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 pour l'obtention du diplôme de Magistère en Informatique

Option : Information & Computation

Approche quantique évolutionnaire pour l'alignement multiple de séquences en bioinformatique

Présenté par :

LAYEB ABDESSLEM

Dirigé par :

Dr. S. MESHOUL

Soutenu le : / / 2005

Devant le jury d'examen composé de :

Pr. Z. BOUFAIDA	Professeur, Université de Constantine.	Président.
Pr. D. NAIMI	Professeur, Université de Constantine.	Examinateur.
Dr. D. SAIDOUNI	Maître de Conférences, Université de Constantine.	Examinateur.
Dr. S. MESHOUL	Maître de Conférences, Université de Constantine.	Rapporteur.
Dr. A. BENSLAMA	Maître de Conférences, Université de Constantine.	Invité.

Remerciements

D'abord, Je remercie Dieu le tout puissant.

Mes vifs remerciements vont à mon encadreur : Mme MESHOUL d'avoir accepté de m'encadrer, et d'avoir suivi mon travail. Sans oublier Mr BATOUCHE responsable de l'équipe "Vision et Infographie" pour m'avoir accueilli dans son équipe et donné les moyens de faire ce mémoire dans de très bonnes conditions.

Je remercie tous les membres du jury qui ont accepté d'évaluer ce modeste travail.

Je remercie également Mr Hichem TALBI et Mr Amer DRAA de leur aide et de leurs critiques qui m'ont beaucoup aidé pour la réalisation de ce mémoire.

Je remercie tout le personnel du laboratoire LIRE et de l'institut d'informatique de l'université de constantine.

Merci à tous les membres de la famille qui, par leur support et encouragement, m'ont permis de m'investir entièrement dans mes études.

Je tiens à remercier toutes les personnes qui, directement ou indirectement ont contribué à la réalisation de ce mémoire.

Liste des matières

Introduction générale

Au cours des années, les biologistes ont collecté une grande masse d'informations concernant les séquences nucléotides et protéiques. Cependant, le traitement et l'interprétation du contenu de ces séquences sont devenus très pénibles. Par exemple un biologiste doit comparer des centaines voir des milliers de séquences afin de tirer les caractéristiques et les rapports entre ces séquences. Cette tâche est d'autant plus difficile à accomplir si l'on doit tenir compte du fait que la taille d'une seule séquence peut atteindre des milliards de bases comme le cas du génome humain. Pour cela, l'introduction d'outils puissants et de méthodes efficaces pour l'analyse et l'interprétation des données biologiques est primordiale. Partant de ce besoin, la bioinformatique en tant que domaine scientifique a émergé. La bioinformatique est un domaine interdisciplinaire qui incorpore plusieurs domaines comme la biologie, la médecine, la physique, la chimie et l'informatique. Cette relation entre l'informatique et la biologie est ordinaire 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 être rendues accessibles. En second lieu, la nature des données est souvent telle qu'une méthode statistique, et par conséquent le calcul informatique est nécessaire. Ceci s'applique en particulier à l'information sur les plans de construction des protéines et l'organisation temporelle et spatiale de leur expression dans la cellule codée par l'ADN. Troisièmement, il y a une analogie forte entre une séquence d'ADN et un programme machine. En effet un ADN représente une machine de Turing et peut être donc utilisé pour la résolution des problèmes d'optimisation combinatoire [Adleman, 94].

On peut distinguer deux grands thèmes en bioinformatique : l'analyse de séquences et la bioinformatique structurelle. Cette dernière concerne l'étude des structures des séquences protéiques et nucléotides. Cependant, L'analyse de séquences constitue la grande partie de la bioinformatique. Ce domaine couvre plusieurs autres sous domaines comme la construction des arbres phylogénétiques, l'assemblage de fragments d'ADN, le réarrangement des génomes, l'alignement de séquences, etc. La plupart des problèmes en bioinformatiques ont été démontrés NP-difficile. Même un problème dont la complexité est polynomiale est difficile à résoudre du point de vue bioinformatique vu la taille colossale des données à traiter. L'enjeu est double, on doit trouver des solutions efficaces à ces problèmes et biologiquement acceptables.

Parmi les tâches les plus importantes en analyse de séquences: l'Alignement Multiple de Séquences qu'on dénote MSA pour (Multiple Sequence Alignment). Le MSA consiste à aligner plusieurs séquences homologues. Le but principal de l'alignement multiple est de montrer les rapports essentiels et les caractéristiques communes entre un ensemble de séquences de protéines ou nucléotides. Cette tâche est une plateforme essentielle pour

plusieurs autres tâches plus complexes comme la recherche de la séquence consensus, la prédiction structurelle et fonctionnelle des séquences nucléiques et protéiques, la construction de l'histoire évolutive des séquences, etc. Cependant trouver un alignement multiple a été démontré NP-difficile. En effet, le MSA est un problème d'optimisation caractérisé par une grande complexité temporelle et spatiale. Pour cela, plusieurs travaux ont été proposés et qu'on peut regrouper en quatre grandes classes. La première utilise une généralisation l'algorithme de Needleman [Needleman et al, 70] pour l'alignement de paires de séquence afin de créer un alignement multiple. Cette méthode donne des solutions optimales mais elle est très gourmande en ressources CPU et mémoire. Elle est donc impraticable dans le cas de plusieurs séquences. La deuxième classe contient les méthodes basées sur une approche progressive [Feng et al, 87]. Le principe est simple : l'alignement multiple est construit graduellement selon un ordre donné des séquences. On démarre par l'alignement de deux séquences puis on ajoute progressivement les autres séquences une par une à l'alignement précèdent. Ces méthodes sont simples, rapides et donnent généralement des alignements de bonnes qualités. Cependant, leur inconvénient majeur est le problème des minima locaux et elles peuvent donc donner des résultats erronés. Pour surmonter ce problème, Les méthodes itératives de la troisième classe paraissent prometteuses. L'idée consiste à l'amélioration itérative d'une solution initiale en appliquant une série de raffinements appropriés sur cette solution. Le processus est réitéré jusqu'à la satisfaction des conditions finales. On distingue deux types de méthodes itératives : les méthodes itératives stochastiques et les méthodes itératives déterministes. Le premier algorithme itératif stochastique proposé dans la littérature utilise un algorithme de recuit simulé [Kim et al, 94]. Cependant cet algorithme est très lent et ne convient pas pour construire un alignement ab- initio [Notredame, 02]. Plus tard, plusieurs autres algorithmes itératifs qui utilisent différentes stratégies comme les algorithmes génétiques AGs [Notredame et al, 96], les algorithmes évolutionnaires [Thomsen et al, 02], la recherche tabou [Riaz et al, 04], ont été proposés. Concernant les méthodes itératives déterministes, l'idée consiste à extraire itérativement une séquence de l'alignement multiple et la réaligner ensuite au reste des séquences. Le processus est réitéré jusqu'il n'a pas plus d'améliorations possibles. Malgré que ces méthodes donnent parfois des résultas meilleurs que les méthodes progressives, leur inconvénient majeur est le temps d'exécution élevé qui peut dépasser celui des méthodes exactes [Notredame,02]. La dernière classe de méthodes d'alignement multiple utilise des modèles probabilistes et statistiques pour effectuer un alignement multiple comme le modèle caché de Markov (HMM). Un modèle caché de Markov emploie des statistiques et des informations préalables sur un ensemble de séquences pour créer toutes les combinaisons possibles des identités, des disparités, et gaps afin de créer un alignement. Les méthodes basées HMMs exigent un grand nombre de séquences (en général 20-100) afin que le modèle fonctionne correctement [Durbin et al, 98].

Indépendamment de la bioinformatique, les algorithmes quantiques évolutionnaires AQEs [Han et al, 01] constituent un nouveau champ de recherche qui combine les algorithmes évolutionnaires classiques et les principes de l'informatique quantique. Ce dernier est un domaine récent en informatique qui essaie de donner des solutions à des problèmes non encore résolus par l'informatique classique. En effet, l'informatique quantique offre des capacités de traitement et de stockage exponentielles grâce à des principes de la mécanique

quantique tels que la superposition, l'enchevêtrement, l'interférence, etc. Cependant, l'utilisation de ce nouveau paradigme est dépendante de la réalisation pratique des machines quantiques. Mais cela n'a pas empêché les chercheurs de combiner les algorithmes classiques et les principes de l'informatique quantique afin de tirer profit des capacités de ce dernier. Contrairement aux algorithmes quantiques purs, les algorithmes inspirés du quantique ne nécessitent pas la présence des machines quantiques. Les algorithmes inspirés du quantique ont démontré leur efficacité dans plusieurs problèmes d'optimisation combinatoire.

Dans le contexte de ce travail de magister, on s'intéresse au problème de l'alignement multiple de séquences. Ce problème est fondamental car il est à la base de plusieurs autres tâches plus complexes en analyse de séquences comme souligné précédemment. L'objectif de ce travail est triple. Dans un premier lieu, il s'agit d'explorer le domaine de la bioinformatique en soulignant les besoins en matière de traitement et d'analyse de l'information biologique. Dans un second lieu, il s'agit de mener une étude investigatrice visant à synthétiser les travaux relatifs à l'alignement multiple de séquences. En fin, en troisième lieu, il est question de proposer une nouvelle approche pour appréhender ce problème. Nous avons adopté une approche quantique évolutionnaire dont le développement a conduit à la proposition de deux méthodes: QEAMSA et QUANTALIGN. Ces dernières reposent sur un noyau de base défini par une représentation quantique appropriée et une dynamique quantique évolutionnaire adaptée. Elles se distinguent des autres méthodes évolutionnaires par une taille de population réduite et un nombre raisonnable d'itérations pour trouver les meilleurs alignements grâce aux principes de l'informatique quantique tels que la superposition d'états, l'interférence et la mutation. Une autre caractéristique de ces méthodes est leur capacité d'optimiser n'importe quelle fonction objectif. Les études expérimentales intensives prouvent que le programme QUANTALIGN produit des alignements qui, hormis la préservation des caractéristiques fonctionnelles et structurelles, sont comparables voir meilleures que d'autres méthodes populaires de l'alignement multiple de séquences.

Par conséquent, ce mémoire est organisé en quatre chapitres. D'abord, le premier chapitre est une introduction au domaine de la bioinformatique et le problème d'alignement de paires de séquences. Le deuxième chapitre présente l'état de l'art relatif à alignement multiple de séquences. Le troisième chapitre propose une introduction à l'informatique quantique et aux algorithmes quantiques évolutionnaires. Finalement, Le dernier chapitre est consacré à l'approche proposée. Il présente son principe, la méthodologie de résolution sous-jacente ainsi que les résultats de l'évaluation de ses performances. Le manuscrit s'achève par une conclusion et perspectives.

Chapitre 1

Introduction à la bioinformatique

"Computers are to biology what mathematics is to physics."
— Harold Morowitz

1.1 Introduction

L'histoire du calcul dans la biologie remonte aux années 20 où les scientifiques pensaient déjà à établir des lois biologiques basées seulement sur l'analyse de données par l'induction. Cependant, seulement le développement des ordinateurs puissants, et la disponibilité des données expérimentales qui peuvent être aisément traitées par le calcul (par exemple, des séquences d'ADN ou d'acides aminés et les structures tridimensionnelles des protéines) ont rendu la bioinformatique un champ indépendant de la biologie moderne. Aujourd'hui, les applications pratiques de la bioinformatique sont facilement disponibles via le "World Wide Web", et sont employées couramment dans la recherche biologique et médicale. La bioinformatique est une branche de biologie se développe rapidement et qui est fortement interdisciplinaire (figure 1.1), en utilisant des techniques et des concepts de l'informatique, des statistiques, des mathématiques, de la chimie, de la biochimie, de la physique, et de la linguistique [Luscombe et al, 01]. La bioinformatique extrait et interprète la connaissance de l'analyse par ordinateur des données biologiques. Celles-ci peuvent comprendre l'information stockée dans le code génétique, mais également des résultats expérimentaux de diverses sources, des statistiques médicales, et la littérature scientifique. La recherche en bioinformatique inclut le développement de méthodes pour le stockage, la récupération, et l'analyse des données. Il existe abondamment d'applications pratiques dans différents secteurs de biologie et de médecine.

Les analyses en bioinformatique se concentrent sur trois types d'ensembles de données : les séquences de génome, les structures macromoléculaires, et la génomique fonctionnelle expérimentale (par exemple des données d'expression). Mais l'analyse bioinformatique est également appliquée à diverses autres données, par exemple les arbres phylogénétiques, les données de rapport à partir des voies métaboliques et les statistiques médicales. Une gamme étendue de techniques sont employées, y compris l'alignement primaire de séquences, l'alignement de structures tridimensionnelles de protéines, la construction d'arbres phylogénétiques, la prévision et la classification de la structure de protéines, la prévision de la structure d'ARN, la prévision de la fonction de protéines, et le groupement des données d'expression. Le développement algorithmique est une partie importante en bioinformatique où des techniques et des algorithmes ont été spécifiquement développés pour l'analyse des données biologiques par exemple, l'algorithme de programmation dynamique pour l'alignement de séquences [Needleman et al, 70].

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 le

composant bioinformatique. Le but des projets de séquençage, par exemple, n'est pas de corroborer ou réfuter une hypothèse, mais de fournir des données brutes pour l'analyse postérieure. Une fois que les données brutes sont disponibles, des hypothèses peuvent être formulées et évaluées *in silico*. De cette manière, les expériences informatiques peuvent répondre aux questions biologiques qui ne peuvent pas être abordées par des approches traditionnelles. Ceci a mené à la fondation de groupes de recherche spécialisés en bioinformatiques. Trois secteurs principaux on été crées à savoir l'organisation de la connaissance dans les bases de données, analyse des séquences, et la bioinformatique structurelle.

Dans ce chapitre on va présenter une petite introduction à la biologie moléculaire en citant les notions biologiques de base nécessaires pour un bioinformaticien. Ensuite, on va donner une définition à la bioinformatique ainsi que les champs d'application de la bioinformatiques. Dans le reste du chapitre on va mentionner les éléments de base de l'analyse bioinformatique des séquences.

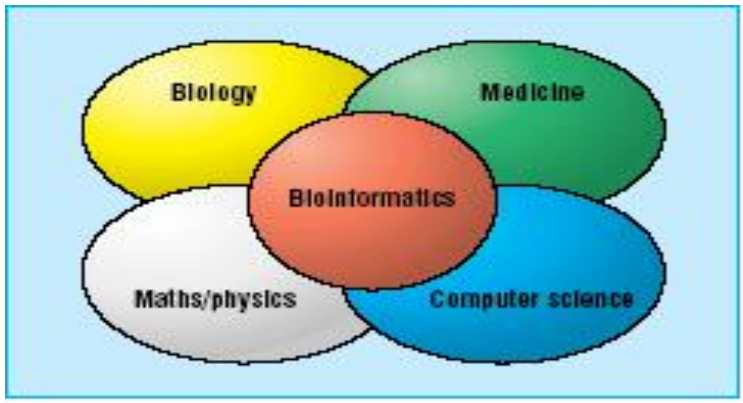


Figure 1.1. L'interaction des disciplines qui a contribuée à la formation de la bioinformatique

1.2 Notions de base en biologie moléculaire

La biologie a présenté plusieurs problèmes complexes et extrêmement durs. L'ADN et les protéines sont les molécules biologiques les plus importantes dans les êtres vivants. Afin de pouvoir donner des solutions informatiques efficaces aux problèmes biologiques, il faut bien avoir une certaine connaissance biologique. Pour cela on va présenter des notions en biologie amplement utilisées en bioinformatique. La plupart de ces notions sont obtenues à partir des références: [Setubal et al, 97], [Melcher, 99] et [Quinkal, 03].

A. Cellule

La cellule est la base de la vie. C'est une solution contenant différentes molécules (figure 1.2) entourées d'une membrane. Elle peut grossir et se diviser sauf les neurones. Elle a un métabolisme qui se résume en l'importation des nutriments et de les convertir en molécules et énergie. Une autre caractéristique est l'interaction avec son environnement. On peut distinguer deux organismes selon le nombre de cellules qu'ils contiennent : les multicellulaires dits eucaryotes et les unicellulaires dits procaryotes.

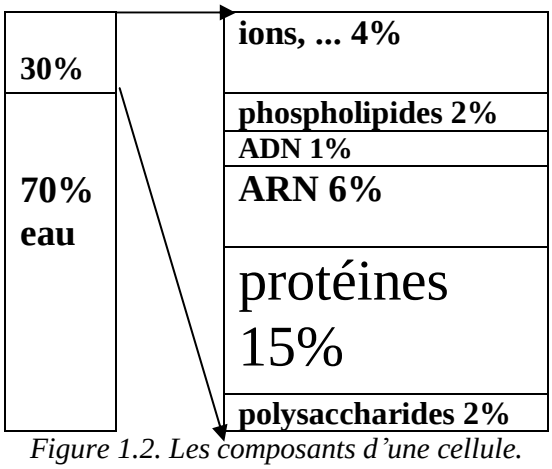


Figure 1.2. Les composants d'une cellule.

B. les acides nucléiques - composants de base

L'information génétique est un élément fondamental pour la compréhension du fonctionnement et l'histoire évolutive d'un être vivant. L'information génétique est stockée dans des polymères de structure relativement simple connus sous le nom d'acides nucléiques. Il en existe deux types, l'acide désoxyribonucléique ou ADN, et l'acide ribonucléique ou ARN. La chaîne principale de ces deux polymères est composée de sucres, le ribose pour l'ARN, et le désoxyribose pour l'ADN (figure 1.3), liés entre eux par des phosphates.

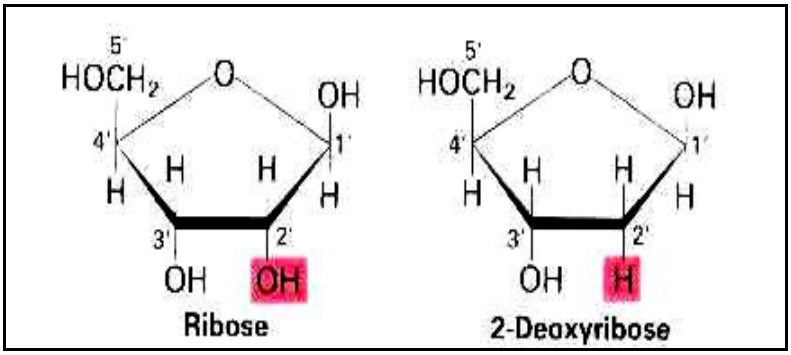


Figure 1. 3 .Structures du ribose et du désoxyribose.

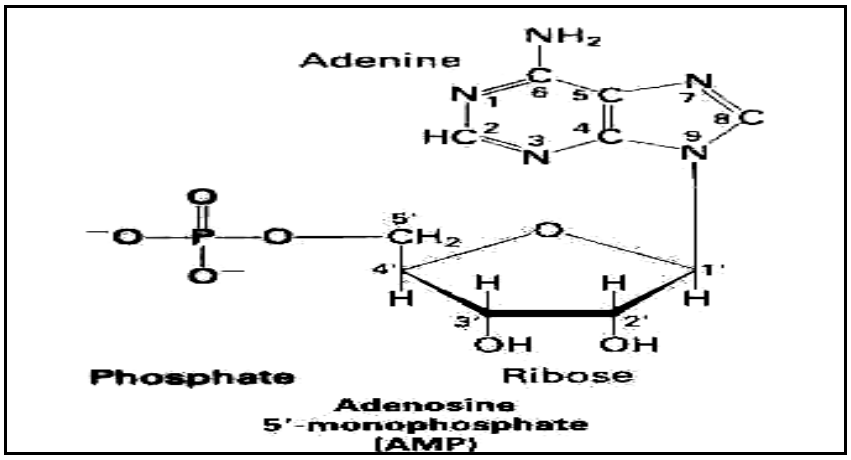


Figure 1.4. Structure de l'adénosine monophosphate, un exemple de nucléotide.

Des bases puriques ou pyrimidiques (figure 1.5) sont attachées au ribose et au désoxyribose. Elles sont attachées au carbone en position 1' (figure 1.4). Elles sont les éléments porteurs d'information de base dans un ADN ou ARN.

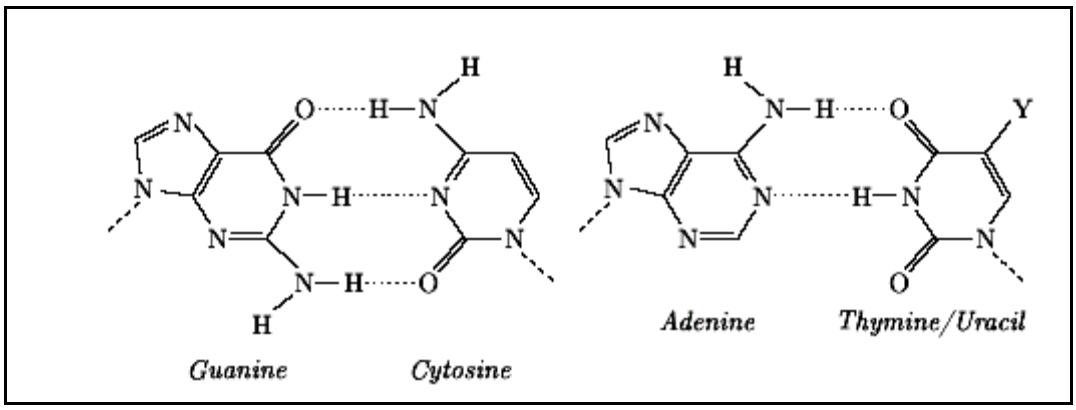


Figure 1.5 .Bases puriques et pyrimidiques. Les acides nucléiques: à droite, quand Y est un atome de hydrogène la base est uracile. Cependant, lorsque Y est un groupe CH_3 la base est Thymine.

L'ADN utilise l'adénine, la guanine, la thymine et la cytosine pour générer un code à quatre lettres (AGTC). Cependant, l'ARN contient de l'uracile au lieu de la thymine. L'uracile et thymine ont le même contenu d'information. L'ensemble du sucre (ribose ou désoxyribose), la base purique ou pyrimidique, et le groupement phosphate constituent un nucléotide, l'élément de base utilisé pour construire les acides nucléiques (figure 1.4).

C. L'ADN

L'ADN ou acide désoxyribonucléique est le matériel où est stockée l'information génétique et l'ensemble des caractères héréditaires d'une cellule. Sa structure moléculaire est similaire à une échelle constituée de quatre sortes de barreaux. Elle est représentée par une double chaîne de nucléotides de forme hélicoïdale (figure 1.6). L'ADN est constitué à partir de quatre sous unités (nucléotides) qui contiennent les bases azotées adénine (A), guanine (G), cytosine (C) et thymine (T). Les bases s'assemblent selon une complémentarité exclusive: A s'apparie uniquement avec T, et G avec C. Le rôle fondamental de l'ADN est de stocker l'information génétique. L'ADN peut être vu comme le matériel ("hardware") dépositaire des caractères héréditaires. C'est la mémoire du code génétique des êtres vivants. Le matériel génétique de l'ADN peut être reproduit par une opération de réplication. Il peut être traité en vue d'élaborer de nouvelle molécules nécessaires au métabolisme des cellules durant les opérations de transcription et de traduction. Il est représenté textuellement comme une séquence orienté de lettres : 5' AGGCTT… 3'.

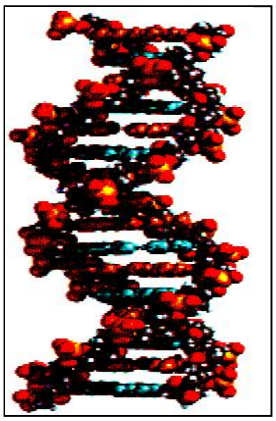


Figure 1.6. Structure spirale de l'ADN.

D. L'ARN

L'ARN possède une structure moléculaire similaire à l'ADN. Cependant, il se distingue par son rôle primordial de messager de l'information génétique. L'ARN transporte l'information entre l'ADN (dont il copie "en négatif" une séquence d'information) et les structures cellulaires. Ces dernières sont chargées de décoder la séquence d'information ramenée par l'ARN en vue de la production des protéines. Il est en quelque sorte le "logiciel" de la cellule. Il existe différents types d'ARN : ARN messager, ARN de transfert, ARN ribosomal. Ils ont tous un rôle particulier dans le processus complexe de synthèse des protéines.

E. Les protéines

Les protéines représentent le plus important et le plus grand composant de la cellule. Elles représentent environs 50% de la matière sèche de la cellule. Les protéines peuvent être classées en deux catégories : les protéines structurelles et les enzymes. Les protéines structurelles sont l'unité de base de la formation des blocks de tissus. Les enzymes joue le rôle d'un catalyseur des réactions chimiques. Sans les enzymes, les réactions biochimiques prennent un grand temps, ou même ne peuvent pas se déclencher. Les protéines sont des chaînes de simples molécules nommées acides aminées. Dans la nature on trouve 20 différents acides aminées. Chaque acide aminé a un carbone central nommé carbone alpha C_a. Un ensemble d'atomes et groupe d'atomes est attachés à C_a : les atomes de hydrogène, un groupe amine NH2, un groupe carboxyle COOH et une racine R . La racine distingue les acides aminés. La figure 1.7 montre la structure d'un acide aminé.

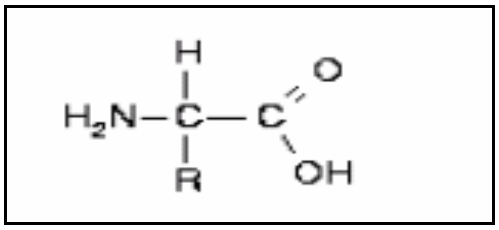


Figure 1. 7. La structure générale d'un acide aminé.

Dans les protéines, les acides aminés sont reliés par des liens peptides. Cela veut dire que l'atome de carbone dans le groupe carboxyle COOH appartenant à l'acide aminé A_i unit avec l'atome de nitrogène du groupe aminé NH2 de l'acide aminé A_{i+1}. Dans un tel lien la molécule de l'eau H2O est libérée. La figure 1.8 montre la formation de lien peptide entre deux acides aminés.

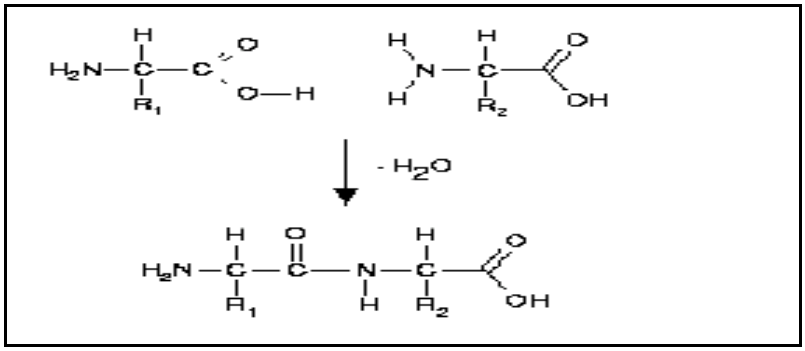


Figure 1.8. La formation de lien peptide entre deux acides aminés.

Pour cette raison les protéines ont un autre nom : les chaînes polypeptides. La répétition du bloc de base …- N-C_a - (CO)- … dans la chaîne donne à chaque protéine une épine dorsale "backbone". Généralement les protéines ont une chaîne de longueur (de 50 à 30000 acides aminés, la moyenne étant d'environ 400).Le premier acide aminé dans la chaîne polypeptide a un groupe aminé complet NH2 nommé *N-terminal end* .cependant le dernier acide aminé a un groupe carboxyle complet nommé *C-terminal end*. Chaque protéine a une orientation qui tourne par convention de *N-terminal end* a *C-terminal end*. Une protéine n'est pas une simple chaîne linéaire d'acides aminés. Elle se pli en une structure tridimensionnelle qui possède une grande influence sur les activités chimiques de la molécule. Les protéines possèdent plusieurs structures : primaire, secondaire, tertiaire et quaternaire.

Toute protéine existe sous une conformation définie par différents niveaux structuraux. La séquence des acides aminés liés dans la chaîne polypeptidique constitue la *structure primaire* de la protéine. Cette séquence est pleinement définie par l'ADN de la cellule. La *structure secondaire* est un premier niveau de repliement, adopté par des portions de la protéine, résultant d'interactions entre des acides aminés voisins sur la chaîne. Deux motifs de repliement caractéristiques peuvent ainsi se former : des hélices (dites hélices α) ou des feuillets (dits feuillets β), réunis par des boucles ou des demi-tours. La structure finale, c'est-à-dire l'organisation des éléments de la structure secondaire entre eux est appelée *structure tertiaire*. C'est la forme que prend la protéine dans l'espace. Elle est due à des interactions entre acides aminés éloignés sur la chaîne peptidique, qui se retrouvent proches dans cette structure tridimensionnelle. Certaines protéines, complexes, sont constituées de plusieurs sous-unités (plusieurs chaînes protéiques). *La structure quaternaire* est l'arrangement spatial de ces différentes unités, nécessaire à l'activité de l'ensemble.

F. La génétique moléculaire

Ø Gène et génome

Le gène est un segment d'ADN ou d'ARN. Il est situé à un endroit bien précis (locus) sur un chromosome. Chaque gène porte une information génétique et a éventuellement une fonction précise. Sa taille varie entre des centaines et un million de bases. On distingue trois types de gènes au niveau fonctionnel : les gènes protéiques, les gènes spécifiant des séquences ARN non traduits, et les gènes régulateurs. Par ailleurs, Le génome est l'ensemble du matériel génétique (patrimoine héréditaire) d'un individu ou d'une espèce. C'est une longue chaîne qui peut atteindre des milliards de bases (table 1.1). Il contient un ensemble de gènes qui est estimé entre 30000 et 35000. Le reste du génome constitue des segments non codants (figure 1.9).

