle désigne donc tout au :

- Développement d'infrastructures et d'outils, tels des systèmes de stockage de données, des logiciels de base de données et de visualisation.
- Développement et la mise en œuvre de méthodes mathématiques et informatiques, s'appuyant sur ces outils, pour analyser des données et les interpréter biologiquement.

Ces deux aspects de la discipline sont d'ailleurs souvent désignés sous deux noms différents en anglais, la **« bioinformatics »** couvrant le développement d'infrastructures et d'outils tandis que le **« computational biology »** désigne la mise au point de méthodes d'analyse spécifique et leur utilisation pour traiter un problème biologique particulier. En

schématisant à l'extrême, la **« bioinformatics »** se rapproche plus d'un travail d'ingénierie en Infrastructure et logiciel, pas toujours spécifique d'ailleurs au domaine d'application que sont les sciences du vivant, tandis que la **« computational biology »** s'apparente à une discipline scientifique à part entière, consistant à modéliser et analyser des systèmes biologiques par des modèles informatiques, en empruntant d'ailleurs de nombreuses approches aux mathématiques et à la physique. [25]

- ## Les applications industrielles de la bioinformatique

La bioinformatique dans son ensemble est donc une activité transverse qui peut être appliquée à de nombreux secteurs des sciences de la vie et des biotechnologies confrontés à l'utilisation et l'étude du vivant. Elle joue de ce fait un rôle important et croissant dans de nombreuses industries allant de la recherche biomédicale à l'agro-alimentaire, en passant par l'énergie et l'environnement.

Les entreprises pharmaceutiques et biotechnologiques sont certainement les premières utilisatrices en volume de la bioinformatique. En effet, elles utilisent de plus en plus de technologies à haut débit, comme la protéomique ou le séquençage, pour étudier les systèmes biologiques qui les intéressent. Elles s'appuient naturellement sur les outils et les méthodes de la bioinformatique pour décoder l'information biologique cachée dans les multiples données qu'elles génèrent, et ainsi faciliter la traduction de ces données en avancées médicales. Un domaine particulier au cœur de la révolution en cours est la pharmaco-génomique, qui vise à prédire la probabilité qu'un individu réponde à un traitement en fonction de son patrimoine génétique ou de marqueurs moléculaire, ouvrant ainsi la voie à la médecine personnalisée. Cette discipline s'appuie sur des traitements informatiques et mathématiques précis, nécessitant des statistiques en grande dimension et de la fouille de données, pour identifier les combinaisons de marqueurs permettant de diagnostiquer avec précision une pathologie et de prédire l'efficacité et la toxicité d'un traitement sur un individu donné. Les enjeux sociétaux et économiques de la médecine personnalisés sont considérables, puisqu'ils' agit d'améliorer la sûreté et l'efficacité des traitements en prenant en compte les spécificités moléculaires de chaque individu. [26]

- **Applications classiques en bioinformatique**

Sources de données	**Tâches bioinformatiques**
Séquences nucléiques	Identification des introns et des exons Prédiction de gènes
Séquences protéiques	Algorithmes de comparaison de séquences Alignement multiple Découverte de régions conservées Recherche de motifs
Structures macromoléculaires	Prédiction de structures secondaires Prédiction de structures tertiaires Identification d'interactions moléculaires
Génomes	Analyse de la structure du génome Phylogénie Analyse de liaison génétique
Expression des gènes	Corrélation de l'expression des gènes Recherche de gènes différentiellement exprimés
Articles biomédicaux	Annotation automatique des entités biologiques Bases de connaissances biomédicales à partir de texte
Voies de signalisation métaboliques	Simulation de réseaux

Tableau 1: Applications classiques en bioinformatique[26].

3. Les banques et les bases de données biologiques :

Les bases de données ont pour mission de rendre publiques la masse des données qui ont été déterminées. Par conséquent la quantité de données annotées qu'elles contiennent, le nombre de séquences, dont certaines ne sont plus disponibles ailleurs, leur grande diversité et la qualité des annotations en font des outils indispensables pour la communauté scientifique. Et leur utilisation se trouve au cœur des pratiques en matière de recherche. Voilà pourquoi leur utilisation est nécessaire au progrès scientifique. En biologie on distingue généralement deux types de bases de données biologiques :

- **Les Bases généralistes :**

Ce sont des bases publiques internationales qui recensent des séquences d'ADN ou de protéines déterminées quelque soit l'organisme ou la méthode utilisée pour les acquérir. La quantité de données annotées qu'elles contiennent, le nombre de séquences, dont certaines ne sont plus disponibles ailleurs, leur grande diversité et la qualité des annotations en font des outils indispensables pour la communauté scientifique. Elles présentent cependant quelques défauts qui ont conduit à la création de bases plus spécifiques : Les données qu'elles contiennent n'ont pas toujours été vérifiées, elle sont parfois trop diverses et nombreuses pour être exploitées efficacement et sont parfois mises à jour avec un certain retard.

On retrouve trois principales bases de données généralistes qui sont connues mondialement: la base de données GenBank du NCBI (http://www.ncbi.nlm.nih.gov/), la base de données d'ADN du Japon (DDBJ) (http://www.ddbj.nig.ac.jpl) et la base du laboratoire de biologie moléculaire européen (EMBL) (http://www.ebi.ac.ukJembll). [27]

- **Les Bases spécialisées :**

Ce sont des bases dont les données ont été classées suivant une caractéristique biologique particulière. Il s'agit par exemple des bases recensant toutes les séquences d'un même génome. [24]

L'utilisation des bases spécialisées comme les bases de motifs est devenue un outil essentiel dans l'analyse des séquences pour tenter de déterminer la fonction de protéines inconnues ou savoir à quelle famille appartient une séquence non encore caractérisée.
PDB (Protein Data Bank) et la base Kegg (Kyoto Encyclopedia of Genes and Genomes) sont des bases spécialisées parmi les plus connues.

4. Les Bases de Motifs

L'utilisation de bases spécialisées comme les bases de motifs est devenue un outil essentiel dans l'analyse des séquences pour tenter de déterminer la fonction de protéines inconnues ou savoir à quelle famille appartient une séquence non encore caractérisée.

- **Les bases de motifs nucléiques**

On peut aisément consulter deux bases de motifs nucléiques spécifiques et qui sont régulièrement mises à jour : ce sont les bases de facteurs de transcription **TFD** (Ghosh,

1993), et **TRANSFAC** (Knüppel et al., 1994). Par exemple, la base TFD correspond aux facteurs de transcription des eucaryotes. Elle contient des informations relatives aux motifs mais également des informations qui concernent les séquences protéiques concernées par la transcription comme par exemple les domaines protéiques en liaison avec l'ADN

- **Les bases de motifs protéiques**

PROSITE (http://www.expasy.org/prosite/) et est bases de donnée qui permettent à l'utilisateur de détecter des motifs dans les séquences de protéines.

PROSITE contient des modèles et des profils spécifiques pour plus d'un millier de familles de protéines ou de domaines. Chacun de ces motifs est livré avec la documentation (en format Word!) Fournissant des informations générales sur la structure et la fonction de ces protéines.

Les bases de données biologiques peuvent aussi être classées en bases de données de séquence ou de structure. Bases de données de séquences est applicable aux deux séquences d'acides nucléiques et les séquences de protéines, alors que la base de données de structure est applicable aux protéines seulement.

5. Exemple de bases données biologiques

On citera notamment les bases suivantes qui sont des projets internationaux et constituent des leaders dans le domaine.

EMBL (European Molecular Biology Laboratory) , **GenBank** ou encore **Swissprot** au rang des bases généralistes et la base de structure **PDB** (Protein Data Bank) et la base **Kegg** (Kyoto Encyclopedia of Genes and Genomes) comme bases spécialisées parmi les plus connues.

- **EMBL (European Molecular Biology Laboratory)** : constitue la principale source de séquences de nucléotides en Europe. La production des séquences se fait en collaboration internationale avec GenBank, des états unis, et DNA Databases, du Japon. Chacune des trois bases de données récolte des séquences et échange avec les autres les données et mises à jour.

EMBL évolue de manière exponentielle. La figure **21** montre son évolution depuis 1982 jusqu'à 2010.

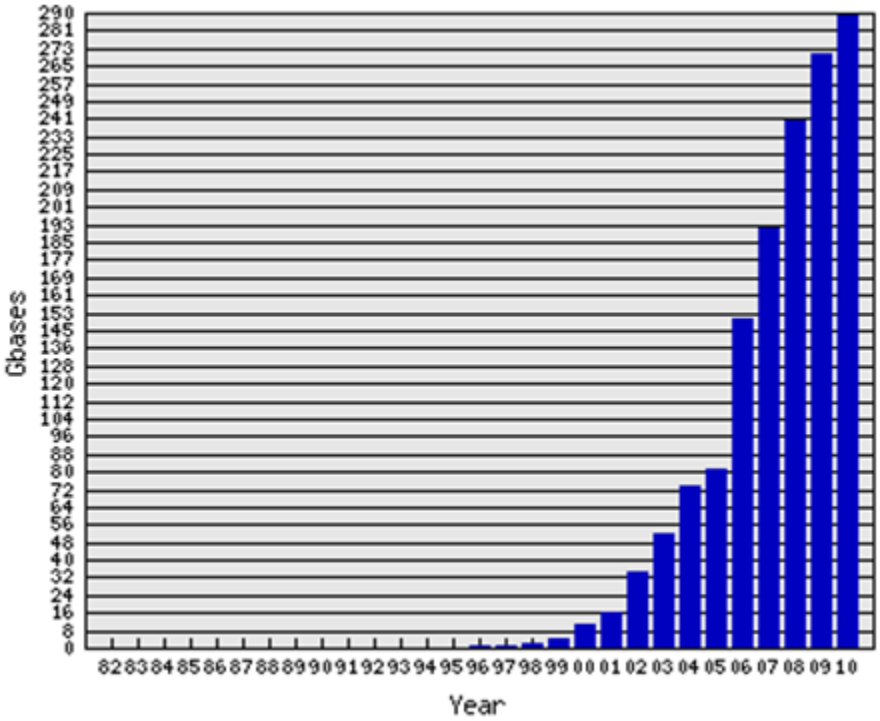


Figure 21: Evolution de la base de données EMBL entre 1982 et 2010 [27].

- **GenBank** : La GenBank est une banque de nucléotide américaine. Toutes les séquences de nucléotides disponibles ainsi que leur traduction en protéines sont en libre accès. Cette base de données a été créée au centre national pour l'information biotechnologique (NCBI) dans le cadre de la collaboration internationale sur le séquençage des nucléotides.

  La GenBank et ses collaborateurs reçoivent des séquences produites dans des laboratoires du monde entier à partir de plus de 100 000 organismes différents. La GenBank continue de grossir à un taux exponentiel doublant de taille tous les dix mois. La figure **22** illustre cette évolution. [28]

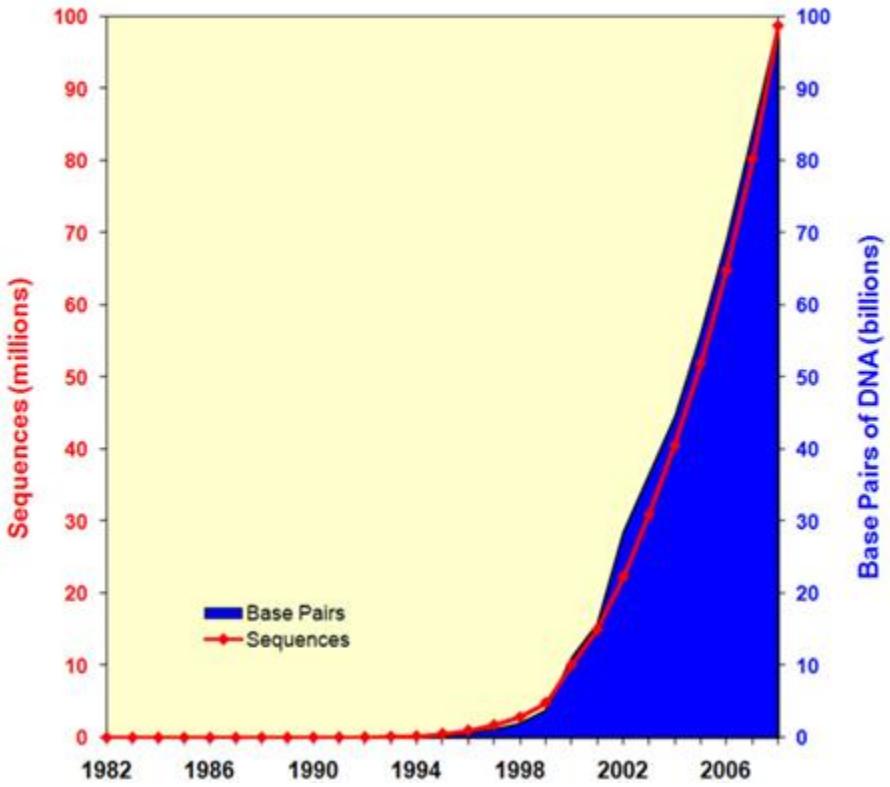


Figure 22: Evolution de la base de données GenBank entre 1982 et 2008 [29].

- **DDBJ** (DNA Data Bank of Japan) : maintenue par le Centre d'Information Biologique de l'Institut National de Génétique, Créée en 1986 et diffusée par le NIG (National Institute of Genetics Japon)

Dès le début des années quatre-vingt, EMBL et GenBank ont travaillé en collaboration, à laquelle s'est associée DDBJ lors de sa création. Les termes de cette collaboration internationale, officiellement formalisée depuis 1988, impliquent notamment un échange quotidien des données collectées, l'application de règles communes relatives à la nomenclature et aux formats utilisés, et le fait qu'une entrée ne peut être modifiée que par celui des trois centres auquel elle avait été initialement soumise (pour éviter les « conflits » dans les mises à jour).

- **PIR-NBRF** : Une base de données protéique créée en 1984 par la NBRF (National Biomedical Research Foundation). Elle est maintenant un ensemble de données issues du MIPS (Martinsried Institute for Protein Sequences, Munich, Allemagne) et de

la banque japonaise JIPID (Japan International Protein Information Database)

- **SwissProt** : créée en 1986 par Amos Bairoch à l'Université de Genève, est actuellement développée en collaboration entre l'Institut Suisse de BioInformatique (ISB-SIB et l'EBI). C'est certainement la banque de séquences protéiques la moins redondante, la mieux annotée, et la plus intégrée, via des références croisées, aux autres banques de données. Les séquences sont soumises directement à SwissProt par leurs auteurs, ou (majoritairement maintenant) extraites des CDS de la banque nucléotidique EMBL.

6. Différents champs liés à la bioinformatique

• La biologie computationnelle

La biologie computationnelle concerne les travaux à l'interface entre l'informatique et les sciences de la vie. Les spécialistes des sciences de la vie étudient des systèmes extrêmement complexes, comme les cellules vivantes, le cerveau humain ou l'évolution de la vie sur notre planète.

• Génomique

La génomique est une nouvelle discipline de la biologie qui vise à l'analyse moléculaire et physiologique complète du matériel héréditaire des organismes vivants.il s'agit de déduire les fonctions des gènes et leurs interactions à partir de leurs séquences, ce qui facilite l'intégration de la génomique dans la physiologie.[28]

• La protéomique

La protéomique étudie le protéome , ensemble des protéines identifiées à partir d'un génome. Toutes les cellules de l'organisme possèdent le même génome, mais ont un protéome différent selon l'organe et le moment du développement de l'individu. La protéomique s'attache à déterminer la localisation, la structure et la fonction de ces protéines. Elle analyse leurs interactions et leurs modifications au cours du temps.[29]

- ## La pharmacogénomique et la pharmacogénétique

Il y a un lien étroit entre la pharmacogénétique et la pharmacogénomique bien qu'il s'agisse de deux disciplines distinctes.

La pharmacogénétique se concentre sur l'étude des différences de métabolisme entre les individus ainsi que sur les capacités d'absorption et d'élimination des médicaments, héritées par ceux-ci. L'analyse génétique se fait grâce à l'étude des expressions phénotypiques, en particulier des différences interindividuelles dans l'équipement enzymatique et de leurs conséquences sur le métabolisme de l'organisme. Selon l'Agence européenne pour l'évaluation des produits médicamenteux (EMEA), la pharmacogénétique peut être définie comme « l'étude des variations interindividuelles dans les séquences d'ADN en relation avec les réponses aux médicaments.» [30]

La pharmaco-génomique va plus loin : grâce au développement des techniques de génétique moléculaire, elle s'applique au gène lui-même et non plus seulement à son expression. Elle englobe la pharmacogénétique et la renouvelle en identifiant les variations du génome responsable des modifications des réponses de l'organisme [31]. La pharmaco-génomique se veut donc l'étude de la totalité des gènes impliqués dans les réactions aux médicaments [32]. La pharmaco-génomique fait aussi référence à l'utilisation de la génomique dans la recherche de nouvelles cibles thérapeutiques [33].

- ## Pharmaco-informatique

Pharmaco-informatique se concentre sur les aspects de la bioinformatique portant sur la découverte de médicaments.

- ## La génomique structurale ou la bioinformatique structurale

L'ensemble des protéines codées sur le génome peut être considéré comme une collection des repliements 3D suffisants pour assurer les principales fonctions cellulaires, comme le métabolisme, la réplication ou la gestion de l'information. Le terme "génomique structurale" a donc été utilisé pour désigner les études de l'ensemble de protéines des génomes, i.e. l'étude du protéome du point de vue de la structure tridimensionnelle.[34]

- **Génomique fonctionnelle (post-génomique)**

La génomique fonctionnelle ou post-génomique étudie le transcriptome (ensemble des ARN messagers transcrits à partir du génome). Le but est de déterminer la fonction des gènes à partir de leurs produits d'expression (ARN et protéines) et d'étudier leur mode de régulation et leurs interactions. [29]

- **La génomique comparative:**

La génomique comparative permet l'étude de l'évolution et de la dynamique des génomes. En effet, l'analyse comparée de chromosomes de multiples espèces renseigne sur les segments conservés entre les espèces au cours de l'évolution, l'ordre préservé des séquences fonctionnelles au sein de ces segments et enfin les mécanismes de réarrangements chromosomiques intra ou inter espèces.[35]

- **Biomédicale informatique / informatique médicale**

C'est l'application des technologies issues de l'information au domaine médical. Avec le développement de nouvelles technologies médicales reposant sur l'enregistrement d'informations complexes (en particulier dans le domaine de l'imagerie médicale) sont apparus de nouveaux besoins : compatibilité des formats de données entre les outils et entre les sites, sureté face aux erreurs de transcriptions, spécifications complètes des paramètres d'enregistrements, anonymat des informations. L'informatique médicale consiste donc à développer des outils matériels et logiciels qui acceptent ces multiples contraintes.[36]

II. Recherche et découverte de motifs:

1. Les motifs biologiques :

Un "motif" (ou « pattern » en Anglais) est un segment court dans une séquence, il est continu et non ambigu. Il peut représenter une structure plus complexe lorsque lui-même est composé de différents "motifs" qui peuvent être plus ou moins éloignés les uns des autres et sa définition peut comporter des exclusions ou des associations de "motifs".

