République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mentouri de Constantine
Faculté des sciences de l'Ingénieur
Département d'Informatique

N° d'ordre :

Série :

Mémoire
Présenté en vue de l'obtention du diplôme de

Magistère en Informatique

Option : Génie Logiciel et Intelligence Artificielle

Optimisation Multi-Objectif Pour l'Alignement Multiple de Séquences

Présenté par :
Nadira Benlahrache

Dirigé par :
Dr. S. Meshoul

Soutenu le **:** …./…./2007

devant le jury d'examen composé de :

Dr. D.E. Saidouni	Université Mentouri de Constantine	*Président*
Dr. A. Chaoui	Université Mentouri de Constantine	*Rapporteur*
Dr. M.K. Kholladi	Université Mentouri de Constantine	*Examinateur*
Dr. N. Zarour	Université Mentouri de Constantine	*Examinateur*

Abstract

The bioinformatics is a discipline which aims at the automatic treatment of biological information. The Multiple Sequences Alignment (MSA) constitutes a fundamental task for many applications into bioinformatics. In order to evaluate a given MSA, several objective functions were defined realising mathematical formulations. However all the suggested methods of alignment in the literature rests on approaches of mono-objective optimization and provide only one potential solution. None of these methods can claim optimality when it is a question of providing to all the correct blows of MSAs to the biological direction. In this dissertation, we investigate the idea to reconsider this problem under a multi-objective vision. The strategy that we propose is a multi-objective genetic algorithm which operates on a whole of alignments whose evolution is ensured by genetic operators and leading to a whole of alignments of better quality. The results obtained are promising and highlight the contribution of multi-objective optimization for the resolution of the problem of the MSA.

Keywords: Multiobjective Optimization, Bioinformatics, Multiple Sequences Alignment, Multiobjective Genetic Algorithm, Pareto Dominance.

Résumé

La bioinformatique est une discipline qui vise le traitement automatique de l'information biologique. L'alignement multiple de séquences (MSA) constitue une tâche fondamentale pour beaucoup d'applications en bioinformatique. Afin d'évaluer un MSA donné, plusieurs fonctions objectif ont été définies moyennant des formulations mathématiques. Cependant toutes les méthodes d'alignement proposées dans la littérature reposent sur des approches d'optimisation mono-objectif et fournissent une seule solution potentielle. Aucune de ces méthodes ne peut prétendre l'optimalité quand il s'agit de fournir à tous les coups des MSAs corrects au sens biologique. Dans ce mémoire, nous investiguons l'idée de reconsidérer ce problème sous une vision multi-objectif. La stratégie que nous proposons est un algorithme génétique multi-objectif qui opère sur un ensemble d'alignements dont l'évolution est assurée par des opérateurs génétiques et conduisant à un ensemble d'alignements de qualité meilleure. Les résultats obtenus sont prometteurs et mettent en évidence l'apport de l'optimisation multi-objectif pour la résolution du problème du MSA.

Mots clés : Optimisation multi objectif, Bioinformatique, Alignement Multiple de Séquences, Algorithme Génétique Multi objectif, Dominance au Sens Pareto.

Remerciements

Mes vifs remerciements vont à Mme S. MESHOUL, mon encadreur, pour m'avoir accueillie, soutenue et encadrée constamment pendant ces deux années de travail.

Je tiens également à remercier Dr A. Chaoui d'avoir accepter d'être le rapporteur de ce mémoire et qu'il trouve ici ma sincère reconnaissance.

Mes remerciements vont ensuite aux membres de jury : D.E. Saidouni, M.K. Kholladi et N. Zarour ; Maîtres de conférence à l'université Mentouri de Constantine, pour avoir accepté de participer au jury et d'évaluer ce travail.

Merci à tous les enseignants du département de l'informatique qui ont participé à ma formation.

Je tiens à remercier tout le personnel du département de l'informatique et particulièrement ceux du laboratoire LIRE.

Je remercie également Mr A. Layeb, pour ses conseils et son aide précieuse.

Ces remerciements ne seraient pas complets sans y avoir associé toute ma famille et belle famille pour leur encouragement et leur intérêt à tous ce que fais. Je remercie particulièrement mon mari pour son soutien moral, sa présence et son écoute. Il a su m'épauler et me faire confiance pendant toutes ces années.

Je ne pourrais oublier mes amies qui m'ont accompagnée et encouragée durant cette période.

Mme Nadira Benslama
Née Benlahrache

Liste des Matières

INTRODUCTION GÉNÉRALE

Introduction

La disponibilité d'une grande quantité d'information sur les séquences d'ADN et en particulier sur les génomes complets de plus de 200 espèces, a ouvert une nouvelle ère dans l'histoire de la biologie moléculaire. Les banques de données dédiées connaissent une croissance rapide et fructueuse. Les ambitions des biologistes et leur curiosité augmentent au même rythme que les découvertes biologiques se succèdent. Les souhaits d'un biologiste actuel sont :

1. Analyser, comprendre et organiser une masse de données biologiques
2. Décoder l'information contenue dans les séquences d'ADN et de protéines
3. Modéliser les structures 3D des protéines et des ARNs structurels et déterminer la relation entre structure et fonction
4. Étudier la régulation des gènes et déterminer les réseaux d'interaction entre les protéines.

Contexte De L'étude

La Bioinformatique se propose comme une science capable de fournir des moyens et des outils pour apaiser la soif des biologistes. La bioinformatique est un domaine pluridisciplinaire où l'informatique joue un rôle prépondérant. C'est une science qui conceptualise la biologie en termes de molécules et applique des " techniques d'informatiques" pour modéliser, analyser, comparer et simuler l'information biologique incluant séquences, structures, fonctions et phylogénie. En bref, la bioinformatique est un système intégré de gestion pour la biologie moléculaire et a beaucoup d'applications pratiques. L'alignement multiple de séquences ou MSA (pour **M**ultiple **S**equence **A**lignment) est un problème fondamental en biologie moléculaire et représente une tâche de base pour beaucoup d'applications en bioinformatique. Il vise à apparier au sens biologique plusieurs séquences nucléiques et protéiques. MSA est le moyen utilisé par les biologistes pour analyser des séquences d'ADN (nucléiques) ou de protéines (protéiques) afin de déterminer leur degré d'homologie ou de divergence. MSA est utilisé pour la construction d'arbres phylogénétiques et pour l'identification des motifs dans des familles de protéines permettant ainsi la prédiction de leur aspect structurel et fonctionnel. Cependant, cette extrême importance de l'alignement multiple de séquences est confrontée à l'extrême difficulté de sa résolution. La raison en est que la recherche d'un alignement de bonne qualité implique souvent l'exploration d'espaces de recherche très vastes et dont la taille devient de plus en plus critique avec le nombre et les tailles des séquences à aligner. Cependant trouver un alignement multiple a été démontré un problème NP-complet, MSA ne peut être résolu par une méthode exacte que pour des séquences de petites tailles et dont le nombre est réduit induisant des espaces de tailles réduites.

Durant la dernière décennie, plus de cinquante méthodes ont été proposées dans ce domaine. Ce nombre risque d'augmenter les prochaines années du fait que le problème reste toujours ouvert et non complètement résolu. Globalement, en excluant les méthodes exactes qui sont capables de fournir un alignement optimal mais elles sont appropriées que pour un nombre très réduit de séquences, les méthodes décrites dans la littérature peuvent être groupées en deux classes : les méthodes progressives et les méthodes itératives telles que ClustalW, SAGA, DIALIGN, T_Coffee, MAFFT, Muscle, Align_M, ProbCons… etc. Étant donné un ensemble de séquences, les méthodes progressives basées sur une approche proposée par Feng et Doolittle, se proposent

d'effectuer l'alignement en partant des séquences les plus similaires et en ajoutant les séquences restantes graduellement une par une selon un ordre pré-établi par un arbre phylogénétique. Elles utilisent souvent la programmation dynamique. La simplicité et la rapidité de traitement sont les avantages majeurs de ces méthodes. Cependant, leur nature gloutonne conduit souvent à des solutions sous optimales. Ce qui explique le recours aux méthodes itératives pour gérer la complexité combinatoire du problème. Leur principe de base consiste à produire un alignement initial et à le raffiner itérativement de manière déterministe ou stochastique.

Problème Posé

L'alignement fourni par ces méthodes est celui qui optimise une fonction objectif choisie ou fonction score. Formulée mathématiquement, cette fonction permet une évaluation quantitative de la signification biologique et évolutionnaire d'un alignement. Sa valeur doit indiquer la relation entre les séquences du point de vue structure et évolution. Le choix d'une fonction objectif est une tâche très délicate. Ceci est justifié par la difficulté même de définir une fonction objectif qui capture fidèlement toute l'information biologique exhibée par un alignement donné. En d'autres termes, comment s'assurer mathématiquement qu'un alignement est correct biologiquement ? Ceci explique l'apparition d'un nombre non négligeable de fonctions objectif telles que *SP* (Sum of Pairs), *WSP* (Weighted Sum of Pairs), *Coffee*, *Consensus*, et *Entropie, etc*. Ces fonctions ont des caractéristiques différentes et permettent d'évaluer les alignements selon des aspects différents. Elles permettent, à des degrés variés, de s'approcher de l'optimum biologique. Cependant l'optimum mathématique ne coïncidant pas souvent avec l'optimum biologique, les questions qu'on peut se poser : Est-il suffisant de se contenter d'une seule fonction score pour chercher un alignement ? N'est-il pas plus intéressant de guider la recherche par plusieurs fonctions objectif afin de tirer le maximum de leur degré de signification biologique, de leur complémentarité voire même des éventuels conflits qu'elles peuvent engendrer ?

Voie De La Recherche

Indépendamment de la bioinformatique, les algorithmes évolutionnaires constituent un vaste champ de recherche. Ce sont des algorithmes qui concrétisent la politique de l'évolution introduite par Darwin. Les algorithmes évolutionnaires sont donc des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle, à savoir les croisements, les mutations, la sélection, etc. L'algorithme évolutionnaire est caractérisé par une population de solutions candidates et un processus de reproduction qui permet combiner les solutions existantes pour produire de nouvelles solutions. Puis la sélection détermine quels individus de la population courante participent dans la nouvelle population. Ce processus est répété plusieurs fois jusqu'à convergence vers des solutions optimales.

Dans le contexte de l'optimisation multi-objectif (MOO), un certain nombre algorithmes évolutionnaires multi-objectif (Multiobjective Evolutionary Algorithm : MOEA) ont été proposés ces dernières années et l'intérêt croissant pour ces méthodes a motivé leur extension à l'origine proposés pour l'optimisation mono-objectif aux variantes multi-objectif.
Les MOEAs sont capables d'optimiser plusieurs fonctions objectif à la fois et fournissent en fin d'exécution un ensemble de solutions potentielles dit l'ensemble optimal.
Ils se caractérisent par leur parallélisme inhérent et leurs possibilités à exploiter des similitudes des solutions par recombinaison, les MOEAs peuvent rapprocher l'ensemble des solutions optimales en une seule exécution. Ils sont très bien adaptés au traitement d'un problème

d'optimisation multi-objectif. Ce domaine est très dynamique et ne cesse de se développer. Il y a plusieurs versions des MOEAs, chacune essaye d'apporter plus d'efficacité à la MOO et de trouver des solutions les plus proches de l'optimalité. Les algorithmes évolutionnaires multi-objectif ont une dynamique itérative. Ils démarrent par une ou plusieurs solutions initiales puis effectuent une série de raffinements à l'aide des opérateurs de modification pour obtenir des solutions meilleures. Les solutions obtenues par les algorithmes évolutionnaires multi-objectif ne sont pas forcément optimales mais elles sont proches de l'optimalité.

Idée De Base

Comme souligné précédemment, les méthodes proposées dans la littérature, reposant sur une seule fonction objectif fournissent des évaluations qui ne sont pas toujours concluantes. Dans ce mémoire, nous investiguons l'idée de considérer plusieurs fonctions objectif simultanément et nous proposons une stratégie de recherche qui vise à identifier un alignement de bonne qualité par optimisation multi-objectif. L'optimisation multi-objectif apparaît comme un cadre naturel pour mener cette étude. La motivation de ce travail est double. D'une part, il s'agit de tirer profit de l'aspect complémentaire et/ou conflictuel des fonctions d'évaluation des alignements et d'aboutir à un bon compromis améliorant la qualité des solutions obtenues et d'autre part, avoir la possibilité d'obtenir plusieurs solutions potentielles donnant ainsi plus de choix au biologistes au stade de prise de décision. En exploitant cette nouvelle idée, nous proposons dans ce mémoire une approche basée sur un algorithme évolutionnaire multi-objectif pour la résolution du problème multi-objectif. Pour cela, nous définissons un schéma de représentation pour coder les alignements, des opérateurs appropriés pour faire évoluer une population initiale et une stratégie de sélection permettant de maintenir les bonnes solutions obtenues au cours de l'évolution. L'algorithme évolutionnaire multi-objectif proposé opère sur un ensemble d'alignements dont l'évolution est assurée par des opérateurs génétiques appropriés et conduisant à un ensemble d'alignements de bonne qualité après un certain nombre d'itérations.

Organisation Du Manuscrit

Dans un souci de clarté, le reste du mémoire est organisé comme suit : le chapitre 1 introduit le domaine de la biologie moléculaire et la bioinformatique. Le chapitre 2 fournit une description du problème de l'alignement multiple de séquences. Le chapitre 3 présente les concepts de base de l'optimisation multi objectif. Le chapitre 4 sera entièrement consacré à l'introduction des algorithmes évolutionnaires multi-objectif. Le chapitre 5 est dédié à la description de l'approche proposée et les résultats expérimentaux. Le mémoire s'achèvera par une conclusion et des perspectives.

Chapitre 1 : Introduction à la Bioinformatique

1.1 Introduction

La biologie moléculaire est un domaine dont l'évolution est permanente. Les grandes quantités de nouvelles données sont produites quotidiennement. Dans seul le projet humain de génome (HGP : Humain genome project), les scientifiques ont séquencé trois milliards de nucléotides de notre structure génétique.
Ce projet a été officiellement lancé par les américains en octobre 1990, sous la commande commune du ministère de l'énergie et des instituts nationaux de la santé. D'après l'institut national de recherche du génome humain (http:\\www.nhgri.nih.gov\HGP\) le but du projet est: «*de construire les cartes génétiques et physiques détaillées du génome humain, pour déterminer la séquence complète de nucléotides de l'ADN humaine, de localiser les 50.000-100.000 gènes estimés dans ce génome, et d'effectuer les analyses semblables sur les génomes de plusieurs autres organismes utilisés intensivement dans les laboratoires de recherches en tant que modèles*. »
Peu après le lancement du programme américain, l'intérêt est devenu international puisque plusieurs autres pays (Royaume-Uni, France, Japon, Canada etc.) se joignent au projet. Le programme initial réclamait l'accomplissement du génome humain pour l'année 2005. Mais l'objectif fut atteint en 2003.

Pourquoi cherche-t-on à séquencer le génome humain ? Quel est donc l'intérêt ?

1.2 Génome et Génomique

Le *génome* d'un organisme vivant constitue l'information génétique qui permet à cet organisme de vivre et d'évoluer. Il contient toute l'information génétique nécessaire au fonctionnement de la cellule et par conséquent de tout l'organisme.
La *génomique* est la science qui a pour but l'étude exhaustive des génomes, elle constitue actuellement un défi scientifique important sur plusieurs plans.
La génomique permet d'étudier l'ensemble des gènes, d'une espèce donnée, leur fonction, leur rôle ainsi que leur répartitions sur les chromosomes et les relations entre eux.
Un génome séquencé est un texte formé de quatre lettres (A, C, G, T), il reste un énorme travail de décryptage pour pouvoir interpréter ce texte et d'explorer les structures et les processus moléculaires qui sont fondamentaux à la vie [Rocha, 00].
En gros trois tâches restent à réaliser :

- Identification des gènes et leur fonction
- Compréhension des réseaux d'interactions moléculaires.
- Comparer ce génome à celui des autres espaces.

Les intérêts d'un tel travail sont majeurs:

- Évolution des espèces (la théorie de l'évolution)
- Fonctionnement des cellules : comprendre les mécanismes de régulation des gènes.
- Médecine : identifier les gènes qui provoquent des maladies et expliquer les causes des maladies complexes.
- Étude de la propagation des maladies.
- Pharmaceutique : aide à la conception des remèdes et des traitements.

- Écologie : préservation de la faune et de la flore.
- Nutrition : Organismes Génétiquement Modifiés (OGM)

1.3 Défis de la Biologie Moléculaire

La disponibilité d'une grande quantité d'information sur les séquences d'ADN et en particulier sur les génomes complets de plus de 200 espèces, a ouvert une nouvelle ère dans l'histoire de la biologie moléculaire. Les banques de données dédiées connaissent une croissance rapide et fructueuse (Figure 1.1). Les ambitions des biologistes et leur curiosité augmentent au même rythme que les découvertes se succèdent.

Ce que souhaite connaître un biologiste moderne:

- Le jeu complet et précis des gènes ainsi que leur position sur le génome
- Le lieu et le moment de l'expression de chaque gène
- La protéine produite par chaque gène
- Le lieu et le moment de l'expression de chaque protéine
- La structure complète et la fonction de chaque protéine
- Les mécanismes cellulaires auxquels participent les protéines

Afin d'apaiser la soif du savoir des biologistes, toute approche de recherche doit être capable de soulever des défis énormes [Luscombe et autres, 01] tels que:

5. Analyser, comprendre et organiser une masse de données biologiques:
    - Plus de 200 génomes complètement séquencés et publiés, dont l'homme (23 paires de chromosomes) et la souris (20 paires de chromosomes). (Voir Table 1.1)
    - Projets de séquençage de plus de 500 **procaryotes** (organismes pluricellulaire avec noyau) et 400 **eucaryotes** (organisme unicellulaire sans noyau).
6. Décoder l'information contenue dans les séquences d'ADN et de protéines :
    - Trouver les gènes
    - Différencier entre introns et exons (voir Figure 1.18)
    - Analyser les répétitions dans l'ADN.
    - Identifier les sites des facteurs de transcription.
    - Étudier l'évolution des génomes.
7. Génomique structurelle:
    - Modéliser les structures 3D des protéines et des ARNs structurels
    - Déterminer la relation entre structure et fonction
8. Génomique fonctionnelle :
    - Étudier la régulation des gènes
    - Déterminer les réseaux d'interaction entre les protéines.

Organisme	chromosomes	Taille du génome (bp
Bactéries	1	400,000 à 10,000,000
Levure	12	14,000,000
Mouche	4	300,000,000
Homme	46	6,000,000,000

Table 1.1 : *Exemples de génomes déjà séquencés*

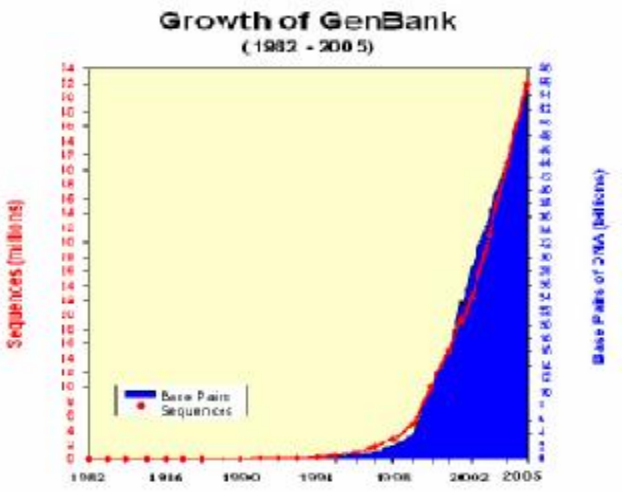


Figure 1.1 : *Évolution des banques de données*

Afin de confronter tous ces défis et répondre aux besoins des biologistes, plusieurs approches sont utilisées pour étudier et analyser les informations contenues dans un génome parmi elles : **la Bioinformatique.**

1.4 La Bioinformatique

Définition : *bio - informatique: science qui conceptualise la biologie en termes de molécules (dans le sens de la chimie-physique) et applique des " techniques d'informatiques " pour comprendre et organiser l'information liée à ces molécules, sur une grande échelle. En bref, la bioinformatique est un système intégré de gestion pour la biologie moléculaire et a beaucoup d'applications pratiques* [Luscombe et autres, 01].

Le mot « bioinformatique » découle donc de l'analyse par ordinateur des données biologiques. Ces données représentent l'information stockée dans le code génétique, mais également des résultats expérimentaux de diverses sources et des statistiques, … etc.
La bioinformatique est une science récente qui évolue rapidement et qui est fortement interdisciplinaire, elle conjugue plusieurs sciences telles que la biologie moléculaire, l'informatique, et les mathématiques (statistiques)… etc. Le but de la recherche dans la bioinformatique est l'organisation et l'extraction des données, la mise en application des algorithmes complexes et le développement des outils de visualisation afin d'atteindre une compréhension exhaustive et une exploitation des informations contenues dans les séquences d'un génome.
L'histoire du calcul dans la biologie moléculaire n'est pas récente mais date des années 20 où les scientifiques pensaient déjà à établir des lois biologiques par induction. Cependant, seulement le développement des ordinateurs puissants, et la disponibilité des données expérimentales qui peuvent être aisément traitées par calcul (par exemple, les séquences d'ADN ou d'acide aminé et des structures tridimensionnelles des protéines) ont lancé la bioinformatique comme un domaine indépendant. Aujourd'hui, les applications pratiques de la bioinformatique sont aisément disponibles sur le Web, et sont largement répandues dans la recherche biologique et médicale.
Le rapport entre l'informatique et la biologie moléculaire est normal pour plusieurs raisons. D'abord, le taux phénoménal de données biologiques produites fournit des défis: des quantités massives de données doivent être stockées, analysées, et doivent être rendues accessibles

[Rocha, 00]. En second lieu, les données sont souvent exprimées comme des formules statistiques, et par conséquent le calcul, est nécessaire. Ceci s'applique en particulier aux informations sur la construction des protéines et de l'organisation temporelle et spatiale de leur expression dans la cellule, troisièmement il y a une analogie forte entre la séquence d'ADN et un programme machine ; une séquence d'ADN représente une machine de Turing [Luscombe et autres, 01].
Mais l'analyse bioinformatique est également appliquée à de diverses autres données, par exemple arbres phylogéniques, les réseaux métaboliques, et les statistiques [Vert, 05]. Une myriade de techniques sont employées, y compris l'alignement de séquences primaires, alignement de la structure 3D de la protéine, la construction d'arbre phylogénétique, la prédiction et la classification de la structure de protéine, la prédiction de la structure d'ARN, la prédiction de la fonction de protéine, et l'expression de données groupées. Le développement algorithmique occupe une partie importante dans la bioinformatique. Des algorithmes complexes ont été spécifiquement développés pour l'analyse et l'accès aux données biologiques, par exemple : l'algorithme de programmation dynamique pour l'alignement de séquence, les programmes d'interrogations des bases de données biologiques tels que : BLAST [Altschul et autres, 90], FASTA [Pearson et Lipman, 88]

La Bioinformatique a un grand impact sur la recherche biologique. Les projets de recherche géants tels que le projet humain de génome, seraient sans signification sans la composante bioinformatique. Une fois que les données brutes sont disponibles, des hypothèses peuvent être formulées et évaluées *in silico* [Vert, 05]. De cette manière, les expériences menées par ordinateur peuvent répondre aux questions biologiques qui ne peuvent pas être abordées par des approches traditionnelles. Ceci a mené à la fondation des laboratoires de recherche dédiés seulement à la bioinformatique

Cette science peut être définie sur trois axes : Acquisition et organisation des données biologiques, conception des logiciels pour l'analyse, la comparaison et la modélisation des données et le dernier axe est l'analyse des résultats produits par les logiciels. Les thèmes traités par la bioinformatique sont :

- Modélisation et représentation de la connaissance en base de Données
- Méthodes de comparaison de chaîne de caractères comme recherche mots et des textes
- Algorithmes et techniques d'alignement de séquences et alignement multiple de séquences.
- Identification de motif et modèle pour des séquences multiples
- Analyse et interprétation : Techniques de data-mining (la fouille des données).
- Représentation graphique des surfaces et des volumes, et comparaison structurelle 3D
- Simulations moléculaires.
- Les analyses statistiques afin de fournir une mesure objective pour la signification des résultats.
- Réalisation des interfaces web pour faciliter l'accès aux banques de données à travers le monde.

Afin de pouvoir comprendre et assimiler les thèmes traités par la bioinformatique, il devient nécessaire de présenter quelques notions de la biologie moléculaire mais sans entrer dans des détails métaboliques et physico-chimiques.

1.5 La Biologie Moléculaire pour un Bio-informaticien

1.5.1 L'ADN (Acide Désoxyribonucléique)

La Figure 1.2 montre un schéma abstrait d'une cellule. Il y a **un noyau** contenant l'ADN. **Les protéines** sont à l'intérieur de la cellule mais en dehors du noyau. Les acides nucléiques, y

compris l'ADN et l'ARN, forment le matériel génétique de tout l'organisme. Ce sont toutes les informations de quoi a besoin un organisme pour fonctionner ainsi que toutes les caractéristiques héréditaires. Ce sont des molécules structurées en chaîne, composées **des nucléotides**

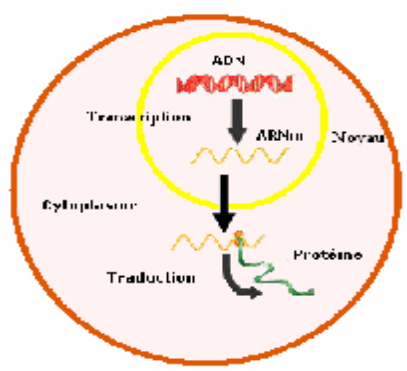


Figure 1.2 : Schéma simplifié d'une cellule

Un *nucléotide* d'ADN (Figure1.3) a 3 composants: un sucre (désoxyribose), un composant d'acide phosphorique (phosphate), et une base d'azote (un des quatre types : Adénine ou Adénosine (A), Guanine (G), Cytosine (C) et Thymine (T)).

L'ADN peut être en *simple brin* ou *double brin*. Un brin simple (aussi appelé polynucléotide) est un Polymère linéaire (Figure 1.4).

On représente un polynucléotide par une séquence orientée de lettres:

5' -A-T-T-C-A-G-G-C-A-T-T-A-G-C- 3'

Les brins de nucléotides peuvent coller ensemble pour former une épine dorsale continue.
Ceci donne une forme d'échelle (Figure 1.5). La forme d'échelle se torde sur elle même pour donner une forme hélicoïdale (Figure1.6). Cette structure est la célèbre " double hélice ", découverte par *Crick et Watson* en 1953.
Les bases ou nucléotides (A, T, C, G) s'organisent en paires selon une complémentarité exclusive: A-T et G-C. C'est cet appariement qui permet un enroulement quasi-parfait en hélice droite des deux chaînes sucre -phosphate qui portent ces nucléotides [Alberts et autres, 02].
La structure est stabilisée par l'interaction (liaisons d'hydrogène) entre les bases et l'empilement successif des paires de nucléotides (figure 1.7).

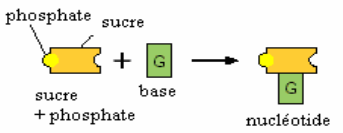


Figure 1.3 : *Construction d'un nucléotide*

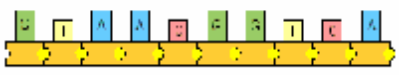


Figure 1.4 : *Un brin d'ADN ou polynucléotide*

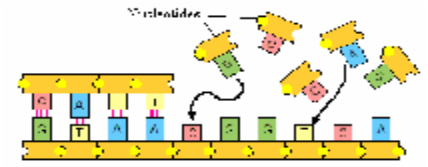


Figure 1.5 : *Construction du 2ème brin d'ADN*

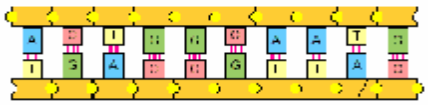


Figure 1.6 *: Double brins d'ADN (Forme d'échelle)*

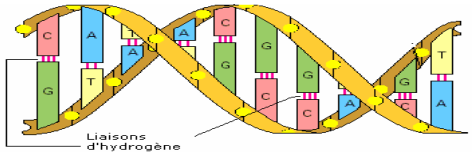


Figure 1.7 : *Double brins d'ADN (Forme hélicoïdale)*

Dans un brin d'ADN, il y a des segments dits codants (*Exons*) (figure 1.8) et des segments non codants (*Introns*). Le premier type qui est l'exon, va participer à la génération d'autres macromolécules (ARNs et par la suite des protéines) contrairement aux introns qui sont sans utilité apparente [Alberts et autres, 02].

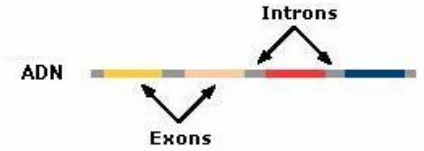


Figure 1.8 : *Exons et Introns dans un brin d'ADN*

1.5.2 Les Chromosomes

Les chromosomes sont des éléments du noyau cellulaire en nombre constant, qui déterminent l'hérédité.
Un chromosome est une structure en bâtonnet, constituée de longues chaînes d'ADN, auxquelles sont fixées des protéines. L'ADN de l'homme est divisée en 23 paires de chromosomes contenus dans le noyau de chacune de ces cellules, 22 paires sont communes aux deux sexes. Les deux chromosomes restants sont les chromosomes sexuels. Chez la femme, ils forment une paire. On les appelle les chromosomes X et l'autre, beaucoup plus court est appelé chromosome Y.

1.5.3 L'ARN

L'ARN (Acide Ribonucléique) ressemble énormément à l'ADN (figure 1.9) mais il y a des différences telles que :

- Le sucre de l'ADN (désoxyribose) et celui de l'ARN est le ribose.
- La Thymine (T) de l'ADN est remplacée par *l'uracile (U)* dans l'ARN
- l'ARN peut s'apparier avec un autre ARN complémentaire mais les ARNs sont généralement simple brin. Contrairement aux brins de l'ADN qui vont en couple.

§ 3 types d'ARNs ont été identifiés : ARN messager (ARNm), ARN ribosomiques (ARNr) et ARN transfert (ARNt). mais d'autres types ont été découverts ces dernières années.

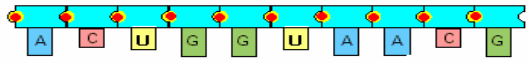


Figure 1.9: *Un brin d'ARN*

Structures d'un brin ARN

Un brin d'ARN peut avoir plusieurs structures : primaire (fig. 1.10), secondaire (fig. 1.11) et tertiaire (fig. 1.12) [Batzoglou, 04]. Cette définition est valable même pour les protéines à qui on peut attribuer encore une structure quartenaire.

5' GCGGAUUUAGCUCAGUUGGGAGAGCGCCAGACUGAAGAUCUGGAGGUCCUGUGUUCGAUCCACAGAAUUCGCACCA 3'

Figure 1.10 : *La structure primaire d'une séquence d'ARNt de la phénylalanine*

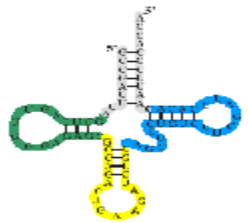


Figure 1.11 : *La structure secondaire d'une séquence d'ARNt de la phénylalanine.*

Figure 1.12 *: La structure tertiaire d'une séquence d'ARNt de la phénylalanine*

1.5.4 Les Protéines

Les protéines sont les macromolécules les plus importantes. Elles sont responsables de presque de toutes les réactions biochimiques qui ont lieu à l'intérieur de la cellule. Les protéines sont de sortes différentes et avec une variété de fonctionnalités. Certaines d'entre elles incluent [Alberts et autres, 02]:

- ü Protéines structurelles: elles sont les bases de construction des divers tissus.
- ü Enzymes: elles catalysent les réactions chimiques essentielles qui auraient pris beaucoup de temps pour se produire.
- ü Transporteuses: elles portent les éléments chimiques qui font partie de l'organisme à d'autres (par exemple les hémoglobines qui portent l'oxygène).

Les protéines se composent de chaîne des acides aminés. Chaque acide aminé a une structure constante. Il y a 20 acides aminés. Deux acides aminés peuvent se joindre, avec un " lien de peptide ", formant une chaîne: un " polypeptide ".
Une séquence protéique est une collection ordonnée de lettres choisis dans l'alphabet = {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. où chacune des lettres correspond à un acide aminé.
Exemple d'acide aminé : « Lysine » codé par la lettre « K ».
Exemple d'une protéine: l'insuline**:**

"F V N Q H L C G S H L V E A L Y L V C G E R G F F Y T P K A"

La synthèse de protéine se produit dans des structures appelées *les ribosomes* situées dans la cellule mais en dehors du noyau. Le modèle de la protéine est dans l'ADN, située dans le noyau. Donc il y a un besoin d'un " messager " pour transférer l'information à partir de l'ADN aux ribosomes. L'ARN est ce messager (ARNm). Il est synthétisé en utilisant l'ADN comme modèle. Ce processus s'appelle ***la transcription*** (Figure 1.13)

Comment interprète-on l'information diffusée par ARNm? Ceci est une séquence de "triplets" de nucléotides, ou *de codons*. Chaque codon indique un acide aminé (figure 1.14). Mais puisqu'il y a $4^3 = 64$ de codons possibles, mais seulement 20 acides aminés, il y a une certaine redondance dans le code où des triplets différents codent le même acide aminé (voir la table 1.2). Cette fonction de codage, f: codon **à** acide aminé est *le code génétique* elle est universel, pour tous les organismes.

N.B. : les trois codons spéciaux : *stop codons;* ils ne codent pas un acide aminé; mais ils indiquent la fin d'une région de codage de protéine sur une grande molécule d'ADN.

La traduction : est le processus par lequel une séquence des codons *est traduite* vers une séquence d'acides aminés (figure 1.13). Une molécule appelée l'ARN de transfert (ARNt*)* permet le passage des codons aux acides aminés. ARNt contient un triplet appelé *anticodon*, celui-ci possède une extrémité à la quelle un acide aminé spécifique vient s'attacher.
ARNt est situé dans le cytoplasme, et porte les acides aminés vers les ribosomes. Les acides aminés rassemblés par les ARNts vont alors collés les uns aux autres pour former une chaîne de peptides appelée polypeptides. Une chaîne de polypeptides peut atteindre une taille de 50 à 30000 acides aminés, la moyenne étant 400 acides aminés.

Après transcription d'ADN et avant la synthèse de protéine, un processus enlève quelques segments de l'ARN (*introns),* laissant seulement les codons significatifs (*exons)* qui seront *ex*primés. Ce processus est appelé «l'*épissage* » [Vert, 05].

La structure de la protéine : La protéine possède quatre structures : primaire, secondaire, tertiaire et quartenaire.
Parmi les paradigmes de la biologie moléculaire, celui de la relation entre structure et fonction. La structure secondaire ou tertiaire peut inférer la fonction d'une protéine. Donc connaître la structure va faciliter l'identification de sa fonction et par conséquent son importance pour tout l'organisme [Batzoglou, 04].

Structure ⟶ ***Fonction***

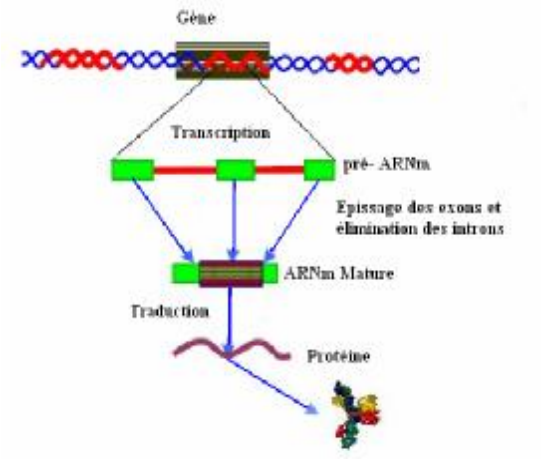


Figure 1.13 : *Synthèse de protéine*

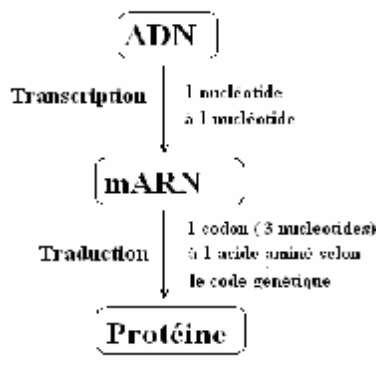


Figure 1.14 : *Processus du codage*

2ème base dans le codon

1ère base dans le codon	U	C	A	G	3ème base dans le codon
U	Phe F Phe F Leu L Leu L	Ser S Ser S Ser S Ser S	Tyr Y Tyr Y STOP STOP	Cys C Cys C STOP Trp W	U C A G
C	Leu L Leu L Leu L Leu L	Pro P Pro P Pro P Pro P	His H His H Gln Q Gln Q	Arg R Arg R Arg R Arg R	U C A G
A	Ile I Ile I Ile I Met M	Thr T Thr T Thr T Thr T	Asn N Asn N Lys K Lys K	Ser S Ser S Arg R Arg R	U C A G
G	Val V Val V Val V Val V	Ala A Ala A Ala A Ala A	Asp D Asp D Glu E Glu E	Gly G Gly G Gly G Gly G	U C A G

Table 1.2 : *Le code génétique des acides aminés.*

1.5.5 Le Gène

Définition 1.2 : un gène est un fragment de l'information génétique (ADN) correspondant à une protéine.

Nous pouvons récapituler ce mécanisme comme « **dogme central** » de biologie moléculaire:

ADN = ARN = protéine = phénotype.

- la transcription est la propriété de passer de l 'ADN à l'ARN
- la traduction est le processus de passer de l'ARN à la protéine

N'importe quelle interférence dans ces étapes changerait le phénotype, c.-à-d. la structure et la fonction de l'organisme.

Le génome est un ensemble de tous les gènes d'un organisme donné.

Ce pendant, on sait aujourd'hui qu'à un gène ne correspond pas forcément à une protéine unique. En effet, l'expression peut subir des modifications:

- post-transcriptionnelles: l'ARN messager transcrit à partir d'un gène, peut être recombiné (certaines parties sont coupées et éliminées : les introns, les autres sont "recollées" entre elles : les exons). C'est ce qu'on appelle l'épissage alternatif, qui peut être modulé en fonction du cycle cellulaire ou de stimulus extérieurs.
- post-traductionnelles: le repliement 3D d'une protéine peut être modifié, par exemple sous l'action d'une protéine particulière. D'autre part, de nombreuses protéines subissent des modifications chimiques (formation de ponts désulfures, ajouts de groupements sucres pour former des glycoprotéines) après leur synthèse.

Ces variations d'expression sont à l'origine de la complexité de l'expression de l'information génétique. Certes, toute l'information génétique est contenue dans l'ADN, mais deux cellules au même contenu ADN peuvent être extrêmement différentes, en fonction du contenu de leur cytoplasme (différentiation des cellules dans un organisme).

Ceci a permis de mieux ajuster le dogme central de biologie :

- avant 1 gène = 1 ARN = 1 protéine
- maintenant 1 gène = x ARNs = xy protéines

1.5.6 Comment les Génomes Sont Régulés ?

Chaque cellule dans le corps contient toute l'ADN, et par conséquent la recette pour n'importe quelle protéine. Mais chaque cellule synthétise sa propre protéine. Il y a ici un certain processus de différentiation.
La différentiation de l'ADN commune dans une variété de types de cellules se produit par le fait qu'un gène peut être en état de marche ou arrêt (allumé ou éteint) [Rocha, 00]. Pour déterminer exactement le produit d'une cellule, il faut être capable de répondre aux questions suivantes :

- Ce qui rend un gène en état de marche ou arrêt
- Quand est-ce qu'un gène est en état de marche ou arrêt?
- Où (en quelles cellules) un gène est en marche?
- Combien de copies du produit gène sont produites?

La réponse à ces questions permettrait aux biologistes de prédire le fonctionnement de n'importe organisme dont on détient le matériel génétique.

1.5.7 Évolution d'un Gène

Un gène peut subir des modifications et des opérations dont le résultat est souvent un nouveau gène. C'est cette évolution qui a donné naissance à plusieurs espèces d'organismes [Alberts et autres, 02]. Et qui a participé à l'enrichissement de la nature. On peut citer quelques opérations de modifications de gènes qui peuvent survenir d'une manière spontanée ou provoquées par des acteurs externes [Batzoglou, 04]:

- *Réplication ou Duplication d'un gène :* un gène existant peut se reproduire afin de créer une paire de gènes identiques (division cellulaire).
- *Mutation* : est définie comme un changement dans la structure d'une séquence d'ADN. C'est la substitution d'un nucléotide par un autre. Ceci peut se produire lors d'une réplication. La mutation peut se manifester à une échelle plus élevée au niveau chromosomique. (voir Figure 1.15)
- *Insertion :* elle est définie comme une insertion d'un nucléotide dans une séquence d'ADN
- *Délétion :* c'est la disparition d'un nucléotide d'une séquence sans qu'il soit remplacé par un autre.

- *Croisement de gènes ou recombinaison* : deux gènes peuvent être cassés et puis reliés pour former un nouveau gène hybride composé des segments de l'ADN qui appartenus aux gènes séparés.
- *Transfert (intercellulaire) horizontal :* un morceau d'ADN peut être transféré à partir du génome d'une cellule à une autre (même d'une espèce à une autre : cas des virus).

Chacune de ces modifications laisse une trace caractéristique dans la séquence d'ADN de l'organisme en affectant son génotype par conséquence son phénotype.

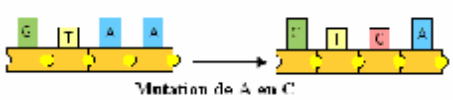


Figure 1.15 : *Une mutation d'un nucléotide vers un autre*

« Évolution des gènes = mutation, insertions délétions, recombinaison »

Le processus évolutionnaire se produit à différents taux. Si les mutations d'ADN se produisent dans des régions non critiques, elles sont incorporées à la prochaine génération. Si les mutations se produisent dans des régions critiques, elles ont peu de chance d'être propagées dans les générations futures. Cependant, quelques mutations ont des effets positifs, et sont conservées. La conservation des séquences implique la fonctionnalité. Le fait que l'évolution n'a pas modifié une région d'une séquence suppose qu'elle soit fonctionnellement importante pour l'organisme [Alberts et autres, 02]:

- Les régions fonctionnelles des gènes (sites catalytiques, de fixation etc.) sont soumises à la sélection. Elles sont relativement préservées par l'évolution car des mutations trop radicales sont désavantageuses.
- Les régions non fonctionnelles ne subissent aucune sélection et divergent rapidement à mesure que les mutations s'accumulent.

Les nouveaux gènes apparaissent surtout par transmutation des gènes ancestraux : on peut donc déduire la fonction de la plupart des gènes par comparaison avec des gènes « homologues » d'autres espèces dont la fonction est déjà connue.

1.5.8 Homologie et similitude des gènes

Le paradigme central de la bioinformatique est : « ***la déduction par homologie*** ».

Terminologie *:*
Identité : proportion des paires de bases (résidus) identiques entre deux séquences exprimée généralement en pourcentage.
Similitude : mesure de la ressemblance entre deux séquences. Le degré de similitude est quantifié par un pourcentage de substitutions conservatives des séquences.
Homologie : deux séquences sont homologues si elles ont un ancêtre commun. Il n'y a pas de degré d'homologie. On ne dit pas : très homologues, faiblement homologues. Deux gènes sont homologues ou ils ne le sont pas.

Toutes les opérations modification de gènes citées ou non dans le paragraphe précédent, permettent :

§ Spéciation : c'est la séparation d'une espèce en deux, chaque population évolue et forme une nouvelle espèce. Cette modification est le fruit d'une insertion, délétion ou mutation au niveau d'un gène (Figure 1.16.A).
§ Les nouvelles espèces héritent des mêmes gènes, mais modifiés
§ Divergence : leurs gènes accumulent des mutations et génèrent d'autres espèces (figure 1.16.B).

Gènes Orthologues et Paralogues :

Deux gènes sont homologues s'ils sont issus d'un même ancêtre.

On distingue les gènes orthologues et gènes paralogues [Alberts et autres, 02]:

- Orthologues : gènes homologues et organismes différents ou espèces différentes
- Paralogues : gènes homologues et issus d'organismes identiques

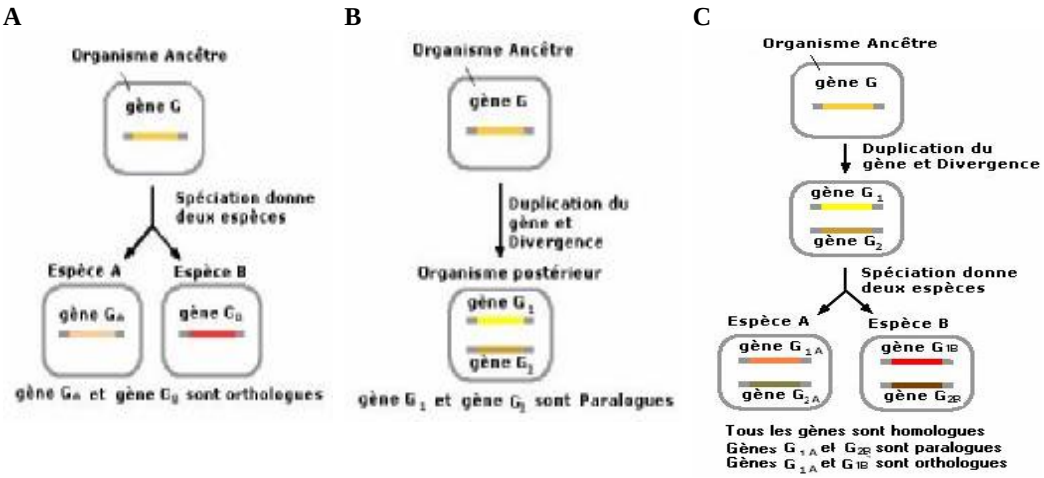


Figure 1.16 : A) Gènes orthologues. B) Gènes paralogues. C) Gènes homologues

Deux gènes homologues : signifie qu'ils ont un ancêtre commun.