Ø Le code génétique

Le code génétique est le système de correspondance entre message génétique et chaîne protéique. Chaque codon de trois bases nucléiques est traduit en un acide aminé (table 1.2). L'ARN messager transcrit de l'ADN a une fonction principale, celle de spécifier une protéine à synthétiser. Une grande partie des autres ARN cellulaires participent aussi à cette tâche, mais en tant que machines à décoder plutôt que de porteurs du code. Le code à déchiffrer

comporte quatre lettres, correspondant aux nucléotides (A, G, C, ou U), alors que sa traduction en compte 20, les acides aminés qui composent toutes les protéines. Le nombre minimum de nucléotides nécessaires pour spécifier un acide aminé est donc de trois, vu que deux lettres tirées d'un alphabet de quatre ne peuvent coder que 16 combinaisons différentes. Avec trois nucléotides, le nombre est de 64, ce qui permet d'incorporer une ponctuation, nécessaire à la bonne lecture du code, et implique une redondance, avec plusieurs triplets de nucléotides spécifiant le même acide aminé. Ces deux prédictions ont été vérifiées expérimentalement, et l'un des grands succès de la biologie moléculaire des années 60 fut de déchiffrer complètement le code génétique.

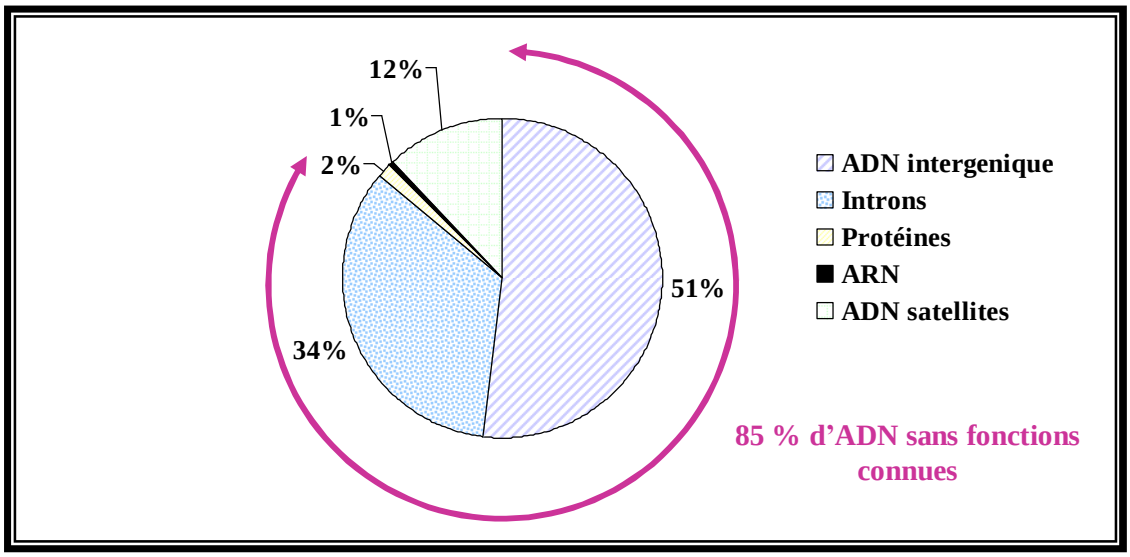


Figure 1.9. Proportion d'éléments fonctionnels dans le génome humain.

Organisme	Nb. chrom.	Nbre gènes	Taille Mb
Amoeba dubia	23	??	670 000
Fougère	23	??	160 000
Homo sapiens	23	30-40.000	3000
Mus musculus	21	30-40.000	3000
Riz	5	??	400
D. melanogaster	4	13600	165
Arabidopsis thaliana	5	26000	120
C. elegans	6	18.000	100
Saccharomyces cerevisiae	16	6000	13
Escherichia coli	1	4000	4,6
Encephalitozoon cuniculi	1	2000	2,9

Table 1.1. Taille de quelques génomes.

Pour que le code soit déchiffré correctement, il faut aussi désigner un cadre de lecture modulo 3. Ceci est accompli par l'introduction d'une méthionine (code AUG) en première position, dans un environnement favorable à l'attachement d'un ribosome. La première apparition du code AUG dans un ARNm indique normalement le cadre de lecture. Il est

souvent précédé de plusieurs codons "stop" (UAA, UAG, UGA) pour éviter toute ambiguïté. Il est clair que des mutations qui introduisent ou enlèvent 1 ou 2 nucléotides de la partie codante d'un gène vont décaler le cadre de lecture, et donc provoquer la synthèse de protéines non fonctionnelles. Ceci est une cause fréquente de mutations naturelles.

	U	C	A	G	
U	UUU Phe [F]	UCU Ser [S]	UAU Tyr [Y]	UGU Cys [C]	U
	UUC Phe [F]	UCC Ser [S]	UAC Tyr [Y]	UGC Cys [C]	C
	UUA Leu [L]	UCA Ser [S]	UAA *STOP*	UGA *STOP*	A
	UUG Leu [L]	UCG Ser [S]	UAG *STOP*	UGG Trp [W]	G
C	CUU Leu [L]	CCU Pro [P]	CAU His [H]	CGU Arg [R]	U
	CUC Leu [L]	CCC Pro [P]	CAC His [H]	CGC Arg [R]	C
	CUA Leu [L]	CCA Pro [P]	CAA Gln [Q]	CGA Arg [R]	A
	CUG Leu [L]	CCG Pro [P]	CAG Gln [Q]	CGG Arg [R]	G
A	AUU Ile [I]	ACU Thr [T]	AAU Asn [N]	AGU Ser [S]	U
	AUC Ile [I]	ACC Thr [T]	AAC Asn [N]	AGC Ser [S]	C
	AUA Ile [I]	ACA Thr [T]	AAA Lys [K]	AGA Arg [R]	A
	AUG Met [M]	ACG Thr [T]	AAG Lys [K]	AGG Arg [R]	G
G	GUU Val [V]	GCU Ala [A]	GAU Asp [D]	GGU Gly [G]	U
	GUC Val [V]	GCC Ala [A]	GAC Asp [D]	GGC Gly [G]	C
	GUA Val [V]	GCA Ala [A]	GAA Glu [E]	GGA Gly [G]	A
	GUG Val [V]	GCG Ala [A]	GAG Glu [E]	GGG Gly [G]	G

Table 1.2. Le code génétique

Ø Le dogme central

Le "dogme central", introduit par Francis Crick à la fin des années 50, postule que dans tous les êtres vivants l'information ne soit transmise que dans un sens: de l'ADN, où repose l'information, à l'ARN, une structure transitoire permettant sa transmission à une machine de traduction, aux protéines, les constituants de base qui font fonctionner les cellules et l'organisme entier. La figure 1.10 ci dessus montre les trois processus de base du dogme central : processus de réplication de l'ADN, de transcription de l'ADN en ARN messager, et de traduction de l'ARN messager en protéine. Le premier postulat du dogme central, largement vérifié, est la capacité d'autoréplication de l'ADN. Ceci avait été immédiatement déduit par Watson et Crick du fait que chaque brin d'ADN spécifie univoquement son complément. La transcription permet d'obtenir une séquence d'ARN à partir de l'ADN. Elle consiste à produire une séquence complémentaire à un brin d'ADN en notant que le complément de la base A (adénine) est la base U (uracile). Finalement, la traduction permet d'obtenir une protéine à partir de la séquence d'ARNm. Elle consiste à traduire un codon de trois bases à un acide aminé suivant la table 1.2.

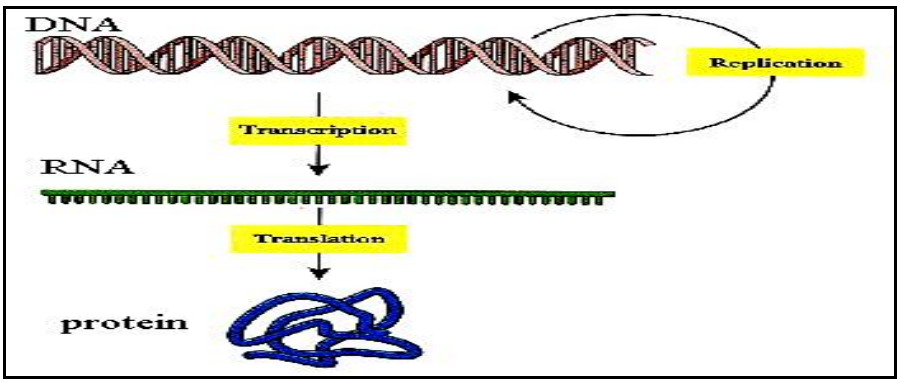


Figure 1.10. Le dogme central.

Ø Mutation, insertion, délétion, duplication

La mutation peut être définie de plusieurs manières. Les modifications de chromosomes sont nommées *mutation chromosomique* (aberration chromosomique). Les mutations incluent les changements dans le nombre total de chromosome, la délétion ou la duplication de gènes ou segment d'un chromosome, et le réarrangement du matériel génétique intra-chromosome ou inter-chromosomes.

A l'échelle moléculaire, une mutation peut être définie comme des changements dans la séquence nucléotide. Une telle mutation est appelée *mutation de gènes.* Elle peut être causée par la substitution d'un nucléotide par un autre ou par *insertion* ou *délétion* dans un ou plus de nucléotides dans un ADN (figure 1.11).

A l'exception du cas de la mutation adaptative qui est le résultat de l'interaction avec un autre organisme, toutes les autres mutations sont ou bien des mutations spontanées ou induites. Il existe quelques chevauchements entre les deux catégories. Les mutations spontanées se produisent dans la nature librement sans l'intervention d'aucun agent. Généralement, elles sont considérées comme des changements aléatoires dans la séquence de nucléotides. La plupart des mutations spontanées apparaissent pendant le processus d'enzymatique de la réplication de l'ADN. Contrairement aux mutations spontanées, les mutations induites sont le résultat de l'influence de n'importe quel agent artificiel. Ces agents peuvent être des radiations ou un large spectre d'un agent chimique.

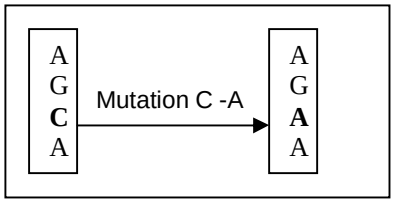


Figure 1.11.Une mutation de gènes.

1.3 Définition et thèmes en bioinformatique

La bioinformatique est constituée par l'ensemble des concepts et des techniques nécessaires à l'interprétation de l'information génétique (séquences) et structurale (repliement 3-D). Vu que la bioinformatique est en pleine évolution, la définition même de la bioinformatique est toujours la matière d'une certaine discussion. En anglais il existe deux termes qui réfère à cette discipline: " Computational biology" et " bioinformatics ". Le premier désigne la partie scientifique du domaine et le deuxième désigne la partie infrastructure [Wilhelm, 00]. Dans d'autres langues comme le français et l'allemand on utilise simplement la traduction du mot "bioinformatics " (bioinformatique en français) pour designer les deux parties de ce domaine.

Définition 1.1: La bioinformatique est le traitement automatique de l'information biologique. Plus spécifiquement, elle effectue la synthèse des données disponibles à l'aide de modèles et de théories, énonce des hypothèses généralisatrices par exemple comment les protéines se replient-elles ou comment les espèces évoluent-elles, et formule des prédictions par exemple localiser ou prédire la fonction d'un gène [Pedersen ,00].

De ce fait le rôle d'un bioinformaticien dépasse le rôle de l'utilisation classique de l'informatique pour l'acquisition de données à la contribution de l'interprétation de

l'information biologique (figure 1.12). Par exemple la prédiction fonctionnelle ou structurelle. La bioinformatique s'applique à tout type de données biologiques: Les séquences d'ADN et de protéines, les structures d'ARN et de protéines, les contenus en gènes des génomes, les réseaux d'interactions entre protéines, Les réseaux métaboliques, les arbres de phylogénie,etc. Le but de la bioinformatique est de faire avancer les connaissances en biologie, en génétique humaine, en théorie de l'évolution, etc. En plus, la bioinformatique est très utile pour la compréhension des maladies complexes [Jourdon, 03] et donc aider à la conception de médicaments.

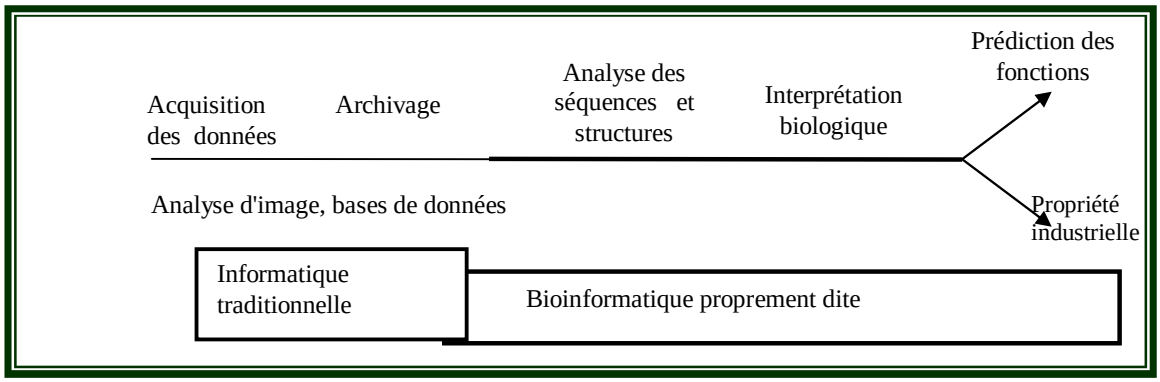


Figure 1.12. La position de la bioinformatique par rapport à l'informatique classique

1.3.1 Chronologie du développement de la bioinformatique

L'histoire de la bioinformatique peut être résumée par la chronologie suivante:

1951: Première séquence protéique (Insuline, Sanger).
1960: Lien entre séquence & structure (Globines, Perutz).
1965: La divergence et la convergence évolutionnaire dans les protéines (Zuckerkandl et al).
1967: La construction des arbres phylogénétiques Fitch & Margoliash.
1970: programme d'alignement global de deux sequences (Needleman & Wunsch).
1971: Premier travaux sur le repliement des ARNs (J. Ninio).
1973: "Génie Génétique" (Cohen et al.).
1974: "Prediction of Protein Conformation" (Chou & Fasman).
1977: Séquençage d'ADN (Sanger, Maxam, Gilbert).
1977: Premier "package" Bioinformatique (Staden).
1978: Bases de données: ACNUC, PIR, EMBL, GenBank.
1981: Los Alamos-GenBank: 270 séquences, 370.000 nucléotides.
1981: Programme d'alignement local (Smith-Waterman).
1985: Programme "Fasta" (Pearson-Lipman).
1990: Programme "Blast" (Altschul *et al.*).
1990: Clonage positionnel et séquençage de NF-1.
1991: "Grail", programme performant pour localiser les gènes (Mural *et al.*).
1991: Étiquettes d'ADNc "EST" (Venter et al., Matsubara et al.).
1992: Séquençage complet du chromosome III de levure.
1995: Première séquence complète d'un micro-organisme (Venter et al.; H. influenza).
1996: Séquence complète de la levure (consortium européen).
1997: Programme "Gapped Blast" (Alschul *et al.*).
1997: 11 génomes bactériens disponibles.
1998: Séquençage du 1er organisme pluricellulaire, Caenorhabditis elegans (100 Mb).

2000 : Séquençage du 1er génome de plante, Arabidopsis thaliana.
2001: Séquençage ("premier jet") complète du génome humain.

1.3.2 Les grands thèmes de la bioinformatique

De toute évidence, l'informatique est devenue un apport fondamental à la biologie moléculaire. En plus de l'utilisation classique des moyens informatiques pour le stockage ou la gestion des données, elle contribue également à l'interprétation de ces données. En absence d'expériences, le traitement informatique de séquences peut par exemple déceler la fonction biologique potentielle d'un gène par la recherche de critères spécifiques (signaux, structures secondaires ou tertiaires...) ou par la recherche de similitudes entre séquences. Les trois grands thèmes constituant les grands défis de la bioinformatique sont [Luscombe et al, 01]:

- **l'analyse, la compréhension et l'organisation d'une masse de données biologiques:**
  - Plus de 169 génomes complètement séquencés et publiés, dont l'homme (23 paires de chros.) et la souris (20 paires de chros.)
  - Projet du séquençage du génome humain.
  - Projets de séquençage de plus de 400 procaryotes et 360 eucaryotes
- **Décoder l'information contenue dans les séquences d'ADN et de protéines:**
  - Trouver les gènes.
  - Différencier entre introns et exons.
  - Analyser les répétitions dans l'ADN.
  - Identifier les sites des facteurs de transcription.
  - Étudier l'évolution des génomes.
- **Génomique structurale et fonctionnelle:**
  - Modéliser les structures 3D des protéines et des ARN structurels.
  - Déterminer la relation entre structure et fonction.
  - Étudier la régulation des gènes.
  - Déterminer les réseaux d'interaction entre les protéines.

Cet apport informatique à la biologie moléculaire concerne principalement quatre aspects:

- **L'acquisition et le stockage des données**

Concerne l'acquisition et le stockage des données biologiques. Généralement, les informations biologiques sont stockées dans une banque de données spécifique. Les banques sont généralement construites autour de thèmes précis comme l'ensemble des séquences d'une même espèce ou les facteurs de transcription. Incontestablement, toutes ces banques de données constituent une source de connaissance d'une grande richesse (figure 1.13) que l'on peut exploiter dans le développement de méthodes d'analyse ou de prédiction.

- **Traitements systématiques des séquences**

Concerne le traitement et l'analyse des séquences biologiques afin de repérer ou de caractériser une fonctionnalité ou un élément biologique intéressant comme la recherche des gènes codants, la détection des introns…etc.

- **Elaboration de stratégies**

L'objectif est d'apporter des connaissances biologiques ou de nouvelles stratégies supplémentaires que l'on pourra ensuite intégrer dans des traitements standard. On peut donner comme exemples la mise au point de nouvelles matrices de substitution des acides aminés ou la détermination de critères spécifiques dans la définition de séquences régulatrices.

Ø Evaluation des différentes approches dans le but de les valider

Concerne la validation des différentes approches existantes et déduire si possible les meilleurs pour un problème donné. Très souvent, tous ces aspects se confondent ou sont étroitement imbriqués pour donner naissance à un ensemble d'outils, d'études ou de méthodes qui convergent vers un but commun que l'on appelle l'analyse informatique des séquences.

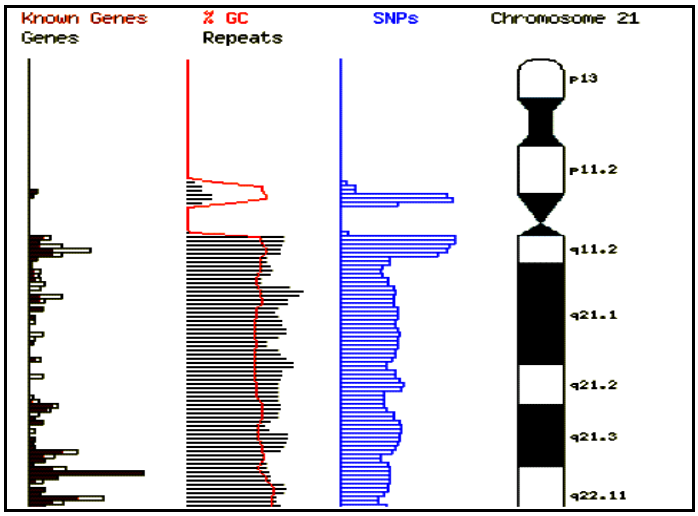


Figure 1.13. Une partie de la représentation du chromosome 21 (de la base Ensembl).

1.3.3 L'analyse de séquences

Les séquences de caractères sont parmi les porteurs primaires d'information dans notre société. Lisant et comparant l'information contenue dans les fragments de vieux textuel des sources aident les historiens à reconstruire l'histoire des objets et des événements. Pour un biologiste intéressé à l'histoire de la vie, les fragments des séquences biologiques décrivant le matériel génétique du fonctionnement d'organes ont plus ou moins le même but des fragments de vieux textes à un historien. Le matériel génétique évolue par des petits changements discrets, appelés des mutations ou des événements évolutionnaires. Vu que le matériel génétique est stocké dans des séquences d'ADN et reflété dans des séquences d'ARN et de protéines, il est utile de comparer deux séquences ou plus pour déterminer si elles sont par exemple évoluées de la même séquence héréditaire. Cela peut aider à impliquer la connaissance au sujet de la parenté des séquences et peut-être utile pour reconstruire une partie de leur histoire évolutionnaire commune. Pour un historien qui doit creuser dans des milliers de pages de texte dans l'ordre pour établir les circonstances d'un événement historique, le grand défi d'un biologiste est de lire et interpréter des milliards de bases du génome afin de bien connaître le fonctionnement et l'histoire évolutive d'un être vivant. Une

fois qu'un génome est complètement séquencé, il peut être sujet à plusieurs analyses (figure 1.14). Ayant pour buts par exemple :

- Identifier les gènes.
- Déterminer la fonction d'un nouveau gène. Une méthode pour présumer la fonction est de trouver un autre gène (probablement d'un autre organisme) dont la fonction est connue et que le nouveau gène a une grande similitude au niveau des séquences. Ceci suppose que la similitude de séquences implique la similitude fonctionnelle (pas toujours vrai).
- Identifier les protéines impliquées dans le règlement de l'expression de gène.
- Identifier les répétitions dans les séquences.
- Identifier d'autres régions fonctionnelles, par exemple les origines de la réplique (sites auxquelles polymérase d'ADN les débuts et les fins de la réplique) des pseudogenes (les séquences qui ressemblent aux gènes mais sont non exprimés.

Plusieurs de ces tâches sont de nature calculatoire. Etant donné la cadence incroyable à laquelle les données de séquences sont produites (figure 1.15), l'intégration de l'informatique, les mathématiques et la biologie est indispensable à l'analyse de ces séquences [Mount, 01].

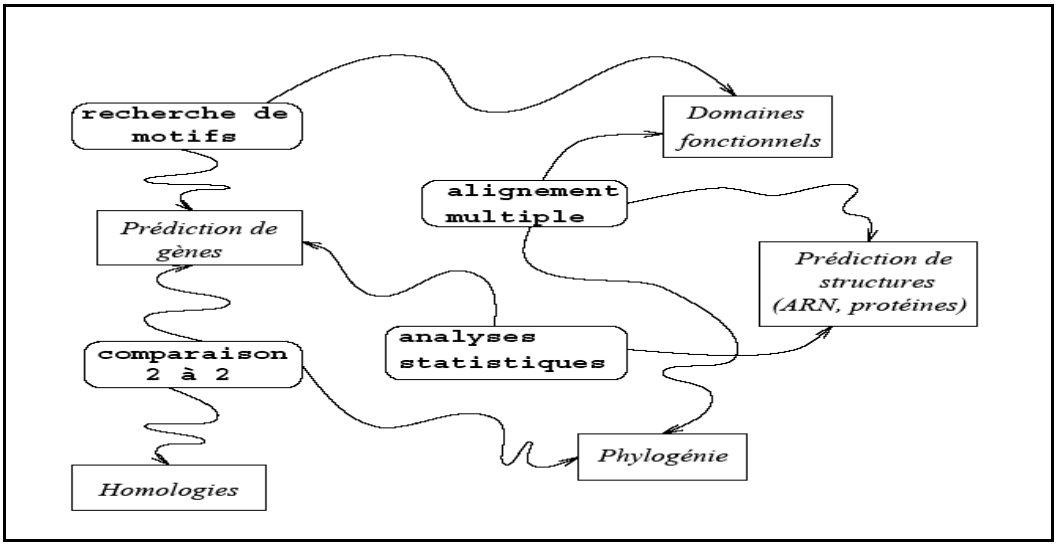


Figure : 1 .14. L'analyse des séquences.

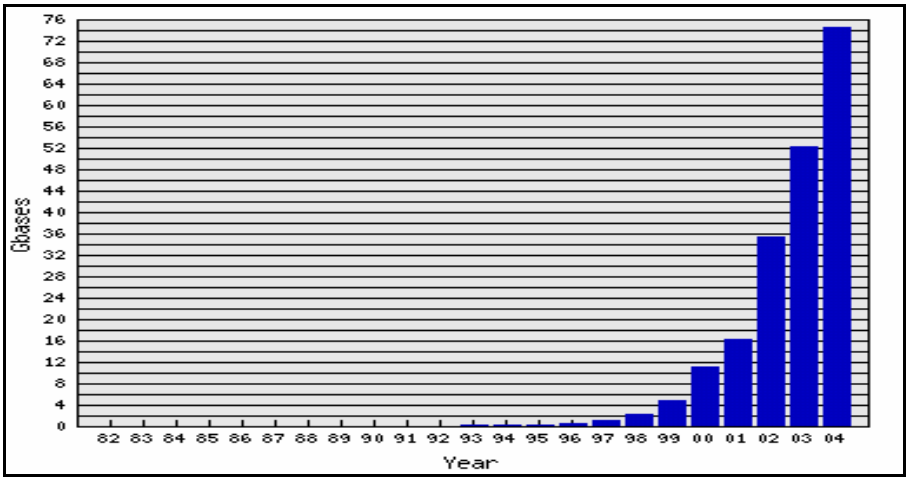


Figure 1.15. L'évolution du nombre de nucléotides dans la banque EMBL (Extrait des statistiques : http://www3.ebi.ac.uk\services\dbstats).

Ø La notion de similarité, d'identité et d'homologie

Deux séquences sont identiques s'ils possèdent une ressemblance parfaite. La similarité est une quantité qui se mesure en pourcentage d'identité. L'homologie quant à elle est une propriété de séquences qui a une connotation évolutive. Deux séquences sont dites homologues si elles possèdent un ancêtre commun. L'homologie se mesure par la similarité. La comparaison de gènes homologues est une approche très efficace dans plusieurs situations. Elle permet la détermination de la fonction et la structure d'une séquence en comparant avec une autre séquence dont la structure et la fonction sont connues. Elle permet également l'identification des régions fonctionnelles au sein des séquences. En plus, elle est très utile pour l'étude et la compréhension de l'histoire évolutive des espèces, par exemple l'étude du développement du virus de la grippe.

Ø La comparaison de séquences

La recherche de similitude entre séquences est une tâche importante dans l'analyse des séquences. Elle est la base de plusieurs autres problèmes bioinformatiques comme la construction des arbres phylogénétique, la recherche des motifs, l'assemblage de fragments d'ADN, etc. Mais cette tâche n'est pas facile car elle nécessite l'élaboration de procédures de calcul efficaces et le modèle utilisé doit être biologiquement acceptable. Parmi les problèmes traités on peut citer la recherche des segments identiques entre plusieurs séquences, la recherche des régions significatives comme les introns et les prometteurs (figure 1.16), la recherche de la super séquence minimale qui englobe un ensemble de séquences, l'alignement multiple de séquences, etc. La plupart des problèmes de comparaison de séquences ont été démontré NP-difficile. Même une solution dont la complexité est polynomiale est difficile à résoudre en bioinformatique vu le grand nombre et la taille colossale des séquences à traiter.

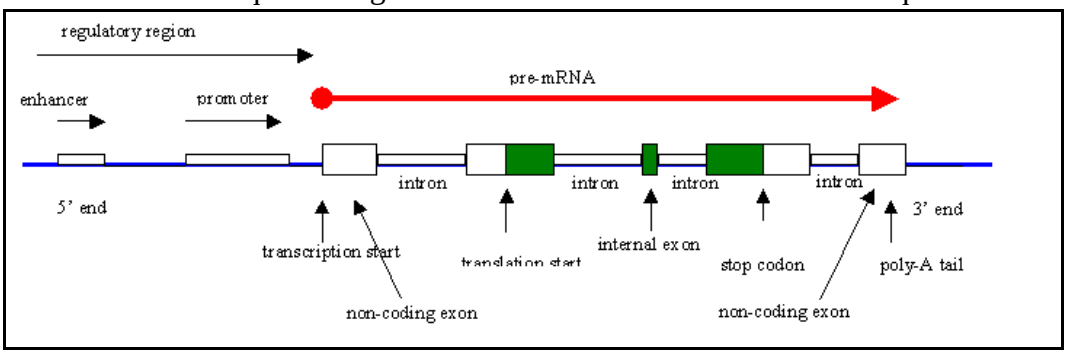


Figure 1.16. Structure d'un gène eucaryote.

1.3.4 L'alignement de paires de séquences

Un alignement multiple de séquences peut-il être vu comme une généralisation d'un alignement de paires de séquences ? Pour pouvoir répondre à cette question, il faut d'abord comprendre ce dernier et investiguer les méthodes proposées pour le résoudre. Etant donné l'importance de l'alignement de deux séquences, il est nécessaire de définir clairement ce problème. On peut décrire le problème comme suit : Étant donné deux séquences de lettres, et une matrice de score pour évaluer deux lettres identiques ou différentes et des pénalités de gaps, le but du problème d'alignement de séquence est de produire un appariement des lettres à partir d'une séquence à l'autre de sorte que le score total est optimal.

Il existe plusieurs types d'alignements : global, local ou semi global [Mount,2001]: L'alignement global consiste à voir à quel point deux séquences sont semblables ,

l'alignement local sert à identifier les régions identiques,cependant,l'alignement semi global détermine si une séquence est incluse dans l'autre. Des insertions et délétions sont introduites afin d'améliorer le score de comparaison (figure 1.17). Mais l'ordre des caractères dans chaque séquence doit être préservé. S'il y a plus de deux séquences, alors c'est un problème d'alignement multiple de séquences. Une grande variété de problèmes biologiques utilise l'alignement des séquences par exemple :

- L'assemblage des fragments d'ADN.
- Détermination des cartes physiques et génétiques de la donnée de sondage sous des protocoles d'expérimentations diverses.
- L'enregistrement, la recherche et la comparaison de séquences D'ADN dans les banques de données.
- Comparaison de deux ou plus de séquences pour les ressemblances.
- La recherche dans les banques de données pour les séquences ou subséquences proches.
- L'exploration de l'occurrence des patterns de nucléotides.
- La détection des éléments informatifs dans les séquences d'ADN est protéines.