Un motif au sens bioinformatique du terme représente une expression qui permet de caractériser un ensemble de séquences d'ADN, d'ARN ou de protéines. Le motif peut concerner les structures primaires, secondaires et tertiaires.

- **Motifs ADN:**

Généralement, ils présentent des séquences caractéristiques des sites de liaison pour des protéines telles que des nucléases, sites de fixation de facteurs de transcription, ou enzyme de restriction. D'autres sont impliqués dans des opérations importantes au niveau de l'ARN, y compris la liaison au ribosome, traitement de l'ARNm (Connectivité, polyadénylation) et terminaison de la transcription.

- **Motif protéine :**

Ils peuvent être chargés des sites d'interaction protéine-protéine ou ADN-protéines, il possède plusieurs séquences susceptibles de constituer des signaux de localisation nucléaire (NLS) ou ils peuvent être en les domaines enzymatiques.
Certains motifs sont précis et non ambigus (comme les codons stop : UGA, UAA, UAG) d'autres peuvent être beaucoup plus flous et complexes (comme les motifs consensus liés à des familles de protéines ou les sites de fixations de facteurs de transcription).

Il existe plusieurs raisons de rechercher des motifs à travers les séquences car ils sont généralement impliqués dans des systèmes de régulation ou définissent des fonctions biologiques. Parmi ces raisons, on peut citer la détermination de la fonction d'une nouvelle séquence (par exemple en localisant un ou plusieurs motifs répertoriés dans des bases de motifs), l'identification dans une séquence nucléique de régions codantes (par exemple en repérant les codons d'initiation et de terminaison, les sites d'épissages et les zones de fixation des ribosomes), la recherche d'un motif particulier dans une séquence (par exemple en identifiant sur une séquence les sites de coupures d'enzymes de restriction ou des promoteurs spécifiques etc...), ou bien l'extraction à partir des banques de données (par exemple extraire des séquences possédant le même signal de régulation ou la même signature protéique pour effectuer des études comparatives ultérieures).

2. Découverte de Motifs vs. Recherche de Motifs

Il ya deux tâches fondamentalement différentes relatives à l'identification de nouveaux modèles dans les séquences biologiques. La première est appelée recherche de Motifs. Il s'agit de trouver de nouvelles occurrences d'un motif connu. Beaucoup de séquences consensus sont connus en biologie et il est important d'avoir des outils qui permettront à quiconque de trouver des occurrences de motifs connus dans de nouvelles séquences.

Il existe des outils logiciels spécialisés pour le filtrage. Certains sont tout à fait général, c'est à dire, ils permettent à l'utilisateur de spécifier un modèle dans le cadre de l'entrée dans une forme spécifique. D'autres sont conçus pour reconnaître un seul modèle. De nombreux outils spécialisés sont disponibles pour la reconnaissance des signaux d'épissage, des éléments de régulation différents, et les éléments structuraux particuliers.

Les auteurs de tels outils spécialisés affiner les paramètres du système pour augmenter la précision de la prédiction.

Bien que la tâche de recherche de Motifs est très important, nous nous concentrerons sur un autre type de problème lié modèle, appelé la découverte de motifs. La tâche est d'identifier un nouveau modèle dans un ensemble de plusieurs séquences.

L’entrée pour les programmes de découverte de motifs se compose de plusieurs séquences, attendions pour contenir le motif. Des séquences d'entrées sont typiquement liées d'une manière quelconque, par exemple, ils sont des membres de la même famille de protéines, séquences fonctionnellement liées, ou les régions en amont de gènes co-régulée. [37]

3. Problème de découverte de motifs

Le problème de découverte de motifs consiste à trouver des modèles représentés dans une collection de séquences. Il est l'un des problèmes d'analyse de séquences classiques, mais n'a toujours pas été résolu de façon satisfaisante, d'une manière exacte et efficace.

Ceci est partiellement dû au grand nombre de possibilités de définir le motif, l’espace de recherche, et la notion de sur représentation. Même pour les formalisations bien définis, le problème est souvent résolu d'une manière ad-hoc avec des heuristiques qui ne garantissent pas à trouver le meilleur motif. [37]

4. Objectifs de découverte de motifs

Le but de découverte de motifs est d'identifier un motif inconnu dans un ensemble donné de séquences. Il ya un grand nombre de modèles potentiels, et il est souvent difficile de décider lequel d'entre eux sont les plus prometteuses.

Définir le «meilleur» modèle dépend de l'utilisation prévue du motif. [38]

5. Représentations de motifs biologiques :

5.1 Alignement de séquences

La ressemblance de séquences peut également être étudiée au niveau de la séquence entière, et non pas au niveau de quelques motifs. Dans le cas de la recherche ou de l'extraction de motifs, les comparaisons se font localement sur les séquences. Il est question de ressemblance (ou similitude) locale. Dans le cas de l'étude, non plus de la présence d'un motif biologique particulier, mais d'une structure globale commune à plusieurs séquences, il est alors question de ressemblance globale. La majorité des méthodes recherchant des similitudes globales entre deux ou plusieurs séquences est basée sur un mécanisme d'alignement des séquences.

L'alignement se lit en deux dimensions. Les lignes correspondent aux séquences, tandis que les colonnes correspondent aux positions dans chacune des séquences. Dans une même colonne, deux acides nucléiques ou aminés qui ne se correspondent pas sont alors des substitutions. Selon les choix de représentations, il est possible ou non d'insérer un « gap » dans la colonne, représentant alors une suppression d'un acide dans la séquence (ou une insertion dans les séquences ne contenant pas de gaps). La lecture de l'alignement vise à mettre en évidence les parties communes à toutes les séquences.

À l'instar des problèmes de la recherche et de l'extraction de motifs, la principale difficulté consiste à modéliser la notion de ressemblance globale, puis à délimiter l'espace des possibles. Ce problème est néanmoins très proche à plusieurs égards du problème précédent, et les modélisations de la ressemblance sont souvent les mêmes. [39**]**

Cela permet de connaître pour chaque base la variabilité à chaque position. L'alignement de ces séquences peut servir à produire une séquence consensus, une table de fréquences ou une matrice de pondération des éléments qui composent le motif.

5.2 Le consensus et les expressions régulières

La séquence consensus représenterait la séquence idéale, une séquence consensus peut être définie comme une séquence des bases qui apparaissent avec la plus grande fréquence sur chaque position lorsque l'on compare des séries de séquences présumées avoir la même fonction. [40]

Celle-ci est construite à partir de l'alphabet IUPAC ou en retenant une seule base, la plus fréquente, pour chacune des positions de la séquence (Figure 23):

These are the official IUPAC-IUB single-letter base codes

Code	Base Description	
G	Guanine	
A	Adenine	
T	Thymine	
C	Cytosine	
R	Purine	(A or G)
Y	Pyrimidine	(C or T or U)
M	Amino	(A or C)
K	Ketone	(G or T)
S	Strong interaction	(C or G)
W	Weak interaction	(A or T)
H	Not-G	(A or C or T)
B	Not-A	(C or G or T)
V	Not-T (not-U)	(A or C or G)
D	Not-C	(A or G or T)
N	Any	(A or C or G or T)

Figure 23: Alphabets IUPAC.

L'écriture d'une séquence consensus peut également, s'effectuer de manière plus précise par rapport au code IUPAC, en utilisant des expressions régulières les "conventions symboliques" ou la grammaire **Prosite**.

La propriété fondamentale des motifs définis par des expressions régulières, autrement dit, si l'on peut représenter un motif biologique d'intérêt par une expression régulière.

Ce type de définition par expression régulière se généralise toute à fait directement aux séquences de protéines. La banque de motifs PROSITE utilise d'ailleurs une syntaxe par expression régulière pour définir ainsi plus d'un millier de motifs ou de signatures protéiques.

Un résumé de la grammaire PROSITE est illustré dans le tableau ci-dessous:

A	Une seule lettre représente la base biologique, dans notre exemple l'Adénine représentée par un A.
X	X signifie n'importe quelle base biologique.
[ATC]	Une liste représente la possibilité de trouver une des bases qui composent la liste pour une position, pour cet exemple : A, T ou C
{T}	Liste d'exclusion : pas de T à cette position
[AT](2)	pour cet exemple : deux A ou T
x(0,1)	Entre 0 et 1 base quelconque.
<C	pour cet exemple : C au début de la séquence
T>	pour cet exemple : T à la fin de la séquence
-	Élément séparateur du motif

Tableau 2: Expressions régulières.

Exemple: <[GT]-C-x(3)-A-T>, ce motifs signifie qu'en première position il y a un G ou un T, puis un C puis 3 bases inconnues puis A puis T en dernière position, Cette écriture signifie que le motif engendré est le suivant : **G**-C-X-X-X-A-T OU **T**-G-X-X-X-A-T

Ainsi, les expressions régulières permettent de visualiser uniquement les bases possibles à chaque position dans le motif et ne prennent pas en compte les statistiques représentant la variation des bases à chaque position.

5.3 Les matrices de pondération

Une autre manière de représenter un motif est de construire une matrice. La forme de base d'une matrice est la matrice de comptage ou matrice d'alignement (Figure 24). Il s'agit simplement de représenter dans un tableau à deux dimensions le nombre de fois que chaque résidu apparaît à chaque position. Chaque colonne représente une position et chaque rangée représente le nombre d'occurrences d'un nucléotide et les cellules indiquer la fréquence de chaque résidu dans chaque position de l'alignement multiple.

Capturer la fréquence de chaque lettre à chaque position dans le modèle afin qu'on puisse dire à quel point un site potentiel correspond au modèle.

Il existe trois types de matrices.

1. Positions Count Matrice (PCM) : Compte entières à chaque position décrivent le nombre de fois où chaque nucléotide a été observée.
2. Positions Frequency Matrice (PFM) : Chaque colonne résume à 1,0, avec la fréquence fractionnaire de chaque base représentée.
3. Positions Weights Matrice (PWM) : dans lequel un modèle d'arrière-plan (de la distribution des nucléotides) pondère les fréquences observées.[41]

Exemple : Etant donnée les séquences suivantes

positions 1 2 3 4 5 6 7 8 9 10

GGGACTTT C C
GGGGATTT C C
GGGGTTTC C C
GGGAATCT C C
GGGAGATT C C
GGGGATTC C C
GGGGAAGC C C
GGGACTTC C C

pos	1	2	3	4	5	6	7	8	9	10
A	0	0	0	4	4	2	0	0	0	0
C	0	0	0	0	2	0	1	4	8	8
G	8	8	8	4	1	0	1	0	0	0
T	0	0	0	0	1	6	6	4	0	0
sum	8	8	8	8	8	8	8	8	8	8

Figure 24: Position Count Matrix (PCM).

A partir de la matrice de comptage, nous pouvons calculer la fréquence relative (équation 1) de chaque nucléotide à chaque position dans le site de fixation (Figure 25). La somme des fréquences dans chaque case vaut 1. On peut aussi calculer les fréquences corrigées des nucléotides (équation 2). L'intérêt des fréquences corrigées réside dans le fait qu'elles permettent de considérer les cas non encore observés. Le fait de ne pas trouver un nucléotide à une position d'un alignement de quelques sites ne veut pas dire qu'avec plus de données on ne trouvera pas ce nucléotide, par exemple sur la matrice de la Figure 24 le fait de ne pas observer A, C ou T dans les huit séquences alignées n'exclut pas un de ces nucléotides si l'on pouvait disposer d'une centaine de séquences. Le terme *k* de l'équation 2 représente le pseudo-poids. Il est fixé par l'utilisateur. Sa valeur est souvent élevée quand on dispose de peu de séquences et faible quand il y a plus de séquences disponibles. En outre, les fréquences corrigées permettent d'éviter des valeurs nulles dans les matrices, ce qui rend possible certains calculs tels que les logarithmes et certaines probabilités.[41]

$$f_{i,j} = \frac{n_{i,j}}{\sum_{i=1}^{A} n_{i,j}}$$

avec : fi,j = fréquence relative du résidu i à la position j ;
ni,j = nombre d'occurrences du résidu i à la position j ;
A = taille de l'alphabet de nucléotides (4).

Équation 1: calcul de la PFM

$$f'_{i,j} = \frac{n_{i,j} + p_i k}{\sum_{i=1}^{A} n_{i,j} + k}$$

avec : $f'i,j$ = fréquence corrigée du résidu i à la position j ;
ni,j = nombre d'occurrences du résidu i à la position j ;
A = taille de l'alphabet de nucléotides (4);
k = pseudo poids ;
p_i = probabilité à priori du résidu i.

Équation 2 : calcul de la fréquence corrigée

$$W_{i,j} = \ln\left(\frac{f'_{i,j}}{p_i}\right)$$

Équation 3: calcul Positions Weights Matrice (PWM) .

L'avantage de la représentation sous forme de matrice est que, contrairement au consensus, on ne perd pas l'information sur les fréquences à chaque position.

pos	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
A	0.00	0.00	0.00	0.50	0.50	0.25	0.00	0.00	0.00	0.00
C	0.00	0.00	0.00	0.00	0.25	0.00	0.125	0.50	1.00	1.00
G	1.00	1.00	1.00	0.50	0.125	0.00	0.125	0.00	0.00	0.00
T	0.00	0.00	0.00	0.00	0.125	0.75	0.75	0.50	0.00	0.00
sum	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Figure 25: Positions Frequency Matrice (PFM).

pos	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
A	0.0278	0.0278	0.0278	0.4722	0.4722	0.2500	0.0278	0.0278	0.0278	0.0278
C	0.0278	0.0278	0.0278	0.0278	0.2500	0.0278	0.1389	0.4722	0.9167	0.9167
G	0.9167	0.9167	0.9167	0.4722	0.1389	0.0278	0.1389	0.0278	0.0278	0.0278
T	0.0278	0.0278	0.0278	0.0278	0.1389	0.6944	0.6944	0.4722	0.0278	0.0278
sum	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Figure 26: matrice de fréquences corrigées

5.4 Les Modèles de Markov cachés (HMMs) :

Les HMMs, introduits par Baum Welch dans les années 60, utilisés a partir des années80 en reconnaissance de la parole, la où ils ont été pleinement exploités, appliqués ensuite à la reconnaissance de texte manuscrit, au traitement d'images, et a la Bioinformatique comme le séquencement de l'ADN.

Les profiles HMM sont des modèles statistiques (comme les profiles) d'alignements multiples de séquences. Ils tiennent compte de toutes les informations concernant la conservation de chaque position dans chaque colonne d'un alignement en assignant une probabilité représentant les résidus préférés par position, ainsi que les insertions-délétions.

Ce modèle est souvent utilisé pour représenter les modules de régulation : les groupes de motifs qui interagissent. En effet, les modules de cis-régulation impliquent souvent des clusters de sites de fixation pour un ou plusieurs facteurs de transcription.

Un modèle de Markov caché est un cadre mathématique qui modélise une série d'observations fondées sur une hypothétique, mais sous-jacente processus caché. Le modèle se compose d'un ensemble d'états et les transitions entre ces états. Chaque état émet un signal basé sur un ensemble de probabilités d'émission et les transitions de manière stochastique à un autre Etat, fondé sur un ensemble de probabilités de transition. Ces deux distributions de probabilité, lorsqu'il est combiné avec la distribution de l'état initial, caractériser entièrement un HMM.

Un tutoriel de HMM utile a été écrit par Rabiner (Rabiner, 1995), et des informations plus détaillées sont disponibles dans (Rabiner et Juang, 1993). En biologie computationnelle, un modèle HMM une famille de séquences apparentées. Ainsi, trois problèmes de Rabiner correspondent à :

- déterminer si une séquence donnée appartient à la famille modélisée.
- la recherche d'un alignement de la séquence donnée dans le reste de la famille.
- la formation du modèle sur la base de membres connus de la famille.

Lorsqu'on a un modèle de Markov caché correspondant à un ensemble de séquences, il y a trois problèmes d'intérêts :

- **Le problème d'évaluation**: Étant donné les paramètres d'un HMM particulier et une séquence d'évènements, quel est la probabilité que cette séquence ait été générée par ce modèle. Ce problème est résolu par l'algorithme « [43]».
- **Le problème de décodage** : Étant donné les paramètres d'un HMM particulier et une séquence d'évènements, quel est la séquence d'états cachés la plus probable, pouvant générer cette séquence. Ce problème est résolu par l'algorithme « Viterbi [44] ».
- **Le problème d'apprentissage** : Étant donné les paramètres d'un HMM particulier et une séquence d'évènements, quel ajustement devons-nous faire aux probabilités d'émissions d'évènements et de transitions d'états pour que le modèle corresponde le plus possible à la dite séquence d'évènements. Ce problème est résolu par l'algorithme « Baum-Welch algorithm [43]».

Avantages	Inconvénients
- Base mathématique solide pour comprendre son fonctionnement. - Variabilité de la forme. - Alignement temporel incorporé systématiquement. - Reconnaissance réalisée par un simple calcul de probabilité cumulée. - Décision globale sans obligation d'utiliser de seuils. - Séparation franche entre données et algorithmes.	- Le choix à priori de la typologie des modèles (nombre d'état, transitions autorisées et règle de transition). - Dégradation des performances si l'apprentissage n'est pas suffisant.

Tableau 3: Les avantages et les inconvénients des HMMs.

6. Les algorithmes de découverte de motifs

La découverte de motifs est généralement utilisée pour détecter des motifs exceptionnels présents dans des jeux de séquences fonctionnellement reliées (ex. promoteurs

de gènes co-exprimés, collection de sites identifiés expérimentalement). Il existe deux grands types d'approches: les méthodes heuristiques basées des techniques d'optimisation des matrices et les méthodes énumératives basées sur le comptage de mots.

- **Méthodes heuristiques**

Un premier groupe de méthodes optimise un score (contenu informationnel ou autre) pour une série de paramètres définis par l'utilisateur (longueur de matrice, nombre de sites à aligner).

Pour chaque motif généré, le score est calculé et les motifs ayant les meilleurs scores sont retenus. Le nombre de matrices possibles dépend du nombre et de la taille des séquences en entrée, de la taille du motif recherché et du nombre et de la répartition attendus des sites.

Le nombre de matrices possibles augmente de façon exponentielle en fonction du nombre de séquences si l'on cherche exactement un site par séquence (par exemple pour les collections de sites), mais ce nombre devient encore plus énorme si l'on estime qu'une séquence peut contenir zéro ou plusieurs sites (ex. séquences non-codantes de gènes co-exprimés). En effet, cela revient à chercher toutes les combinaisons possibles d'une longueur données (taille de la matrice) parmi toutes les positions d'un jeu de séquences. Afin de pallier ce problème, les algorithmes basés sur les matrices ne testent qu'une partie des possibilités et retournent le meilleur score obtenu.[42]

- **Les méthodes énumératives**

Le principe de ces méthodes et de compter les occurrences de chaque oligo-nucléotide d'une taille donnée et de détecter ceux qui ont une fréquence exceptionnellement élevée (ou faible, selon les options choisies).