Deux gènes similaires : implique des protéines similaires puis une fonction similaire.

1.6 Les Banques de Données Biologiques

Les premières banques de données biologiques sont apparues au début des années 80 sous l'initiative de quelques équipes de recherches. Leur principale mission est de rendre publiques les séquences qui ont été déterminées.

Les données biologiques stockées dans ces banques sont des séquences primaires d'ADN, d'ARN et de protéines. Les données peuvent être soumises et consultées par l'intermédiaire du Web. Les séquences stockées dans ces banques sont obtenues de plusieurs manières différentes. Il y a celles isolées à partir d'une cellule, déduites à partir de la séquence nucléique par simple traduction (cas des séquences d'ARN ou protéines) ou encore par génie génétique.

Les données stockées doivent être consultées d'une manière significative (fig. 1.17) , et souvent le contenu de plusieurs banques de données doit être consulté simultanément et en corrélation les uns avec les autres. Des langages spéciaux ont été développés pour faciliter cette tâche (tels que le système de récupération de séquence « SRS » et le système « Entrez »). Certaines bases de données fournissent la fonctionnalité d'accès aux séquences mais encore des liens vers d'autres bases de données et les résultats d'analyse déjà obtenus. Par exemple, SWISSPROT [Bairoch et Apweiler, 00] contient des séquences de protéine ainsi que des annotations décrivant la fonction d'une protéine. Des structures 3D des protéines sont stockées dans des bases de données spécifiques [Berman et autres, 00]. On peut trouver des banques spécialisées

pour le stockage des motifs. En outre, des bases de données de la littérature scientifique (telles que PUBMED, MEDLINE) fournissent des fonctionnalités additionnelles, par exemple elles peuvent rechercher les articles scientifiques semblables basés sur l'utilisation de la reconnaissance des mots. Ils ont développé des systèmes d'identification des textes qui extraient automatiquement l'information concernant un sujet tel que la fonction d'une protéine à partir des résumés des articles scientifiques.

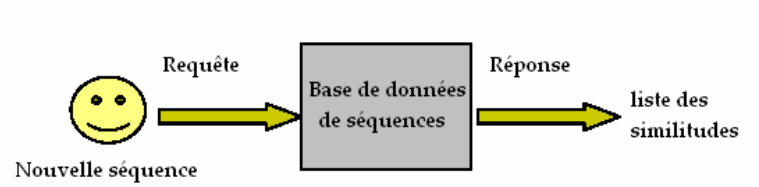


Figure 1.17 : Interrogation d'une base de données

1.6.1 Les Banques de Séquences Nucléiques :

Nous citons les banques les plus populaires malgré que l'accès soit toujours contrôlé via des mots de passe:

§ **EMBL** : banque européenne créée en 1980 et financée par l'EMBO (European Moleculary Biology Organization), [Hamm et Cameron, 86] elle est aujourd'hui diffusée par l'EBI (European Bioinformatics Institute, Cambridge, UK).
§ **GenBank** : créée en 1982 par la société IntelliGenetics et diffusée maintenant par le NCBI (National Center for Biotechnology Information, Los Alamos, US). [Bilofsky et Burks, 88] elle est soutenue par le NIH (National Institute of Health). Elle possède plus de 50 millions séquences stockées
§ **DDBJ** (Dna Data Bank): créée en 1986 et diffusée par le NIG (National Institute of Genetics, Japon).

La collaboration entre les deux premières banques a commencé relativement tôt. Elle s'est étendue en 1987 avec la participation de la DDBJ. Ils ont adopté un système de conventions communes : "The DDBJ/EMBL/GenBank Feature Table Definition" en 1990 qui a défini un format unique pour la description des caractéristiques biologiques qui accompagnent les séquences dans les banques de données nucléiques.

1.6.2 Les Banques de Séquences Protéiques :

§ **PIR-NBRF** : 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 à l'Université de Genève et maintenue depuis 1987 dans le cadre d'une collaboration, entre cette université (via ExPASy, Expert Protein Analysis System) et l'EBI. Celle-ci regroupe aussi des séquences annotées de la banque PIRNBRF ainsi que des séquences codantes, traduites de l'EMBL.

1.6.3 Les Banques de Motif

§ **Prosite** : La base de données dédiées aux stockages des motifs protéiques ayant une signification biologique [Hofmann et autres, 99] peut être considérée comme un dictionnaire de motifs.

Les bases de ce type ont pour mission le recensement dans des catalogues les séquences des différents motifs pour lesquels une activité biologique a été identifiée.

1.7 L'Analyse des Séquences

Les données primaires des projets séquençage sont des séquences d'ADN. Celles-ci sont devenues vraiment exploitables à travers leur annotation. Plusieurs étapes d'analyse avec des outils de la bioinformatique sont nécessaires pour partir d'une séquence d'ADN crue et atteindre des séquences annotées d'une protéine:

§ Établir la séquence correcte des fragments contigus d'ADN pour obtenir une séquence continue;
§ Trouver les emplacements de déclenchement de transcription et la traduction, trouver des sites de promoteurs, et des ORFs (Open Reading Frame = cadre ouvert de lecture);
§ Trouver emplacements d'épissage, introns, exons;
§ Traduire la séquence d'ADN en une séquence de protéine
§ Comparer la séquence d'ADN à des séquences connues homologues de protéine afin de vérifier les exons… etc.
§ Déterminer la structure (surtout la structure tertiaire 3D) puis la fonction de la protéine par comparaison à d'autres séquences semblables.
§ Déterminer une origine et/ou une histoire évolutive commune (phylogénie).

Pour un bioinformaticien, une séquence biologique est un MOT ou une chaîne de caractères dont on ne peut manipuler que sa structure primaire présentée généralement dans un format donné.
L'analyse des données biologiques consiste en général à chercher un motif dans une séquence, aligner deux ou plusieurs séquences, comparer un motif ou séquence avec les données d'une banque et établir un lien phylogénétique…etc.

1.7.1 La Recherche d'un « Motif » dans une Séquence

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" [Batzoglou, 04].
Les motifs sont souvent recherchés dans des séquences car ils sont généralement impliqués dans des systèmes de régulation ou ils définissent des fonctions biologiques comme 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, 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) [Rocha, 00] .

Exemple de recherche de motif:

Séquence **:** C T G T G T G T A C A T G T G de longueur 15
Motif **:** T G T G de longueur 4

Position :	1	**2**	3	**4**	5	6	7	8	9	10	11	**12**	13	14	15
Séquence	C	T	G	T	G	T	G	T	A	C	A	T	G	T	G
		T	G	T	G										
				T	G	T	G								

T G T G

Solution **:** Ensemble de positions : *{2, 4, 12}.*

1.7.2 Alignement de Deux Séquences

Un alignement de deux séquences (appelé souvent '*Alignement deux à deux'*) est une mise en correspondance entre les résidus avec une possible insertion des espaces (gaps) afin d'obtenir des séquences de longueur égales. Toutes les correspondances sont autorisées à condition que l'ordre des résidus soit respecté.
Trois situations sont possibles pour une position donnée de l'alignement :

- Les caractères sont les mêmes : identité
- Les caractères ne sont pas les mêmes : Substitution
- L'une des positions est un gap (espace) : Insertion/Délétion

Exemple d'alignement de deux séquences :

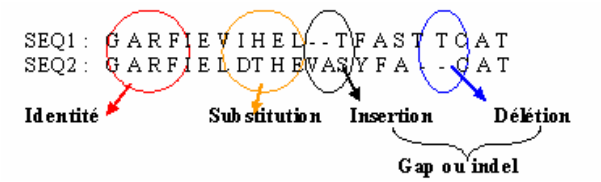


Figure 1. 18 : Alignement de deux séquences protéiques

1.7.3 Évaluation d'un Alignement

Cependant, il est clair que pour deux séquences données quelconques il y a plusieurs alignements possibles. Il est devenu alors nécessaire de pouvoir déterminer quel est le meilleur alignement ou plutôt l'optimal si possible.
Évaluer un alignement revient alors à mesurer sa qualité en déterminant la distance qui sépare les deux séquences. Le score d'un alignement est la somme des scores de toutes les positions de bases (résidus) prises deux à deux.

Exemple d'évaluation *:*
On peut attribuer une valeur positive à des symboles alignés identiques et une pénalité (valeur négative) à une substitution ou à un gap.

Si l'on considère l'exemple précédent :
Score (*identité*) = 2
Score (*substitution*) = -1
Score (*gap*) = -2
Le score de cet alignement serait alors :

SEQ1 : G A R F I E V H E L - - T F A T T C A T
SEQ2 : G A R F I E L T H E V A S Y F - - C A T **score total**
2+2 +2+2+2+2 -1 -1 -1 -1 -2 -2 -1 -1 -1 -2 -2 +2+2+2 = **+3**

Pour évaluer un alignement, le poids de chaque paire de résidus (identité ou substitution) dépend de la nature des résidus mis en correspondance. Le calcul de score d'un alignement de deux séquences A et B de longueur équivalente L est alors :

$$\text{Score (A, B)} = \sum_{i=1}^{L} SC(Ai, Bi) \qquad (1.1)$$

1.7.4 Le Système de Score

Définition 1.1: Un système de score est le coût à attribuer aux opérations élémentaires (identité, substitution, délétion et insertion) de comparaisons de séquences.
En général, on a besoin donc :

- Des systèmes de scores qui soient « biologiquement pertinent »
- Des matrices de substitution et donc des scores individuels *Sc*(ai, bj), dont le choix dépend de la relation recherchée entre les deux séquences :
  - Relation structurelle (propriétés physico-chimiques)
  - Relation d'homologie (évolution moléculaire)

1.7.5 Les Matrices de Substitution

Le choix d'une matrice de substitution gouverne le système des scores et par conséquent influe sur les résultas obtenus.
Il existe deux types de matrices de substitution à utiliser selon la nature des séquences nucléiques ou protéiques.

v Matrices de Scores pour l'ADN

§ La matrice Identité

Cette matrice consiste en l'attribution d'un score 1 en cas d'identité sinon un zéro

	A	C	G	T
A	**1**	0	0	0
C	0	**1**	0	0
G	0	0	**1**	0
T	0	0	0	**0**

§ La matrice de Transition/Transversion

Dans cette matrice en prends en considération l'effet des actions des transitions (A à G, G à A, C à T, et T à C) et transversion (les autres passages entre nucléotides)
Identité=3
Transition= 1, Transversion = 0.

	A	C	G	T
A	**3**	0	1	0
C	0	**3**	0	1
G	1	0	**3**	0
T	0	1	0	**3**

§ La matrice BLAST

La matrice identité Blast. C'est une matrice de même principe que la matrice Identité sauf que les valeurs attribuées en cas d'identité et substitution sont différentes de 1 et 0.
On remarque que la substitution ici est fortement pénalisée.

	A	C	G	T
A	**5**	-4	-4	-4
C	-4	**5**	-4	-4
G	-4	-4	**5**	-4
T	-4	-4	-4	**5**

v Matrices de score pour les protéines :

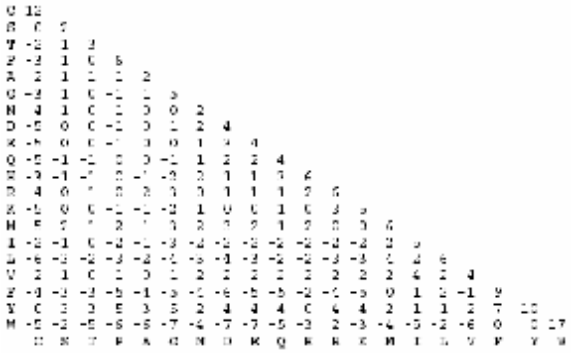
Deux grandes familles de matrices (log odds matrix)

§ Matrices PAM

Les matrices PAM pour « Percent Accepted Mutation/ Accepted Point Mutation » [Dayhoff

et autres, 78], sont construites par étude de segments pris dans des séquences protéiques homologues (moins de 15% de différences).

PAM***x*** : x % de mutations acceptées entre les séquences qui ont servi à construire la matrice
Les fréquences de substitutions observées (ou probabilité conditionnelle : appelée "odd") sont transformées en logarithme de probabilité, normalisé en unité d'évolution. Le logarithme est utilisé pour que dans les programmes de recherche de ressemblance, la somme de ces éléments donne le logarithme de la probabilité pour la séquence entière (le modèle étant Markovien : indépendance de fréquences de substitution).
Les éléments diagonaux de la matrice indiquent une évolution sans substitution.
Pour PAM1, leur somme est telle qu'elle correspond à une probabilité de 99/100 (1 mutation pour 100 résidus : d'où le nom PAM : *accepted point mutation*)
L'indépendance des fréquences et les éléments de la matrice étant des logarithmes de fréquences, on peut calculer PAM (N) en élevant PAM1 à la puissance N, par exemple pour PAM120, il faut multiplier PAM1 par elle-même 120 fois.



PAM 250

Remarque : la valeur Sc(Xi, Yj) sera :
- s = 0 : les probabilités observées et attendues sont identiques
- s < 0 : les probabilités observées sont inférieures aux attendues
- s > 0 : les probabilités observées sont supérieures aux attendues

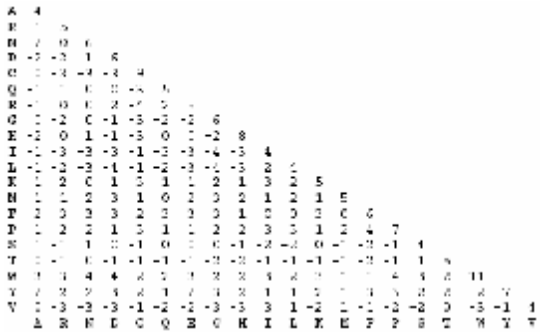
§ Matrice BLOSUM

Ces matrices *BLOSUM* (Blocks Substitutions Matrices) [Henikoff et Henikoff, 92] sont construites par analyse de séquences de protéines. Les séquences sont découpées en blocs (2000 résidus au total) par rapport au pourcentage d'acides aminés inchangés.

BLOSUM ***x*** : matrice obtenue à partir de séquences présentant au minimum x % d'identité (similitude) entre elles.

Une matrice "d'odds" est calculée à partir des blocs d'alignement pour chaque valeur de similitude, et ensuite chaque élément est transformé en unité d'information en prenant le logarithme du rapport de la valeur observée à la valeur qu'on obtiendrait au hasard. Cette matrice est ensuite normalisée. Les correspondances entre BLOSUM et PAM, basées sur la théorie de l'information sont :

- PAM250 ---> BLOSUM45
- PAM160 ---> BLOSUM 62
- PAM120 ---> BLOSUM 80

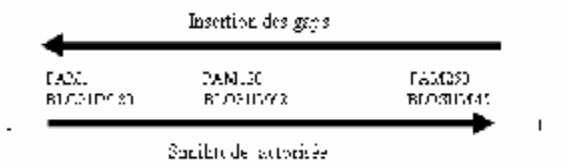


La matrice BLOSUM 62

v Choix de la Matrice Protéique

Vu la diversité des types de matrices, le choix d'une matrice dépend du type d'analyse que l'on veut faire. Il n' y a pas une matrice idéale et un grand nombre d'études comparatives sur les matrices ont mis en évidence (de manière schématique) que :

- Pour des séquences similaires et courtes, il est préférable d'utiliser une matrice BLOSUM élevée ou PAM faible.
- Pour des séquences divergentes et longues, il est préférable d'utiliser une matrice BLOSUM faible ou PAM élevée.
- La matrice BLOSUM 62 semble être la matrice la plus utilisée pour la comparaison avec les banques de données, et pour un grand nombre de logiciels d'alignement de séquence, elle semble être la matrice par défaut.



1.7.6 Pénalité des Gaps

L'opération d'insertion/délétion présente un coût qu'il faut pouvoir estimer pour représenter au plus proche de la réalité biologique [Batzoglou, 04].
Plusieurs systèmes de pondération ont été proposés :

§ *Pénalité fixe par gap*

Dans ce cas on affecte un gap une valeur fixe sans tenir compte de sa position dans la séquence ni de sa longueur. *: P=k*
L'exemple décrit dans la *section 1.7.3* en est un cas.

§ *Pénalité variable ou fonction Affine*

Le score d'un gap au niveau d'une séquence est désormais pénalisé en fonction de sa longueur. Le score d'une séquence de gap x est alors calculé par la formule suivante :

$$P = GOP(x) + (L-1) * GEP(x) \qquad (1.2)$$

P : le coût global du gap de longueur L

GOP(x) : la pénalité fixe d'ouverture d'insertion indépendante de la longueur (GOP : Gap Open Penalty)
GEP(x) : la pénalité d'extension pour un gap (GEP : Gap Extension Penalty)
L : la longueur du gap.

Avec ce système, une longue insertion est plus pénalisante qu'une courte, ce qui revient en fait à minimiser l'introduction même d'une insertion. Autrement dit, on facilitera souvent dans un alignement le fait d'avoir peu d'insertions, éventuellement longues, plutôt que d'avoir beaucoup d'insertions d'un seul élément.
En général, La pénalité d'ouverture est plus grande que celle d'extension (inspiré des mécanismes évolutif)
Le score total d'un alignement avec gap est la somme des scores d'identité ou de substitution des résidus alignés auquel on retranche les pénalités d'ouverture et d'extension de gaps.

- Exemple :

Le score de cet alignement devient en utilisant une fonction affine avec :
Si l'on considère l'exemple précédent :

Score (*identité*) = 2
Score (*substitution*) = 1
Score (*GOP*) = -2
Score (*GEP*) = -1

Le score de cet alignement serait alors :

SEQ1 : G A R F I E V H E L - - T F A T T C A T
SEQ2 : G A R F I E L T H E V A S Y F - - C A T **score total**
2+2+2+2+2+2 +1+1 +1+1 -2 -1 +1+1+1 -2 -1 +2+2+2 = **19**

Une autre version de la fonction affine :

$$P = GO\ P + GEP * log\ (L) \qquad (1.3)$$

Dans la littérature, on trouve plusieurs forme de la fonction affine telle que la fonction affine généralisée et autres, mais la fonction affine (formule (1.2) est pour l'instant la plus utilisée par les méthodes d'alignement pour des raisons de complexité (*O(mn)* avec m et n tailles des séquences alignées).
Les différentes méthodes d'alignements considèrent que les gaps situés à la fin d'une séquence ne doivent pas pénaliser un alignement car ils sont considérés meilleurs que ceux introduits au milieu.

1.7.7 Les Méthodes d'Alignement de Deux Séquences

Il existe deux types d'alignements de séquences : global et local.
Le premier prend en considération l'ensemble des résidus de chacune des séquences. Si les longueurs des séquences sont différentes, alors la plus courtes va subir des insertions de gaps afin d'arriver à aligner les deux séquences d'une extrémité à l'autre. Cependant dans un alignement global, si uniquement des segments courts sont très similaires entre deux séquences, les autres parties des séquences risquent de diminuer le poids de ces régions. C'est pourquoi d'autres algorithmes d'alignements, dits locaux, basés sur la localisation des zones de similarité sont nés. Le but de ces alignements locaux est de trouver sans prédétermination de longueur les zones les plus similaires entre deux séquences. L'alignement local comporte donc une partie de

chacune des séquences et non la totalité des séquences comme dans la plupart des alignements globaux

v Alignement Global

Plusieurs méthodes ont été développées afin de réaliser un alignement global de deux séquences le plus correct que possible. Parmi ces méthodes et qui sont toujours utilisées on trouve des méthodes graphiques et autres qui utilisent la programmation dynamique.

§ Le DotPlot

C'est un outil graphique [Staden, 82] qui réalise l'alignement de deux séquences.
Ayant deux séquences X et Y à aligner. Une manière intuitive consistait en une présentation matricielle de l'alignement. La matrice de point (*dot-matrix*), peut être construite ainsi :
- chacun des deux axes correspond à une séquence (Figure 1.19)
- une croix si xi = yj (où xj est un élément de la première séquence X et yj un élément de la deuxième séquence Y, sinon rien.
- Si une suite de croix consécutives dans une diagonale est observée alors ceci indique des identités (similitude) entre des parties des deux séquences.

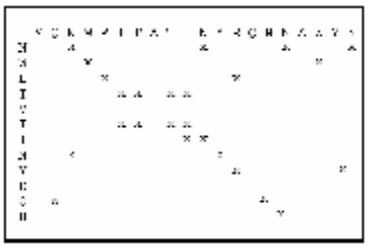


Figure 1.19 : *Dotplot de deux séquences*

Cette méthode peut également être utilisée pour rechercher sur une séquence des répétitions directes ou des palindromes en comparant la séquence sur elle-même.
Cette méthode est très simple d'utilisation mais l'interprétation des résultats obtenus est difficile.

§ Méthode Exacte ou Programmation Dynamique

Pour pouvoir comparer deux séquences de longueur L, il faut un temps de calcul proportionnel à L^2. L'exploration de chaque position de chaque séquence pour la détermination éventuelle d'une insertion augmente le temps de calcul d'un facteur 2*L. La programmation dynamique est un moyen qui permet de limiter cette augmentation pour conserver un temps de calcul de $O(L^2)$. Elle est basée sur le fait que tous les évènements sont possibles et calculables mais que la plupart sont rejetés en considérant certains critères. Needleman et Wunsch (1970) sont les premiers à avoir introduit ce type d'approche pour un problème biologique et leur algorithme reste une référence dans le domaine.

§ Algorithme Needleman & Wunsch

Basé sur la programmation dynamique (la récursivité), cet algorithme [Needleman et Wunsch, 70] ne calcule pas la différence entre deux séquences mais la similarité. Considérons deux séquences A(1,n) B(1,m).
Un tableau à deux dimensions est rempli ligne après ligne (en partant de la dernière) et pour chaque ligne, colonne après colonne (en partant de la dernière) en obéissant à la règle suivante :

Le score S(i,j) est le nombre maximum de correspondance entre les deux parties de séquences A(i,n) et B(j,m) (en prenant tous les chemins possibles à partir de (i,j)) et en appliquant une fonction de score :

- score pour une identité =1
- score pour une substitution, une insertion ou délétion = 0

La formule de récurrence est :

$$S(i,j) = \max \begin{cases} \text{Si } ai = bj+1 \quad S(i, j+1) -1 + s(ai, bj), \quad \text{sinon } S(i, j+1) + s(ai, bj) \\ \text{Si } ai+1 = bj+1 \quad S(i+1,j+1) -1 + s(ai, bj), \quad \text{sinon } S(i+1,j+1) + s(ai, bj) \\ \text{Si } ai+1 = bj \quad S(i+1,j) -1 + s(ai, bj), \quad \text{sinon } S(i+1, j) + s(ai, bj) \\ \text{Avec évidemment} \quad S(n+1, j) = S(i, m+1) = 0 \end{cases} \qquad (1.4)$$

La similarité entre les deux séquences est égale à la valeur de S(1,1) et l'alignement est un graphe qui a pour origine S(1,1) et parcourt la matrice pour des i et j croissant en recherchant l'élément maximal voisin.

Exemple : Soient deux séquences protéiques « VTEERDEF » et « ITSHEAL ».
On construit la matrice initiale (Figure 1.20.A) à partir d'une matrice de substitution PAM 250 puis on procède à la transformation en appliquant les formules de l'équation 1.2, on obtient alors la matrice transformée (figure 1.20.B)

	V	T	E	E	R	D	A	F
L	2	-2	-3	-3	-3	-4	-2	2
T	0	3	0	0	-1	0	1	-3
S	-1	1	0	0	0	0	1	-3
H	-2	-1	1	1	2	1	-1	-2
E	-2	0	4	4	-1	3	0	-5
A	0	1	0	0	-2	0	2	-4
L	2	-2	-3	-3	-3	-4	-2	2

A : Matrice initiale

	V	T	E	E	R	D	A	F
L	14	7	6	6	4	4	0	2
T	10	12	9	9	6	4	3	-3
S	8	10	9	9	7	4	3	-3
H	6	7	9	8	9	5	1	-2
E	2	4	8	8	3	7	2	-5
A	2	3	2	2	0	2	4	-4
L	2	-2	-3	-3	-3	-4	-2	2

B : Matrice transformée

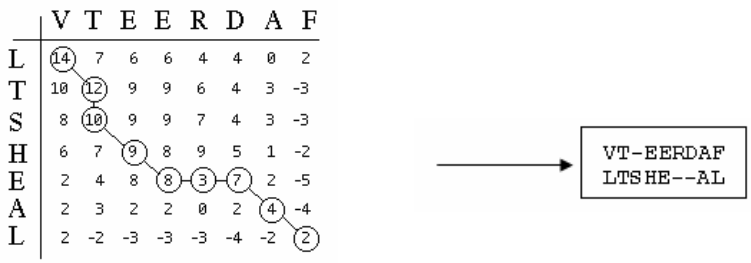
Figure 1.20 : *Construction de la Matrice transformée*

Le but est ensuite de trouver le meilleur alignement global, à partir de la matrice transformée (Figure 1.20.B). Pour cela, on établit dans la matrice un chemin qui correspond au passage des scores sommes les plus élevés, ceci en s'autorisant trois types de mouvements possibles et en prenant comme point de départ le score maximum présent dans la matrice transformée (pour cet exemple : 14). Needleman et Wunsch nomment ce passage le *chemin des scores maximum* (Figure 1.21.A).

Les mouvements autorisés pour tracer le chemin sont :

a) le mouvement diagonal qui correspond au passage de la case (i,j) à la case (i+1,j+1). C'est le mouvement privilégié.
b) le mouvement vertical qui correspond au passage de la case (i,j) à la case (i,j+1), ce qui donne une *insertion* sur la séquence en i.
c) le mouvement horizontal qui correspond au passage de la case (i,j) à la case (i+1,j), ce qui donne une *délétion* dans la séquence en j.

Dans cet exemple, les pénalités pour les insertions ne sont prises en compte.
On obtient l'alignement optimal suivant :



A : Le chemin des scores maximum **B** : L'alignement optimal obtenu

Figure 1.21 : *Passage de la matrice vers l'alignement*

v Alignement Local

Ce type d'alignement est favorable aux séquences divergentes car un alignement global serait non significatif. Pour l'alignement Local, Smith et Waterman une méthode exacte qui permet d'aligner deux séquences en essayant d'aligner des segments communs ou motifs.

§ L'algorithme Smith & Waterman

L'algorithme de Smith et Waterman [Smith et Waterman, 81] est décrit pour l'alignement local de deux séquences. Il identifie les sous séquences maximales de deux séquences par programmation dynamique.

La différence essentielle de cet algorithme avec l'algorithme de Needleman et Wunsch est que n'importe quelle case de la matrice de comparaison (initiale) peut être considérée comme point de départ pour le calcul des scores sommes et que tout score somme qui devient inférieur à zéro stoppe la progression du calcul des scores sommes et il sera réinitialisé par la valeur 0. La case concernée peut être considérée comme nouveau point de départ. Cela implique que le système de score choisi possède des scores négatifs pour les mauvaises associations qui peuvent exister entre les éléments des séquences.
L'équation utilisée pour le calcul de chaque score somme pendant la transformation de la matrice initiale prend alors l'expression suivante:

$$S(i,j) = \max \begin{cases} Sc(i,j) + S(i+1, j+1) \\ Sc(i,j) + \max S(x, j+1) - P \\ Sc(i,j) + \max S(i+1, y) - P \\ 0 \end{cases} \quad \text{avec } i+2 < x < m \text{ et } j+2 < y < n \qquad (1.5)$$

Où S(i,j) est le score somme de la case d'indice i et j, Sc le score élémentaire de la case d'indice i et j de la matrice initiale issu d'une matrice de substitution et P la pénalité donnée pour une insertion.

Exemple : Si l'on considère les séquences de l'exemple précédent, la Figure 1.22.A montre la matrice initiale construite à partir d'une matrice de substitution PAM250 et où l'on a conservé que les scores positifs ou nuls et éliminé les scores négatifs. Le score des pénalités

d'insertion/délétion est fixé à la valeur 6. La transformation de matrice initiale donnera après application des formules de l'équation 1.3, la matrice transformée suivante (Figure 1.22.B):

	V	T	E	E	R	D	A	F
L	2							2
T	0	3	0	0		0	1	
S		1	0	0	0	0	1	
H			1	1	2	1		
E		0	4	4		3	0	
A	0	1	0	0		0	2	
L	2							2

A : Matrice initiale

	V	T	E	E	R	D	A	F
L	8	7	0	0	0	0	0	2
T	6	6	9	3	0	0	1	0
S	2	6	3	9	1	0	1	0
H	0	3	5	2	9	1	0	0
E	0	0	4	4	0	7	0	0
A	0	1	0	0	0	0	4	0
L	2	0	0	0	0	0	0	2

B : Matrice transformée

Figure 1.22 : *Construction de la matrice transformée*

Pour déterminer l'alignement local optimal, on démarre à partir de la case qui possède le meilleur score et on se dirige vers la diagonale de préférence. Sinon on effectue les mouvements décrits précédemment pour l'algorithme Needleman et Wunsch.

	V	T	E	E	R	D	A	F
L	8	7	0	0	0	0	0	2
T	6	6	(9)	3	0	0	1	0
S	2	6	3	(9)	1	0	1	0
H	0	3	5	2	(9)	1	0	0
E	0	0	4	4	0	(7)	0	0
A	0	1	0	0	0	0	(4)	0
L	2	0	0	0	0	0	0	(2)

A : le chemin optimal

```
3       8
  EERDAF
  TSHEAL
2       7
```

B : l'alignement local optimal

Figure 1.23 : D*étermination de l'alignement optimal*

La construction du chemin correspondant à l'alignement local optimal, il débute à la position où se trouve le score maximum de la matrice transformée ici c'est le score 9.
Si les régions trouvées entre les deux séquences recouvrent la totalité de celles-ci, alors on peut considérer l'alignement local comme étant un alignement global.

1.7.8 Comparaison avec les Banques de Séquences

Lorsque l'alignement d'une séquence est réalisé contre une banque, le problème du temps de calcul devient prédominant. Les algorithmes précédents sont trop gourmands en ressource. La façon de voir la ressemblance a été posée d'une manière différente pour contourner cet obstacle et des heuristiques ont été proposées.

v Le Logiciel 'FAST Alignment' (FASTA)

L'algorithme [Pearson et Lipman, 88] est basé sur l'identification rapide des mots communs entre la séquence à analyser et les séquences de la banque. La taille du mot recherché est un paramètre que l'on peut choisir entre 1 et 6 (2 pour les protéines, 4 à 6 pour l'ADN). FASTA cherche les mots en se basant sur la technique de Dotplot où il identifie les diagonales ayant le plus grand score et en prenant en considération les pénalités de mésappariement. Les scores dix meilleures diagonales vont être recalculés en utilisant une matrice PAM250. Les dix meilleures

diagonales vont êtres alors rattachées pour former une seule séquence en tenant compte des insertions et délétions.
Ce sont les scores *des diagonales* qui vont être utilisés pour classer les séquences de la base. Les recherches peuvent être optimisé en appliquant un des algorithmes Needleman-Wunsch ou Smith-Waterman mais seulement sur la zone contenant les dix diagonales.
Cet algorithme a toutefois un inconvénient : des régions de faible homologie peuvent être ajoutées.

v Le Logiciel 'Basic Local Alignment Search Tool' (BLAST)

§ Contrairement à FASTA, l'algorithme BLAST [Altschul et autres, 90] ne sélectionne que les séquences qui contiennent des régions de grande similitude. En plus, l'algorithme est basé dans sa conception sur un modèle statistique. L'unité fondamentale de BLAST est le HSP (High-scoring Segment Pair). C'est un couple de segments identifiés sur chacune des séquences comparées, de longueur égale mais non prédéfinie.
Toutefois, cet algorithme présente une limite : pour une longueur de segment égale à *n* (n fixé) une séquence est similaire à 90% par rapport à la séquence de référence, avec un mésappariement tous les *n* acides aminés, elle ne sera pas repérée. Les versions postérieures de BLAST viennent pour régler ce problème, en prenant en compte les délétions et en ajoutant des pénalités en cas d'ouverture de gap, et même en cas d'extension de celui ci. BLAST2 est plus trois fois plus rapide que son aîné.

1.8 La Phylogénie

La similitude entre des mécanismes moléculaires des organismes qui ont été fortement étudiées, suggère que tous les organismes sur terre ont eu un ancêtre commun [Rocha, 00]. Ainsi toutes les espèces ont des liens de parentés et cette relation s'appelle une *phylogénie*. Habituellement le rapport peut être représenté par un *arbre phylogénétique*. Le rôle de la phylogénétique est de construire cet arbre à partir des observations sur les organismes existants (figure 1.24).

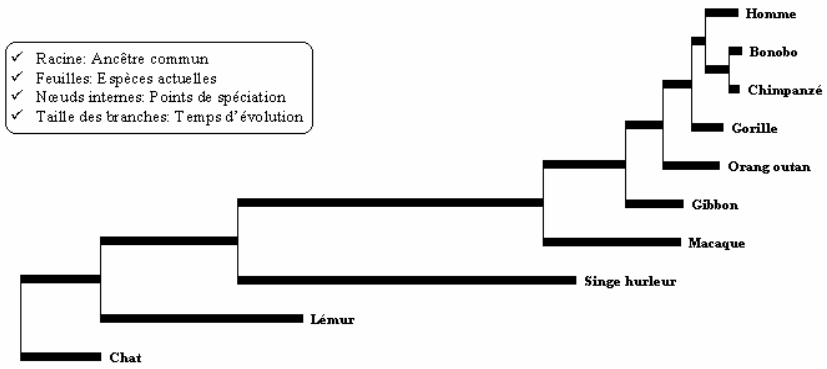


Figure 1.24 : Arbre *phylogénétique des espèces*

Les recherches ont montré que les séquences moléculaires fournissent des ensembles de caractères communs qui peuvent porter une grande quantité d'information. Si l'on possède un ensemble de séquences d'espèces différentes, on est capable de les employer pour fonder une phylogénie probable de l'espèce en question. Ceci assume que les séquences sont descendues d'un certain gène ancêtre commun d'une une espèce ancêtre commune.

1.8.1 Méthodes de Reconstruction d'Arbres

Afin de déterminer les similitudes et liens entre éléments d'un arbre (en général des séquences), plusieurs méthodes ont été suivies telles que :

- La méthode de *parcimonie* essaye de trouver l'arbre le plus parcimonieux, c.à.d, celui qui explique le lien entre deux séquences avec le moins de mutations (substitutions/ insertions/délétions) possibles (fig. 1.25). Cette méthode est valable pour les séquences très poches.

- Les méthodes de *distances* commencent par calculer la distance entre les séquences et essaye de trouver l'arbre qui approche le mieux cette distance. Le calcul des distances peut tout simplement compter le nombre de mésappariements entre les deux séquences dans l'alignement ou utiliser un modèle stochastique tel que le modèle de *Kimura*,où la probabilité d'un changement dépend des bases (A<->G et C<->T sont plus fréquentes), on a deux probabilités. Donc les transitions et les transversions ont une probabilité différente.
- La méthode du *Maximum de vraisemblance* est basée sur un modèle probabiliste évolutif et elle cherche l'évolution la plus probable. Elle cherche à trouver le scénario pour lequel la probabilité d'obtenir actuellement les données observées est la plus grande possible. Cette probabilité s'appelle "Vraisemblance". En d'autres termes, on cherche la valeur qui maximise la probabilité d'observer les résultats effectivement observés. La proximité des séquences n'est pas importante.

Un véritable arbre phylogénique possède une racine (fig. 1.26), ou l'ancêtre terminal de toutes les séquences. Quelques algorithmes fournissent des informations au sujet de l'endroit de la racine. D'autres, basé *parcimonie*, sont non informatifs au sujet de sa position.

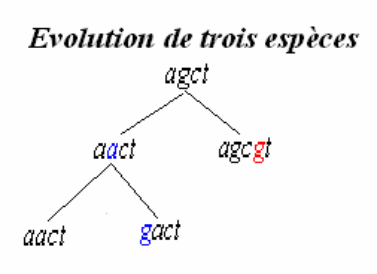


Figure 1.25: *Accumulation des Substitution/insertion*

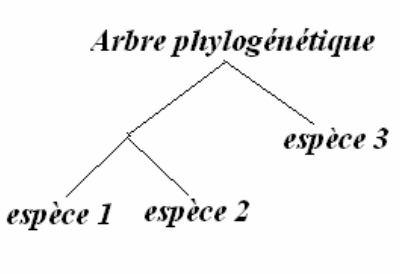


Figure 1.26 : *Apparition des espèces*

1.8.2 Unweighted Pair Group Method with Arithmetic Mean (UPGMA)

Cette méthode est utilisée pour reconstruire des arbres phylogénétiques si les séquences ne sont pas trop divergentes.
UPGMA emploie un algorithme de clustering qui utilise des moyennes arithmétiques. On procède par une clustérisation des séquences, à chaque fois que l'on fusionne deux clusters les plus proches, on crée un nouveau noeud sur l'arbre.
L'arbre peut être imaginé comme étant dirigé vers le haut, où chaque noeud est ajouté au-dessus des autres, et les longueurs des arcs sont déterminées par la différence dans les tailles des noeuds au dessus et au bas d'un arc. UPGMA fournit un arbre sans racine.
D'abord on définit *la distance dij* entre deux clusters C_i et C_j c'est la moyenne distance entre les paires de séquences de chaque cluster:

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq} \qquad (1.5)$$

Où $|C_i|$ et $|C_j|$ dénotent le nombre de séquences dans les clusters i et j respectivement.
Si $C_k = C_i \cup C_j$ et si C_l est n'importe quel autre cluster, alors :

$$d_{kl} = \frac{d_{il}|C_i| + d_{jl}|C_j|}{|C_i| + |C_j|} \qquad (1.6)$$

Exemple **:** Si on considère la matrice de distances associées à un groupe de 6 éléments et que l'on veuille obtenir l'arbre associé:

	A	B	C	D	E
B	2				
C	4	4			
D	6	6	6		
E	6	6	6	4	
F	8	8	8	8	8

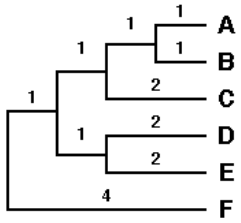


Figure 1.27 : *Un arbre phylogénétique construit par la méthode UPGMA*

1.8.3 Neighbor-Joining (NJ)

Elle est basée sur la recherche d'une paire d'OTU (operational taxonomic units : feuille de l'arbre) qui minimise la longueur totale des branches de l'arbre et ceci à chaque étape de regroupement.
Cette méthode développée par Saitou et Nei [Saitou et Nei, 87). Elle tente de corriger la méthode UPGMA afin d'autoriser un taux de mutation différent sur les branches. Elle construit un arbre phylogénétique sans racine à partir d'un indice d'écart (par exemple distance ou dissimilitude entre séquences).
Les données initiales permettent de construire une matrice qui donne un arbre en étoile. Cette matrice de distances est ensuite corrigée afin de prendre en compte la divergence moyenne de chacune des séquences avec les autres. L'arbre est alors reconstruit en reliant les séquences les plus proches dans cette nouvelle matrice. Lorsque deux séquences sont liées, le noeud représentant leur ancêtre commun est ajouté à l'arbre tandis que les deux feuilles sont enlevées. Ce processus convertit l'ancêtre commun en un noeud terminal dans un arbre de taille réduite.
Exemple :

	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8

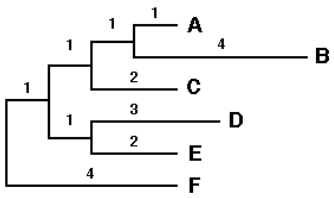


Figure 1.28 : *Un arbre phylogénétique construit par la méthode NJ*

1.9 L'Alignement Multiple de Séquences

La mise en évidence de similitude entre séquences sera renforcée si plusieurs séquences voisines issues de plusieurs espèces partagent des éléments en commun. L'alignement multiple permet d'aligner globalement ces séquences et conduire en particulier le repérage des séquences fortement conservées dans une famille des protéines de gènes. Les conservations repérées ont une importance particulière dans la fonction catalytique ou indispensable à la stabilité d'une structure 3D de la protéine. De même, l'étude de la ressemblance et la diversité autour de ces séquences communes, permet par de nombreuses méthodes d'aborder la filiation évolutive de ces gènes [Nei, 87] et par là même conduire à des études phylogénétiques de plus en plus précises. Les différents objectifs visés par un alignement multiple de séquences sont:

- Alignement de protéines homologues
- Identification de résidus importants (conservés)
- Extraction de motifs communs
- Génération de séquences *consensus*
- Création de signatures fonctionnelles : constitution d'un dictionnaire de signatures

L'alignement fera l'objet d'une étude plus exhaustive au niveau du chapitre prochain, où un exposé complet lui sera consacré.

1.10 Le Réseau Thématique en Bioinformatique

Les différentes méthodes d'analyses citées précédemment sont complémentaires les unes par rapport aux autres. Certaines constituent comme étape initiale, d'autres se présentent comme étape obligatoire pour atteindre un objectif fixé (Figure 1.29).

Ayant rassemblé un ensemble de séquences, on peut penser à leur stockage dans une base de données pour les partager avec d'autres utilisateurs comme on peut procéder à un alignement deux à deux pour chercher une homologie si elle existe. Ces séquences peuvent être alignées par un algorithme d'alignement multiple afin d'inférer un lien de parenté à travers un arbre phylogénétique ou de déterminer des motifs communs. L'annotation génomique aide à l'identification de la structure et la fonction des séquences protéiques homologues et facilite la recherche dans les banques des données.

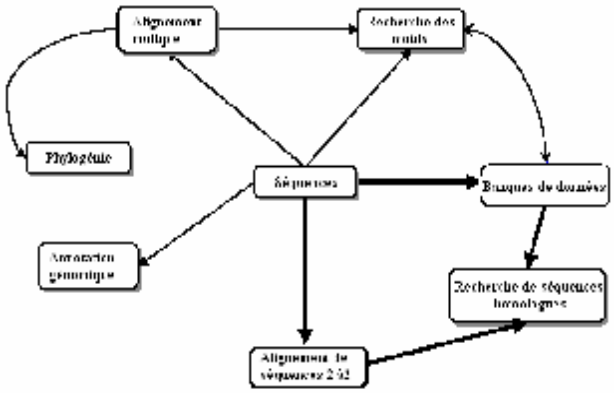


Figure 1.29 : *Réseau des thèmes bioinformatiques*

1.11 Formalisme de la Recherche en Bioinformatique

Comment un bioinformaticien réagit face à une question biologique ?
En général, les questions biologiques sont souvent formulées comme suit :

- À quoi ressemble une séquence biologique ?
- En quoi elle diffère des autres séquences ?
- Est-ce qu'elle est déjà répertoriée dans une banque ?
- Quelle est sa structure secondaire ? tertiaire ?
- Quelle est sa fonction ?
- Etc.

Un bio-informaticien équipé de techniques et de méthodes (généralement des algorithmes et programmes) d'un côté et d'un ensemble de séquences (séquence requête et banque de données) d'un autre coté, va procéder à une comparaison dans la quelle il se fixe un objectif à atteindre (chercher homologie, motifs, liens de parenté, etc.).

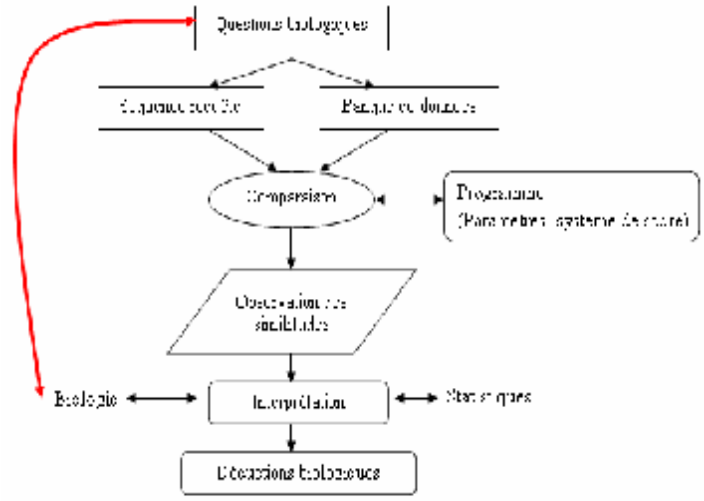


Figure 1.30 : *Formalisme des recherches bioinformatiques*

Le résultat d'une comparaison est souvent exprimé comme un degré de similitude ou de ressemblance, ajusté par des lois statistiques. L'*interprétation purement biologique* est réservée au biologiste.

1.12 Conclusion

Avec le déluge courant des données biologiques, les méthodes informatiques sont devenues indispensables aux investigations biologiques. À l'origine développée pour l'analyse des séquences biologiques, la bioinformatique couvre maintenant un large éventail de domaines comprenant la biologie structurelle, génomique et l'étude de l'expression des gènes. Dans ce chapitre, nous avons fourni une introduction sur la biologie moléculaire (notions de base) et une vue d'ensemble de l'état actuel de ce domaine. En particulier, nous avons montré les différentes pratiques d'analyse d'informations et de bases de données biologiques qui sont généralement employés.
Nous avons en particulier présenté la nature de l'information utilisée par un bio-informaticien et les domaines d'utilisation de la bioinformatique. Parmi les thèmes abordés par la

bioinformatique, il y a l'alignement multiple de séquences et qui constitue pour certaines pratiques une étape primordiale pour sa progression. Pour ce thème, nous avons consacré un chapitre entier, le suivant.