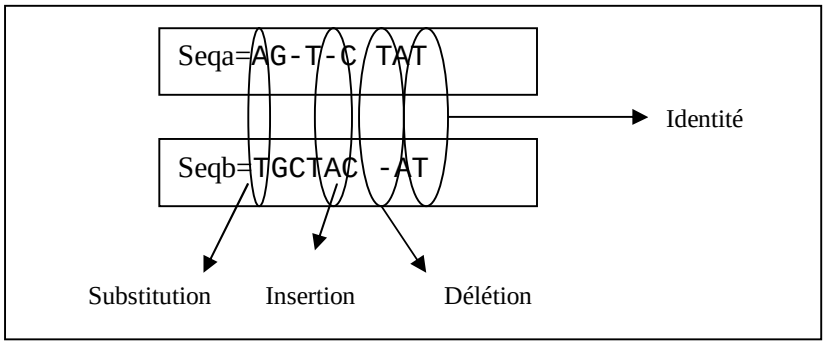


Figure 1. 17. Les différentes situations dans un alignement.

1.3.5 Les méthodes utilisées dans l'alignement de paires de séquences

Il existe plusieurs méthodes pour aligner deux séquences. La plus exhaustive est celle qui calcule le score de tous les alignements possibles. Cependant, cette méthode est inutilisable vu le nombre exponentiel d'alignements possibles (un problème NP-difficile). Heureusement, il existe des méthodes exactes et qui trouvent un alignement optimal de paires de séquences dans un temps de l'ordre de $o(n^2)$. Ces méthodes s'appuient sur la programmation dynamique. Needleman et Wunsch [Needleman et al, 70] ont donné des algorithmes de comparaison de séquences en se basant sur cette approche pour un problème biologique. Nous allons présenter les algorithmes et les méthodes les plus utilisés dans l'alignement de deux séquences.

1.3.6 Les méthodes d'alignements globales

A. La matrice DOT

Cette méthode graphique consiste à mettre une séquence horizontalement et l'autre verticalement puis pour chaque identité entre les bases on met un point (figure 1.16). Un filtre est appliqué pour éliminer le bruit. Les répétitions directes apparaissent comme des petits diagonaux parallèles or les répétitions inversées apparaissent comme des courtes diagonales perpendiculaires (figure 1.18). Les répétitions directes ou inverses sont localisées en utilisant un programme out une fonction appropriée d'analyse d'ADN. En général, on ne regarde pas les correspondances caractère par caractère, mais k caractères à la fois, afin d'éliminer le bruit

(par exemple, k=10 pour l'ADN ou l'ARN, k=4 pour les protéines). Cette méthode est simple et très informative. Cependant, l'identification est l'interprétation des résultas sont encore difficiles [Jovanovich ,04].

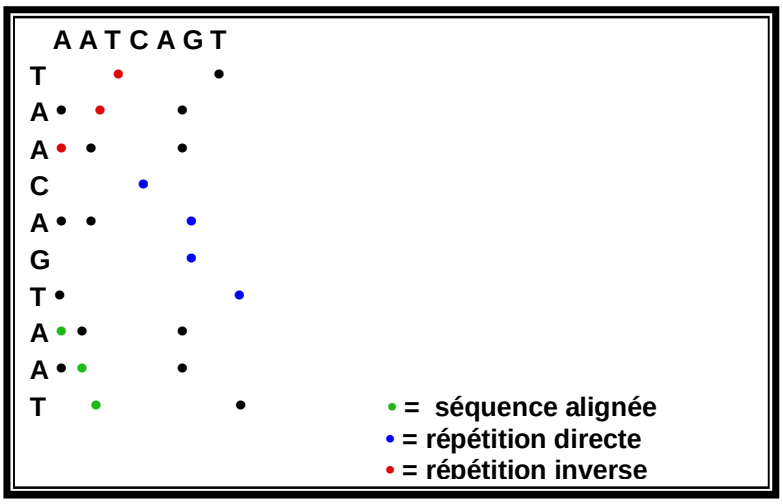
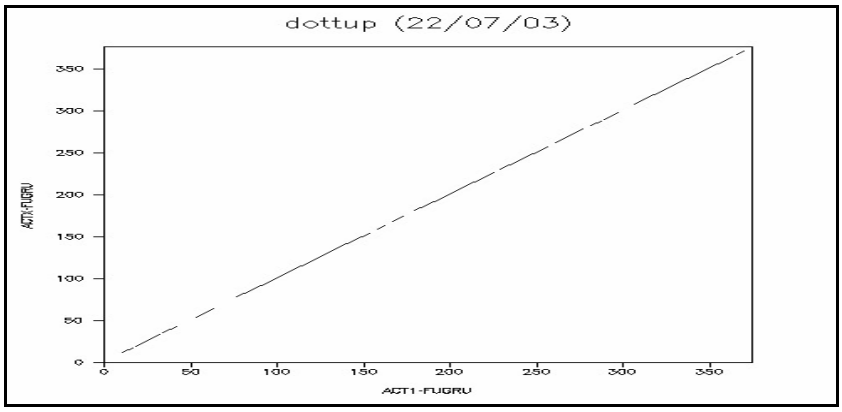
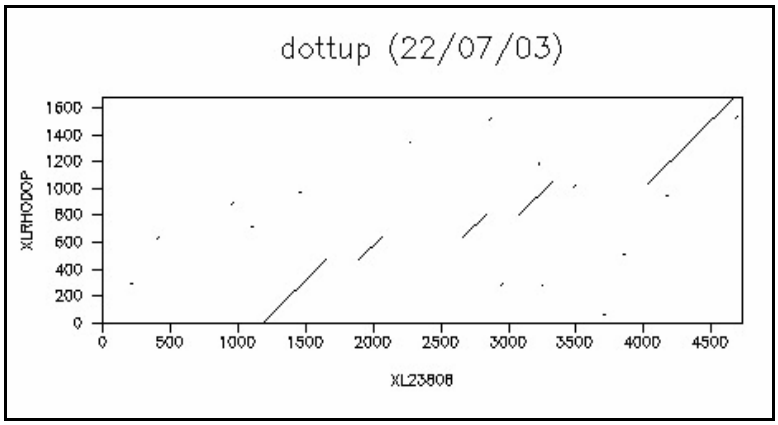


Figure 1.18. La matrice DOT pour la comparaison entre deux séquences.

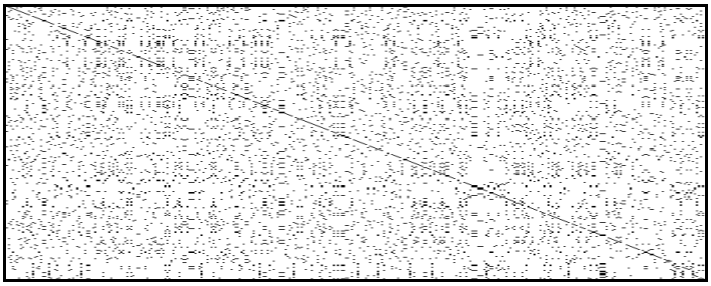
Voici quelques exemples : Les deux séquences suivantes ont visiblement la même origine, et la même longueur : les segments correspondent aux parties communes.



Dans le graphique suivant, on voit un ARNm et l'ADN qui a servi à le produire : les deux sont identiques, mais l'ARNm a été obtenu en retirant certains morceaux de l'ADN (les introns -- les morceaux qui restent sont les exons).



Voici un dotplot d'une séquence par rapport à elle-même (les paramètres ne sont pas les mêmes que pour les graphiques précédents, car on désire plus de détails. Ici, on n'a que du bruit).



B. L'algorithme de Needleman et Wunsch pour l'alignement global

L'idée est basée sur le principe de la programmation dynamique. Soit A et B deux séquences de longueur m et n. L'algorithme se compose de trois étapes [Needleman et al, 70]:

1- Construction d'une matrice (n, m) de comparaison qui correspond aux deux séquences à aligner. On attribue à cette matrice les valeurs appropriées de comparaison de bases selon la matrice de scores élémentaires choisie (figure 1.19).

2- La matrice est transformée par addition de scores. Cette opération est effectuée ligne par ligne en commençant par le coin droit inférieur et en terminant par le coin gauche supérieur. Pour chacune des cases de la matrice transformée, le score somme est calculé de la manière suivante:

$$S(i, j) = Se(i, j) + \max(S(x, y)) \qquad (1.1)$$

Avec $i<x<=m$ *et* $y=i+1$ ou $x=i+1$ *et* $j<y<=n$. Où *S(i,j)* : somme de la case d'indice i et j, *Se* : le score élémentaire de la case d'indice i et j de la matrice initiale et *max S(x,y)* correspond en fait au score somme maximum déjà présent dans la matrice de comparaison en cours de transformation (figure 1.19).

3- Pour trouver le meilleur alignement, 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. Needleman et Wunsch nomment ce passage le chemin des scores maximum (figure 1.20). Les mouvements autorisés pour tracer ce chemin sont :

- Le mouvement diagonal qui correspond au passage de la case (i, j) à la case (i+1, j+1). C'est le mouvement que l'on privilégie.
- 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.
- le mouvement horizontal qui correspond au passage de la case (i, j) à la case (i+1, j), ce qui donne une insertion dans la séquence en j.

La figure 1.19 montre les étapes de l'algorithme de Needleman et Wunsch pour aligner les séquences protéiques : VTEERDEF et lTSHEAL :

Figure 1.19. Un exemple d'application de l'algorithme de Needleman et Wunch.

Après la transformation de la matrice de comparaison initiale, le but est ensuite de trouver le meilleur alignement global, à partir de la matrice transformée. Pour cela, on établit dans la matrice un chemin qui correspond au passage des scores sommes les plus élevés en prenant en compte les trois types de mouvements possibles et en prenant comme point de départ le score maximum présent dans la matrice transformée (figure 1.20).

Dans notre exemple le coût de pénalité pour les insertions est non introduit. Mais il est possible d'incorporer celles-ci dans la méthode. Pour cela il suffit de soustraire dans le calcul de chaque score somme une pénalité en fonction de la position du score "*max S(x,y)*" considéré. Ainsi l'équation précédente prend la forme suivante:

$$S(i, j) = se(i, j) + \max(S(i+1, j+1), S(x, j+1) - P, S(i+1, y) - P \qquad (1.2)$$

Avec $i+2<x<m$ *et* $j+2<y<=n$. Où *S(i,j)* est le score somme de la case d'indice i et j. *se* le score élémentaire de la case d'indice i et j de la matrice initiale et *P* la pénalité donnée pour une insertion.

L'algorithme de Needleman et Wunsch est largement utilisé dans les programmes d'alignement. Sa complexité est de l'ordre de $o(n^2)$, cependant cette complexité reste encore mauvaise surtout dans l'alignement multiple des séquences ou l'alignement contre une banque de séquences.

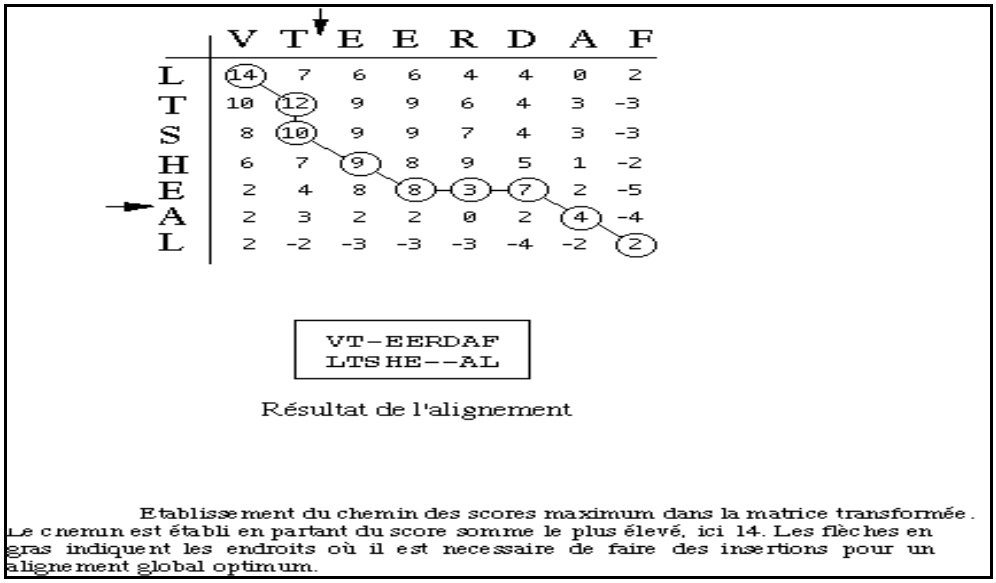


Figure 1.20. Le chemin optimal qui donne l'alignement optimal.

1.3.7 Les méthodes d'alignement local

L'alignement local consiste à trouver les motifs conservés dans deux séquences. Il existe plusieurs méthodes pour effectuer cette tâche.

A. L'algorithme de Smith et Waterman pour l'alignement local

Pour effectuer un alignement local il est préconisé d'utiliser l'algorithme de Smith et Waterman [Smith et al, 81]. La différence essentielle avec l'algorithme de Needleman et Wunsch réside dans le choix du premier point du départ arbitraire pour le calcul des scores sommes. Deuxièmement, la progression du calcul des scores somme est stoppée par tout score somme qui devient inférieur à zéro. La case pointée est alors réinitialisée à zéro et peut être considérée comme nouveau point de départ. Cela implique que le système de scores 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

$$S(i,j) = \max \left| \begin{array}{c} se(i,j) + S(i+1, j+1) \\ se(i,j) + \max S(x, j+1) - P \\ se(i,j) + \max S(i+1, x) - P \\ 0 \end{array} \right| \begin{array}{l} \\ avec \quad i+2 < x \le m \\ et \quad j+2 < y \le n \\ \\ \end{array} \qquad (1.3)$$

Où *S(i,j)* est le score somme de la case d'indice i et j, se le score élémentaire de la case d'indice i et j de la matrice initiale et *P* la pénalité donnée pour une insertion. L'exemple suivant montre le calcul d'un alignement local (figure1.21).

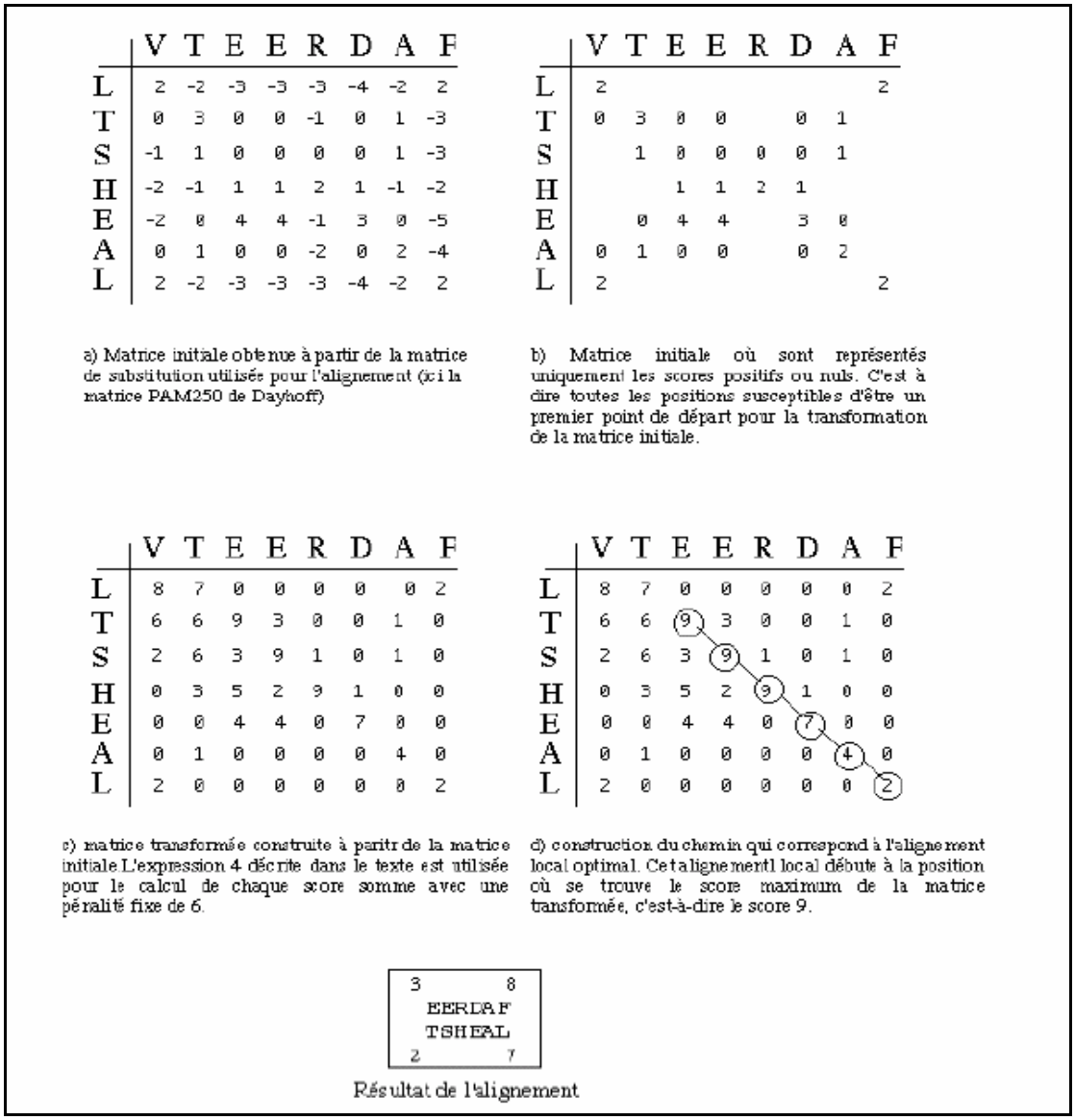


Figure 1.21. Le calcul d'un alignement local.

B. Les méthodes approchées

Les biologistes alignent éventuellement des séquences dont la taille dépasse des centaines de bases contre une banque de séquences qui contient des millions de séquences. Le problème du temps devient prédominant or les algorithmes précédents sont trop consommables en temps et en espace. Plusieurs approches ont été développées pour donner des algorithmes efficaces.

- **FASTA**

Lipman et Pearson ont remarqué que les ressemblances recherchées au "niveau biologique" concernent des fragments de séquences. De plus, dans ces fragments, la fréquence de substitution est beaucoup plus grande que celle d'insertion ou délétion. L'idée de base de FASTA est basée sur la méthode de la diagonale, similaire à la représentation "dot-matrix". La ressemblance se définit par comparaison de paire de fragments de chacune des séquences (fragment : partie de même longueur de chacune des deux séquences, en dot-matrix c'est un morceau d'une diagonale). Ces deux parties contiennent des mots communs séparés par des zones de substitution. Une fonction score est attribuée pour un fragment et la ressemblance est

mesurée par le fragment de score maximum [Lipman et al, 88]. L'algorithme se divise en 4 étapes:

1) Le codage numérique des séquences en k-uplets : mots de longueur k (4 pour les acides nucléiques, 2 pour les protéines). Ceci permet une efficacité beaucoup plus grande pour la deuxième étape.

2) recherche du fragment de plus haut score pour chaque diagonale qui est le score de la diagonale (fragment = suite de mots séparés par de régions de substitution dont la longueur maximale est prédéfinie (paramètre M)).

3) les scores des dix meilleures diagonales vont être recalculés en utilisant une matrice de substitution (PAM 250). C'est ce score qui est listé dans les résultats sous l'appellation init1dans les programmes antécédents à FASTA (FASTP). Pour FASTA, ce score (initn) est recalculé en essayant d'enchaîner à partir de la meilleure diagonale les fragments restants des 9 autres diagonales en tenant compte des insertions ou délétions dues au changement de diagonale.

4) Les résultats sont classés par rapport au score précédent, et pour les meilleurs, un alignement et un nouveau score (Opt) entre la séquence requête et la séquence de la banque, sont calculés à partir de l'algorithme de Needleman et Wunsch légèrement modifié.

- **BLAST**

L'idée originale consiste à combiner à l'algorithme de recherche de paires de segments homologues entre deux séquences une statistique qui permet de les classer. La recherche se fait en plusieurs étapes [Altschul et al, 90]:

1) recherche de mots communs (w-mer) entre la séquence requête et les séquences de la banque (celle-ci peut se faire à l'aide de matrice de substitution). Par défaut c'est Blosum62 qui est utilisée.

2) pour les mots "communs" supérieurs à un certain score (Expected), le mot est étendu si possible à droite et à gauche et si le nouveau score obtenu est supérieur au "cut-off" S, le segment est alors retenu (mis dans la liste des HSP : high scoring pair).

3) pour une séquence de la banque pour laquelle au moins un segment satisfait à la condition précédente, une valeur ("p-value"), basée sur une statistique de Poisson, qui évalue la probabilité que deux séquences "au hasard" aient en commun ces HSP, est calculée et les résultats sont triés par valeur croissante. Il est à remarquer que les valeurs de E et S sont en relation. La valeur par défaut de S est calculée à partir de la valeur de E. Quant à la valeur par défaut de E, elle est le nombre de fois qu'un tel segment est attendu "au hasard" dans la banque (10 pour la valeur par défaut). Il est évidemment dépendant de la taille de la banque, de la fréquence des lettres (acides nucléiques, acides aminés). Le fait d'utiliser des matrices de substitution dans la première étape, pose un problème pour des segments déclarés ressemblant entre deux

séquences alors qu'au niveau biologique cela n'a peut être pas de sens. Deux filtres sont prédéfinis:

- région de faible complexité (SEG)
- région de répétitions internes (XNU)

Ces filtres vous permettent par exemple d'éliminer des ressemblances dues uniquement à des régions acides ou basiques, etc.

Ces programmes offrent différents traitements possibles :

- **Comparaison de séquence nucléique / banque nucléique** : Efficace pour rechercher des éléments régulateurs conservés au cours de l'évolution, de gènes spécifiant des ARN structuraux, d'éléments transposables ou rétrovirus intégrés dans les génomes, ou encore de fragments de vecteur de clonage laissés par erreur dans la séquence. Ex : FASTA - BLASTN
- **Comparaison de séquence protéique / banque protéique** : Dans le cas d'une séquence codante, il est beaucoup plus efficace d'effectuer la recherche au niveau protéique que nucléique, car la protéine est plus informative, donc plus discriminante, grâce à son alphabet de 20 lettres et non pas seulement de 4). Ex : FASTA - BLASTP
- **Comparaison de séquence protéique / banque nucléique (traduite dans les 6 phases)** : Utile pour rechercher des similitudes protéiques avec les séquences des banques nucléiques dont les régions codantes ne sont pas toujours connues ou pas correctement annotées. Ex : TFASTA - TBLASTN
- **Comparaison de séquence nucléique (traduite dans les 6 phases) / banque protéique** : Utile pour comparer une séquence nucléique dont on ne connait pas la position des régions codantes ou qui contient des erreurs (entraînant un risque de décalage de phase de lecture) avec une banque de séquences protéiques. Ex : BLASTX - BLASTC
- **Comparaison de séquence nucléique (traduite dans les 6 phases) / banque nucléique (traduite dans les 6 phases) :** Combine les avantages de TBLASTN et BLASTX, mais au prix d'un ralentissement de la recherche. Ex : TBLASTX
- **Comparaison de séquence protéique / banque protéique :** Lorsque l'on compare les séquences deux à deux, il existe une probabilité non négligeable de détecter des similitudes simplement dues au hasard (faux positifs). Cette probabilité diminue fortement si on compare simultanément plus de deux séquences. BLAST3 a été développé pour rechercher rapidement des alignements multiples entre une séquence requête et deux séquences de la banque. Ex : BLAST3

1.3.8 Matrices de similarités utilisées dans la comparaison de séquences

Pour mesurer la similarité entre deux séquences la plupart d'algorithmes de comparaison utilisent un système de score. C'est un coût attribué aux opérateurs élémentaires de comparaison (insertion, délétion, identité et substitution). Dans la pratique, toutes les lettres

représentent des composés chimiques tels que les acides aminés. En raison de leurs propriétés d'évolution et chimique, quelques acides aminés ont des scores d'identité plus grands que d'autres et certains ont de plus hauts scores de disparité aussi bien. Le degré d'identité entre deux lettres peut être représenté dans une matrice, généralement appelée une matrice de substitution. Un exemple est montré sur la table 5. Le choix de la matrice de substitution est un élément clé pour avoir des alignements de hautes qualités. Donc, La comparaison entre deux lettres de l'alphabets nucléotide ou protéique est plus grande qu'une simple comparaison de lettres. Afin de rendre l'opération de comparaison biologiquement acceptable plusieurs matrices ont été élaborées.

Ø Matrices de score pour les séquences de nucléotides

Pour évaluer les alignements de séquences nucléiques on utilise souvent les matrices d'identités. La matrice d'identité est utilisée pour calculer la ressemblance entre deux séquences. Généralement, on utilise la distance d'édition par exemple 1 pour deux caractères identiques et 0 pour les autres cas (figure 1.22.a). On peut aussi attribuer différents scores aux différentes situations dans un alignement (substitution, délétion et insertion) (figure 1.22.b).

a

$$\begin{pmatrix} & A & C & G & T \\ A & 1 & 0 & 0 & 0 \\ C & 0 & 1 & 0 & 0 \\ G & 0 & 0 & 1 & 0 \\ T & 0 & 0 & 0 & 1 \end{pmatrix}$$

b

$$\begin{pmatrix} & A & C & G & T \\ A & 4 & -1 & -1 & -1 \\ C & -1 & 4 & -1 & -1 \\ G & -1 & -1 & 4 & -1 \\ T & -1 & -1 & -1 & 4 \end{pmatrix}$$

Figure 1.22. Matrices pour les séquences de nucléotides. a: matrice unitaire, b: matrice avec des scores négatifs pour les lettres différentes.

Mais la comparaison de nucléotides est plus grande qu'une simple comparaison de lettres. En fait, le passage des nucléotides A à G, G à A, C à T, et T à C se fait par transition. Les autres passages entre nucléotides se font par transversion, et sont plus rares. Il faut donc utiliser une matrice de distance qui tient compte de ces différences (figure 1. 23).

$$\begin{pmatrix} & A & C & C & T \\ A & 1.36 & -1.6 & -0.37 & -1.6 \\ C & -1.6 & 1.36 & -1.6 & -0.37 \\ G & -0.37 & -1.6 & 1.36 & -1.6 \\ T & -1.6 & -0.37 & -1.6 & 1.36 \end{pmatrix}$$

Figure 1. 23. Matrice de Transition/Transversion.

Ø Matrices protéiques

Plusieurs matrices de score on été élaborées pour calculer le degré de similarité entre les séquences protéiques. Généralement, Il existe quatre catégories de matrices de score : matrices d'identité, matrices basées sur le code génétique, matrices de similarité chimique et enfin les matrices de substitutions. La matrice d'identité est la plus simple forme de matrices

de score protéiques. Elle consiste à l'attribution d'un score positive aux pairs d'acides aminés identiques et un score nul ou négatif pour les autres cas. Ce type de matrices est moins efficace, et n'a aucune signification biologique. Les matrices d'identité sont généralement non utilisables dans les programmes d'alignement. Dans les matrices de code génétique, le score des acides aminés est obtenu à partir des similarités dans le code génétique [Fitch, 66]. Auparavant, cette matrice a été la matrice standard dans les programmes d'alignement mais aujourd'hui, cette matrice n'est plus le premier choix pour l'alignement de séquences protéiques. Concernant les matrices basées sur la similarité chimique, les acides aminés avec les mêmes propriétés physicochimiques comme la hydrophobicité, la polarité et la charge, reçoivent de hauts scores [McLachlan, 72]. Finalement, Les matrices de substitutions sont élaborées en se basant sur des observations statistiques des fréquences de substitutions observées dans les alignements de séquences. Tôt, des matrices de substitutions ont été crées en analysant manuellement des séquences alignées [Dayhoff et al, 78]. Récemment, des matrices de substitutions ont eu l'avantage de l'analyse des alignements établi sur les premières matrices [Henikoff et al, 92]. Une large expérience avec des matrices de substitutions suggère qu'elles soient supérieures à l'identité simple, au code génétique, ou aux matrices basées sur les propriétés physico-chimiques. Par conséquent, les matrices de substitutions sont devenues les matrices de scores les plus utilisés. Généralement, deux séries de matrices de substitutions sont les plus populaires dans la pratique BLOSUM est PAM dans les programmes de comparaison de séquences.

1) PAM (*Percent Accepted Mutation*)

Dans une matrice PAM, le score donné à une paire d'acides aminés est la mesure de probabilité du changement d'un acide aminé à un autre dans une famille de protéines. D'abord, un concept des unités PAM a été introduit pour mesurer la quantité de changement évolutionnaire dans une séquence de protéines. Deux séquences *S* et *T* sont définies pour être un PAM unité divergé si une série de points de mutations admises (sans l'insertion ou la suppression) peut convertir *S* à *T* avec une moyenne d'une substitution d'un acide aminé par cent acides aminés [Dayhoff et al, 78].

Afin de créer la matrice PAM1, des séquences convergentes dans une famille de protéines ont été alignées. Pour chaque pair d'acides aminés, la fréquence des substitutions entre ces deux acides aminées dans un alignement a été déterminée. Ces probabilités ont été placées dans une matrice représentant tous les changements possibles d'acides aminés. Ensuite, la matrice a été normalisée en des valeurs qui représentent la probabilité qu'un d'acide aminé parmi cent subit le changement (unités de PAM). Pour les pairs d'acides aminés (a, b), soit M (a, b) la fréquence observée de substitution, et P la fréquence prévue, donc :

$$Score\ (a, b) = 10\ log10\ (M\ (a,)\ de\ b/P) \qquad (1.4)$$

Les scores sont arrondis jusqu'au prochain nombre entier comme indiqué dans la table 1.3. Un score positif indique que les substitutions entre les acides aminés a et b sont plus probables. Un score nul indique que les substitutions entre a et b se produisent à un taux de base aléatoire. Un score négatif indique que les substitutions entre a et b sont moins probables. Une série de matrices de PAM, telles que PAM160 ou PAM250, a été construite à partir de la

matrice PAM1. PAM250 est la matrice de la famille PAM la plus recommandée pour effectuer des alignements.