Les mots les plus fréquents ne sont pas forcément les plus pertinents puisqu'ils reflètent les biais de composition des séquences analysées (par exemple une richesse générale en A+T). Afin de détecter les mots surreprésentés, il faut comparer les fréquences obtenues à des fréquences attendues au hasard.[42]

7. Outils d'analyses de séquences

- **AlignACE (Aligns Nucleic Acid Conserved Elements):** http://atlas.med.harvard.edu/ est un programme qui utilise la reconnaissance de formes pour rechercher des éléments conservés dans un ensemble de séquences d'ADN.
- **MEME (Multiple Em for Motif Elicitation) :** http://meme.sdsc.edu/meme/intro.html est un outil permettant de découvrir des motifs dans un groupe de séquences d'ADN ou de protéines apparentées.
- **Weeder :** http://159.149.109.9/weederaddons/pvalue.html Il fait parti d'un ensemble d'outils de découverte de motifs dans des séquences nucléiques.
- **QuickScore :** http://algo.inria.fr/dolley/QuickScore/ Bas é sur un algorithme d'une recherche exhaustive qui estimes la probabilité de rare ou fréquent mot dans un texte génomique. Algorithmes statistiques reposent sur l'hypothèse de base que les nucléotides dans le génome d'un organisme sont générés aléatoirement selon un modèle de probabilité.
- **SeSiMCMC : (The Sequence Similarities by Markov Chain Monte Carlo)** http://favorov.bioinfolab.net/SeSiMCMC/ algorithme qui trouve des motifs d'ADN de longueur inconnue et la structure complexe, tels que des répétitions directes ou des palindromes avec entretoises variables dans le milieu en un ensemble de séquences d'ADN non alignées.
- **YMF (Yeast Motif Finding)** : Utilise un algorithme de recherche exhaustive pour trouver des motifs avec un z-scores élevé.
- **ANN-Spec (Artificial Neural Network Specificity):** Un procédé pour découvrir des sites ayant une meilleure spécificité de liaison du facteur de transcription en utilisant une matrice de poids.

- **Consensus :** http://stormo.wustl.edu/consensus/ Ce programme détermine le motif consensus dans des séquences non alignées. L'algorithme est basé sur une représentation matricielle d'un motif de consensus.
- **GLAM (Gapless Local Alignment of Multiple sequences):** http://zlab.bu.edu/glam/ est un programme de découverte de motifs fonctionnels communs à un ensemble de séquences de nucléotides. GLAM tente de trouver ces motifs par l'obtention de la meilleure possibilité sans vide (gaps), d'alignement multiple de segments de la séquence.

8. Conclusion

Dans ce chapitre, nous avons présenté les éléments de base de la biologie moléculaire et aussi présenté le domaine de la bioinformatique et Représenté les motifs biologiques, le développement des technologies sous-jacentes, et aussi le projet du génome humain. Nous constatons la survenue de nouvelles données biologiques, leur résolution à donner naissance à une discipline nouvelle qu'est la bioinformatique.

Chapitre III

Hybridation De Méthodes

I. Introduction

Une méthode de recherche est rarement aussi efficace pour exploiter que pour explorer l'espace de recherche. La solution est d'associer à une méthode dont la capacité d'exploration est très élevée à une méthode caractérisée par une bonne exploitation de l'espace de recherche, d'où l'émergence actuelle de méthodes hybrides, qui s'efforcent de tirer parti des avantages spécifiques d'approches différentes en les combinant à différents niveaux. Les méthodes hybrides ont rapidement gagné du terrain en réussissant à produire les meilleurs résultats pour de nombreux problèmes.

II. Méthodes hybrides

1. Définition

Une méthode hybride est une méthode de recherche constituée d'au moins de deux méthodes de recherche distinctes. Elle consiste à exploiter les avantages respectifs de plusieurs méthodes en combinant leurs algorithmes suivant une approche synergétique.

Une méthode hybride peut être mauvaise ou bonne selon le choix et les rôles de ses composants. Pour définir une méthode hybride efficace, il faut savoir caractériser les avantages et les limites de chaque méthode. Par exemple, les algorithmes génétiques sont très performants lorsqu'il s'agit d'explorer l'espace de recherche, mais ils s'avèrent ensuite incapable d'exploiter efficacement la zone vers laquelle la population des solutions converge.

Il est alors plus intéressant d'utiliser dans ce stade une autre méthode permettant une bonne exploitation comme par exemple le recuit simulé ou une autre heuristique d'amélioration. Il faut souligner qu'il faut être prudent sur le choix des méthodes à hybrider ainsi sur le problème de multiplication des paramètres.

Actuellement, les approches hybrides gagnent en popularité car ce type d'algorithme produit généralement les meilleurs résultats pour plusieurs problèmes d'optimisation combinatoire [45]. Les approches hybrides ont permis d'obtenir de bons résultats dans une grande variété de problèmes théoriques d'optimisation combinatoire tels le problème du voyageur de commerce, le problème de coloration de graphe, le problème d'affectation quadratique, le problème de tournée de véhicules, le séquençage d'ADN ou encore le calcul des trajectoires des satellites.

Les premières approches hybrides proposées ont combiné des métaheuristiques à base de populations (algorithmes génétiques) avec des métaheuristiques à solution unique (recuit simulé ou recherche tabou) [46].

Plusieurs approches hybrides combinant les métaheuristiques et les méthodes exactes. Etant donné que les méthodes exactes se limitent généralement à de petites instances pour les problèmes d'optimisation combinatoire difficiles, tel que décrit précédemment, l'hybridation de métaheuristiques avec les méthodes exactes peut devenir une alternative très intéressante car les deux méthodes ont des particularités bien différentes qui peuvent être associées pour produire de meilleurs résultats [45]+[46].

2. Motivation de l'hybridation

Malgré le succès des métaheuristiques pour résoudre les problèmes complexes, plusieurs problèmes ont été rencontrés, à savoir :

- la présence de bruit lors de l'évaluation des solutions.
- la distinction entre une solution optimale locale et une autres globale est parfois difficile à percevoir.
- le problème de la convergence prématurée
- le problème d'ajustement des paramètres.

L'hybridation a permet d'avoir un compromis entre l'exploration et l'exploitation de l'espace de recherche des solutions. Pour avoir une bonne exploitation, un algorithme est utilisé pour localiser les meilleures régions de l'espace de recherche, un autre est utilisé pour converger vers l'optimum global. L'hybridation est utilisée aussi pour optimiser les paramètres généraux. Par exemple un algorithme génétique est utilisé pour trouver les meilleurs paramètres d'un algorithme d'optimisation par colonie de fourmis.

On peut conclure que l'hybridation de métaheuristiques est la voie la plus prometteuse pour l'amélioration de la qualité des solutions dans beaucoup d'applications réelles. Ainsi, Le choix d'une approche hybride devient aujourd'hui déterminant pour obtenir de meilleures performances lors de la résolution des problèmes complexes.

3. Exemples d'hybridation

Hybridation 1 : Algorithmes de colonie de fourmis ACO + Recherche locale (*ex. hillclimbing)*

Principe :

1. exécuter une itération de l'algorithme ACO pour *m* fourmis
2. exécuter une heuristique de recherché locale pour chacune des *m* fourmis
3. continuer jusqu'à critère d'arrêt vérifié, sinon aller à 1

Hybridation 2 : Algorithme évolutionnaire EA + Recherche locale (*ex. hillclimbing)*

Principe :

- le même principe précédent (appelé algorithmes mimétique *Memetic Algorithms*).

Hybridation 3 : **d'autres approches hybrides**

Principe

1. exécuter une méthode standard (ex EA).
2. appliquer une méthode de recherché locale pour améliorer la solution obtenue.

4. Classification des stratégies d'hybridation

L'hybridation peut prendre place à deux niveaux : soit plusieurs méthodes de recherche coopèrent au sein d'une méthode hybride, soit une heuristique (ou une méthode de recherche) est insérée à l'intérieur d'une autre méthode de recherche. Dans ce dernier cas, il est nécessaire de définir à quel niveau, c'est-à-dire dans quel composant de la méthode de recherche va prendre place l'hybridation. Plusieurs classifications ont été proposées dans la littérature, les plus importantes sont décrites ci-dessus.

4.1 Classification 1 :

Une classification des techniques d'hybridation en s'appuyant sur les travaux du groupe PERFORM (Parallel gEnetic algoRithms FOR optiMization) du laboratoire LIFL de Lille. Les approches hybrides sont classifiées en trois catégories, selon leurs architectures :

- hybridation séquentielle
- hybridation parallèle synchrone
- hybridation parallèle asynchrone [47]

- **Hybridation séquentielle** : c'est le type d'hybridation le plus populaire. Elle consiste à appliquer plusieurs méthodes de telle manière que le (ou les) résultat(s) d'une méthode serve(nt) de solution(s) initiale(s) à la suivante. Par exemple utiliser l'algorithme de recuit simulé pour améliorer les solutions obtenues par un algorithme génétique. C'est la technique d'hybridation la plus simple (figure 27). [47]

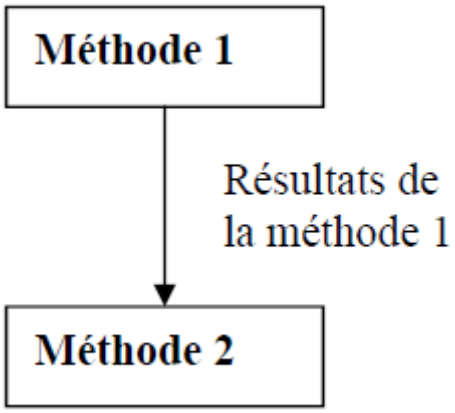


Figure 27: Hybridation séquentielle.

- **Hybridation parallèle synchrone** : cette hybridation est réalisée par incorporation d'une méthode de recherche particulière dans un opérateur. Elle est plus complexe à mettre en œuvre que la précédente. L'objectif est de combiner une recherche locale avec une recherche globale dans le but d'améliorer la convergence. Par exemple une recherche tabou est utilisée pour générer des solutions initiales pour un algorithme d'optimisation par colonies de fourmis (figure 28) [47].

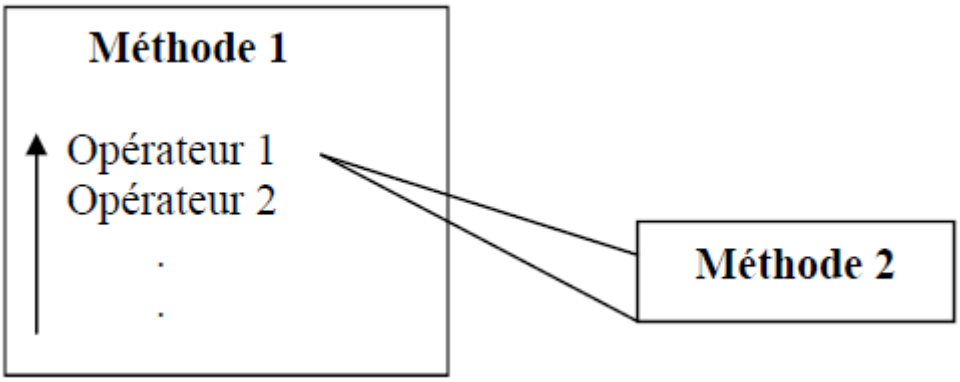


Figure 28: Hybridation parallèle synchrone.

- **Hybridation parallèle asynchrone (coopérative)** : les méthodes hybrides appartenant à cette classe sont caractérisées par une architecture telle que deux algorithmes A et B sont impliqués simultanément et chacun ajuste l'autre. Les Algorithmes A et B partagent et échangent de l'information tout au long du processus de recherche (figure 29). [47]

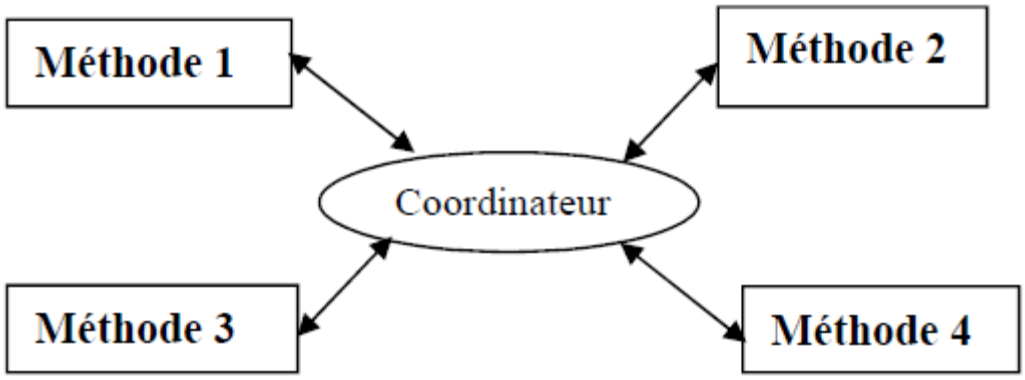


Figure 29: Hybridation parallèle asynchrone (coopérative).

4.2 Classification 2 :

Une classification des méthodes hybrides incluant les métaheuristiques et les méthodes exactes a été proposée par Dimistrescu et Stùtzle [57]. Le critère considéré est celui de l'objectif de la méthode exacte utilisée. Dimistrescu et Stùtzle proposent la classification suivante :

1. Utilisation d'un algorithme exact dans une recherche locale pour une exploration intensive de voisinage;
2. Utilisation d'une heuristique pour réduire l'espace de recherche d'une méthode exacte;
3. Utilisation une méthode exacte pour définir les bornes des solutions pour une heuristique constructive;
4. Utilisation des informations fournies par une méthode exacte pour orienter un algorithme de recherche locale ou constructif; et
5. Utilisation d'une méthode exacte pour remplacer une fonction spécifique de la métaheuristique.

III. Hybridation basée EA

Les algorithmes évolutionnaires sont devenus très utilisés pour résoudre les problèmes complexes. Plusieurs techniques sont regroupées dans cette catégorie : Les algorithmes génétiques, la programmation génétique et les stratégies évolutionnaires, etc.

Elles sont basées tous sur le même concept en simulant l'évolution d'un individu par sélection, mutation et reproduction. Les avantages des algorithmes évolutionnaires sont multiples, à savoir la simplicité de l'approche, sa robustesse face aux changements de

l'environnement et enfin sa flexibilité. Les algorithmes évolutionnaires sont faciles à implémenter par rapport à d'autres métaheuristiques. Par contre, pour plusieurs problèmes complexes, les algorithmes évolutionnaires ne réussissent pas à trouver la solution optimale, d'où la nécessité de les hybrider avec d'autres méthodes bio-inspirées, techniques d'apprentissage, heuristiques, etc.

L'idée essentielle d'hybridation des algorithmes évolutionnaires consiste essentiellement à exploiter pleinement la puissance de recherche de méthodes de voisinage et de recombinaison des algorithmes évolutifs sur une population de solutions. Un tel algorithme utilise une ou plusieurs méthodes de voisinage sur les individus de la population pendant un certain nombre d'itérations ou jusqu'à la découverte d'un ensemble d'optimums locaux et invoque ensuite un mécanisme de recombinaison pour créer de nouveaux individus. La recombinaison doit impérativement être adaptée au problème traité. En effet, cette approche a permis de produire d'excellents voire les meilleurs résultats sur des benchmarks réputés de problèmes de référence. C'est le cas par exemple du problème du voyageur de commerce, de la coloration des graphes, de la clique maximale et de l'affectation quadratique. Malgré la puissance de l'approche hybride, le temps de calcul nécessaire peut devenir prohibitif à cause du nombre d'individus manipulés dans la population. Une voie pour résoudre ce problème est la parallélisation de ces algorithmes sur des machines parallèles ou sur des systèmes distribués. Nous venons d'évoquer l'hybridation d'un algorithme évolutionniste avec des méthodes de voisinage. Il est également possible d'hybrider d'autres types d'approches, par exemple une heuristique gloutonne et une méthode de voisinage ou un algorithme évolutif.

Un exemple typique est la méthode GRASP Cette hybridation consiste à combiner une méthode de recherche locale (méthode de voisinage) avec une méthode évolutive [48].
Les raisons de cette hybridation sont :

- améliorer les performances d'un algorithme évolutionnaire (vitesse de convergence) ;
- améliorer la qualité de la solution obtenue ;
- élargir le domaine d'application des algorithmes évolutionnaires.

Ci-dessous un exemple d'un algorithme génétique hybride :

Algorithme génétique hybride

Étape 1 *(initialisation)*

a) Générer une population de configurations P

b) Appliquer une recherche locale à chacune des configurations de P

Étape 2 *(évolution et terminaison)*

a) Choisir p1 et p2 dans P

b) Générer une configuration e par une recombinaison de p1 et p2

c) Améliorer e par l'application de la recherche locale

d) Insérer la nouvelle configuration e dans la population

e) Terminer et retourner la meilleure solution trouvée si la condition d'arrêt vérifiée

Étape 3 (*mise à jour*)

f) Re-organisation de la population (ex, élimination des configurations non performantes de la population

Étape 4

Aller à (étape 2) jusqu'à critère d'arrêt satisfait.

Il existe plusieurs manières pour hybrider les algorithmes évolutionnaires, qu'on peut les résumer comme suit :

- La population initiale peut être créée avec une heuristique spécifique au problème ;
- Quelques solutions obtenues par un algorithme évolutionnaire peuvent être améliorés par une recherche locale (algorithme mimétique) ;
- Des opérateurs de croisement et de mutation peuvent exploiter les connaissances préalables du problème ;

  Une heuristique est utilisée pour le décodage des solutions en exploitant des connaissances propres du problème (voir figure 30).