Chapitre 2 : Alignement Multiple de Séquences

2.1 Introduction

L'alignement multiple des séquences d'ADN ou de protéines est une des techniques les plus utilisées dans l'analyse de séquence. Il est considéré parmi les problèmes les plus difficiles en bioinformatique.
L'alignement multiple de séquences (Multiple Sequence Alignment : MSA) est une tâche cruciale et très importante en biologie moléculaire. MSA offre aux biologistes un moyen pour analyser des séquences d'ADN ou de protéines et de déterminer par la suite leur degré d'homologie ou de divergence. *MSA* est utilisé dans la construction des arbres phylogénétiques et identifier les motifs dans des familles de protéines, ceci permet de prédire leur aspect structurel et fonctionnel.
La qualité d'une comparaison ou d'une prédiction dépend de la qualité du *MSA*. Jusque récemment le choix d'une méthode pour la construction des alignements multiples de séquence (MSAs) a été limité à une poignée de packages mais une augmentation récente des données génomique a poussée l'élaboration de plusieurs nouvelles méthodes, plus précises et plus rapides que les anciennes. Dans la pratique, ce large choix a également rendu difficile le choix objectif de la méthode appropriée pour un problème spécifique.
Pendant la dernière décennie, plus de 50 méthodes ont été décrites dans ce domaine et 20 uniquement pendant l'année 2005 [Wallace et autres, 06]. Ce nombre risque d'augmenter car aucune parmi elles n'est totalement efficace pour tout type de séquences.
Pour étudier l'évolution de gène à travers un éventail d'organismes, les biologistes ont besoin des outils précis pour l'alignement multiple de séquences des familles de protéines. L'obtention des alignements précis, cependant, est un problème informatique difficile en raison non seulement du coût informatique élevé mais également du manque de fonctions objectives appropriées pour la qualité de mesure d'alignement. Il a été démontré que MSA est un problème NP-Complet [Wang, et Jiang, 94]. Donc la résolution d'un MSA par une méthode exacte paraît une mission difficile voire impossible. Les méthodes proposées dans la littérature sont en général des heuristiques qui tentent d'approcher un alignement optimal sans l'atteindre réellement ceci est dû à la complexité des données biologiques.
Dans ce chapitre, nous allons commencer par exposer les principales fonctions objectif utilisées puis les méthodes les plus récentes conçues pour résoudre le problème de MSA selon les approches utilisées.

2.2 Définition Formelle d'un Alignement Multiple

Un alignement multiple de séquences (MSA) est en réalité un agencement de plusieurs séquences biologiques dans le but de mettre en valeur leur similitude et convergence.
Un alignement multiple dépend du nombre de séquences ainsi que de leur longueur. Un MSA est souvent facile à réaliser lorsque les séquences sont issues de la même famille dans le cas contraire, séquences divergentes, MSA devient très délicat car il est difficile de repérer les zones homologues et les aligner ensemble.

La figure 2.1 présente un alignement multiple de séquences protéiques. Les séquences sont plus ou moins divergentes. Si l'on considère la première séquence (en haut de la figure) comme séquence ancêtre, les quatre autres seraient parues dans la nature par une succession d'opération de mutations, insertions ou délétions. Les colonnes (toutes en C ou W) sont considérées les zones conservées, et elles reflètent généralement une fonction commune entre ces séquences.

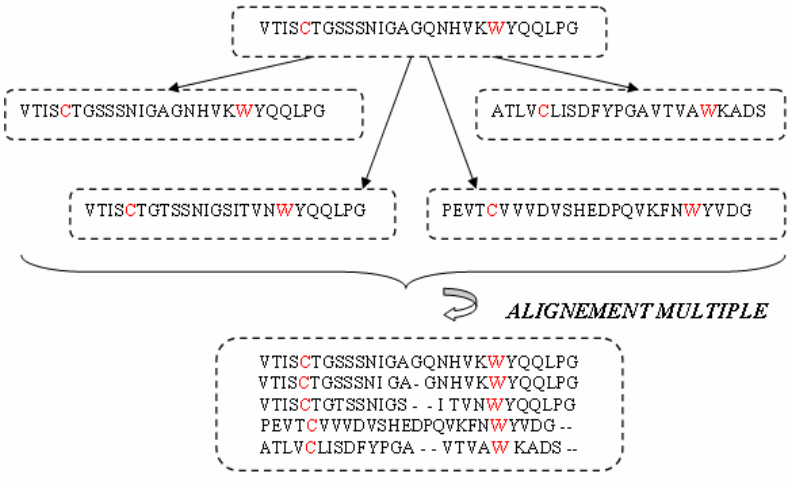


Figure 2.1 : *L'Alignement multiple de séquences protéiques.*

Pour pouvoir manipuler ces séquences et les analyser *in silico*, une présentation formelle est devenue nécessaire.

Définition 2.1: Soit Σ un alphabet sans le caractère '-' et $\Sigma' = \Sigma \cup \{'-'\}$, en plus, soient *S1*, … , *Sk* les K séquences sur Σ avec des longueurs n1,…… , nk. Soit A l'alignement multiple de *S1*,… , *Sk*. *A* est une matrice de dimension K×L avec les propriétés suivantes [Reinert, 03] :

- Max {n1,…,nk } ≤ L ≤ $\sum_{i=1}^{K} ni$.
- A[i][j] ∈ Σ' ∀ 1≤ i ≤ K; 1≤ j ≤ L.
- La $i^{ème}$ ligne Ai sans gap est égale à Si.
- Il n'a y a pas de colonnes ne contenant que de gaps.

2.3 Évaluation de MSA et Fonctions Objectif

En général, l'alignement optimal est celui qui optimise une fonction objectif (FO). Une fonction objectif ou *méthode de score* est une expression mathématique qui essaye d'attribuer une évaluation quantitative à la signification biologique et évolutionnaire d'un alignement.
Ainsi, ces méthodes tentent de trouver le MSA optimal qui maximise ou minimise une FO. Le choix d'une FO peut s'avérer une tâche très délicate car le problème est purement biologique. Comment s'assurer mathématiquement qu'un alignement est correct biologiquement ? D'où l'apparition d'un nombre non négligeable de F.Os qui tentent toutes de définir un alignement optimal mathématiquement, mais malheureusement l'optimum mathématique coïncide rarement avec l'optimum biologique [Notredame, 02] mais les fonctions objectif nous permettent de s'approcher de celui ci.
Toutes les méthodes de score essayent de donner une évaluation quantitative à la signification biologique et évolutionnaire d'un alignement. Cependant, en raison de la nature complexe des données biologiques, toutes les méthodes de score ont leurs limitations. Il n'y a aucune norme universelle pour mesurer la qualité d'un alignement multiple de séquences.

Définition 2.2 : En général, une fonction objectif est une expression mathématique qui permet d'évaluer la qualité d'un résultat d'un traitement. Dans notre cas, elle va servir à l'évaluation des alignements multiples obtenus et de décider lequel est meilleur.

Soit $\boldsymbol{f}: \boldsymbol{A} \rightarrow \Re$. $\boldsymbol{f}$ est une fonction qui attribue à chaque élément de l'ensemble des alignements obtenus $\boldsymbol{A}$ = {A1, A2,…..An} une valeur réelle qui indique approximativement sa qualité.

Le but de $\boldsymbol{f}$ est de permettre de trouver l'alignement $Ai \in \boldsymbol{A}$ tel que

$$f(Ai) = \begin{cases} Max(f(Aj, j = 1,..N) \\ ou \\ Min(f(Aj, j = 1,..N) \end{cases} \qquad (2.1)$$

Max ou *Min* selon si l'on veut mesurer la similitude ou la distance entre les séquences alignées.
Dans ce qui suit, il y a une exposition des différentes fonctions utilisées pour des méthodes différentes.

2.3.1 La Somme des Paires (Sum of Pairs : SP)

C'est la fonction la plus répandue, elle est simple à réaliser.
Elle consiste à sommer les scores des paires de séquences alignées dans un alignement multiple Ai avec Ai $\in \boldsymbol{A}$ (ceci par référence aux définitions précédentes).
Soit Ai un alignement de K séquences {*S1*,…, *Sk*} ;

$$\boldsymbol{SP}(\text{Ai}) = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} Sc(SiSj). \qquad (2.2)$$

Avec Sc(*SiSj*) est le score de l'alignement de la paire des séquences Si et Sj. Ce score peut être calculé par une mesure de distance ou de similitude.
Généralement, on utilise une fonction d'identité qu'il faut bien sûr la maximiser ou une fonction de distance que l'on doit minimiser.

Exemple : soit l'alignement A^* suivant :

S1 : a c - c d b -
S2 : - c - a d b d
S3 : a - b c d a d

Si on considère que la fonction des distances est :

d (x, x) =0, d(x, y)= 1 pour $x \neq y$ (y compris les gaps)

Le ***SP*** de l'alignement A^* = Sc(S1S2) + Sc(S1S3) + Sc(S2S3)
= 3 + 4 + 5 = 12.

Cette fonction a la particularité de prendre en considération toutes les informations sur les alignements des paires de séquences.
Dans certaines versions de SP, on introduit une pénalité pour les gaps. Plusieurs fonctions d'évaluations de gap existent dans la littérature telle que la fonction *affine* (voir chapitre 1) *;* à la rencontre d'un gap, on affecte une pénalité GOP (Gap Open Penalty) et à chaque extension de celui ci, on affecte une pénalité GEP (Gap Extension Penalty)

Inconvénient : L'un des problèmes de cette fonction [Lipman et autres, 89] est que si dans l'ensemble des séquences à aligner, il y a une grande similitude entre un sous- ensemble des séquences, cette similitude va influencer le score de l'alignement global au profit de ce sous-ensemble.

2.3.2 Weighted Sum of Pairs (WSP)

C'est une amélioration de SP, introduite par [Altschul et autres, 89]. Elle consiste à attribuer des poids aux différentes paires de séquences à aligner. Ces poids peuvent être obtenus en dressant un arbre phylogénétique reliant ces séquences selon leurs similitudes et distances. Deux méthodes sont souvent utilisées pour la construction d'un tel arbre et qui sont N.J [Saitou et Nei, 87] et UPGMA [Sneath et Sokal, 73] présentées dans le chapitre précédent.
Le choix d'une ou de l'autre méthode dépend de la nature des séquences à aligner ; si celles ci ont plus ou moins de similitudes entre elles.
La formule générale de cette fonction est :

$$\boldsymbol{WSP}(\mathrm{A}) = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} Wij * sc(SiSj). \qquad (2.3)$$

On peut l'écrire encore avec la formule suivante par référence à la définition1 :

$$\boldsymbol{WSP}(\mathrm{A}) = \sum_{h=1}^{L} \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} Wij * sc(A[i,h], A[j,h]). \qquad (2.4)$$

Avec $sc(A[i,h], A[j,h])$ = le score de la paire de résidus alignés dont les coordonnés sont (i,h) et (j,h).
Wij est le poids déterminé à partir de l'arbre phylogénique reliant les K séquences.

2.3.3 La Fonction Consensus

D'après Martin Tompa [Tompa, 00], il est parfois préférable de passer par une séquence consensus ***C*** pour évaluer la qualité d'un alignement multiple de plusieurs séquences. L'idée consiste donc à trouver une séquence ***C*** : $c_1c_2\ c_3$ …..c_L où L est la longueur de l'alignement, de telle sorte que chaque caractère c_i de ***C*** minimise le score de celle ci.

Définition 2.3 : Ayant un alignement de N séquences $S_1S_2\ S_{3.....}\ S_{N,}$ le caractère consensus c_i d'une colonne i , i=1,..L, est celui qui minimise la somme des distances entre lui et les autres caractères de cette colonne ;

$$d(\mathrm{i}) = \sum_{j=1}^{N} d(S_J'[i], c_i) \qquad (2.5)$$

avec S' est S alignée. La séquence consensus étant *C* : $c_1c_2\ c_3$ …..c_L, les erreurs d'alignements sont définis par $\sum_{i=1}^{L} d(i)$

Exemple : soit l'alignement suivant :

S1 :	**a**	**c**	**-**	**c**	**d**	**b**	**-**
S2 :	**-**	**c**	**-**	**a**	**d**	**b**	**d**
S3 :	**a**	**-**	**b**	**c**	**d**	**a**	**d**
Consensus :	**a**	**c**	**-**	**c**	**d**	**b**	**d**

Si on considère que la fonction des distances est :

d(x,x)=0, d(x,y)= 1 pour $x \neq y$ alors $\sum_{i=1}^{L} d(i) = 6.$

La séquence Consensus sert à déterminer le score de l'alignement multiple, en sommant les scores des alignements de paires entre chaque séquence de l'alignement multiple et la séquence Consensus.

2.3.4 La Fonction Profil

Cette fonction a pour objectif de calculer le profil d'un alignement A [Reinert et autres, 03]. Ce profil est une représentation numérique d'un MSA qui représente les caractéristiques communes d'une famille de protéines. La fonction Profil est utilisée pour déterminer le degré d'appartenance d'une protéine à une famille. On peut signaler qu'il est utile dans l'alignement des séquences pas trop divergentes. Il permet de déterminer des régions conservées dans une séquence ou plusieurs. C'est la somme des fréquences d'apparition de chaque résidu dans chaque colonne de l'alignement.

Exemple : soient les scores de similarités :

sc(a, b) = s(x,-) = sc(-, x) = -1,
sc(a, c) = -3, sc(b, c) = -2
sc(a, a) = sc(b, b) = sc(c, c) = 2.

Soient quatre séquences a1, a2, a3, et a4:

Les séquences

S1 = a b c - a
S2 = a b a b a
S3 = a c c b -
S4 = c b – b c

Matrice des fréquences

Profil :	C1	C2	C3	C4	C5
a :	0.75		0.25		
b :		0.75		0.75	
c :	0.25	0.25			0.25
- :			0.25	0.25	0.25

Calcul du score du Profil

Colonne		Valeur de la colonne	
a	**1**	= 0.75*2 -0.25*3	= 0.75
a		= -1.0*1	= -1.0
b	**2**	= 0.75*2 -0.25*2	= 1.0
-	**3**	= -0.25*1 - 0. 50*1 - 0.25*1	= -1.0
b	**4**	= 0.75*2 - 0.25*1	= 1.25
c	**5**	= 0.25*2 - 0. 50*3 - 0.25*1	= -1.25

Le score du profil = -0.25

Le score d'un profil peut être trouvé par la formule suivante :

$$\text{Prof (A)} = \sum_{i=1}^{L} \sum_{x=1} p_{ix} * d(x, y) \qquad (2.6)$$

§ Avec quelques modifications et une nouvelle appellation, la méthode ***Muscle*** [Edgar, 04] utilise une fonction Profil pour évaluer l'alignement d'une paire de séquence pour un alignement multiple (à l'étape 3 de la méthode). Ici elle est appelée la fonction ***Log-Expectation*** **(LE)** dont la formule est la suivante :

$$LE^{xy} = (1-f^{x}_{G})\,(1-f^{y}_{G}) * \log \sum_{i}\sum_{j} f^{x}_{i}\, f^{x}_{j}\, p_{ij} \;/\; p_i\, p_j \qquad (2.7)$$

Avec i et j des bases d'acides aminés.
Et f^x_i et f^x_G sont les fréquences des résidus i et les gaps observés dans la colonne x
p_i est la probabilité de i d'après la matrice PAM 240 VTML.
p_{ij} est la probabilité que i et j soient alignés l'un avec l'autre.

2.3.5 La Mesure d'Entropie

La mesure d'entropie en MSA est la somme d'entropie des colonnes [Nicholas et autres, 02]. L'entropie est en générale une mesure de variation des informations utilisée souvent dans la théorie de l'information introduite par **Shannon**. Pour chaque colonne, l'entropie est calculée par la formule suivante :

$$\text{Entropie } (A[:][i]) = -\sum_a c_{ia} * \log(p_{ia}) \qquad (2.8)$$

Où c_{ia} est le nombre du caractère *a* dans la colonne A[:][i], p_{ia} est la probabilité du caractère *a* dans la colonne i :

$$p_{ia} = c_{ia} / \sum_{a'} c_{ia'} \qquad (2.9)$$

Une colonne reçoit un zéro d'entropie si tous les caractères alignés dans la colonne sont identiques (p_{ia} =1). Plus la colonne est variable, plus l'entropie est haute. Le but est donc de trouver l'alignement qui minimise l'entropie de l'ensemble des colonnes d'un alignement.

Remarque : cette fonction n'utilise pas les scores des matrices de substitutions.

2.3.6 La Fonction Coffee

Coffee (Consistency-based Objective Function For alignmEnt Evaluation) [Notredame et autres, 98]. Elle fournit un score global de l'alignement, appliquée pour l'évaluation les alignements produits par la méthode *SAGA* [Notredame et autres, 96].
Cette fonction a la particularité d'avoir utilisé un nouveau concept : 'Consistency' en anglais dont la signification dans ce contexte est 'Consistance'. Ce concept fut introduit la première fois par [Gotoh, 90].

§ *Concept « Basée consistance » (Consistency-based)*
" La prévention est la meilleure médecine" telle est la devise d'une fonction objectif basée consistance [DO et autres, 05].
Pour n'importe quel alignement multiple, les alignements par paires induits, sont nécessairement cohérents c'est-à-dire, ayant un alignement multiple contenant trois séquences X, Y, et Z, si la position Xi s'aligne avec la position Zk et Zk s'aligne avec Yj dans les alignements respectifs X-Z et Z-Y, alors Xi doit être alignée avec Yj dans l'alignement X-Y.
Les techniques basées consistance appliquent ce principe à l'envers. Des séquences intermédiaires sont utilisées pour guider l'alignement par paires de X et de Y, tel que il est nécessaire pendant les étapes d'un alignement progressif. Ajuster le score d'une paire de résidus Xi - Yj en s'appuyant sur la position Zk qui est alignée à la fois avec Xi et Yj, les fonctions objectif basées consistance incorporent l'information multiple de séquence pour évaluer des alignements par paires.
C'est une fonction qui nécessite un traitement préalable des informations contenues dans les séquences à aligner. Pour mieux comprendre, voila les étapes nécessaires à la réalisation de ce score :

§ *Construction de la Bibliothèque*

On construit une structure dite *bibliothèque*, qui contient tous les alignements possibles de deux paires de séquences de l'ensemble fourni au départ. Ces alignements de paires peuvent être construits par une méthode déjà existante telle que *ClustalW* [Thompson et autres, 94]. Donc, la bibliothèque pour *N* séquences contient *N(N-1)/2* alignements auxquels on attribue à chaque alignement un poids *Wij* qui correspond au nombre de paires de résidus alignés correctement (le nombre d'identité) dans l'alignements des séquences *Si* et *Sj* .

La bibliothèque aura la forme suivante :

Alignement de 2 séquences	Poids/identité
A __ ___x_ ___ B _____ y_ _	$W_{AB} = 70$
A ___ _ __ _ _ C ____ ___ __	$W_{AC} = 80$
……... ……...	……
B _____ _____ D ___ _ _____	……..

Une fois que la bibliothèque est construite, elle sera utilisée et consultée pour l'évaluation de chaque alignement multiple produit par l'algorithme SAGA. Cette bibliothèque va servir en réalité comme une matrice de substitution pour définir le poids à accorder à une paire de résidus. Les informations utilisées sont désormais déduites de l'ensemble des séquences et non pas de l'extérieur.

§ *Évaluation d'un alignement*

Ayant obtenu un alignement multiple donné, celui-ci va être évalué paire par paire de séquences. L'idée est de comparer chaque paire de résidus de l'alignement de deux de séquences avec les paires de résidus contenus dans la bibliothèque. Cette fonction va évaluer la consistance entre l'alignement multiple et les alignements des paires de séquences contenus dans la bibliothèque.

La formule suivante sert pour évaluer le score global d'un alignement :

$$\textbf{\textit{Score\_coffee}} = \left[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} Wij \times Score(Aij)\right] / \left[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} Wij \times Len(Aij)\right] \qquad (2.10)$$

Aij : l'alignement de deux séquences dans l'alignement global.

Wij : le poids de l'alignement des séquences Si et Sj représenté dans la bibliothèque. ***Score(Aij)*** : nombre des paires de résidus alignés dans ***Aij*** et qui apparaissent dans la bibliothèque Les résidus sont identifiés par leur position dans leur séquence respective.

Len(Aij) : la longueur de l'alignement Aij des séquences Si et Sj.

Cette définition permet de donner un score global pour un alignement, dans le cas d'un score local, une autre fonction de score a été donnée :

$$\text{Score de résidu } (S^{x}_{i}) = \sum_{j=1, j\neq i}^{N} Wij \times occurence(A_{i,j}) \Big/ \sum_{j=1, j\neq i}^{N} Wij \qquad (2.11)$$

occurrence $(A^{x,y}_{i,j})$ = Nombre des occurrences des paires $A^{x,y}_{i,j}$ dans la bibliothèque (0 ou 1)

Score de séquence = c'est la somme des scores des résidus divisé par le nombre des résidus dans la séquence.

2.4 Les Approches d'Alignements Multiple de Séquences

Dans la littérature, on rencontre trois catégories essentielles ou approches suivies pour construire un MSA. Néanmoins, ces approches sont parfois fusionnées, concaténées ou/et associées pour construire une seule méthode [Edgar, 04].
On distingue l'approche Exacte qui tente de donner plus de longévité à la programmation dynamique dans ce domaine et de déterminer un alignement optimal proprement dit comme elle le fait pour aligner deux séquences. De l'autre côté, on rencontre des heuristiques qui à leur tour se bifurquent en deux approches : Progressive et Itérative.
Les méthodes qui suivent l'approche progressive, sont reconnues d'être très rapides [Rong et Hansen, 04] et donnent des résultats assez satisfaisants mais leur inconvénient est le fait de s'arrêter sur les minima locaux et si une erreur est commise au début de l'alignement, elle va se propager sur l'alignement final.
L'approche itérative est une manière très simple, rapide et efficace permettant d'améliorer des méthodes d'alignement multiples. L'itération peut être employée pour améliorer le résultat d'un logiciel existant avec n'importe quelle fonction objectif. Elle peut également être incorporée à une stratégie progressive d'alignement pour établir des alignements à partir de zéro pour produire encore de meilleurs résultats [Wallace, 04].

2.4.1 L'Approche Exacte

L'approche exacte n'est autre qu'une généralisation des méthodes de programmation dynamique de [Needleman et Wunsh, 70] et [Smith et Waterman, 81] décrites au niveau du chapitre 1 section 1.73.
La méthode de programmation dynamique utilisée pour aligner deux séquences, a été appliquée à l'alignement de plusieurs séquences (N dimensions) tels que MSA [Lipman et autres, 89], DCA [Stoye et autres, 97].
Ce type de méthodes représente de gros problèmes : Le temps de calcul et l'espace mémoire.

- Dans la pratique, un alignement devient délicat pour un nombre de séquence $N > 3$, et même impossible pour $N = 10$
- Pour N séquences de longueur L, l'alignement optimal (au sens mathématique) nécessite :
    - Un temps de calcul proportionnel à $2^n L^n$.
    - Un espace mémoire proportionnel à L^n.
- Exemple : pour 10 séquences de 100 résidus, et 10^{-9} secondes de temps de calcul par colonne, nécessite alors :
  Temps total = $2^{10} * 100^{10} * 10^{-9} \approx 10^{14}$ s (> 3 10^6 années)
  Espace mémoire = 10^{11} GB.

Le problème de l'alignement multiple exacte a été démontré être un problème *NP-complet*. D'où le recours aux méthodes approchées ou heuristiques.

2.4.2 L'Approche Itérative

L'approche itérative a été employée plusieurs fois comme méthode d'optimisation pour produire des alignements multiples. Parfois elle est utilisée seule ou en combinaison avec d'autres méthodes. L'itération a un grand avantage parce qu'elle est souvent très simple soit en termes de code des algorithmes, soit en termes de complexité temporelle et spatiale.
Les étapes d'un alignement itératif :

- Repérer les deux séquences avec la plus forte similarité et les aligner avec une méthode de programmation dynamique.
- Trouver la séquence qui est la plus proche du profil obtenu avec les 2 séquences précédentes et l'aligner avec les deux autres par une méthode d'alignement profil-séquence.

- Répéter ceci jusqu'à ce que toutes les N séquences soient inclues dans l'alignement multiple
- Enlever la séquence S1 et la réaligner avec le profil obtenu avec les séquences de S2…Sn
  - Répéter ceci pour toutes les autres séquences de S2 à Sn.
- Répéter l'étape précédente un certain nombre de fois ou arrêter le processus à convergence du score de l'alignement.

2.4.3 L'Approche Progressive

L'alignement progressif [Taylor, 87] est l'heuristique la plus répandue pour aligner un grand nombre de séquences. L'alignement multiple est construit progressivement en alignant des paires de séquences suivies des paires d'alignements/profils. Un arbre guide détermine l'ordre dans lequel les séquences vont être alignées, les plus proches d'abord. Cette technique est employée dans différents packages d'alignement multiple tels que MULTALIGN [Barton et Sternberg, 87], ClustalW [Thompson et autres, 94], et T-Coffee [Notredame et autres, 00] …etc.
Un alignement multiple progressif suit les étapes suivantes [Feng et Doolittle, 87]:

- Alignement deux à deux de toutes les séquences.
- Construction d'une matrice de distances entre toutes les séquences.
- Détermination de l'ordre selon lequel les séquences seront alignées en utilisant la notion de clustering :
  - Alignement de deux séquences
  - Alignement d'une séquence et d'un profil
  - Alignement de deux profils

Problèmes majeurs des alignements multiples progressifs :

- Les alignements entre sous-groupes sont gelés. Si une erreur est produite au début, aucune modification ou correction ultérieure n'est possible.
- Les erreurs dans les alignements des sous groupes initiaux se propagent dans tous l'alignement.

2.5 Les Méthodes d'Alignement Multiple Exactes

2.5.1 La Méthode MSA

C'est une tentative de rendre les algorithmes de la programmation dynamique conçus pour aligner deux séquences, opérationnels pour un alignement multiple. Sachant que la complexité temporelle et spatiale augmente proportionnellement avec le nombre et la longueur des séquences à aligner. Même la matrice de score devient elle aussi multidimensionnelle.
Le programme de **MSA** emploie un algorithme intelligent pour réduire le volume de la matrice de la programmation dynamique multidimensionnelle. L'algorithme de Carrillo et de Lipman [Lipman et autres, 89] était mis en application dans MSA.
Le score d'un alignement multiple généré par une heuristique est la somme des scores de tous alignements deux à deux définis pour l'alignement multiple. Sachant que :

- le score *SP* pour toute paire de séquences extraite de l'alignement multiple optimal, devrait être inférieur au score *SP* optimal de l'alignement de paires de séquences.
- le score *SP* total d'un alignement optimal devrait être plus grand que celui d'un alignement obtenu par des méthodes heuristiques.

En plaçant la limite inférieure et la limite supérieure, seulement un espace restreint doit être exploré dans la table de score multidimensionnelle [Layeb, 05]. Toutes ces considérations ont participé à la réduction du temps de calcul d'une manière significative.

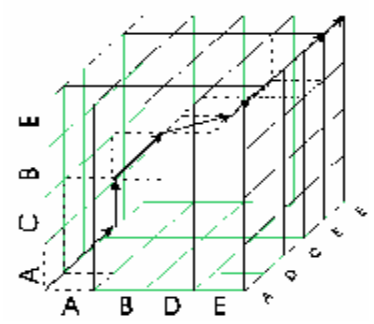


Figure 2.2 : *Tracé d'un alignement tridimensionnel*

D'une manière générale, MSA produira des alignements meilleurs que la plupart des programmes d'alignement multiple de séquences. L'inconvénient de MSA est qu'il exige une énorme quantité en temps machine et en mémoire (en particulier pour des séquences divergentes) et directement proportionnelle à la longueur et le nombre des séquences. Toutes ces contraintes font du MSA un programme inexploitable pour de longues et nombreuses séquences malgré sa capacité à déterminer l'alignement multiple optimal.

2.5.2 La Méthode DCA

DCA (Divide and Conquer Algorithm) [Stoye et autres, 97], c'est une heuristique basée sur l'idée « diviser puis conquérir ». Le principe consiste à découper les séquences à aligner en sous ensembles de segments (Fig.2.3). Ces segments doivent avoir une taille assez petite pour faciliter leur traitement par MSA. Les sous alignements produits sont alors concaténés pour former un seul alignement multiple final.

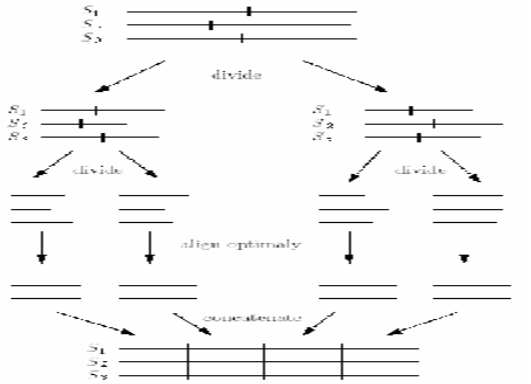


Figure 2.3 : *Déroulement de l'algorithme DCA*

Comme étant une méthode exacte, elle hérite des même inconvénients des méthodes de ce type : la complexité temporelle et spatiale

2.6 Les Méthodes d'Alignements Itératives

2.6.1 La Méthode SAGA

C'est un algorithme génétique itératif [Notredame et Higgins, 96] qui démarre par une population d'alignement, puis raffine les solutions par des opérateurs spécifiques tels que la mutation jusqu'à l'obtention d'une solution plus ou moins optimale. C'est une heuristique qui se rapproche de la solution optimale mais aucune certitude qu'elle le soit réellement.

Algorithm 2.1: SAGA	
Initialisation	
Evaluation:	1. create G0
	2. evaluate the population of generation n (Gn)
	3. if the population is stabilised then END
Breeding*:*	4. select the individuals to replace
	5. evaluate the expected offspring (EO)
	6. select the parent(s) from Gn
	7. select the operator
	8. generate the new child
	9. keep or discard the new child in Gn+1
	10. goto 6 until all the children have been successfully put into Gn+1
	11. n = n+1
	12. goto Evaluation
End :	13. end

Chaque génération est évaluée par la fonction objectif (*WSP*) pour déterminer quels sont les alignements les plus acceptables et aptes à passer dans la génération suivante. Ceci est appelé le phénomène de la sélection biologique *« seuls les meilleurs survivent ».*

G0, Gn et Gn+1 sont respectivement la population initiale, courante et la population de la génération future. L'algorithme commence par la génération des individus de la population G0 d'une façon aléatoire, qui vont subir immédiatement une évaluation afin de déterminer le niveau de ces solutions. Si les solutions obtenues ont atteint un seuil d'optimalité alors l'algorithme s'arrête sinon on passe à l'étape suivante et qui consiste en la génération de nouvelles solutions en faisant subir à la population courante une série d'opérations génétiques telles que la sélection, croisement et mutation. Les nouvelles solutions obtenues ne sont maintenues dans la nouvelle génération que si elles présentent un certain niveau d'efficacité. L'algorithme s'arrête après un certain nombre d'itération. La meilleure solution de la dernière population serait considérée la solution optimale de l'algorithme.

SAGA a la particularité de pouvoir optimiser n'importe quelle fonction objectif. Plus tard [Notredame et autres, 98] ont utilisé SAGA pour valider une nouvelle fonction objectif : *Coffee*. Les résultats sont considérés nettement meilleurs que ceux fournis par la première approche.

2.6.2 La Méthode DIALIGN

DIALIGN est une méthode pour l'alignement multiple développée par [Morgenstern et autres, 98]. L'algorithme de DIALIGN est basé sur les alignements par paires de séquence (alignement deux à deux) et multiple en comparant des segments entiers de séquences au lieu d'une traditionnelle comparaison de chaque résidu.

Des alignements par paires sont construits de paires segments de même longueur sans insertion ou délétion de gaps. Ces paires de segments s'appellent les 'diagonales' ou (motif) observable sur le graphe d'un DOTPLOT. Par conséquent DIALIGN n'emploie aucune pénalité de gap. Une fois une diagonale est considérée dans un alignement, elle est fixe et ne peut pas être enlevée à une étape postérieure de l'algorithme. Une diagonale n'est pas choisie selon son poids, mais plutôt selon si le motif décrit par cette diagonale, apparaît dans plus de deux séquences, alors il est préféré aux motifs qui apparaissent dans seulement deux séquences. Cette approche est particulièrement efficace et convenable pour la détection d'une homologie

locale. Sa consommation en termes de durée de calcul et en espace mémoire est considérée raisonnable [Lambert, 03].
Dialign-t [Subramanian et autres, 05] est une version plus récente de Dialign-2, locale et progressive.

2.7 Les Méthodes d'Alignement Progressives

Les algorithmes d'alignement multiple progressifs sont basés sur les informations obtenues d'un arbre guide construit au préalable. Cet arbre définit un certain rapprochement entre les séquences (homologie ou similitude). Puis on construit progressivement l'alignement multiple en respectant l'ordre défini par l'arbre. Vu le nombre de méthodes développées, cette approche paraît la plus utilisée malgré ses inconvénients.

2.7.1 La Méthode ClustalW

ClustalW [J.D. Thompson et autres, 94] est un programme qui met en action les principes de l'alignement progressif tout en essayant d'échapper au piège des erreurs qui peuvent se produire au début de l'alignement et nuire à sa qualité dans la fin.
Dans ClustalW, les auteurs essayent donc de respecter la démarche progressive mais en apportant des modifications et des nouvelles considérations.

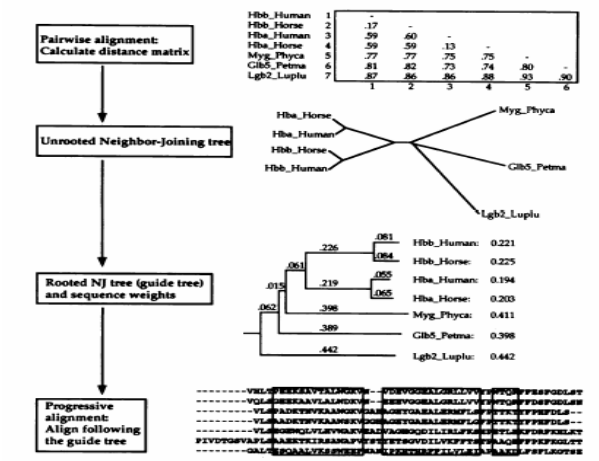


Figure 2.4: Le déroulement de l'algorithme de ClustalW

La *première étape* de ClustalW (Fig.2.4) consiste à aligner les paires de séquences à fin de déterminer la matrice des distances. ClustalW utilise des matrices de substitutions différentes pour la programmation dynamique à des moments différents de l'alignement. Les matrices changent selon la divergence ou la convergence des deux séquences à aligner. L'avantage est que les séquences divergentes sont plus ou moins bien alignées.
Dans la *deuxième étape*, ClustalW utilise la méthode N.J [Saitou et Nei, 87] pour construire un arbre guide et calculer les poids des séquences.
Pendant la *troisième étape* : alignement progressif proprement dit, ClustalW n'affecte pas la même valeur de pénalité d'un gap quelque soit sa position dans la séquence mais essayent de distinguer entre les gaps du début, du milieu et de la fin de la séquence.

Dans ClustalW, il y a une grande étude et des nouvelles propositions sur la manière de faire changer les valeurs affectées à un gap selon sa position dan une séquence ou dans un alignement de séquences.
Une particularité de ClustalW est qu'il possède une interface graphique conviviale contrairement aux autres méthodes

2.7.2 La Méthode T-Coffee

T-Coffee (Tree-based Consistency Objective Function for alignmEnt Evaluation) [Notredame et autres, 00] est une méthode qui essaye de pallier les problèmes de l'alignement progressif. Elle fait tout d'abord un prétraitement des données ; construction d'une bibliothèque [Notredame et autres, 98] qui contient des alignements de paires de séquences fournis à partir de deux types d 'algorithmes d'alignement: global et local produits par deux méthodes connues (*ClustalW* et *Lalign de FASTA*).
En réalité T-Coffee réalise le même alignement progressif que ClustalW mais elle essaye d'échapper aux erreurs commises par ClustalW en utilisant des informations supplémentaires [Reinert, 03].

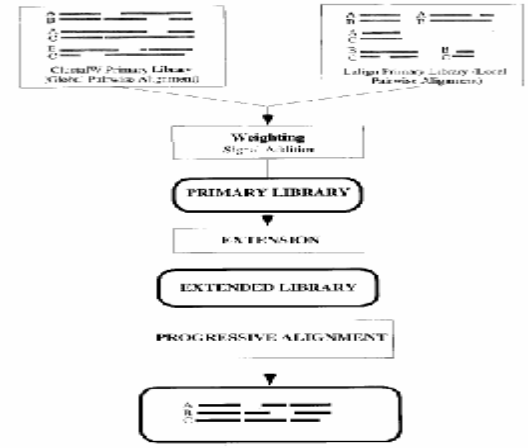


Figure 2.5 : le déroulement de T-Coffee

Les étapes de base de T-Coffee sont:

1. *Produire des bibliothèques primaires des alignements*.

- Une bibliothèque concernant les alignements globaux produits par ClustalW
- Une bibliothèque concernant les alignements locaux produits par Lalign.
- Dans une bibliothèque, chaque alignement est représenté comme une liste de paires de résidus correspondants. Chaque paire de résidus dans la bibliothèque est considérée une contrainte à prendre en considération lors de l'évaluation de l'alignement.

2. *Déduire des poids de la bibliothèque*
Les poids dans chacune des bibliothèques sont calculés avec un pourcentage d'identité, une mesure qui est considérée être un indicateur raisonnable quand les séquences alignées ont plus que 30% d'identité.
3. *Combiner les bibliothèques ensemble dans la bibliothèque primaire*
Tous ces alignements contiennent de l'information qui est plus ou moins fiable. Par conséquent T-Coffee emploie leur combinaison pour confirmer la fiabilité des alignements. Le processus consiste alors à additionner les poids d'une paire de résidus si cette dernière apparaît dans les deux bibliothèques et ne garder qu'une seule entrée dans la bibliothèque finale.

4. *Extension la bibliothèque*.
T-Coffee utilise une stratégie dont le but est de calculer les poids que reflète l'information contenue dans toute la bibliothèque. Pour le faire, on utilise *une approche* de triplet. Ceci fonctionne comme suit :

- Prendre chaque paire de résidus de la bibliothèque et de vérifier l'alignement de ces résidus avec les paires de résidus alignés dans les autres séquences.
- Soit la paire de résidus (Ai, Bj) avec A et B les séquences des résidus correspondants et i,j leur indices respectifs dans les séquences. Soit X_{AB} le poids de l'alignement AB dans la bibliothèque primaire.
- Si Ai est aligné avec Ck et si Ck (C une troisième séquence) est aligné avec Bj avec des poids respectifs Y_{AC} et Z_{CB} alors le poids de (Ai, Bj) sera calculée ainsi:

$$X_{AB} \text{ ß } X_{AB} + \text{Min}(Y_{AC}, Z_{CB}). \qquad (2.13)$$

Ce nouveau poids sera temporairement le poids de Ai et Bj dans la bibliothèque étendue car il risque d'augmenter à chaque fois que l'on examine un nouveau triplet de séquences.

5. *Employez la bibliothèque étendue pour l'alignement progressif*.
Afin de calculer l'alignement progressif nous calculons la matrice de distance en utilisant bibliothèque étendue. Elle est employée pour calculer un arbre guide en utilisant la méthode N.J.
Les gaps présents dans le premier alignement sont fixes et ne peuvent pas être décalées plus tard. En alignant deux groupes de séquences (contenant probablement seulement une séquence) *les scores* moyens de la bibliothèque étendue sont employés pour chaque colonne.
N.B. On remarque aussi que T-Coffee n'utilise pas une fonction objectif proprement dite comme le fait la méthode SAGA.
La complexité est $O(N^3L^2)$ avec N le nombre de séquences et L la longueur de l'alignement.

2.7.3 La Méthode MAFFT

MAFFT [Katoh et autres, 02] est un nouveau programme pour le problème de MSA. Il exploite les caractéristiques physico-chimiques des acides aminés qui composent les protéines pour établir le degré de similitude ou de divergence entre elles.
Une fois les valeurs de ces caractéristiques sont obtenues on applique une transformation de Fourier pour déterminer des relations entre les séquences à aligner afin de pouvoir générer un arbre guide comme toute méthode progressive le fait.

MAFFT a introduit deus nouvelles techniques telles que :
1) les régions homologues sont rapidement identifiées par l'exploitation de la transformation de Fourier (FFT) où dans la quelle chaque acide aminé des séquences est représenté par un vecteur contenant les valeurs de volume et la polarité.
2) une simplification du système de score pour avoir un temps de calcul réduit en faveur d'une recherche de l'exactitude soit pour les séquences de longues insertions et délétions soit pour des séquences divergentes de même longueur.
Deux heuristiques furent développées alors:

- Méthode progressive: (FFT-NS-2)
- Méthode itérative de raffinement (FFT-NS-i).

Le temps de la CPU a été sérieusement réduit par cette méthode en comparant avec les méthodes existantes.

2.7.4 La Méthode PCMA

PCMA (Profile Consistency Multiple Sequence Alignment) [Pei et autres, 03] est programme progressif d'alignement multiple des séquences qui combine deux stratégies d'alignement.
Des séquences fortement semblables sont alignées d'une manière rapide comme dans ClustalW, constituant les groupes pré-alignés. La méthode T-Coffee est appliquée pour aligner les groupes relativement divergents, elle est basée sur la comparaison et la consistance (consistency) profil-profil. La fonction de score pour les groupes pré-alignés est basé sur une nouvelle méthode de comparaison de profil-profil qui est une généralisation de l'approche de PSI-blast [Altschul et autres 97] de la comparaison profil- séquence. PCMA équilibre la rapidité et l'exactitude d'une manière flexible et convient à aligner un grand nombre de séquences.

PCMA est une méthode progressive. Elle s'effectue en deux étapes :
§ *La première étape* : si deux séquences voisines quelconques ou groupes pré-alignés ont une moyenne d'identité par paire de séquences au dessus d'un certain seuil, par exemple 40%, elles sont alignées par l'algorithme de ClustalW pour constituer un nouveau groupe pré-aligné.
À la fin de la première étape, les séquences semblables forment des groupes pré-alignés avec une similitude relativement basse entre groupes voisins.

§ La deuxième étape : une mesure de consistance (Consistency) est appliquée (génération et extension de la bibliothèque) aux groupes pré-alignés, d'une manière semblable comme dans le programme de T-Coffee. Après la mesure de la consistance par l'extension de la bibliothèque, les groupes pré-alignés sont progressivement alignés les uns avec les autres en optimisant une fonction objectif pour former l'alignement final.

La fonction de score utilisée pour évaluer les alignements locaux est basée sur une nouvelle méthode de comparaison profil-profil COMPASS (Comparison Of Multiple Protein Alignments With Assessment of Statistical Significance). Cette fonction construit des alignements profil–profil locaux optimaux et évalue analytiquement les E-values pour les similitudes détectées.

Le système de score et le calcul de *E*-value sont basés sur une généralisation de l'approche de PSI-blast [Altschul et autres, 97] pour la comparaison profil-séquence.

2.7.5 La Méthode MUSCLE

La méthode MUSCLE [Edgar, 04] emploie deux mesures de distance pour une paire de séquences: une distance de k-mer de (pour une paire non alignée) et le Kimura distance (pour une paire alignée). Un k-mer est une subséquence contiguë de longueur k également connu sous le nom de mot ou k-tuplet. Les séquences homogènes possèdent plus de k-mers en commun que prévu par hasard. Cette mesure n'exige pas un alignement, elle donne un avantage significatif de vitesse contrairement à Kimura.

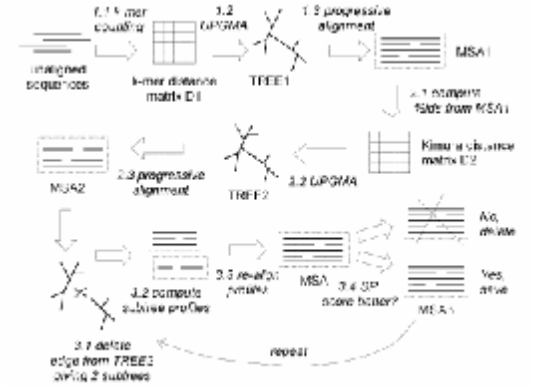


Figure 2.6 : Déroulement de l'algorithme MUSCLE

La méthode MUSCLE peut être décrite en trois étapes essentielles (Figure 2.6) :
L'étape 1 : Le but de la première étape est de produire rapidement un alignement multiple avec plus d'exactitude possible. Ceci est basé sur la détermination d'une matrice D1 de distances à partir de la distance de k-mers entre toutes les paires de séquences.
La matrice obtenue est alors clustérisée par UPGMA, pour produire un arbre binaire TREE1. Un alignement progressif MSA1 est construit alors en suivant l'ordre dicté par l'arbre.

Étape 2 : La source d'erreur principale à l'étape progressive est la mesure approximative de distances k-mer, qui a comme conséquence un arbre sous optimal. MUSCLE re-estime donc l'arbre en utilisant la distance de Kimura, qui est plus précise mais exige l'utilisation un alignement dans ce cas c'est MSA1 donnant ma matrice D2. D2 va subir le même procédé de clustérisation afin de produire un arbre binaire TRRE2 et progressivement construire l'alignement MSA2

Étape 3 : C'est une étape d'amélioration. TREE2 est divisé en deux sous arbres en supprimant la branche qui les relient. Celle ci est choisie en parcourant l'arbre à partir de la racine. Le profil de l'alignement multiple dans chaque sous arbre est alors calculé. Un nouvel alignement multiple est produit en réalignant les deux profils.
Si le score de PS est amélioré, le nouvel alignement est gardé, autrement il est rejeté et l'étape 3 est alors répétée jusqu' à la convergence ou jusqu'à ce que une limite définie soit atteinte.
Considérée la plus rapide et plus exacte, la méthode MUSCLE est la plus répandue actuellement avec ClustalW.