2) BLOSUM (*BLOc SUbstitution Matrix*)

Dans les matrices de BLOSUM, des scores pour chaque pairs d'acides aminés sont déterminés par des observations des fréquences de substitutions dans les blocs d'alignements locaux des protéines reliés dans la base de données BLOCS [Henikoff et al, 92] . Dans BLOCS il existe 3.000 blocs de séquences fortement conservées représentants des centaines de groupes de protéines. De même que la construction des matrices de PAM, les logs de probabilité de la fréquence de substitution pour chaque pair d'acides aminés sont calculés pour établir une matrice de BLOSUM. Il y a une série de matrices BLOSUM, chaque BLOSUM*n* dénoté pour un certain n, où le nombre *n* indique le degré de similarité des séquences desquels la matrice de BLOSUM a été dérivée [Henikoff et al 92]. Par exemple, la matrice BLOSUM30 est crée à partir des alignements de séquences partageants pas plus de 30% d'identité. On pense que la matrice BLOSUM62 est une bonne matrice globale tandis que BLOSUM45 est recommandée pour des séquences plus divergentes et BLOSUM100 est suggéré pour des séquences fortement liées [Henikoff et al, 92]. La matrice BLOSUM62 a été trouvée plus semblable à la matrice PAM250 mais elle est plus efficace pour trouver les membres des familles de protéines.

Ø Le choix d'une matrice protéique

Comme on a vu, il existe plusieurs matrices de scores et il est souvent difficile de choisir la matrice appropriée dans les différents programmes de comparaison de séquences protéiques. Car de toute évidence, la sensibilité des méthodes dépend aussi de la qualité des matrices. En effet, il n'existe pas une matrice universelle pour tous les types de comparaisons. Les études ont montré que l'utilisation des matrices différentes selon le type de similarité recherché a commencé à être suggérée par exemple le programme Clustal [Thompson et al, 94] pour l'alignement multiple de séquences utilise des matrices différentes selon le degré de similarité des paires de séquences. Chaque matrice est bien adaptée à des comparaisons bien spécifiques. Par exemple, la matrice PAM250 de Dayhoff donne un poids trop important à l'identité et n'est pas bien adaptée à la comparaison de protéines très distantes car elle ne renferme pas suffisamment d'informations structurales. Altschul [Altschul, 91] a fait une étude sur l'utilisation des matrices de la famille PAM et il recommande la matrice PAM40 pour retrouver des alignements courts avec des protéines très semblables et les matrices PAM120 et PAM250 pour des alignements plus longs et de plus faible ressemblance. Il préconise également l'utilisation de la PAM120 lorsque l'on ne connaît pas a priori le degré de ressemblance de deux séquences comme c'est le cas par exemple dans les programmes de recherche de similitudes avec les banques de données.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	2																			
R	–2	6																		
N	0	0	2																	
D	0	–1	2	4																
C	–2	–4	–4	–5	4															
Q	0	1	1	2	–5	4														
E	0	–1	1	3	–5	2	4													
G	1	–3	0	1	–3	–1	0	5												
H	–1	2	2	1	–3	3	1	–2	6											
I	–1	–2	–2	–2	–2	–2	–2	–3	–2	5										
L	–2	–3	–3	–4	–6	–2	–3	–4	–2	2	6									
K	–1	3	1	0	–5	1	0	–2	0	–2	–3	5								
M	–1	0	–2	–3	–5	–1	–2	–3	–2	2	4	0	6							
F	–4	–4	–4	–6	–4	–5	–5	–5	–2	1	2	–5	0	9						
P	1	0	–1	–1	–3	0	–1	–1	0	–2	–3	–1	–2	–5	6					
S	1	0	1	0	0	–1	0	1	–1	–1	–3	0	–2	–3	1	3				
T	1	–1	0	0	–2	–1	0	0	–1	0	–2	0	–1	–2	0	1	3			
W	–6	2	–4	–7	–8	–5	–7	–7	–3	–5	–2	–3	–4	0	–6	–2	–5	17		
Y	–3	–4	–2	–4	0	–4	–4	–5	0	–1	–1	–4	–2	7	–5	–3	–3	0	10	
V	0	–2	–2	–2	–2	–2	–2	–1	–2	4	2	–2	2	–1	–1	–1	0	–6	–2	4

Table 1.3. Matrice PAM.

En ce qui concerne la famille de matrices BLOSUM, Henikoff et al [Henikoff et al, 93] ont évalué plusieurs matrices en utilisant le programme BLAST de recherche de similitude sans insertion-délétion. Leur étude a établit que les matrices dérivées directement des comparaisons de séquences ou des comparaisons de structure sont supérieures à celles qui sont extrapolées du modèle d'évolution de Dayhoff. En particulier ils concluent que la matrice BLOSUM 62 permet d'obtenir les meilleurs résultats. Néanmoins, Pearson [Pearson, 96] dans une étude comparative de différentes méthodes de recherche avec les bases de données a pu montrer l'importance de l'algorithme et de son paramétrage dans l'utilisation des matrices de substitution. Ainsi, les matrices de type PAM déterminées à partir de données récentes comme celles de Jones [Jones ,99] peuvent donner des résultats comparables à ceux obtenus avec les meilleures matrices de type BLOSUM (62 ou 50 par exemple).

Vogt et al [Vogt et al ,95] ont testé l'influence des autres paramètres sur les matrices protéiques comme les pénalités d'insertion-deletion et le choix de la méthode d'alignement (figure 1.24). Leur étude montre que l'ensemble des matrices donne de meilleurs résultats avec les alignements globaux et que leurs performances peuvent varier très significativement selon le système de pénalité d'insertion-délétion que l'on choisit. Dans cette étude, la matrice établie par Gonnet et al [Gonnet et al, 92] est celle qui donne les meilleurs résultats. Cette dernière a été construite à partir d'une base de données protéique de 8 344 353 acides aminés ou chaque séquence a été comparée à l'ensemble des séquences de la banque. Tous les alignements significatifs recensés servent ensuite à générer une matrice avec une distance PAM de 250. Dans cette étude, Vogt et ses collaborateurs ont donné les matrices les plus performantes, les BLOSUM 50 et 62 ainsi que la matrice de structure tertiaire de Johnson et Overington [Johnson et al, 93]. Cette dernière a été développée à partir de l'étude de 235 structures protéiques regroupées en 65 familles pour lesquelles on connaissait au moins la structure tridimensionnelle de trois séquences. Les correspondances entre BLOSUM et PAM, basées sur la théorie de l'information sont :

PAM250 -> blosum45, PAM160 -> blosum62, PAM120 -> blosum80.

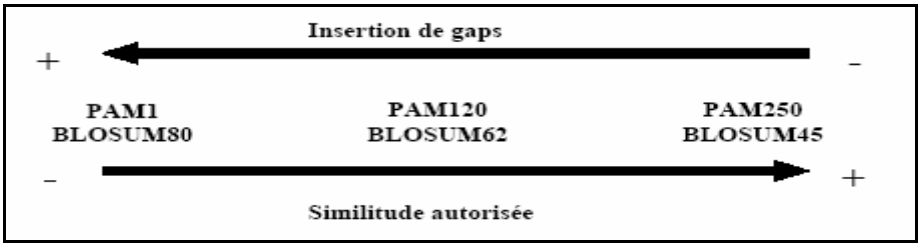


Figure 1.24. La relation entre la similitude des séquences, l'insertion des gaps et matrice protéique utilisée.

L'évaluation des matrices est très liée aux méthodes d'expertise utilisées et que leur usage est fortement corrélé aux types d'algorithmes et de paramétrages utilisés. En conclusion, il apparaît tout de même que les matrices plutôt basées sur les comparaisons de séquences comme celle de Gonnet ou les BLOSUMs de Henikoff ou sur des structures tridimensionnelles semblent donner plus souvent de meilleurs résultats que celles basées principalement sur le modèle de Dayhoff.

1.3.9 Le coût des gaps

Un gap est une suite de blancs insérée dans les séquence afin d'optimiser le score d'un alignement. Un gap représente une évolution biologique produite dans une séquence que ce soit des insertions, des suppressions ou des substitutions [Vingron et al, 94]. Le concept de gap dans un alignement est très important dans plusieurs applications biologiques. La suppression ou l'insertion d'une subséquence entière apparaît généralement comme un seul événement mutationnel. En effet, plusieurs de ces événements mutationnels peut créer des gaps de différentes longueurs. On a besoin de pénaliser les gaps comme un ensemble quand on essaie d'aligner deux séquences d'ADN pour éviter d'assigner une grande pénalité pour ces mutations. Concernant les protéines, deux séquences protéiques doivent être relativement similaires sur plusieurs intervalles mais différentes dans les intervalles où une séquence contient des sous-unités protéiques et l'autre séquence ne contient pas. L'introduction de gaps va nous aider de traiter ce cas comme des 'bons alignements ', bien qu'il existe une longue suite d'opérations d'insertion/délétion dans les séquences. Afin d'avoir un bon alignement, il faut avoir de bonnes pénalités pour les gaps. Pour cela plusieurs modèles ont été élaborés pour pénaliser les gaps :

Ø Le modèle de gaps avec une pénalité constante

Le premier modèle est d'utiliser une fonction constante où le score d'un gap est indépendant de sa taille. Le Score d'un alignement entre les séquences S et T contenant k trous est donné par la formule suivante :

$$score(alignement) = \sum_{i=1}^{n} sc(s_i, t_i) - k * pg \qquad (1.5)$$

s_i, t_i sont les lettres des séquences S, T respectivement. *pg* est la pénalité de gaps de taille k.

Ø Le modèle de gaps avec une pénalité affine

C'est le modèle de pénalité de gaps le plus utilisé dans les programmes d'alignements. La fonction affine est biologiquement la plus significative. Elle est constituée de deux parties : une pénalité pour ouvrir un gaps dénoté par 'gap open' (GOP) et une pénalité pour l'extension

d'un gap existant dénoté par 'gap extended' (GEP). Donc le coût d'un gap est donné par la formule suivante:

$$Cost\ (gaps) = GOP + Ne * GEP \qquad (1.6)$$

Ne est le nombre d'espaces (ou dash) dans un gap. Prenant par exemple l'alignement suivant:

```
ABTPACCT---A
A-T--CGTPPNX
```

Si on prend les score de similarité : 3 pour identité, -1 pour substitution et les pénalités de gaps suivantes : GOP=4, GEP=0.2. Dans cet exemple on a 3 gaps, de 1, 2,3 espaces chacun donc le score global (= 6.2) de l'alignement ci-dessus est calculé par la formule suivante :

$$\sum_{i=1}^{n} sc(s_i, t_i) - Ng * GOP - Ne * GEP \qquad (1.7)$$

Il est parfois très utile de mettre les gaps de l'extrémité sans coût. Ceci a l'avantage d'arrêter les effets indésirables comme des résidus simples bondissant à la frontière de l'alignement. Observons les deux alignements suivants :

```
ACTAACCT    ACTAACCT
A---ACGT    ---AACGT
```

Bien qu'ils possèdent le même score d'alignement, le deuxième alignement (a droite) est biologiquement le plus significative.

Ø Le modèle de gaps avec une fonction convexe

Une autre fonction de coût est la fonction convexe où chaque espace supplémentaire est mois coûteux que le précèdent. Le score d'un gap de taille *ng* est donné par la formule suivante :

$\cos t(gaps) = w_t + \log_e(ng)$ Où *ng* est la longueur de gap.

Ø Ie modèle de gaps avec une fonction affine généralisée

M.A. Zachariah et al [Zachariah et al, 05] ont donné en 2005 un modèle généralisé de la fonction affine. Ce nouveau modèle est similaire au modèle classique le la fonction affine. Néanmoins, Ce modèle est capable de détecter les homologies entre les protéines et donner des alignements de haute qualité (figure 1.25.a). L'évaluation de la fonction affine généralisée a montré qu'elle aligne peu de pairs de résidus par rapport à l'ancienne fonction affine, mais elle achève une haute précision par-résidus (figure 1.25.b). Comme la fonction affine classique, la fonction affine généralisée donne un coût fixe dénoté par *a* pour les alignement. Un autre coût *b* est donné pour les résidus "gappés". Les pairs de résidus non alignés peut être inclus dans le gap et reçoit un coût c. Le coût pour un gap contenant k_1 résidus dans une séquence et k_2 dans l'autre séquence avec $k_1 >= k_2$ est donné par la formule suivante :

$$Coût = a + b(k_1 - k_2) + ck_2 \qquad (1.7)$$

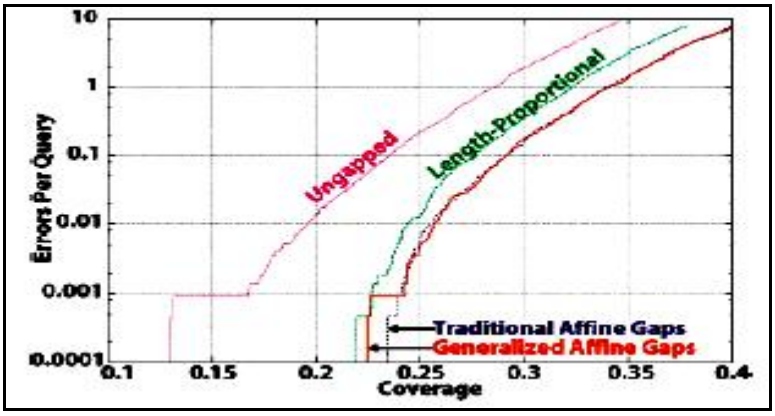


Figure 1.25.a : l'erreur de couverture entre la fonction affine généralisée et 3 autres modèles de gaps. [Zachariah et al, 05].

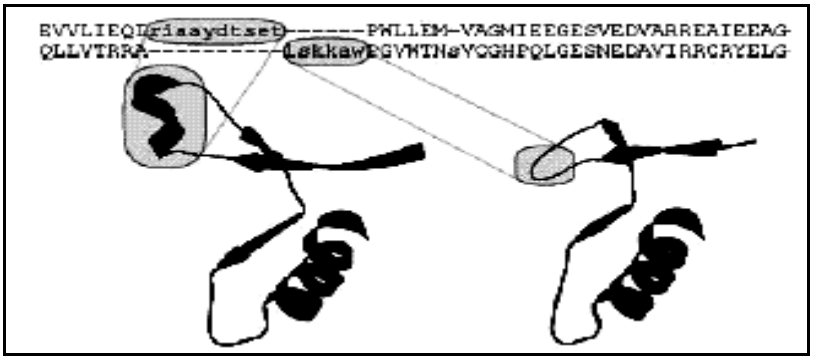


Figure 1.25.b : La motivation de l'utilisation de la fonction affine généralisé. [Zachariah et al, 05].

1.3.10 Problèmes et limites de la recherche de similarités pour inférer une fonction

Plusieurs limitations peuvent nuire la recherche de similarité en utilisant une méthode automatique comme dans les cas suivants :

- **Les gènes inconnus :** Quand un gène ne ressemble à aucun autre, on l'appelle "orphelin". Quand le génome de la levure a été obtenu, près de la moitié de ses gènes n'avaient pas d'homologues connus dans les banques.

- **Les erreurs :** Les informations présentes dans les banques ne sont pas toujours justes et peuvent contenir des erreurs. Donc, il est recommandé d'avoir d'outils d'identification directe de gènes qui ne reposent pas sur des homologies afin de ne pas propager des erreurs.

- **Les gènes orthologues et paralogues :** Une fois qu'une certaine similarité est mise en évidence, il est nécessaire de séparer les gènes orthologues des paralogues. Il est fréquent que certains gènes se dupliquent. Un gène orthologue conserve généralement sa fonction première. Cependant, les gènes paralogues peuvent évoluer indépendamment et acquérir des fonctions complètement différentes. Seule une analyse de leur évolution via la construction d'arbres phylogénétiques permet de différencier ces deux cas.

- **Le "bricolage de l'évolution" :** Une autre difficulté de la recherche de fonctions provient des réarrangements qui s'opèrent lors des étapes séparant le gène de la protéine fonctionnelle :

  - L'épissage alternatif : pour un même gène et dans un même organisme, l'élimination des introns peut être différente selon la cellule concernée. Ainsi, pour un même gène,

l'ARNm sera différent et donnera naissance à une protéine différente. C'est un véritable phénomène d'économie pour la cellule.

- Ø L'association de fragments provenant de gènes différents permet l'émergence de fonctions totalement nouvelles.

- **La maturation post-traductionnelle de la protéine :** Les protéines, synthétisées dans les ribosomes du cytoplasme, vont migrer grâce à des signaux d'adressage spécifiques vers les mitochondries, les lysosomes, les peroxysomes, etc. Elles peuvent aussi traverser le réticulum endoplasmique et passer par l'appareil de Golgi pour être sécrétées dans le milieu extracellulaire. Une fois traduite, la protéine peut subir une maturation post-traductionnelle : au cours de leur transfert, les protéines subissent une série de modifications biochimiques (glycosylation, hydroxylation) les modifiant profondément, de telle sorte que la protéine finale est bien différente de la molécule directement codée par le génome. Pour toutes ces raisons, les résultats produits par les logiciels ne constituent que des hypothèses qui doivent être, elles aussi, vérifiées par une démarche expérimentale en laboratoire, notamment par l'observation des effets de l'altération ou de la délétion du gène dans l'organisme, ou dans l'un de ceux qui lui sont apparentés.

1.4 Conclusion

La bioinformatique est un domaine de recherche très actif qui inclut plusieurs domaines comme l'informatique, la biologie et les mathématiques. Plusieurs problèmes sont encore non résolus comme l'assemblage d'ADN, la phylogénie, l'alignement multiple de séquences, etc. dans ce chapitre on a essayer de donner un état d'art sur la bioinformatique. On a donnée également les principes de base de l'alignement de paires de séquences comme les méthodes de la programmation dynamique pour l'alignement de séquences, les matrices de scores les plus utilisées en alignements de séquences, et les différents modèles de gaps utilisés pour pénaliser les différentes opérations d'insertions/délétions "indels". Dans le prochain chapitre on va traiter le problème d'alignement multiple de séquence MSA.

Chapitre 2
L'alignement multiple de séquences

"Scientists have isolated the gene that makes scientists want to isolate genes"
—Randy Glasbergen

2.1 Introduction

L'apparition des grands projets de génome a mené à une explosion des données de séquences dans les bases de données. L'analyse des familles de protéines, la compréhension de leurs tendances évolutionnaires et la détection des homologies sont maintenant les premiers objectifs de ces projets. Les outils d'annotation et d'analyse de génomes comme la prédiction de pli, la modélisation de l'homologie, la fixation de protéine-ligand et les algorithmes de "clustering" se fondent fortement sur des alignements multiples précis.

L'alignement multiple de séquences MSA (Multiple Sequence Alignment) consiste à aligner plusieurs séquences dans leur intégralité afin de tirer les relations entre une famille de séquences (figure 2.1). Le but principal de l'alignement multiple est de montrer les rapports essentiels et les caractéristiques communes entre un ensemble de séquences de protéines ou de nucléotides. Le MSA permet de caractériser les régions conservées et les régions variables au sein d'une famille de séquences (figure 2.2). Il permet aussi de construire la séquence consensus de plusieurs séquences alignées. Le MSA contribue efficacement à une meilleure compréhension de l'évolution des séquences biologiques. En plus, l'alignement multiple est également utilisé dans plusieurs autres domaines comme la bioinformatique structurelle où le MSA est utilisé pour la prédiction structurelle et fonctionnelle des protéines.

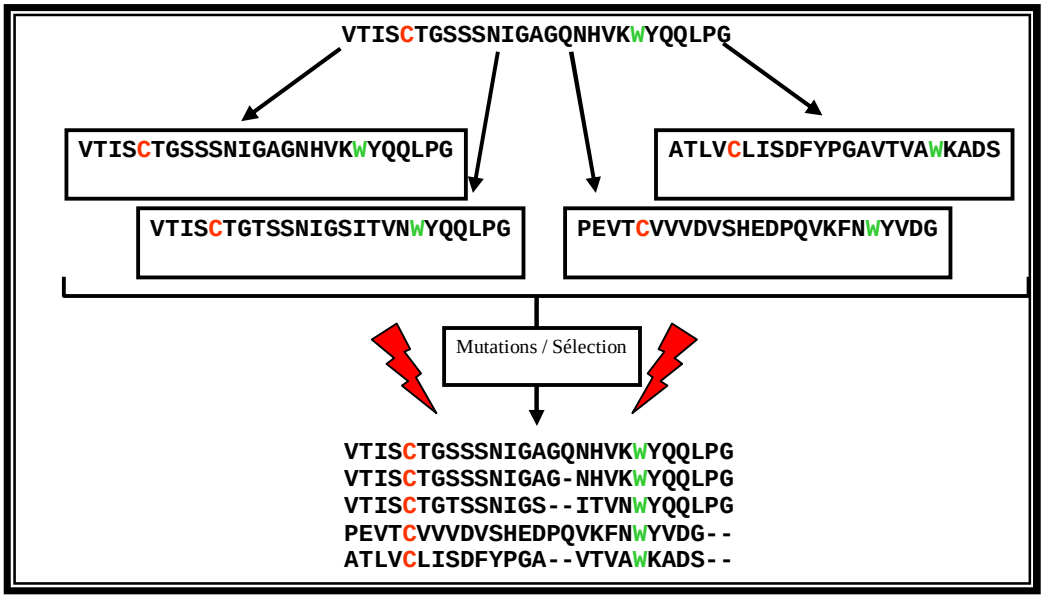


Figure 2.1. Alignement multiple : une histoire.

Malheureusement, La construction manuelle d'un alignement multiple est une opération très fastidieuse et non praticable. Pour cela la construction automatique des alignements est

devenue aujourd'hui une tâche importante en bioinformatique. Par ailleurs, le MSA est caractérisé par une grande complexité temporaire et spatiale [Wang et al, 94]. En effet, pour aligner 3 séquences de taille 1000, il faut 1000^3 soit 1 Go de mémoire alors que les biologistes ont souvent plusieurs centaines de séquences à aligner et ils veulent des solutions optimales dans des courtes durées. Donc, le problème d'alignement multiple est plus complexe qu'une simple et directe généralisation d'alignement de paires de séquences.

Résoudre le problème d'alignement multiple soulève trois questions fondamentales:

- Quel type de séquences à aligner faut-il choisir ?
- Comment juger la qualité d'un alignement ?
- Comment trouver un bon alignement multiple de séquences ?

Ces questions imposent de faire trois choix quand on veut effectuer un alignement de séquences.

- Le choix de l'ensemble de séquences.
- Le choix d'une fonction objectif permettant la comparaison de séquences.
- Le choix d'une stratégie de recherche.

Le choix des séquences à aligner est un problème typiquement biologique. C'est au biologiste de déterminer quel ensemble de séquences faut-il aligner. En effet, les relations de convergence ou de divergence entre les séquences à aligner ont un grand effet sur la qualité d'alignement obtenu. Cependant, la grande difficulté dans l'alignement multiple de séquences est de qualifier un alignement, et savoir si biologiquement il est bon. Cette difficile question peut être seulement répondue en utilisant une fonction objectif mathématique capable de mesurer la qualité biologique d'un alignement. En effet, une bonne fonction objectif va conduire vers un bon alignement du point de vue biologique. Pour cela plusieurs fonctions objectifs ont été proposées tel que la somme des paires SP [Nicolas et al, 02], T-COFFEE score [Notredame et al, 98], le score profil, etc. Malheureusement, il n'existe pas à cet instant, une fonction objectif dont l'optimal mathématique est corrélé avec l'optimal biologique. En conséquence, il n'existe pas une fonction mathématique pour l'évaluation biologique d'un alignement multiple de séquences. Le seul moyen utilisé pour tester l'efficacité biologique des méthodes d'alignement est l'utilisation des bases d'alignements de références (benchmarks). Ces dernières contiennent des familles de séquences dont l'alignement multiple optimal (du point de vue biologique) est connu et généralement crée à la main. Le troisième problème lié à l'alignement multiple est calculatoire. Supposant que nous avons à notre disposition un ensemble adéquat de séquences et une fonction objectif parfaite, le calcul mathématique de l'alignement optimal est une tâche très complexe pour qu'une méthode exacte soit employée [Wang et al, 94]. Même si la fonction utilisée dedans est une simple maximisation du nombre d'identités parfaites dans chaque colonne, le problème est déjà hors de portée pour plus de trois séquences. C'est pourquoi toutes les méthodes courantes d'alignement multiple sont des heuristiques et aucune d'eux ne garantit une meilleure optimisation.

Il est commode de classifier les algorithmes existants en quatre catégories principales : exact, progressif et itératif et statistique. Les algorithmes exacts sont des heuristiques de haute qualité qui fournit habituellement un alignement très près de l'optimalité [Stoye et al, 97]. Néanmoins, Elles peuvent seulement manipuler un petit nombre de séquences (< 20) et sont limités à la fonction objectif de la somme de paires. Par ailleurs, Les algorithmes d'alignements progressifs sont de loin les plus répandus. L'alignement progressif est construit progressivement selon un ordre de séquences. Son grand avantage est la vitesse, la simplicité et une sensibilité raisonnable, même s'il est de nature heuristique et qui ne garantit pas un bon niveau d'optimisation. Troisièmement, les méthodes d'alignement itératives utilisent des algorithmes capables à produire un alignement initial de basse qualité et ensuite de le raffiner par une série de raffinements itératifs jusqu'il n'y plus d'améliorations qui peuvent être apportées. Les méthodes itératives peuvent être déterministes ou stochastiques selon la stratégie d'amélioration utilisée. Contrairement aux méthodes précédentes, la dernière classe de méthodes d'alignement utilise des modèles statistiques comme le modèle caché de Markov HMM pour construire un alignement multiple. Les méthodes basées HMM nécessitent un grand nombre de séquences pour que le modèle HMM fonctionnent correctement.

Dans ce chapitre d'abord, une petite introduction aux arbres phylogénétiques est présentée. Ces derniers sont largement utilisés dans la plupart des méthodes d'alignement multiple de séquences [Jiang et al, 94]. Dans la section suivante, on présente les fonctions objectifs les plus utilisées dans le MSA. Ensuite en va présenter les différentes approches utilisées pour résoudre le problème MSA. Finalement on donne une comparaison entre les différentes méthodes d'alignement multiple.

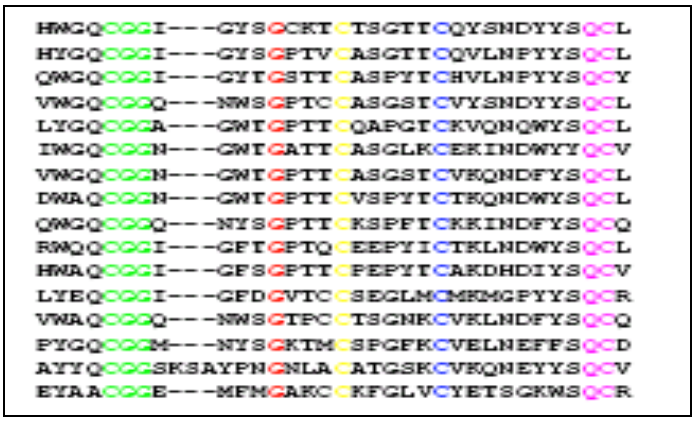


Figure 2.2. Site de fixation de la cellulose (extrait de prosite entrée PS00562).

2.2 Les arbres phylogénétiques

Les arbres phylogénétiques joue un grand rôle dans la compréhension de l'histoire évolutive des espèces comme par exemple l'évolution des virus. Ce qui permet de trouver le remède approprie pour un tel virus. L'évolution des espèces est comprise comme un processus divergent modélisé le plus souvent sous la forme d'un arbre (figure 2.3) dont les noeuds feuilles représentent les espèces contemporaines et les nœuds internes les espèces ancestrales. La phylogénie a plusieurs buts. Elle permet une meilleure compréhension des mécanismes de l'évolution et les mécanismes moléculaires associés. Elle est également très utile dans l'étude de

la biodiversité. Dans le cadre de l'alignement multiple, elle est amplement utilisée dans la plupart des méthodes d'alignement. Elle permet de déterminer l'ordre d'alignement des séquences dans les méthodes progressives. Un bon arbre phylogénétique de séquences va conduire vers un bon alignement [Jiang et al, 94]. La deuxième utilisation des arbres phylogénétiques en MSA est pour calculer les poids des séquences dans la fonction objectif WSP [Altschul et al, 89]. La construction de ces arbres est fondée sur le concept de l'horloge moléculaire où les mutations se font d'une manière hasardeuse et les longueurs d'arcs sont proportionnelles aux durées écoulées. On a deux types d'arbres pour représenter une phylogénie :

- Les arbres enracinés : tous les espèces ont un ancêtre commun.
- Les arbre non enraciné : pas d'ancêtre commun.