Plusieurs hybridations sont proposées dans la littérature [49] hybridant un algorithme évolutionnaire avec :

1- Un autre algorithme évolutionnaire ;

2- Un réseau de neurones ;

3- La logique floue ;

4- Un algorithme d'optimisation par essaim de particule ;

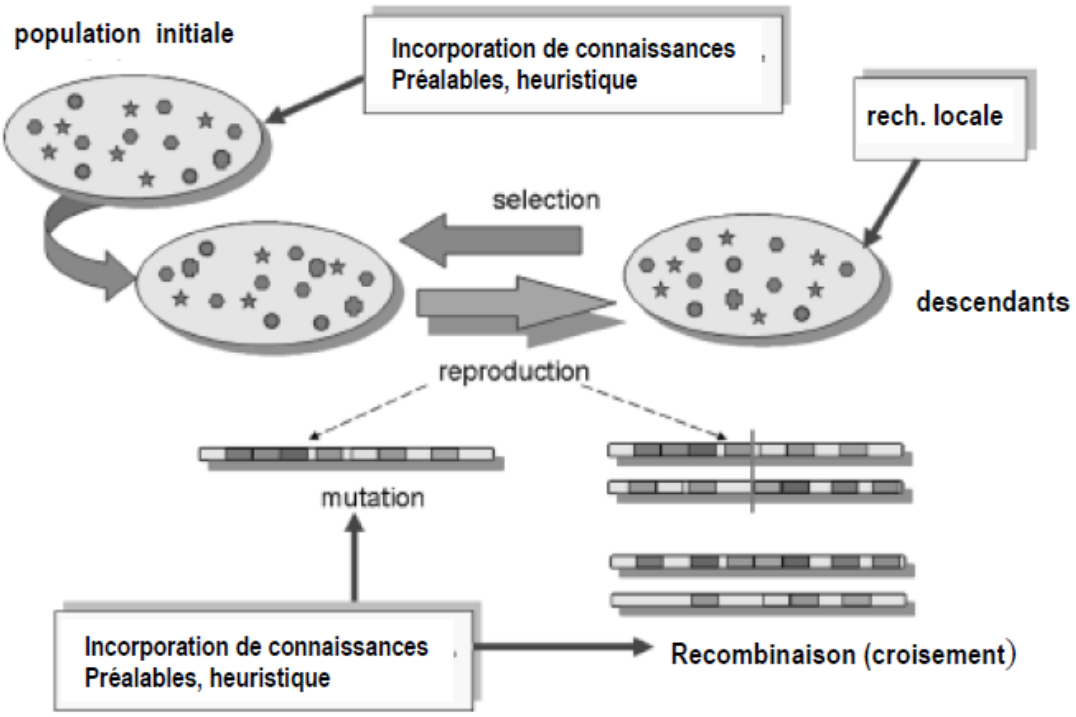


Figure 30: hybridation des algorithmes évolutionnaires [49].

1. Hybridation d'un algorithme évolutionnaire avec un autre AE

Dans Tan et al [50], une technique de classification évolutionnaire hybride à deux phases a été proposée afin d'extraire les règles de classification à utiliser dans le domaine de la prévention médicale. Dans la première phase, un algorithme évolutionnaire est utilisé pour limiter l'espace de recherche en évoluant une population de règles candidates utiles.

La programmation génétique est appliquée pour évoluer les attributs nominaux pour la structuration des règles et l'algorithme génétique est utilisé afin d'optimiser les attributs numériques, ce qui permet une classification concise des règles. Les règles candidates sont ensuite utilisées dans la deuxième phase afin d'optimiser l'ordre et le nombre de règles dans le but de former un ensemble de règles cohérentes, précises et compréhensibles. [50]

2. Hybridation d'un réseau de neurones avec AE

Wang a proposé une approche hybride pour améliorer les performances des algorithmes évolutionnaires pour un problème d'optimisation par simulation [51].

L'optimisation par simulation vise à déterminer les meilleures valeurs des paramètres d'entrée, lorsque la fonction objectif et les contraintes ne sont pas explicitement connues.

Un algorithme évolutionnaire est utilisé pour explorer les bonnes solutions dans l'espace de recherche. Le réseau de neurone est utilisé pour déterminer la valeur de la fonction fitness.

D'autre part, l'évaluation de la fonction objective dans les algorithmes évolutionnaire est souvent coûteuse, pour résoudre ce problème une approche hybride a été proposée [52]. Elle est basée sur une combinaison des réseaux de neurones et les algorithmes génétiques, l'intégration du réseau de neurones a permet d'accélérer significativement la convergence de l'algorithme génétique et par conséquent améliorer la qualité de solution obtenue.

3. Hybridation logique floue avec AE

Dans Lee & Takagi, [53], les auteurs ont proposé un algorithme génétique à paramétrage dynamique (DPGA) qui utilise un contrôleur flou (*Fuzzy logic controller FLC*) pour contrôler les paramètres généraux de l'algorithme, tels que le taux de mutation (Pm), le taux de croisement (Pc), et la taille de la population (N), vu leurs importance pour réaliser un compromis entre l'exploitation et l'exploration de l'espace de recherche. Les entrées du FLC sont représentées par une combinaison de mesures de performance, et les sorties peuvent être n'importe quel paramètre de contrôle de l'algorithme génétique.

Par exemple, le taux de mutation (Pm) peut être adapté pour empêcher une convergence prématurée et accélérer l'optimisation. Les règles correspondantes permettant d'ajuster le taux de mutation peuvent être formulées comme suit :

- si convergence alors Pm = 0,6
- si pas de convergence alors Pm= 0,05

4. Hybridation d'algorithme d'optimisation par essaim de particules avec AE

Une approche hybride nommée AE-PSO, combinant les algorithmes évolutionnaires et l'algorithme d'optimisation par essaim de particules a été proposée par Shi et al [54]. Son principe consiste à exécuter les deux techniques simultanément. Après n itérations, sélectionner P individus de chaque technique pour les échanger. L'individu ayant une meilleure fitness a effectivement plus de chances d'être sélectionné. Les principales étapes de l'approche proposée sont représentées ci-dessous:

1. Initialiser EA et PSO.
2. Exécuter EA et PSO simultanément.
3. Mémoriser la meilleure solution en tant que solution finale et s'arrêter si le critère d'arrêt est satisfait dans l'un des deux systèmes.

4. Effectuer le procédé hybride après n itérations en sélectionnant les P individus en fonction de leur fitness des deux systèmes puis les échanger. Passez à l'étape 3.

Une autre technique hybride combinant GA et PSO appelé GSO *(Genetic Swarm Optimization)* a été proposé par Grimaldi et al [55] pour résoudre un problème d'optimisation en électromagnétique. Cette technique est basée sur une forte coopération entre GA et PSO, car elle maintient l'intégration de ces deux techniques tout au long de son exécution. À chaque itération, la population est divisée en deux parties, chacune est évaluée respectivement par une technique. Les solutions seront ensuite recombinées lors de la mise à jour de la population, qui est de nouveau divisée en deux parties de façon aléatoire dans l'itération suivante.

IV. Hybridation basée PSO

L'inconvénient de PSO est que l'essaim peut converger prématurément. Ainsi la rapidité du flux d'information entre les particules, entraîne la création de particules similaires ce qui augmente la possibilité d'être piégé dans un optimum local. Un autre inconvénient est celui du réglage des paramètres qui influe totalement sur la performance de l'algorithme. L'augmentation du paramètre d'inertie ω va augmenter la vitesse des particules en favorisant en plus l'exploration (recherche globale) et moins l'exploitation (Recherche locale). D'autre part, la réduction de ω diminue la vitesse des particules et favorise en plus l'exploitation et moins l'exploration. Ainsi trouver la meilleure valeur pour le paramètre n'est pas une tâche facile et elle peut différer d'un problème à l'autre. Pour surmonter les limitations de PSO, plusieurs approches hybrides ont été proposées.

Une méthode hybride d'optimisation par essaim de particules, basée sur la fusion du PSO et l'algorithme de sélection clonale CSA a été proposé par Wang et al [56].

Cette méthode a été appliquée avec succès dans le domaine d'optimisation des fonctions non linéaires. Les étapes de la méthode proposée sont :

1. Initialiser l'essaim constitué par des groupes de n particules ;
2. Pour chaque groupe, générer j particules ;
3. Cloner ces particules à l'aide d'affinité liée à la mutation ;
4. Mettre à jour les vitesses des particules clonées ;
5. Mettre à jour les positions des particules clonées ;
6. Evaluer la fitness de chaque particule dans le groupe, et la meilleure est considérée comme le vainqueur du groupe et va concourir avec les autres groupes ;
7. Si le critère de convergence prédéfini est atteint, alors fin.

8. Sélectionnez les groupes ayant la meilleure fitness et de générer aléatoirement d'autres groupes supplémentaires pour remplacer les groupes ayant atteint leur maturité. Retourner à 2

Valdez et al. Radha et al. [58] ont intégré le concept de logique floue dans une approche hybride de l'AG et l'algorithme PSO. Le nouvel algorithme hybride appelé PSO+GA répartit les individus sur les deux populations GA et PSO. Cette approche diffère des autres, dans le sens où la décision d'affecter un individu à l'une des population est gérée par des règles floues.

Talbi et Batouche [59] ont proposé une hybridation du PSO avec DE (algorithme d'évolution différentielle), en appliquant les opérateurs de DE sur la meilleure particule obtenue par PSO. Cet algorithme est appelé DEPSO, et a été appliqué sur le problème de traitement d'images médicales.

D'autre part, Khamsawang et al. Radha et al [58] ont proposé un algorithme hybride amélioré basée sur l'optimisation par essaim de particules et l'algorithme d'évolution différentielle (appelé PSO-DE) pour résoudre le problème d'expédition. Dans PSO-DE, les opérateurs de mutation de l'évolution différentielle sont utilisés pour améliorer l'exploration du PSO, ils sont activés si les valeurs de vélocité de PSO sont proches de zéro, ou si elles dépassent les bornes.

L'algorithme HybPSO proposé dans Yannis et al. [60], combine l'algorithme standard PSO et une heuristique de recherche adaptative de voisinage pour résoudre le problème de tournées de véhicules.

V. Conclusion

Pour surmonter les problèmes introduction des approches hybrides est nécessaire. Par exemple, les algorithmes génétiques sont très performants lorsqu'il s'agit d'explorer l'espace de recherche, mais ils s'avèrent ensuite incapables d'exploiter efficacement la zone vers laquelle la population des solutions converge.

Nous avons fait le point sur les différentes stratégies de classification qui existent dans la littérature, ainsi que sur quelques approches hybrides proposées.

Chapitre IV

Machines à Vecteurs Supports Et Algorithme Génétique et GA-SVM

I. Les Machines à Vecteurs Supports

1. Introduction

L'apprentissage automatique basé sur la notion de généralisation à partir d'un grand nombre de données couvre des domaines tels que la reconnaissance de forme et la régression ne cesse d'avoir un développement dans ses méthodes et techniques, ceux-ci ne les a pas empêché de dévoiler des limites qui réduisent leur efficacité face à la complexité des problèmes du domaine, en même temps d'autre méthodes ont étaient misent en œuvre et dès leur première apparition elles ont surpassé les méthodes existantes auparavant, les SVMs sont une de ces nouvelles méthodes largement utilisés récemment

2. Définition

Les **"Support Vector Machins"** appelés aussi « maximum margin classifier» (en français machine à vecteur de support ou séparateur à vaste marge) sont des techniques d'apprentissage supervisé basés sur la théorie de l'apprentissage statistique (généralement considérés comme la 1ére réalisation pratique de cette théorie [61]) et respectant les principes du (SRM) « structural risk minimization » (trouver un séparateur qui minimise la somme de l'erreur de l'apprentissage [62]), un SVM repose sur les 2 notions de vaste marge et fonction Kernel.

Les SVMs sont considéré comme l'un des modèles les plus important parmi la famille des méthodes a Kernel, Ils ont gagné une forte popularité grâce à leur succès dans la reconnaissance des chiffre manuscrit avec un taux d'erreur de 1.1% en phase de test (le même taux marqué par un réseaux de neurone soigneusement construit) [63]

3. Historique

L'émergence des SVMs a commencé autour des débuts des années 1990s, néanmoins d'autres travaux et recherches sur l'apprentissage machine par les mathématiciens russes Vladimir Vapnik et Alex Chervonenkis ont fortement contribué à leur apparition, notamment la 1ére description de l'implémentation d'un modèle proche a un SVM apparu dans la traduction en Anglais en 1982 de l'ouvrage de Vapnik «Estimation of Depndences Based on Empirical Data » (édité en 1er lieu

en russe en 1979), que l'exploration de la notion d'hyper plan a marge maximale l'a précédé.[64]

Le model initiale a marge maximale a connu des extensions importante en 1992 qui ont formé le model finale par l'utilisation de la Kernel trick d'Aizeman proposé par Boser, Guyon & Vapnik, présenté dans un article a la conférence COLT 92, finalement les SVMs sous leur forme actuelle ont étaient introduits en 1995 par V.Vapnik & C.Cortes après l'introduction du « soft margin ». [64]

Les limites statistiques des SVMs sont apparues en 1998 par Barlett & Shawe-Taylor sur la généralisation des SVM à marge dure (hard margin), suivie en 2000 par une autre critique montrant les limites de la généralisation des algorithmes à marge souple (soft margin) par Shawe-Taylor et Cristianini. [64]

4. Domaines d'application

Vu leur composition comme des techniques d'apprentissage, les SVMs sont utilisés dans les domaines de :

- **Reconnaissance de formes/Classification :**
  - ♦ Vision Machine: Identification de visage, reconnaissance d'expression faciale : Surpasse les approches alternatives (1.5% taux d'erreur) [65]
  - ♦ Reconnaissance des chiffres manuscrits: les résultats d'USPS (service de la poste des états unis) databatse comparable à la meilleure approche (1.1% taux d'erreur) [61]
  - ♦ Catégorisation de texte : un exemple populaire est le corpus de texte de l'agence Reuteurs qui a collecté 21450 documents d'information datant de 1997 et les a partitionnés en 135 catégories différentes. [61]
- **Bioinformatique:** prédiction de la structure des protéines, prédiction du progrès d'une maladie. [65]
- **Régression:** estimation et prédiction des valeurs des fonctions [61]

L'application des SVMs s'est étendue aux domaines d'apprentissage non supervisé comme:

- ♦ Reduction de dimension: ACP non lineaire [65]
- ♦ Clustering
- ♦ Novelty detection (détection de nouveautés) [65]

5. Notions de base

5.1 Hyperplan

Plaçons-nous dans le cas d'une classification binaire (i.e. les exemples à classifier réparties en 2 classes). On appelle *hyperplan séparateur* un hyperplan qui sépare les deux classes **figure 31**, en particulier il sépare leurs points d'apprentissage. Comme il n'est en général pas possible d'en trouver un, on se contentera donc de chercher un hyperplan discriminant qui est une approximation au sens d'un critère à fixer (maximiser la distance entre ces deux classes).[66]

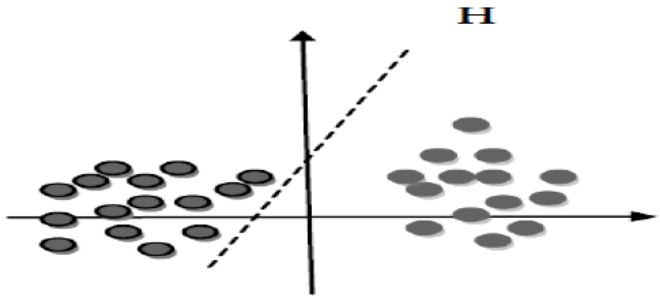


Figure 31: L'hyperplan H qui sépare les deux ensembles de points [67].

5.2 Vecteurs supports

Pour une tâche de détermination de l'hyperplan séparable des SVM est d'utiliser seulement les points les plus proches (i.e. les points de la frontière entre les deux classes des données) parmi l'ensemble total d'apprentissage, ces points sont appelés *vecteurs supports* **figure 32.** [66]

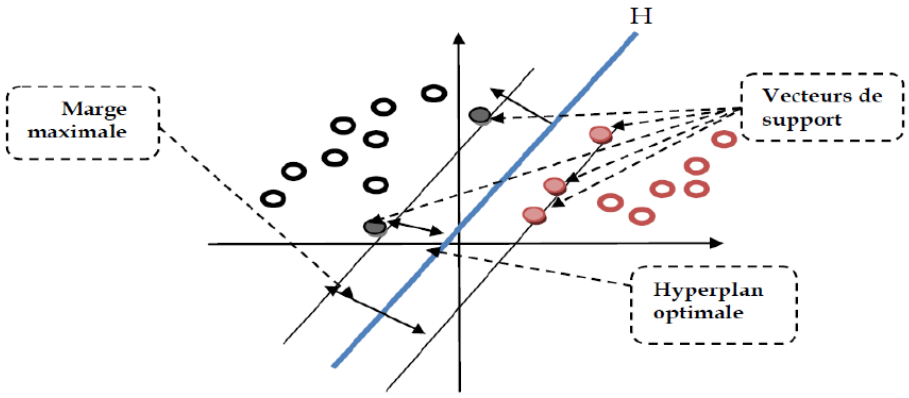


Figure 32: L'hyperplan H optimal, vecteurs supports et marge maximale[67].

5.3 Marge

Il existe une infinité d'hyperplans capable de séparer parfaitement les deux classes d'exemples.

Le principe des SVM est de choisir celui qui va maximiser la distance minimale entre l'hyperplan et les exemples d'apprentissage (i.e. la distance entre l'hyperplan et les vecteurs supports), cette distance est appelée la marge (**figure 32)**.

6. Propriétés fondamentales

- **Pourquoi maximiser la marge ? :**

Intuitivement, le fait d'avoir une marge plus large procure plus de sécurité lorsqu'on classe un nouvel exemple. De plus, si l'on trouve le classificateur qui se comporte le mieux vis-à-vis des données d'apprentissage, il est clair qu'il sera aussi celui qui permettra au mieux de classer les nouveaux exemples.

Dans le schéma **figure 33**, la partie droite nous montre qu'avec un hyperplan optimal, un nouvel exemple reste bien classé alors qu'il tombe dans la marge. On constate sur la partie gauche qu'avec une plus petite marge, l'exemple se voit mal classé.

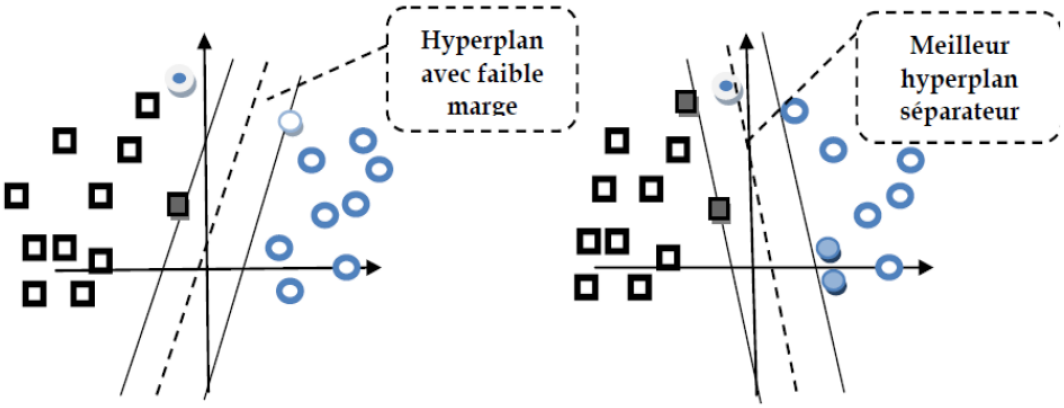


Figure 33: meilleur hyperplan séparateur [67].

7. Linéarité et non-linéarité :

Parmi les modèles des SVM, on constate

- les cas linéairement séparables
- les cas non linéairement séparables.