2.7.6 La Méthode MLAGAN

Pour comparer des génomes entiers des espèces différentes, les biologistes ont besoin de plus en plus des méthodes d'alignement qui sont assez efficace pour manipuler de longues séquences, et assez précis pour aligner correctement les caractères biologiques conservés entre les espèces éloignées. LAGAN (Limited Area Global Alignment of Nucleotides), est un système pour l'alignement global rapide de deux séquences génomiques homologues, MLAGAN (Multi-LAGAN) est une version d'alignement multiple progressif à base de LAGAN [Brudno et autres, 03].
MLAGAN est un nouveau système d'alignement multiple qui aligne des séquences génomiques dans deux phases principales:

1 *Un alignement progressif :* Un alignement multiple de K séquences est construit dans K-1 étapes d'alignement deux à deux, où dans chaque étape, deux séquences, ou deux alignements

multiples intermédiaires, sont alignés. MLAGAN emploie LAGAN comme sous-programme d'alignement deux à deux, et présente de nouvelles méthodes pour évaluer un alignement multiple avec une fonction affine des gaps.

2 *Une itération facultative*: phase d'amélioration qui enlève successivement chaque séquence de l'alignement multiple, et la réaligne au reste de l'alignement, jusqu'à ce qu'aucune amélioration significative ne soit observée.

LAGAN utilise la fonction SP pour évaluer les substitutions et la fonction Consensus pour les gaps rencontrés.

2.7.7 La Méthode ProbCons

ProbCons [Do et autres, 05] est une nouvelle méthode et un outil pratique pour l'alignement multiple progressif de séquences protéiques basé sur la probabilité de consistance (Consistency probability). ProbCons réalise statistiquement une amélioration significative par rapport à d'autres méthodes tout en préservant une vitesse pratique.
La probabilité de consistance est une nouvelle fonction de score pour des comparaisons des séquences multiples. ProbCons optimise la fonction basée Consistance (Consistency based) mais construit sur des modèles probabilistes selon les modèles cachés de Markov.

v Le Modèle Caché de Markov (HMM : Hidden Markov Model)

Le modèle caché de Markov est un modèle stochastique composé de grand nombre d'états reliés entre eux, où chaque état émis un symbole observable. Les probabilités d'émission des symboles sont les probabilités d'émission possible de chaque symbole par un état. Cette séquence d'états est cachée et seulement la séquence de symbole émise est observable [Lambert et autres, 03].

Les probabilités de transition d'état sont les probabilités de se déplacer de l'état actuel à un nouvel état en utilisant la distribution stochastique déterminée par l'état de la chaîne cachée de Markov.

Dans le cadre de MSA, le modèle caché de Markov (HMM) fournit une formulation alternative du problème d'alignement de séquences dans lequel, la génération d'un alignement est directement modelée comme un processus de Markov de premier ordre impliquant des émissions et des transitions d'états.
Le chemin le plus probable pour aligner une séquence sur un profil HMM est trouvé par l'algorithme de Viterbi. HMM peut simultanément trouver un alignement et un modèle de probabilité des substitutions, insertions et délétions, qui est le plus consistant. Pour construire un l'alignement multiple, il faut calculer pour chacun séquence un alignement individuel de Viterbi [Lambert et autres, 03].

Soit S une séquence, P est la probabilité d'émission d'une lettre à partir d'un état du modèle :
La Séquence $S = (X_1, X_2, \ldots, X_i, \ldots)$; X_i Î A (alphabet),

- Modèle de Markov de la séquence S:
  - Mémoire d'ordre m :
    $P(X_i = b | X_{i-m} = a1, \; X_{i-m+1} = a2, \ldots, X_{i-1} = a_m)$

- Modèle de Markov d'ordre 1 de la séquence S:

$P(X_i = b | X_{i-1} = a) = \pi(a, b)$.

C'est la probabilité que X_i soit égal à b sachant que X_{i-1} est égal à a.

- Probabilité de la séquence sous ce modèle :

$$P(S) = \pi(X_1) * \prod_{i=2}^{N} \pi(X_{i-1}X_i)$$

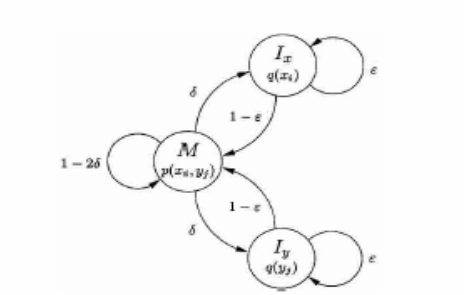


Figure 2.7: Transition entre états dans un modèle HMM

La figure 2.7 montre un HMM de base pour l'alignement de séquence entre deux séquences, *X* et *Y*. L'état *M* émet deux lettres, une de chaque séquence, et correspond aux deux lettres alignées ensemble. L'état *Ix* émet la lettre dans la séquence *X* qui est alignée avec un gap et pareil pour l'état *Iy* émet une lettre dans la séquence *Y* qui est alignée avec un gap. Trouver l'alignement le plus susceptible selon ce modèle en employant l'algorithme de Viterbi correspond à appliquer Needleman-Wunsch avec des paramètres appropriés.
Le logarithme de la probabilité d'émission $p(.,.)$ *à M* correspond à une matrice de score de substitution, tandis que les paramètres de la fonction affine de gaps peuvent être dérivés des probabilités de transition δ et ε .

Fondamentalement, *ProbCons* est un modèle caché de Markov (Pair-HMM) basé sur l'approche progressive d'alignement. Elle diffère principalement de la plupart des approches typiques dans son utilisation de « *l'exactitude maximum attendue »* plutôt que l'alignement de Viterbi, et *de la transformation de la probabilité de consistance* pour incorporer l'information multiple de conservation des séquences pendant l'alignement par paires. ProbCons emploie le HMM représenté sur la figure 2.7 pour indiquer la distribution des probabilités pour tous les alignements entre une paire de séquences. Les probabilités d'émission, ce qui correspond aux scores traditionnels de substitution, sont déduits de la matrice BLOSUM62 [Henikoff et Henikoff, 92]. Les probabilités de transition, qui correspondent aux pénalités de gaps, sont formées avec la maximisation d'espérance (EM).

Ayant m séquences, $S = \{S_1, S_{2,} \ldots\ldots S_m\}$, les étapes de l'algorithme sont :

§ *Étape 1* : Calcul de la matrice des probabilités postérieures :
Pour chaque paire de séquences *x, y* appartenants à S, on calcule la matrice Pxy qui contient les probabilités de toutes les paires de résidus (xi,yj) alignées dans un alignement *a**. *a** est un alignement de x et y généré par la modèle.

§ *Étape 2* : calcul des alignements corrects attendus :
Dans cette étape, on détermine le nombre de paires correctement alignées entre deux séquences divisé par la longueur de la séquence la plus courte.

$$\mathbf{E}_{a^*}(accuracy(a,a^*)|x,y) = \frac{1}{\min\{|x|,|y|\}} \sum_{x_i - y_j \in a} \mathbf{P}(x_i - y_j \in a^*|x,y). \qquad (2.14)$$

Pour chaque pair de séquence x et y, calculer l'alignement *a* qui maximise l'exactitude attendue par la programmation dynamique.

$$E(x, y) = \mathbf{E}_{a^*}(accuracy(a, a^*) \mid x, y). \qquad (2.15)$$

§ *Étape 3* : Transformation de la probabilité de consistance
Réévaluer le score **P**(xi-yj ∈ a* | x,y) en appliquant une transformation de la probabilité de consistance qui incorpore les similitudes de x et y à d'autres séquences de S dans la comparaison de l'alignement de paire de x-y :

$$\mathbf{P}'(x_i - y_j \in a^* | x,y) \leftarrow \frac{1}{|S|} \sum_{z \in S} \sum_{z_k} \mathbf{P}(x_i - z_k \in a^* | x,z) \mathbf{P}(z_k - y_j \in a^* | z,y). \qquad (2.16)$$

§ *Étape 4* : Construction de l'arbre guide
Après la détermination de la matrice des scores à partir de la formule (2.16), la construction de l'arbre guide devient possible par clustérisation. Comme mesure de similitude entre deux séquences, on utilise *E*(x, y) (Equ. 2.15).

§ *Étape 5* : alignement progressif :
L'alignement se fait par groups de séquences selon l'ordre décrit par l'arbre guide. La qualité de ces alignement est mesurée par la fonction SP où les résidus alignés seront affectés d'un score transformé **P'**(xi-yj ∈ a* | x, y) et une pénalité de gap nulle.

2.7.8 La Méthode Align-M

L'alignement multiple de séquences fortement divergentes est un problème pour lequel les programmes disponibles tendent à se montrer faibles vis-à-vis lui. D'une façon générale, ceci est dû au choix de la fonction de score qui ne décrit pas exactement la réalité biologique, ou à l'heuristique utilisée qui ne peut pas explorer l'espace des solutions assez bien. À cet égard, un nouveau programme, Align-M [Van Walle et autres, 04] est présenté. Il emploie une approche locale *non-progressive* pour guider un alignement global.

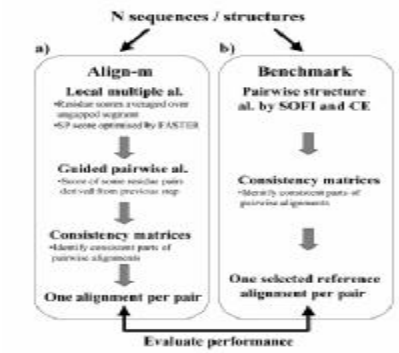


Figure 2.8 : Les étapes de l'algorithme *Align_M*

Le procédé utilisé par Align-M pour aligner des N séquences de comporte 3 étapes principales (Fig. 2.8.a).
Premièrement, un ensemble des alignements multiples locaux (de meilleur score) est déterminé, représentant l'information spécifique au problème d'alignement multiple courant. Un alignement à ce stade peut être une colonne à travers l'ensemble des séquences. Son score *Sc* est calculé par la formule (2.17) déduite de la fonction de score SP mais divisé par le nombre de paires alignées.

Les colonnes avec des scores élevées sont considérées des colonnes bien alignées et elles serviront de repères pour trouver un alignement global meilleur.

$$Sc = \sum_{i=1}^{N-1} \sum_{l=i+1}^{N} S(C(i), C(j)) \Big/ \frac{N(N-1)}{2} \qquad (2.17)$$

Deuxièmement, cette information (les scores des colonnes) est combinée avec le score de la matrice de substitution, afin de produire, pour chaque paire de séquences, un ou plusieurs alignements de paires par la programmation dynamique.
Troisièmement, seulement les parties de ces alignements qui sont considérées suffisamment consistantes vis à vis elles mêmes sont maintenues dans l'alignement final.

2.7.9 La Méthode M-Coffee

M-Coffee [Wallace et autres, 06], est une *méta-méthode* qui permet de combiner plusieurs méthodes d'alignements multiples de séquences (MSA) et rendre un seul MSA. M-Coffee est une simple prolongation de T-Coffee et utilise la notion de consistance (Consistency) pour évaluer un alignement consensus. Cette procédure est robuste par rapport aux variations des choix des méthodes qui participent à la construction des MSAs. M-Coffee travaille sur une collection de quinze MSAs pré-alignés.

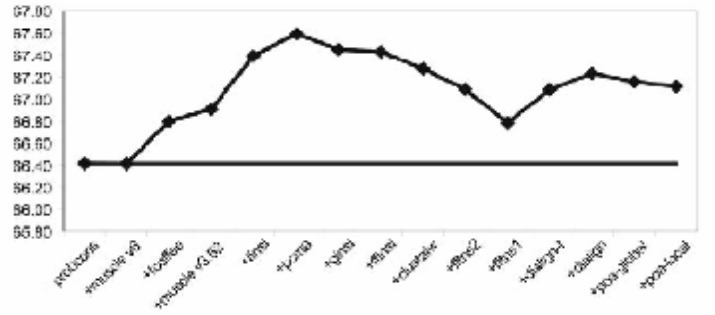


Figure 2.9 : l'évolution du score par insertion de méthodes

La figure 2.9 montre l'évolution de score à chaque fois qu'une nouvelle méthode est ajoutée au package.

§ La méthode M-Coffee montre en réalité une certaine complémentarité entre les méthodes développées jusqu'ici. Rassemblées sous une méta-méthode, les méthodes utilisées arrivent à aligner des séquences avec un degré de qualité qu'aucune ne peut l'atteindre individuellement.

2.8 Étude Comparative des Méthodes

Le nombre des méthodes d'alignement multiple de séquences qui ont été développées reflète l'importance des alignements multiples dans l'analyse de séquence de jour en jour et la variété des buts pour lesquels elles sont nécessaires. Ceci soulève la question quelle méthode faut il employer et quels critères devraient être employés en comparant des méthodes. *ClustalW* [Thompson et autres, 94] est l'une des méthodes les plus répandues et de ceci reflète son interface graphique facile à utiliser. D'autres méthodes sont plus rapides et/ou plus précises *MAFFT* [Katoh et autres, 02] et *MUSCLE* [Edgar, 04]. Quelques packages sont relativement lents mais semblent donner des alignements d'un niveau d'exactitude assez élevé ou acceptent en entrée des données hétérogènes ; *T-Coffee* [Notredame et autres, 00] et *ProbCons* [Do et autres, 05]. Autres méthodes sont spécialisées pour aligner des génomes entiers *MLAGAN*

[Brudno et autres, 03] tandis que d'autres sont spécialisées pour des alignements multiples locaux *DIALIGN* [Morgenstern, 98] et *Align_M* [Walle et autres, 04].
Toutes les méthodes présentées dans la section précédentes ont certainement des avantages et des inconvénients. Leurs défaillances sont dues généralement soit au choix de l'approche elle-même, soit au choix de la fonction objectif à optimiser, ou bien tout simplement à la complexité des données biologiques traitées. Par contre, toutes ces méthodes exposent toujours une nouvelle idée et un nouvel état d'esprit dans la prise en charge de l'analyse biologique. Toutes les facettes de cette analyse ont été prises en charge par l'une ou l'autre méthode.

La table 2.1 propose une tentative de comparaison des méthodes selon les critères suivant : l'approche suivie, nature de l'alignement entre deux séquences, la fonction objectif utilisée, la complexité de l'algorithme et enfin l'originalité de la méthode.

Méthode	***Approche***	***Locale/ Globale***	***Fonction objectif***	***complexité***	***Originalité de la méthode***
ClustalW	Progressive	Globale	SP/ WSP	$O(N^2L^2)$	Adaptation du calcul des scores selon la position des gaps
SAGA	Itérative	Globale	WSP en 96 Coffee en 98	Non définie	Lente mais une bonne utilisation des algorithmes génétiques
T-Coffee	Progressive	Globale	Basée Consistance	$O(N^2L^2)+$ $O(N^3L)+$ $O(N^3)+$ $O(NL^2)$	Utilisation des bibliothèques locales comme matrice de substitution.
MAFFT : FFTNS	Progressive	Globale	SP	$O(NlogN)$ + $O(N^2)$	Utilise une transformée de Fourier pour générer l'arbre guide.
FFTNSi	Prog/Itérative	Globale	SP	idem	C'est FFTNS avec une étape de raffinement itérative
ProbCons	Progressive	Globale	Probabiliste (HMM)/SP	$O(N^3)$	Exploite le concept de HMM.
MUSCLE	Prog/Itérative	Globale	Log-Exp/ SP	$O(N^3L)$	La plus rapide, introduit les distances de k-mer et Kimura
Dialign	Itérative	Locale	Basée Consistance	$O(N^4L^2)$	Comparaison entre segments et non pas entre résidus
PCMA	Progressive	Globale	COMPASS	Non définie	Utilise ClustalW ou T-Coffee selon la nature des séquences. puis raffine les résultats.
Align_M	Non progressive	Locale	SP	$O(N^2L^2)$	Grande capacité à aligner des séquences très divergentes.
MLAGAN	Progressive	Globale	SP/ Consensus	Non définie	Alignement multiple des génomes entiers

Table 2.1 : Tableau récapitulatif des méthodes de MSA

2.9 Benchmarks

En face de ce grand nombre de méthodes de MSA, il est devenu très difficile voire impossible de dire quelle méthode d'alignement est meilleure qu'une autre.
Une évaluation et une comparaison complète des programmes d'alignement exigent un grand nombre d'alignements corrects de référence qui peuvent être employés comme cas de tests.

[McClure et autres, 94] a montré que l'exécution des programmes d'alignement dépend du nombre de séquences, le degré de similitude entre les séquences et le nombre d'insertions dans un alignement. D'autres facteurs peuvent également affecter la qualité d'alignement telle que la longueur des séquences, l'existence de grandes insertions et de prolongements de N/C-terminal.
Pour évaluer la qualité d'un alignement, une méthode est répandue actuellement, est l'utilisation des bases de références.

Ces bases utilisent un grand nombre d'alignements de référence dits « vrais » comme test. Cette méthode consiste à déterminer la capacité d'un programme d'alignement face à des familles de séquences dont l'alignement optimal est connu. Plusieurs bases d'alignements de référence ont été élaborées comme BaliBase [Thompson, et autres, 99], et OxBench [Raghava et autres, 03] Prefab[Edgar, 04], SABmark [Van Walle et autres, 05].

- **La Base de référence ' BaliBase'**

BAliBASE [Thompson, et autres, 99] se compose de 142 alignements de référence, contenant plus de 1000 séquences avec 200.000 résidus. Les alignements sont divisés en cinq catégories hiérarchiques de référence. Chacune des catégories peut être encore subdivisée en plus petits groupes, selon la longueur de séquence et les pourcentages de similitude :
La référence 1 : contient des alignements de (moins de 6) séquences équidistantes, dont le pourcentage d'identité entre deux séquences est dans une marge indiquée. Toutes les séquences sont de longueur semblable, pas de grandes insertions ou prolongements de gaps.
La référence 2 : aligne jusqu'à trois séquences orphelines (moins de 25% d'identité) de la référence 1 avec une famille d'au moins de 15 séquences très proches.
La référence 3 : se compose de familles de séquences équidistantes divergentes (un pourcentage d'identité <25%).
La référence 4 : est divisée en deux sous-catégories contenant des alignements de jusqu'à 20 séquences comprenant des prolongements de N/C-terminal (jusqu'à 400 résidus),
La référence 5 : contient des séquences de longues insertions internes (jusqu'à 100 résidus).
Dans BAliBASE des blocs « *noyau »* (cores) sont annotés pour les alignements qui incluent des régions qui doivent être correctement alignées. Les blocs excluent des régions où il y a une possibilité d'ambiguïté. Ceci peut être un facteur important affectant la signification des comparaisons statistiques des programmes d'alignement.

Pour évaluer la qualité d'un alignement, BaliBase offre un programme en C, Bali-Score qui permet d'évaluer le résultat d'un programme en comparant l'alignement fourni avec le jeu de test correspondant. En résultat, Bali-Score édite deux valeurs qui déterminent la qualité de l'alignement qui sont la valeur SPS (Sum of Pairs Score) qui indique le nombre de paires de résidus correctement alignées et CS (Columns Score) qui test l'habilité du programmes à aligner toute la séquence et qui indique le nombre de colonnes correctement alignées.
Actuellement, BaliBase est la base de référence la plus utilisée pour les tests protéiques. D'un autre côté, il existe une base dédiée aux alignements des séquences nucléiques et qui est *BraliBase* [Gardner, 05].

2.10 Conclusion

Dans ce chapitre ont été introduites les notions de base d'un alignement multiple de séquences ainsi que les différentes mesures de score utilisées pour évaluer celui ci, suivies par les principales méthodes d'alignement multiple de séquences publiées et utilisées. Une étude comparative a été présentée en vue de mettre en valeur les différences et les ressemblances entre ces méthodes. Le nombre des méthodes de MSA risquent d'augmenter dans le futur car le problème est considéré non totalement résolu. D'autant plus qu'aucune des méthodes publiées

n’est totalement efficace et il n’y a pas une méthode à la quelle on peut attribuer la qualité d’être la *meilleure.*

L’objectif de ce mémoire est d’introduire une nouvelle approche dans le contexte des MSA, c’est une approche basée sur l’optimisation multi-objectif. D’après l’étude comparative, il apparaît que les méthodes décrites utilisent une seule fonction objectif à la fois. L’idée sous jacente à notre travail est d’étudier la possibilité de considérer plusieurs mesures à la fois.

A cet effet, le chapitre suivant (3) sera totalement consacré à l’introduction des notions de base de l’optimisation multi-objectif.

Chapitre 3 : L'Optimisation Multi-Objectif

3.1 Introduction

La vie réelle foisonne de problèmes qui cherchent une solution. La majorité de ces problèmes ont des solutions qui ne sont pas forcément convenables selon un ou plusieurs critères bien définis. La plupart des problèmes d'optimisation réels sont décrits à l'aide de plusieurs objectifs ou critères souvent contradictoires et parfois complémentaires qui doivent être optimisés simultanément. Pour les problèmes n'incluant qu'un seul objectif, l'optimum cherché est clairement défini, celui-ci reste à formaliser pour les problèmes d'optimisation multi-objectif.

Prenons le cas d'une personne souhaitant acheter une maison. La maison idéale est celle qui est peu chère avec beaucoup d'espace et bien située si possible, mais cette maison idyllique n'existe pas. Notre acheteur va donc devoir identifier les meilleurs compromis possibles correspondants à son budget.
Plusieurs autres problèmes peuvent être décrits de la même manière tels que l'établissement d'un emploi du temps scolaire ; c'est un problème multi-objectif de nature car il faut en même temps optimiser plusieurs objectifs tels que : le volume horaire à enseigner, l'occupation des locaux, la charge horaire par enseignant, le nombre de matière … etc.

Traditionnellement, Les problèmes multi-objectif ont été abordés comme problèmes d'optimisation mono-objectif après la combinaison de plusieurs critères dans une simple valeur scalaire. De l'autre côté et pendant les dernières années, il y a eu l'apparition d'un certain nombre de métaheuristiques multi-objectif dont le but est d'obtenir un ensemble de solutions de compromis pour des problèmes d'optimisation multi-objectif dans une seule exécution et sans besoin de convertir le problème en mono-objectif au risque que celui-ci perd sa signification. La plupart de ces techniques ont réalisé un grand succès dans l'optimisation des problèmes réels multi-objectif.

Un problème d'optimisation *combinatoire* est défini par un ensemble fini de solutions discrètes et deux fonctions objectif ou plus associant à chaque solution une valeur (la plupart du temps, une valeur réelle). Ainsi, un problème d'optimisation combinatoire consiste en l'optimisation (minimisation ou maximisation) de certains critères sous différentes contraintes permettant de délimiter l'ensemble des solutions réalisables (ou solutions admissibles).

L'optimisation combinatoire multi-objectif regroupe une large classe de problèmes ayant des applications dans de nombreux domaines applicatifs.

Problèmes académiques :

- Ordonnancement : exemple l'emploi du temps scolaire.
- Cheminement, Arbre recouvrant.
- Voyageur de commerce, problème d'Affectation.
- Routage de véhicules.
- Sac à dos multi-critères.

Autres Problèmes réels Multi-Objectif **:**

- Télécommunications : Design d'antennes, affectation de fréquences...
- Aéronautique : ailes d'avions, moteurs...
- Environnement : gestion de la qualité de l'air, distribution de l'eau, …
- Transport : tracé autoroutier, gestion de containers …
- Ordonnancement : Établissement d'un emploi du temps
- Finances, Robotique, ...

Résoudre un problème d'optimisation combinatoire multi-objectif nécessite l'étude de trois points particuliers :

- La définition de l'ensemble des solutions réalisables,
- L'expression des objectifs à optimiser,
- Le choix de la méthode d'optimisation à utiliser.

Les deux premiers points relèvent de la modélisation du problème, le troisième de sa résolution [Barichard, 03].

3.1.1 Le Choix de la Méthode d'Aide à la Décision

L'un des objectifs de l'optimisation multi-objectif est de modéliser les choix du décideur (*homme*) ou plutôt ses préférences et donc définir l'ensemble des solutions réalisables et acceptables. La première décision qui doit être prise en traitant un problème d'optimisation multi-objectif est donc sur la façon dont on combine la recherche et les processus décisionnels. Ceci peut être fait dans une de trois manières suivantes [Collete et Siarry, 02] :

a. ***La prise de décision et puis la recherche*** **:**

Ceci est également connu comme *l'approche a priori* parce que les préférences pour chacun des objectifs doivent être définies par le décideur et puis on lance la recherche des diverses solutions satisfaisant ces préférences. Cette approche nécessite une bonne connaissance a priori du problème.

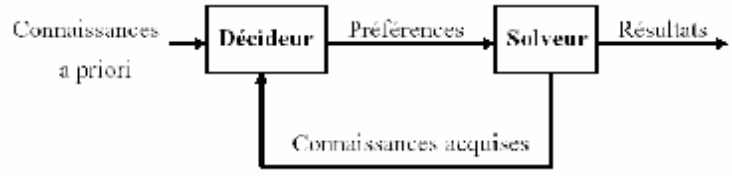


Figure 3.1 :

b. La Recherche et puis la prise de décision **:**

Ceci est également connu comme l'approche a *posteriori* parce que les diverses solutions vont être tout d'abord trouvées puis alors le décideur choisit la plus appropriée. Toutes les solutions sont présentées alors au décideur si elles représentent un compromis entre les divers objectifs.

***c. Recherche et prise de décision interactive* (*progressive*):**

Dans cette approche le décideur intervient pendant la recherche pour la guider vers les solutions prometteuses en ajustant les préférences dans le processus.

Ce chapitre présente une introduction aux concepts fondamentaux de l'optimisation combinatoire multi-objectif, la modélisation d'un problème, la dominance de Pareto et surface de compromis, puis donne une vue d'ensemble sur un certain nombre de métaheuristiques multi-objectif récemment développées.

3.2 Notions de base de l'Optimisation Multi-Objectif

L'optimisation multi-objectif (MultiObjective Optimization: MOO) ou multicritère est un domaine dont l'importance est justifiée par la nature multi-objectif des problèmes d'optimisation à résoudre dans la réalité. Elle se propose de traiter les problèmes qui nécessitent la satisfaction de plusieurs critères en même temps. Ces critères sont parfois conflictuels et parfois complémentaires. Un des aspects caractérisant l'optimisation multi-objectif est qu'elle fournit un ensemble de solutions réalisant le meilleur compromis entre les critères considérés. Cet ensemble de solution est appelé dans la littérature l'ensemble optimal de Pareto.

En général un problème d'optimisation multi-objectif est caractérisé par un espace de recherche de solutions, de deux ou plusieurs fonctions objectif et un ensemble de contraintes.
L'espace de recherche est défini comme un espace d'état et il constitue l'ensemble des domaines de définitions des variables du problème. Cet espace est généralement un espace fini.
Les variables peuvent être réelles, entières ou booléennes. Les problèmes qui utilisent des variables de type différents sont assez complexes à résoudre.
La fonction objectif représente le but à atteindre. Elle permet de définir un espace de solutions potentielles au problème.
L'ensemble des contraintes définit des conditions sur l'espace de recherche que les variables doivent satisfaire.
Une méthode d'optimisation est alors la recherche d'un point ou l'ensemble des points dans l'espace de recherche qui minimisent (ou maximisent) les fonctions objectif sous l'ensemble des contraintes.

3.2.1 Formulation Mathématique

Dans ce qui suit, quelques définitions nécessaires à la compréhension de certains concepts de base que nous utiliserons dans la suite.
On considère un problème multi-objectif ayant n paramètres (variables de décisions) et k objectifs à optimiser (critères) [Collette et Siarry, 02], [Barichard, 03]:

Définition 3.1 :
Vecteur de décision : C'est le nom donné au vecteur $\vec{x} = (x_1, x_2, x_3 \ldots x_n)$. Il correspond à l'ensemble des variables du problème.

Définition 3.2 :
Vecteur objectif ou vecteur de critères $\vec{f}(\vec{x}) = f_1(\vec{x}), f_2(\vec{x}), \ldots\ldots f_k(\vec{x})$. $\vec{f}$ regroupe les k fonctions objectif à optimiser avec bien sûr $k \geq 2$.

Une optimisation multi-objectif revient alors à :

$$\begin{array}{ll} \textit{Minimiser} & \vec{f}(\vec{x}) \quad \textit{(fonctions objectif à optimiser).} \\ \textit{avec} & \vec{g}(\vec{x}) \le 0 \quad \textit{(m contraintes d'inégalité).} \\ \textit{et} & \vec{h}(\vec{x}) = 0 \quad \textit{(p contraintes d'égalité).} \end{array} \qquad (3.1)$$

On a $\vec{x} \in \Re^n$, $\vec{f}(\vec{x}) \in \Re^k$, $\vec{g}(\vec{x}) \in \Re^m$ *et* $\vec{h}(\vec{x}) \in \Re^p$. $\vec{x}$ est appelé vecteur de variables de décision.
$\Re^n$ est considéré l'ensemble des solutions réalisables ou faisables

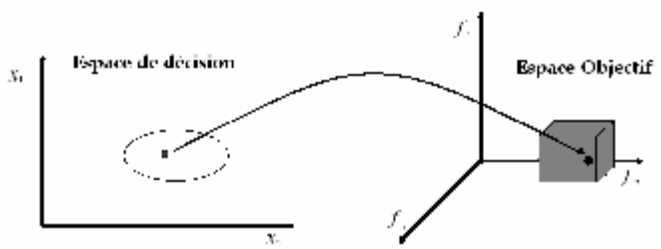


Figure 3.2: L'espace de décision et l'espace objectif (trois fonctions objectif)

En effet, pour un problème à deux objectifs (bi-objectif), la solution optimale cherchée est un ensemble de points correspondant aux meilleurs compromis possibles pour résoudre un problème.

Un problème de maximisation peut être aisément transformé en problème de minimisation (et vice versa) en considérant l'équivalence suivante :

$$\text{maximiser } \vec{f}(\vec{x}) \quad \textbf{ó} \quad \text{minimiser } (-\vec{f}(\vec{x})). \qquad (3.2)$$

3.2.2 La Convexité d'un Espace de Recherche

Un problème MOO admet un ensemble de solutions réalisables *S*. *S* est dit convexe (figure 3.3) si et seulement si : étant donnés deux point A et B quelconques de cette ensemble *S*, l'ensemble des points reliant A à B appartient à *S* autrement :

$$\forall A \textit{ et } B \in S, \ \textit{segment}(A,B) \subset S.$$

La convexité est le premier indicateur de la difficulté du problème. En effet, certaines méthodes MOO trouvent une difficulté de résoudre des problèmes non convexes de manière optimale [Barichard, 03].

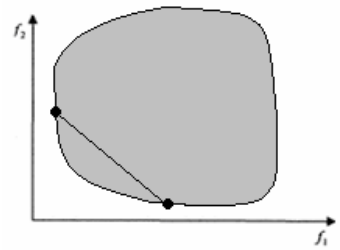


Figure 3.3 : Un espace de recherche Convexe

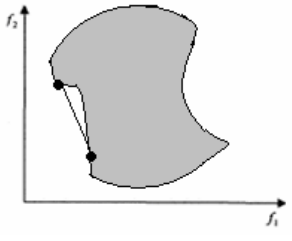


Figure 3.4 : Un espace non convexe

3.2.3 La Dominance au sens Pareto

Contrairement à l'optimisation mono-objectif, la solution d'un problème multi-objectif n'est pas unique, mais est *un ensemble de solutions non dominées*, connu comme l'ensemble des solutions *Pareto Optimales* (P.O).
Le concept de *Dominance* fut introduit pour la première fois en 1896 par Vilfredo Pareto dans le domaine de l'économie.

Définition 3.3:
Dans le cas d'un problème de minimisation, on dit qu'un vecteur de décision $\vec{x} \in \Re^n$ domine vecteur de décision $\vec{y} \in \Re^n$ (on note : $(\vec{x}\ \mathbf{p}\ \vec{y})$ si et seulement si :

$$\forall\, i \in \{1,\ldots,k\} : fi(\vec{x}) \leq fi(\vec{y}) \quad \wedge\ \exists\, j \in \{1,\ldots,k\} \text{ tel que } fj(\vec{x}) < fj(\vec{y}) \tag{3.3}$$

Définition 3.4 :

- P est dit ensemble optimal de Pareto si et seulement si :
- P est un sous ensemble de $\Re^n$: P Í $\Re^n$
  " $\vec{x} \in P$, il n'y a pas un vecteur $\vec{y}$ Î $\Re^n$ tel que $\vec{x}$ *est* dominé par $\vec{y}$

Ainsi, toute solution de l'ensemble Pareto peut être considérée comme optimale puisque aucune amélioration ne peut être faite sur un objectif sans dégrader la valeur relative à un autre objectif.

Définition 3.5 :
Un point A 'domine faiblement' un point B si et seulement si :

$$\forall\, i \in \{1,....k\}\ \ f_i(A) \leq f_i(B) \tag{3.4}$$

L'ensemble correspondant à l'ensemble optimal de Pareto dans l'espace des objectifs est appelé l'ensemble des solutions non dominées ou le *Front Pareto* (Figures 3.5 et 3.6) (ou encore la frontière de Pareto).

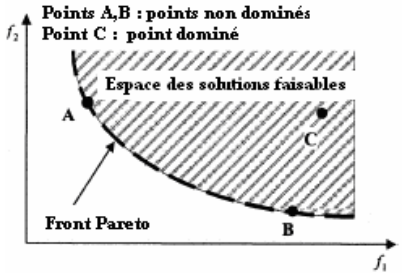


Figure 3.5: Cas de Minimisation bi-objectif

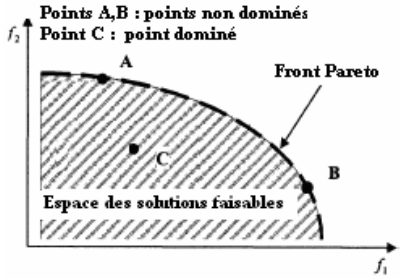


Figure 3.6 : Cas de Maximisation bi-objectif

3.2.4 Propriétés de la Relation de Dominance

La relation binaire de dominance, telle qu'elle est définie ci- dessous :

- Elle n'est pas réflexive, car une $(\vec{x}\ \mathbf{p}\ \vec{y})$ solution ne se domine pas elle-même.
- Elle n'est pas symétrique car on a jamais $(\vec{x}\ \mathbf{p}\ \vec{y})$ et $(\vec{y}\ \mathbf{p}\ \vec{x})$
- Elle n'est pas antisymétrique du fait de l'existence des solutions Pareto optimales.

- Elle est transitive, car $(\vec{x}\ \mathbf{p}\ \vec{y})$ et $(\vec{y}\ \mathbf{p}\ \vec{z})$ ceci implique que $(\vec{x}\ \mathbf{p}\ \vec{z})$.

La relation de dominance est donc une relation d'ordre partiel strict sur l'espace des décisions.

3.2.5 Les Points particuliers 'Idéal' et 'Nadir'

Le point *Idéal* (figure 3.7) est un point optimal et comme son nom l'indique possède comme valeur pour chaque objectif considéré la valeur optimale. C'est un point non réalisable car si c'était le cas, le problème admettrait alors une seule solution optimale qui optimise tous les objectifs à la fois. Dans ce cas les fonctions objectif sont considérées non contradictoires mais complémentaires. Le point Idéal est utilisé dans certaines approches comme point de référence pour orienter la recherche vers le front de Pareto

Le point *Nadir* est un point qui possède comme coordonnées, les pires valeurs possibles pour les objectifs considérés dans l'espace des solutions réalisables. Ce point peut servir pour délimiter l'espace de recherches dans certaines méthodes de MOO.

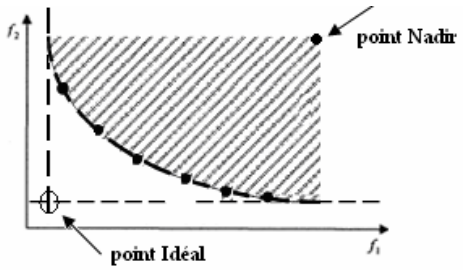


Figure 3.7 : Point Idéal et point Nadir pour un problème bi-objectif

3.2.6 Front Pareto Optimal

Le front Pareto est aussi appelé l'ensemble des solutions efficaces ou la *surface de compromis*.
Quand un ensemble de solutions de compromis (non dominées) est visé (recherche et puis prise de décision), ces solutions représentent une bonne approximation du front Pareto optimal. Le front de Pareto optimal est donc l'ensemble de toutes les solutions non dominées dans l'espace multi-objectif. Cependant, quand les solutions dans l'ensemble obtenu ne se trouvent pas sur le front optimal de Pareto, on devrait se rapporter à cet ensemble comme le front non dominé obtenu ou le front Pareto connu.
Puisque dans l'optimisation basée Pareto les résultats finals doivent être un ensemble de solutions non dominées, d'autres aspects importants sont à considérer pour évaluer la qualité du front non dominé obtenu. Parmi ces aspects il y a:

- Le nombre de solutions non dominées obtenu.
- La proximité entre le front obtenu (connu) et le front Pareto optimal (si connu) (Figure3.8).
- La distribution des solutions non dominées sur le front. (figure3.10)

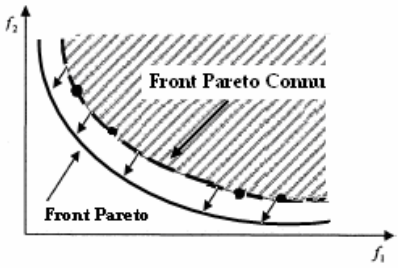


Figure 3.8 : le front Pareto optimal et le front Pareto obtenu

3.2.7 La Structure du Front Pareto

Plusieurs méthodes pour évaluer la qualité du front Pareto obtenu ont été proposées [Knowles et Corne, 02]. En fait, quand les chercheurs parlent de la qualité de l'ensemble des solutions non dominées obtenu, des informations sont très rarement fournies concernant la diversité des solutions dans l'espace de décision. Ceci est extrêmement important parce que bien que les solutions non dominées obtenues puissent être bonnes et distribués sur le front dans l'espace objectif, il se peut que l'une ou l'autre soient également structurellement différentes ou très semblable. Le décideur peut exiger avoir des solutions :

- **Semblables en structure et en valeurs objectives.** Par exemple des solutions qui sont très semblables et bien que chacune d'elles est non dominée, peut-être les décideurs sont intéressés par uniquement une certaine partie de la surface de compromis.

- **Semblables en structure mais très différentes en valeurs objectives.** Comme précédemment, les solutions sont très semblables mais les décideurs veulent des solutions de toute la surface de compromis.

- **Différentes en structure et en valeurs objectives.** Comme dans le cas précédent, des solutions de toute la surface de compromis sont exigées mais ces solutions doivent être très différentes structurellement.

- **Différentes en structure mais semblables en valeurs objectives.** Dans ce cas-ci, peut-être les décideurs exigent des solutions de certaine qualité semblable en ce qui concerne le compromis des objectifs mais veulent voir des solutions qui sont réellement très différentes.

En conclusion, il est à remarquer que deux aspects importants sont à prendre en considération. Tout d'abord, la méthode de résolution doit converger le plus possible vers la frontière de Pareto, de façon à s'en approcher au maximum mais elle doit également proposer des solutions diversifiées et bien distribuées sur le front afin d'avoir un bon échantillon représentatif et ne pas se concentrer sur une zone de l'espace objectif.

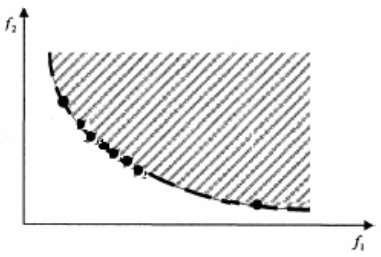


Figure 3.9 : Front Pareto non distribué

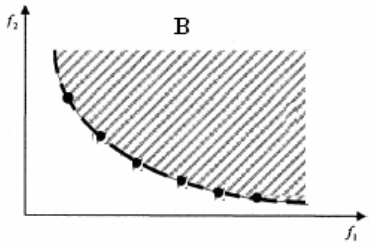


Figure 3.10 : Front Pareto uniforme et distribué

3.3 **Approches de résolution de problème Multi-objectif**

On peut distinguer deux grandes de familles de méthodes de résolution pour traiter un problème multi-objectif : les méthodes exactes et les méthodes approchées.
On remarque tout d'abord que peu de travaux dans la littérature sur les méthodes exactes dans le contexte de la résolution des problèmes d'optimisation multi-objectif. La raison est le fait que la taille de l'espace de recherche se développe exponentiellement à mesure que la taille de problème augmente. Ceci rend inutilisable les méthodes d'optimisation exactes. Des méthodes approchées telles que les heuristiques spécifiques et les métaheuristiques sont souvent employées pour aborder les grands exemples de ces problèmes pour obtenir les solutions proches de l'optimalité dans une quantité raisonnable de temps de calcul. Dans ce chapitre nous nous intéressons aux méthodes approchées (heuristiques)
Une heuristique est un algorithme de résolution ne fournissant pas nécessairement une solution optimale pour un problème d'optimisation donné. Cependant une bonne heuristique possède plusieurs caractéristiques :

- Elle est de complexité raisonnable (peut être polynomiale mais en tout cas efficace en pratique) ;
- Elle fournit le plus souvent une solution proche de l'optimum;
- La probabilité d'obtenir une solution de mauvaise qualité est faible ;
- Elle est simple à mettre en œuvre.

Ces heuristiques peuvent elles-mêmes être classées selon d'autres considérations (Figure 3.11):

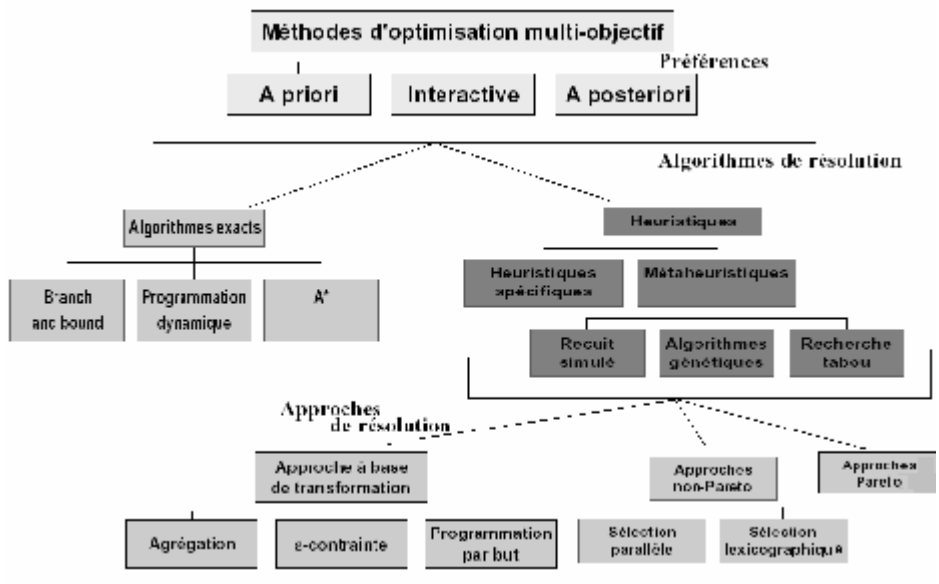


Figure 3.11 : Classification *des approches MOO.*

3.3.1 Les Approches de résolutions à base de transformation

Ces approches transforment le problème initial afin de se ramener à la résolution de un ou plusieurs problèmes mono-objectif. Parmi ces méthodes, on peut citer les méthodes d'agrégation, les méthodes ε-contraintes ou encore les méthodes de programmation par but. En général ces

méthodes nécessitent une connaissance du problème et ne fournissent qu'une seule solution [CoelloCoello et autres, 02]. Elles peuvent alors être classées dans la famille des méthodes d'optimisation a priori.

- **L'agrégation ou Combinaison des objectifs**

C'est une des méthodes classiques pour évaluer l'efficacité des solutions dans l'optimisation multi-objectif. Elle se rapporte à convertir le problème multi-objectif en mono-objectif en combinant les divers critères dans une valeur scalaire simple. La manière la plus commune de faire ceci est en attribuant des poids à chacun des critères puis on effectue une sommation, une multiplication ou toute autre combinaison des opérations arithmétiques.
Ces techniques s'appellent fonction d'agrégation parce que elles combinent (ou " agrègent") tous les objectifs dans une seule fonction.

$$\begin{array}{ll} \textit{Minimiser} & \left\{ \begin{array}{l} f_{\Sigma} = \sum_{i=1}^{k} w_i f_i(\mathbf{r}) \\ \mathbf{r}\mathbf{r} \\ \text{avec} \quad \mathbf{r} \in \Re^n,\ \mathbf{v} \\ \end{array} \right. \\ \textit{avec} & \mathbf{r}\mathbf{r} \\ \textit{et} & \end{array} \qquad (3.5)$$

Sachant que : $\sum_{i=1}^{k} w_i = 1.$

Bien sûr, si l'on veut privilégier un objectif sur d'autres, il suffit de lui affecter un coefficient plus élevé.

- **Les e-contraintes ou Alternation des objectifs**

C'est également une approche qui a été employée pendant beaucoup d'années. Elle se rapporte à optimiser un critère à la fois tout en imposant des contraintes aux autres critères restants. La difficulté ici est sur la façon d'établir le choix sur le critère privilégié par rapport aux autres puisque ceci peut avoir un effet sur le succès de la recherche.

$$\textit{Minimiser} \left\{ \begin{array}{l} f_i(\mathbf{r}) \le e_i, \quad i = 1,...k \\ \mathbf{r} \\ \text{avec } \mathbf{r} \in \Re^n,\ \mathbf{v} \end{array} \right. \qquad (3.6)$$

Pour obtenir des points intéressants et bien distribués sur la surface de compromis, les valeurs de ε_i doivent être choisies judicieusement. Il est clair qu'une bonne connaissance du problème a priori est requise.

- **Le But à atteindre ou programmation par but**

Dans cette approche, on transforme le problème multi-objectif en un problème à un seul objectif où l'on cherche à minimiser l'écart relatif par rapport à un point de référence appelé but, fixé par la méthode ou le décideur. Il existe plusieurs manières de caractériser la distance entre un point de référence (le but) et un autre, notamment à l'aide d'une norme.

3.3.2 Les Approches Non Pareto

Ces approches sont ni Pareto ni agrégatives. Elles transforment le problème d'origine et elles effectuent leur recherche en traitant indépendamment chacun des objectifs et d'une façon équivalente.