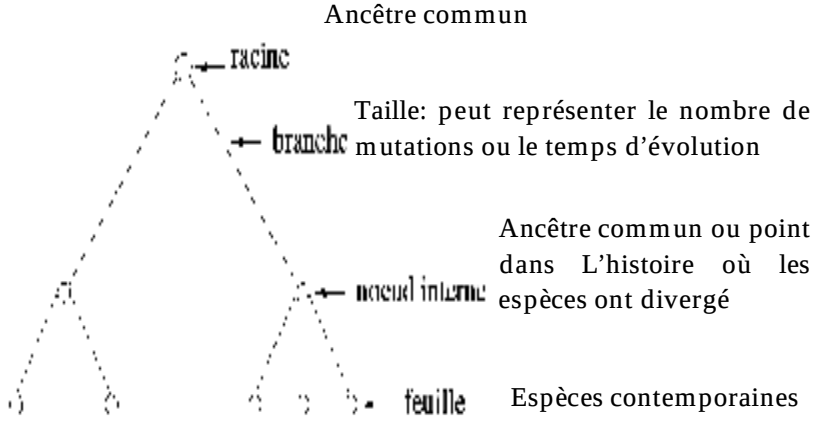


Figure 2.3. Un arbre phylogénétique enraciné.

2.2.1 Notion d'homologie et d'homoplasie

Afin de structurer les classifications sur la base de monophylies, il faut pouvoir distinguer les similitudes héritées d'un ancêtre commun des similitudes qui ne le sont pas. On a deux types de similitude : l'homologie et l'homoplasie. Homologie caractérise des descendants qui ont le même parent. Cependant, l'homoplasie caractérise deux individus qui ont des similitudes sans être du même parent. Les homoplasies sont aujourd'hui divisées en deux types : les convergences et les réversions. Une convergence correspond à l'apparition d'un caractère indépendamment chez deux taxons (ou davantage). Une réversion représente une réapparition chez un taxon d'un caractère identique au parent (figure 2.4) [Berry, 97].

2.2.2. Les méthodes de construction d'arbres phylogénétiques pour le MSA

En phylogénie, il existe trois grandes classes de méthodes de construction d'arbres phylogénétiques: les méthodes basées sur la distance, les méthodes probabilistes et les méthodes basées sur le caractère [Berry, 97]. Les deux dernières méthodes sont les plus précises. Cependant, leur complexité est exponentielle. Les méthodes basées sur la distance sont les plus utilisées dans les méthodes d'alignement vu leur complexité polynomiale. Ces méthodes sont basées sur le calcul de la distance entre les séquences prise deux à deux. La distance entre deux séquences est calculée suivant un model mathématique (mutation, temps...).

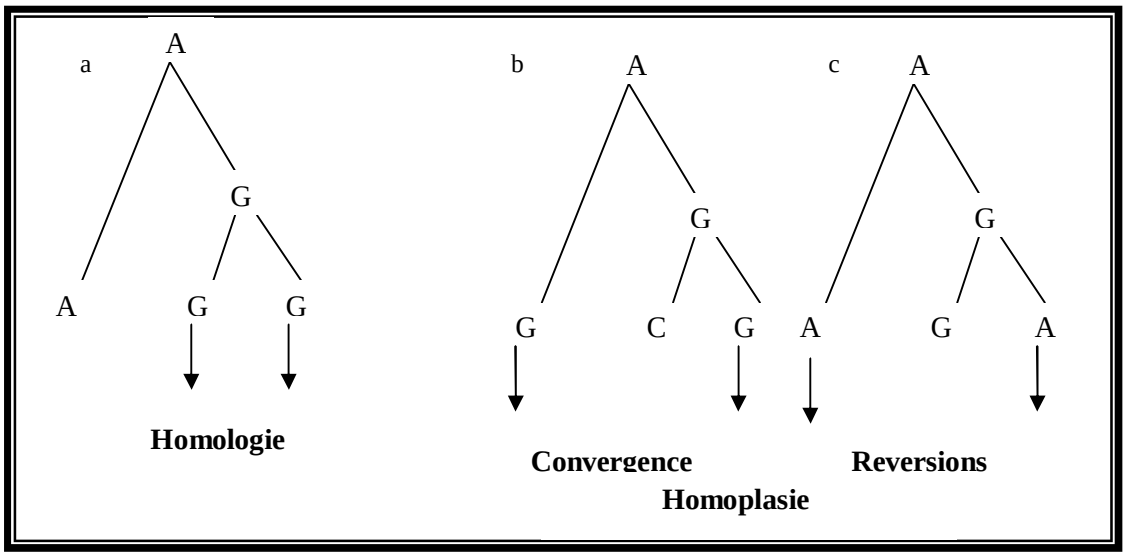


Figure 2.4. Les différentes catégories de ressemblances suite à l'évolution d'un caractère. (a): ressemblance due à l'homologie, (b,c): ressemblance due à l'homoplasie suite à une convergence ou à une réversion.

Ø La méthode UPGMA (Unweight Pair Group Method with Arithmetic mean)

Parmi les méthodes les plus utilisées dans la construction des arbres phylogénétiques, la méthode UPGMA. UPGMA utilise un algorithme de "clusterisation" séquentiel dans lequel les relations sont identifiées dans l'ordre de leur similarité et la reconstruction de l'arbre se fait pas à pas grâce à cet ordre. Il y a d'abord l'identification des deux séquences les plus proches. Ce groupe est ensuite traité comme un tout, puis on recherche la séquence la plus proche et ainsi de suite jusqu'à ce qu'il n'y ait plus que deux groupes. Cette méthode est très utilisée dans les méthodes d'alignement multiple de séquences [Thompson et al, 94]. Cependant, L'inconvénient majeur de cette méthode est la sensibilité de la méthode à des taux de mutations différents sur les différentes branches. L'algorithme détaillé peut être trouver dans [Fitch et al, 67]

Ø La méthode NJ (Neighbor-Joining)

Cette méthode développée par Saitou et Nei [Saitou et al 87], elle essaie de surmonter les inconvénients de la méthode UPGMA afin de donner des taux de mutations différents sur les branches. Une matrice est construise à l'aide des données initiales pour donner 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 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.

2.3 Fonction objectif d'un alignement multiple

Dans le cas de deux séquences, le problème de score d'un alignement est facile. C'est la somme des scores des identités, insertions/suppressions et substitutions qui existent entre les deux séquences. Néanmoins, dans le cas d'alignement multiple, le problème est plus complexe. En effet, il n'existe pas une fonction objectif globale qui mesure efficacement la qualité d'un

alignement. Une bonne fonction de score si elle existe doit contenir toutes les informations biologiques qui existent entre les séquences à aligner. Pour cela, plusieurs fonctions de score on été proposées. Nous allons présenter dans la suite les fonctions les plus utilisées dans les algorithmes d'alignement pour évaluer un alignement multiple.

2.3.1 La somme des paires SP (Sum of Pairs)

La somme des pairs SP est la méthode la plus utilisée dans les méthodes d'alignement multiple. Ayant un alignement de n séquences, le SP score est égale à la somme de tous les scores d'alignement par paires possibles des séquences prise deux à deux.

$$SP = \sum_{i=1}^{n-1} \sum_{j=i}^{n} sc(S_i, S_j) \qquad (2.1)$$

Où *sc(Si,Sj)* est le score d'alignement des séquences Si,Sj de taille *m* extrait de l'alignement multiple. Le score *sc(Si,Sj)* d'alignement de deux séquence est donné par la formule suivante.

$$\text{sc(Si, Sj)} = \sum_{i=1}^{m} sc(a_i, b_j) - Pénalité(GAPS) \qquad (2.2)$$

SP score utilise généralement une fonction affine pour pénaliser les gaps.

$$Pénalité\ (gaps) = GOP + Ne * GEP \qquad (2.3)$$

Ne est le nombre d'espaces (ou dash) dans un gap, *GOP* est la pénalité d'introduire un nouveau gap et *GEP* est la pénalité pour étendre un existant gap. L'exemple suivant montre comment calculer le score d'un alignement en utilisant cette fonction. Soit l'alignement suivant des séquences S1, S2 et S3:

```
S1:AGCTAA-A
S2:A-CTAATA
S3:A--TCATA
```

Soit les scores de similarité pour les différentes situations : sc (A; B) = 2 pour A =B, sc(A;-) = sc(-;A) = -2 pour A ≠ '-', -1 sinon. A, B sont des lettres quelconques. Le score total de cet alignement est égal à:

$$Score\ (alignement) = Sc(S1,S2) + Sc(S1,S3) + Sc(S2,S3) = 16$$

Dans quelques problèmes d'alignement multiple de séquences, l'optimisation du SP peut engendrer des alignements incorrects quand il y a un grand nombre de séquences issues de quelques espèces et peu de séquences d'autres espèces. Pour cela des poids sont attribués aux séquences pour diminuer ce biais de tel sorte que les séquences convergentes reçoivent de petits poids est les séquences les plus divergentes reçoivent de grands poids [Altschul et al ,89]. Généralement, les poids sont calculés directement à partir de l'arbre guide construit initialement. Le score pondéré des paires WSP (Weighted Sum of Pairs) est calculé par la formule suivante:

$$\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} Wij.score(Si, Sj) \qquad (2.4)$$

L'inconvénient majeur de cette fonction est la difficulté d'établir les bons paramètres d'alignement comme la matrice de substitutions et les pénalités de gaps, qui peuvent être déterminés empiriquement par une large analyse d'alignements [Thompson et al, 94].

2.3.2 La mesure de l'entropie

Le score basé entropie est préféré dans les études statistiques et mathématiques des alignements [Nicolas et al, 02]. La mesure d'entropie en MSA est la somme d'entropie des colonnes. Pour chaque colonne, l'entropie est calculée par la formule suivante :

$$Entropie \quad (i) = -\sum_{a} c_{ia} \log(\ p_{ia}\ ) \tag{2.5}$$

Où c_{ia} est le nombre du caractère a dans la colonne *i*, p_{ia} est la probabilité du caractère *a* dans la colonne *i*.

$$p_{ia} = c_{ia} \Big/ \sum_{a'} c_{ia'} \tag{2.6}$$

Une colonne reçoit un zéro d'entropie si tous les caractères alignés dans la colonne sont identiques. Plus la colonne est variable, plus l'entropie est haute. L'entropie de colonne est maximum s'il y a des nombres égaux de tous les caractères possibles dans la colonne. Quand le score d'entropie est employé comme fonction objectif, le but est de réaliser l'entropie minimum.

2.3.3 Le score Consensus

Soit A[:][i] une colonne quelconque d'un alignement multiple *A*. La lettre x_i dénote la iéme consensus si le consensus-erreur est minimal. La concaténation des lettres consensus donne la séquence consensus. Le but est trouver l'alignement *A** qui minimise la somme des erreurs consensus sur toutes les colonnes. Le score consensus d'un alignement *c (A)* est défini par la formule suivante: [Reinert, 03]

$$c(A) = \sum_{i=1}^{l} \sum_{j=1}^{k} d(x_i, A[j][i]) \tag{2.7}$$

La figure 2.5 suivante montre un exemple de calcul d'un score consensus d'un alignement. Soit les scores suivants pour les différentes situations : d (A; B) = 2 pour A ≠ B, d(A;-) = d(-;A) = 1 pour A ≠ '-', 0 sinon (dans cet exemple le score d est une mesure de distance entre deux lettre contrairement à l'exemple précèdent).

```
              a1 = - G C T G A T A T A A C T
              a2 = G G G T G A T - T A G C T
              a3 = A G C G G A - A C A C C T
                   ------------------------
consensus   :      - G C T G A T A T A X C T
column value:      2 0 2 2 0 0 1 1 2 0 3 0 0 = 13
```

Figure 2.5. Le calcul de score Consensus d'un alignement.

Dans la séquence consensus, X dénote une lettre quelconque.

2.3.4 Le profil alignement

Soit *A* un alignement, le profil de *A* donne la fréquence relative de chaque lettre dans chaque colonne. L'alignement d'une chaîne *S* a un profil a pour but de calculer la somme pondérée des lettre de *S* au colonne du profile [Reinert, 03]. L'exemple de la figure 2.6 suivante montre le calcul du score d'un alignement en utilisant un profil. Ayant les scores de similarité suivants : s(A;B) = s(X;-) = s(-;X) = -1, s(A;C) = -3, s(B;C) = -2 Et s(A;A) = s(B;B) = s(C;C) = 2.

```
a1 = A B C - A      Profile: C1   C2   C3   C4   C5
a2 = A B A B A             A: .75        .25        .50
a3 = A C C B -             B:       .75        .75
a4 = C B - B C             C: .25   .25  .50        .25
                           -:             .25  .25  .25
-------------------------------------------------------
      column  column value
   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 - 0.5 *3 - 0.25*1   = -1.25
                                          ----------
                                            -0.25
```

Figure 2.6. Le calcul de score profil d'un alignement

2.3.5 La fonction T-COFFEE (Tree-based Consistency Objective Function for alignment Evaluation)

T-COFFEE [Notredame et al, 98] est une autre fonction intéressante pour évaluer un alignement. C'est un moyen simple et flexible pour effectuer un alignement en utilisant des sources de données hétérogènes. T-COFFEE procède d'abord par la génération d'une librairie de tous les alignements de paires de séquences possibles. Cette librairie est construite en utilisant des alignements globaux et locaux. Ensuite, elle calcule le degré d'identité entre l'alignement courant et la librairie d'alignements. Le score global donnant la qualité d'un alignement est donné par la formule suivante :

$$Score\_TCOFFEE = \frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} * Score(A_{ij})}{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} * Len} \qquad (2.8)$$

Où *N* est le nombre de séquences, *Len* est la longueur de l'alignement. W_{ij} est le pourcentage d'identité entre deux séquences alignées S_i, S_j. A_{ij} est l'alignement des séquences Si,Sj obtenu par projection à partir de l'alignement multiple obtenu. Et en fin, *score (Aij)* est le degré d'identité entre l'alignement A_{ij} et son correspondant dans la librairie d'alignements. Généralement, les programmes basés sur cette fonction donnent des alignements de bonne qualité comme le programme T-COFFEE, SAGA, etc. En effet, l'utilisation de la fonction T-COFFEE dans les méthodes d'alignement a donné des meilleurs résultats par rapport aux méthodes existantes. Notredamme et al ont démontré que la corrélation entre l'optimum de T-COFFEE et l'optimal biologique est plus grande parce que T-COFFEE utilise efficacement des

informations sur les séquences comme les poids des séquences[Notredame et al,98]. La figure 2.7 montre les étapes du calcul de la fonction T-COFFEE pour un exemple donné.

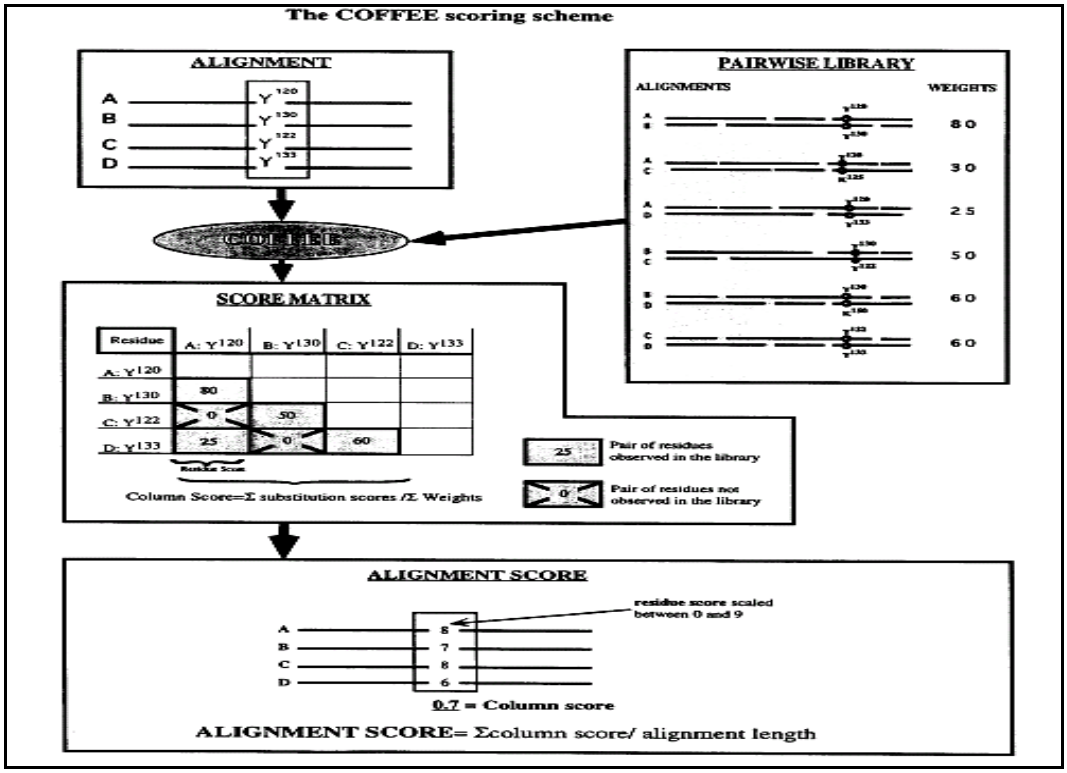


Figure 2.7. Les étapes de la fonction de score T-COFFEE [Notredame et al, 98].

2.4 Les approches de résolution du problème MSA

Le problème MSA a été démontré NP-difficile [Wang et al, 94]. Pour cela, plusieurs méthodes qui utilisent des différentes stratégies on été proposées pour résoudre ce problème. Mais aucune méthode n'a pu résoudre efficacement ce problème. L'enjeu est double, il faut trouver des solutions optimales dans des délais raisonnables. De point de vue la stratégie utilisée pour résoudre le problème MSA, on peut classer les méthodes d'alignement en quatre grandes classes : les méthodes exactes, les méthodes progressives, les méthodes itératives et les méthodes basées sur des modèles statistiques. Il existe encore un autre classement de ces méthodes basé sur le type de la méthode d'alignement de paires de séquences utilisée dans l'algorithme général: méthodes globales et méthodes locales. Les méthodes globales alignent les séquences du début à la fin. Elles sont basées sur l'algorithme de Needleman-Wunsch. Le score dans ces méthodes est défini par la somme des scores des pairs de résidus moins les pénalités de gaps. Le but est donc de maximiser ce score afin de trouver un bon alignement. Cependant la plupart des méthodes locales essaient de trouver un ou plusieurs motifs conservés partagés par toutes les séquences. Au cours des dernières années, plusieurs méthodes hybrides ont été développées. Elle combines entre les méthodes globales est les méthodes locales comme les méthodes basées segment (le programme DIALIGN) [Subramanian et al, 05].

2.4.1 Les méthodes exactes

On peut généraliser l'algorithme de la programmation dynamique de Needleman-Wunsh [Needleman et al, 70] pour l'alignement de paires de séquences aux alignements multiples de *n*

séquences en employant une table de score *n-dimentionnelle*. Suivant les indications de la figure 2.8, dans une table de score bidimensionnelle, la valeur dans une cellule est la forme dérivée de l'une de ses trois voisins or dans le cas d'une table de score tridimensionnelle, la valeur d'une cellule dépend de sept voisins. Pour *L* séquences de longueur N, la taille de la table est L^N, le temps de calcul de chaque cellule est 2^l-1 et le temps de calcul de chaque alignement de paires de séquences candidat est N(N-1)/2. Donc la complexité temporelle pour la programmation dynamique multidimensionnelle est $o(N^2 2^L N^L)!$. Cette approche est tellement gourmande en matière de ressources, elle devient impraticable pour N > 4. La figure 2.9 montre un exemple de table de score d'un alignement de trois séquences ainsi que le chemin optimal qui donne l'alignement idéal.

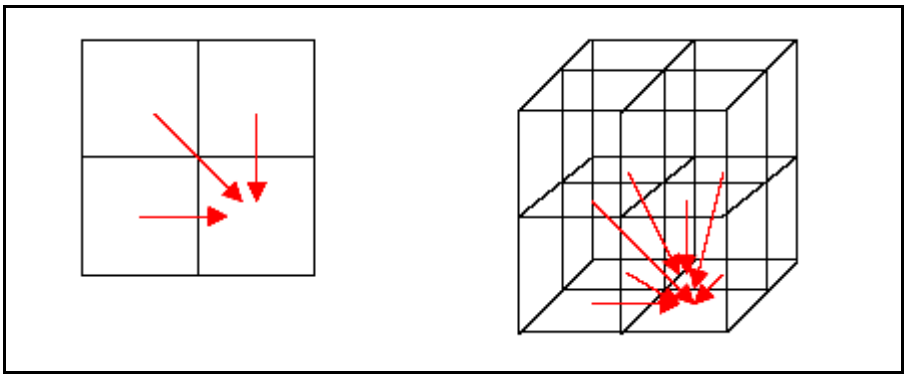


Figure 2. 8. Tables de score à deux et trois dimensions.

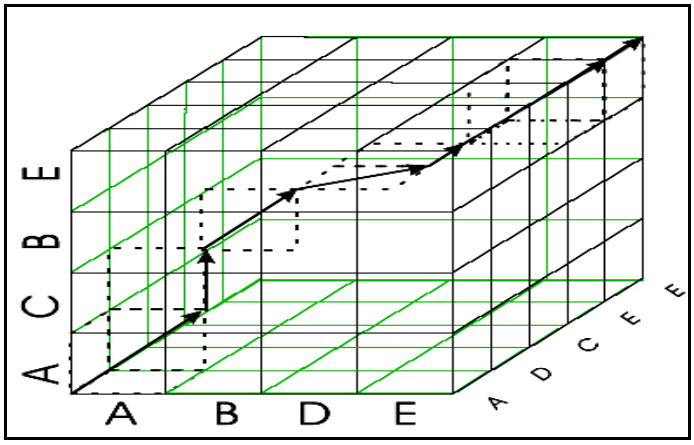


Figure 2. 9. Le trace back dans un alignement de trois séquences.

Plusieurs heuristiques on été utilisées pour rendre La programmation dynamique multidimensionnelle faisable pour l'alignement d'un nombre modéré de séquences avec des longueurs raisonnables. Une méthode appelé MSA a été mis en application basé sur l'optimisation de Lipman [Lipman et al, 89]. Plus tard, plusieurs améliorations on été apportées pour réduire la complexité temporelle et spatiale de MSA [Gupta et al, 95]. L'idée principale de l'algorithme de Lipman est la suivante: premièrement, le score SP pour n'importe quel paire de séquences extraite de l'alignement multiple optimal, devrait être inférieur au score SP optimal de l'alignement de paires de séquences. Deuxièmement, 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 n-dimensionnelles. Suivant les indications de la figure 2.10, ceci

peut réduire le temps de calcul considérablement. Les étapes principales de l'algorithme optimisé de MSA peuvent être trouvées dans [Lipman et al, 89].

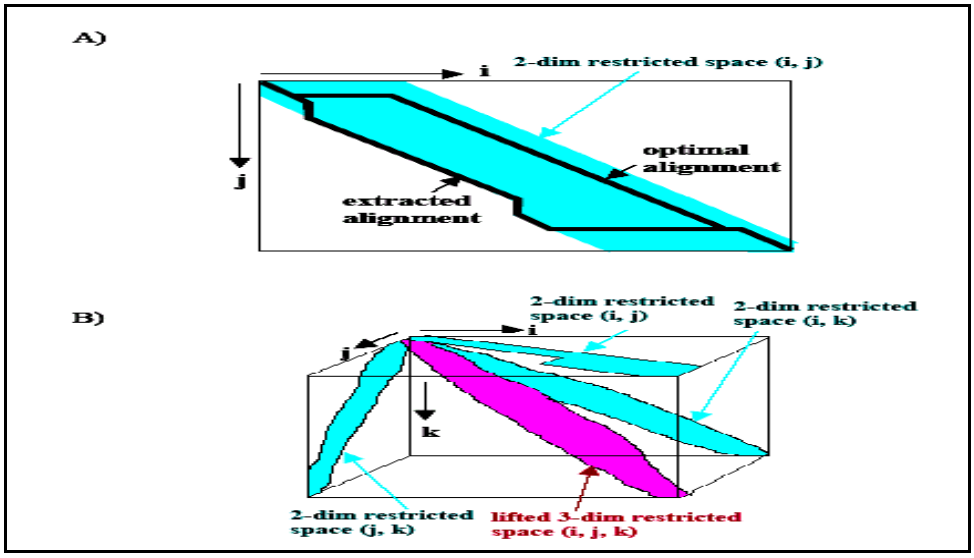


Figure 2. 10. Un exemple de l'utilisation de l'espace restreint dans le programme MSA.

MSA garantit une solution optimale ou proche de l'optimale. Cependant, la solution est trop coûteuse en matière de temps CPU et la taille de mémoire utilisée. En effet, elle peut facilement atteindre un niveau prohibitif selon le nombre, les longueurs et la diversité des séquences. En utilisant un mini-ordinateur géant avec 4 gigaoctets de mémoire physique, MSA peut aligner 20 séquences de la phospholipase A2 qui a approximativement 130 caractères [Nicolas et al, 02]. Récemment, Une autre heuristique qui a maintenu les méthodes exactes encore populaires, est l'algorithme DCA décrit par Stoye [Stoye et al, 97]. DCA est un algorithme basé sur l'idée "*divide to conquer*" (figure 2.11). Le principe consiste à découper les séquences en sous ensembles de segments. Ces derniers doivent être aussi petits pour qu'ils puissent être traités par la méthode MSA. Les sous alignements produits sont ensuite rassemblés par l'algorithme DCA.

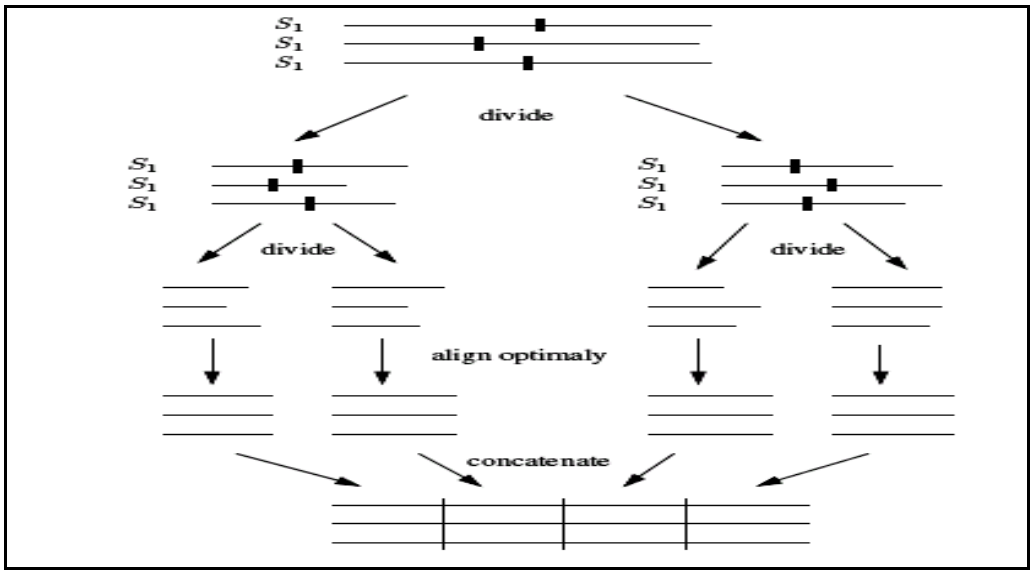


Figure 2. 11. les étapes de l'algorithme DCA.

2.4.2 Les méthodes progressives

Vu la complexité des méthodes exactes, plusieurs heuristiques ont été proposées pour surmonter l'impraticabilité de ces méthodes. Parmi les méthodes les plus répandues et les plus populaires en MSA, on trouve les méthodes progressives. Ces méthodes sont simples, rapides et donnent généralement des solutions acceptables. L'algorithme de l'alignement progressive a été premièrement décrit par Hogeweg [Hogeweg et al, 84] et plus tard redéfini par Feng [Feng et al, 87] et Taylor [Taylor, 88]. Les méthodes d'alignement multiples les plus employés couramment sont basés sur l'exécution de cet algorithme comme la méthode CLUSTAL W [Thompson et al, 94] qui est considérée comme la méthode standard d'alignement multiple. L'idée principale des méthodes progressives est tirée de l'algorithme glouton. L'alignement multiple commence par la construction d'une succession des alignements de paires de séquences. D'abord, deux séquences sont choisies et alignées par l'algorithme standard d'alignement de paires de séquences. Ensuite, une troisième séquence est choisie et alignée au premier alignement. Ce processus est réitéré jusqu'à ce que toutes les séquences soient alignées. Bien que tous les algorithmes progressifs adoptent la même stratégie de base, ils appliquent des stratégies différentes pour choisir l'ordre d'alignement des séquences. Ils appliquent également des modifications à l'algorithme de programmation dynamique pour aligner deux sous-groupes de séquences. L'exemple suivant montre les étapes de construction d'un alignement multiple en utilisant une simple méthode progressive : Ayant les 5 séquences suivantes :

```
S1= ATTCGGATT
S2= ATCCGGATT
S3= ATGGAATTTT
S4= ATGTTGTT
S5= AGTCAGG
```

La méthode d'alignement progressive commence d'abord par l'alignement de deux séquences par exemple S1 et S2 :

```
S1: A T T C G G A T T
S2: A T C C G G A T T
```

Ensuite on ajoute une troisième séquence à l'alignement précèdent (soit S3). S3 sera alignée avec la séquence la plus proche par exemple S1. L'alignement de S1 et S3 insère deux gaps en S1, alors on doit propager les deux gaps dans la séquence S2 (le principe *"once a gap, always a gap"*). Les positions des gaps ajoutés sont conservées définitivement.

```
S2: A T C C G G A T T - -
S1: A T T C G G A T T - -
S3: A T G - G A A T T T T
```

De la même façon on ajoute les séquences S4 et S5.

```
S2: A T C C G G A T T - -
S3: A T G - G A A T T T T
S1: A T T C G G A T T - -
S4: A T G T T G - T T - -
```

Apres l'alignement de S5, on aura le résultat d'alignement multiple de séquences suivant :

```
S2: A T C C G G A T T - -
S3: A T G - G A A T T T T
S4: A T G T T G - T T - -
S1: A T T C G G A T T - -
S5: A G T C A G G - - - -
```

La complexité des méthodes progressives est généralement inférieure à $o(N^3L^2)$ dans le pire des cas, (L est le nombre des séquences et N est la taille de la plus grande séquence). Elle est nettement inférieure à celle des méthodes exactes $o(N^2 2^L N^L)$. Actuellement, La technique d'alignement multiple la plus rapide est la méthode MAFFT (table 2.1). MAFFT utilise un nouvel algorithme pour l'alignement de paires de séquences basé sur la transformation de Fourier [Katoh et al, 02].