Les premiers sont les plus simples des SVM car ils permettent de trouver facilement le classificateur linéaire. Dans la plupart des problèmes réels il n'y a pas de séparation linéaire possible entre les données, le classificateur de marge maximale ne peut pas être utilisé car il fonctionne seulement si les classes de données d'apprentissage sont linéairement séparables.[66]

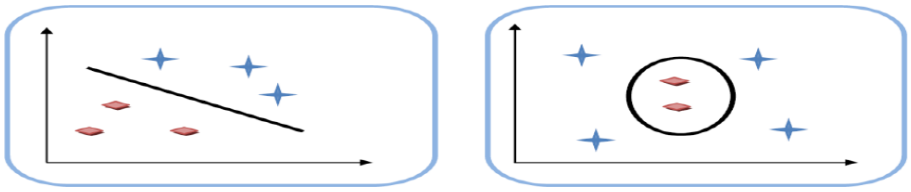


Figure 34: à gauche cas linéairement séparable, à droite non linéairement séparable [67].

- **Cas non linéaire :**

Pour surmonter les inconvénients des cas non linéairement séparable, l'idée des SVM est de changer l'espace des données. La transformation non linéaire des données peut permettre une séparation linéaire des exemples dans un nouvel espace.
On va donc avoir un changement de dimension. Ce nouvel espace est appelé « espace de redescription ». En effet, intuitivement, plus la dimension de l'espace de redescription est grande, plus la probabilité de pouvoir trouver un hyperplan séparateur entre les exemples est élevée.

On a donc une transformation d'un problème de séparation non linéaire dans l'espace de représentation en un problème de séparation linéaire dans un espace de redescription de plus grande dimension. Cette transformation non linéaire est réalisée *via* une fonction noyau.
En pratique, quelques familles de fonctions noyau paramétrables sont connues et il revient à l'utilisateur de SVM d'effectuer des tests pour déterminer celle qui convient le mieux pour son application. On peut citer les exemples de noyaux suivants : polynomial, gaussien, sigmoïde et laplacien (voir figure 35).

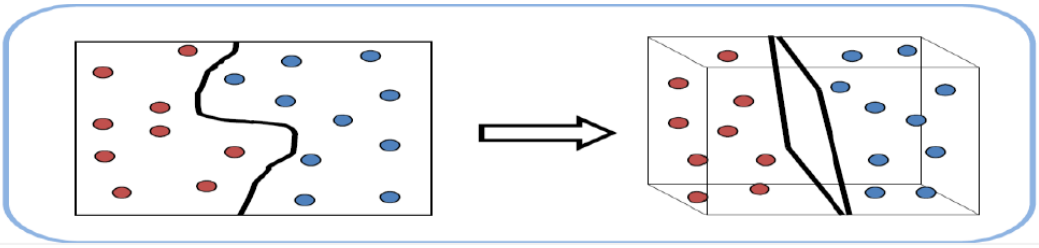


Figure 35: Transformation des données dans un espace de grande dimension [67].

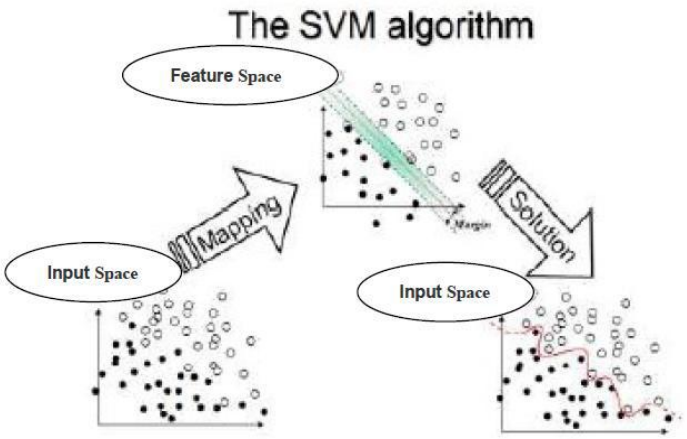


Figure 36 : Machines à vecteur support pour non linéairement séparable.

8. Les fonctions Noyau (Kernel) :

En pratique, l'astuce du noyau consiste à réécrire un algorithme ou toutes les relations entre données d'entrée peuvent s'écrire sous forme de produits scalaires, en remplaçant ce produit scalaire par une fonction scalaire de deux variables (noyau) comme nous l'avons cité dans la partie précédente. L'astuce du noyau permet ainsi de généraliser un algorithme linéaire manipulant des vecteurs :

- pour traiter les vecteurs de façon non linéaire (parce que les données présentent des non linéarités qu'il est utile d'exploiter pour le problème visé) ;
- pour manipuler d'autres types d'objets que les vecteurs, c'est-à-dire prendre en compte des structures de données assez complexes et non homogènes (séquences, chaines, graphes,…etc.). Donc il suffit de trouver la bonne fonction noyau qui exprime le mieux la similarité entre les différentes structures.

L'idée retenue dans SVM va dans un autre sens : on va tenter de trouver un mapping de l'espace d'entrée vers un autre espace (feature space) dans lequel les données sont linéairement séparables.[68]

Linéaire $k(x,x') = x \cdot x'$

Polynomial $k(x,x') = (x \cdot x')^d \, ou (c + x \cdot x')^d$

Gaussien $k(x,x') = e^{-\|x-x'\|^2/\sigma}$

Laplacien $k(x,x') = e^{-\|x-x'\|/\sigma}$

Sigmoïde $k(x,x') = \tanh(\alpha_0 (x \cdot x') + \beta_0)$

9. Avantages et inconvénients

✓ Les avantages:

- Absence d'optimum local.
- Contrôle explicite du compromis entre la complexité du classifieur et l'erreur.
- Possibilité d'utilisation de structure de données comme les chaines de caractères et arbres comme des entrées.
- Traitement des données a grandes dimensions.

✓ Les inconvénients :

- Demande des données négatives & positives en même temps.
- Besoin d'une bonne fonction Kernel.
- Problèmes de stabilité des calculs dans la résolution de certains programmes quadratiques à contraintes.

II. Les algorithmes génétiques

L'algorithme génétique représente une célèbre métaheuristique évolutionnaire. Il a été proposé par Jhon Holland en 1975 [69]. L'algorithme génétique s'inspire des mécanismes biologiques tels que les lois de Mendel et la théorie de l'évolution proposée par Charles Darwin [70].

Son processus de recherche de solutions à un problème donné imite celui des êtres vivants dans leur évolution. Il utilise le même vocabulaire que celui de la biologie et la génétique classique, on parle donc de: individu, population et génération.

- ✓ **Un individu**: est composé d'un ou de plusieurs chromosomes. Il représente une solution possible au problème traité.
- ✓ **Une population**: est représentée par un ensemble d'individus (i.e. l'ensemble des solutions du problème).
- ✓ **Une génération**: est une succession d'itérations composées d'un ensemble d'opérations permettant le passage d'une population à une autre.

L'algorithme génétique fait évoluer une population composée d'un ensemble d'individus pendant un ensemble de génération jusqu'à ce qu'un critère d'arrêt soit vérifié. Le passage d'une population à une autre est réalisé grâce à des opérations

d'évaluation, de sélection, de reproduction (croisement et mutation) et de remplacement.

L'algorithme commence la recherche avec un ensemble d'individus. A chaque itération de la procédure de recherche, les meilleurs individus sont sélectionnés pour survivre et se reproduire. La sélection des individus est fondée sur leurs qualités qui sont mesurées à partir d'une fonction appelée « fonction objectif ou fonction fitness ». Ensuite, les individus (appelés parents) sont sélectionnés pour subir des opérateurs de croisement et de mutation permettant la génération d'une autre population d'individus (appelés enfants). Les individus de la nouvelle population seront évalués pour remplacer une partie des individus de la population courante. La figure 37 illustre un schéma général des étapes du processus de recherche de l'algorithme génétique.

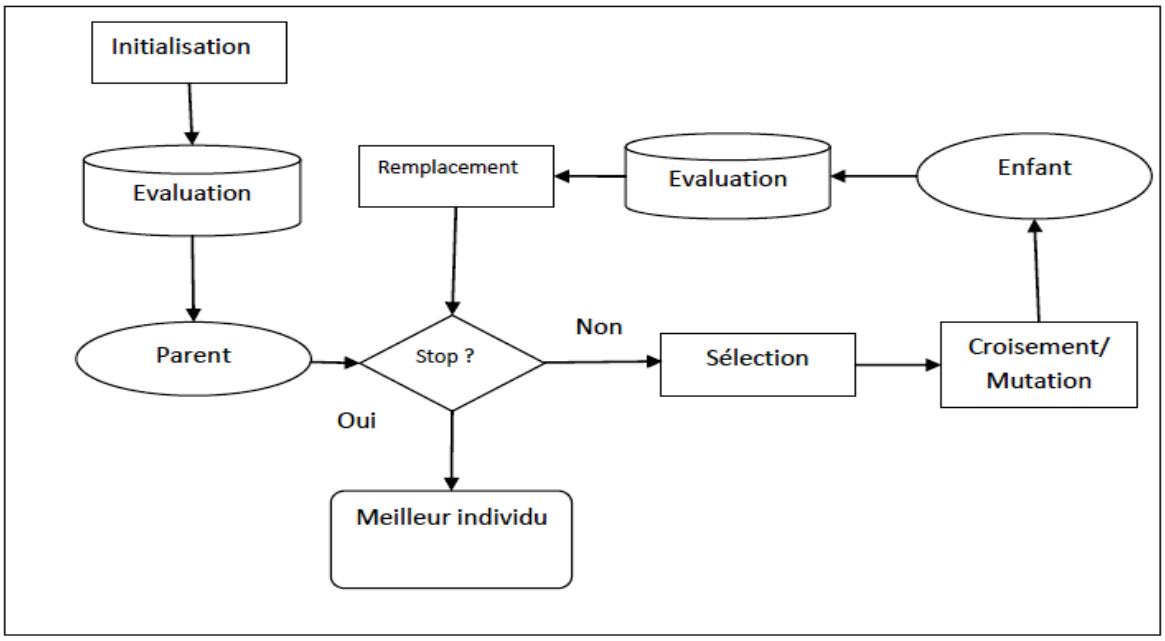


Figure 37 : Démarche d'un algorithme génétique. [71]

1. Le codage :

Les algorithmes génétiques travaillent sur des codages et non sur des solutions réelles. Ces codes sont appelés chromosomes. Il faut définir et coder convenablement le problème afin de cerner l'espace des solutions possibles. Ce codage doit, de plus, être aussi compact que possible pour permettre une évolution rapide. Historiquement le codage utilisé par les algorithmes génétiques était le codage binaire. Cependant, ce type de codage n'est pas toujours bon pour certains problèmes d'optimisation pour cela d'autres types de codage sont utilisés, le codage entier, le codage réel…etc. [20]

2. Population initial :

Si la position de l'optimum dans l'espace de recherche est totalement inconnue, le choix de la population initiale se fait d'une manière aléatoire. Si par contre, des informations à priori sur les problèmes sont disponibles, il paraît bien évidemment naturel de générer les individus dans un sous-espace particulier afin d'accélérer la convergence. Dans l'hypothèse où la gestion des contraintes ne peut se faire directement, les contraintes sont généralement incluses dans le critère à optimiser sous forme de pénalités et lorsque c'est possible ne générer que des éléments de la population respectant les contraintes. [20]

3. Evolution de population :

Pour calculer le coût d'un point de l'espace de recherche (individu), on utilise une fonction d'évaluation ou fonction fitness f, L'évaluation d'un individu ne dépendant pas de celle des autres, le résultat fourni par la fonction d'évaluation va permettre de sélectionner ou de refuser un individu pour ne garder que ceux qui présentent le meilleur coût en fonction de la population courante : c'est le rôle de la fonction fitness. Cette méthode permet de s'assurer que les individus performants seront conservés, alors que les individus peu adaptés seront progressivement éliminés de la population.[20]

4. La sélection :

La sélection permet d'identifier statistiquement les meilleurs individus d'une population et d'éliminer les mauvais. Un individu ayant une forte valeur d'adaptation a alors plus de chances d'être sélectionné qu'un individu mal adapté à l'environnement (ceci va permettre de favoriser la reproduction des « bons » individus).

La sélection ne crée aucune nouveauté, elle se contente de choisir parmi la population mère quelles seront ou ne seront pas en mesure de contribuer à la création de la population fille, suivant une stratégie particulière. Cependant, la sélection est relativement délicate à manipuler. [20]

Un excès de sélection peut entraîner une perte de la diversité au sein de la population qui bloque ainsi la résolution du problème donné, en convergeant par exemple vers des optima locaux. Une sélection trop faible pose la

difficulté inverse : la non convergence de l'algorithme. Pour cela on distingue deux techniques de sélection : la sélection stochastique et la sélection déterministe.

On trouve essentiellement quatre types de méthodes de sélection différentes :

- La méthode de la "loterie biaisée" (roulette wheel) de GoldBerg;
- La méthode "élitiste";
- La sélection par tournois;
- La sélection universelle stochastique.

5. Le croisement :

Traditionnellement, le croisement est vu comme l'opérateur de recherche essentiel d'un algorithme génétique. Après la sélection, un croisement peut éventuellement avoir lieu. Dans ce cas, deux individus de la nouvelle génération échange une ou plusieurs partie(s) de leur génotype pour former deux individus différents de ceux d'origine. Cet opérateur est essentiel, car il permet d'obtenir de nouveaux individus distincts de ceux déjà existants et donc d'explorer tout l'espace de recherche. Il existe différentes méthodes pour croiser deux chromosomes :

- **le croisement à un point** : Ce type de croisement est standard dans les algorithmes génétiques. Il consiste à choisir un emplacement aléatoirement sur une chaîne et d'intervertir tous les gènes d'un coté de ce point entre les deux chaînes.
- **le croisement à n points :** Le croisement à n points est une généralisation du croisement à un point avec n points de coupure sur les chaînes. Il s'agit alors de déterminer n points sur ces chaînes, puis d'échanger sur les chaînes les blocs entre ces points afin d'obtenir les enfants. [20]

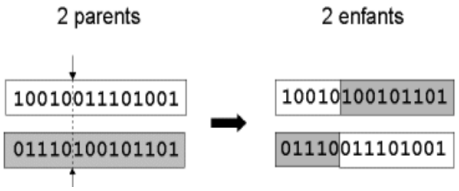


Figure 38 croisement avec 1 point.

:

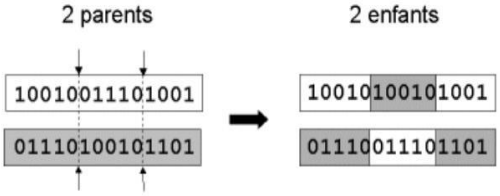


Figure 39: croisement avec 2 points.

6. Mutation :

La mutation est exécutée seulement sur une seule chaîne. Elle représente la modification aléatoire et occasionnelle de faible probabilité de la

valeur d'un caractère de la chaîne, pour un codage binaire cela revient à changer un 1 en 0 et vice versa (figure 40). Cet opérateur introduit de la diversité dans le processus de recherche des solutions et peut aider l'AG à ne pas stagner dans un optimum local.

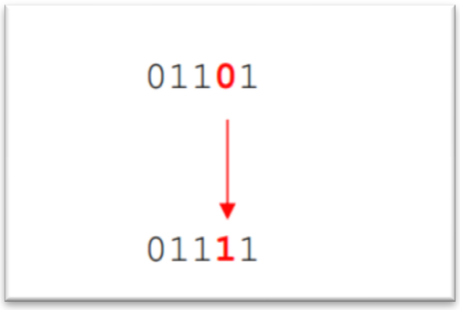


Figure 40 : Représentation d'une mutation de bits dans une chaîne.

7. Élitisme

L'opérateur d'élitisme est une méthode de sélection qui permet de retenir seulement les meilleurs individus d'une population pour la génération suivante. Si les individus ne sont pas gardés, ils risquent de ne pas survivre pour les prochains cycles après les opérations de croisement ou la mutation. Concrètement l'élitisme consiste donc à copier un pourcentage (normalement choisi comme un paramètre de l'AG) des meilleurs individus dans la nouvelle génération pour éviter qu'ils soient effacés de la population.

➢ Résumé des étapes de l'algorithme génétique :

[Début] : générer aléatoirement une population de n chromosomes.

[Evaluation] : évaluer l'adaptation (fitness) f(i) de chaque chromosome dans la population.

[Nouvelle génération] : créer une nouvelle population en répétant les étapes suivantes jusqu'à ce que la nouvelle population est complétée.

1. [Sélection] : Sélectionner des chromosomes parents de la population relativement à leurs fitness (meilleur fitness, une grande chance d'être sélectionné,…)

2. [Croisement]: Avec la probabilité du croisement fixée, croiser les parents pour former les nouveaux descendants. Si aucun croisement n'a eu lieu les descendants sont des copies exactes des parents.

3. [Mutation] : Avec la probabilité de mutation fixée, muter le nouveau descendant à chaque locus (position dans le chromosome).

4. [Acceptation] : Placer les nouveaux descendants dans la nouvelle population

[Remplacer] : Utiliser la nouvelle population générée pour l'exécution de l'algorithme de nouveau.

[Test] : Si la condition fin est satisfaite stop et retourner la meilleure solution dans la population courante.

[Boucler] : Aller à l'étape 2.

Algorithme génétique

définition de le fonction objectif f(x) , x=(x1,…,xn)

Représentation de la solution sous forme de chromosome

Génération de la population initiale

Evaluation de fitness

Initialiser les probabilités de croisement (pc) et mutation (pm)

t←1

tant que (t < max-génération) **faire**

Générer de nouvelle solution par mutation et croisement

si pc > rand **alors**

Croisement

f**in**

si pm > rand **alors**

Mutation

fin

Accepter les nouvelles solutions si leur fitness est amélioré

Sélectionner les meilleures solutions actuelles pour la nouvelle génération

t←t+1

fin

Décodage et visualisation des résultats

III. Combinaison des algorithmes génétiques et SVM

L'application des algorithmes génétiques à l'optimisation des problèmes complexes peut conduire à un effort de calcul importante comme un résultat de l'évaluation répétée de la fonction objectif (s) et la nature de la recherche qui basée sur la population.

C'est souvent le cas où l'évaluation de la fonction d'objectif est coûteuse, par exemple, lorsque la valeur est obtenue en suivant les simulations du système de calcul coûteux. Parfois sensiblement grand nombre de générations peut être nécessaire pour trouver la valeur optimale de la fonction objective. En outre, dans certains cas, algorithme génétique peut faire face à des problèmes de convergence.

Dans ce mémoire, un algorithme hybride est présenté. Il est basé sur une combinaison de la machine à vecteur de support et les algorithmes génétiques.