- **Les Approches à sélection parallèle**

L'exemple le plus classique est l'algorithme VEGA (Vector Evaluated Genetic Algorithm) [Schaffer, 85] (voir chapitre 4 pour plus de détails). Cette méthode tente d'optimiser en parallèle chaque objectif sur un ensemble de solutions indépendamment des autres objectifs.
Elle produit en réalité des solutions expertes pour un seul objectif. Ce type de méthodes a souvent du mal à trouver les solutions de compromis puisqu'elles se focalisent sur les parties extrêmes du front.

- **Les Approches à sélection lexicographique ou préférences**

Dans cette méthode, les objectifs sont considérés rangés par ordre d'importance par le décideur [Collette et Siarry, 02]. La solution optimale est alors obtenue en minimisant le premier objectif d'abord, le deuxième et ainsi de suite jusqu'au dernier. Ce type de méthode représente les mêmes inconvénients que les précédentes.

3.3.3 Les Approches Pareto

Ces approches utilisent la notion de dominance pour comparer les solutions entre elles. L'un des premiers à discuter de l'intérêt de l'utilisation de la notion de dominance pour la recherche de solutions est [Goldberg, 75]. Ce type de méthodes ne fait subir aucune transformation au problème multi-objectif ni une préférence pour un objectif par rapport à un autre. Les objectifs sont traités de la même façon et les solutions optimales sont celles qui ne sont pas dominées au sens de Pareto. En général, dans ce type d'approche, une seule exécution suffit pour approcher la frontière Pareto. En plus d'après [Fonseca et Fleming, 95] même si la nature contradictoire des critères n'est pas prouvée, les métaheuristiques basée Pareto pourraient trouver la solution idéale et c'est la meilleure pour tous les critères. Dans ce qui suit quelques exemples de métaheuristiques MOO.

3.4 Quelques Métaheuristiques Multi-Objectif

La classe des Métaheuristiques pour l'optimisation multi-objectif sont ceux qui emploient méthodes approchées et utilisent souvent la recherche locale ou l'exploration du voisinage pour conduire la recherche ou comme composant important du processus (approches d'hybride). Plusieurs métaheuristiques multi-objectif employant la recherche locale ont été proposées dans la littérature. Certaines de ces métaheuristiques multi-objectif extraites de la littérature sont brièvement décrites ci-dessous.

3.4.1 Métaheuristiques à Base de la Recherche Taboue

La recherche Taboue (Tabu Search) est introduite par Glover en 1986. Cette méthode a un fonctionnement très proche de celui de la descente. La recherche taboue examine un échantillonnage de solutions de $N(x)$ et retient la meilleure x' même si x' est plus mauvaise que x. Cependant ceci peut entraîner des cycles, par exemple :
Pour un cycle de longueur = 2, on a : x **à** x' **à** x **à** x'….

Pour échapper à cela, on mémorise les *k* dernières solutions dans une liste appelée *Liste Taboue* .cette liste sera utilisée comme une mémoire à court terme qui permet de ne plus prendre ou choisir une solution qui a été déjà visitée.

L'algorithme *Multiobjective Tabu Search* MOTS [Hansen, 97] est une extension basée population de la *Recherche taboue*. Il utilise un ensemble de poids pour guider la recherche vers la frontière de Pareto. Chaque solution maintient sa propre liste taboue et les poids sont ajustés afin de maintenir les solutions éloignées de leurs voisines et essayer donc de couvrir la surface entière du compromis.

Beaucoup d'autres approches de cette classe ont été proposées et étudiées dans la littérature. Par exemple, la variante de *Recherche taboue* de [Baykasoglu et autres, 99] qui manipule une seule solution à la fois mais une liste additionnelle de solutions non dominées trouvées pendant la recherche est conservée à fin de guider la recherche. [Gandibleux et Freville, 00] ont proposé une autre approche de la Recherche taboue qui emploie l'adaptation de poids pour le problème bi-objectif de sac à dos.

3.4.2 Métaheuristiques à Base du Recuit Simulé

Le recuit simulé (Simulated Annealing) s'inspire du processus du recuit physique. Le processus du recuit simulé répète une procédure itérative qui cherche des solutions de coût plus faible tout en acceptant de manière contrôlée des solutions qui dégradent la fonction de coût [Barichard, 03].

- **Pareto Simulated Annealing (PSA)**

PSA [Czyzak et Jaskiewicz, 98] est une extension basée population du recuit simulé proposé pour des problèmes d'optimisation combinatoire multi-objectif. Une population de solutions est maintenue dont le voisinage est exploré de la même manière du recuit simulé classique, mais des poids sont accordés pour chaque objectif dans chaque itération afin d'assurer une tendance de couvrir la surface du compromis. Les poids de chaque solution sont ajustés afin d'augmenter la probabilité de l'éloigner de son voisinage le plus proche d'une manière semblable à celle utilisée dans MOTS [Hansen, 97].

- **Multiobjective Simulated Annealing (MOSA)**

Cette approche [Ulungu et autres, 99] est une autre extension de la méthode du recuit simulé dans laquelle une fonction d'agrégation pondérée est employée pour évaluer la fitness des solutions de diverses régions pour essayer de les rapprocher vers la surface de compromis. L'algorithme manipule seulement une solution à la fois mais maintient une population avec les solutions non dominées trouvées pendant la recherche.

- **Simulated Annealing for Multiobjective Optimisation**

C'est une autre utilisation la méthode de recuit simulé [Suppapitnarm et autres, 00] dans laquelle une température est associée à chaque objectif dans le problème. L'algorithme utilise seulement une solution à la fois et le procédé de recuit ajuste chaque température indépendamment selon les performances de la solution pour chaque critère pendant la recherche. Une archive est employée pour stocker toutes les solutions non dominées visitées.

3.4.3 Les Métaheuristiques à base d' Algorithmes Évolutionnaires Multi-Objectif

Une autre alternative de toutes ces méthodes et qui connaît un succès chez les chercheurs est l'approche évolutionnaire. La littérature foisonne d'articles et de techniques qui mettent en oeuvre des algorithmes évolutionnaires multi-objectif (Multiobjective Evolutionary Algorithm : MOEA).
Pourquoi autant d'intérêts pour les *MOEAs*? Les algorithmes évolutionnaires semblent particulièrement appropriés pour résoudre problèmes d'optimisation multi-objectif, parce qu'ils manipulent un ensemble de solutions faisables simultanément (appelée population). Ceci permet de trouver plusieurs éléments de l'ensemble optimal Pareto en une seule exécution de l'algorithme, au lieu d'effectuer une série d'exécutions séparées comme est le cas dans des techniques traditionnelles de programmation. En plus, les algorithmes évolutionnaires sont moins susceptibles à la forme ou à la continuité du front de Pareto (par exemple, ils peuvent facilement traiter des fronts de Pareto discontinus ou concaves).

Vu l'intérêt particulier que nous accordons à ce type d'approches, un chapitre entier est consacré à l'introduction des approches de conception et de réalisation des algorithmes évolutionnaires multi-objectif développés pendant ces dernières années.

3.5 Mesures de Performances des Métaheuristiques

De nombreux indicateurs de performances ont été proposés dans la littérature [Zitzler et autres, 03].Voici certaines de ces mesures classées en fonction de leur objectif : mesurer la qualité d'un front isolé, comparer deux fronts...

3.5.1 Indicateurs de Qualité s'appliquant à un Seul Front

L'objectif de ces mesures est de fournir une valeur numérique donnant des indications sur la diversité et/ou la distribution des solutions composant le front. Ces mesures sont très utilisées dans la littérature car elles permettent de qualifier un front indépendamment d'autres fronts.
Pourtant elles sont souvent à utiliser avec précaution car ne permettent pas, en général, d'utiliser les valeurs obtenues pour comparer différents fronts. En voici quelques-unes.

- *ONVG* - Overall Non-dominated Vector Generation

Cette mesure comptabilise le nombre de solutions non dominées générées par l'algorithme [Van Valdhuizen, 99]. Cette mesure indépendante, facile à calculer, doit être manipulée avec précaution si elle est utilisée pour comparer des fronts.

- *Schott's spacing metric*

Cette métrique, basée sur un calcul de distance entre les solutions, a pour objectif de mesurer la distribution des solutions le long du front [Knowles et Corne, 02]. Utilisée avec d'autres mesures, elle donne une indication intéressante.

- *Entropie*

L'entropie utilise la notion de niche pour évaluer la distribution des solutions sur le front [Basseur et autres, 02]. Plus proche de 1 est la valeur obtenue, meilleure est la distribution.

3.5.2 Mesures Utilisant une Référence

Ce type de mesure utilise une référence, qui peut être un point ou l'ensemble Pareto optimal (lorsqu'on a la chance de le connaître), pour évaluer la qualité d'un front. En voici quelques exemples.

§ *Métrique S*

Proposée par [Zitzler et autres, 03] cette mesure calcule l'hypervolume de la région multidimensionnelle comprise entre le front et un point de référence. L'inconvénient de cette métrique est que le résultat dépend du point de référence choisi. Ainsi, la difficulté réside dans le choix de ce point qui doit en particulier être dominé par toutes les solutions du front.

§ *Rapport d'erreur*

Ce rapport compare le front obtenu avec le front optimal si il est connu bien sûr [Van Valdhuizen, 99]. Il dénombre les solutions n'appartenant pas au front optimal. Plus le rapport est faible, meilleur est le front.

§ *Distance par rapport au front optimal*

Plusieurs auteurs ont proposé de mesurer la distance entre le front à étudier et le front optimal [Knowles et Corne, 02]. Ils préconisent d'utiliser la distance minimale, maximale, moyenne...

3.5.3 Mesures Comparant deux Fronts Pareto

La comparaison de deux fronts permet de comparer deux méthodes différentes. Lorsque le front optimal est connu cela permet également d'avoir une performance absolue de la méthode sous étude.

§ *Mesure de contribution*

La mesure de contribution entre deux fronts (*Cont*(F1, F2)) permet d'évaluer la proportion de solutions Pareto apportée par chacun des fronts [Basseur et autres, 02]. Lorsqu'un front est totalement dominé sa contribution est nulle et *Cont*(F1, F2) + *Cont*(F2, F1) = 1. Ainsi, une contribution supérieure à 0,5 indique une amélioration du front.

§ *Métrique C*

Cette mesure *C*(F1, F2) indique *le rapport* de solutions du front F2 *faiblement* dominées par les solutions du front F1 [Knowles et Corne, 02]. Lorsque *C*(F1, F2) = 1, le front F2 est totalement dominé par F1.

3.6 Conclusion

Ce chapitre avait pour objectif d'introduire dans un premier temps, les concepts fondamentaux de l'optimisation combinatoire multi-objectif tels que la modélisation d'un problème multi-objectif, l'intervention du décideur dans le processus de décision, la notion de la dominance de Pareto et la structure de la surface de compromis sachant que les méthodes de résolution des problèmes de la MOO sont capables d'offrir un ensemble de solutions dites optimales pour un seul problème en établissant un compromis entre les différents critères. Puis une classification de ces méthodes de résolution a été introduite tout en essayant de présenter plusieurs approches utilisées pour aborder les problèmes de l'optimisation multi-objectif, telles que la transformation du problème multi-objectif en un ou plusieurs problèmes mono-objectif, les méthodes non Pareto et en particulier les méthodes dites Pareto. Dans ce contexte, quelques métaheuristiques ont été exposées particulièrement celles qui utilisent les approches à base de recuit simulé, la recherche taboue et les algorithmes évolutionnaires. À ces derniers, les algorithmes évolutionnaires multi-objectif, une étude plus exhaustive leur a été réservés au niveau du chapitre suivant (chapitre 4).
Enfin, une brève introduction des techniques d'analyse de performances des méthodes multi-objectif a été évoquée.

Chapitre 4 : Algorithmes Évolutionnaires Multi-objectif

4.1 Introduction

La théorie évolutive darwinienne, proposée par Charles Darwin, repose sur deux postulats simples:

- Dans chaque environnement, seul les espèces les mieux adaptées survivent au cours du temps, les autres sont condamnées à disparaître.
- Au sein de chaque espèce, le renouvellement des populations est essentiellement dû aux meilleurs individus de l'espèce.

De cette théorie d'évolution, l'homme acquit la connaissance sur ses propriétés mathématiques et il définit pour l'informatique, les algorithmes d'optimisation appelés "*Algorithmes évolutionnaires*". Les algorithmes évolutionnaires sont donc des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle, à savoir les croisements, les mutations, la sélection, etc.
L'algorithme évolutionnaire est caractérisé par une population de solutions candidates et un processus de reproduction qui permet combiner les solutions existantes pour produire de nouvelles solutions. Puis la sélection détermine quels individus de la population courante participent dans la nouvelle population. Ce processus est répété plusieurs fois jusqu'à convergence vers des solutions optimales.

On peut distinguer trois grandes classes d'algorithmes évolutionnaires : *les algorithmes génétiques* [Holland, 75 ; Goldberg, 89], les *stratégies d'évolution* ou évolutionnistes [Schwefel, 81] et la programmation génétique [Fogel, 00]. Ces méthodes se différencient par leur manière de représenter l'information et par leur façon de faire évoluer la population d'une génération à l'autre.

Les Algorithmes Génétiques

Les algorithmes génétiques (Genetic Algorithms : GAs) sont probablement les algorithmes les plus connus et utilisés dans le calcul évolutionnaire. L'idée de ce système était d'étudier, dans le cadre de la psychologie/biologie, le processus d'adaptation des populations à la base des données sensorielles introduites au système grâce à des détecteurs binaires.
Dans un GA, chaque individu représente un point de l'espace d'état auquel on associe un vecteur de valeur de critères à optimiser. On génère ensuite une population d'individus aléatoirement pour laquelle l'algorithme génétique s'attache à sélectionner les meilleurs individus tout en assurant une exploration efficace de l'espace d'état. Les GAs diffèrent des algorithmes classiques d'optimisation et de recherche essentiellement en quatre points fondamentaux [Deb, 99]:

- Les GAs utilisent un codage des éléments de l'espace de recherche et non pas les éléments eux-mêmes.
- Les GAs recherchent une solution à partir d'une *population* de points et non pas à partir d'un seul point.

- Les GAs n'imposent aucune régularité sur la fonction étudiée (continuité, dérivabilité, convexité…). C'est un des gros atouts des algorithmes évolutionnaires en général.
- Les GAs ne sont pas déterministes, ils utilisent des règles de transition stochastiques.

La programmation génétique

La programmation génétique (PG) est un paradigme permettant la programmation automatique d'ordinateurs par des heuristiques basées sur les mêmes principes d'évolution que les AGs [Fogel, 00]. La différence entre la PG et les AG réside essentiellement dans la représentation des individus. En effet, la PG consiste à faire évoluer des individus dont la structure est similaire à des programmes informatiques. La PG représente les individus sous forme d'arbres, (des graphes orientés et sans cycle), dans lesquels chaque nœud est associé à une opération élémentaire relative au domaine du problème. Plusieurs autres représentations comme des programmes linéaires et des graphes cycliques, ont été utilisées depuis. La PG est particulièrement adaptée à l'évolution de structures complexes de dimensions variables.

Les stratégies d'évolution

La stratégie d'évolution (SE) a été présentée pour la première fois par Rechenberg et Schwefel en 1960. Les SEs représentent les individus comme un ensemble de caractéristiques de la solution potentielle. En général, cet ensemble prend la forme d'un vecteur de nombres réels de dimension fixe. Les SEs s'appliquent à une population de parents à partir de laquelle des individus sont sélectionnés *aléatoirement* afin de générer une population de descendants qui vont ensuite être modifiés par des *mutations*. Les paramètres de la stratégie d'évolution (tels que la taille de la population, le taux de mutation …etc.), évoluent eux aussi dans le temps selon les mêmes principes que les paramètres caractérisant les individus [Schwefel, 81]. Il y a quelques différences entre un GA et la stratégie d'évolution :

- La stratégie d'évolution a été conçue comme un optimiseur d'une fonction mais les algorithmes génétiques ont été à l'origine développés pour démontrer les avantages du croisement dans une évolution simulée.
- La reproduction dans les algorithmes génétiques est *proportionnelle au fitness* mais pas dans la stratégie d'évolution (le choix est aléatoire).
- L'algorithme génétique fait une distinction entre *le génotype* et *le phénotype* d'un individu tandis que dans la stratégie d'évolution originale toutes les deux coïncident.
- Dans la stratégie d'évolution, *les parents et leurs enfants* peuvent concurrencer pour survivre dans la prochaine génération mais pas dans l'algorithme génétique original.
- *La mutation* est la force principale pour conduire une stratégie d'évolution tandis que c'est le *croisement* pour l'algorithme génétique. Les deux approches se sont rapprochées et les systèmes hybrides utilisent la mutation et le croisement.

Dans le cadre de l'optimisation multi-objectif (MOO) et en 1989, D.E. Goldberg [Goldberg, 89] a écrit un ouvrage dont lequel il a décrit l'utilisation des algorithmes évolutionnaires dans le cadre de résolution de problèmes multi-objectif concrets, et sa proposition consistant à utiliser la notion d'optimalité de Pareto dans la sélection, a permis de mieux faire connaître ces derniers dans la communauté scientifique et a marqué le début d'un nouvel intérêt pour cette technique d'optimisation multi-objectif. Un certain nombre algorithmes évolutionnaires multi-objectif (Multiobjective Evolutionary Algorithm : MOEA) ont été proposés ces dernières années et l'intérêt croissant pour ces méthodes a motivé leur extension à l'origine proposés pour l'optimisation mono-objectif.

Les MOEAs sont très bien adaptés au traitement d'un problème d'optimisation multi-objectif. En raison de leur parallélisme inhérent et leurs possibilités à exploiter les similitudes des solutions

par recombinaison, les MOEAs peuvent rapprocher le front Pareto optimal en une seule exécution [Zitzler et autres, 00]. Ce domaine est très dynamique et ne cesse de se développer. Il y a plusieurs versions des MOEAs, chacune essaye d'apporter plus d'efficacité à la MOO et de trouver des solutions les plus proches de l'optimalité.
Lors de l'utilisation d'un MOEA deux points essentiels sont à régler [Zitzler et autres, 00]:

- Comment calculer la valeur du fitness (ou d'adaptation) d'un individu et quelle est la procédure de sélection qu'il faut utiliser à fin de s'approcher au maximum de la frontière de Pareto.
- Comment maintenir la diversité au sein de la population, empêcher une convergence prématurée et par conséquent atteindre une frontière de Pareto distribuée et uniforme.

Dans la littérature, une distinction peut être observée au niveau des différents MOEAs. Elle se situe dans la technique de sélection utilisée. Cette étape qui est très importante, consiste à choisir parmi les solutions qu'elles sont celles qui sont considérées meilleures et peuvent par conséquent participer au processus de reproduction dans la prochaine itération de l'algorithme. La sélection basée *dominance* au sens Pareto a connu plus de succès que les autres techniques. Les MOEAs, comme toutes autres techniques de la MOO, offre à la fin un ensemble de solutions optimales. Pour que cet ensemble soit bien représentatif et diversifié, les récents MOEAs exploitent des techniques des heuristiques de partage et de diversification et qui sont souvent exprimées sous différents aspects tels que le *Sharing (Niching)* [Horn et Nafpliotis, 93], *Clustering* [Zitzler et Thiele, 98] ou *Crowding* [De Jong, 75]. Ces techniques augmentent souvent la complexité des algorithmes mais garantissent en parallèle une meilleure configuration du front Pareto optimal. Les dernières améliorations et suggestions apportées aux MOEAs incitent à l'hybridation de ceux-ci avec d'autres heuristiques de recherche en vue d'une meilleure exploration de l'espace de recherche [Talbi, 00]. Les heuristiques qui utilisent la recherche dans le voisinage ont été utilisées pour donner plusieurs versions de MOEAs hybrides. Une récente classification des MOEAs est utilisée. On parle des MOEAs élitistes et les non élitistes. Vu leur aspect stochastique, les MOEAs peuvent ne pas conserver les bonnes solutions rencontrées pendant leur exécution et les rendre disponibles à la fin de l'exécution de l'algorithme, de ce fait est née la notion d'élitisme qui assure simplement que les meilleurs individus de la population actuelle seront pris dans la prochaine génération. L'élitisme peut être traduit par le fait de stocker les meilleures solutions dans une population secondaire qui servira comme une réserve pour produire de nouvelles solutions pour les générations future Knowles et Corne, 04
Dans ce chapitre, nous allons présenter les caractéristiques générales des MOEAs, les paramètres importants à ajuster, et les différentes techniques utilisées pour les améliorer. En fin, nous allons décrire quelques MOEAs performants de la littérature suivie par une étude comparative.

Supprimé : [

Supprimé :].

4.2 Aspect Général d'un MOEA

Les algorithmes évolutionnaires sont des méthodes stochastiques qui démarrent par une ou plusieurs solutions initiales puis effectuent une série de raffinements à l'aide des opérateurs de modification pour obtenir des solutions meilleures. Les solutions obtenues par les algorithmes évolutionnaires ne sont pas forcément optimales mais elles sont proches de l'optimalité. La différence des algorithmes évolutionnaires multi-objectif (MOEAs) et ceux qui sont mono-objectif est que les MOEAs optimisent plusieurs fonctions objectif à la fois et fournissent en fin d'exécution un ensemble de solutions dit l'ensemble optimal. Les MOEAs ont les caractéristiques suivantes :

- Ils sont à base de population
- Ils permettent l'héritage du patrimoine génétique.
- La politique suivie est 'les mieux *adaptés* survivent'
- Ils effectuent une amélioration globale de l'ensemble des solutions.
- Les Paramètres à ajuster pour un MOEA sont :

- Les fonctions objectif à optimiser (plus de deux fonctions)
- La taille de la population des solutions.
- Les opérateurs génétiques (mutation ou/et croisement).
- La technique de la sélection des parents pour les opérations génétiques.
- Le remplacement de la population.

Un MOEA est un algorithme itératif qui s'exécute en plusieurs étapes importantes : chaque étape a un rôle très important dans l'évolution de l'algorithme et la convergence vers les solutions optimales. Un MOEA peut avoir l'aspect suivant :

Algorithme 4.1 : *MOEA*

t ß 0 ;
A^0 ß ∅ // la population à la génération 0
Tant que terminaison (A^t, t) = faux faire
    t ß t+1 ;
    B^t ß générer (A^{t-1}) // créer des descendants
    B^t ß évaluer (A^{t-1}) // évaluation de la qualité des solutions
    A^t ß sélectionner (A^{t-1}, B^t) ; // mise à jour de la population par des nouvelles
    // solutions
Fin Tant que.
Afficher ((A^t)*) // l'ensemble des solutions optimales finales

La terminologie employée pour décrire les composants d'un MOEA est empruntée à la génétique [Jourdan, 03]:

- Les chromosomes sont les éléments à partir desquels sont élaborés les solutions (individus).
- La population est l'ensemble des chromosomes.
- La reproduction est l'étape de combinaison des chromosomes. La mutation et le croisement génétiques sont des opérations de reproduction.

4.2.3 Le Génotype et le Phénotype

Dans la nature, les êtres vivants se croisent et interagissent les uns avec les autres. Chaque individu est caractérisé par un *génotype* indépendant de l'environnement où il vit. Le *génotype* est le codage de ces traits en gènes. Les opérateurs génétiques fonctionnent au niveau génotypique tandis que le mécanisme de sélection opère au niveau *phénotypique*. Le *phénotype* d'un individu est l'ensemble des traits caractéristiques de celui-ci. D'une manière simple, on peut dire :

Génotype : est la codification d'une solution réalisable.

Phénotype : chaque génotype représente une solution potentielle à un problème d'optimisation. La valeur de cette solution potentielle pour une fonction objectif *f* est appelée le phénotype (figure 4.1).

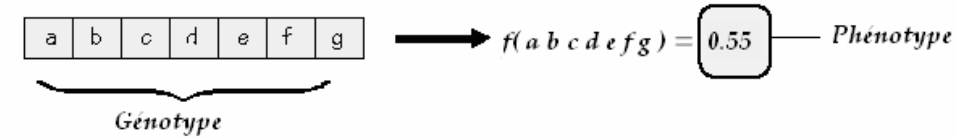


Figure 4.1 : *Génotype et Phénotype d'un individu*

4.2.2 Le Codage

Cette étape associe à chacun des points de l'espace d'état (les solutions) une structure de données. Cette structure conditionne le succès des algorithmes.

Chaque individu (solution) de la population va être **codé** à la manière d'un gène. Par exemple, le plus souvent, on fait correspondre une chaîne binaire à l'individu (cette chaîne binaire est une image de la position de l'individu dans l'espace de recherche). Le codage binaire est surtout spécifique aux GAs.
Dans le cas d'un codage binaire un individu est appelé **chromosome**. Il correspond au codage sous forme de **gènes** (un gène ici est un bit) d'une solution potentielle à un problème d'optimisation.
Un des avantages du codage binaire est que l'on peut facilement coder toutes sortes d'objets : des réels, des entiers, des valeurs booléennes, des chaînes de caractères… Cela nécessite simplement l'usage de fonctions de codage et décodage pour passer d'une représentation à l'autre. Un autre avantage est que ce type de codage permet de créer des opérateurs de croisement et de mutation simples (par inversion de bits par exemple).

.2.1 La Population

En général, Les MOEAs manipulent une population d'individus ou *chromosomes*. La recherche des bonnes solutions s'effectu sur un domaine de possibilités plus important. L'utilisation de la notion de population favorise l'exploration de l'espace de recherche et la convergence vers les solutions optimales. C'est une des grandes forces des MOEAs. Toutefois, certaines stratégies évolutionnistes utilisent une population à un seul individu [Knowles et Corne, 00a].
Un mécanisme de génération de la population initiale est nécessaire pour faire démarrer l'algorithme. Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures. Le choix de la population initiale conditionne fortement la rapidité et l'efficacité de l'algorithme. La façon la plus simple est de générer aléatoirement les individus [Jourdan, 03 ; Barichard, 03].

.2.2 La Génération

Une génération est une population à un instant *t*. Les MOEAs utilisent les opérateurs de *croisement* et/ou de *mutation* pour faire évoluer les populations. Ils produisent ainsi une nouvelle génération d'individus plus adaptés. Chaque génération opère sur une nouvelle population issue des individus de la génération précédente.

.2.3 La Fonction d'Adaptation ou fitness

A chaque individu, on associe une valeur d'**adaptation** (on appelle aussi cette valeur l'**efficacité** ou **fitness**). Cette valeur d'adaptation va correspondre à la performance d'un individu dans la résolution d'un problème donné. Par exemple, si l'on considère un problème de maximisation d'une fonction, la valeur de l'adaptation de l'individu croîtra avec sa capacité à maximiser cette fonction. La définition de la fonction d'adaptation devient plus compliquée dans le cas d'optimisation multi-objectif, car il y a plusieurs fonctions à optimiser et pas une seule. En général, dans de telles situations, la dominance de Pareto est le seul moyen pour bien exprimer la fonction d'adaptation. Elle est la clé de voûte des MOEAs. Il est très important de définir une bonne fonction d'adaptation pour un problème donné, ce qui est certainement la plus grosse difficulté des MOEAs.

.2.4 Le Croisement

En biologie, le terme croisement (Cross-over en anglais) désigne le moment dans l'étape de la fécondation au cours duquel les bagages génétiques du père et de la mère sont échangés. Les chromosomes mâles et femelles s'associent et échangent leurs gènes afin de donner naissance à un individu original mais possédant des caractéristiques provenant des deux parents.

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants (figure 4.3). Bien que ce phénomène ait lieu dans toute reproduction, il n'est utilisé dans les MOEAs qu'avec une probabilité *p* et ce afin de ne pas faire diverger l'algorithme, ou au contraire le faire converger trop prématurément.
Il existe plusieurs types de croisements :

- **Le Croisement à 1-Point**

Le croisement à 1-point dont le fonctionnement est le suivant : on suppose que les individus sont codés sur des chaînes de longueur L.

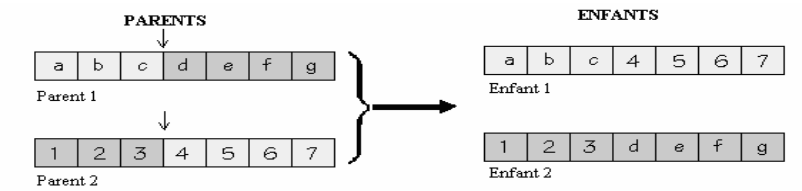


Figure 4.2 : *Le croisement à 1 point entre deux individus*

Le principe est de créer deux nouveaux individus à partir des deux parents, on génère au hasard un entier *a* et le premier enfant va avoir les *a* premiers gènes du parent1 et les *L– a* derniers gènes du parent2, et vice versa pour le deuxième enfant.
Dans cet exemple, l'Enfant$_1$ reçoit les trois premiers gènes du Parent$_1$ et les quatre derniers gènes du Parent$_2$; l'Enfant$_2$ quant à lui, hérite les trois premiers gènes du Parent$_2$ et les quatre derniers gènes du Parent$_1$.
Il existe également le croisement à deux points (figure 4.3) et le croisement multipoint.

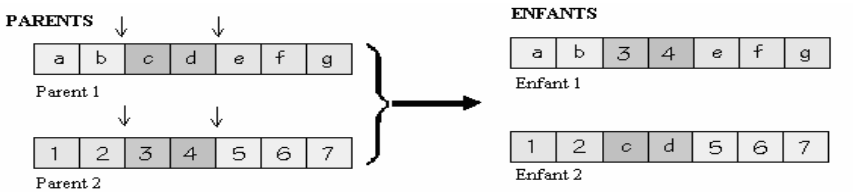


Figure 4.3 : Croisement à deux points

- **Le Croisement Uniforme**

Dans ce type de croisement, on utilise un masque de croisement (*mask*), qui consiste en un vecteur généré aléatoirement, de longueur identique aux chaînes parents, et composé de 0 et 1. Lorsque le bit du masque vaut 0, le premier enfant hérite le bit correspondant du premier parent, sinon il hérite celui du second parent. Le second enfant est le complémentaire du premier. Ce croisement peut être considéré comme une généralisation du croisement multipoint sans connaissance préalable du point de croisement (Figure 4.4).

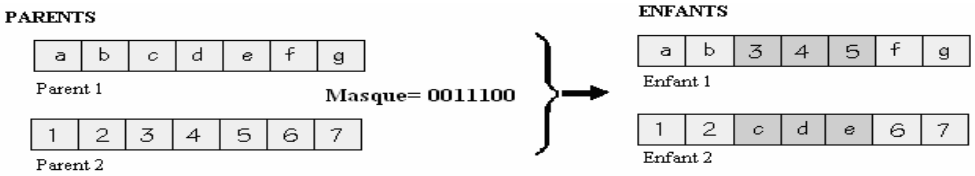


Figure 4.4: *Croisement uniforme*

.2.5 La Mutation

Cet opérateur génétique consiste à modifier le génotype d'un individu. La mutation a pour but de garantir l'exploration de l'espace d'états.
Les propriétés de convergence des MOEAs sont fortement dépendantes de cet opérateur sur le plan théorique, et un algorithme peut converger rien qu'en utilisant des mutations. Pour les problèmes discrets, l'opérateur de mutation consiste généralement à tirer aléatoirement un gène dans le chromosome et à le remplacer par une valeur aléatoire (Figure 4.5)

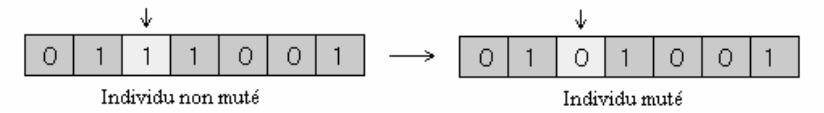


Figure 4.5 : *La mutation d'un gène vers un autre*

De même que pour le croisement, il existe la mutation 1-point qui modifie l'état d'un unique gène et la mutation multipoint qui modifie plusieurs gènes.
Le lieu de la mutation est tiré au hasard, dans l'exemple la figure 4.5 est : Le troisième gène est inversé.
L'intérêt de cet opérateur est de créer de nouveaux individus si la population à tendance à s'uniformiser. Il est appelé avec une faible probabilité p_{mu} sur chaque gène, cependant la détermination de cette valeur est délicate puisque p_{mu} est fonction de la taille N de la population et de la longueur des chaînes codant les individus. Cependant dans la pratique il apparaît préférable que p_{mu} diminue au cours du temps et reste particulièrement faible.

.2.6 La Sélection

Le rôle principal de l'opérateur de sélection dans une méthode d'optimisation est d'imposer une direction pour le processus de recherche. La direction de la recherche devrait être en accord avec les préférences du décideur.
La sélection a donc pour objectif d'identifier les individus qui doivent se reproduire et par conséquent elle permet de décider sur la survie ou la disparition d'un individu. Cet opérateur ne crée pas de nouveaux individus mais identifie les individus sur la base de leur fonction d'adaptation, les individus les mieux adaptés sont sélectionnés alors que les moins bien adaptés sont écartés [Deb, 99]. L'opérateur de sélection doit être capable de favoriser les meilleurs éléments selon le ou les critères à optimiser (minimiser ou maximiser). Ceci permet de donner aux individus dont la valeur est, plus grande une probabilité plus élevée de contribuer à la génération suivante c-à-d participer au croisement ou à la mutation.
Il existe plusieurs méthodes de sélection, les plus connues sont la «roulette biaisée», la « sélection par tournoi » et le « ranking ».

Un des principaux avantages des algorithmes évolutionnaires pour l'optimisation multi-objectif est qu'ils permettent non seulement la mise en oeuvre d'approches non Pareto (agrégation des objectifs, ..), mais aussi l'implémentation d'approches Pareto. En effet, certains mécanismes de sélection implémentent la relation de dominance réalisant ainsi une sélection Pareto [Barichard, 03].

4 Sélection par le Rang ou le Ranking

Cette notion de rang (Ranking), initialement proposée par [Goldberg, 89], propose de trier la population d'un algorithme évolutionnaire de telle façon que toutes les solutions non dominées

soient de meilleur rang. Cette notion de ranking peut ensuite être implémentée de différentes façons.
Une sélection Pareto utilise la relation de dominance pour affecter des rangs aux individus de la population, faisant apparaître la notion de front. Cette technique fut reprise et implémentée dans l'algorithme NSGA2 [Deb et autres, 00b] : initialement, tous les individus non dominés

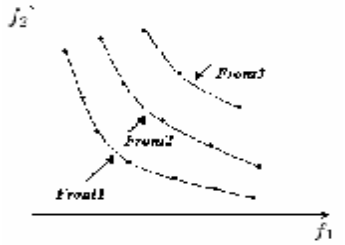


Figure 4.6 : *la Sélection par le Rang selon NSGA2*

de la population reçoivent le rang 1 et sont retirés temporairement de la population (Figure 4.6). Puis, les nouveaux individus non dominés reçoivent le rang 2 avant d'être à leurs tours retirés de la population. Le processus s'itère tant qu'il reste des individus dans la population. La valeur d'adaptation de chaque individu correspond à son rang dans la population. Ainsi, l'évaluation d'un individu ne dépend pas uniquement de lui même, mais aussi de tous les individus de la population [Barichard, 03].
Autres techniques de calcul de rang ont été présentées par [Bentley et Wakefield, 97] telles que:

Méthode 1: Non-Dominated Sorting (NDS): Décrite par [Goldberg, 89]. Les valeurs de fitness par objectif sont traitées indépendamment et non combinées. Les valeurs pour le même objectif dans différentes solutions sont directement comparées. Des solutions sont rangées dans l'ordre de non-dominance, avec la meilleure solution étant la moins dominée par les autres.

Méthode 2: Weighted Maximum Ranking (WMR) : Cette méthode de rang est déduite de l'algorithme VEGA [Schaffer, 85]. WMR construit des listes de valeurs de fitness pour chaque objectif. Les *n meilleures* solutions de chaque liste sont alors extraites, et des paires aléatoires sont choisies pour la reproduction.

Méthode 3: Weighted Average Ranking (WAR) : Pour chaque objectif, les valeurs de fitness des solutions sont extraites dans une liste à part. Ces listes alors sont individuellement triées dans l'ordre croissant ou décroissant. Chaque solution de la population aura alors un rang différent par rapport à l'objectif considéré. Le rang moyen de chaque solution est alors calculé et qui est la moyenne de ses rangs pour tous les objectifs. Cette valeur permet aux solutions d'être triées selon le meilleur rang moyen. Ainsi, plus le rang moyen d'une solution est haut, plus grande est sa chance d'être sélectionnée.

- **Sélection par Tournoi**

Cette technique de sélection s'effectue en deux étapes, tout d'abord on réalise un tirage aléatoire sur l'ensemble de la population des *n* individus qui vont participer au tournoi. *K* individus sont tirés au sort dans la population (*K* est un paramètre appelé taille du tournoi) [Jourdan, 03].
Dans cette première étape tous les individus ont la même chance d'être sélectionnés. Dans la seconde étape, on compare les fitness des *K* individus sélectionnés pour garder le meilleur. Il existe différentes sélections par tournoi : déterministe ou probabiliste. Dans le cas du tournoi déterministe, le meilleur des *K* individus gagne le tournoi. Cette technique de sélection est la plus élitiste, car la probabilité qu'un mauvais individu soit sélectionné est très faible.

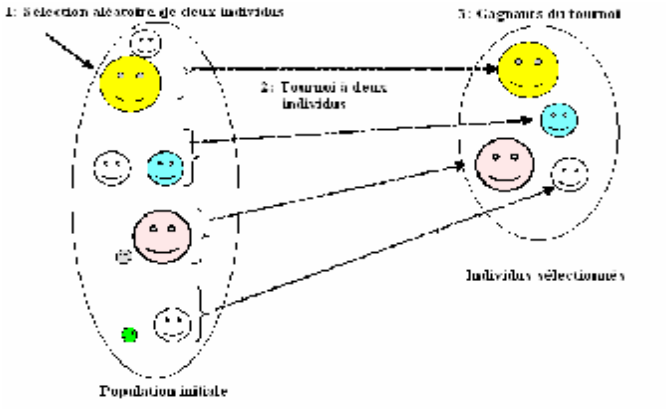


Figure 4.7 : *La sélection par tournoi binaire*

Dans le cas probabiliste, chaque individu peut être choisi comme vainqueur avec une probabilité proportionnelle à sa fitness.
La figure 4.7 montre un exemple de sélection par tournoi entre deux individus (tournoi binaire).

- **Sélection par la Roulette Biaisée**

Le principe de la roulette biaisée " wheel selection" consiste à associer à chaque individu une probabilité de sélection proportionnelle à sa fitness (figure 4.8). On reproduit ici le principe de tirage aléatoire utilisé dans les roulettes de casinos avec une structure linéaire [Jourdan, 03].

$$P_{sélection}(S_i) = \frac{Fitness(S_i)}{\sum_{j=1}^{n} Fitness(S_j)} \qquad (4.1)$$

On tire alors un nombre aléatoire de distribution uniforme entre 0 et 1, puis on " regarde " quel est le segment sélectionné. Avec ce système, les grandes probabilités (Formule 4.1), c-à-d les bons individus, seront plus souvent sélectionnées que les petites.

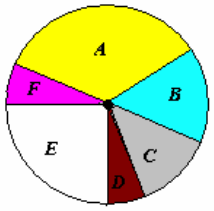


Figure 4.8 : *La roulette Biaisée*

Cependant, cette méthode n'est pas parfaite : lorsque la dimension de la population est réduite. Il est difficile d'obtenir en pratique l'espérance mathématique de sélection en raison du peu de tirages effectués.

4.2.9 Le Critère d'Arrêt

L'arrêt d'un MOEA est une décision très difficile car il n'est pas évident de savoir quand l'optimum est atteint.

Généralement, trois critères sont utilisés par les MOEAs pour décider sur l'arrêt de l'exécution de celui-ci :

- Arrêt après un certain nombre d'itérations ou générations
- Arrêt lorsque il n'y a pas d'améliorations au niveau du fitness des solutions après un certain nombre de générations.
- Arrêt lorsque il y a une perte de la diversité génétique (toutes les solutions sont devenues ou presque identiques).

Malheureusement, les MOEAs de la littérature révèlent rarement ce détail qui est loi d'être insignifiant.

.2 Les Techniques avancées d'amélioration des MOEAs

Les MOEAs ont montré une certaine efficacité dans la résolution des problèmes de la MOO. Mais vu leur aspect stochastique, les MOEAs peuvent ne pas conserver les bonnes solutions rencontrées pendant leur exécution et les rendre disponibles à la fin de l'algorithme, de ce fait est née la notion d'élitisme qui préconisent que les meilleurs individus de la population actuelle seront pris dans la prochaine génération. Une autre difficulté que peut rencontrer un MOEA est qu'un individu ayant une très bonne valeur de fitness, a tendance à se multiplier aux dépens des autres individus de la population et par conséquent le MOEA va se coincer autour d'un seul optimum et fournir à la fin d'exécution un font Pareto mal distribué. D'un autre côté, il y a une récente tendance à faire combiner les MOEAs avec d'autres méthodes de recherche et d'exploration de l'espace des solutions en vue d'obtention de résultats meilleurs que ceux obtenus par les MOEAs seuls. Ces méthodes hybrides ont montré leur efficacité pour trouver des solutions approchées satisfaisantes pour un grand nombre de problèmes.
En général les améliorations suggérées et apportées aux MOEAs évoluent autour de trois axes : l'élitisme, le maintien de la diversité et l'hybridation (Coopération).

.2.1 L'Élitisme

L'élitisme peut être traduit par le fait de stocker les meilleures solutions dans une population secondaire qui servira comme une réserve pour produire de nouvelles solutions pour les générations futures Knowles et Corne, 03

Supprimé : [

Supprimé :].

Une première implémentation de ce mécanisme dans un algorithme génétique est présentée dans [De Jong, 75]. L'élitisme a été introduit pour conserver les bonnes solutions lors du passage de la génération courante à la prochaine génération. Conserver ces solutions pour les générations futures a permis d'améliorer les performances des algorithmes sur certains problèmes.
Réaliser un algorithme élitiste dans le cadre des problèmes multi-objectif est plus difficile que pour les problèmes mono-objectif. En effet, la meilleure solution n'est plus un unique individu, mais tout un ensemble dont la taille peut aller jusqu'à dépasser la taille maximale de la population actuelle. Deux adaptations du mécanisme élitiste sont considérées : la première approche regroupe les algorithmes fondés sur les travaux de De Jong. Ces algorithmes tentent de conserver pour les générations futures les k meilleurs individus [Deb et autres, 00a]. Mais comment sélectionner k individus si l'ensemble des solutions non dominées actuel comporte plus de k solutions? Il y a ici un risque de perdre une partie du l'ensemble des solutions optimales, et le concept de l'élitisme n'est plus complètement présent.
La seconde approche est la plus récente, elle consiste à utiliser une population externe d'individus dans laquelle est stocké le meilleur ensemble des solutions non dominées découvertes [Zitzler et Thiele, 99]. Cet ensemble est mis à jour continuellement pendant la recherche, et les individus stockés continus à pouvoir être choisis par l'opérateur de sélection. Ils peuvent ainsi se reproduire et transmettre leurs caractéristiques aux générations suivantes.
Actuellement, les algorithmes élitistes obtiennent de meilleurs résultats pour un grand nombre de problèmes multi-objectif [Zitzler et Thiele, 99; Deb et autres, 02].

.2.2 Le Maintien de la Diversité

Démarrant par une population d'individus non homogène, la diversité de cette population doit en effet être entretenue par un MOEA au cours des générations afin de parcourir le plus largement possible l'espace d'état. Les MOEAs classiques sont réputés pour être très sensibles quant au choix de la population initiale ainsi qu'aux mauvais échantillonnages lors de la sélection. Cette fragilité est observable sur le plan de la perte de diversité ou ce qu'on appelle aussi la dérive génétique. Pour pallier cet inconvénient plusieurs techniques visant à maintenir la diversité dans la population ont été proposées dans la littérature.

v La Fonction de Partage ou Sharing.

Le sharing ou l'heuristique de partage a été introduite par Goldberg et Richardson (1987). Le sharing pénalise les valeurs de fitness en fonction du taux de groupement de la population dans le voisinage d'un individu. Elle permet d'éviter l'agglomération des individus au niveau de l'optimum global au détriment des optima locaux (figure 4.9). Pour effectuer cette distribution (figure 4.10), la valeur de fitness de chaque individu (*x*) est dégradée par un compteur de niche *m*(*x*), calculé pour ce même individu par rapport au nombre d'individus semblables dans la population.