ALIGNER	complexité
ClustalW	$o(N^2L^2)$
MAFFT	$o(N^2)$
T-COFFEE	$o(N^2L^2)+o(N^3L)+o(N^3)+o(NL^2)$
MUSCLE	$o(N^3L)$

Table 2.1. La complexité de quelques programmes progressifs.

Ø Alignement progressif de Feng-Doolittle

L'algorithme progressif d'alignement de Feng et Doolittle [Feng et al ,87] est la méthode la plus utilisée dans l'alignement de séquences. Il est la base de plusieurs programmes de MSA comme CLUSTAL et ALIGN. L'ordre d'alignement de séquences est basé sur un arbre guide. La seule différence entre ces programmes réside seulement dans la stratégie utilisée pour créer l'arbre guide. L'algorithme de base de Feng et Doolittle est composé des étapes suivantes :

1. Calculer tous alignements pairs possibles entre les séquences.
2. Convertir les scores d'alignements en distances et calculer l'arbre guide en utilisant la matrice distances.
3. Effectuer l'alignement multiple en démarrant par les séquences les plus proches puis on ajoute les séquences une par une selon l'ordre donné par l'arbre guide.

Ø Clustal : Le meilleur programme progressive pour MSA

CLUSTALW [Thompson et al, 94] est le programme d'alignement multiple le plus populaire. CLUSTALW est basé sur l'alignement progressif de Feng et Doolittle enrichie par des améliorations importantes. CLUSTALW utilise deux pénalités de gaps, une pénalité d'ouverture de gaps et une pénalité d'extension de gaps. CLUSTAL W calcule dynamiquement les pénalités de gaps pour adapter les valeurs pour chaque ensemble de séquences. CLUSTALW donne la possibilité d'augmenter la probabilité d'avoir des gaps dans les régions hydrophiles (les résidus hydrophiles peuvent être spécifiés), correspondant souvent à des boucles ou des "coils" (régions dans lesquelles les gaps peuvent être plus communément rencontrés). CLUSTALW permet aussi la non pénalisation des gaps en extrémité de séquence. ClustalW change automatiquement la matrice de score utilisée (dans la même série de matrice,

telle que la série de BLOSUM). En plus, CLUSTAL W assigne un poids à chaque séquence en se basant sur l'arbre guide. Ceci contribue à la réduction des erreurs du au surreprésentation d'une sous-famille dans l'alignement. Donc, les séquences les plus proches reçoivent de petits poids or les séquences les plus divergentes reçoivent de grands poids. L'algorithme CLUSTAL suit les étapes suivantes (figure 2.12.a, 2.12.b):

1) Alignement de paire de séquences et matrice de distance

Un score de similitude est calculé pour chaque paire de séquences. CLUSTAL dispose de deux méthodes pour alignement pair : alignement rapide selon un algorithme d'alignement approximatif global rapide et un algorithme d'alignement lent en utilisant un algorithme de programmation dynamique. Les scores de similarité entre les séquences sont transformés ensuite en matrices de distances.

2) l'arbre guide

La méthode construit alors un dendrogramme (un "arbre guide") (figure 2.12.b), c'est à dire un arrangement traduisant les relations globales de parenté entre les séquences : celui-ci indique l'ordre à partir duquel l'alignement multiple graduel sera établi. L'arbre guide est construit en utilisant la matrice de distances crée dans l'étape précédente et la méthode de Neighbour-Joining [Saitou et al, 87] pour la construction des arbres phylogénétiques.

3) L'alignement progressif

La procédure de base dans cette étape utilise une série d'alignements pairs pour aligner les groupes de séquences pour construire graduellement l'alignement multiple. L'ordre d'alignement est donné par l'arbre guide. On procède à partir des feuilles jusqu'à la racine de l'arbre guide. A chaque étape un algorithme de programmation dynamique est appliqué avec une matrice de poids et des pénalités pour l'ouverture et l'extension des gaps.

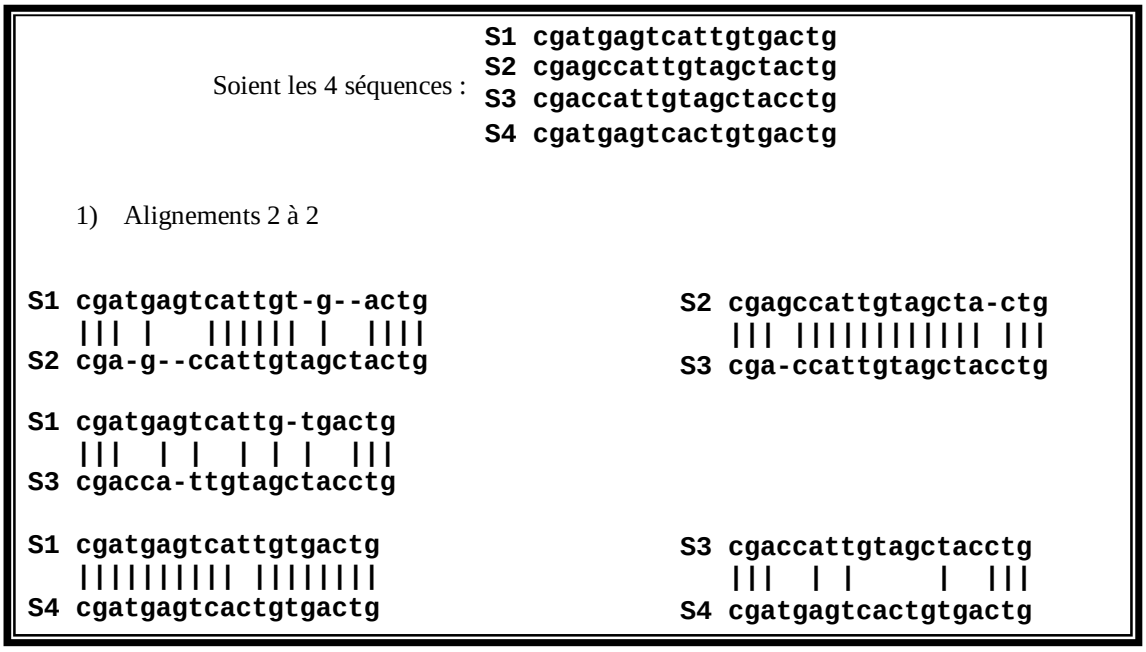
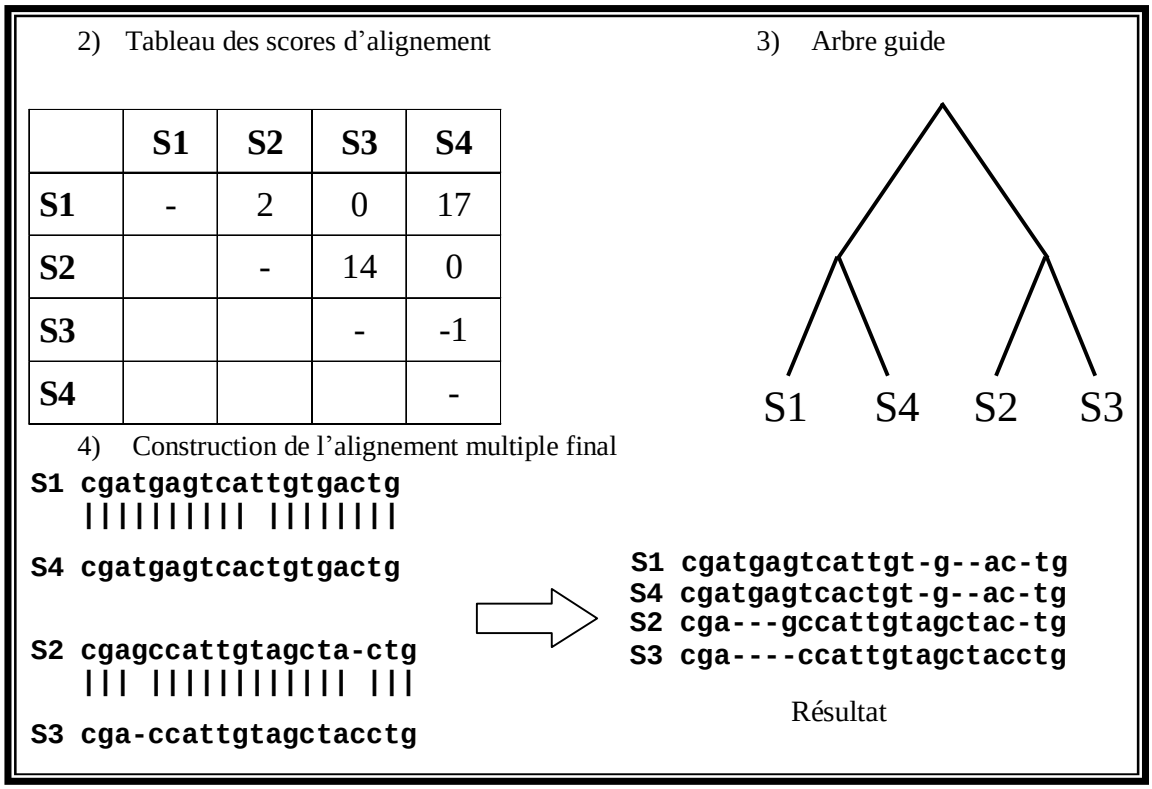


Figure 2.12.a. Les Etapes d'alignement d'un ensemble de séquences par Clustal. 1:l'alignement de tous les paires de séquences possibles.



	S1	S2	S3	S4
S1	-	2	0	17
S2		-	14	0
S3			-	-1
S4				-

Figure 2.12. b. Les Etapes d'alignement d'un ensemble de séquences par Clustal. 2:transformation des score d'alignement en distances, 3: construction de l'arbre guide, 4: construction de l'alignement final

Il y a quelques cas spécifiques où CLUSTALW est connu pour avoir des problèmes.

- Si les séquences sont seulement semblables dans quelques petites régions, alors que les parties les plus grandes ne sont pas semblables, dans ce cas CLUSTALW peut avoir des problèmes pour aligner toutes les séquences correctement. Ceci s'explique par le fait que CLUSTALW est une méthode d'alignement global et non locale. Dans ce cas, il est préférable de couper les parties semblables avec un autre outil (éditeur de texte).
- Si une séquence contient une grande insertion comparée au reste, alors il peut y avoir des problèmes, pour la même raison que le point précédent.
- Si une séquence contient un élément réitéré (tel qu'un domaine), cependant une autre séquence contient seulement une copie de l'élément. Il y a beaucoup de protéines qui contiennent des copies multiples et très semblables d'un domaine, ainsi on devrait trouver d'autres programmes plus sophistiqués pour ce problème.

Généralement, les méthodes progressives présentent les inconvénients majeurs suivants :

- la solution est très dépendante du premier alignement de paires. C'est-à-dire que le mauvais choix des premières séquences à aligner va donner des alignements de mauvaises qualités.
- La propagation des erreurs produites dans les premiers alignements aux alignements suivants.
- Le choix des paramètres est très fastidieux. Il n'existe pas une méthode universelle pour régler les paramètres d'alignements tel que les pénalités des gaps et la matrice de substitutions utilisée.

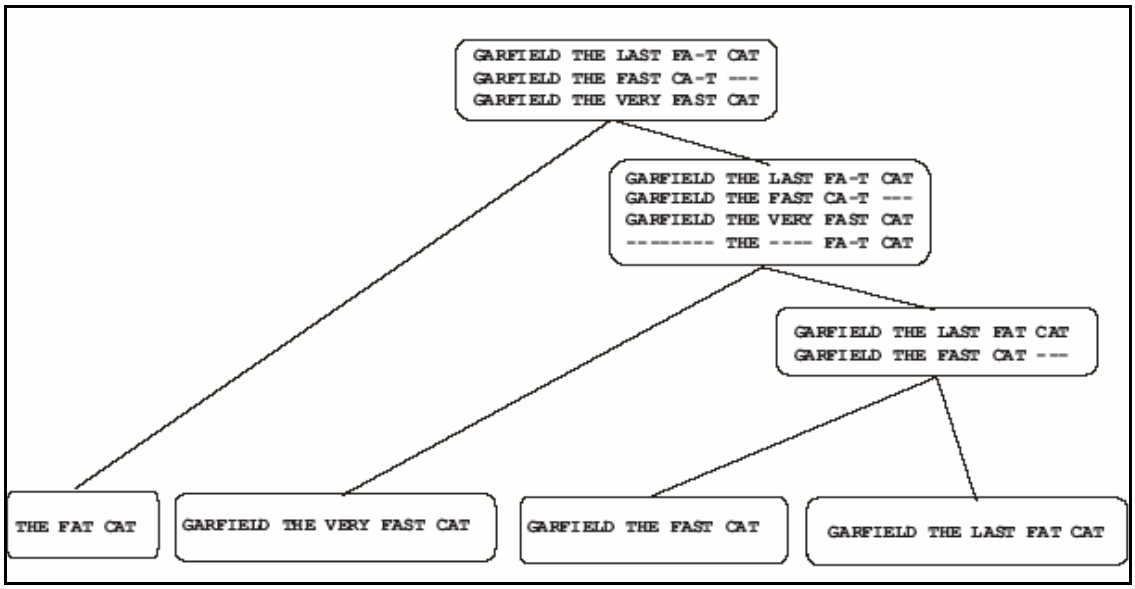


Figure 2.13. La limite des méthodes progressives [Notredame, 02].

La figure 2.13 montre comment une stratégie progressive d'alignement peut être trompée. Dans l'alignement initial les séquences 1 et 2, CLUSTAL W a le choix entre aligner CAT avec CAT et faire un gap interne ou faire une substitution entre C et F et avoir un gap terminal. Puisque les gaps terminaux sont moins pénalisants que les gaps internes, les arrangements de score de CLUSTAL W préfère l'ancien. À la prochaine étape, quand la séquence supplémentaire est ajoutée, il s'avère que l'alignement des deux CAT à l'étape précédente aurait mené à un meilleur SP score [Notredame, 02].

2.4.3 Les méthodes itératives

Les méthodes itératives on été introduites pour corriger les problèmes des méthodes progressives. Le principe est simple : une solution initiale est obtenue ensuite un ensemble de raffinements est appliqués sur cette solution. Les nouvelles solutions sont évaluées en utilisant une fonction objectif. Le processus est réitéré jusqu'à la satisfaction des conditions de terminaison. Les méthodes itératives peuvent être divisées en deux classes dépendantes de la stratégie de raffinements utilisée [Notredame, 02] : déterministes et stochastiques.

2.4.4 Les méthodes itératives déterministes

L'idée consiste à itérativement extraire une séquence de l'alignement multiple et la réaligner ensuite au reste des séquences. Le processus est réitéré jusqu'il n'y a pas plus d'améliorations possibles. Plusieurs méthodes itératives utilisent cette stratégie. Seulement, la seule variation consiste dans le choix des séquences à réaligner. Dans la méthode AMPS [Barton et al, 87], les séquences sont choisies selon l'ordre d'entrée et réaligner une à une. Par ailleurs, dans l'algorithme de Berger et Munsen [Berger et al, 91] les séquences sont divisées en deux blocs et le choix des séquences est fait d'une manière aléatoire. Le meilleur algorithme itératif déterministe est PRRP [Gotoh, 1996] proposé par Gotoh. PRRP a donné des bons résultas dans les tests. PRRP optimise itérativement la fonction objectif WSP. La pénalité des gaps suit une fonction affine. Les poids des séquences et les alignements sont optimisés simultanément. L'algorithme de PRRP est constitué de quatre étapes : La construction de l'arbre

phylogénétique, le calcul des poids, alignement multiple de séquences et la comparaison de poids. L'algorithme se termine lorsque le poids total de l'arbre converge.

2.4.5 Les méthodes itératives stochastiques

Plusieurs méthodes itératives stochastiques on été proposées pour résoudre le MSA. Ces méthodes sont basées sur des différentes approches stochastiques tel que le recuit simulé RS [Kim et al, 94]. Les algorithmes évolutionnaires qui incluent les algorithmes génétiques [Notredame et al, 96], [Nguyen et al, 02] et [Thomsen et al, 02], l'algorithme tabou [Riaz et al, 04] et les algorithmes basés sur le modèle caché du Markov (Hidden Markov Model) (HMM) [Durbin et al, 98]. Les méthodes basées sur le recuit simulé étaient les premières méthodes itératives stochastiques décrites pour aligner simultanément un ensemble de séquences. L'alignement multiple de séquence par RS commence par une solution aléatoire. Puis, un ensemble de perturbations est appliqué sur la solution initiale par exemple la mutation d'un gap ou l'introduction d'un autre. L'alignement résultat est évalué en utilisant une fonction objectif. Le nouvel alignement est accepté si il est meilleur que le précèdent ou avec une probabilité du recuit. Le processus est réitéré jusqu'au gel du système. Les méthodes basées sur le recuit simulé donne généralement une solution optimale mais leur inconvénient majeur est la lenteur. Cette limitation sérieuse rend l'utilisation du recuit simulé pour le MSA impraticable. Pour cela d'autres méthodes basées sur les algorithmes évolutionnaires et les modèles cachés de Markov ont été proposées pour surmonter les limites de RS. Les algorithmes génétiques sont utilisés avec succès pour traiter le problème d'alignement multiple. Un algorithme génétique AG est une technique d'optimisation inspirée du phénomène de la sélection naturelle [Holland, 1975], [Whitley, 1993]. Cette technique est très utile pour les problèmes d'optimisation combinatoire où les méthodes traditionnelles échouent. L'avantage principal des AGs est simplement, on a besoin d'une fonction de fitness pour évaluer les différentes solutions. Un autre avantage est son parallélisme ce qui permet une implémentation efficace dans les architectures parallèles. Le meilleur algorithme basé génétique est l'algorithme SAGA [Notredame et al, 96]. SAGA est considéré comme une référence dans le domaine d'alignement multiple et c'est très intéressant de faire un bref survol sur la structure et le fonctionnement de cet algorithme.

Ø SAGA : un algorithme génétique pour l'alignement multiple de séquences

SAGA (Sequence Alignment by Genetic Algorithm) de Notredame et al [Notredame et al, 96] est un algorithme génétique dédié pour résoudre le problème d'alignement multiple. Il suit les mêmes principes de l'algorithme génétique simple décrit par Goldberg. Chaque population est constituée d'un ensemble d'alignements (individu). Chaque individu est représenté par une matrice dont les lignes représentent les séquences alignées. Un ensemble d'opérations de mutations et de croisement est appliqué sur la population afin de diversifier la recherche. L'alignement trouvé dans chaque itération est évalué en utilisant une certaine fonction objectif. Une caractéristique de SAGA est sa capacité d'optimiser plusieurs types de fonctions objectifs. Le pseudo code du SAGA est le suivant

```
Initialisation    1. Create G0
Evaluation        2. Evaluate the population generation n (Gn)
                  3. If the population is stabilised then END
                  4. Select the individuals to replace
                  5. Evaluate the expected offspring (EO)
Breeding          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
```

Dans l'étape `Initialisation,` une population G0 est crée aléatoirement. Cette population initiale se compose d'un ensemble d'alignements contenant seulement des gaps terminaux. SAGA utilise une population de taille de 100. Pour créer un de ces alignements, un offset aléatoire est choisi pour toutes les séquences et chaque séquence est déplacée vers la droite en s'accordant à son offset. Les alignements initiaux de la génération zéro sont les parents des enfants utilisés pour peupler la génération une. Dans l'étape `Evaluation,` la fitness de chaque alignement est calculée selon une fonction objectif. SAGA est très flexible, n'importe quelle fonction objectif peut être utilisée. Dans l'étape `Breeding`, un ensemble d'individus est sélectionné pour les opérations de mutations est de croisements. Les mutations insèrent ou décalent des gaps tandis que les croisements combinent entre deux alignements (figure 2.14). De façon générale, 20 opérateurs co-existent dans SAGA et concourent pour être utilisés. La méthode ne garantit pas l'optimalité cependant, les résultats ont montré que SAGA est aussi bon que MSA de point de vue mathématique sur 13 ensembles de tests. Mais le grand inconvénient de SAGA est le temps d'exécution qui peut atteindre des jours pour les séquences de grande taille.

Ø Les limites des algorithmes génétiques pour le MSA

Nous avons vu que les AGs ont prouvé leur capacité pour résoudre les problèmes complexes d'optimisation avec un niveau raisonnable d'exactitude. Ceci indique clairement l'importance et l'intérêt de ces méthodes dans le domaine de l'analyse de séquences. Cependant, une fois appliqué à ce type de problèmes, les AGs souffrent de deux inconvénients principaux : ils sont très lents et incertains. L'incertitude veut dire qu'un AG ne fournisse pas forcement deux fois la même réponse, dû à la nature stochastique du processus d'optimisation et à la difficulté de l'optimisation. Ceci peut avoir des conséquences graves si l'alignement est utilisé par un biologiste comme un outil de prévision et décision dans un laboratoire. En effet, SAGA atteint son meilleur alignement après plusieurs exécutions. L'autre inconvénient des AGs est la lenteur. Cet inconvénient empêche SAGA d'être une partie clé dans les projets d'alignement où des millions d'alignements à établir car les biologistes veulent des alignements optimaux dans peu de temps.

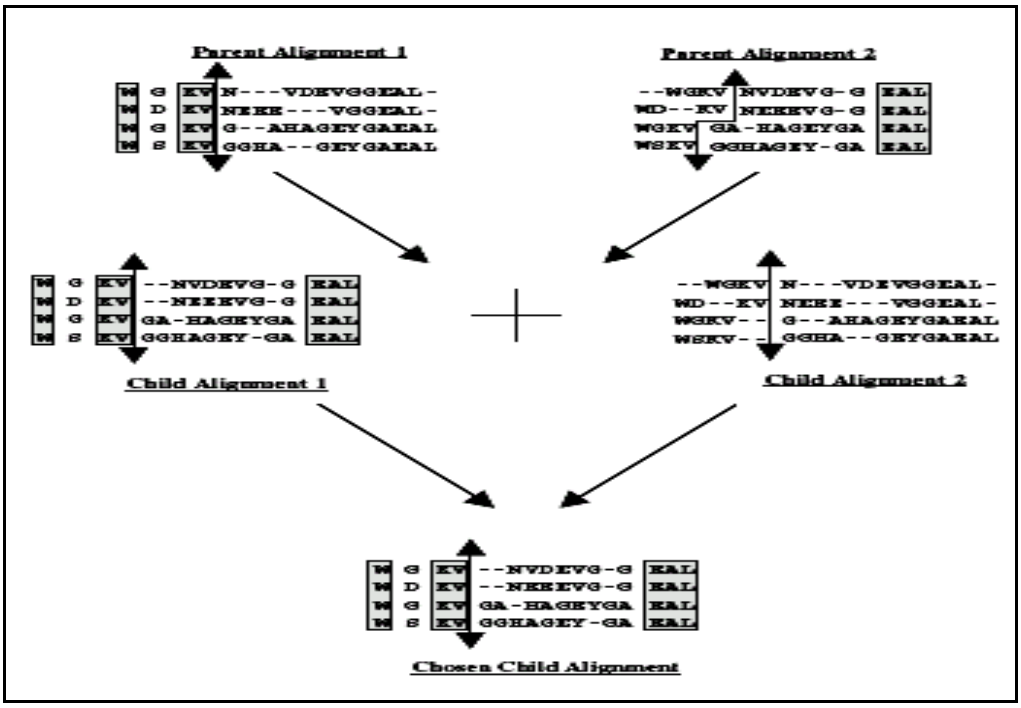


Figure 2.14. Un exemple de croisement dans SAGA [Notredame et al, 96].

On trouve également d'autres méthodes itératives basées sur des modèles statistiques et probabilistes comme le modèle caché de Markov HMM. Dans la suite on va décrire les méthodes basées sur le modèle HMM.

2.4.6 Les méthodes d'alignement basé sur un modèle statistique

Contrairement aux méthodes précédentes, ce type de méthodes n'utilise pas les algorithmes de la programmation dynamique pour effectuer un alignement. Elles sont basées sur des modèles statistiques tel que le modèle caché de Markov [Durbin et al, 98].

Ø Le modèle HMM pour MSA

Un modèle caché de Markov emploie des statistiques et des informations préalables sur un ensemble de séquences pour créer toutes les combinaisons possibles des identités, des disparités, et gaps afin de créer un alignement. Les méthodes basées HMMs exigent un grand nombre de séquences (en général 20-100) afin que le modèle fonctionne correctement. La différence entre les méthodes précédentes d'alignement multiple et le modèle HMM est que les pénalités d'insertion/suppression et l'ordre des séquences ne sont pas exigés. Les résultats des méthodes basées sur les modèles cachés de Markov sont comparables aux autres programmes d'alignement multiple de séquence [Mount, 2001]. Aujourd'hui le meilleur programme pour l'alignement multiple est basé sur un modèle HMM, PROBCONS [Brundno et al, 04]. En particulier, un certain modèle caché structuré de Markov est appelé un profil HMM. Un profil est employé pour construire des alignements multiples. En supposant qu'un alignement multiple correct est donné. En utilisant cet alignement, un profil HMM peut être construit et utilisé pour trouver et marquer les identités potentielles des nouvelles séquences. La structure standard d'un profil HMM se compose d'une série de N états de matches identiques séparés par des probabilités de transition de 1 (figure 2.15). Le modèle traite les insertions et les suppressions

séparément. Le modèle standard est illustré ci-dessous, où les états "match" représentent les positions conservées dans la famille de séquences. Les états "insert" représentent les sous-séquences qui ont été insérées dans quelques membres de la famille et les états "delete" sont des états silencieux représentant les sous-séquences qui ont été supprimées dans quelques membres de la famille [Durbin et al,98].

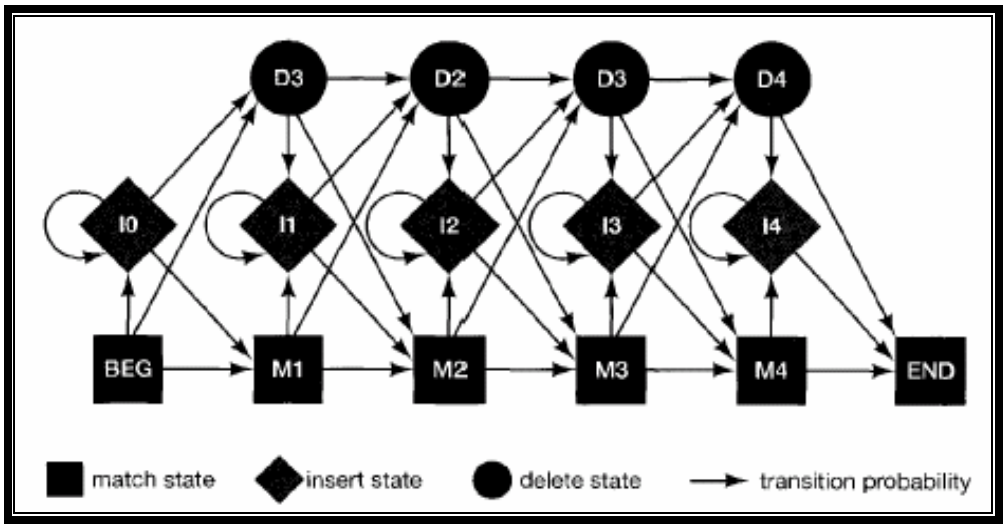


Figure 2.15. La structure d'un modèle HMM.

Pour un alignement donné de séquences, le chemin le plus probable à travers le modèle est déterminé par un algorithme de décodage. L'algorithme de Viterbi trouve le chemin le plus probable à travers le profil HMM récursivement.

Un alignement multiple de séquence est créé en calculant un alignement de Viterbi pour chaque séquence. Les résidus alignés sur le même état de match du profil HMM sont alignés dans les colonnes. Ceci provoque une différence importante entre les alignements multiples créés par HMM et ceux produits par les méthodes traditionnelles. Le choix d'où ou comment mettre les résidus "insert" dans l'alignement est arbitraire. Les états inserts représentent les segments d'une séquence qui sont inconvertissable et pas clairement alignées. Selon Durbin, le HMM est biologiquement la plus réaliste vue de l'alignement multiple de séquences [Durbin et al, 98].

On distingue dans cette classe des méthodes itératives comme le programme HMMT qui utilise l'algorithme de recuit simulé pour maximiser la probabilité que le modèle HMM représente la séquence alignée [Eddy, 95], et les méthodes progressives comme PROBCONS dont le fonctionnement est décrit dans ce qui suit.

2.4.7 Un algorithme progressif basé sur un modèle HMM

PROBCONS [Brundno et al, 04] est une nouvelle méthode progressive basée sur le modèle caché de Marcov HMM. PROBCONS est actuellement la méthode la plus précise pour l'alignement multiple de séquences. Elle a donné les meilleurs résultats dans les tests par rapport aux autres packages de MSA. PROBCONS diffère des autres méthodes HMM par l'utilisation d'un algorithme de *la précision maximum prévue* au lieu de l'algorithme de Verterbi et d'une nouvelle fonction objectif : *la transformation probabiliste consistante*. Cette nouvelle fonction incorpore les informations de l'alignement multiple conservées pendant

l'alignement de paire de séquences. PROBCONS utilise le modèle HMM (figure 2.16) pour spécifier la distribution probabiliste sur tous les alignements entre les séquences. Les émissions probabilistes correspondent à la traditionnelle matrice score de substitutions basée sur la matrice BLOSUM62. Les probabilités de transition qui correspondent aux pénalités de gaps sont déduites en utilisant une prévision maximisée EM (*expectation maximisation).* La figure 2.16 montre l'alignement basé HMM entre deux séquences x et y. L'état M émet deux lettres, une de chaque séquence et correspondent aux deux lettre alignées. L'état L émet une lettre associée à un gap dans la séquence x. de la même façon l'état I_y émet une lettre dans la séquence y qui est alignée avec un gap. Trouver l'alignement optimal selon l'algorithme de Virterbi correspond à appliquer l'algorithme de Needleman-Wunsch avec des paramètres appropriés. La fonction de logarithme de probabilité d'émission à M correspond à la matrice score de substitution. Les paramètres de la fonction affine de gaps peuvent être déduits à partir de probabilités de transition δ et ε [Brundno et al, 04]. Les détailles de L'algorithme de PROBCONS peuvent être trouvées dans [Brundno et al, 04].