1. Motivation de combinaison

SVM est utilisé pour améliorer la convergence de l'algorithme génétique à la recherche d'un optimum global. L'efficacité de la méthode de calcul proposée est illustrée par l'application d'un certain nombre de cas de test.

Les résultats montrent que, dans le procédé hybride proposé, l'intégration de la machine à vecteur de support dans la procédure de l'algorithme génétique peut accélérer la convergence et améliorer les performances significatives de l'algorithme génétique et améliorer la qualité de la solution.

2. AG- SVM pour la découverte de motifs

Les algorithmes génétiques ont été largement utilisés dans le domaine de la découvert des motifs, ces algorithmes permettent également d'utiliser les fonctions d'adaptation (fitness) pour évaluer (le score) les solutions. Ces fonctions d'adaptation (fitness) ne doivent pas être constante pour tous les problèmes .Dans notre projet SVM est utilisée pour évaluation (fitness) des solutions.

3. La combinaison AG- SVM

Dans ce mémoire un algorithme hybride est présenté [73] pour la résolution d'un problème en bioinformatique, à savoir la découverte de motifs biologiques. Cet algorithme est basé sur l'intégration de machine à vecteur de support (SVM) dans une procédure d'algorithme génétique. Dans l'algorithme proposé, l'efficacité de

l'algorithme génétique est améliorée en utilisant les capacités d'apprentissage de machines à vecteur de support.

En combinant les deux procédés, les avantages des deux méthodes sont exploités pour produire un procédé d'optimisation hybride qui est à la fois robuste et rapide.

Dans l'algorithme GA-SVM, l'algorithme génétique est utilisé pour explorer les bonnes solutions dans l'espace de recherche, SVM est utilisé pour déterminer la valeur de la fonction fitness.

4. L'objectif de AG-SVM

L'objectif de ce projet est d'identifier les motifs qui représentent des éléments réglementaires dans les séquences biologiques. L'entrée de l'algorithme se compose de trois ensembles des séquences. Le premier est un ensemble de séquences de **promoteur** qui a la probabilité de contenir les éléments réguliers. Le second est un ensemble de séquences **d'arrière-plan** représentant une partie aléatoire de l'ADN qui ne peuvent comporter des séquences de **promoteur**. Le troisième est un **ensemble des trains** est un ensembles des éléments réguliers vérifiés expérimentalement fonctionnels pour l'entrainement du SVM.

L'approche suivie dans ce projet pour appliquer un **AG-SVM** dans notre problème (découvert de motif) peut être résumée comme suit:

1) Déterminer une représentation pour les motifs.
2) Évaluer les motifs à l'aide de SVM et en plus une fonction d'adaptation statique bien définie.
3) Regrouper (cluster) la population sur la base de certaines métriques.
4) Exécutez l'algorithme génétique de répéter les étapes ci-dessus jusqu'à un critère d'arrêt qui est le nombre des générations .

- **La représentation des Motifs**

La représentation de motifs utilisés dans ce projet est une combinaison de deux matrices de fréquences de position (PFM) et des matrices de poids de position (PWM).

- **Évaluation des motifs**

En calculer la valeur de fitness de chaque motif en utilisant le taux de classification correct (**accuracy**) avec SVM et en utilise aussi une fonction

d'adaptation (**statique fitness**) bien définie. Les deux méthodes d'évaluations (**fitness)** utilisées pour évaluer les motifs peuvent être décrites comme suit :

- **Fitness aves SVM**

Chaque motif est évalué par la précision de classification SVM (**accuracy**) , d'autres termes, un individu (motif) de population conduisant à un taux de classification correct élevé est considéré comme mieux qu'un individu conduisant à un taux de classification correct faible.

Donc, pour chaque motif x : **Fitness (x) = accuracy de SVM (x)**

- **Fitness statique**

Une fonction statique décrite comme la façon dont elle peut aider à différencier entre les deux ensembles de séquences (ensemble de séquences de **promoteur et** les séquences **d'arrière-plan**). La fonction d'adaptation (fitness) utilisée pour évaluer les motifs peut être décrite comme suit :

1. Le motif est converti en sa représentation **PWM**
2. Le score de PWM pour chaque séquence (**promoteur et d'arrière-plan** est calculé comme suit:

a) Le score du **PWM** à chaque offset de la séquence d'entrée est calculée à l'aide de la **PWM** généré ci-dessus.

b) La valeur maximale de score dans l'ordre est considérée comme le meilleur score pour la totalité de la séquence.

c) Cette valeur est divisée parla valeur maximale possible pour une **PWM** de la taille spécifiée afin de normaliser à une plage de **[0, 1].**

d) Ce processus est répété pour chaque séquence dans les deux ensembles de données.

3. La meilleure valeur moyenne de match est ensuite calculée pour chacun des deux ensembles de données.

La fitness du motif = L'écart entre les deux valeurs moyennes, calculées comme suit :

Meilleur score moyen pour la séquence promoteur - Meilleur score moyen pour la séquence d'arrière-plan

- **Fitness final (GA-SVM)**

Le score final de fitness est la moyenne de ces scores :

Fitness(c) = (fitness SVM (x) + fitness statique (x)) / 2

- **Regroupement (cluster) de la population**

En raison que les éléments de régulation fournissent des sites de liaison pour des facteurs de transcription, il est important de garder à l'esprit qu'il y a toujours une possibilité de la présence de plusieurs motifs dans les régions régulatrices de séquences biologiques. Un algorithme génétique fonctionne par l'évolution de la population avec SVM, tels que la plus forte solution représente toujours la meilleure solution possible. Toujours en obtient une solution unique à la fin du processus, ou très similaire solutions.

La technique de regroupement de la population (clustering) a été choisie pour ce projet. Il s'agit d'utiliser un algorithme de clustering pour diviser la population.

Dans ce projet en utilise **l'algorithme de leader** pour le **regroupement** , est un algorithme en une passe utilisé pour diviser l'espace en M partitions en calculant la distance **euclidienne** entre toutes les paires d'objets et en la comparant à un seuil **T.**

Nous avons utilisé. La version modifiée de l'algorithme de leader est une passe multi-mode .Cette méthode ne nécessite pas de calcul des seuils. Elle commence par un objet initial central, et sur le premier passage, trouve le plus loin de l'objet à partir du central. Cet objet devient le leader du groupe suivant. Le passage suivant consiste à trouver un objet qui est le plus éloigné de son leader correspondant. Cet objet est ensuite devient le leader du prochain cluster. Ce processus est répété jusqu'à ce que le nombre requis de groupes été formés

5. Les étapes de AG-SVM dans le projet

Dans ce qui suit, les étapes de l'algorithme proposé sont expliquées en détail :

1- La procédure commence par la génération aléatoire d'une population initiale.

2- Evaluer chaque individu dans la population initiale en utilisant le **fitness SVM** et le **fitness statique**. SVM est utilisé comme évaluateur de fitness en raison de son avantage unique dans le traitement de l'espace de haute solution. Il est généralement admis que l'hyperplan optimal avec le plus petit nombre de vecteurs de support a une capacité d'adaptation plus large et la performance de classification la plus élevée.

3- Regrouper la population (cluster)

4- Sélection : Pour effectuer les mutations et croisement, les parents sont sélectionnés à l'aide d'une méthode appelée **La roue de la fortune** .La méthode de La roue est la plus couramment utilisée pour le processus de sélection. Toutefois, elle ne fonctionne pas très bien quand il existe de grandes différences entre la fitness des objets dans la population. Ainsi, s'il y a des objets qu'ont une fitness élevée comme 80%, et d'autres ont une fitness bien moindre, ces objets de fitness inférieurs auront une très petite chance d'être sélectionnés. Pour résoudre ce problème, la méthode de sélection utilisée est le Classement. Cette méthode un peu modifiée a la méthode de la roue. Elle classe tous les objets en fonction de la fitness (de 1 à N, où N est la taille de la population). Les portions de la roue sont ensuite attribuées en fonction de ces classes, au lieu de la fitness. Cela donne tous ces objets une chance d'être sélectionné en fonction de leurs rangs.

Les étapes de la méthode de sélection selon le Classement sont:

1) Classement de tous les éléments du groupe sur la base de leur fitness, de telle sorte que l'élément avec la fitness la plus basse est de classe1, et celui avec la plus haute fitness devra être une classe N.
2) Trouver la somme de tous les classes
3) Le pourcentage de la roue attribuée à chaque élément est égale à: (class *100) /somme des classes.
4) Tourner la roue de la fortune.
5) Déterminer l'élément correspondant à l'emplacement sélectionné.
6) Sélectionnez cet élément

5- Reproduction : Il y a trois méthodes sont utilisées pour l'accouplement :

- **Elitisme :** Élitisme implique que l'instance avec la meilleure valeur de fitness est autorisée à se propager à la génération suivante , sans aucune modification . Ceci est fait parce que de nombreuses fois la meilleure solution disparaissent en raison de la mutation et le croisement. L'élitisme permet la meilleure solution de la population à avancer tel quelle. Si la solution n'est pas assez bonne, elle se supprime au cours de l'évolution.

- **Mutation** : La mutation est appliquée à la probabilité de **10%** par position nucléotidique **PFM**. Les étapes de processus de mutation sont les suivantes :
  1) La position de nucléotide de la **PFM** est sélectionnée en utilisant une probabilité de 10% par position.
  2) Pour la position choisie, la base (A, C, T ou G) est choisie au hasard.
  3) La fréquence de la base est modifiée en fournissant une norme écart dans la plage de +/- 0,5.
  4) Si la fréquence de la position est supérieure à 1, alors ces changements sont annulés et la valeur d'origine est restaurée.
  5) Les fréquences des bases restantes sont encore choisie au hasard de telle sorte que la somme totale des fréquences de toutes les bases de la position sélectionnée ne dépasse pas 1.
  6) Ajouter cette descendance à la nouvelle population

- **Croisement :** Dans des circonstances normales, les résultats de croisement dans la génération résultent en deux enfants. Le mécanisme de croisement utilisé dans le cadre de ce projet, toute fois génère une seule génération. La technique utilisée est appelée croisement uniforme. Dans le croisement uniforme, la valeur de chaque position de la génération est dérivée de un de ses parents normalement avec 50% de probabilité de sélection de un des deux parents.

  Dans certains cas, le pourcentage de probabilité de sélectionner les données de position peut être différent, selon quels parents avec une probabilité plus élevée doivent être favorisée .La méthode de croisement uniforme est appliquée comme suit:

1) Sélectionnez deux parents à l'aide de la méthode de sélection.
2) Prenez la taille de la progéniture égale à la taille du plus grand parent.
3) Pour chaque position de la progéniture, sélectionnez le parent dont les valeurs seront utilisées avec une probabilité de sélection de un des deux parents de 50%.
4) Copiez les valeurs de fréquence du parent sélectionné pour la position considéré.

5) Répétez les étapes (3) et (4) jusqu'à ce que le nombre des postions couverts = taille du plus petit parent.
6) Copiez les valeurs pour le reste des positions à partir du plus grand parent
7) Ajouter cette descendance à la nouvelle population

6- Répéter l'étapes **2 , 3 , 4 et 5** jusqu'à le nombre des générations est accéder

6. Description de l'algorithme SVM pour l'évaluation

Dans notre algorithme AG / SVM, nous utilisons un SVM pour évaluer la qualité d'un motif X. On utilise deux ensembles des séquences pour le train de SVM .Le premier ensemble est l'ensemble des éléments réguliers vérifiés expérimentalement fonctionnels pour l'entrainement positive .Le deuxième est un ensemble de séquences non réglementaires choisies au hasard (non-codantes pour le train négative de SVM.

Dans notre étude, nous avons choisi d'utiliser le noyau **RBF,** parce que RBF a été utilisé dans plusieurs études antérieures pour la classification de données de puces à ADN

Nous appliquons une méthode **Leave-One-Out** de validation croisée (**LOOCV**) pour calculer la précision moyenne (taux de classification correcte) d'un SVM formés avec ce motif dans **l'ensemble de train** (Collection des régions régulatrices vérifiés expérimentalement fonctionnels) de **SVM** (train positive). La procédure de **LOOCV** signifie qu'un échantillon de l'ensemble de données est considéré comme un cas de test tandis qu'un SVM est formé sur tous les autres échantillons, et cette évaluation est répétée pour chaque échantillon. Nous utilisons **LOOCV** avec **10-fold** cross-validation

Pour chaque motif x : **Fitness (x) = accuracy de SVM (x)**

Pour l'évaluation de chaque motif **x** :

Mets le motif **X** dans l'ensemble de train positive

1. Diviser l'ensemble d'entrée sur 10 échantillons

Pour chaque échantillon **E** :

- Fait le train de SVM avec 9 échantillons
- Fait le test avec l'échantillon **E**

2. Calculer le taux de classification correcte

Calculer la le taux totale de classification correcte (**accuracy**)

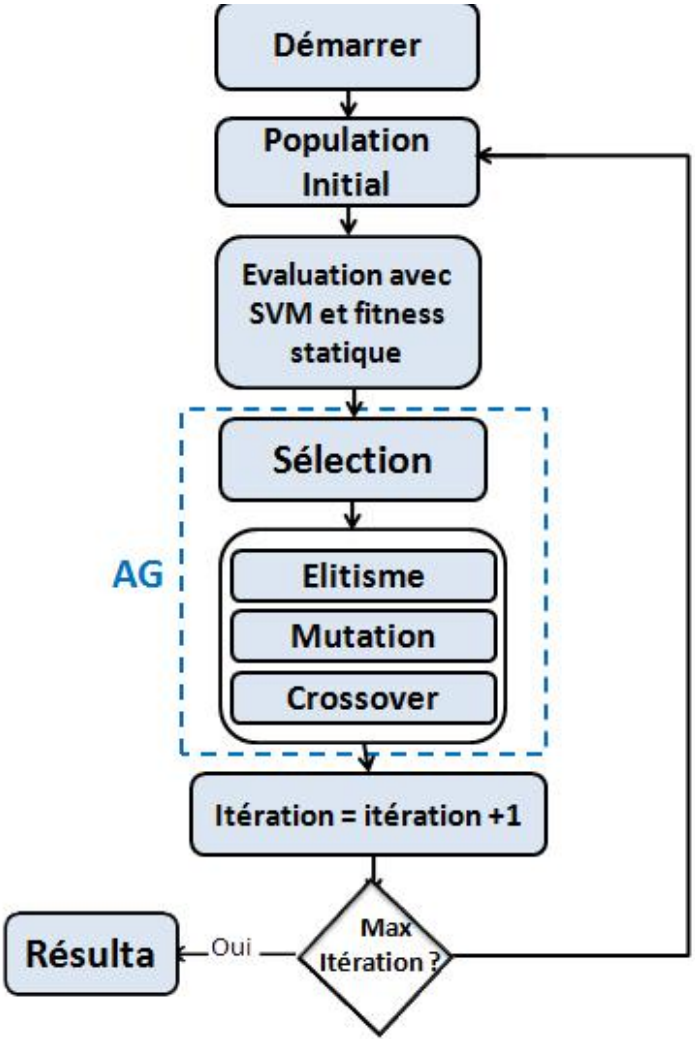


Figure 41: Diagramme combinaison entre GA et SVM

7. Description de l'algorithme

L'algorithme GA-SVM peut être montré de la manière suivante :

Étape 1 : Créer population initiale au hasard, le nombre de la population est popsize

Etape 2 : calculer la précision (Accuracy) qui sera la valeur de fitness dans l'algorithme génétique et Évaluer la valeur de fitness pour chaque population par SVM et fitness statique

Répéter

Etape 3 : Opération génétique: la sélection, croisement, mutation

Étape 4 : Si non l'itération maximale N, puis Etape 2;

8. Pseudo code

Input: DNA sequences

Output: motif

1: S←population Initialization ()

2: **while** not stop condition **do**

3: **for** each individual xi of the population S **do**

4: evaluate (xi) with SVM and static fitness

end for

5: clustering population S on numcluster group

6: for each cluster **(i)**

7:for each individu (xi) de cluster (i)

8: evaluate (xi) with SVM and static fitness

end for

9: find the individu (hi) wich have the max fitness

10: **elitism** (hi)

11: delete (hi) from cluster (i)

12: find two individu (hi) and (li) wich have the max fitness

13: ci =**crossover** (hi , li) or mi = **mutation** (hi)

14 : **if** (crossover)

delete (hi ,li) from cluster (i)

put (ci) in the next generation

else

delete (hi) from cluster(i)

put (mi) in the next generation

end for

end while

18: **Output:** best solution found

IV. Conclusion

Les séparateurs à vaste marge sont une méthode de classification qui montre de bonnes performances dans la résolution de problèmes variés. Cette méthode a montré son efficacité dans de nombreux domaines d'application tels que le traitement d'images, la catégorisation de textes ou le diagnostic médical, et ce même sur des ensembles de données de très grande dimensions. De même, les algorithmes génétiques sont très puissants est sont utilisées pour explorer les bonnes solutions dans l'espace de recherche

En combinant les deux algorithmes, les avantages des deux méthodes sont exploités pour aboutir à un procédé hybride qui est à la fois robuste et rapide.

Chapitre V

Implémentation Et Résultats Expérimentaux

1. Introduction

Dans ce chapitre, nous présentons le Matériel ainsi le langage de programmation utilisés. Des données on été utilisées pour valider l'approche proposée. Les tests utilisés, et les résultats obtenus sont présentés.

2. Matériel et langage de programmation :

Nous avons réalisé notre application sur un micro ordinateur ayant les caractéristiques suivantes :

- Micro processeur Dual-core 2.20GHz 2.20GHz ,
- RAM 2 Go,
- Disque dur 320Go,
- Système d'exploitation Windows 7Titan.

Pour le langage de programmation nous avons opté pour le MATLAB, car ce langage évolue nos recherches et nous fait gagner du temps grâce à ses fonctions prédéfinies.

MATLAB est un langage de calcul scientifique de haut niveau et un environnement interactif pour le développement d'algorithmes, la visualisation et l'analyse de données, ou encore le calcul numérique. En utilisant MATLAB, on peut résoudre des problèmes de calcul scientifique plus rapidement qu'avec les langages de programmation traditionnels, tels que C, C++.