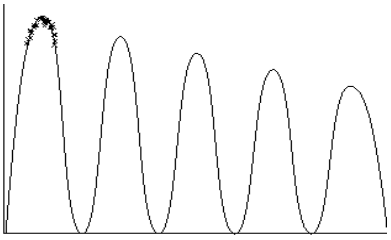


Figure 4.9 : Sans *sharing*

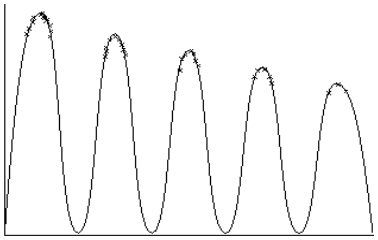


Figure 4.10 : *Avec sharing*

La nouvelle fonction de partage calculée (*shared fitness) f'* (formule 4.2) est obtenue en divisant la valeur de fitness de l'individu par le compteur de niche. Le compteur de niche *m(x)* donne une estimation du nombre d'individus qui se trouvent dans le voisinage de l'individu x. Ce coefficient est calculé par rapport à tous les individus (*y*) de la population courante.

$$f'(x) = f(x) \Big/ m(x) \tag{4.2}$$

Compteur de niche est calculé par :

$$m(x) = \sum_{y \in pop.} sh(dist(x, y)) \tag{4.3}$$

Avec *dist* (x,y) est la distance entre l'individu x et y ;
Et *sh(dist(x, y))* : fonction décroissante de *dist(x, y)* , tel que :

$$sh(0) = 1 \quad \text{et} \quad sh(d >= s_{sh}) = 0 ;$$

$$sh(dist(x, y)) = \begin{cases} 1 - \left(\dfrac{dist(x, y)}{s_{sh}} \right)^{a} & si \quad dist(x, y) < s_{sh} \\ 0 \quad \text{sinon} \end{cases} \tag{4.4}$$

s_{sh} : Rayon de niche, fixé dans la plupart des cas par l'utilisateur en fonction de la distance minimale de séparation voulue entre les différents optimums.
α : est un paramètre qui permet d'amplifier ou de diminuer l'influence de la proximité de y à x. en général α est choisi égal à 2 .
Pour maintenir la diversité le long du front de Pareto, l'individu qui a le plus petit compteur de niche est sélectionné,
La distance entre deux individus, *d(x, y)*, peut être définie dans l'espace des objectifs ou dans l'espace de recherche. Ce choix dépend souvent du problème, car le maintien de la diversité dans l'espace objectif, bien qu'il soit souvent plus simple à réaliser, n'assure pas forcément le maintien de la diversité dans l'espace de recherche [Barichard, 03].
La difficulté de cette fonction est le réglage des paramètres α et s_{sh} et le choix de la métrique utilisée (dist). Sa complexité est de $O(N^2)$.

v Le Clustering

Dans certains MOEAs et lorsque la taille de l'ensemble Pareto optimal devient très grande et dépasse un certain seuil fixé au départ, ils utilisent une technique de *clustering* pour réduire la taille de celui-ci en supprimant certaines solutions. Le but étant d'alléger le front Pareto et lui donner une structure bien distribuée et uniforme. Les raisons de cette pratique sont :

- § Représenter toutes les solutions au décideur est inutile si le nombre est très grand
- § Réduire le temps de calcul car ceci risque de devenir grand si celui-ci est très grand.
- § Dans le cas des fonctions continues, garder toutes les solutions n'est pas nécessaire.

Les techniques de clustering ont été étudiées intensivement dans le contexte des analyses de cluster [Morse, 80] et ont été appliquées avec succès pour déterminer des partitions d'une collection relativement hétérogène d'éléments. La technique de clustering a été utilisée par l'algorithme SPEA [Zitzler et Thiele, 99]. Au début de la procédure, chaque individu constitue son propre cluster, puis on fusionne deux à deux les clusters les plus proches en terme de distance. Cette étape est itérée jusqu'à l'obtention du nombre désiré de clusters. Une fois les clusters identifiés, il ne reste plus qu'à choisir un représentant par cluster. Ce représentant peut être déterminé de plusieurs façons, par exemple en prenant le barycentre du cluster. C'est ce représentant qui sera gardé, les autres éléments étant tout simplement supprimés. Les étapes de la méthode de clustering sont illustrées à la figure 4.11.

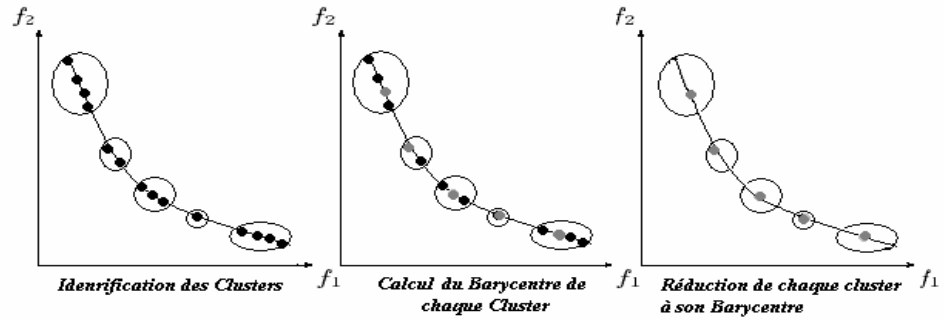


Figure 4.11 : *Illustration de Clustering pour un problème Bi-objectif*

v La Technique de Remplacement ou Crowding

Le *Crowding* (Encombrement ou Surpeuplement) fut introduit par De Jong [De Jong, 75]. La principe originel consistait à remplacer les individus d'une population par d'autres plus

performants. Les individus situés dans des zones différentes de l'espace d'état ne sont pas en concurrence pour le même optimum, alors que les individus proches sont en compétition. En remplaçant les individus semblables, le *crowding* permet de conserver les différentes niches de la population tout en accélérant la convergence.

- On sélectionne deux parents puis on génère deux enfants par croisements et mutation
- En suite on forme des couples parents enfants et on calcule la similarité parent-enfant.
- Pour finir le couple le plus proche est inséré dans la population.

C'est un algorithme de complexité *O(2N)* appliqué sur le génotype. D'autres algorithmes appliquent le crowding sur le phénotype et utilise non pas la similarité mais plutôt la divergence car elles tentent d'éloigner les solutions les unes des autres.
Le crowding est considéré aussi meilleur que le sharing dans la répartition des solutions sur le front Pareto
L'approche par crowding consiste à déterminer un représentant par niche découverte. À la différence du sharing, où tous les individus sont susceptibles d'être sélectionnés et de participer aux phases de croisement, mutation et sélection, avec le crowding, seuls les représentants participeront aux différentes étapes de l'algorithme [Barichard, 03].

Calcul de la distance de crowding

La distance de *crowding* d'une solution (S*i*) se calcule en fonction du périmètre de l'hypercube formé par les points les plus proches de (S*i*) sur chaque objectif [Deb et autres, 02]. La figure 4.12 montre une représentation à deux dimensions associée à la solution (S*i*). Le calcul de la distance de *crowding* nécessite, avant tout, le tri des solutions selon chaque objectif, dans un ordre ascendant. Ensuite, pour chaque objectif, les individus possédant les valeurs limites (la plus petite et la plus grande valeur de fonction objectif ; S_1 et S_l) se voient associés une distance infinie (∞).

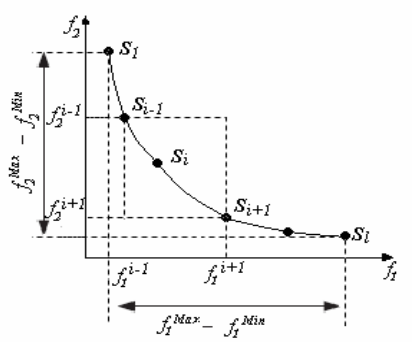


Figure 4.12 : *La distance de crowding*

Pour les autres solutions intermédiaires, on calcule une distance de *crowding* égale à la différence normalisée des valeurs des fonctions objectif de deux solutions adjacentes. Ce calcul est réalisé pour chaque fonction objectif. La distance de *crowding* d'une solution est calculée en sommant les distances correspondantes à chaque objectif. Cette distance de crowding correspond alors à la mesure de sa proximité à ses plus proches voisines, rapportée à la taille du front.
L'algorithme suivant montre la procédure de calcul de la distance de toutes les solutions non dominées de l'ensemble P :

Algorithme 4.2 : *Distance-Crowding ;*

$l = |P|$; // Le nombre de solutions dans l'ensemble P
Pour chaque i
  poser $P[i]_{distance} = 0$; // Initialiser les distances
Pour chaque objectif m
  P = Trier (P,m) // Trier selon la valeur de l'objectif
  $P[1]_{distance} = \infty$;
  $P[l]_{distance} = \infty$;
Pour i = 2 à l-1
  $P[i]_{distance} = P[i]_{distance} + (f_m^{i+1} - f_m^{i-1}) / (f_m^{Max} - f_m^{Min})$

Dans cet algorithme, f_m^{i+1} *et* f_m^{i-1} représentent respectivement la valeur de la m^e fonction objectif de la solution S_{i+1} et S_{i-1}, alors que les paramètres f_m^{Max} *et* f_m^{Min} représentent les valeurs maximale et minimale de la m^e fonction objectif. Après ce calcul, toutes les solutions de P auront une distance métrique $P[i]_{distance}$
Cet algorithme est de complexité $O(MN \log(N))$, où M est le nombre d'objectifs du problème et N le nombre d'individus à traiter. Une fois tous les $P[i]_{distance}$ calculées, il ne reste plus qu'à les trier par ordre décroissant. Pour deux solutions qui appartiennent au même front c-à-d ayant le même rang, on préfère la solution qui est localisée dans la région où la densité de solutions est moindre, soit l'individu possédant la plus grande valeur de distance de *crowding*.

- **Introduction de Nouveaux Individus (Random Immigrant)**.

Cette technique introduite par [Grefenstette, 92]. Appelée aussi Migration de diversité Stochastique [Jourdan, 03]. Une partie des individus de la population est remplacée par des individus générés aléatoirement. On suppose que se sont les mauvais individus qui sont remplacés par les nouveaux dans la population. D'après [Grefenstette, 92] le taux de remplacement peut atteindre les 30% de la population. L'avantage de cette méthode est qu'elle permet d'introduire de la diversité dans la population sans dégrader le processus de la convergence.

.2.3 L'hybridation

Une autre façon d'améliorer les performances d'un algorithme ou de combler certaines de ses lacunes consiste à le combiner avec une autre heuristique [Talbi, 00]. L'idée dans ce cas est de lancer le MOEA en premier lieu afin d'approcher la frontière Pareto, après quoi une autre technique s'occupera du raffinement des solutions trouvées par le MOEA afin de mieux approcher l'ensemble des solutions optimales. Dans ce contexte le MOEA est utilisé pour son pouvoir d'exploration et sa capacité de fournir une approximation de l'ensemble du front Pareto. La technique associée utilise les solutions trouvées par le MOEA comme point de départ, en vue de les améliorer.
Ce principe général, appelé hybridation, peut s'appliquer pour un grand nombre de méthodes. Une multitude d'algorithmes hybrides ont fait leur apparition ces dernières années. Un cas particulier de l'hybridation entre deux méthodes consiste à combiner un algorithme évolutionnaire et une méthode de recherche locale [Barichard, 03 ; Jourdan, 03].
Dans une telle hybridation, on substitue souvent la mutation par une méthode de recherche locale. En effet, les méthodes de recherche locale remplacent la solution courante par une autre voisine. Il y a donc peu de modifications séparant les deux solutions. Pour ce type de méthodes, on peut ainsi citer : la méthode PSA [Czyzak et Jaszkiewicz, 98] combinant un algorithme génétique et le recuit simulé. M-PAES [Knowles et Corne, 00b] combine un algorithme génétique et la recherche locale.

.3 Les Principaux MOEAs Développés

Dans la plupart des MOEAs développés, il s'agit de satisfaire deux points importants [Deb, 99] :
- Trouver des solutions aussi proches que possible des vraies solutions Pareto optimales.
- Trouver un ensemble de solutions très variées tout le long du front.

Dans la littérature, on distingue deux générations de MOEAs non-élitistes et élitistes.

.3.1 Les MOEAs non Élitistes

Vector Evaluated Genetic Algorithm (VEGA)

C'est peut-être le premier MOEA [Schaffer, 85] développé. À chaque génération, un nombre de sous populations est généré par sélection en fonction de l'un des *k* objectifs (figure 4.13). Il y aura donc autant de population que d'objectifs. Chaque population d'individus sera utilisée pour optimiser un objectif distinct. Puis les *k* populations sont mélangées pour en former une seule et des opérateurs génétiques sont alors appliqués pour reproduire la nouvelle population de la génération future.
Avantage : C'est une technique facile à mettre en œuvre.
L'inconvénient : cette technique va produire des individus excellents pour un objectif et non pour l'ensemble des objectifs. Toutes les solutions de moyenne performance, qui peuvent être de très bons compromis, risquent de disparaître avec ce type de sélection.

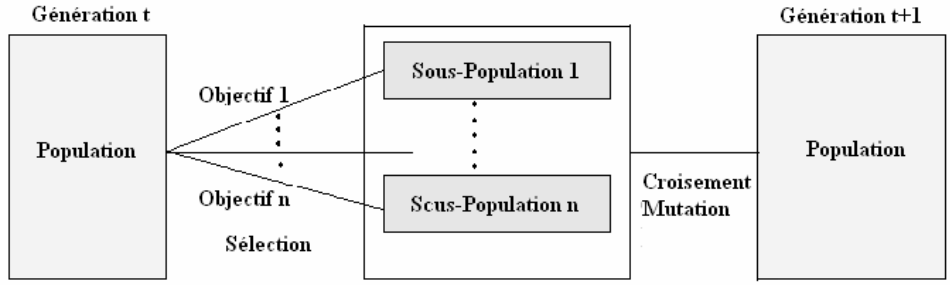


Figure 4.13 : *Le fonctionnement de VEGA*

4 Multi-objectif Genetic Algorithm (MOGA)

Dans cet algorithme [Fonseca et Fleming, 93] ont introduit pour la première fois la notion de dominance au sens de Pareto. À chaque individu est affecté un rang selon le nombre des individus dans la population qui le dominent, c.-à-d. toutes les solutions non dominées sont assignées du rang 1.
Une valeur de fitness est assignée à chacun des individus en utilisant une interpolation entre le meilleur et le plus mauvais rang. La fitness attribuée dépend du nombre des individus ayant le même rang. La fitness de tous les individus ayant le même rang est ramenée à une moyenne et cette valeur est assignée à tous.
Le rang est calculé selon la formule suivante :

$$rang(x,t) = 1 + p^t \qquad (4.5)$$

Avec x est un individu dans la population de la génération *t* et p^t est le nombre des individus qui le dominent.

Algorithme 4.3 : *MOGA ;*
Initialisation de la population Évaluation des objectifs Assignation d'un rang basé sur la dominance Assignation d'une efficacité à partir de ce rang Pour i =1 à N Sélection aléatoire proportionnelle à l'efficacité ; Croisement ; Mutation ; Évaluation des individus ; Assignation d'un rang basé sur la dominance ; Assignation d'une efficacité à partir de ce rang ; Fin Pour

Fonseca et Fleming ont introduit une fonction de partage (sharing) entre les solutions du même rang. Des distances normalisées entre chaque paire de solutions de même rang sont calculées dans l'espace des objectifs :

$$d_{ij} = \sqrt{\sum_{k=1}^{M} \left(\frac{f_k^i - f_k^j}{f_k^{Max} - f_k^{Min}}\right)^2} \qquad (4.6)$$

Où f_k^{Max} *et* f_k^{Min} sont les valeurs maximale et minimale de la fonction objectif k atteintes sur la population courante. La fonction de partage $Sh(d_{ij})$ est calculée selon l'équation (4.4) avec α=1 et le compteur de niches est donné par la formule suivante :

$$cn_i = \sum_{j=1}^{|r_i|} Sh(d_{ij}) \qquad (4.7)$$

Où $|r_i|$ est le nombre des individus ayant le rang r_i. La nouvelle valeur de fitness F' est obtenue par

$$F_i' = F_i / cn_i \qquad (4.8)$$

Les valeurs de fitness 'partagée' *F'* ne sont plus les mêmes pour les individus du même rang, les solutions situées dans le régions éparses auront une meilleure valeur.

Avantage : Cet algorithme obtient des solutions de bonne qualité et son implémentation est facile.
Inconvénient : L'utilisation de la sélection par rang a tendance à répartir la population autour du même optimum. Les auteurs utilisent une fonction de sharing mais les performances dépendent de la valeur du paramètre (σ_{sh}).

5 Niched Pareto Genetic Algorithm (NPGA)

Proposé par [Horn et Napfliotis, 93], cette méthode utilise une sélection par tournoi des individus basé sur le concept de la dominance au sens Pareto (voir section 4.2.9.2). À chaque tournoi, deux individus candidats, sont sélectionnés aléatoirement dans la population initiale. Au lieu de limiter la comparaison aux deux individus, un sous-ensemble d'individus (ou ensemble de comparaison) est également sélectionné aléatoirement dans la population. Si l'un des deux individus est non dominé par le sous-ensemble, il sera sélectionné pour la reproduction. Si les deux individus sont dominés ou non-dominés, alors une fonction de partage sera appliquée afin de les départager. Le candidat ayant le plus petit compteur de niche est sélectionné.

Les auteurs de cette méthode [Horn et Napfliotis, 93], ont précisé qu'à travers des expérimentations, la valeur du paramètre $t_{dom,}$ qui représente la taille de groupe d'individus de sous-ensemble auxquels seront comparés les deux individus concurrents , influe sur les performance de l'algorithme :

- Si $t_{dom} \approx 1\%$ de la population totale, la convergence sera très lente et élitisme ne sera pas bien représenté. Trop d'individus dominés persistent dans la population
- Si $t_{dom} \approx 10\%$ de la population totale, une bonne distribution des individus est alors obtenue.
- Si $t_{dom} > 20\%$ de la population totale, il y a une convergence prématurée.

Avantage : D'après [CoelloCoello, 01], puisque l'algorithme *NPGA* n'est pas basé sur le classement de Pareto (*Pareto ranking)* de tous les individus de la population, mais sur seulement une partie, à chaque génération, il est donc considéré comme étant plus rapide que les algorithmes basés sur le tri.
Inconvénient : Le principal désavantage de cet algorithme est qu'il nécessite, en plus de spécifier le facteur de partage (σ_{sh}) est un paramètre supplémentaire qui est la taille du tournoi (t_{dom}).

La version améliorée de cet algorithme, appelée NPGA-2 est décrit dans [Erickson et autres, 01]. Comme pour le *NPGA*, le compteur de niche est calculé en utilisant les individus de la population partiellement remplie de la génération suivante, plutôt que d'utiliser la population de la génération courante.
Avantage : si le paramètre t_{dom} est beaucoup plus petit que N, le coût de calcul ne dépendra presque pas du nombre d'objectifs, ce qui rend NPGA particulièrement intéressant pour la résolution des problèmes à plusieurs objectifs
Inconvénient : NPGA exige la définition de deux paramètres t_{dom} et σ_{sh}.

.3.2 MOEAs Élitistes

Dans cette catégorie, vont être exposés les MOEAs qui utilisent le concept d'élitisme pour faire converger rapidement l'algorithme vers le front Pareto optimal.

6 Strength Pareto Evolutionary Algorithm 2 (SPEA-2).

Cet algorithme est proposé par [Zitzler et Thiele, 99]. C'est un algorithme qui implante trois techniques communes à d'autres approches telles que : l'utilisation de la dominance au sens de Pareto pour évaluer et sélectionner les individus, l'utilisation d'une population secondaire (archive externe) pour stocker les solutions non-dominées et l'utilisation du clustering pour maintenir la diversité basée sur la notion de niche. La figure 4.14 extraite de [Zitzler et Thiele, 99] montre le fonctionnement de SPEA :

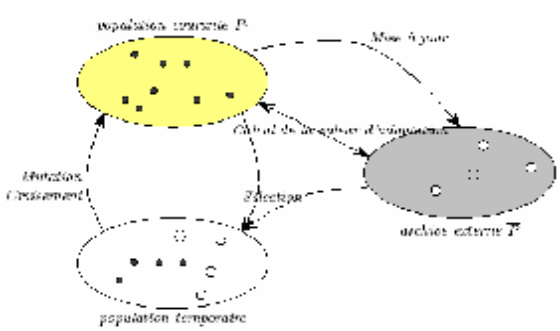


Figure 4.14 : *Illustration du déroulement*

Pour commencer, SPEA2 crée aléatoirement une population initiale *P0* à partir de laquelle il rempli l'archive externe $\overline{P}$ par les individus non dominés dans *P0*. À chaque itération, il calcul la valeur fitness des individus des deux populations. Selon les valeurs de fitness obtenues, les individus vont être sélectionnés pour les opérations de croisement et mutation afin de générer de nouveaux individus. Avant d'entamer une nouvelle génération, l'archive est mis à jour par les individus non dominés nouvellement produits. Si après cette mis à jour, des individus se révèlent dominés, ils seront automatiquement supprimés de l'archive.
Si la taille de l'archive après insertion, excède la taille permise, une réduction par clustering est alors effectuée.

Algorithme 4.4 : *Pseudo-code de l'algorithme SPEA .*

Initialiser la population *P0* et créer l'archive externe vide $\overline{P}$;
Mise à jour de $\overline{P}$ à partir des individus non dominés de *P0*
Tantque critère d'arrêt non rencontré faire
    Calcul de la valeur de fitness pour tous les individus de *Pt* + $\overline{P}$
    Sélection dans *Pt* + $\overline{P}$ en fonction de la valeur de fitness
    Croisement
    Mutation
    Mise à jour de $\overline{P}$ à partir des individus non dominé de *Pt*
FinTantQue

Le calcul de la valeur de fitness s'effectue en deux étapes de la manière suivante :
La première étape consiste à affecter une valeur, appelée valeur de force ($S \in [0,1]$), aux individus de l'archive $\overline{P}$. Cette valeur est déduite à partir du nombre d'individus qu'un élément $i \in \overline{P}$ domine faiblement dans la population courante Pt :

$$S(i) = \frac{Nombre\ des\ do\min és\ par\ i}{\left|P_t\right| + 1} \qquad (4.9)$$

La valeur de fitness d'un individu $i \in \overline{P}$ est égale à sa valeur de Force : F(i) = S(i).
Pour un individu j de la population courante Pt, la valeur de fitness est calculée en sommant les valeurs de force des individus de $\overline{P}$ dominant j plus 1.
Le chiffre 1 est ajouté au total de la somme pour éviter que des individus de $\overline{P}$ aient une valeur de fitness plus grande que certains individus de Pt.
Ainsi, plus un individu est dominé par les individus de $\overline{P}$ et plus sa valeur d'adaptation décroît, diminuant donc ses chances d'être sélectionné.

Avantage: cette méthode distribue efficacement les individus sur la surface de compromis.
Inconvénient : le calcul de fitness dépend de la taille de l'archive externe choisie par l'utilisateur. Cette taille qui risque de fausser la performance de l'algorithme car si l'archive est trop grande, il y a risque de non convergence, d'autre part si elle est trop petite, l'effet de l'élitisme peut être perdu. Et enfin le mécanisme de clustering utilisé à tendance à ne pas préserver les individus qui se situent sur les deux extrémités de la surface de compromis. Ceci peut influencer la représentation du front optimal final.

La version améliorée de cette technique, appelée SPEA2 est décrite dans [Zitzler et autres, 01]. SPEA2 a trois différences principales avec son prédécesseur:

- Il incorpore une technique de calcul de la valeur de fitness « plus raffinée » qui tient compte pour chaque individu d'une information supplémentaire sur la densité de l'individu dans la population.

§ Il emploie une adaptation d'une technique « $k^{ième}$ plus proche voisin » où la densité d'un individu est une fonction décroissante de la distance entre cet individus et on $k^{ième}$ voisin.
§ Il a une méthode de réduction d'archive basée sur la troncation et pas sur le clustering qui garantit la conservation des solutions du front de Pareto.
§ Seuls les individus du front participent à la production.

7 Non dominated Sorting Genetic Algorithm 2 (NSGA-2)

[Deb et autres, 00b] ont proposé une nouvelle version de l'algorithme *NSGA* [Srinivas et Deb, 95] est le NSGA-2. Il est considéré plus efficace que son prédécesseur.
NSGA-2 est un algorithme élitiste n'utilisant pas d'archive externe pour stocker l'élite. Pour gérer l'élitisme, NSGA-2 assure qu'à chaque nouvelle génération, les meilleurs individus rencontrés soient conservés pour la génération suivante.

Dans cet algorithme, on utilise une population de taille (*N*), La population est ensuite triée selon un critère de non-dominance pour identifier les différents fronts *F1*, *F2*.. , etc. comme le montre la figure 4.15. Les meilleurs individus vont se retrouver dans le ou les premiers fronts. Une nouvelle population (P_{t+1}) est formée en ajoutant les premiers fronts au complet (premier front *F1*, second front *F2*, etc.) tant que ceux-ci ne dépassent pas *N/2*. Si le nombre d'individus présents dans (P_{t+1}) est inférieur à (*N/2*), une procédure de *crowding* est appliquée sur le premier front suivant (*Fi*), non inclus dans (P_{t+1}).
Le but de cet opérateur est d'insérer les ($N/2$ - $|P_{t+1}|$) meilleurs individus qui manquent dans la population (P_{t+1}).
Une fois que la première moitié des individus appartenant à la population (P_{t+1}) sont identifiés, la moitié restante de la population P_{t+1} est créée par sélection, croisement et mutation.

NSGA-2 se distingue donc de son prédécesseur par :
§ Il utilise une approche élitiste qui permet de sauvegarder les meilleures solutions trouvées lors des générations précédentes.
§ Il utilise une procédure de tri basée sur la non-dominance, plus rapide.
§ Il ne nécessite aucun réglage de paramètre.
§ Il utilise un opérateur de comparaison basé sur un calcul de la distance de *crowding* (voir la section 4.3.2.3).

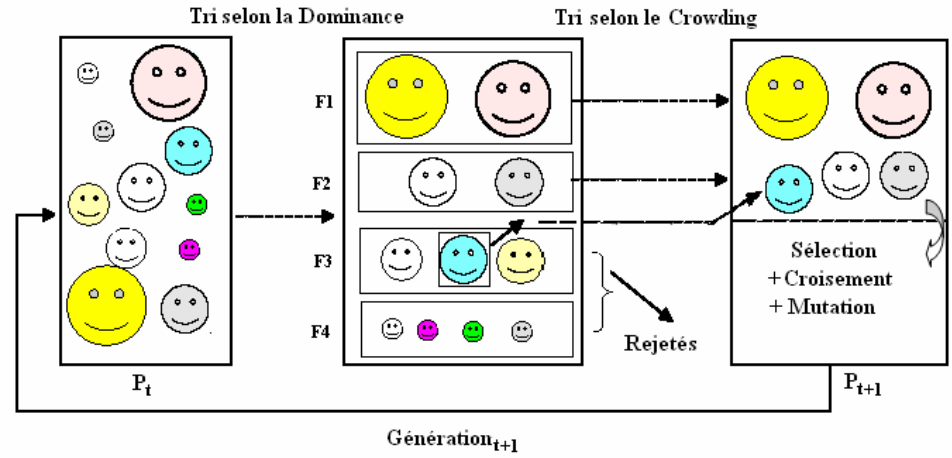


Figure 4.15 : *Déroulement de NSGA-2*

Avantage : cette nouvelle version de NSGA n'exige aucun paramètre à fixer pour le maintien de la diversité. Elle a réduit la complexité de l'algorithme.

Inconvénient : le mécanisme d'évolution de NSGA-2 est tel qu'à partir d'une certaine génération toute la population est contenue dans le premier front Pareto. À ce stade, des solutions situées dans des régions très peuplées peuvent être éliminées en laissant la place à des solutions non-dominées dans la population courante mais qui ne sont pas optimales.

8 Pareto-Archived Evolutionary Strategy (PAES).

PAES est un algorithme classé stratégie d'évolution. Les auteurs [Knowles et Corne, 00a] de cet algorithme ont été motivés pour l'utilisation de la stratégie d'évolution car ils ont remarqué que lors de la résolution des problèmes de MOO dans le domaine de télécommunication, les stratégies basées sur le voisinage telles que la recherche taboue et le recuit simulé, donnent de meilleurs résultats que les approches qui utilisent une population de solutions.

Cet algorithme commence par générer aléatoirement une solution initiale et puis, une solution candidate est produite dans chaque itération au moyen de mutation. Une archive externe (de taille limitée) est maintenue pour rassembler les solutions non-dominées. La solution candidate est écartée si elle est dominée par la solution courante ou n'importe quelle autre solution dans l'archive externe.

Le solution candidate est ajoutée à l'archive et devient la solution courante si elle domine la solution courante. Si aucune d'elles ne domine l'autre, la décision sur la solutions qui va devenir la solution courante et si elle va être ajoutée ou pas à l'archive est basée sur le mécanisme de l'encombrement. D'autres variantes de cet algorithme avec la population ont été également proposés [Knowles, 01].

Algorithme 4.5 : *PAES* ;

```
Génération aléatoire d'une solution S
Ajouter S à l'archive ;                                  // S est la Solution courante
Répéter
Production d'une solution C par mutation de S ;   // C est la solution candidate
Évaluation de C
Si S domine C alors
        Écarter m ;
Sinon   Si C domine S
            Remplacer S par C et ajouter m à l'archive ;
        Sinon   Si C est dominé par un membre de l'archive alors
                    On écarte C ;
                Sinon   S = Meilleur (S, C, archive)
                Finsi
        Finsi
Finsi ;
Jusqu'à condition d'arrêt valide ;
Fin
```

La fonction *Meilleur()* aura pour rôle de déterminer tout d'abord si l'archive n'est pas pleine dans ce cas ajoute *C* à l'archive. Dans le cas contraire, elle détermine la solution à supprimer de l'archive pour pouvoir insérer *C*. La solution à supprimer doit appartenir à une région plus encombrée que celle de *C*. Puis elle détermine laquelle de deux solutions *S* et *C* qui va devenir la solution courante. Bien sûr celle qui appartient à une zone moins encombrée est sélectionnée.

Pour l'application de la technique de crowding, les auteurs utilisent une grille adaptative pour évaluer combien la région dans laquelle chaque solution se situe est peuplée. Cette grille découpe l'espace objectif en hypercubes (Figure 4.16) selon un paramètre *d* à fixer qui détermine le nombre des hypercubes.

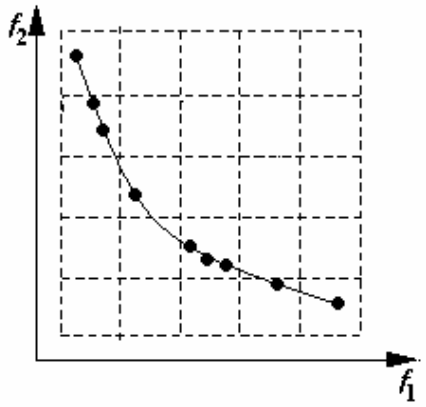


Figure 4.16 : *La grille des de crowding bi-objectif*

Avantage : facile à mettre en œuvre. Plus rapide.
Inconvénient : deux paramètres doivent être fixés : la taille de l'archive et le paramètre de diversité *d.*

9 **Memetic Pareto Archived Evolutionary Strategy (M-PAES)**

Comme il a été déjà signalé, l'hybridation est devenue une approche très convoitée par les chercheurs car elle ouvre devant eux de nouveaux horizons d'exploitation de différentes techniques et ainsi profiter de leurs avantages pour rendre les MOEAs plus performants. Il semblait alors pertinent de développer des approches hybrides entre les MOEAs et des heuristiques adaptées aux problèmes de la MOO. Les techniques basées sur l'hybridation sont souvent désignées sous l'appellation *Algorithmes Mémétiques* [Knowles et Corne, 00b].

M-PAES [Knowles et Corne, 00b] est une variante mémétique de PAES. Cet algorithme mémétique incorpore une population et un opérateur de croisement mais emploie le même mécanisme de sélection que PAES. L'heuristique introduite est la recherche locale. Deux archives sont employées, une est l'archive globale des solutions non dominées et l'autre va servir d'ensemble de comparaison dans la phase de recherche locale. Les deux archives sont vidées après chaque recherche locale et remplie de solutions produites à partir de l'archive globale. Les auteurs de l'algorithme ont précisé que cette version mémétique a surpassé l'algorithme original sur des exemples d'essai du problème multi-objectif de S*ac à dos*.

Toujours dans le contexte des algorithmes mémétiques, [Knowles et Corne, 04] proposent un framework simple afin de guider la conception d'un algorithme mémétique (algorithme 4.6). Cet algorithme peut servir de guide pour toutes approches MOO.

Algorithm 4.6: *Candidate Memetic Algorithm framework for MOO*
1: P := Initialize(P) 2: A := Nondom(P) 3: while stop criterion not satisfied do 4: while stagnation criterion not satisfied do 5: C := SelectFrom (P ∪ A; sel_sched(succ(SEL))) 6: C' := Vary (C; var_sched(succ(VAR))) 7: C" := LocalSearch (C'; ls_sched(succ(LS))) 8: P := Replace (P ∪ C"; rep_sched(succ(REP))) 9: A := Reduce (Nondom(A ∪ P); red_sched(succ(RED))) 10: end while 11: P := RandomImmigrants(P; imm_sched(succ(IMM))) 12: end while 13: return(A)

Ci-dessous une brève description des étapes de cet algorithme:
Ligne 1 : une population P de solutions est initialisée aléatoirement.
Ligne 2: l'archive A est initialisée par les solutions non-dominées de P.
Ligne 4 : installation d'une boucle intérieure dans laquelle un critère d'arrêt peut être vérifié.
Lignes 5 à 9 : donnent une description de la mise à jour de la population et l'archive. Cinq fonctions différentes sont utilisées dans l'ordre et correspondant à la sélection, reproduction, recherche locale, remplacement dans la population courante et mise à jour de l'archive.
Ligne 11 : insertion de nouveaux individus dans la population pour la diversification
Ligne 13 : retourne les éléments de la dernière archive obtenue.

.3.3 Autres Algorithmes Évolutionnaires Multi-Objectif

Les algorithmes décrits ci-dessus sont justes un échantillon du vaste nombre de méthodes proposées dans la littérature ces dernières années. D'autres approches incluent 'MultiObjective Messy Genetic Algorithm' (MOMGA) I et II [Van Valdhuizen et Lamont, 00 ; Zydallis et autres, 01] et 'Pareto Convergent Genetic Algorithm de (PCGA) [Kumar et Rockett, 02]. Ou encore, les variantes de Micro-GA, les algorithmes génétiques cellulaires, les méthodes d'optimisation d'essaim de particules et les algorithmes génétiques basés Agent.

.4 Étude Comparative

D'une façon générale, on constate que tous les MOEAs emploient les opérateurs génétiques standard (mutation ou/et croisement) et les différences entre ces algorithmes se concentrent autour des stratégies utilisées pour la sélection et la diversification.

Mise à part VEGA, plusieurs MOEAs sont apparus en introduisant la notion de dominance au sens de Pareto dans la sélection tels que : MOGA, NSGA ou encore NPGA. Pour ces algorithmes, la qualité d'une solution est évaluée en fonction de sa dominance au sein de la population et la diversité est maintenue à l'aide d'une stratégie de "sharing". Puis, dans le but d'assurer la convergence vers le front Pareto optimal, la question de préservation de l'élite est devenue fondamentale. Ainsi de nouveaux algorithmes, pouvant faire intervenir ou non des archives de solutions non dominées, ont été proposés tels que : SPEA-2, PAES ou encore NSGA-2. Ainsi une nouvelle classification des MOEAs est utilisée. Cette classification distingue les algorithmes non élitistes, n'ayant aucun opérateur de préservation de l'élite et les algorithmes élitistes prévoyant un opérateur pour préserver l'élite des solutions. Il a été prouvé que la stratégie d'élitisme augmente de manière significative les performances des MOEAs [Deb et autres, 02]. Par conséquent les MOEAs élitistes sont certainement plus performants que ceux

qui ne le sont pas. Par contre il est difficile de dire que la performance d'un des algorithmes SPEA-2, NSGA-2 ou PAES (tous élitistes) est meilleure de celles des deux autres.

Il convient de noter que d'après Knowles et Corne, l'opérateur de mutation utilisé dans PAES peut être considéré comme une forme de la recherche locale [Knowles et Corne, 00b] puisque il effectue des petites modifications sur une solution pour en donner une solution voisine.

Le tableau 4.1 résume quelques caractéristiques des MOEAs cités dans ce chapitre.

Méthode	Type de MOEA	Population	Technique sélection	Maintien diversité	Archive	complexité
VEGA	Génétique	oui	Roulette	----	non	O(NM)
MOGA	Génétique	oui	Ranking	Sharing	non	$O(NM^2)$
NPGA	Génétique	oui	Tournoi	Sharing	non	$O(N^2)$
NSGA2	Génétique	oui	Tournoi	Crowding	oui	$O(NM^2)$
SPEA2	Génétique	oui	Ranking	Clustering	oui	$O(NM^2)$
PAES	Strat.Évol.	non	---	Crowding	oui	O(NM*d*)
M-PAES	Strat.Évol.	oui	---	Crowding	Oui (2)	---

Tableau 4.1

.5 Conclusion

Nous avons présenté dans ce chapitre, les concepts de base des algorithmes évolutionnaires multi-objectif. Les MOEAs sont basés sur la théorie de l'évolution de Darwin. Par analogie avec le monde biologique, un MOEA fait évoluer une population d'individus à l'aide de divers opérateurs : sélection, croisements, mutations. Parmi ces algorithmes, on distingue les algorithmes génétiques qui utilisent un codage des paramètres sous forme de chaîne binaire, par analogie avec l'ADN. Les MOEAs cités dans la littérature sont classés en deux générations : non-élitistes et élitistes. La seconde génération a montré ses performances et les MOEAs développés récemment sont tous élitistes. NSGA-2 et SPEA-2 sont actuellement les MOEAs les plus populaires. Plusieurs applications ont été développées mettant en œuvre les politiques de ces deux algorithmes.
Le succès des MOEAs pendant cette dernière décennie s'explique notamment par leur capacité à trouver une bonne approximation de l'ensemble Pareto optimal en une seule exécution de l'algorithme, à la différence des approches traditionnelles de la MOO, qui ne trouvent qu'une solution 'compromis' à la fois (d'autant que cette solution dépend fortement du choix subjectif de certains paramètres).

Dans un chapitre précédent (chapitre 2) nous avons exposé un problème important de la bioinformatique (MSA) qui nécessite une technique de résolution assez particulière. Ce problème réel est toujours un sujet de recherche et aucune méthode actuelle ne peut prétendre le résoudre totalement. Dans le chapitre suivant, nous allons exposer une approche multi-objectif évolutionnaire (génétique) pour résoudre le problème de MSA.

Chapitre 5 : Approche Évolutionnaire Multi-Objectif pour l'Alignement Multiple des Séquences

o Introduction

Nous avons introduit dans le chapitre 2, l'importance de l'alignement multiple de séquences (Multiple Sequence Alignment : MSA) en biologie moléculaire. Malheureusement, ce problème a été qualifié d'être NP_complet et il ne peut être résolu via une méthode exacte que pour un nombre très limité de séquences.
Pendant la dernière décennie, plusieurs méthodes ont été décrites dans ce domaine. Globalement, en excluant les méthodes exactes, les méthodes décrites pour le MSA peuvent être groupées en deux classes : les méthodes progressives et les méthodes itératives telles que Clustal_W, SAGA, DIALIGN, T_Coffee, Muscle, Align_M… etc (voir chapitre 2, section 3). Étant donné un ensemble de séquences, les méthodes progressives se proposent d'effectuer l'alignement en partant des séquences les plus similaires et en ajoutant les séquences restantes graduellement une par une selon un ordre pré-établi. Elles utilisent souvent la programmation dynamique. La simplicité et la rapidité de traitement sont les avantages majeurs de ces méthodes. Cependant, leur nature gloutonne conduit souvent à des solutions sous optimales. Ce qui explique le recours aux méthodes itératives pour gérer la complexité combinatoire du problème. Leur principe de base consiste à produire un alignement initial et à le raffiner itérativement de manière déterministe ou stochastique.
L'alignement optimal fourni par ces méthodes est celui qui optimise généralement une fonction objectif choisie ou fonction score. Formulée mathématiquement, cette fonction permet une évaluation quantitative de la signification biologique et évolutionnaire d'un alignement. Sa valeur doit indiquer la relation entre les séquences du point de vue structure et évolution. Le choix d'une fonction objectif est une tâche très délicate. Ceci est justifié par la difficulté même de définir une fonction objectif qui capture fidèlement toute l'information biologique exhibée par un alignement donné. En d'autres termes, comment s'assurer mathématiquement qu'un alignement est correct biologiquement ? Ceci explique l'apparition d'un nombre non négligeable de fonctions objectif telles que *SP*, *WSP*, *Coffee*, *Consensus*, *Entropie* et *Log Expectation*, etc. (voir chapitre 2 section 2). Ces fonctions ont des caractéristiques différentes et permettent d'évaluer les alignements selon des aspects différents. L'optimum mathématique ne coïncidant pas souvent avec l'optimum biologique, ces fonctions permettent, à des degrés variés, de s'approcher de l'optimum biologique.

o Problématique : analyse de cas

Reposant sur une approche mono-objectif, les méthodes proposées dans la littérature fournissent des évaluations qui ne sont pas toujours concluantes.
J.Thompson et ses collaborateurs [Thompson et autres, 99] ont construit, une base de données de références (BaliBase) qui contient une liste d'alignements dits « vrais » recueillis de la littérature. Ces alignements ont servi aux auteurs et via un programme ***Bali_score*** d'évaluer les performances de plusieurs méthodes et programmes. Les résultats de ces évaluations sont publiés sur le site (http://www-bio3d_igbmc.u-strasbg.fr/balibase.result) et exprimés sous forme de deux scores : *CS* (le pourcentage de colonnes correctement alignées) et *SPS* (le pourcentage

Mis en forme :
Police :Italique, Police de script complexe :Italique

Mis en forme :
Police :Italique, Police de script complexe :Italique

des paires de résidus correctement alignés). Sachant qu'un score (SPS ou CS) est considéré optimal lorsque il est très proche ou égal à 1. Les résultats publiés ont révélés les forces et les faiblesses des méthodes évaluées (Table 5.1). Une vérité ne peut être ignorée en observant ces résultats est qu'aucune méthode quelque soit ses capacités n'arrive à aligner correctement tous les types de séquences. En effet, ces difficultés sont issues certainement de la nature complexe des données biologiques, mais en plus de l'absence d'une fonction objectif ou une norme universelle capable d'identifier nettement le meilleur alignement du point de vue biologique et évolutionnaire.

Référence 1 - V1

	Dataset	PRRP	ClustalW	SAGA	DIALIGN	T_Coffee	Align_m
Score CS	1aboA	0.560	0.545	0.529	0.359	**0.703**	0.526
	1idy	0.606	**0.705**	0.342	0.018	0.566	0.080
	1r69	**0.837**	0.481	0.550	0.406	0.325	0.225
	1tvxA	0.378	**0.438**	0.278	0.306	0.228	0.244
	1ubi	0.498	0.415	0.452	0.000	0.488	0.428
	1wit	**0.991**	0.982	0.899	0.851	0.842	0.763
	2trx	0.494	0.754	**0.801**	0.728	0.500	0.235

Table 5.1: *Résultats de test de BaliBase*

Dans un souci de bien cerner le problème, nous avons procéder à une série de tests afin d'observer de près le comportement de différentes méthodes d'alignements multiple existantes face à plusieurs fonctions objectif prises de la littérature.

Dans ce qui suit, il y a des exemples d'exécution de méthodes récentes de MSA reconnues plus ou moins pour leur efficacité et décrites de façon exhaustive dans le chapitre 2. Nous avons utilisé Muscle, ClustalW, T_Coffee et Align_m. D'un autre côté, nous avons implémenté deux fonctions objectif, *WSP* et *Entropie*. Ces fonctions choisies de la littérature, ont des caractéristiques différentes et permettent d'évaluer les alignements selon des aspects différents. L'évaluation est faite sur des alignements obtenus par l'exécution de ces méthodes. Les tables suivantes montrent des exemples d'évaluation les alignements fournis par les quatre méthodes citées. L'évaluation est effectuée premièrement par les trois fonctions objectif en maximisant les fonctions WSP et Consensus et minimisant l'Entropie, puis par la programme *Bali_score.*

Considérant les scores fournis par Bali_score comme des scores de références, et supposons que la mesure de qualité utilisée est la fonction objectif WSP.

Dataset= 1idy_ref3	T coffee	Muscle	Align_m	ClustalX
WSP * 1.0e+003	1.5057	1.2432	**1.8454**	0.9842
Entropie	115.7560	**106.4731**	140.5397	108.7872
CS (Balibase)	0	0.3640	0	**0.4090**
SPS (Balibase)	0.5550	**0.6460**	0.3140	0.5810

Table 5.2

Dans ce cas certainement les alignement fournis par Muscle ou ClustalX (Table 5.2) ne seront jamais pris pour de bons alignements car leur scores WSP sont inférieurs à celui de l'alignement fourni par Align_m. Malheureusement ce dernier possède des scores SPS et CS nettement inférieurs ce qui signifie qu'il est l'alignement le moins bon parmi les quatre.

La table 5.3 expose un autre cas de figure. Les meilleurs alignements par rapport à Bali_score, sont ceux produits par les méthodes Muscle et ClustalX, pourtant l'alignement fourni par ClustalX possède des scores moyens par rapport aux trois fonctions et il n'optimise aucune fonction. En effet, il représente un bon compromis entre les deux fonctions WSP et Entropie.

Dataset= 1idy_ref2	**T coffee**	**Muscle**	**Align_m**	**ClustalX**
WSP * 1.0e+004	1.6836	2.6594	**2.6802**	2.4627
Entropie	93.7729	**92.6415**	102.6476	92.8495
CS (Balibase)	**0.2000**	**0.2000**	0	**0.2000**
SPS (Balibase)	0.7480	**0.8100**	0.6590	0**.8100**

Table 5.3

Deux alignements différents d'un même ensemble de séquences, peuvent être considérés tous les deux optimaux (table 5.4) vis-à-vis Bali_score et pourtant ils ont des scores différents pour les deux fonctions. L'un optimise WSP, l'autre optimise l'Entropie.

Dataset=1ad2_ref1	**Tcoffee**	**Muscle**	**Align_m**	**ClustalX**
WSP * 1.0e+003	5.7817	6.0530	**6.0818**	5.9287
Entropie	367.444	**357.725**	374.182	363.017
CS (Balibase)	0.8060	**1.0000**	**1.0000**	0.8060
SPS (Balibase)	0.9230	**1.0000**	**1.0000**	0.9230

Table 5.4

Une autre situation a été rencontrée, est que s'il coïncide qu'un alignement possède le meilleur score des deux fonctions simultanément, il est certainement le meilleur parmi l'ensemble des alignements présentés (Table 5.5).