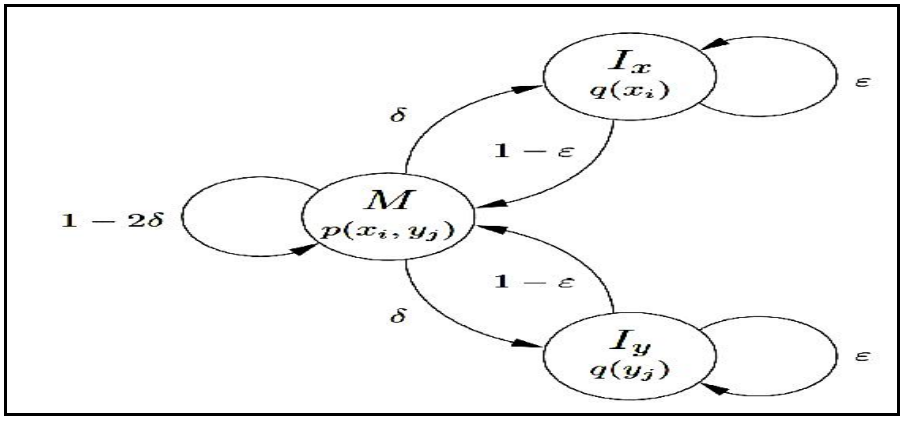


Figure 2.16. L'alignement pair base HMM entre deux séquences x et y. l'état M émet deux lettres.

2.5 L'évaluation des alignements

Il existe essentiellement deux méthodes pour l'évaluation des alignements : les méthodes statistiques [karlin et al, 93] et la méthode d'évaluation par des tests références (bases d'alignements de références). Cette dernière est la plus utilisée pour l'évaluations des algorithmes d'alignement.

2.5.1 Mesures statistiques de la qualité d'un alignement

Nous savons maintenant obtenir un alignement, comment pouvons nous estimer la qualité de cet alignement et de ses scores? Il existe plusieurs approches pour déterminer la qualité d'un alignement. La première approche est la technique bayésienne qui est basée sur la comparaison de différents modèles. La seconde approche est basée sur une approche statistique en calculant la chance d'avoir un score d'identité plus grand que l'observé. Cette manière considère la signification dans plusieurs situations. Pour déterminer si un alignement est statistiquement significatif nous pouvons faire un test de permutation.

(a). Réarrangement aléatoire des résidus dans une ou toutes les séquences.

(b). Aligner les nouvelles séquences permutées.

(c). Marquant les scores obtenus pour cet alignement.

La répétition de ces étapes un grand nombre de fois produit une distribution des scores d'alignement pour lesquels pourrait être prévu les séquences aléatoirement réarrangées. Puis, nous pouvons regarder à la distribution du maximum de N scores d'identité des séquences aléatoires et indépendantes. Si la probabilité que ce maximum est plus grand que les meilleurs scores observés est petite, l'observation peut être considérée comme significative. Selon la règle de Doolittle pour des séquences de protéine : une identité plus de 25% suggérera l'homologie, moins que 15% serait incertain et pour les cas entre 15- 25%, un argument statistique solide serait exigée. Cependant, le pourcentage de l'identité est défini après l'alignement et dépend à peine de la méthode et de ses paramètres. Si un alignement est plein de gaps, la similitude est surestimée. N'importe quel programme standard produira une statistique indiquant le niveau de confiance qui devrait être attaché à un alignement [Lambert et al, 03].

2.5.2 Les bases de Tests pour l'évaluation d'un algorithme d'alignement

Une autre méthode amplement utilisée pour évaluer la qualité biologique des algorithmes d'alignement est l'utilisation d'un grand nombre d'alignements précis de référence comme des tests. Cette méthode consiste à déterminer la capacité des programmes d'alignement face à des familles de séquences dont l'alignement biologique optimal est connu. Un alignement test doit avoir des séquences qui partagent des similarités structurelles signifiantes. Plusieurs bases d'alignements de référence ont été élaborées comme BAliBASE [Thompson et al, 99a], PREFAB [Edgar, 04] SABmark [Van Waller et al, 04],OXbench [Raghava et al,03], etc.

2.5.3 Comparaison de programmes

Thompson et al [Thompson et al, 99b] ont effectué un énorme travail afin d'évaluer et comparer 10 méthodes d'alignement. Un des objectifs de cette étude était d'établir un système objectif de benchmark qui peut être employé pour comparer, évaluer et améliorer les méthodes d'alignement multiples. Pour cela ils ont crée une base d'alignements de références BALIBASE (table 2.2). Les alignements de la base BALIBASE ont fourni de vrais tests contenant des protéines ou des modules dont les structures tridimensionnelles sont déterminées. Les dix méthodes utilisées dans leur étude utilisent différentes stratégies pour effectuer des alignements multiples (figure 2.17). Nous trouvons des méthodes progressives globales utilisant un algorithme global de la programmation dynamique comme CLUSTALX et MULTALIGN, des méthodes progressives locales utilisant un algorithme local de la programmation dynamique comme PIMA, des méthodes progressives itératives comme PRRP et DIALIGN, une méthode itérative basée AG comme SAGA et on trouve également une méthode basée sur le modèle caché de Markov HMM : HMMR.

2.5.4 La base de référence BALiBASE

Pour déterminer l'efficacité des méthodes, Thompson et al ont crée une base d'alignements de références BALiBASE. Cette base contient 142 alignements références, divisés en cinq ensembles de références. Chaque ensemble contient au moins 12 alignements représentatifs qui

représentent les problèmes les plus rencontrés lors de l'alignement de vraies familles de protéines (table 2.2). Les alignements des séquences partageant le même pli tridimensionnel ont été validés pour assurer l'alignement des résidus fonctionnels et conservés. Des blocs noyaux (*core blocks)* sont définis pour chaque alignement en tant que régions qui peuvent être sûrement alignées. Les blocs noyaux (représentant 58% des résidus en alignements) excluent spécifiquement les régions tridimensionnelles ambiguës ou non-superimposable telles que les structures secondaires distinctes, les limites secondaires indépendantes de structure ou les régions boucles structurellement incertaines [Thompson et al, 99a].

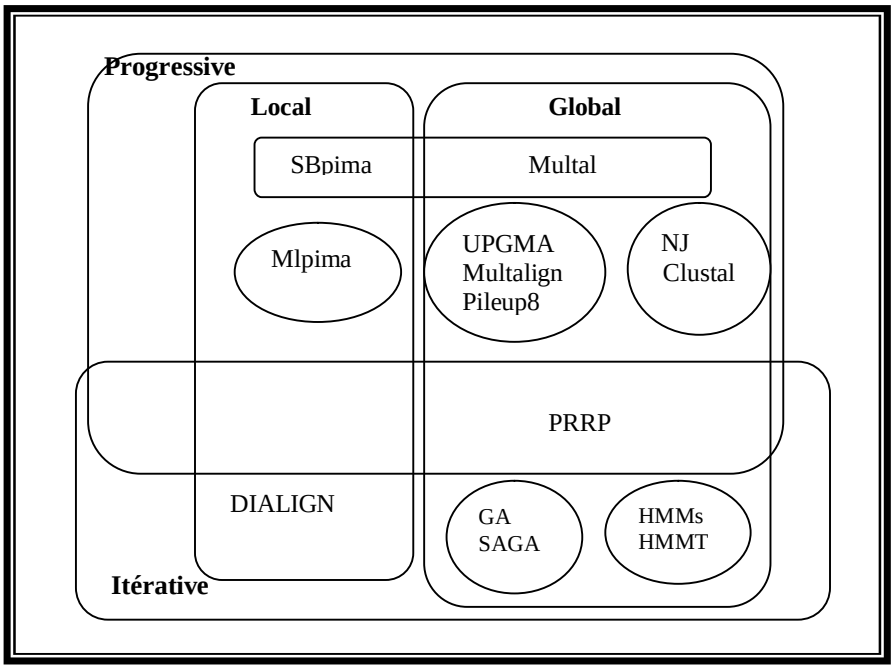


Figure 2.17. Les différents programmes utilisés dans l'étude comparative.

Les alignements de la référence 1 se composent d'un nombre restreint de séquences équidistantes de longueur semblable. Les séquences ne contiennent pas des extensions ou des insertions. La référence 2 contient des alignements d'une famille contenant des séquences étroitement liées avec une identité >25% plus des séquences éloignées de la famille avec une identité <20% et qui partagent un pli commun (séquences orphelines).cette référence est conçue pour évaluer l'exactitude des programmes d'alignement selon deux critères :

- La stabilité de l'alignement de famille quand des séquences orphelines sont introduites dans les ensembles de séquences.
- La qualité de l'alignement des séquences orphelines.

La référence 3 est utilisée pour démontrer la capacité des programmes d'aligner correctement les familles divergentes équidistantes en un seul alignement. Les alignements de référence se composent de un jusqu'à quatre familles, avec une identité <25% entre deux séquences quelconques de familles différentes.

Les références 4 et 5 contiennent des séquences avec de larges extensions N/C-terminal ou des insertions internes, respectivement. Afin d'évaluer la capacité d'un programme d'identifier la présence des insertions, les blocs de noyau dans BAliBASE définissent seulement les motifs les

plus conservés flanquant l'extension/insertion. Ces essais ne sont pas conçus pour juger la qualité globale d'un alignement [Thompson et al, 99a].

Référence	Courte (<100 résidus)	Moyenne (200-300 résidus)	Longue (>400 résidus)
Référence 1: séquences équidistantes de longueurs similaires			
V1 (<25% identité)	7	8	8
V2 (20-40% identité)	10	9	10
V3 (>35% identité)	10	10	8
Référence 2: famille versus orpheline	9	8	7
Référence 3: familles équidistantes divergentes	5	3	5
Référence 4: extension N/C-terminal	12		
Référence 5: insertions	12		

Table 2.2. Le contenu de la base BALiBASE.

2.5.5 L'évaluation des alignements par BALIBASE

Pour évaluer la qualité d'alignement, Thompson et al ont crée deux mesures différentes. La somme des pairs SPS détermine le nombre de résidus correctement alignés. Elle est utilisé pour déterminer si les programmes sont capables d'aligner quelques ou toutes les séquences correctement en comparant avec la référence r. Ayant un alignement de taille *l* contenant *N* séquences est un alignement de référence de taille l_r de N séquences. SPS peut être calculé colonne par colonne par la formule suivante :

$$SPS = \frac{\sum_{i=1}^{l} S_i}{\sum_{i=1}^{l_r} S_{ri}} \qquad (2.9)$$

Où S_i et S_{ri} sont les scores de i^{éme} colonne dans l'alignement test et l'alignement référence respectivement. Le score S_i est défini de la manière suivante :

$$S_i = \sum_{j=1}^{N} \sum_{K=1}^{N} P_{ijk} \qquad (2.10)$$

Où P_{ijk}=1 si le résidu est aligné avec chaque autre dans l'alignement référence et 0 autrement.

Le deuxième score utilisé est le score de colonnes. Il donne le pourcentage de colonnes correctement alignées. Le CS test l'habilité de programmes pour aligner toutes les séquences. Pour la i^{éme} colonne le score C_i=1 si tous les résidus dans la colonne sont alignés comme dans la référence et 0 autrement. Le CS global est donné par la formule suivante :

$$CS = \frac{\sum_{i=1}^{l} C_i}{l} \qquad (2.11)$$

En plus de ses scores, Thompson et al ont effectué des tests statistiques. Pour chaque alignement de référence un score est calculé estimant l'exactitude de l'alignement produit par

chaque programme. Ils utilisent la médiane pour mesurer la localisation et l'interquartile directionnel pour la mesure de la dispersion. Le premier et le troisième quartile donnent une idée de la forme de la distribution. Deux statistiques tests ont été utilisées. Le test de Friedman indique si quelques programmes réalisent mieux que d'autres programmes des alignements de références. Les tests de Wilcoxon ont été utilisés pour déterminer si un changement dans les conditions d'un alignement, tel que l'addition des séquences orphelines (la référence 2) ou une augmentation dans le nombre des membres de familles (la référence 3), conduit à une différence significative entre les scores pairs. Les résultats détaillés de cette étude sont disponibles sur le site de BALiBASE.

2.5.6 La comparaison des méthodes non-iteratives et des méthodes itératives

Les quatre programmes les plus réussis dans les conditions distinctes d'alignement examinées, PRRP, SAGA, CLUSTALX et DIALIGN (table 2.3). Il convient de noter que trois de ces programmes emploient des stratégies itératives pour raffiner l'alignement. Le seul programme progressif qui a réussi d'aligner le plus grand nombre d'alignements est CLUSTALX. Il est clairement meilleur que les programmes progressifs traditionnels d'alignement, bien que pour de longues séquences les paramètres par défaut puissent ne pas être optimaux. Les nouveaux algorithmes itératifs offrent souvent une exactitude améliorée d'alignement, ils améliorent efficacement un alignement si assez d'information est incluse dans l'ensemble de données de séquences, comme noté dans les cas d'espèce des familles équidistantes des séquences. Cependant, les méthodes itératives peuvent parfois être instables en présence d'un biais dans l'ensemble de séquences, comme la présence d'une séquence orpheline simple, l'itération peut diverger loin de l'alignement correct. Des programmes locaux, DIALIGN, qui emploie itérativement un algorithme local d'alignement de segment, est le plus réussi. En revanche, l'itération mise en application dans HMMT n'exécute pas aussi efficacement les tests qui incluent jusqu'à 25 séquences. Même dans les tests contenant 100 séquences, HMMT ne se range pas au-dessus des programmes globaux. L'application d'une stratégie itérative améliore clairement l'exactitude de l'alignement dans certaines conditions. Néanmoins, il est évident que le choix de l'algorithme fondamental mis en application à chaque itération est largement important. Un grand inconvénient des techniques itératives courantes est le temps d'exécution énorme. Comme exemple, pour 89 séquence d'histone se composant de 66-92 résidus, le temps-CPU exigé pour l'alignement est 161 s pour CLUSTALX, 13 649 s pour DIALIGN et 13 209 s pour PRRP [Thompson et al, 99b].

2.5.7 La comparaison des méthodes globales et des méthodes locales

Dans cette étude, Thompson et al ont testé l'effet de l'utilisation des algorithmes de programmation dynamique global ou local dans l'alignement multiple. Les programmes globaux d'alignement essayent d'aligner les séquences sur leurs longueurs entières, tandis que les programmes locaux recherchent seulement les motifs les plus conservés. L'algorithme d'alignement le plus efficace dépend de la nature des séquences à aligner. Les algorithmes globaux produisent les alignements les plus précis et les plus fiables dans les tests comportant des séquences équidistants, des familles de séquences divergentes et l'alignement des séquences

orphelines d'une famille. Cependant, les programmes locaux tel que DIALIGN réussissent bien dans les tests contenant de larges extensions N/C-terminal et d'insertions internes. DIALIGN , qui met en application un alignement local et un gap-libre de segment, est le programme le plus réussi à localiser les blocs de flanquement fortement conservés de noyau. Cependant, tout l'alignement en dehors des motifs les plus conservés demeure incertain. Les programmes globaux qui tendent à favoriser un alignement situé sur la même droite des longueurs entières des séquences sont moins réussis, souvent produisant une déviation d'alignement totale des séquences [Thompson et al, 99b].

DATA	PRRP	CLUSTAL	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT
v1(ref1)	0,692	0,647	0,592	0,487	0,536	0,504	0,559	0,571	0,439	0,132
v2(ref1)	0,935	0,932	0,920	0,893	0,910	0,909	0,927	0,911	0,894	0,455
v3(ref1)	0,968	0,970	0,962	0,922	0,961	0,955	0,960	0,962	0,888	0,812
AVR Ref1	0,877	0,864	0,841	0,788	0,821	0,810	0,834	0,832	0,763	0,487
Ref2	0,541	0,583	0,586	0,384	0,379	0,371	0,517	0,429	-	0,401
Ref3	0,532	0,446	0,506	0,314	0,267	0,372	0,303	0,323	-	0,175
Ref4	0,323	0,361	0,289	0,853	0,794	0,705	0,292	0,710	-	-
Ref5	0,700	0,705	0,642	0,836	0,508	0,572	0,627	0,639	-	-

Table 2.3. BALiBASE score des méthodes utilisées dans l'étude [Thompson et al, 99b].

Finalement, on a créé une table (table 2.4) qui résume les méthodes récentes et moins récentes utilisées dans le MSA. Dans cette table, on a mentionné pour chaque méthode sa classe, la fonction utilisée comme fonction objectif et les avantages et les inconvénients.

Méthode	Classe	Fonction	Avantage	Problème
MSA [Lipman et Pearson, 88]	Exacte.	Maximiser la somme des paires SP.	La méthode la plus optimale.	Non praticable pour un grand nombre de séquences.
DCA [Stoye et al, 97]	Exacte.	Maximiser la somme des paires SP.	Donne des solutions optimales ou proches de l'optimal.	• Limitée, requiert Plus de mémoire. • Nécessite MSA
CLUSTALW (94) [Thompson et al.,94]	Progressive/ Globale.	Utilise la somme pondérée des paires WSP.	Standard des méthodes MSA, rapide, Précise et Facile à utiliser.	• Donne une solution locale. • Moins efficace devant les séquences hautement divergentes.
T-COFFEE [Notredame et al., 00]	Progressive/ Globale.	T-COFFEE.	• Corrige les erreurs des méthodes progressives. • Plus d'information	La méthode progressive la plus lente.

			sur les séquences. • Plus précise que CLUSTAL.	
PROBCONS [Brundno et al, 04]	HMM/ Progressive/ Globale.	la transformation probabiliste consistante.	La méthode la plus précise.	Très lente par rapport aux méthodes progressives
MAFFT [Katoh et al, 02]	Progressive/ Globale	Transformation de Fourier FFT	La méthode la plus rapide.	Moins précise que T-COFFEE.
PRRP [Gotoh, 96]	Itérative/Déterministe/Globale.	WSP	Plus optimal que les méthodes progressives.	Lente.
SAGA [Notredame et al,96]	Iterative/ Stochastique/ Global. (AG)	WSP, T_COFFEE…	Le meilleur algorithme basé AG.	• Très lente. • Alignements trouvés ne sont pas forcément optimaux.
HMMT [Eddy,95]	HMM/ Itérative/Globale.	Fonction HMM	• Donne Plus d'un Alignement.	• Des mauvais résultats dans les tests BALIBASE. • Nécessite un grand ensemble de séquences. • Nécessite des conditions initiales justes.
DIALIGN [Morgenstern ,04]	Progressive/ Itérative/ Locale	Fonction basée consistance	Efficace dans le cas des séquences avec large insertion ou NC-terminal	• Lente par rapport à CLUSTAL. • Moins précise dans les alignements globaux.

Table 2.4. Une table récapitulative des différentes méthodes d'alignement multiple de séquences.

2.6 Conclusion

Nous avons vu dans ce chapitre les notions de base de l'alignement multiple de séquences. On a présenté les différentes fonctions de score utilisées pour l'évaluation mathématique des alignements multiples. Finalement, une comparaison de différentes méthodes d'alignement est citée dans ce chapitre. La synthèse de ce chapitre montre la difficulté du problème de l'alignement multiple de séquences. En effet, aucune méthode n'a pas réussi largement dans ce domaine.

Comme on a dit dans l'introduction générale nous allons utiliser une nouvelle approche dite quantique évolutionnaire pour traiter le problème de MSA. Il s'agit des algorithmes évolutionnaires enrichis par des concepts quantiques. Donc une présentation de l'informatique quantique paraît être nécessaire. Dans le prochain chapitre nous allons présenter l'informatique quantique, les ordinateurs quantiques, les algorithmes purement quantiques et les algorithmes inspirés de la quantique.

Chapitre 3

Les principes de base de l'informatique quantique

"No exponential is forever. Your job is to delay forever."
— Andrew Gordon Moore 2003.

3.1 Introduction

Les problèmes informatiques sont devenus de plus en plus fastidieux et complexes que se soit en temps ou en espace. La plupart des problèmes NP-difficiles sont caractérisés par des grandes complexités spatiotemporelles et nécessitent des capacités de traitement et de stockage énorme. Pour ces raisons, les chercheurs essayent de trouver d'autres architectures qui offrent en même temps des ressources de stockage et de traitement puissantes. En effet, plusieurs nouveaux paradigmes informatiques ont immergé ces dernières années. Ces nouveaux paradigmes se basent sur des différents concepts comme la biologie ou la physique. Parmi les nouveaux paradigmes informatiques qui ont suscité beaucoup d'intérêt on distingue l'informatique biomoléculaire [Adleman, 94]. Ce paradigme utilise les caractéristiques de l'ADN pour résoudre les problèmes NP-difficiles. Un autre paradigme très intéressant est l'informatique quantique. L'informatique quantique utilise les spécificités de la mécanique quantique pour le traitement et la transformation de l'information [Le Bellac ,03].

L'informatique quantique est un domaine en émergence faisant appel à plusieurs spécialités : physique, génie, chimie, informatique et mathématiques. L'objectif visé par cette intégration de connaissances est la réalisation d'un ordinateur quantique. L'utilisation d'un tel ordinateur est pour effectuer certains calculs beaucoup plus rapidement qu'avec un ordinateur fonctionnant de façon standard (ordinateur classique). Cette accélération est rendue possible en tirant profit des phénomènes quantiques tels que les superpositions d'états, l'enchevêtrement et l'interférence. Le bit quantique (qubit) est l'unité d'information de base en informatique quantique. Un qubit peut être dans l'état 0, 1, ou une superposition de 1 et 0 ce qui permet de représenter un nombre exponentiel d'états avec un ensemble restreint de qubits (pour n qubits en peut représenter 2^n états). Une autre caractéristique de la mécanique quantique est la linéarité des opérations. Cette caractéristique permet en informatique quantique un parallélisme exponentiel. En effet, une opération est appliquée en parallèle sur tous les états présents en superpositions [Shor, 98].

Les recherches dans ce domaine sont accélérées aussi bien sur le plan pratique que théorique. Théoriquement, plusieurs algorithmes quantiques ont vu le jour en essayant de démonter la puissance de l'informatique quantique. Parmi les algorithmes qui ont mis l'informatique en scène, l'algorithme de factorisation des nombres de Peter Shor [Shor, 94]. Cet algorithme a une complexité polynomiale or le meilleur algorithme classique a une complexité exponentielle. Un autre algorithme très important est l'algorithme de Grover [Grover, 96] pour la recherche d'une donnée dans une liste désordonnée de données. Cet algorithme trouve l'élément recherché dans un temps quadratique. Du coté pratique, IBM a pu

construire une machine quantique à 7 qubits. Cependant, la construction d'une machine contenant des centaines voir des milliers de qubits n'est pas une chose facile à cause des erreurs qui peuvent surgir dues à la décohérence du système quantique. Cette réalisation nécessite des outils théoriques et technologiques plus sophistiqués. En espérant voir dans les années à venir la conception à grande échelle d'un ordinateur quantique.

Mais la recherche en informatique quantique ne s'est pas arrêtée à ce niveau à cause de la non disponibilité des machines quantiques. Plusieurs chercheurs s'intéressent à la combinaison entre des algorithmes classiques et les principes de la mécanique quantique. L'avantage des ces algorithmes est qu'ils ne nécessitent pas la présence d'une machine quantique. Plusieurs algorithmes ont été créés en se basant sur cette idée comme les Algorithmes Quantiques Evolutionnaires AQEs [Han et al ,01].

Ce chapitre est divisé en deux volets. Dans le premier volet on va présenter les fondements de l'informatique quantique. Des circuits quantiques et un algorithme quantique démonstratif sont présentés. Dans le deuxième volet, on va présenter les algorithmes inspirés du quantique. Des définitions et des exemples sont également cités.

3.2 Notions de base

La théorie quantique a été introduite suite à la non capacité de la mécanique classique d'interpréter certains phénomènes physiques. Cette théorie est basée sur un formalisme mathématique solide. La mécanique quantique est fondée sur les quatre postulats suivants[Preskil, 98] :

- *Etat (states)* : Un système physique dans la mécanique quantique est complètement décrit par un vecteur d'état dans l'espace de Hilbert.
- *Observables (observables)*: C'est la propriété qu'un système physique peut être mesurer Cependant dans le cas de la mécanique, les propriétés physiques sont représentées mathématiquement par des opérateurs hermétiques,
- *Mesure (measurement):* en mécanique quantique, la mesure projette le vecteur d'état du système en un nouveau vecteur d'état.
- *La dynamique (dynamics):* L'évolution du temps d'un système quantique isolé est décrite par une transformation unitaire.

Afin de mieux comprendre les phénomènes quantiques, nous allons présenter l'expérience de polarisation des photons.

3.2.1 L'expérience de la polarisation du photon

Cette expérience démontre certains principes de la mécanique quantique à travers les photons et leur polarisation [Rieffel et al, 00]. Les photons sont les seules particules que nous pouvons directement observer. L'expérience est simple et peut être établie avec un équipement minimal. On a besoin d'une source puissante de la lumière comme un pointeur laser et de trois polaroïds (filtres de polarisation). Un faisceau de lumière rayonne sur un écran de projection. Les filtres A, B et C sont polarisés respectivement horizontalement, à 45^{o} et

verticalement. Ils peuvent être placés en intersection avec la lumière. L'expérience est composée des étapes suivantes

Etape 1 : On insère le filtre *A* et en supposant que la lumière entrante est aléatoirement polarisée. Donc, l'intensité de la lumière sortante va être la moitié de l'intensité de l'entrante. Les photons sortants sont maintenant polarisés horizontalement (figure 3.1).

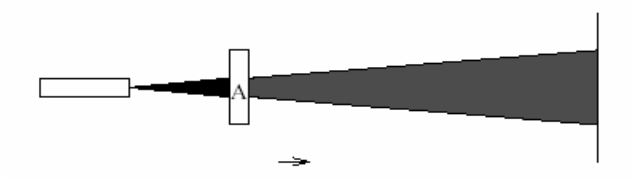


Figure 3.1. L'insertion du filtre A.

La fonction du filtre A ne peut pas être expliquée telle que A est une passoire qui seulement laisse passer les photons déjà polarisés horizontalement.

Etape 2 : Quand le filtre C est inséré (figure3.2), l'intensité de la lumière sortante est zéro. Le filtre C ne laisse passer aucun des photons horizontalement polarisés.

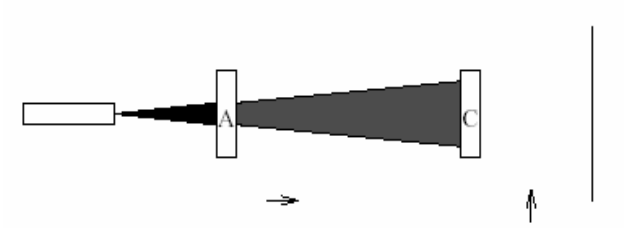


Figure 3.2. L'ajout du filtre C.

Etape 3 : Finalement, Après l'insertion du filtre B entre A et C (figure 3.3), une petite quantité de la lumière sera visible sur l'écran, exactement un huitième (1/8) de la lumière originale.

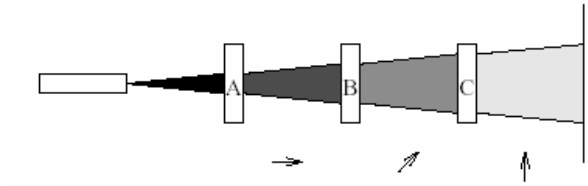


Figure 3.3. L'ajout du filtre B.

Cette expérience a montré un paradoxe. L'expérience classique suggère que l'ajout d'un filtre doit diminuer seulement la quantité de la lumière or dans cette expérience, la quantité de la lumière a augmenté. On peut modéliser l'état de polarisation du photon par un vecteur pointé dans la direction appropriée. Toute polarisation arbitraire peut être exprimée comme un combinaison linéaire : $a|\uparrow\rangle + b|\rightarrow\rangle$ des deux vecteurs de base :

$|\uparrow\rangle$: Polarisation verticale.

$|\rightarrow\rangle$: Polarisation horizontale.

Le vecteur d'état sera un vecteur unitaire : $|a|^2+|b|^2 = 1$, parce que nous sommes intéressés seulement par la direction de la polarisation, Donc généralement, la polarisation d'un photon peut être exprimée comme : $a|\uparrow\rangle+b|\rightarrow\rangle$. Où a et b sont deux nombres complexes tels que $|a|^2+|b|^2 = 1$.

Le principe de mesure de la mécanique quantique révèle que tout dispositif de mesure d'un système à deux dimensions a un ensemble associé de vecteurs de base. La mesure d'un état est la projection de cet état à l'un des vecteurs du dispositif de mesure associé. La probabilité qu'un état est mesuré selon un vecteur de base $|u\rangle$ est égale au carré de l'amplitude du composant de la projection de l'état original sur le vecteur de base $|u\rangle$ (figure 3.4). Sachant qu'il peut exister plusieurs dispositifs de mesure ce qui entraîne des résultats de mesure différents. Si on a par exemple la mesure de l'état quantique suivant :

$\psi = a|\uparrow\rangle + b|\rightarrow\rangle$ donne $|\uparrow\rangle$, alors l'état ψ prend définitivement la valeur $|\uparrow\rangle$. C'est-à-dire une nouvelle mesure de cet état va donner certainement $|\uparrow\rangle$ avec la probabilité 1. En mécanique quantique on ne peut pas prédire l'état original d'un état après l'opération de mesure.