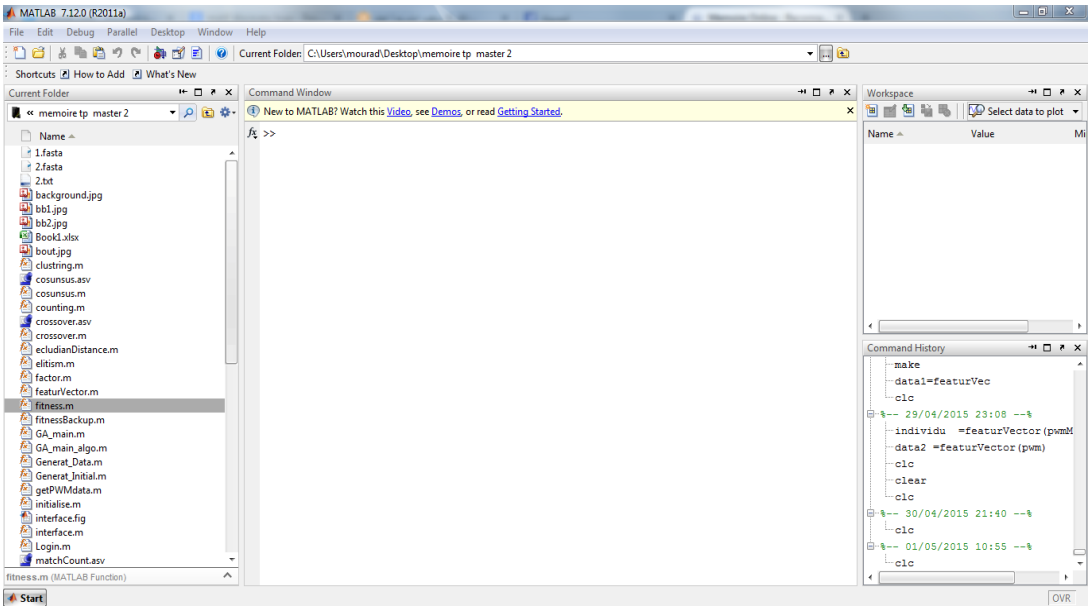


Figure 42: Fenêtre principale de MATLAB2011

3. But de l'étude:

Le but de notre étude est de développer une application qui permet la découverte des motifs ADN (bioinformatique) en utilisant une technique hybride (combinaison).

Combiner l'algorithme génétique (optimisation) et support vecteur machine (apprentissage automatique) nous conduit à obtenir une solution plus optimale que d'utiliser chaque méthode a part.

4. Description des données

Cette section donne des détails sur les divers ensembles des données qui ont été utilisés dans notre algorithme, quatre ensembles de données ont été utilisés :

- Collection des régions régulatrices vérifiées expérimentalement fonctionnels de **Drosophila (les mouches des fruits)**, on le divise en deux ensembles : un pour l'entrainement du classificateur SVM (pour l'entrainement positive) et l'autre pour le test de l'algorithme (AG-SVM) (on vérifie si l'algorithme est capable de détecter ces motifs). **http://autosome.ru/iDMMPMM/**
- Des séquences non réglementaires choisies au hasard de **Drosophila** non-codantes pour l'entrainement négative du classificateur SVM et l'ensemble des séquences d'arrière-plan. **http://www.ensembl.org/**

- Les séquences d'ADN de **Drosophila** qui contiennent des sites de fixation de facteur de transcription au niveau des régions promotrices définies expérimentalement (**promoteur**).

http://labs.biology.ucsd.edu/Kadonaga/DCPD.htm

5. Les paramètres de L'algorithme

Paramètre	**Valeur**
La longueur de population initiale	**10– 20 n**
La taille de la population initiale	**20 PFMs**
Ratio (de mutation /croisement)	**7 : 3**
Ration de Mutation1	**100%**
Ration de Mutation2	**0%**
Fréquence de changement dans la Mutation1	**+0.5**
Probabilité de sélection pour les deux parents dans le croisement	**50%**
Nombre des clusters	4

Tableau 4: Table des paramètres de GA

Paramètre	**Contenu**
Type de model	SVM (support vector machine)
SVM kernel fonction	RBF (Radial basis **)**
La méthode de validation	K-fold cross validation
Le nombre de fold cross validation (K)	10

Tableau 5: Table des paramètres de SVM

6. Les interfaces de l'application

Figure 43: Interface d'authentification

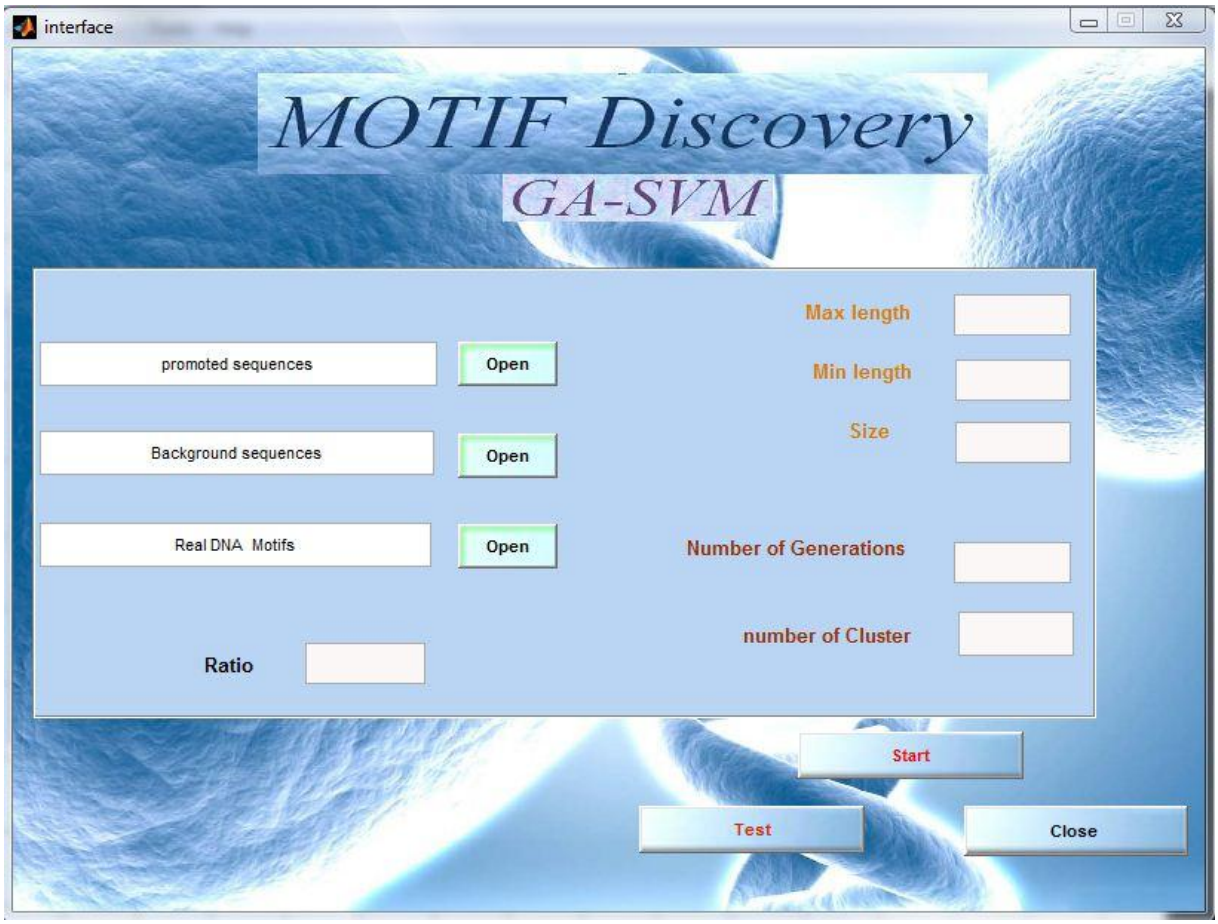


Figure 44: Interface principal du programme

Initialement, on choisit le fichier qui contient la séquence promoteur (le fichier doit être avec une extension .**txt** ou .**Fasta**) et le fichier qui contient la séquence d'arrière-plan et le fichier qui contient la séquence réelle. Après, nous introduisons les paramètres :

- **Initial population :** représente la palette de la population initiale.
- **Min length :** la taille minimale.
- **Max length :** la taille maximale.
- **Size :** nombre des matrices PFM dans la population initiale.
- **Mutaion to crossover ration :** représente la proportion de mutation pour le croisement (cette valeur est proportionnelle à **10**).
- **Number of generations :** le nombre de générations.
- **Number of cluster :** nombre de clusters.

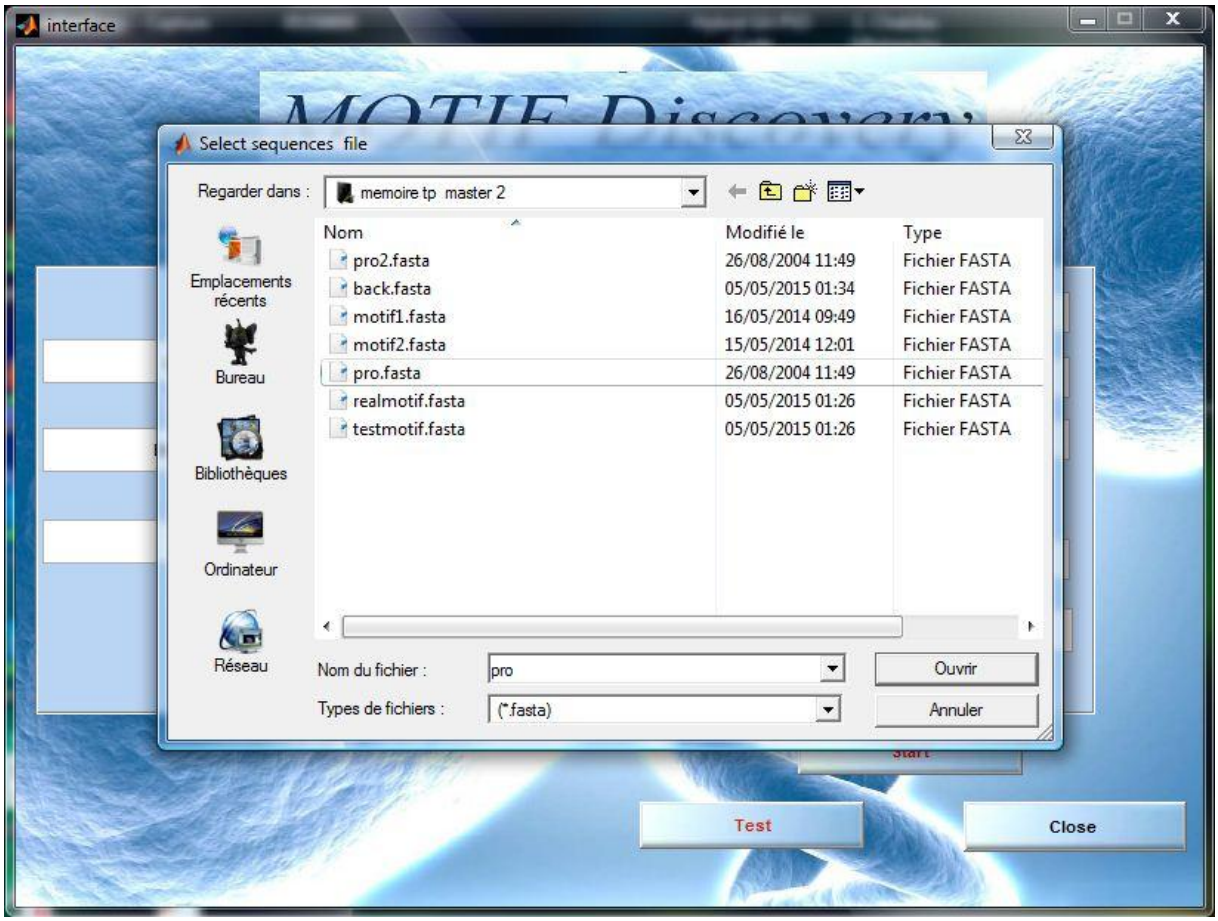


Figure 45: Chargement des fichiers des séquences

Les valeurs des instances du test sont représentées dans la figure ci-dessous :

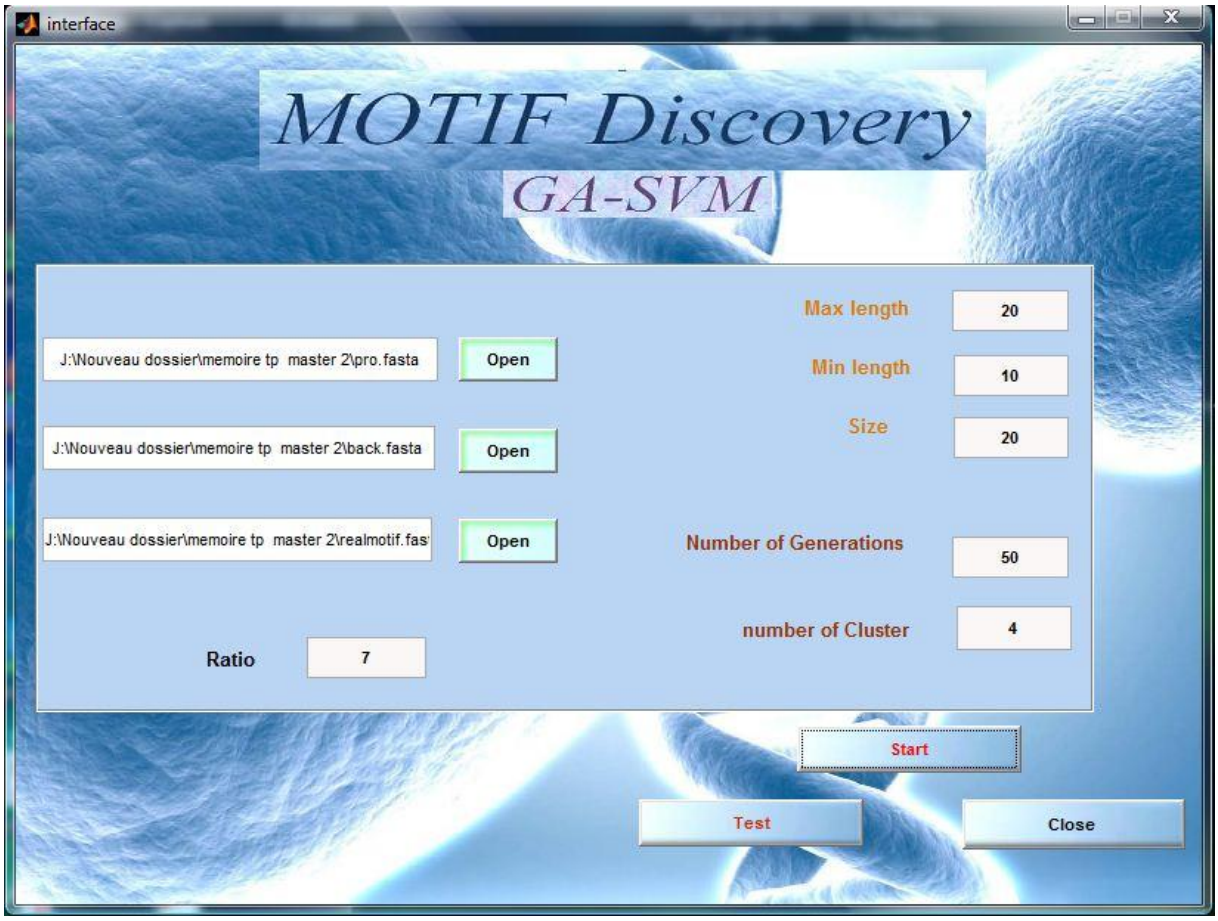


Figure 46: Introduction des Paramètres.

7. Tests et Résultats

Pour tester notre algorithme, On choisie un cas qui est le plus simple.

Dans ce test, on utilise une partie aléatoire d'ADN réel , des motifs réel (Collection des régions régulatrices vérifiés expérimentalement fonctionnels de **Drosophila**) sont insérés dans plus que la moitié de ces séquences dans des positions aléatoires.

En teste est ce que l'algorithme capable de détecter ces motifs .En calcule le pourcentage de simulation entres les résultats de AG-SVM et les motifs que nous avons insérer

l'ensemble des motifs réel insère :

```
>seq_0
TTGTGCGATTATAGT
>seq_1
AAAACGGCTTGCAAT
>seq_2
CCAAAGGATTTGTCC
>seq_3
ACAAAAAATTATGCG
>seq_4
ATCGCGGATTGAGGA
>seq_5
AGCCATGCTTAATTA
>seq_6
ACTAATGATCAACTT
>seq_7
AACAAATATTTGCAT
>seq_8
CATTTGACTTTTTGG
>seq_9
TGTCAAGGCTAAAGG
```

Pour calculer la performance de cet algorithme, on l'exécute plusieurs fois (10 fois pour ce test) et on calcule le pourcentage de **succès** de chaque test.

le pourcentage de **succès** égale a le pourcentage maximum de simulation entre chaque séquence des l'ensembles des motifs et chaque séquence de l'ensemble de résulta

On applique 10 essais et on calcule le pourcentage de succès du test.

Pourcentage de succès = la somme des pourcentages de succès /nombre totale des tests

L exécution d'un test avec 50 génération , nous a donné les résultat en bas avec un parentage de **succès** = 60%

GAAGATGTGGG
GTAGATGTGGG
GTAAGAAGCACAAG
GTAAGAAGCACTCG
AAGGTTAAAGTA
AGGGTTACAGTA
GATAGAGACTAAATGACTG
GGTAGAGCTTAATTGACTG
GATAGAGGCTAAATGACTG
GATAGAGTCTAAATGACTG
CCCGATGCGG
CCGGCTGCGG
CCCGGTGCGG
CCCGTTGCGG
AGACGCCTCCACC
ATACGCCTGCCGC
GCGGCAAACA
GCGGCTACCA
GTAAACGGTACAAGTAG
GTTCACGGTGCAAGTGG
GGTCTTTCGCGTGAATT
GGTCTTTCGCGTGACTT
CGATTAGCGCAGT
CGATTTGCGCAGT
CATCTATCGTCACGG
CTTCTATTGTCACGG
CACCTCTCAATCGCT
CACCTCTTGATTGCT
CCCACATGTCTCCTCGCC
CTCCCATTTCTTCTTGTC
CCCGCATGTCTCCTCGCC
CCCTCATGTCTCCTCGCC
GGGAGCTAAATGAAGGTTG
GTGAGCTAATTGAAGGTTG
ATATAAGGGCTATC
ATATTAGGGCTATC
GCCGTGTAAGCT
CAATACATTAAAATAATG
CAATGCATTATAATATTT
CAATACATTAAAATAATG
CAATGCATTATAATATTT
CAATACATTAAAATAATG
CAATGCATTATAATATTT

- **Pourcentage de succès**

```
numGenerat =

    50

per =

    60
```

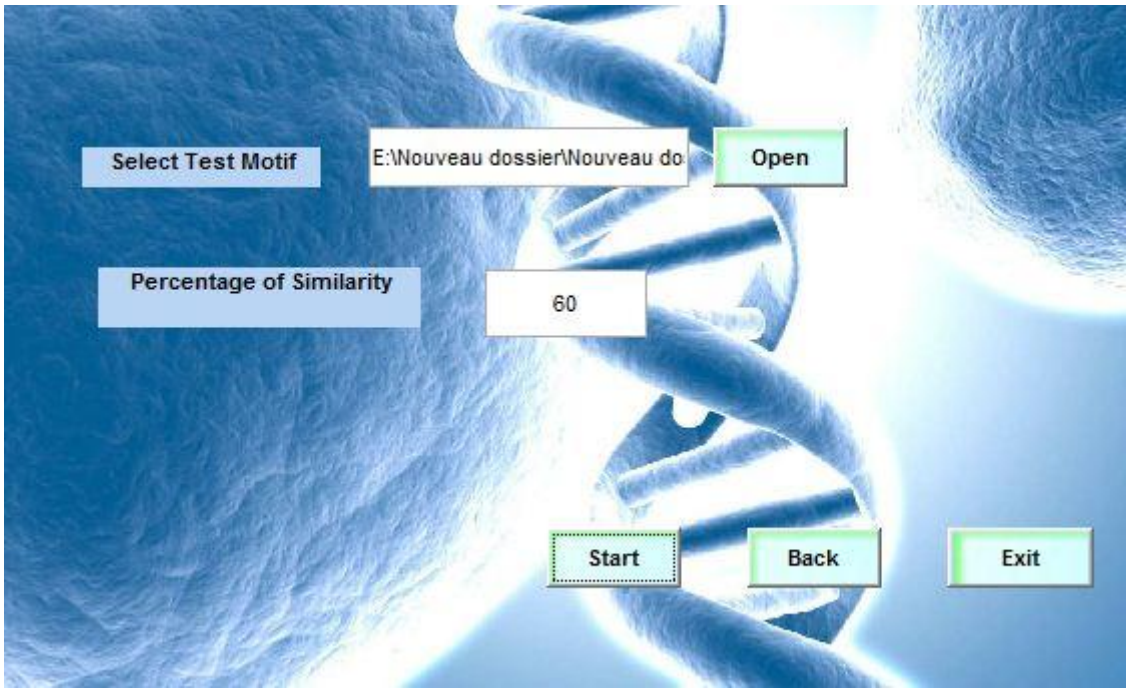


Figure 47: Pourcentage de succès

Le tableau suivant récapitule les résultats obtenus :

La longueur de séquence	Nombre d'essais	Nombre des générations	Nombre des séquences promoteur	Nombre des séquences d'arrière-plan	Pourcentage de succès
100	**10**	**50**	**50**	**50**	**6.5/10 = 65%**

Tableau 6: Les résultats des tests.