Dataset=Kinase_ref5	Tcoffee	Muscle	Align_m	ClustalX
WSP * 1.0e+003	3589.6	**3627.8**	2016.1	137.1846
Entropie	135.9499	**132.7154**	189.9794	137.1846
CS (Balibase)	0.8390	**0.8870**	0	0.7580
SPS (Balibase)	0.9020	**0.9320**	0.2700	0.8860

Table 5.5

Suite à ces observations, nous avons pu établir le constat suivant:

- Évaluer un alignement en se basant sur une seule fonction objectif n'est pas toujours une évaluation concluante.
- Les alignements évalués n'optimisent pas forcément les fonctions WSP et Entropie simultanément.
- Si un alignement est considéré meilleur pour plusieurs fonctions, il est considéré de même par Bali_score.
- Enfin, nous pouvons conclure que les méthodes de MSA utilisées actuellement n'échouent peut être pas lors de l'établissement de l'alignement optimal mais elles peuvent ne pas l'identifier car il est mal évalué.

Vu la complexité des données biologiques et la variété des méthodes existantes et le fait qu'aucune n'est totalement efficace pour aligner tout type de séquences [Thompson, 05], et en l'absence d'une fonction objectif capable d'évaluer et d'identifier un alignement optimal quelque soit la nature des séquences alignées, le problème du MSA reste toujours ouvert et non complètement résolu.
Une idée alors apparaît très intéressante, est le fait de traiter le problème de MSA comme un problème d'optimisation multi-objectif où différentes fonctions de score sont optimisées simultanément, chose qui n'a pas encore été investiguée. L'optimisation multi-objectif apparaît comme un cadre naturel pour mener cette étude. Elle permet non seulement la prise en compte de plusieurs fonctions objectif simultanément mais aussi d'aboutir à un meilleur compromis se traduisant par une meilleure qualité des solutions et donnant lieu non pas à une seule solution potentielle mais plusieurs solutions potentielles. Ceci a l'avantage de fournir aux biologistes plus de choix au stade de prise de décision.

Dans ce travail de magistère, on est intéressé à la résolution du problème de MSA en investiguant l'apport des approches évolutionnaires multi-objectif. Nous proposons une approche basée sur un algorithme génétique multi-objectif. Pour cela, nous définissons un schéma de représentation pour coder les alignements, des opérateurs appropriés pour faire évoluer une population initiale et une stratégie de sélection permettant de maintenir les bonnes solutions obtenues au cours de l'évolution.

Pour plus de clarté, le reste du chapitre est organisé comme suit : La section 2 fournit une formulation multi-objectif du problème MSA. Les sections 3, 4 et 5 sont dédiées à la description de l'algorithme évolutionnaire multi-objectif proposé, sa dynamique globale et sa complexité. Les résultats expérimentaux feront l'objet de la section 4. Le chapitre s'achèvera par une conclusion.

o Formulation du Problème

§ L'Alignement Optimal

Le problème du MSA peut être défini en spécifiant un couple , où est un ensemble de toutes les solutions faisables qui sont des alignements potentiels des séquences et C est une fonction C : qui attribue à chaque alignement une valeur réelle qui indique approximativement sa qualité. Cette fonction est appelée la fonction du ***Score*** de l'alignement ou mesure, méthode de calcul de score t autres, a.

L'alignement multiple de séquences optimal est un alignement qui vérifie la condition suivante :

optimal et optimal (5.1)

Supprimé : <#>¶
Supprimé : ¶
Supprimé : (Ω,C)
Supprimé : Ω
Supprimé : Ω à R
Supprimé : [Meshoul e
Supprimé : 05
Supprimé :]
Supprimé : A' ∈Ω
Supprimé : C
Supprimé : = {Max(C (A)
Supprimé : / A∈Ω}
Supprimé : C(A')
Supprimé : = C

Dans le cadre de l'optimisation multi-objectif, plusieurs fonctions vent être optimisées afin de déterminer l'alignement optimal.

L'utilisation de plusieurs fonctions objectif simultanément nous permettra de capturer les meilleures caractéristiques de chacune, les points forts de l'une vont combler les imperfections de l'autre. En règle générale, dans l'optimisation multi-objectif, si les fonctions objectifs sont conflictuelles, l'optimisation fournira un ensemble de solutions réalisant le bon compromis. Dans le cas contraire, on obtient une solution unique qui représente la solution idéale ou le point idéal (chapitre 3, section 2).

§ Définition Formelle d'un MSA Multi-Objectif

Un MSA Multi Objectif (MSAMO) peut être formulé sous la forme générale suivante :

- Ayant un ensemble d'alignements multiple de séquences P
- Soit f_1, f_2, …., f_n un ensemble de ***n*** objectifs ou mesures de score.
- Trouver l'alignement optimal $A \in P$ tel que A maximise $\{f_1, f_2, ….. f_n\}$

Dans ce qui suit, trois mesures de score ont été choisies pour être utilisées comme fonctions objectif et qui sont la mesure WSP, Entropie et Consensus. MSAMO peut alors prendre la forme suivante :

$$\text{MSAMO} = \begin{cases} Maximiser WSP(A) \\ Minimiser Entropie(A) \\ Maximiser Consensus(A) \end{cases} \quad (5.2)$$

Le principe de dualité utilisé dans le contexte d'optimisation permet de transformer une minimisation en une maximisation en multipliant la fonction objectif concernée par -1 Le problème de MSA multi-objectif peut être reformulé comme suit :

$$\text{MSAMO} = \begin{cases} Maximiser\ WSP(A) \\ Maximiser(-Entropie(A)) \\ Maximiser\ Consensus(A) \end{cases} \quad (5.3)$$

Dans ce qui suit , la solution proposée pour le problème de MSA traité sous l'angle de l'optimisation multi objectif. La solution a la forme d'un algorithme génétique multi-objectif qui optimise plusieurs fonctions objectif et essaye de converger vers un ensemble de Pareto uniforme et distribué.

o Une Approche Évolutionnaire Multi-Objectif pour le MSA (MsaMO)

Notre approche est un algorithme génétique multi-objectif qui admet un ensemble d'alignements qu'il fait évoluer via des opérateurs génétiques (Croisement, Mutation et Sélection) pour aboutir à un ensemble d'alignements de bonne qualité voire « optimaux » après un certain nombre de générations. Il maintient trois types de populations : la population initiale définie au début, la population courante définie à chaque génération et la population secondaire utilisée comme archive pour garder trace des meilleures solutions trouvées au cours des itérations. Elle sert comme source de production pour chaque population courante.

La méthode mise en œuvre est inspirée des méthodes d'optimisation évolutionnaires multi-objectif élitistes telles que SPEA2 et NSGA2. Pour la mise en œuvre d'une telle méthode, les éléments suivants doivent être définis :

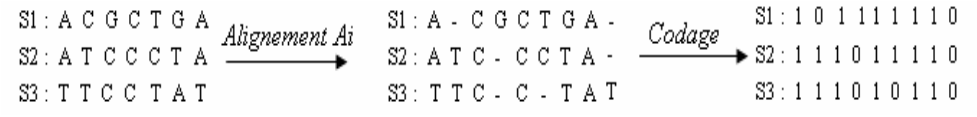
- La représentation ou le codage des individus de la population
- La taille population initiale
- Le mécanisme de génération de la population initiale
- La population secondaire.
- La technique d'évaluation des individus.
- La méthode de sélection d'individus qui vont participer à la génération de la population future.
- Le mécanisme de croisement.
- Le mécanisme de la mutation.
- La technique du maintien de diversité au sein de la population secondaire
- Et enfin le critère d'arrêt

§ Le Codage des Individus

Cette étape est très importante pour un algorithme génétique, elle doit permettre une bonne représentation de tous les individus de la population comme elle doit assurer une bonne interprétation de leurs caractéristiques génotypes et phénotypes.

Un individu étant un alignement potentiel. On doit donc en premier lieu trouver une représentation appropriée permettant la mise en œuvre de l'algorithme multi-objectif. Le schéma de représentation que nous avons adopté associe la valeur 1 à un résidu et 0 au gap. Ainsi le codage d'un individu se traduit par une matrice binaire dont les dimensions dépendent du nombre de séquences à aligner et la longueur maximale obtenue après insertion d'un certain nombre de gaps. La figure 5.2 illustre le codage adopté.

Dans notre cas, les individus sont des alignements de séquences **,** par exemple d'ADN dont l'alphabet est composé des quatre nucléotides (A, C, G, T) ou de protéines. Un ensemble de séquences nucléiques peut se présenter comme suit :



(a) *(b)* *(c)*

Figure 5.2 : *Codage d'alignement : (a) Séquences initiales, (b) Alignement potentiel, (c) Codage binaire correspondant*

Avantage du codage binaire : facilite le codage des opérations de croisement et mutation.

§ La Population Initiale

La population initiale est la matière première avec laquelle doit démarrer un algorithme génétique. Le processus de génération des individus de cette population doit être bien choisi afin de pouvoir fournir à l'algorithme un ensemble d'individus potentiels et diversifiés. Le but est donc de trouver un nombre d'alignements différents pour le même ensemble de séquences pour pouvoir faire évoluer l'algorithme, plusieurs approches sont possibles :

- Dans la méthode *SAGA* [Notredame et autres, 96] les gaps sont insérés aléatoirement dans les séquences de telle sorte qu'elles soient toutes de même longueur. On peut diversifier l'insertion des gaps pour une seule séquence de telle façon à obtenir plusieurs versions d'une même séquence ce qui facilitera la construction de plusieurs alignements différents. Mais cette technique purement aléatoire ne garantit pas la production des solutions intéressantes et risque de rendre la progression de l'algorithme très lente**.**

- Une deuxième méthode consiste en l'implémentation d'une méthode d'alignement progressive selon l'approche de Feng & Doolittle [Feng et Doolittle, 87]. Puis se servir de celle-ci pour générer des alignements pour la population initiale. Ensuite procéder à une suite de raffinements itératifs pour aboutir à la solution optimale. Pour diversifier les solutions lors de leur construction, on utilise des arbres phylogénétiques construits par des méthodes différentes et ainsi on obtient des matrices de poids différents. Ces matrices fourniront des valeurs différentes qui guideront la construction des alignements. Le résultat de cette méthode donne certes des embryons d'alignements mais la procédure du raffinement sera plus lente sans qu'il y ait une garantie de trouver l'optimal recherché.

- Une troisième approche consiste à utiliser des méthodes d'alignement multiple déjà existantes. Ces méthodes d'alignements sont disponibles sur le web et se diffèrent par la manière de construire l'alignement : progressive/itérative et global/local. Chacune d'elles possède une approche différente d'aligner, ce qui offre une diversité entre les individus de la population secondaire. En plus les alignements fournis sont plus ou moins des bons alignements.

L'avantage de cette dernière approche est d'offrir dès le départ des bons alignements par rapport à leur méthode de construction. Pour cette raison nous avons opté pour leur utilisation comme moyen de génération d'alignements pour la population initiale. L'algorithme MSAMO va raffiner ces solutions et donc converger rapidement vers une ou plusieurs solutions de bonne qualité.

§ La Population Secondaire

Vu le comportement stochastique des algorithmes évolutionnaires en général et par conséquent celui de MSAMO, il n'est pas peu probable que des mauvaises solutions restent et persistent jusqu'à la fin de l'algorithme, ou que les solutions optimales se perdent au fur et à mesure que de l'algorithme progresse.
Utiliser une population secondaire (PS) ou *archive* composée essentiellement des solutions non dominées au sens de Pareto, aura pour rôle de sauvegarder les meilleures solutions jusqu'à la fin de l'exécution de l'algorithme (figure 5.1).
Chaque nouvelle itération démarre par la génération de nouveaux individus pour la population. La production des nouveaux individus consiste en l'application des opérateurs génétiques (Croisement et Mutation) sur une copie des individus de la population secondaire. Ce mécanisme garantit la survie des meilleurs individus et les protège contre toute élimination non souhaitée.
À la fin de chaque génération, PS est mise à jour par l'insertion des nouveaux individus non dominées de la population courante. PS est donc un dépôt dans lequel on garde périodiquement de nouvelles solutions non dominées, puis on retire les solutions qui sont devenues dominées après chaque insertion. La procédure de la mise à jour et de réduction de PS fera l'objet du paragraphe 5.3.9.

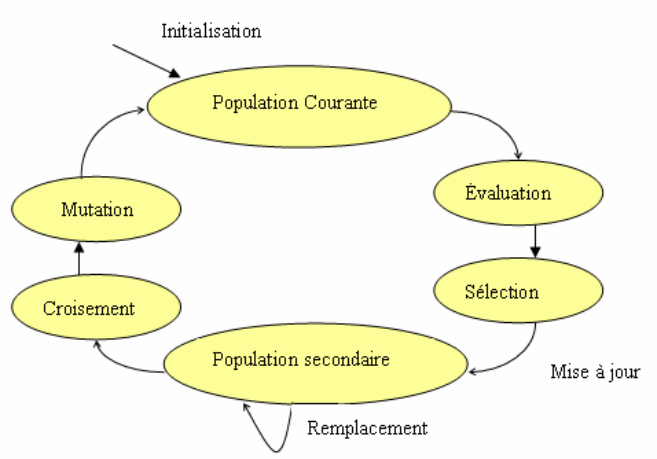


Figure 5.1 : *Illustration du fonctionnement de MSAMOp*

§ *La Taille de la Population*

Dans un algorithme génétique, le nombre des individus de la population doit être suffisamment grand pour offrir une grande possibilité d'exploration de l'espace des solutions et afin de pouvoir trouver et sélectionner des individus pour les opérations de croisement, mutation. Dans le cas des MSAs, un seul individu (alignement) est une matrice dont les dimensions peuvent varier d'un ensemble de séquences vers un autre. Si l'on prend des exemples de datasets (ensemble de séquences) de la base Balibase, un dataset peut contenir de 4 à 27 séquences dont la longueur de chaque séquence peut aller de 63 à 485 caractères. Représenter une population de 100 individus par exemple devient une solution trop coûteuse en espace. Avec les moyens dont nous disposons, nous avons opté pour une population de 20 individus pour la population initiale et 6 pour la population secondaire. Nous pensons que ses valeurs nous permettrons de maintenir une population représentative et diversifiée.

§ **Les Fonctions Objectif Utilisées**

Dans la littérature, on rencontre plusieurs fonctions objectif pour évaluer des alignements, de très simple au plus complexe. Pour implémenter notre approche, nous avons choisi trois parmi elles. Ces fonctions sont décrites de façon exhaustive au niveau du chapitre 2 section 2.3.
La première est la mesure de score **Consensus** [Tompa, 00], c'est une méthode qui permet de déterminer le score de l'alignement à partir d'une séquence appelée '*Consensus'* construite à partir des séquences de l'alignement et qui est la séquence qui minimise la somme des distances entre elle et les séquences alignées. L'avantage de cette fonction est qu'elle permet de mieux évaluer les séquences convergentes en se basant sur une mesure de distances. Dans le cas d'utilisation d'une matrice de substitution telle que PAM ou BLOSUM, on parle alors d'une maximisation de score et non pas d'une minimisation.

Le score consensus peut être calculé via la formule suivante :

$$\text{Consensus (A)} = \sum_{i=1}^{L} d(i) \qquad (5.4)$$

La séquence consensus *C* : $c_1c_2\ c_3\ \ldots c_L$ où L est la longueur des séquences de l'alignement. *d(i)* est la distance calculée entre une colonne *i* de *A* et le résidu c_i correspondant de la séquence *C*.

$$d(i) = \sum_{j=1}^{N} d(S_j[i], c_i) \qquad (5.5)$$

Le résidu consensus c_i d'une colonne i (i = 1,…L) est celui qui minimise la somme des distances entre lui et les autres résidus de cette colonne ; S_j est une séquence alignée de *A*.

La deuxième étant l'**Entropie** [Nicholas et autres, 02], elle nous fournit l'information sur le taux de variation des informations contenues dans les séquences alignées ; plus les colonnes d'un alignement sont bien alignées plus leur entropie est faible.

Si l'on considère p_{ia} la probabilité du caractère *a* dans la colonne *i* d'un alignement *A :*

$$p_{ia} = c_{ia} / \sum_{a'} c_{ia'} \qquad (5.6)$$

Où c_{ia} est le nombre du caractère *a* dans la colonne A[:][i].
Pour chaque colonne de A, l'entropie est calculée par la formule suivante :

$$\text{Entropie (A[:][i])} = \sum_{a} c_{ia} * \log(p_{ia}) \qquad (5.7)$$

Plus la colonne est variable, plus l'entropie est haute. Le but est donc de trouver l'alignement qui minimise la somme des scores d'entropie de toutes les colonnes d'un alignement.

La troisième fonction objectif utilisée est la fonction **WSP** (Weighted Sum of pairs) [Altshul et autres, 89]. Cette fonction a été choisie pour sa capacité à mieux évaluer les séquences divergentes. Le calcul de score étant basé sur les valeurs de poids fournis par des arbres phylogénétiques. L'utilisation des poids permet de ne pas trop pénaliser les séquences distantes. Cette fonction objectif nécessite la construction d'un arbre phylogénique pour la détermination des poids entre séquences. Et utilise une fonction affine des gaps qui permet d'attribuer des pénalités lors de la rencontre de ceux-ci et même en cas de leur extension.
L'arbre phylogénétique est construit selon la méthode N.J [Saitou et Nei, 87].
La formule générale de cette fonction est :

$$\text{WSP (A)} = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \sum_{h=1}^{L} Wij * sc(A[i,h], A[j,h]) \qquad (5.8)$$

Avec *A* l'alignement à évaluer composé de K séquences et ***sc****(A[i,h], [j,h])* : le score de la paire de résidus alignés dont les coordonnés sont (i,h) et (j,h). *Wij* est le poids des séquences i et j, déterminé à partir de l'arbre phylogénique reliant les K séquences.

L'objectif visé derrière le choix de ces fonctions est de pouvoir évaluer d'un alignement selon des aspects différents. Ces aspects peuvent s'avérer complémentaires et parfois contradictoires (voir exemple cités en Introduction) et d'obtenir par conséquent un ensemble de solutions optimales (ensemble Pareto) mais si l'ensemble obtenu est réduit à une solution, elle sera la solution idéale car elle sera la seule qui optimise toutes les fonctions à la fois.

§ Évaluation des Individus et Affectation de Rang

Afin de pouvoir sélectionner les meilleurs individus de la population courante puis enrichir PS, il faut tout d'abord évaluer ces individus pour établir un ordre d'efficacité ou d'adaptation. Le problème étant considéré multi-objectif, l'évaluation des individus va se faire selon plusieurs fonctions objectif. A chaque individu, est associé un *vecteur* de taille égale au nombre de fonctions objectif, il contient les valeurs des fonctions objectif calculées séparément.
Pour établir un ordre d'efficacité entre les individus d'une population, nous avons utilisé le concept du Ranking (voir chapitre 4, section 4.2)
L'ordre d'efficacité est établi comme suit : Le meilleur est l'individu non dominé au sens Pareto D'après l'équation (3 .2), son rang est 'un'. Si par contre, un individu est dominé par un autre individu et par un seul, son rang est 'deux' et ainsi de suite jusqu'au dernier. Le rang d'un individu est déterminé alors par le nombre d'individus qui le dominent. L'algorithme ci dessous illustre le mécanisme d'établissement des rangs.

Algorithme 5.1 : *MSArang*

A : Solution à comparer (vecteur de dimension n = nombre des F.Os)
SC: Ensemble des solutions
$A_{rang} = 1$;
  Pour tout $B \hat{I} SC$ *et* $B \neq A$
    Si $B_1 > A_1$ *et* $B_2 > A_2$……*et* $B_n > A_n$ *alors* A_{rang} ++ ;
  Fin si / sinon les solutions sont incomparables*
  Fin pour
Fin MSArang

Cet ordre va être exploité pour la sélection des individus pour les opérations de croisement et de mutation.

§ La Sélection

Cette opération permet de sélectionner les individus aptes à la reproduction et donc participer aux opérations de croisement et de mutation. Les individus aptes sont souvent les mieux classés lors de l'évaluation. Dans le cadre de notre approche, nous avons utilisé la technique de sélection basée sur le rang par l'exploitation de l'algorithme ***MSArang*** afin d'établir un rang entre les individus de la population, puis sélectionner ceux qui ont le rang égal à '1'. Ces derniers composeront le front Pareto, serviront à l'enrichissement de PS et la génération de nouveaux individus.

§ Le Croisement

C'est une opération classique des algorithmes génétiques (voir chapitre 3, section 2). Elle consiste en la combinaison du matériel génétique de deux individus de la population et en

produire deux nouveaux individus (Enfants ou Descendants) héritant des caractéristiques de leurs parents.
Dans notre cas, on prend deux alignements de la population secondaire, on coupe ces deux alignements chacun en deux parties. La position du croisement sur les deux alignements est tirée au hasard. Les quatre parties obtenues vont servir pour former deux nouveaux alignements. Les nouveaux alignements peuvent être insérés dans la population de la génération courante (Figure.5.3).

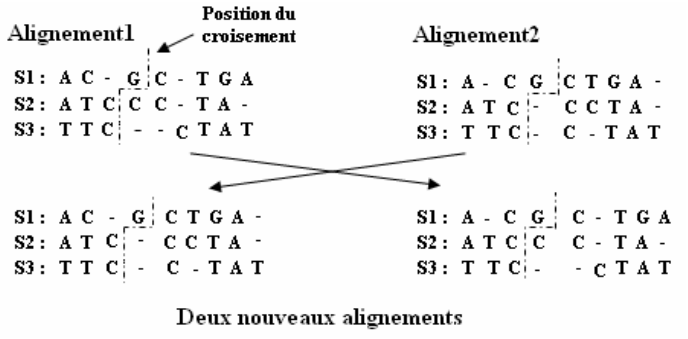


Figure 5.3 *: Le croisement entre deux alignements*

En ce qui concerne notre approche, et sachant que les alignements initiaux sont issus de sources différentes par conséquent ils peuvent avoir les séquences alignées dans des ordres différents. Cependant deux alignements parents ne peuvent être croisés que s'ils ont le même ordre de séquences. (Figure 5.4).
Il est également important de s'assurer également que les alignements produits, ne sont pas une copie des alignements initiaux ni identiques à d'autres alignements déjà insérés dans la population.

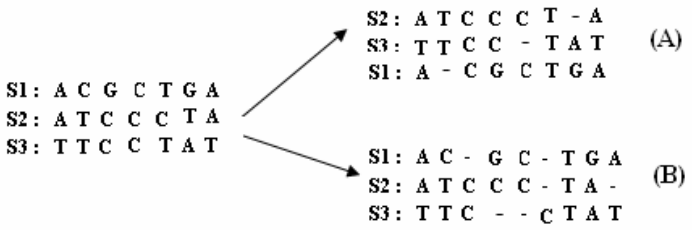


Figure 5.4 : *(A) et (B) sont deux alignements différents du même ensemble de séquences mais dans un ordre différent pour chacun*

§ La Mutation

Ayant un alignement la mutation consiste à permuter un gap et un résidu dans une séquence (figure 5.5) ou un ensemble de gaps consécutifs avec un autre de résidus La taille d'un ensemble ici est une condition nécessaire. Les éléments à permuter doivent être voisins dans la même séquence. L'opération de mutation consiste donc à apporter une certaine diversification dans la population, elle est perçue comme une technique d'exploration de l'espace de recherche.
Cette opération s'effectue sur une ou plusieurs séquences d'un alignement, elle introduit plus de diversité dans les alignements nouvellement produits. Comme les alignements de notre population initiale sont générés par des méthodes différentes, ils peuvent ne pas être tous alignés dans le même ordre de séquences, par conséquent il arrive que l'on ne puisse pas effectuer l'opération du croisement. Dans une telle situation et pour atteindre la taille maximale de la

population, la mutation devient le seul outil capable de produire de nouveaux alignements à partir de ceux déjà crées en y introduisant une modification.

Figure 5.5 : *Mutation d'une position*

Plusieurs opérations de mutations ont été exploitées [Notredame et autres, 96 ; Layeb, 05] à fin de s'assurer que chaque opération de mutation fournit réellement un nouveau individu (figure 5.6). Ces opérations cherchent à permuter pas un seul gap avec un résidu mais des blocs avec d'autres.

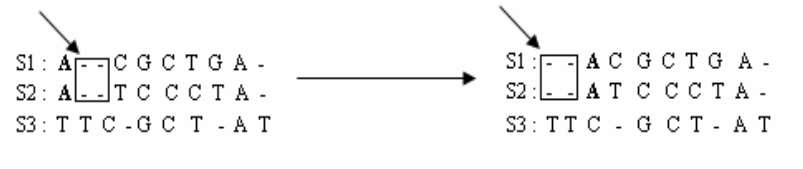


Figure 5.6 : *Mutation d'un bloc de gaps*

§ La Réduction de la Population Secondaire

Pour des raisons de coûts en temps et en espace, la taille de la population secondaire doit être fixe. Lorsque le nombre des solutions non dominées rencontrées après un certain nombre de générations, dépasse la taille permise, il faut trouver un moyen de sélectionner parmi les individus non dominés ceux qui vont être éliminés. Plusieurs techniques ont été développées ayant pour rôle de réduire la taille de *PS* et de la ramener à la taille permise. Cette réduction ne doit pas entraîner ni une perte des bonnes solutions ni une perte de diversification. Il faut alors choisir soigneusement les éléments à éliminer tout en conservant un front Pareto uniforme et distribué. La technique utilisée pour assurer une bonne réduction au niveau de notre *PS* est le *Clustering* basée sur la distance de *Hamming* [Jourdan, 03]. Cette technique a été choisie pour sa faible complexité de *O (N log N).*

Algorithme 5.2 *: Clustering ;*

```
/* C étant l'ensemble des clusters et  PS population secondaire à réduire
C = È_i{x} avec x Î PS                  /* Initialiser l'ensemble C par les éléments de PS
/* fusionner deux à deux  les clusters les proches en terme de distance
  Tant que  |C|  >  TailleMax faire
       Pour tous les couples (x, y) de C  faire
            Si distance (x, y) < Mindistance alors
                 Mindistance = distance(x, y)
            Finsi
       Fin pour
       C1= {cluster(x), cluster(y)} ;       /* regrouper les deux clusters dans un seul
       C= C \ {cluster(x), cluster(y)} ;    /* supprimer les clusters de C
       C= C È C1 ;                          /* ajouter à C le nouveau cluster C1
   Fin tant que
/* calcul du barycentre de chaque cluster pour déterminer le représentant du cluster
   PS = Æ ;                        /* Initialiser PS
  Pour  chaque x  Î  C faire
        y = Calcul_barycentre(x) ;
       PS = PS È y
  Fin pour
 Fin clustering.
```

Les alignements de PS vont être regroupés en clusters. Initialement, les clusters vont contenir des paires d'alignements les proches du point de vue de distance de Hamming. Le regroupement peut s'étendre aux paires de clusters si la taille désirée (TailleMax) n'est pas atteinte.

Pour réduire effectivement la population secondaire, les clusters obtenus vont être réduits chacun à un seul alignement représentatif. L'alignement représentatif est le barycentre du cluster et il est défini comme étant l'élément qui minimise la somme des distances entre lui et tous les éléments de son cluster. Une fois que le barycentre est déterminé, il sera inséré dans population secondaire par contre les autres alignements vont êtres ignorés.
Dans le cas où le cluster ne possède que deux alignements, le choix sera porté sur l'alignement qui possède la meilleure valeur d'entropie.

§ Le Critère d'Arrêt de l'Algorithme

Les algorithmes génétiques sont des méthodes itératives qui ont besoin d'un grand nombre d'itérations pour pouvoir converger vers les solutions optimales.
Le choix d'un critère d'arrêt peut se révéler une tâche très difficile car on ne sait pas si l'objectif de l'algorithme est atteint ou non. Plusieurs critères ont été évoqués dans le chapitre précédent. Pour notre algorithme, il semble que le nombre d'itérations est le seul critère utilisable. On ne peut pas utiliser le critère d'arrêt : « la perte de diversité » puisque notre approche assure le maintien de la diversité via la technique de clustering. Le troisième critère d'arrêt qui consiste à arrêter l'algorithme si son meilleur individu n'évolue plus après un certain nombre de génération, nous semble être un critère non convenable pour les algorithmes génétiques qui sont itératifs et convergent vers le front Pareto optimal en exécutant le plus grand nombre d'itérations possible.

MSAMO utilise donc le nombre des itérations (générations) comme un critère d'arrêt.

o Description de la Dynamique Globale

MSAMO [Benlahrache & Meshoul, 07] démarre par une population initiale générée par des méthodes de MSA existantes auxquels sont ajoutés d'autres alignements générés à partir de ceux-ci par mutation.

Une première évaluation est nécessaire pour déterminer quels sont les meilleurs alignements selon les trois fonctions objectif utilisées. Un rang est alors attribué à chaque alignement selon la dominance au sens de Pareto en exploitant l'algorithme *MSArang*. La population secondaire se voit alors attribuer tous les alignements non dominés c-à-d ayant le rang égal à un. Le nombre de générations étant fixé à N, et tant que ce critère d'arrêt n'est pas atteint, MSAMO va procéder à une suite d'opérations : générer une population par des croisements entre les individus de la population secondaire, si le nombre d'individus produits n'atteint pas la taille fixée, alors ces individus vont subir des mutations pour produire d'autres individus.
La nouvelle population va par la suite subir une évaluation afin de déterminer s'il y a parmi ses individus, des nouveaux éléments non dominés pour les insérer dans la population secondaire.

Une fois les nouveaux individus non dominés sont déterminés et insérés dans la population secondaire, une autre évaluation des éléments de cette population devient nécessaire, car il se peut que les nouveaux individus insérés dominent ceux déjà existants et il va falloir alors les supprimer de la population secondaire.
Puisque la taille de la population secondaire est fixe, et si après un certain nombre de génération, le nombre des individus non dominés dépasse la taille permise alors une réduction de la population secondaire est entamée. La réduction se fait via un algorithme de clustering afin de préserver des éléments diversifiés et bien dispersés sur le front Pareto. Ce qui suit est l'algorithme qui résume le fonctionnement général de notre méthode MSAMO.

Algorithme 5.3 *: MSAMO*

t=0 ;
Initialiser la population P0 ;
Évaluer individus de la population P0 selon les trois objectifs;
Affecter un rang aux individus de P0 selon la dominance de Pareto ;
Créer la population secondaire P' à partir des individus non dominées de P0 ;
Si taille de P' dépasse un seuil alors Réduction de PS ;
Tant que critère d'arrêt non rencontré (t <=N) faire
5 t = t+1 ;
6 Créer la population de Pt à partir de P' par croisement et mutation;
7 Évaluer individus de la population Pt selon les trois objectifs;
8 Affecter un rang aux individus de Pt selon la dominance de Pareto ;
9 Sélection dans Pt en fonction de leur rang ;
10 Mise à jour de P' à partir des individus non dominés de Pt ;
11 Si taille de P' dépasse un seuil alors Réduction de PS ;
Fin Tant Que ;
Fin MSAMOp.

- **La Complexité de l'Algorithme**

La complexit*é* d'un algorithme est définie comme étant le nombre d'opérations à effectuer pour résoudre le problème. Pour notre algorithme, et pour une population de M individus (Alignements), où chaque alignement est une matrice de dimension (N*L) avec N le nombre de séquences dont la taille de la plus longue est L, la complexité de notre algorithme est dépendante de plusieurs facteurs. Les étapes qui coûtent considérablement le temps de la CPU hormis l'étape de la génération des solutions initiales sont :

- La complexité de la fonction objectif WSP (WSP = M*(N*(N-1)/2*L))
- La complexité de la fonction objectif Entropie (Ent = M*L*N)
- La complexité de la fonction objectif Consensus (Con = M*L*N)
- La complexité des opérations de modifications (croisement + mutation) = (M/2)*(N*L) + M
- La complexité de la réduction de l'archive dont la taille est M' = M'logM'
- La complexité d'une itération est égale à la somme des complexités précédentes.
- La complexité de K itérations est égale à :

$$= K*(M*(N*(N-1)/2*L)) + M*L*N + M*L*N + (M/2)*(N*L) + M + M'logM')$$

La complexité de MSAMO peut être évaluée à :

$$O(M*N^2*L + 2* M*L*N + M'logM') \quad (5.9)$$

La formule (5.4) indique un degré de complexité assez élevé, ce degré dépend directement de la taille de l'information manipulée (le nombre des séquences et leur taille) ainsi du nombre des fonctions objectif utilisées pour l'évaluation.

Implémentation et Évaluation

Environnement de Travail

MSAMO [Benlahrache et Meshoul, 07] manipule beaucoup de structure de données et surtout les matrices et les vecteurs. Le langage le plus approprié à ce type de manipulation est MATLAB. La version MATLAB7 a été utilisée pour implémenter notre approche. Ce choix est aussi motivé par la portabilité MATLAB7 ce qui permet l'implémentation de notre algorithme sur différentes plateformes. Cependant MATLAB7 est un peu lent car c'est un langage interprété ceci risque d'augmenter significativement le temps d'exécution de l'algorithme. Le matériel utilisé est un micro-ordinateur Pentium4 dont la fréquence est de 1Ghertz avec 640 Ko de Ram sous le système WindowsXP.

Évaluation et Discussion des Résultats

Pour évaluer les performances de la méthode proposée, plusieurs datasets de *BaliBase* ont été utilisés. BaliBase est une base de données biologique de référence construite à des fins de validation des méthodes de MSA [Thompson, 99]. Elle fournit un programme d'évaluation Bali_score calculant les mesures CS (% colonnes correctement alignées) et SPS (% des paires de résidus correctement alignées). Cette base de référence (Benchmark) est amplement utilisée pour l'évaluation des performances des méthodes d'alignements. *Bali_score* est utilisé pour évaluer des alignements de séquences protéiques. Pour le test des alignements des séquences nucléiques (ADN) et la prédiction structurelle, il y a le programme *ARNz* de la base *BRaliBase* [Gardner et autres, 05].

Pour générer les solutions initiales, nous avons exploité les méthodes : T_Coffee, Muscle, Align_m et ClustalX. La taille de la population initiale est de 20 et celle de la population secondaire est 6 (≈ 1/3 de la population initiale). La matrice de substitution utilisée est *BLOSUM62*. Pour la manipulation des séquences, une transformation vers le format FASTA est

obligatoire. Pour une évaluation par Bali_score, les alignements obtenus doivent être transformés en format MSF.

Notre approche vise quatre objectifs:

- Identifier le meilleur ou les meilleurs alignements parmi plusieurs ;
- Améliorer ces alignements ;
- Obtenir un ou plusieurs alignements optimaux ;
- Obtenir si possible un front Pareto uniforme et bien distribué.

Les deux premiers objectifs visent la qualité biologique des alignements par contre les deux derniers cible une bonne optimisation multi-objectif.
Pour pouvoir bien évaluer les résultats obtenus vis-à-vis les objectifs fixés, nous avons défini deux mesures :

1: Le taux d'identification T_{id}
2: Le taux d'amélioration T_{am}

§ Le taux d'identification des meilleurs alignements est défini comme suit:

$$T_{id} = \frac{Nombre\ d'alignements\ corrects\ identifiés}{Nombre\ total\ des\ alignements} \tag{5.10}$$

Les classes de test pour lesquels le taux d'identification est faible, représentent généralement des classes difficiles à améliorer car tout simplement le meilleur l'alignement n'est pas celui qui optimise les fonctions objectif utilisées. Il faut noter que ce taux d'identification est fortement conditionné par le nombre et le choix des fonctions objectif.

§ Le taux d'amélioration est défini par la formule suivante :

$$T_{am} = \frac{\left(f_1(x') - f_1(x)\right) + \left(f_2(x') - f_2(x)\right)}{f_1(x) + f_2(x)} \tag{5.11}$$

Avec f_1 *et* f_2 les mesures de score (ou fonctions objectif) et x' est l'alignement x amélioré par MSAMO. Appliquer ce taux aux fonctions objectif (WSP, Consensus et Entropie) permet d'indiquer la capacité de MSAMO à apporter des modifications positives au niveau des alignements.
Exploiter ce taux pour les scores fournis par le programme Bali_score (le pourcentage de paires de résidus correctement alignés (SPS) et le pourcentage des colonnes correctement alignées (CS)) permet d'indiquer que les modifications apportées aux alignements sont fructueuses et évoluent dans le bon sens (formule 5.6):

$$T_{am} = \frac{\left(CS(x') - CS(x)\right) + \left(SPS(x') - SPS(x)\right)}{CS(x) + SPS(x)} \tag{5.12}$$

Il faut rappeler que les améliorations sont faites sur les alignements identifiés comme étant les meilleurs par MSAMO.
Il y a dans ce qui suit un aperçu sur un échantillon des résultats obtenus après l'application de MSAMO sur des différents datasets protéiques et nucléiques.

§ **Les Séquences Protéiques**

§ **référence 1 :** Un nombre restreint des séquences approximativement équidistantes :

Ce test est destiné à tester l'effet de la taille et la similitude des séquences sur la performance des programmes d'alignement. Cet ensemble de test contient trois sous ensembles. V1 qui contient des séquences difficiles à aligner avec moins de 25% d'identité. V2 représente un ensemble de séquences avec un pourcentage d'identité variant entre 20 et 40%. Pour le dernier sous ensemble V3, le pourcentage d'identité est supérieur à 35 %.

§ Le Test V1

Les tables 5.5.A représente les scores obtenus par application des mesures de scores sur les alignements obtenus des différentes méthodes d'alignement multiple de séquences. L'alignement identifié par MSAMO comme étant le meilleur, est effectivement le meilleur par rapport à l'alignement de référence. La table 5.5.B présente les résultats obtenus après 500 générations.
L'alignement initial a été amélioré. Cette amélioration a produit par conséquent un nombre d'alignements (5), tous optimaux du point de vue MSAMO. Ces alignements composent donc le front Pareto (*FP*). Puisque la taille de FP est supérieure à 1, cela signifie que les fonctions objectif utilisées sont conflictuelles et non pas complémentaires.

DataS :1hfh_ref1	**T-coffee**	**Muscle**	**Align_m**	**ClustalX**
Rang	**2**	**1**	**4**	**3**
WSP	3589.16	**3627.8**	2016.1	3138.3
Entropie	135.9	**132.71**	189.97	137.18
Consensus	1795	**1811**	1009	1583
CS (Balibase)	0.839	**0.887**	0	0.758
SPS (Balibase)	0.902	**0.932**	0.270	0.886

Table 5.5.A

Sol1	**Sol 2**	**Sol 3**	**Sol 4**	**Sol 5**
1	**1**	**1**	**1**	**1**
3.6574	3.66811	3.6999	3.6999	**3.7097**
131.33	131.40	*132.67*	*132.95*	133.06
1828	1838	***1847***	***1847***	1852
0.92	**0.92**	**0.92**	**0.92**	**0.92**
0.951	**0.951**	**0.951**	**0.951**	**0.951**

Table 5.5.B

On remarque qu'il y a une amélioration au niveau de toutes les FOs ainsi que les scores CS et SPS. Les alignements 3 et 4 (sol 3 et sol 4) ont eu, en général, les mêmes scores. Sont-elles des clones d'une même solution ? Non car avec un peu d'observation (Figure 5.6), on remarque il y a du point de vue génotype une différence qui peut être significative point de vue biologique.

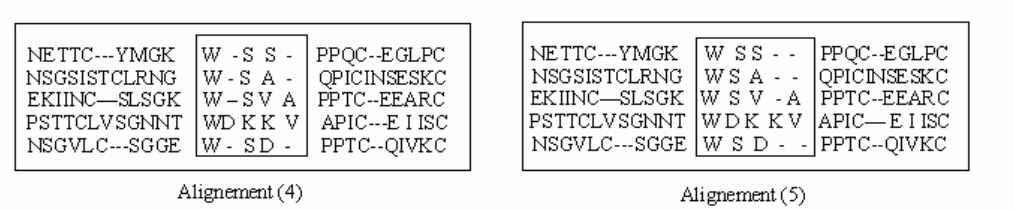


Figure 5.7: *Deux alignements ayant le même phénotype et pas le même génotype*

La figure 5.8.A montre le graphe du front Pareto de l'exemple cité. Dans ce graphe, nous avons présenté les fonctions WSP et Entropie. On peut y observer une frontière de Pareto distribuée. La fonction Consensus et WSP évoluent dans le même sens (Figure 5.8.B).

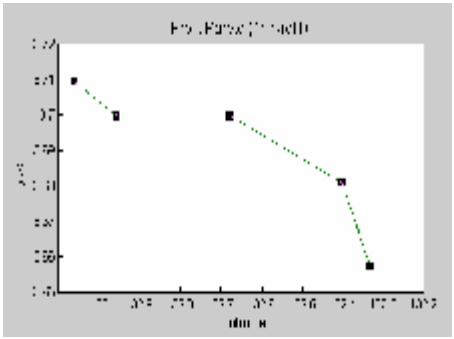


Figure 5.8.A : *Le front de Pareto obtenu pour WSP et Entropie*

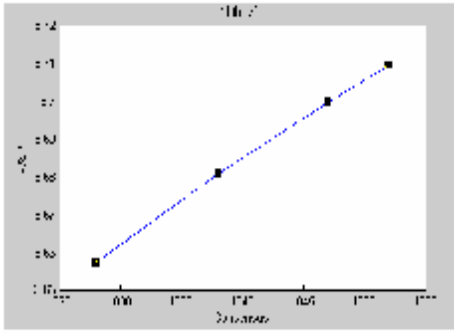


Figure 5.8.B : *WSP et Consensus sont complémentaires*

D'après les courbes représentées dans les figures 5.8.A et 5.8.B, nous pouvons déduire que les fonctions WSP et Consensus sont complémentaires, par contre WSP et Entropie sont conflictuelles, même remarque peut être faite pour le couple : Consensus et Entropie.

§ Le Test V2

Les résultats obtenus avec ce test sont moins bons que ceux du test précédent malgré que les séquences alignées aient un pourcentage d'identité variant entre 20 et 40 %, ce qui les rendait normalement plus faciles à manipuler que celles du test V1.

DataS :1sbp_ref1	**T-coffee**	**Muscle**	**Align_m**	**ClustalX**
Rang	**2**	**1**	**3**	**2**
WSP*10.e+02	1.3989	**1.9080**	-0.5336	0.3636
Entropie	353.03	**327.08**	344.28	335.06
Consensus	742	**956**	-263	209
CS (Balibase)	0.357	**0.671**	0.457	0.414
SPS (Balibase)	0.590	**0.771**	0.547	0.594

Table 5.6.A

Sol1	**Sol 2**	**Sol 3**	**Sol 4**	**Sol 5**
1	**1**	**1**	**1**	**1**
2.0117	1.9990	2.0294	1.9685	**2.1334**
325.94	325.78	326.21	**325.40**	331.06
1008	1001	1017	986	**1069**
0.671	0.671	0.671	0.600	**0.672**
0.771	0.767	0.771	0.743	**0.777**

Table 5.6.B

Cet exemple nous montre la capacité de notre algorithme à évaluer et identifier le meilleur alignement, puis le raffiner avec un T_{am} =0.17 mais il a échoué dans la construction d'un front Pareto uniforme et bien distribué, ceci est peut être dû au nombre restreint de solution non dominées qu'il a rencontré.

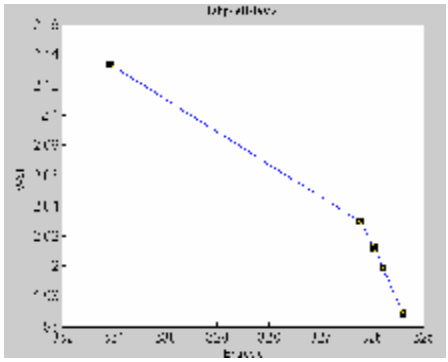


Figure 5.9 : *Front Pareto de 1sbp-ref1*

§ **Le Test V3**

Notre algorithme a rencontré une difficulté à améliorer les scores SPS et CS de cette classe de test. Les séquences de ce test peuvent atteindre des tailles allant jusqu'à 335-395 et leur nombre jusqu'à 14 séquences par dataset.

La table (5.7) contient quelques exemples d'exécution de MSAMO sur la référence Ref1 de Balibase. On constate que MSAMO parvient à améliorer les scores SPS et CS pour les datasets du test V1 de Ref1. Par contre ses résultats sont mitigés pour les autres tests.

Référence		**Data-set**	**T_Coffee**		**Muscle**		**Align_M**		**ClustalW**		**MSAMO**		**|FP|**
			SPS	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	
Ref1	**V1**	1hfh	0.90	0.83	0.93*	0.88*	0.2	0.0	0.88	0.75	**0.95**	**0.92**	6
		1cpt	0.96	0.91	0.97*	0.97*	0.97	0.96	0.93	0.87	**0.98**	**0.99**	6
	V2	1Sbp	0.59	0.35	0.77*	0.67*	0.45	0.54	0.59	0.41	**0.78**	0.67	6
		Kinase	0.80	0.66	0.90*	0.85*	0.87	0.78	0.79	0.63	0.90	0.85	5
	V3	1ajsA	0.51	0.25	0.51*	0.36*	0.11	0.08	0.57	0.27	0.51	0.36	6
		gal4	0.61	0.38	0.78	0.43	0.54	0.43	0.69*	0.54*	**0.70**	**0.55**	4

Table 5.7

Les valeurs en *gras* sont les scores améliorés.
Les valeurs suivies par des *astérisques* * sont les scores des alignements considérés meilleurs par MSAMO.
|*PF*| est la taille de l'ensemble final des solutions optimales obtenues à la fin de l'exécution de MSAMO.

§ **Référence 2** : Une famille de séquences divergentes avec des séquences orphelines

Dans ce test on examine la capacité du programme à aligner les séquences orphelines divergentes (10-20% d'identité entre la famille les orphelines) avec une famille de séquences très homologues (> 25% d'identité).

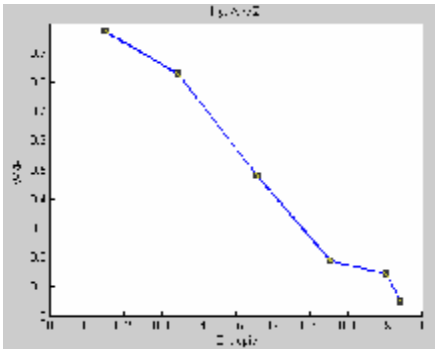


Figure 5.10

MSAMO a complètement échoué à faire évoluer les scores SPS et CS des datasets de ce test. Aucune amélioration n'a été observée sur les scores SPS et CS.

Référence	Data-set	T_Coffee		Muscle		Align_M		ClustalW		MSAMO		|FP|
		SPS	CS	SPS	CS	SPS	CS	SPS	CS	SPS	CS	
Ref2	1csy	0.89*	0.12*	0.86	0.40	0.87	0.0	0.86	0.12	0.89	0.12	6
	tgxA	0.76	0.17	0.84*	0.26*	0.69	0.0	0.89	0.08	0.84	0.26	5

Table 5.8

§ **Référence 3** : Familles des séquences liées

Ce test est conçu en vue de tester la capacité des programmes à aligner correctement des familles de séquences approximativement équidistantes (< 20% d'identité) composées de séquences fortement liées (> 25% d'identité) dans un alignement multiple.

Référence	Data-set	T_Coffee		Muscle		Align_M		ClustalW		MSAMO		|FP|
		SPS	CS	SPS	CS	SPS	CS	SPS	CS	SPS	CS	
Ref3	1idy	0.55	0.0	0.64*	0.36*	0.31	0.0	0.58	0.40	**0.73**	0.36	6
	1ubi	0.66	0.29	0.70*	0.51*	0.57	0.31	0.66	0.26	0.70	0.51	6

Table 5.9

La table 5.9 montre deux exemples d'exécution de MSAMO sur deux tests de la référence 3, une amélioration est observée sur le SPS du premier dataset (1idy) sans que le score CS ne soit modifié. Par contre le deuxième reste inchangé malgré que le fait |FP| = 6, indique que l'alignement a subit plusieurs modifications. Ces modifications ont donné naissance à un nombre d'alignements finals sans effet positif apparent.

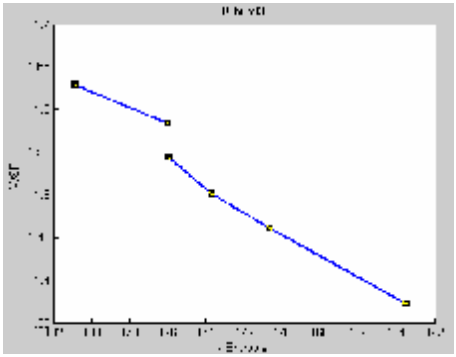


Figure 5.11

§ **Référence 4 :** *Prolongements des N/C-Terminal*

Ce test permet d'évaluer la qualité des programmes capables d'aligner des familles de séquences contenants des longs gaps et de taille inégale. Le but de ce test pour MSAMO est d'évaluer si le programme est capable de déplacer des blocs de noyaux entourés de longues insertions de gaps au début ou à la fin.

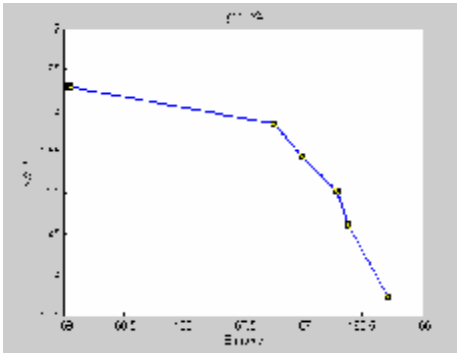


Figure 5.12

Dans ce test, MSAMO n'a pas pu améliorer les scores SPS et CS, en plus les longs gaps insérés dans les séquences, augmentent significativement la taille d'un alignement. Ceci a augmenté la complexité de l'algorithme et par conséquent la durée de son exécution.

Référence	**Data-set**	**T_Coffee**		**Muscle**		**Align_M**		**ClustalW**		**MSAMO**		**|FP|**
		SPS	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	
Ref4	1ycc	0.86	0.45	0.88*	0.48*	0.77	0.39	0.87*	0.48*	0.87	0.45	6
	2abk	0.66	0.0	0.66	0.0	0.47*	0.82*	0.66	0.0	0.47	0.82	6

Table 5.10

§ **Référence 5 :** *Insertion Interne*

Contrairement à la référence 4, dans ce test, les insertions de longs gaps sont internes. Les séquences sont toujours de taille inégale. L'inégalité des séquences va induire l'insertion des longs gaps ce qui va produire des alignements de taille très élevée.

DataS :1htm2_ref5	**T-coffee**	**Muscle**	**Align_m**	**ClustalX**
Rang	2	**1**	2	3
WSP*10e+03	2.4873	**2.7274**	2.4948	2.0530
Entropie	299.415	**294.582**	307.864	307.142
Consensus	12849	**13780**	12639	10609
CS (Balibase)	0.7650	**0.7650**	0.7650	0.4120
SPS (Balibase)	0.9190	**0.9570**	0.9100	0.7690

Table 5.11.A

Sol1	**Sol 2**	**Sol 3**	**Sol 4**
1	**1**	**1**	**1**
2.7289	2.7293	**2.7298**	2.7274
294.867	295.026	295.426	**294.582**
13788	13790	**13792**	13780
0.7650	0.7650	0.7650	0.7650
0.9570	0.9570	0.9570	0.9570

Table 5.11.B

MSAMO semble être incapable d'améliorer les scores SPS et CS pour ce test, cependant les valeurs des trois fonctions objectif ont évoluées. Cela signifie que les modifications apportées sont insuffisantes. La maximisation de la fonction WSP semble pousser la dégradation automatique de la fonction Entropie (table 5.11.B).

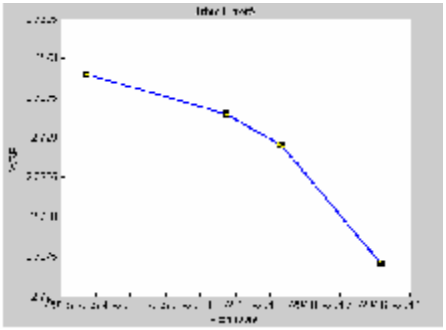


Figure 5.13

L'échec de MSAMO devant les séquences de la référence5 n'a pas empêché l'obtention d'un front Pareto bien distribué (figure 5.13).

Référence	**Data-set**	**T_Coffee**		**Muscle**		**Align_M**		**ClustalW**		**MSAMO**		**|FP|**
		SPS	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	**SPS**	**CS**	
Ref5	1thm1	0.93	0.77	0.93*	0.77*	0.92	0 .74	0.86	0.51	0.93	0.77	4
	1thm2	0.76	0.91	0.95*	0.76*	0.91	0.76	0.76	0.41	0.95	0.76	4
	2cba	0.80	0.55	0.83*	0.66*	0.76	0.46	0.79	0.55	0.83	0.66	5

Table 5.12

En conclusion nous pouvons dire que MSAMO a montré une grande capacité à identifier les meilleurs alignements protéiques. Sachant que parmi les alignements fournis par les méthodes d'alignement multiple utilisées, il y a des alignements qui n'optimisent aucune des fonctions objectif mais ils sont considérés les meilleurs par Bali_score. Ce type d'alignement est difficile à repérer voire impossible avec les fonctions utilisées. La table 5.13 décrit l'apport de l'utilisation de plusieurs fonctions objectif simultanément.

Les datasets	WSP	WSP + Entropie
ref1_v1:	40 %	83 %
ref1_v2:	60 %	95 %
ref1_v3:	20 %	57 %
ref2:	20 %	60 %
ref3:	14 %	72 %
ref4:	20 %	75 %
ref5:	34 %	100 %
Moyenne :	29 %	**77 %**

Table 5.13 : Les *taux identification* T_{id} *par nombre de Fonctions objectif pour les séquences protéiques*

Pour uniquement 500 générations, nous avons obtenus les pourcentages d'améliorations moyens suivants par classe de référence (table 5.14). Le taux ne dépassant pas les 10 %, indique une faible amélioration des scores pour les fonctions objectif, ceci est immédiatement traduit par un taux très faible aux niveaux des scores SPS et CS.

Datasets	ref1_v1	ref1_v2	ref1_v3	ref2	ref3	ref4	ref5
T_{am} (FOs)	7.77	2.72	0.33	0.42	1 .87	9.93	0.36
T_{am} (SPS,CS)	3.42	0.44	3.00	0.0	4.50	0.0	0.0

Table 5.14: Les *taux d'amélioration T_{am} par rapport aux FOs optimisées et SPS et CS (les séquences protéiques)*

Ces taux d'améliorations dépendent fortement de la nature des séquences alignées, les fonctions objectif utilisées et principalement du nombre d'itérations effectuées. Malheureusement, vu la complexité de l'algorithme et la taille des données, faire exécuter celui-ci pour un grand nombre d'itérations, rendra le temps de réponse très grand.

§ Les Séquences Nucléiques

Dans Bralibase, on trouve également cinq familles de références : II Introns, 5SrRNA, SRP, tRNA et U5. Elles se distinguent entre elles par le nombre et la taille des séquences à aligner ainsi que le degré d'homologie entre les séquences. Comme mesures de score de référence utilisées ici, il y a le SPS fourni par *Bali_score* comme est le cas pour les séquences protéiques. Le score SC est remplacé par le score SCI (Structure Conservation Index) [Gardner et autres, 05]. Cette dernière mesure permet d'indiquer le degré de conservation de l'information sur la structure secondaire des séquences ARN alignées. Ce score peut être déterminé par le programme *RNAz* (version 1.0). Si le score SCI est ≈ 1, cela signifie que les séquences sont bien alignées et permet de donner une bonne indication sur la structure secondaire des séquences alignées. Cependant un score SCI > 1 signifie l'existence d'une structure ARN commune entre les séquences. Dans ce qui suit, nous allons montrer quelques exemples de l'application de MSAMO sur les séquences nucléiques. On a réduit le nombre de fonctions objectif à optimiser à deux (WSP et Entropie), après avoir constaté que la fonction Consensus fournit rarement une information différente à celle fournie par WSP.

- **La Référence II Introns**

Dans cette classe, les séquences nucléiques sont de très faible homologie (<25% d'identité).

Dataset : aln1	**T-coffee**	**Muscle**	**ClustalX**
Rang	3	**1**	2
WSP	-651.3	**-92.24**	-92.84
Entropie	69.57	**58.56**	59.05
SPS	0.717	**0.720**	0.593
SCI	0.43	**0.57**	0.62

Table 5.15.A

Sol1	**Sol2**	**Sol3**	**Sol4**	**Sol5**
1	1	1	1	1
25.44	28.60	**34.01**	29.64	32.59
53.83	**55.07**	55.55	55.10	55.27
0.662	0.666	0.666	0.662	0.666
0.62	**0.62**	**0.62**	**0.62**	**0.62**

Table 5.15.B

La figure 5.14 montre le front Pareto de l'ensemble final des solutions optimales du dataset *Aln1* de II Introns, nous pouvons y constaté une distribution des solutions sur le front.

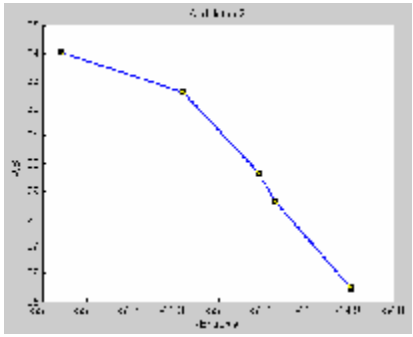


Figure 5.14

La référence 5SrRNA

MSAMO n'arrive pas à améliorer d'un cran le score SPS pour ce type de séquences, le score ICS par contre a progressé. Les alignements de cette classe de test semblent être les plus difficile à faire améliorer. Le front Pareto représenté (figure 5 .15.A) montre un niveau de complexité d'évaluation de ce type de données. Plusieurs alignements ont pratiquement le même score d'entropie mais la différence est grande entre les scores de WSP.

La Référence SRP

Les alignements de cette référence ont tous bénéficié d'amélioration, mais ces améliorations sont très de petite taille. Le nombre d'alignements contenus dans l'ensemble Pareto final est moyen mais leur dispersion sur le front est quasiment équidistante (Figure 5.15.B).

Référence	Dataset	T_Coffee		Muscle		Clustal		MSAMO		|FP|
		SPS	ICS	SPS	ICS	SPS	ICS	SPS	ICS	
II Introns	Aln1	0.71	0.43	0.72	0.57*	0.59	0.62	0.66	**0.62**	**5**
	Aln2	0.79	0.49	0 .82	0.57*	0.80	0.57	**0.84**	**0.70**	**1**
	Aln3	0.80	0.51	0.75	0.49*	0.69	0.47	**0.80**	0.49	**2**
5S rRNA	Aln1	0.94	0.81	0.94	0.66*	0.89	0.94	0.94	**0.89**	**4**
	Aln2	0.95	0.81	0.97	0.72	0.97	0.85*	0.97	**0.87**	**5**
	Aln3	0.94	0.78	0.94	0.82*	0.89	0.67	0.93	0.79	**1**
SRP	Aln1	0.76	0.41	0.87	0.68*	0.86	0.77	**0.88**	**0.70**	**5**
	Aln2	0.75	0.20	0.93	0.73*	0.91	0.70	**0.94**	0.73	**4**
	Aln3	0.84	0.60	0 .87	0.70	0.91	0.81*	**0.92**	**0.81**	**3**
tRNA	Aln1	0.37	0.00	0.56	0.45*	0.50	0.33	**0.62**	**0.48**	**3**
	Aln2	0.34	0.09	0.40	0.53*	0.44	0.48	**0.47**	**0.57**	**6**
	Aln3	0 .72	0.17	0.85	0.86*	0.85	0.88	**0.92**	**0.99**	**5**
U5	Aln1	0.73	0.03	0.55	0.48*	0.58	0.59	0.55	**0.49**	**5**
	Aln2	0.70	0.07	0.70	0.29*	0.70	0.65	**0.72**	**0.30**	**1**
	Aln3	0.21	0.00	0.78	0.39*	0.75	0.45	**0.80**	**0.44**	**1**

Table 5.16

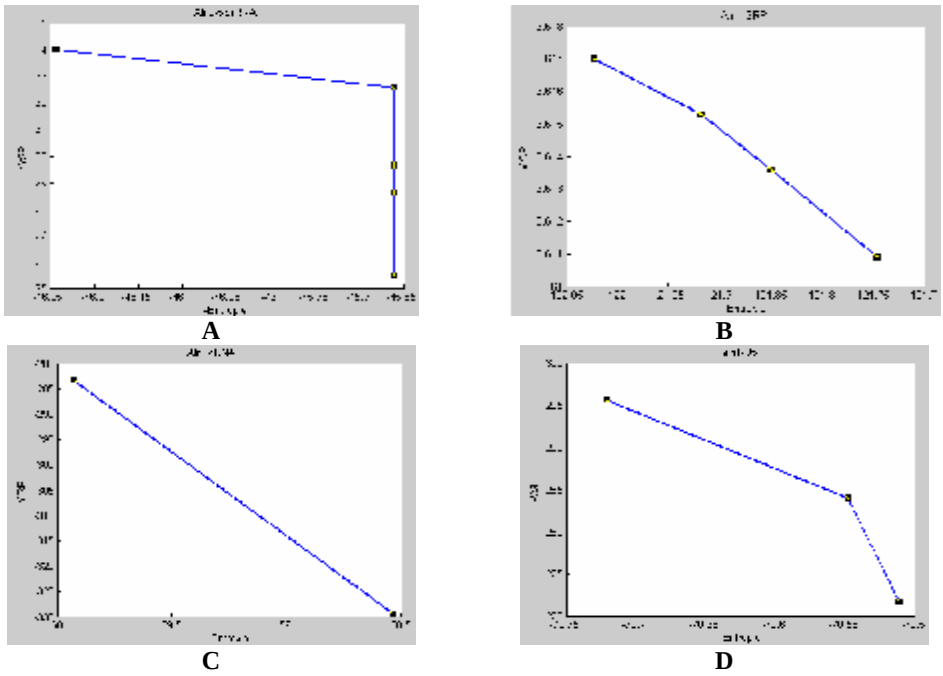
La Référence tRNA

Les résultats obtenus sur les datasets de la référence tRNA sont très intéressants. Nous avons observé des améliorations à tous les niveaux d'évaluation (fonctions objectif ou SPS et CS).
Le front Pareto représenté par la figure 5.15.C est réduit à deux points malgré que l'ensemble Pareto final possède 3 solutions (table 5.16). Deux solutions des trois ont les mêmes

caractéristiques phénotypes (elles sont confondues sur le graphe) mais elles sont différentes du point de vue génotype. Cette situation a été rencontrée plusieurs fois sur des tests différents. Ce Front Pareto ne fait que confirmer la remarque faite au début sur le fait que les fonctions WSP et Entropie sont conflictuelles.
Les résultats de la table 5.16 montre la capacité de MSAMO à améliorer les alignements de cette classe de référence avec un taux d'amélioration moyen égal à 10.73 %.

La référence U5

MSAMO s'est comporté avec les alignements de cette référence de la même manière qu'avec ceux de SRP. On n'a pu observer des améliorations sur tous les alignements mais elles sont minimes. L'ensemble final des solutions optimales est souvent réduit à un seul alignement optimal.



A **B**

C **D**

Figure 5.15

Les résultats de tous les tests effectués et les fronts de Pareto représentés confirment l'idée du fait que les fonctions objectif WSP et Entropie sont conflictuelles.
Pour les séquences nucléiques, nous avons constaté que MSAMO assure une meilleure performance par rapport celle montrée au niveau du raffinement des séquences protéiques. Les alignements nucléiques soumis à MSAMO ont pratiquement tous été améliorés (96%). Par contre au niveau de l'identification des meilleurs alignements, MSAMO s'est montré mois efficace avec une moyenne de 68% contre 78% pour les séquences protéiques (table 5.14 et table 5.19).

Datasets	WSP	WSP + Entropie
II Introns	30 %	67 %
5SrRNA	25 %	66 %

SRP	30 %	83 %
tRNA	15 %	75 %
U5	15 %	50 %
Moyenne :	23 %	**68 %**

Table 5.17 : Les *taux identification (T_{id}) par nombre de Fonctions objectif pour les séquences nucléiques (BRalibase)*

La table 5.18 est un récapitulatif des taux d'amélioration observés sur les datasets de BRalibase. Ayant obtenu un taux égal à 15 % pour les fonctions objectif et supérieur à 10 % pour les scores SPS et ICS, MSAMO se montre plus performant dans le raffinement des alignements nucléiques et la prédiction structurelle des RNAs.

Datasets	II Introns	5SrRNA	SRP	tRNA	U5	Moyenne
T_{am} (FOs)	**15**	5.6	2 .53	11.4	11.6	9.32
T_{am} (SPS,CS)	4.43	4.3	0.7	**10.7**	3.5	4.75

Table 5.18 : Les *taux identification T_{am} par nombre de Fonctions objectif pour les séquences protéiques*

Enfin, nous pouvons constater que MSAMO bénéficie de l'efficacité des méthodes utilisées pour générer les alignements initiaux, puis améliore la qualité de ceux-ci. Les figures 5.16.A et 5.16.B montre comment MSAMO avec un nombre réduit d'itérations parvient à améliorer la moyenne des scores SPS et ICS par rapport à l'ensemble des datasets de Bralibase.

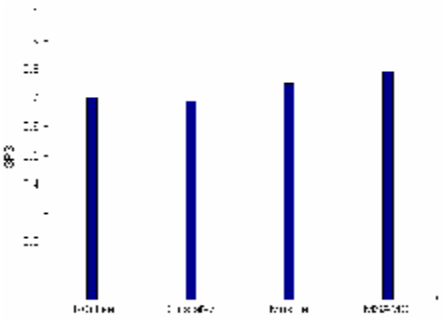


Figure 5.16.A

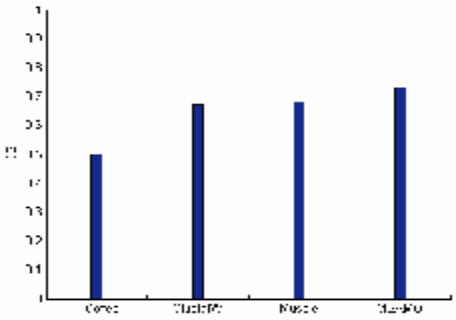


Figure 5.16.B

o Conclusion

Nous avons présenté dans ce chapitre, une approche évolutionnaire multi-objectif élitiste non agrégative permettant d'évaluer et de sélectionner les meilleures solutions possibles puis les raffiner par application d'opérateurs génétiques. La sélection utilisée est basée sur la notion de dominance de Pareto et utilise le concept du Ranking. Les solutions sont évaluées selon trois objectifs : WSP, Entropie et le Consensus. L'approche développée a été testée sur plusieurs datasets de Balibase (séquences protéiques) et BRalibase (séquences nucléiques).
Le problème d'espace mémoire ne d'est pas posé au cours de la validation ce qui n'est pas le cas du temps d'exécution. La procédure s'est avérée assez lente pour certains datasets (Ref4 et Ref5). Pour un nombre d'itérations réduit, les résultats obtenus sont des bons indicateurs sur la capacité de l'approche à identifier et raffiner des alignements de séquences. L'algorithme MSAMO a montré plus d'efficacité sur les séquences nucléiques. Les résultats obtenus sont meilleurs en comparant avec ceux fournis pour les séquences protéiques.

Les résultats obtenus sont très encourageants. Il serait intéressant d'étudier d'autres fonctions objectifs dans un contexte multi-objectif pour identifier l'ensemble le mieux adapté pour l'alignement des séquences.
Nous pensons que l'exécution de MSAMO sur un matériel plus performant et l'utilisation d'un langage compilé tel que le langage C au lieu de MATLAB, aurait donné des résultats nettement meilleurs.

Conclusion Générale

Dans le cadre de ce travail de magistère, nous avons traité un problème très important en bioinformatique celui de l'alignement multiple de séquences (MSA). Ce problème a toujours été défini comme un problème mono-objectif où les différentes méthodes développées cherchent à optimiser une seule fonction objectif. Nous avons démontré l'incapacité de ces méthodes à identifier et évaluer un alignement optimal, pour la simple raison que la fonction objectif utilisée sert à évaluer un seul aspect de l'alignement mais pas tous les aspects possibles.

Nous avons présenté dans ce mémoire, une approche évolutionnaire multi-objectif élitiste *MSAMO*, permettant d'évaluer et de sélectionner les meilleures solutions possibles puis les raffiner par l'application des opérateurs génétiques. La sélection utilisée est basée sur la notion de dominance de Pareto. Les solutions sont évaluées selon trois objectifs : WSP, Entropie et le Consensus.

Les résultats obtenus suite à l'application de notre algorithme sur des séquences protéiques et nucléiques sont très prometteurs. L'utilisation simultanée de plusieurs fonctions objectif au lieu d'une seule a permis de mieux identifier les bons alignements. Le raffinement des alignements est assuré par des opérateurs génétiques tels que le croisement et la mutation. Ces opérateurs ont permis l'obtention d'une multitude d'alignements différents. La technique de sélection utilisée favorise le classement des bons alignements au premier rang, ce qui a facilité leur identification. Le concept d'élitisme a permis à notre algorithme de préserver les meilleures solutions jusqu'à la fin de son exécution. La technique de réduction de l'archive par clustering, a autorisé l'obtention d'un front Pareto assez distribué.

MSAMO a été testé sur plusieurs datasets de Balibase et Bralibase, et il a montré sa capacité à améliorer la qualité des alignements et son efficacité dans l'évaluation et la sélection des meilleurs alignements. Toutefois notre algorithme assez complexe, a rencontré un problème temporel. Le grand nombre de séquences et leur taille l'ont rendu très lent. Ce qui nous a empêché de l'essayer pour un nombre d'itérations plus important.

Cependant, on peut dire que notre algorithme a montré une grande capacité dans la prédiction structurelle des ARNs.

Actuellement il n'existe pas des approches multi-objectif traitant le problème de *MSA*, il nous a été difficile de comparer *MSAMO* avec d'autres méthodes similaires. Enfin, on peut dire que notre approche a la capacité de déterminer l'alignement « *consensus* » parmi plusieurs alignements fournis par de différentes méthodes de MSA.

Les perspectives :

Exploiter cette approche pour comparer les fonctions objectif existantes et évaluer leurs capacités à estimer la qualité d'un MSA afin de déterminer quel est l'ensemble adéquat pour une meilleure identification des bons alignements

Étendre les tests pour d'autres méthodes d'alignements multiple existantes. Telles que ProbCons, MAFFT … etc.

Utiliser d'autres benchmarks tels que : PreFab, Sabmark, pour évaluer la qualité des résultats obtenus après raffinement.

Afin de pallier le problème de temps d'exécution, il serait intéressant d'introduire le concept « quantique», dont les travaux publiés ont montré la capacité des algorithmes génétiques quantiques à réduire significativement le temps d'exécution [Meshoul et autres, 05a] d'un côté et la capacité de ce concept à s'adapter aux problèmes de l'optimisation multi-objectif d'un autre côté [Meshoul et autres, 05b].

Bibliographie

[Alberts et autres, 02] : B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, "Molecular Biology of the Cell", Garland Science, 4ème édition 2002 à travers le site de NCBI : http://www.ncbi.nlm.nih.gov/books

[Altshul et autres, 89] : S.F. Altshul, R.J. Carroll and D.J. Lipman, "weights for data related by a tree", *J. Mol. Biol.* Vol. 207, pp. 647-653, 1989.

[Altschul et autres, 90] : S. F. Altschul, W. Gish, W. Miller, E.W. Myers, D. J. Lipman, " Basic Local Alignment Search Tool", *J. Mol. Biol.*, Vol. 215, pp. 403-410, 1990.

[Altschul et autres, 97] : S.F. Altschul,., T.L Madden, A.A. Schaffer, J. Zhang, Z. Zhang, Z. Miller, and D.J Lipman, "Gapped BLAST and PSIBLAST: a new generation of protein database search programs". *Nucleic Acids Res.*, Vol. 25, pp. 3389–3402, 1997.

[Bairoch et Apweiler, 00] : A Bairoch, R. Apweiler "The SWISS–PROT protein sequence database and its supplement TrEMBL in 2000". *Nucleic Acids Res.* Vol. 28, pp. 45–48, 2000.

[Barichard, 03] : V. Barichard, « Approches hybrides pour les problèmes multiobjectifs » Thèse de doctorat Informatique école Doctorale d'Angers, Novembre 2003.

[Barton et Sternberg, 87] : G. J. Barton, and M. J. Sternberg, "A strategy for the rapid multiple alignment of protein sequences. Confidence levels from tertiary structure comparisons". *J. Mol Biol*, Vol. 198, pp. 327-37, 1987.

[Basseur et autres, 02] M. Basseur, F. Seynhaeve, and E.-G. Talbi, « Design of multiobjective evolutionnary algorithm: application to the flowshop scheduling problem ». In Congress on *Evol. Compt.* (CEC'02), IEEE Press, pp. 1151–1156, 2002.

[Batzoglou, 04] : S. Batzoglou , "Sequence Alignment' I: CS262 Winter 2004: Lecture II, 2004.

[Baykasoglu et autres, 99] : A. Baykasoglu, S. Owen, N. Gindy, « A Taboo Search Based Approach to Find the Pareto Optimal Set in Multiple Objective Optimisation », Engineering Optimization, Vol. 31, pp. 731-748, 1999.

[Benlahrache et Meshoul, 07] N. Benlahrache et S. Meshoul, " Optimisation multi objectif pour l'alignement Multiple de Séquences" à apparaître dans le proceeding COSI'07 Oran. 2007.

[Bentley et Wakefield, 97]: P. J. Bentley1 and J. P. Wakefield,"Finding Acceptable Pareto-Optimal Solutions using Multiobjective Genetic Algorithms". 1997.

[Berman et autres, 00] HM. Berman, J. Westbrook, Z. Feng, G. Gilliland, TN. Bhat, H. Weissig, IN. Shindyalov, and PE. Bourne," The Protein Data Bank". *Nucleic Acids Res.* 28: pp 235–242, 2000.

[Bilofsky et Burks, 88] : H.S. Bilofsky and C.Burks, "GenBank: the genetic sequence data bank", *Nucleic Acids Res.* Vol. 16, No. 5, pp. 1861–1865, 1988.

[Brudno et autres, 03]: M. Brudno, C.B. Do, G.M. Cooper, M.F. Kim, E. Davydov, **"**Alignment of Genomic DNA LAGAN and Multi-LAGAN: Efficient Tools for Large-Scale Multiple**",** *Genome Res.* No. 13 pp. 721-731, 2003.

[Carrillo et Lipman, 88] : H. Carrillo and DJ. Lipman, « The multiple sequence alignment problem in biology », *SIAM J. Appl. Math*. Vol. 48, pp. 1073-1082, 1988.

[CoelloCoello, 99] : C.A. CoelloCoello, "A Comprehensive Survey of Evolutionary Based Multiobjective Optimization Techniques", Knowledge and Information Systems, Vol. 1, No. 3, pp. 269-308, 1999.

[CoelloCoello, 01]: C.A. CoelloCoello, “A Short Tutorial on Evolutionary Multiobjective Optimization”, *First International Conference on Evolutionary Multi-Criterion Optimization*, Springer-Verlag, Lecture Notes in Computer Science No. 1993, pp. 21-40. 2001.

[CoelloCoello et autres, 02] : C. A. CoelloCoello, D. A. Van Veldhuizen, and G. B. Lamont, “Evolutionary Algorithms for Solving Multi-Objective Problems”, *Kluwer Academic Publishers*, 2002.

[Collete et Siarry, 02] : Y. Collete et P. Siarry, « Optimisation multiobjectif » Eyrolles, édition 2002.

[Czyzak et Jaszkiewicz, 98] : Czyzak P., Jaszkiewicz A. “Pareto Simulated Annealing - a Metaheuristic for Multiple-Objective Combinatorial Optimization”, *Journal of Multi-Criteria Decision Analysis*, Vol. 7, No. 1, pp.34-47, 1998.

[Dayhoff et autres, 78] : M.O. Dayhoff, R.M. Schwartz and B.C. Orcutt, “A model of evolutionary change in proteins”, Atlas *Protein Seq. Struct.*, vol.6 pp. 345-362. 1978.

[De Jong, 75] : K.A. De Jong, “ An analysis of the behaviour of a class of genetic adaptive systems”. PhD thesis, University of Michigan, 1975.

[Deb, 99] : K. Deb, “Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Tests Problems”, Evol. Compt., Vol. 7, No. 3, pp. 205-230, 1999.

[Deb et autres, 00a] : K. Deb, A. Pratab, and T. Meyariban, “Constrained Test Problems for Multi-objective Evolutionary Optimization”, Proceedings of the 1st International Conference on Evolutionary Multi Criterion Optimization EMO 2001 Lecture Notes in Computer Science, Vol. 1993, Springer, pp. 284-298, 2001.

[Deb et autres, 00b] : K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm for multi-objective optimization: NSGA-II. In Proceedings of the Parallel Problem Solving from Nature VI (PPSN-VI), pp. 849-858, 2000.

[Deb, 01] : K. Deb, “Multi-Objective Optimization Using Evolutionary Algorithms”, Wiley, 2001.

[Deb et autres, 02] K.Deb, L.Thiele, M.Laumanns, E.Zitzler, “Scalable Multi-Objective Optimization Test Problems”, Proceedings of the 2002 Congress on Evol. Compt. CEC 2002, pp. 825-830, 2002.

[Do et autres 05] : C.B. Do, M. Mahabhshyam, M. Brudno, and S. Batzoglou, “ProbCons: Probabilistic consistency-based multiple sequence alignment”, *Genome res.* Vol. 15. pp. 330-340. 2005.

[Edgar, 04] : R.C Edgar, “MUSCLE: multiple sequence alignment with high accuracy and high throughput”, *Nucleic Acids Res.* Vol. 32, No. 5, pp. 1792-1797, 2004.

[Erickson et autres, 01]: Erickson M., Mayer A., Horn J., “The Niched Pareto Genetic Algorithm 2 Applied to the Design of Groundwater Remediation Systems”, Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization EMO 2001 Lecture Notes in Computer Science, Vol. 1993, Springer, pp. 681-695,2001.

[Feng et Doolittle, 87] : D.F. Feng and R.F Doolittle. “Progressive sequence alignment as a prerequisite to correct phylogenetic trees”. *J. Mol. Evol.*, Vol. 25, pp.351-360, 1987.

[Fogel, 00] : D. Fogel, “Evolutionary Computation: Toward a New Philosophy of Machine” Intelligence (second edition), IEEE Press, 2000.

[Fonseca et Fleming, 93] : C.M. Fonseca and P.J. Fleming, “Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization”, Proceedings of the Fifth International Conference on Genetic Algorithms San Mateo USA, pp. 416-423, 1993.

[Fonseca et Fleming, 98] : C.M. Fonseca and P.J. Fleming, "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms" – Part 1: A Unified Formulation, IEEE Transactions on Systems, Man and Cybernetics, Vol. 28, No. 1, pp. 26-37, 1998.

[Gandibleux et Freville, 00] : X. Gandibleux and A. Freville, « Tabu Search Based Procedure for Solving the 0-1 MultiObjective Knapsack Problem: The Two Objectives Case », Journal of Heuristics, Vol. 6, No. 3, pp. 361-383, 2000.

[Gardner et autres, 05] : P.P. Gardner, A. Wilm and S. Washietl, " A benchmark of multiple sequences alignement programs upon structural RNAs", *Nucleic Acids Res.;* Vol.33, No. 8, pp. 2433-2439. 2005.

[Goldberg, 89] : D.E. Goldberg, *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison-Wesley, 1989.

[Grefenstette, 92] : J.J. Grefenstette, « Genetic Algorithm for Changing Environnement", Ed. R. Manner and B. Maanderik, Parallel Problem Solving from Nature 2, Brussels, pp. 137-144, 1992.

[Hamm et Cameron, 86] : G.H. Hamm and G.N. Cameron," EMBL: the data Library", *Nucleic Acids Res.;* No.14, vol. 1, pp. 5-9. 1986.

[Hansen, 97] : M.P. Hansen, "Tabu Search for Multiobjective Optimization: MOTS", Technical Report Presented at 13th International Conference on MCDM, Technical University of Denmark, 1997.

[Henikoff et Henikoff, 92] : S. Henikoff and J. Henikoff. "Amino acid substitution matrices from protein blocks".Proceedings of the National Academy of Sciences, No. 89, pp. 915-919, 1992.

[Hertz et Klober, 00] : A. Hertz, D. Klober, "A Framework for the Description of Evolutionary Algorithms", European Journal of Operational Research, Vol. 126, No.1, pp.1-12.2000.

[Hofmann et autres, 99] : K. Hofmann, P. Bucher, L. Falquet and A. Bairoch," The PROSITE database, its status in 1999", *Nucleic Acids Res.;* Vol. 1, No.27, pp. 215-219. 1999.

[Holland, 75] : J.H. Holland, " Adaptation in natural and arti_cial systems", PhD thesis, University of Michigan Press, 1975.

[Horn et autres, 94] : J. Horn, N. Nafpliotis, D.E. Goldberg, "A Niched Pareto Genetic Algorithm for Multiobjective Optimization", Proceedings of the First IEEE Conference on Evol. Compt., IEEE World Congress on Computational Intelligence, Piscataway USA, pp. 82-87, 1994.

[Jourdan, 03] : L. Jourdan, " Métaheuristiques pour l'extraction des connaissances: Application à la génomique", thèse de Doctorat, Université des Sciences et Technologies de Lille UFR. No. 3368, 2003.

[Katoh et autres, 02] : K. Katoh, K. Misawa, K. Kuma and T. Miyata, "MAFFT : a novel method for rapid multiple sequence alignment based on fast Fourier transform", *Nucleic Acids Res.* Vol. 30 no. 14 pp. 3059-3066, 2002.

[Knowles et Corne, 00a] : J. Knowles and D. Corne., "Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy", *Evol. Compt.*, Vol. 8, No. 2, pp. 149-172, 2000.

[Knowles et Corne, 00b] : J. Knowles and D. Corne., "M-PAES A Memetic Algorithm for Multiobjective Optimization", Proceedings of the 2000 Congress on *Evol. Compt.* CEC 2000, pp. 325-332, 2000.

[Knowles, 01] : J. Knowles, "Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization", PhD Thesis submitted to the Department of Computer Science, University of Reading, UK, 2001.

[Knowles et Corne, 02] : J. Knowles and D. Corne, "On Metrics for Comparing Nondominated Sets", Proceedings of the 2002 Congress on *Evol. Comput.* CEC2002, Hawaii USA, IEEE Press, pp. 711-716, 2002.

[Knowles et Corne, 04] : J .Knowles and D.Corne." Memetic Algorithms for Multiobjective Optimization: Issues, Methods and Prospects", 2004.

[Lambert et autres, 03] : C. Lambert, J. V. Campenhout, X. DeBolle and E. Depiereux, "Review of Common Sequence Alignment Methods: Clues to Enhance Reliability", *Current Genomics*, vol. 4, pp. 131-146, 2003.

[Layeb, 05] : A. Layeb, « Approche quantique évolutionnaire pour l'alignement multiple de séquences en bioinformatique », mémoire de Magistère, Département d'Informatique, Université Mentouri Constantine, 2005.

[Lionnet et Croquette, 05] : T. Lionnet e V. Croquette, « Introduction à la Biologie Moléculaire » http://www.phys.ens.fr/~biolps/, 2005.

[Luscombe et autres. 01] : N.M. Luscombe, D. Greenbaum, and M. Gerstein, "What is bioinformatics? An introduction and overview ", Yearbook of Medical Informatics USA, pp. 83-100, 2001.

[McClure et autres, 94] : M.A. McClure, T. K. Vasi and W.M. Fitch, « Comparative analysis of multiple protien sequence alignment methods », Mol. Biol. Evol. Vol. 11 pp. 571-592. 1994.

autres, a : S .Meshoul, A. Layeb and M. Batouche, "A Quantum Evolutionary Algorithm for Effective Multiple Sequence Alignment", In Lecture Notes on Artificial Intelligence (LNAI) EPIA 2005, pp. 260-271, 2005.

Supprimé : [Meshoul et
Supprimé : 05
Supprimé :]

autres, b: S.Meshoul, K. Mahdi, and M.Batouche: A Quantum Inspired Evolutionary Framework for Multi-objective Optimization. In Lecture Notes on Artificial Intelligence (LNAI) EPIA 2005 pp.190-201, 2005.

Supprimé : [Meshoul et
Supprimé : 05
Supprimé :]

[Morgenstern et autres 98] : B.Morgenstern, K.Frech, A. Dress and T.Werner, "DIALIGN: Finding local similarities by multiple sequence alignment", *Bioinformatics*, Vol. 14, No. 3 pp. 290-294, 1998.

[Morse, 80] : J.N. Morse, "Reducing the size of the nondominated set: pruning by clustering", Computers and Operations Research, No. 7, pp. 55-66, 1980.

[Needleman et Wunsch, 70] : S. Needleman and C. Wunsch , "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *J. Mol. Biol.* No. 48, pp. 443-453, 1970.

[Nicholas et autres, 02] : H.B.Nicholas, A.J. Ropelewski and D.W. Deerfield, " Strategies for multiple sequence alignment ", *Biotechniques*, Vol. 32, No. 3 pp. 572-591, 2002.

[Notredame et Higgins, 96] : C. Notredame and D.G. Higgins," SAGA: Sequence alignment by genetic algorithm", *Nucleic Acids Res.* Vol. 24, No. 8 pp. 1515-1524, 1996.

[Notredame et autres, 98] : C. Notredame, L. Holm and D.G. Higgins, «Coffee: an objective function for multiple sequence alignments », *Bioinformatics*, Vol. 14, No. 5 pp. 407-422, 1998.

[Notredame, 02]: C. Notredame, "Recent progress in multiple sequence alignment: a survey", *Pharmacogenomics*, Vol. 3, No. 1, 2002.

[Notredame et autres, 00] : C. Notredame, D.G. Higgins and J. Heringa, «T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment », *J. Mol. Biol.* Vol. 302, pp. 205- 217, 2000.

[Pearson et Lipman, 88] W. R. Pearson and D. J. Lipman : "Improved tools for biological sequence comparison", Proceedings of the National Academy of Sciences, Vol. 85 , 2444-2448. 1988.

[Pei et autres, 03] : J. Pei, R. Sadreyev and N.V. Grishin, "PCMA: fast and accurate multiple sequence based profile consistency", *Bioinformatics.* Vol. 19, pp. 427-428, 2003.

[Rocha, 00] : E.P.C. Rocha « Analyse exploratoire des génomes bactériens » Thèse de doctorat, Université de Versailles, 2000.

[Rong et Hansen, 04] : Z. Rong and E.A. Hansen. "K-Group A* for Multiple Sequence Alignment with Quasi-Natural Gap Costs" 16th IEEE International Conference on Tools with Artificial Intelligence. Boca Raton, FL. November, 2004

[Reinert, 03] : K. Reinert, "Introduction to multiple Sequence Alignment ", Algorithmische *Bioinformatik*, WS, 03, 10, 2003.

[Schaffer, 85] : J.D. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, Genetic Algorithms and Their Applications", Proceedings of the First International Conference on Genetic Algorithms, pp. 93-100, 1985.

[Schwefel, 81] : H.P. Schwefel, "Numerical optimization of computer models", Wiley, Chichester, 1981.

[Saitou et Nei, 87] : N. Saitou, and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees". *Mol. Biol. Evol*., Vol. 4, pp. 406-425. 1987.

[Smith et Waterman, 81]: T. Smith and M. Waterman, "Identification of common molecular subsequence". *J. Mol. Biol* . Vol. 147, pp. 195-197. 1981.

[Sneath et Sokal, 73] : P.H.A. Sneath, and R.R. Sokal, "Numerical Taxonomy". Freeman, San Francisco. 1973.

[Srinivas et Deb, 94]: N. Srinivas and K. Deb," Multiobjective optimization using non dominated sorting in genetic algorithms", *Evol. Comput.* , Vol. 2, No.3 pp. 221-248, 1994.

[Stoye et autres, 97] : J. Stoye, V, Moulton, and A. W. Dress, « DCA, an efficient implementation of the divide and conquer approach to simultaneous multiple sequence alignment", Comput. Appl. Biosc., Vol. 13, No. 6, pp. 625-631, 1997.

[Suppapitnarm et autres, 00] : A.Suppapitnarm, A. Seffen, G.T. Parks and P.J. Clarkson, « A Simulated Annealing Algorithm for Multiobjective Optimisation », Engineering Optimization, Vol. 33, No. 1, pp. 59-85, 2000.

[Talbi, 00] : E-G. Talbi, « Une taxinomie des métaheuristiques hybrides », ROADEF'2000, 2000.

[Taylor, 87] : W. R Taylor, "Multiple sequence alignment by a pairwise algorithm". *Comput Appl* Biosci, Vol. 3, pp. 81-7. 1987

[Thompson et autres, 94] : J.D. Thompson, D.G. Higgins and T.J. Gibson, "CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice", *Nucleic Acids Res.* Vol. 22 No. 22 pp. 4673-4680, 1994.

[Thompson et autres, 99] : J.D. Thompson, F. Plewniak, and O. Poch, « BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs". *Bioinformatics*, Vol. 15, pp. 87–88, 1999.

[Thompson et autres, 05] : J. D. Thompson, P. Koehl , R. Ripp and O. Poch, "BAliBASE 3.0: Latest developments of the multiple sequence alignment benchmark", 2005.

[Ulungu et autres, 99] : E.L. Ulungu, J. Teghem, P.H. Fortemps, and D.Tuyttens, « MOSA Method: A Tool for Solving Multiobjective Combinatorial Optimization Problems », Journal of Multicriteria Decision Analysis, Vol. 8, pp. 221-236, 1999.

[Van Valdhuizen, 99] : D. Van Valdhuizen. « Multiobjective Evolutionary Algorithms : Classifications, Analyses, and nex Innovations. PhD thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[Van Valdhuizen et Lamont, 00] : D. Van Valdhuizen, G.B. Lamont, "Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art", Evol. Compt., Vol. 8, No. 2, pp. 125-147, 2000.

[Van Walle et autres, 04] : I. Van Walle, I. Lasters, and L. Wyns, "Align-m: a. new algorithm for multiple alignment of highly divergent sequences", Oxford University Press, 2004.

[Van Walle et autres, 05] : I. Van Walle, I. Lasters, and L. Wyns, "SABmark—a benchmark for sequence alignment that covers the entire known fold space". *Bioinformatics*, Vol. 21, pp. 1267–1268. 2005.

[Vert, 05] : J. P. Vert : « Introduction à la biologie moléculaire et à la bioinformatique » cours de Master Recherche M2, 2004/2005

[Wallace et autres, 06] : I. M. Wallace, O. O'Sullivan, D. G. Higgins and C. Notredame. « M-Coffee: combining multiple sequence alignment methods with T-Coffee", *Nucleic Acids Res.*, 2006, Vol. 34, No. 6, pp.1692–1699.

[Wang, et Jiang, 94] : L. Wang, and T. Jiang, "On the complexity of multiple sequence alignment". *J. Compt. Biol.*, Vol. 1, pp. 337–348, 1994.

[Zitzler, 99] : E. Zitzler. Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[Zitzler et Thiele, 99] : E. Zitzler and L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach", IEEE Transactions on Evol. Compt., Vol. 3, No. 4, pp. 257-271, 1999.

[Zitzler et autres, 00] : E. Zitzler, D. Kalyanmoy and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results", Evol. Compt., Vol. 8 No. 2, pp. 173-195, 2000.

[Zitzler et autres, 01] : E. Zitzler, M. Laumanns and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization", Proceedings of the EUROGEN 2001 - Evolutionary Methods for Design, Optimisation and Control with Applications to Industrial Problems, Barcelona Spain, 2001.

[Zitzler et autres, 03] : E. Zitzler, L. Thiele, M. Laumanns, C.M Fonseca, and V.G. Fonseca, "Performance assessment of multiobjective optimizers : an analysis and review ", IEEE Transactions on Evol. Compt., Vol. 7 No.2 pp. 618–630, 2003.