8. Comparaison entre découverte de motifs par GA et découverte de motifs par GA-SVM

Le travail de [72] sur le problème de découverte de motifs utilise un algorithme génétique simple. Notre algorithme utilise un algorithme hybride qui combine les algorithmes génétiques et SVM et cela pour améliorer les résultats et accélérer le temps de calcul.

Le tableau suivant illustre une petite comparaison des résultats obtenus des deux algorithmes :

Type d'algorithme	La longueur de séquence	Nombre d'essais	Nombre des générations	Nombre des séquences promoteur	Nombre des séquences d'arrière-plan	Pourcentage de succès
GA-SVM	**100**	**10**	**50**	**50**	**50**	**6.5/10 =65 %**
GA-SVM	**100**	**10**	**100**	**50**	**50**	**14/15 = 93%**
GA	**100**	**10**	**50**	**50**	**50**	**5/10 = 50%**
GA	**100**	**10**	**100**	**50**	**50**	**13/15 = 87%**

Tableau 7: comparaison GA et GA-SVM Les résultats des tests

9. Le temps d'exécution GA et GA-SVM :

- Pour l'algorithme GA-SVM, Nous constatons que la durée moyenne de l'exécution d'une seule génération avec les instances montré dans la table précédente est de : 30 seconds. Pour 50 générations l'logarithme prend un temps plus long qui est de : 25 minute.
- Pour l'algorithme GA, Nous constatons que la durée moyenne de l'exécution d'une seule génération avec les instances montré dans la table précédente est de : 42 seconds. Pour 50 générations l'logarithme prend un temps plus long qui est de : 35 minute.

Cela nous conduit à dire que l'algorithme GA-SVM est plus optimal que l'algorithme GA pour la découverte de motifs biologique. Il a apporté une rapidité au niveau temps de calculs, aussi un grand pourcentage en le comparant avec GA algorithme.

10.Conclusion

Apres avoir implémenté l'algorithme proposé, des tests ont été conduit pour valider sa performance.

Dans ce chapitre, nous avons fait des tests sur notre algorithme et décrit les résultats obtenus. Ensuite, une petite comparaison entre GA et GA-SVM est établie pour dégager la meilleure solution.

Conclusion Générale et perspectives

Conclusion Générale

L'évolution rapide de l'informatique fut associée à une montée éclatante de problèmes informatiques de tous genres. La technologie des ordinateurs s'est développée à un rythme effréné, rendant possible le traitement de problèmes de plus en plus compliqués. Mais les problèmes sont devenus tellement complexes qu'il n'est plus possible à l'être humain de compter uniquement sur la puissance de son ordinateur ni même sur celle de son super calculateur.

Généralement, un problème complexe peut être exprimé sous la forme d'un problème d'optimisation, dans lequel on définit une fonction objectif que l'on recherche à minimiser ou à maximiser selon le contexte. Différentes techniques de résolution ont été développées pour résoudre ses problèmes complexes.

Les méthodes hybrides ont prouvées leur efficacité pour résoudre des problèmes complexes divers, dans plusieurs domaines de recherche et d'ingénierie.

Le travail de ce mémoire a mis l'accent sur l'utilité de l'hybridation des méthodes (hybridation entre l'optimisation AG et l'apprentissage automatique SVM) pour la découverte des motifs ADN dans le domaine de la bioinformatique.

L'apport de l'hybridation est totalement bénéfique : solutions améliorées, convergence plus rapide, et minimisation de temps.

Perspectives

Au terme de ce travail, il parait tout à fait intéressant de suggérer des études complémentaires. En particulier de généraliser l'approche GA-SVM pour d'autres problèmes, et utilisé d'autres hybridation pour des problèmes dans le domaine de la bioinformatique par exemple PSO-GA-SVM pour une meilleure solution ou une solution optimale.

Aussi, une comparaison est envisagée entre les différentes méthodes qui existent pour bien extraire les avantages de chacune. Le résultat peut servir a une combinaison qui peut conduire a des taux de succès plus élevé.

Bibliographie

1. A. Cornuéjols, L. Miclet, Y.Kodratoff, « Apprentissage Artificiel, Concepts et algorithmes ».
2. V. Vapnik, "The Nature of Statistical Learning Theory". Springer Verlag, New York, USA, 1995.
3. . P.Vincen « Modèles à noyaux à structure locale », Thèse de Phd en informatique, Université de Montréal,2003.
4. Marref Nadia, Apprentissage Incrémental & Machines à Vecteurs Supports, Présenté en vue de l"obtention du diplôme de Magister en Informatique, Université HADJ LAKHDAR – BATNA,2013.
5. Kobdani, H. and Schütze, H. Sucre : A Modular System for Coreference Resolution. In Proceedings of the 5th International Workshop on Semantic Evaluation (ACL 2010), pages 92–95. (2010).
6. http://docs.happycoders.org/orgadoc/artificial_intelligence/classification_documents /Classification.pdf.
7. Adèle Désoyer, Apprentissage d'un modèle de résolution automatique de la coréférence à partir d'un corpus de français oral, Master Professionnel Spécialité
8. Documents Electroniques et Flux d'Informations, Université Paris Ouest - Nanterre La Défense,2014.
9. J. Andersson ; A Survey for Multiobjective Optimization in engineering Design; Rapport technique, LiTH-IKP-R-1097; 2000.
10. Rémy-Robert, Alexandre Joseph ; Systèmes Interactifs d'Aide à l'Élaboration de Plannings de Travail de Personnel Contraintes, Aide à la Décision, Représentation Combinatoire des Préférences, Équité et Résolution par Décomposition Arborescente et par Consistance; Université Joseph Fourier-Grenoble1, Science et Géographie ; Page 28; 07 Novembre 2003.
11. Cours optimisation chapitre 4 problèmes np-complets.
12. Alain Berro ; Optimisation Multiobjectif et Stratégie d'évolution en environnement Dynamique (thèse de Doctorat); Université des sciences sociales ToulouseI ; page 14, 27, 29; 18 Décembre 2001.
13. Mehdi Rouan Serik , implémentation de méthodes de recherches locales sur les architectures multi et many-cœurs. Application au problème d'affectation quadratique

à 3 dimension.(Diplôme de Magister) Université d'Oran Faculté des Sciences Département D'Informatique.

14. Amira Gherboudj, Méthodes de résolution de problèmes difficiles académiques (diplôme de Doctorat), Université de Constantine2,2012/2013.
15. Souhil MOUASSA, Optimisation de l'écoulement de puissance par une méthode métaheuristique (technique des abeilles) en présence d'une source renouvelable (éolienne) et des dispositifs FACTS,(Magister En Electrotechnique).2012.
16. Kirkpatrick, S.; Gelatt Jr, C. D.; Vecchi, M. P. (1983). "Optimization by Simulated Annealing". (1983).
17. R. Benabid , "Optimisation Multi-objectif de la Synthèse des FACTS par les Particules en Essaim pour le Contrôle de la Stabilité de Tension des Réseaux Electriques," Université de Laghouat mémoire de Magister, 2007.
18. A. Colorni, M. Dorigo, et V. Maniezzo. An investigation of some proprieties of an "ant algorithm". In Manner and Manderick, pages 509–520, 1992.
19. Sidi Mohamed Douiri, Souad Elbernoussi, Halima Lakhbab, Cours des Méthodes de Résolution Exactes Heuristiques et Métaheuristiques, MASTER CODES, CRYPTOGRAPHIE ET SÉCURITÉ DE L'INFORMATION, Université Mohammed V, Faculté des Sciences de Rabat.
20. S. Kemmoé , L. Deroussi, M. Gourgand et A. Quilliot ; Des Essaims Particulaires Efficaces Pour l'optimisation Combinatoire ; Congrès du GDR MACS, Pôle STP ; Octobre 2004.35 P. Moscato, « Memetic algorithms : A short introduction », in D. Corne M. Dorigo et F. Glover, editors, new ideas in optimization, pages 219-234, McGraw-Hill, New York, 1999.
21. Laurent Noé. Recherche de similarités dans les séquences d'ADN : modèles et algorithmes pour la conception de graines efficaces.2005.
22. Pr. C. Housset et Pr. A. Raisonnier. Cours Biologie Moléculaire Université Pierre et Marie Curie. 2009-2010.
23. Pr. Alami – Ouahabi Naïma. cours Biologie Moléculaire 1ère ANNEE PHARMACIE. 2012.
24. HISTOIRE DE BIOLOGIE cours 1ére année LMD SNV.
25. Les applications industrielles de la bio-informatique1Jean-Philippe Vert Directeur du Centre de Bio-informatique à MINES ParisTech .

26. Gérard Ramstein. Application de techniques de fouille de données en Bio-informatique. Ecole Centrale de Nantes Université de Nantes Ecole des Mines de Nantes.
27. Gael Le Mahec Gestion des bases de données biologiques sur grilles de calcul 2008.
28. Zakia CHALLAL . MAGISTER LA RÉPLICATION DE DONNÉES DANS LES GRILLES BIOLOGIQUES.
29. http://biochimej.univangers.fr/Page2/BIOINFORMATIQUE/7ModuleBioInfoJMGE/6 BasesDo nees/1BasesDonnees.htm.
30. European Agency for the Evaluation of Medicinal Products (EMEA), Position Paper in Terminology in Pharmacogenetics, Londres, European Agency for the Evaluation of Medicinal Products, p. 3,2002.
31. Franck Serusclat , Rapport 20, Génomique et informatique : l'impact sur les thérapies et sur l'industrie pharmaceutique, Paris, Office parlementaire d'évaluation des choix scientifiques et technologiques, 2000 .
32. Yann Joly, "Accès aux médicaments : le système international des brevets empêchera t-il les pays du tiers monde de bénéficier des avantages de la pharmacogénomique", 1 Les cahiers de la propriété intellectuelle 138, 2003.
33. Urs A. Meyer, Introduction to Pharmacogenomics : Promises, Opportunities, and Limitations, Dans : Julio Licinio et Ma-Li Wong (dir.), Pharmacogenomics – The Search for Individualized Therapies,Weinheim, Wiley-VCH, p. 5,2002.
34. http://wwwabi.snv.jussieu.fr/erocha/webthese/genomique.html
35. THOMAS DERRIEN. L'analyse comparée des génomes : applications à l'identification de nouveaux gènes canins. Institut de Génétique et Développement - UM6061 - CNRS – Rennes.
36. D.Gillain-B.Wiesen.cours d'informatique.DES Imagerie Médicale 2013-2014.
37. Pattern Discovery: Methods and Software, Brona Brejova, Tomas Vinar, Ming Li.
38. bioinformatics.oxfordjournals.org/content/25/12/i356.full.
39. A. MANCHERON, THÈSE DE DOCTORAT: Extraction de Motifs Communs dans un Ensemble de Séquences.
40. Biochimie Par Reginald H. Garrett,Charles M. Grisham.
41. Jean Valéry TURATSINZE, Evaluation des outils de prédiction d'éléments cis-régulateurs dans les génomes de mammifères, Mémoire présenté en vue de l'obtention du grade de licencié en Sciences Biologiques, Année académique 2004-2005.

42. Élodie Darbo. Découverte d'éléments cis-régulateurs impliqués dans l'activation transcriptionnelle du génome zygotique dans l'embryon précoce de drosophila melanogaster, l'Université de la Méditerranée.
43. Baldi,P. and Brunak,S. Bioinformatics: the Machine Learning Approach MIT Press, 2001.
44. G. D. Forney, Jr., "The Viterbi algorithm," Proc. IEEE, vol. 61, pp. 268–278, March 1973.
45. Talbi, E.G., S. Cahon & N. Melab, Designing cellular networks using a parallel hybrid metaheuristic, Journal of Computer Communications 30(4): 698-713, 2007.
46. Talbi, E.-G. Metaheuristics: From Design to Implementation, Wiley 2009.
47. Duvidier, Etude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop, Thèse de Doctorat, université du littoral France, décembre 2000.
48. J. K. Hao, P. Galinier & M. Habib, Metaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. Revue d'Intelligence Artificielle, vol. 13 (2), pp. 283-324, 1999.
49. C. Grosan & A. Abraham, Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews, Studies in Computational Intelligence (SCI) 75, pp. 1–17, 2007.
50. K.C. Tan, Q. Yu, C.M. Heng & T.H. Lee, Evolutionary computing for knowledge discovery in medical diagnosis, Artificial Intelligence in Medicine, 27(2), pp. 129-154, 2003.
51. L. Wang, A hybrid genetic algorithm-neural network strategy for simulation optimization, Applied Mathematics and Computation, 170(2), pp. 1329-1343, 2005.
52. A.A. Javadi, R. Farmani & T.P. Tan, A hybrid intelligent genetic algorithm, Advanced Engineering Informatics, pp. 255-262, 2005 .
53. M.A. Lee & H. Takagi, Dynamic control of genetic algorithms using fuzzy logic techniques, In Forrest S (Ed.), Proceedings of the 5th International Conference on Genetic Algorithms, Morgan Kaufmmann, San Mateo, pp. 76–83, 1993.
54. X. H. Shi, Y. C. Liang, H.P. Lee, C. Lu & L. M. Wang: An improved GA and a novel PSO-GA-based hybrid algorithm. Inf. Process. Lett. 93(5): 255-261, 2005.
55. E. A. Grimaldi, F. Grimacia, M. Mussetta , P. Pirinoli & R. Zich , A new hybrid genetical-swarm algorithm for electromagnetic optimization, In Proceedings of

International Conference on Computational Electromagnetics and its Applications, Beijing, China, pp. 157-160, 2004.

56. X. Wang, X. Z. Gao & S. J. Ovaska, A novel particle swarm based method for nonlinear function optimization, International Journal of Computational Intelligence Research. ISSN 0974-1259 vol.4, no.3, pp. 281-289, 2008.
57. I. Dimistrescu, & T. Stutzle , Combinations of local search and exact algorithms, Lecture Notes in Computer Science 2611: 211-223, 2003.
58. T. Radha , P. Millie , A. Ajith & B. Pascal, "Particle Swarm Optimization: A Survey on Hybridization Perspectives", In Applied Mathematics and Computation, Elsevier Science, vol. 217, no. 12, pp. 5208 - 5226, 2011.
59. H. Talbi & M. Batouche, Hybrid particle swarm with differential evolution for multimodal image registration, in Proc. IEEE International conference on industrial Technology, vol. 3, pp. 1567-1572, 2004.
60. M. Yannis, M. Magdalene & D. Georgios, A hybrid particle swarm optimization algorithm for the vehicle routing problem. Engineering Applications of Artificial Intelligence 23, pp. 463-472, 2010.
61. Bernhard Scholkopf, Alexander J. Smola “Learning with Kernels, Support Vector Machines, Regularization, Optimization, and Beyond”, the MIT Press 2002
62. Vojislav Kecman, “Learning and Soft Computing Support Vector Machines, Neural Networks, and Fuzzy Logic Models”, the MIT Press 2001
63. L. Bottou et al. “Comparison of classifier methods: a case study in handwritten digit” recognition. Proceedings of the 12th IAPR International Conference on Pattern Recognition, vol. 2.
64. History of Support Vector Machines [en ligne]. <http://www.svms.org/history.html> (9/11/2012) .
65. Colin Campbell, Yiming Ying “Learning with Support Vector Machines SYNTHESIS LECTURES ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING #10”, Morgan & Claypool publishers 2011.
66. H.Mohamadally et B.Fomani , " SVM : Machines à Vecteurs de Support ou Séparateurs à Vastes Marges" Versailles St Quentin, France . janvier 2006.
67. S.Khellat Kihel, « Les séparateurs à vaste marge Bi- classe »,Université des sciences et de technologie d"Oran, exposé de Master2, 2012.

68. J. Callut, «Implémentation efficace des Support vector Machines pour la classification » Mémoire présenté en vue de l‟obtention du grade de Maître en informatique. Université libre de Bruxelles . département informatique, 2003.
69. J.H. Holland. Genetic Algorithms and the optimal allocation of trials, SIAM Journal of Computing. Vol. 2, N° 2, pp. 88-105, 1973.
70. C. Darwin. The origin of species by means of natural selection. Mentor Reprint, New York, 1859.
71. K. Ghali. Méthodologie de conception système a base de plateformes reconfigurables et programmables. Thèse de doctorat. Université Paris XI, France. 2005.
72. Djenhi Wiame, Benzebouchi Imene, Classification en bioinformatique par des approches basées sur l'intelligence computationnelle : Découverte de Motifs par les Algorithmes Génétiques, Mémoire préparé En vue de l'obtention du diplôme de Master 2014.
73. Edmundo Bonilla Huerta, Béatrice Duval, et Jin-Kao Hao, A Hybrid GA/SVM Approach for Gene Selection and Classification of Microarray Data. LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers, France {edbonn, bd, hao}@info.univ-angers.fr.