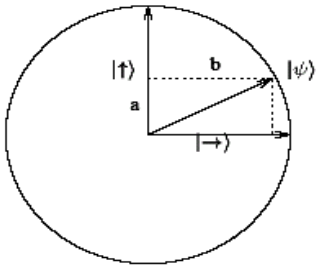


Figure 3.4 Mesure : Une projection sur la base.

Donc, d'après la mécanique quantique, les filtres jouent le rôle des dispositifs de mesure. Le filtre ne laisse passer que les photons qui ne sont pas orthogonaux à sa polarisation. Par exemple, le filtre A mesure la polarisation des photons relativement au vecteur de base $|\rightarrow\rangle$, correspondant à sa polarisation. Cependant, tous ceux qui sont reflétés par le filtre C ont une polarisation $|\uparrow\rangle$. Quant à l'écran, il mesure l'état quantique des photons relativement à sa base. Finalement, le filtre B va mesurer les photons relativement à la base : $\{1/\sqrt{2}\ (|\uparrow\rangle+|\rightarrow\rangle),\ 1/\sqrt{2}\ (|\uparrow\rangle-|\rightarrow\rangle)\}$. C'est ce qu'on écrit comme : $\{|k\rangle, |j\rangle\}$ tel que

$$|\rightarrow\rangle = 1/\sqrt{2}\ (|k\rangle-|j\rangle).$$

$$|\uparrow\rangle = 1/\sqrt{2}\ (|k\rangle+|j\rangle).$$

Donc les photons qui sont mesurés selon $|k\rangle$ passent à travers le filtre. Les photons qui passent par A avec l'état $|\rightarrow\rangle$ vont être mesurés par B comme $|k\rangle$ avec une probabilité de ½. Par conséquence 50 % des photons passés par A vont passer par B et dans l'état $|k\rangle$. Idem, les

photons seront mesurés par le filtre C comme $|\uparrow\rangle$ avec une probabilité de 1/2. Donc seulement 1/8 (un huitième) des photons originaux vont passer à travers la séquence des filtres : A, B et C (figure 3.5).

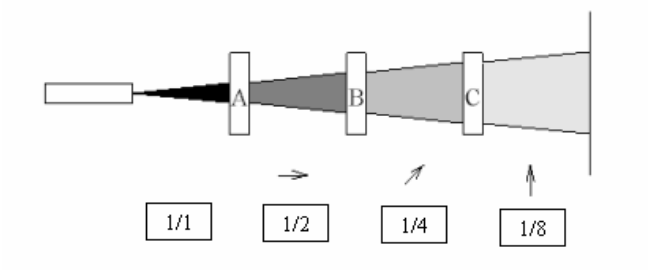


Figure 3.5. Le pourcentage de la lumière après chaque ajout des trois filtres.

3.2.2 Espaces d'états et notation Bra/Ket

Un espace d'états d'un système quantique est constitué des positions, moments, polarisation, spins…etc, des différentes particules. Il est modélisé par un espace de Hilbert des fonctions des ondes. Dans l'informatique quantique, nous avons besoin seulement des systèmes quantiques finis et il est suffisant de considérer des espaces vectoriels complexes de dimensions finies avec un produit interne qui sont franchis par des fonctions abstraites d'onde comme $|\rightarrow>$. Les espaces d'états quantiques et les transformations actant sur ces états peuvent être décrits en terme des vecteurs et des matrices. Ou en utilisant la notation Bra/Ket inventée par Dirac [Dirac, 59]. Kets $|x\rangle$ dénote un vecteur colonne et elle est spécifiquement utilisée pour décrire les états quantiques. Cependant, le "Bra" $\langle x|$: Dénote la transformée conjuguée de $|x\rangle$. Par exemple, Les orthogonaux $\{|0\rangle,|1\rangle\}$ peuvent être exprimés comme $\{(1,0)^T, (0,1)^T\}$. Donc, toute combinaison linéaire complexe de $|0\rangle$ et $|1\rangle$, $a|0\rangle + b|1\rangle$ peut être écrite $(a,b)^T$. le choix de l'ordre des vecteurs de base est arbitraire. Par exemple, on peut représenter $|0\rangle$: $(0,1)^T$ et $|1\rangle$: $(1,0)^T$.

La combinaison de $\langle x|$ et $|y\rangle$ comme dans $\langle x\|y\rangle$ ($\langle x|y\rangle$), dénote le produit interne des deux vecteurs. Vu que $|0\rangle$ est un vecteur unitaire nous avons : $\langle 0|0\rangle = 1$, et comme $|0\rangle$ et $|1\rangle$ sont orthogonaux nous avons : $\langle 0|1\rangle = 0$. La notation $|x\rangle\langle y|$ est le produit externe de $|x\rangle$ et $\langle y|$. Par exemple, $|0\rangle\langle 1|$ est la transformation qui change $|1\rangle$ à $|0\rangle$ et $|0\rangle$ à $(0,0)^T$.

$$|0\rangle\langle 1|\ |1\rangle = |0\rangle\langle 1|1\rangle = |0\rangle$$

$$|0\rangle\langle 1|\ |0\rangle = |0\rangle\langle 1|0\rangle = 0\ |0\rangle = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

De la même façon, $|0\rangle\langle 1|$ peut être écrite dans une forme matricielle où :

$|0\rangle = (1,0)^T, \langle 0| = (1,0)$, $|1\rangle = (0,1)^T$, $\langle 1| = (0,1)$.

Ce qui donne donc, le produit suivant: $|0\rangle\langle 1| = (1,0)^T\ (0,1) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$.

Cette notation nous donne une façon adéquate pour indiquer les transformations dans les états quantiques, en terme de ce qui parvient aux vecteurs de base. Par exemple : la transformation qui échange $|0\rangle$ et $|1\rangle$ est donnée par : $X = |0\rangle\langle 1| + |1\rangle\langle 0|$. Mais dans la plupart des auteurs en informatiques quantiques préfèrent la notation la plus intuitive :

$$X: \quad |0\rangle \rightarrow |1\rangle.$$
$$|1\rangle \rightarrow |0\rangle.$$

Ceci spécifie explicitement le résultat de la transformation dans les vecteurs de base [Rieffel et al, 00].

3.2.3 Le qubit

Une idée géniale est déduite de cette expérience. On peut utiliser la polarisation des photons pour transmettre de l'information. En comparant avec le paradigme classique , On décide, tout à fait arbitrairement, d'attribuer la valeur 1 du bit à un photon polarisé Suivant Ox($|\rightarrow\rangle$) et la valeur 0 un photon polarise suivant Oy ($|\uparrow\rangle$). D'ou vient l'idée de l'informatique quantique. L'information quantique est la théorie de l'utilisation des spécificités de la physique quantique pour le traitement et la transmission de l'information. L'unité fondamentale de l'information quantique est le bit quantique ou, plus simplement qubit (de l'anglais 'quantum bit'). Ce terme a été introduit par B. Shumacher. Comme sa contrepartie classique, un qubit peut prendre deux valeur dénotées |0> et |1>. Un qubit vit donc dans un espace d'Hilbert à deux dimensions et peut être dans une superposition de ces deux états orthogonaux a|0>+ b|1>. La base {|0>, |1>} correspond aux deux polarisations du photon$|\rightarrow\rangle, |\uparrow\rangle$. Il est intéressant de mentionner que, selon le théorème de Holevo, seulement un bit classique peut être extrait d'un bit quantique. Lorsque l'on s'intéresse à un seul qubit, le vecteur de Bloch b(t) sur la sphère de Bloch est une représentation utile (figure 3.6). Sous leur apparente ressemblance les bits classiques et quantiques offrent des possibilités totalement différentes dues aux étranges propriétés des qubits que nous décrirons dans la table 3.1 [Blais, 02].

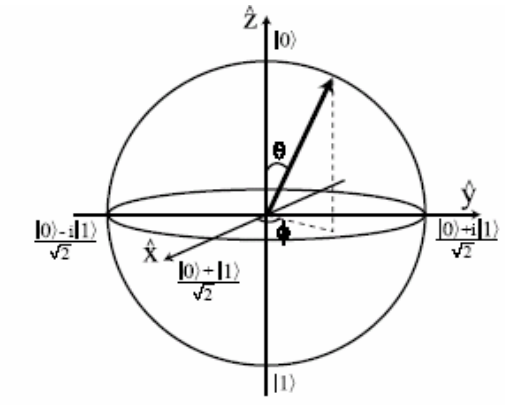


Figure 3.6. La sphère de Bloch.

classique	quantique
Un bit classique a toujours une valeur définie.	On ne peut pas savoir la valeur de l'état d'un qubit non mesuré.
Un bit peut avoir exclusivement 1 ou 0.	Un qubit peut être dans une superposition de 1 et 0 simultanément.
Le bit peut être copié sans être détérioré.	Impossible de copier un qubit dans un état inconnu.
Un bit peut être lu sans affecter sa valeur.	Lire un qubit dans une superposition changera sa valeur.

Table 3.1. Comparaison entre le bit classique et le qubit.

3.2.4 Représentation d'un état quantique

Il existe plusieurs notations pour représenter un état quantique qu'on peut résumer comme suit [Rieffel et al, 00] :

Ø *Notation de Dirac*

Les physiciens en théorie quantique utilisent souvent une notation pratique inventée par Dirac et qui diffère quelque peu de la notation mathématique usuelle. Un qubit Q est représenté par une combinaison de $|0\rangle$ et $|1\rangle$ comme suit :

$$|Q\rangle = a\,|0\rangle + b\,|1\rangle \qquad (3.1)$$

Tel que :

$$|a|^2 + |b|^2 = 1 \qquad (3.2)$$

Le qubit peut avoir l'état 0 (1) avec une probabilité $a^2 (b^2)$ respectivement.

Ø *Notation vectorielle*

Un qubit peut être représenté par une forme matricielle par :

$$0 \rightarrow \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad 1 \rightarrow \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Ø Notation matricielle

On utilise la matrice de densité pour représenter un qubit :

$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$: Pour représenter l'état |0>.

$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$: Pour représenter l'état |1>.

3.2.5 Registre quantique

Une notion très importante en informatique quantique est le registre quantique. Il est constitué d'un ensemble de qubits, c'est une superposition arbitraire de n qubits. Contrairement à un registre classique de n bits qui représente à instant précis une seule valeur parmi les 2^n valeurs possibles, un registre quantique peut contenir les 2^n valeurs possibles en même temps grâce au principe de la superposition d'états [Simon, 94]. La table 3.2 montre l'équivalence en bits classiques pour avoir la même puissance d'un ensemble de qubits. Un registre est représenté par la notation suivante :

$$\Psi = \sum_{x=0}^{2^k-1} C_x \left| X \right\rangle \tag{3.3}$$

Les amplitudes C_x satisfont la propriété :

$$\sum_{x=0}^{2^k-1} \left| C_X \right|^2 = 1 \tag{3.4}$$

NB qubits	NB de bits classiques nécessaires pour obtenir la même puissance
10 20 30 300	1000 1 000 000 1 000 000 000 000 Plus que le nombre d'atomes dans l'univers!

Table 3.2. L'équivalence en bits classique pour avoir la même puissance d'un registre quantique.

3.2.6 Les fondements de l'informatique quantique

L'informatique quantique est basée principalement sur les cinq principes suivants inspirés de la mécanique quantique : la superposition d'états, l'Interférence, L'enchevêtrement, le non déterminisme et la non duplication [Preskill, 98].

A. La superposition

Un qubit peut être dans l'état 0 comme il peut être dans l'autre état 1. Mais, il peut être également dans les deux états en même temps. Ce phénomène est appelé la superposition d'état. Ce principe offre donc des capacités de traitement et de stockage exponentielles. En effet, un registre quantique de taille n peut contenir à la fois 2^n valeurs différentes.

B. L'interférence

L'interférence est l'une des caractéristiques des systèmes ondulatoires. D'après les postulats de la mécanique quantique, Les particules quantiques ont une double nature ; corpusculaire et ondulatoire. Donc le phénomène d'interférence est applicable dans la mécanique quantique. Il a comme rôle d'augmenter (interférence constructive) ou diminuer (interférence destructive) l'amplitude d'un état et par conséquence sa probabilité d'être observé. L'interférence est très importante en calcul quantique. Elle permet d'augmenter la probabilité d'avoir les résultats escomptés. Prenant l'exemple suivant :

Soit un qubit décrit par la formule vectorielle suivante : $|y\rangle = 1/\sqrt{5}\begin{pmatrix}2\\1\end{pmatrix}$.

Et soit l'opération *O* qui fait l'interférence décrit par la matrice suivante : $O = 1/\sqrt{2}\begin{pmatrix}1 & 1\\1 & -1\end{pmatrix}$.

L'application de *O* sur le qubit donne le résultat suivant :

$$O|y\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\1 & -1\end{pmatrix}\frac{1}{\sqrt{5}}\begin{pmatrix}2\\1\end{pmatrix} = \frac{1}{\sqrt{10}}\begin{pmatrix}3\\1\end{pmatrix}.$$

La notation de Dirac pour ce résultat est :

$$|y\rangle = \frac{3}{\sqrt{10}}|0\rangle + \frac{1}{\sqrt{10}}|1\rangle.$$

On voit clairement que l'amplitude de l'état $|0\rangle$ a été augmentée tandis que l'amplitude de l'état $|1\rangle$ est diminuée.

C. L'enchevêtrement

Ce phénomène n'a pas d'analogie classique, c'est la corrélation entre deux qubits : par exemple l'état : $|y\rangle = 1/\sqrt{2}\left(|00\rangle + |11\rangle\right)$ est en enchevêtrement, si le premier qubit est mesuré en 1(0) alors le deuxième qubit est sans aucun doute vaut 1(0) respectivement. Mathématiquement, l'état global d'un système quantique est dit enchevêtré s'il est impossible de l'écrire sous forme d'un produit tensoriel de deux de ses sous systèmes :

$$|f_{AB}\rangle \neq |f_A\rangle \otimes |f_B\rangle \qquad (3.5)$$

Ce principe permet de réaliser des calculs irréalisables sur des ordinateurs classiques. L'algorithme quantique de Shor [Shor, 94] pour la factorisation des grands nombres est un bon exemple. Grâce à ce principe, Shor a donné un algorithme pour la factorisation des grands nombres dont la complexité est polynomiale.

D. Le non déterminisme

Contrairement à la physique classique, la mécanique quantique est indéterministe c'est-à-dire que les mêmes causes ne donnent pas forcement les mêmes résultats. La valeur d'un qubit en superposition n'est pas connue avant l'opération de mesure mais on peut augmenter la probabilité d'avoir un état en augmentant son amplitude.

E. Le non clonage

Le clonage d'un état indéterminé est impossible en mécanique quantique. Reprenons l'exemple du photon ; on ne réussi jamais à dupliquer un photon dont on ne sait rien car il est difficile de connaitre complètement son état quantique. Si on essaie dés qu'on commence à mesurer certaines quantités observables par exemple la polarisation selon une direction, on détruit l'état quantique du photon et on ne peut pas mesurer d'autres observables. Ceci est la conséquence directe de la linéarité des transformations unitaires. Supposons que U est une transformation unitaire qui fait le clonage. Ayant deux états quantiques orthogonaux |a> et |b>. L'application de U donne :

U(|a0>)=|aa> et U(|b0>)=|bb>.

Considérons l'état $|c\rangle = 1/\sqrt{2}\,(|a\rangle + |b\rangle)$, L'application de U sur c donne deux valeurs différentes :

- Par linéarité : U(| c0 >) = (U(| a0 >) + U(| b0 >)) = (| aa > + | bb >).
- Par clonage : U (| c0 >) = | cc > = 1/2(| aa > + | ab > + | ba > + | bb >)

Ce qui est contradictoire.

3.2.7 La mesure quantique

La mesure quantique est une opération très importante en informatique quantique. C'est la possibilité de mesurer les états classiques de la mémoire. Avant l'opération de mesure, un bit quantique n'a pas une valeur fixe contrairement au bit classique qui a toujours une valeur définie. Un bit classique peut prendre une valeur de 0 ou 1, cependant un qubit peut être dans une superposition des deux au même temps. Théoriquement, La mesure quantique est une forme spécifique d'interaction entre l'ordinateur quantique et un dispositif physique externe de mesure. D'une manière simple, la mesure est la projection de l'état d'un qubit sur l'une des bases de l'espace de Hilbert (figure 3.7). La mesure d'un qubit en superposition donne une seule valeur au qubit et elle dépend des amplitudes de ce qubit. Comme la mécanique quantique est indéterministe le résultat de la mesure est probabiliste. On ne peut pas avoir forcement la même valeurs après deux mesures consécutives [Rieffel et al, 00].

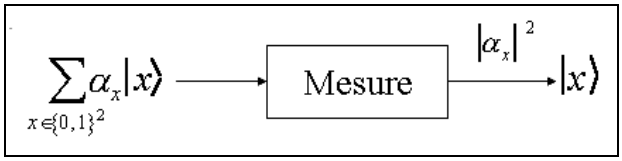


Figure 3.7. Mesure quantique.

3.2.8 Opérations logiques quantiques

Le deuxième postulat de la mécanique quantique dit : *L'évolution du temps d'un système quantique isolé, est décrite par une transformation unitaire*. Donc, une opération dans le paradigme quantique doit satisfaire les conditions suivantes :

- Unicité : $f' * f = I$ (où *f'* est l'inverse de *f*, *I* : matrice d'identité)
- Linéarité : Si $|\psi\rangle = \sum_{i=1}^{n} Ci|bi\rangle$ alors $f(|\psi i\rangle) = \sum_{i=1}^{n} Ci|\, f(bi)\rangle$.

La dernière caractéristique est l'un des avantages de l'informatique quantique sur l'informatique classique. Grâce à cette caractéristique, une opération est appliquée sur tous les états en superposition parallèlement [Simon, 94].

3.2.9 Circuits quantiques simples

Afin de pouvoir réaliser un ordinateur quantique, l'informatique quantique doit offrir les circuits de base pour la construction d'une telle machine. Mais, plusieurs portes classiques comme la porte AND sont des opérations irréversibles or les opérations dans le paradigme quantique doivent être réversibles. Heureusement, Deutsch [Deutsch, 85] a démontré la

possibilité de construire des portes quantiques réversibles pour n'importe quelle fonction classique. En fait, il est possible de concevoir une machine de Turing quantique universelle. Plusieurs portes ont été crées qui effectuent des opérations classiques telles que le OR, AND, NOT, ou des opérations propres au paradigme quantique telle que la superposition. Parmi les circuits les plus importants pour la construction des ordinateurs quantiques on distingue les suivants [Rieffel et al, 00], [Blais et al, 00]:

- *Circuits de Pauli* : Les circuits de Pauli constituent un ensemble de portes élémentaires permettant de construire toutes les portes logiques. N'importe quelle autre opération M peut être écrite sous la forme d'une combinaison linéaire de ces trois matrices plus la matrice d'identité :

$M = ljI + \sum_{i}^{3} lipi$. I : matrice d'identité, Pi matrice de Pauli (X, Y, Z).

I : $|0\rangle \rightarrow |0\rangle$, $|1\rangle \rightarrow |1\rangle$ $\quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

X : $|0\rangle \rightarrow |1\rangle$, $|1\rangle \rightarrow |0\rangle$ $\quad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Y : $|0\rangle \rightarrow -|1\rangle$, $|1\rangle \rightarrow |0\rangle$ $\quad \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$

Z : $|0\rangle \rightarrow |0\rangle$, $|1\rangle \rightarrow -|1\rangle$ $\quad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

Les noms de ces transformations sont conventionnels : **I** : est la transformation identité. **X :** est la négation. **Z**: est la transformation de décalage de phase. **Y = Z* X** est une combinaison des deux.

- *La transformation de Hadamard* : C'est une operation tres imporatnte dans les algorithmes quantiques :

$$U_{WH}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \qquad U_{WH}|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Cette porte permet de créer des superpositions cohérentes et de représenter donc 2^n valeurs dans un registre à n qubits comme suit :

$$\begin{aligned} U_{WH}|0\rangle \otimes U_{WH}|0\rangle \otimes \mathbf{K} \otimes U_{WH}|0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \mathbf{K} \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= \frac{1}{\sqrt{2^n}}(|00\mathbf{K}0\rangle + |00\mathbf{K}1\rangle + \mathbf{K} + |11\mathbf{K}1\rangle) \\ &= \frac{1}{\sqrt{2^n}}(|0\rangle + |1\rangle + |2\rangle + \mathbf{K} + |2^n - 1\rangle) \end{aligned}$$

Ø *Controlled-Not* : c'est une opération binaire. Elle est très utile pour effectuer des opérations contrôlées. Elle change le deuxième bit si le premier est à 1, et laisse le bit inchangé sinon (figure 3.8).

C_{Not} :

$$\begin{aligned} |00\rangle &\rightarrow |0\,0\rangle \\ |01\rangle &\rightarrow |0\,1\rangle \\ |10\rangle &\rightarrow |1\,1\rangle \\ |11\rangle &\rightarrow |1\,0\rangle \end{aligned} \quad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Figure 3.8. Représentation matricielle du C_{Not}..

Ø *La porte de Toffoli :* L'opération AND est une opération non réversible. Par ailleurs, cette porte est très importante pour effectuer des calculs. Heureusement, ce problème a été résolu par la création de la porte Toffoli. Cette porte binaire permet la construction de la porte AND. La porte de Toffoli constitue un élément indispensable pour la construction d'un futur ordinateur quantique.

La porte de Toffoli*: T (|x, y, 0>)=|x, y, x AND y>*.

Un circuit quantique complexe qui effectue une tâche bien précise comme l'adition est composé d'un ensemble de circuits simples (figure 3.9).

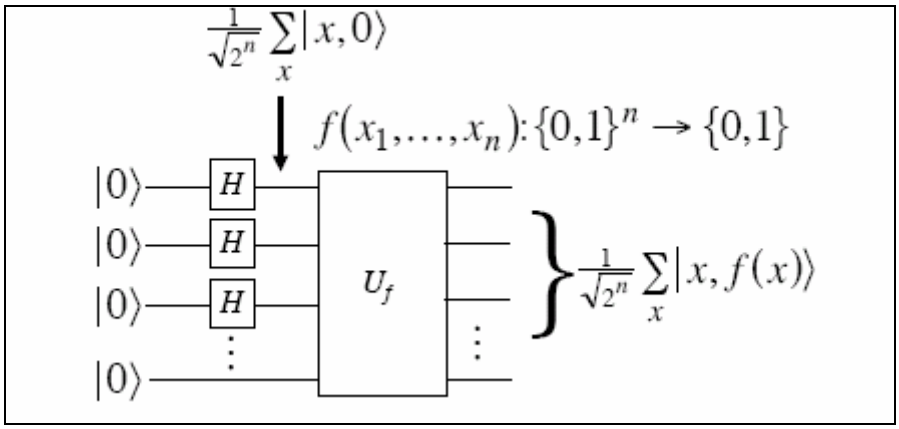


Figure.3.9. Un circuit quantique complexe.

3.2.10 Le paradigme standard

Le paradigme quantique a été énoncé par D. Divencenzo. Il consiste des cinq critères de base qu'un ordinateur quantique doit satisfaire [Blais, 02]:

1. Qubits : Il doit y avoir des qubits.
2. Initialisation : Il doit être possible d'initialiser le registre quantique dans un état connu, souvent choisi comme : $|0\rangle = |000...00\rangle$.
3. Calcul : un registre quantique est manipulable par un ensemble universel de portes logiques.
4. Cohérence : il faut minimiser ou éliminer toute interaction du système quantique avec l'environnement externe pendant les calculs.

5. Mesure : Chaque qubit doit pouvoir subir une mesure projective et la valeur du bit classique qui en résulte doit être obtenue avec une grande efficacité.

3.2.11 Les algorithmes quantiques

Afin d'exploiter efficacement la puissance de l'informatique quantique, il faut créer des algorithmes quantiques efficaces. Un algorithme quantique est constitué d'une suite d'opérations quantiques sur des systèmes quantiques. Ces opérations quantiques peuvent être des portes logiques quantiques simples ou bien des circuits quantiques. En informatique quantique, on a la possibilité de représenter 2^n valeurs possibles dans un seul registre quantique de taille n et de les traiter parallèlement et cela grâce aux concepts de la superposition d'états, la linéarité des opérations et l'enchevêtrement. Le principe de la superposition permet un stockage exponentiel. Par ailleurs, la linéarité et l'enchevêtrement permettent un traitement massivement parallèle. Les algorithmes quantiques sont donc plus efficaces que les algorithmes classiques. Profitant de ces capacités, plusieurs algorithmes ont vu le jour qui essayent de démontrer l'efficacité de l'informatique quantique pour la résolution des problèmes dont la solution classique est coûteuse en temps et en espace. Cependant, l'écriture d'un algorithme quantique est encore une tâche dure. En effet, Les gens ne sont pas encore familiarisés avec des concepts étranges tels que la superposition d'états et l'enchevêtrement. En 1994 Shor [Shor, 94] a surpris le monde en donnant un algorithme polynomial pour la factorisation de grands nombres, un autre algorithme qui a rendu l'informatique quantique plus intéressante est celui de Grover [Grover, 96] pour la recherche dans une base de données non triée dont la complexité est quadratique. Nous présentons dans ce qui suit les grandes lignes de l'algorithme de Grover

Ø L'algorithme de recherche de Grover

Cet algorithme démontre bien que l'informatique quantique peut être plus puissante que les ordinateurs classiques pour la résolution de certains problèmes NP-difficiles. Grover a démontré que son algorithme permet de trouver un élément satisfaisant une condition donnée dans une base non triée dans un temps quadratique. L'algorithme est constitué des étapes suivantes :

1. Création d'une superposition en appliquant un Hadamard sur un registre de taille n initialisé à|0..0>.
2. répéter les étapes suivantes $\sqrt{2^n}$ fois :
3. pour tout état qui satisfait l'oracle appliquer une opération shift (inverser la phase)
4. appliquer sur le registre une opération de rotation par moyenne
5. mesurer le registre : on obtient l'état désiré avec une grande probabilité.

Dans la première étape on utilise l'opération de Hadamard pour créer une superposition cohérente d'états (figure 3.10). Cette dernière contient donc une superposition de toutes les solutions possibles. La deuxième étape de l'algorithme de Grover consiste à inverser l'amplitude des bonnes solutions (figure 3.11).

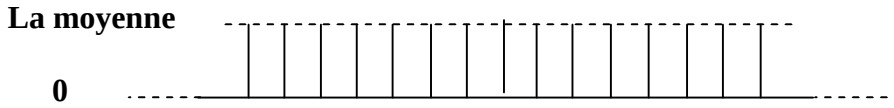


Figure 3.10. Création d'une superposition d'états.

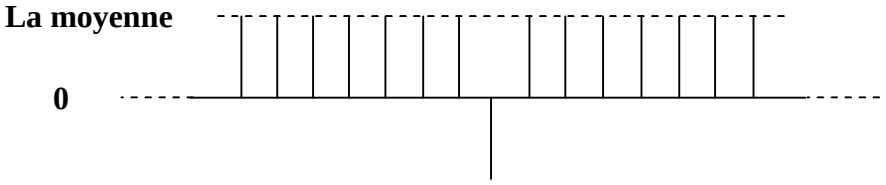


Figure 3.11. Inversement de l'amplitude de la bonne solution.

La troisième opération consiste à augmenter l'amplitude de la bonne solution en utilisant une opération d'inversement par rapport au moyen (figure 3.12).

$$D = \begin{pmatrix} \frac{2}{N} & \frac{2}{N} & \cdots & \frac{2}{N} \\ \frac{2}{N} & \frac{2}{N} & \frac{2}{N} & \frac{2}{N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{2}{N} & \frac{2}{N} & \frac{2}{N} & \frac{2}{N} \end{pmatrix}.$$

Figure 3.12. Une opération d'inversement par rapport au moyen.

Apres un certain nombre d'itérations, l'amplitude de la bonne solution sera grande et les amplitudes des autres solutions seront diminuées (figure 3.13). Ce qui augmente la probabilité de mesurer la bonne solution. Finalement on applique une opération de mesure pour extraire la solution.

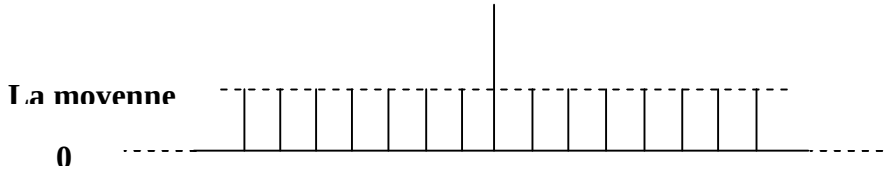


Figure 3.13. L'augmentation de l'amplitude de la bonne solution.

Grover a démonté qu'après avoir faire $\frac{p}{4}\sqrt{2^n}$ d'itérations, le taux d'échec de mesurer la bonne solution est de 2^{-n}. Par ailleurs, l'augmentation des itérations n'augmente pas la probabilité de la mauvaise solution. Dans le cas où il existe b états qui satisfont la condition, la complexité est de l'ordre O ($\sqrt{2^n / b}$).

Ø L'algorithme de Grover revisité

L'algorithme de Grover revisité est plus général que l'algorithme classique de Grover. Ce dernier s'applique seulement dans le cas d'une superposition d'amplitudes uniforme (le nombre d'états dans la superposition égale à 2^n(n le nombre de qubit). Si on l'applique sur

une superposition d'amplitudes arbitraire on aura une mauvaise probabilité d'avoir la bonne solution. La raison est simple : après la première itération on aura deux types d'états indésirables : les états qui ne satisfont pas la condition de la recherche, et les états qui n'existent pas dans la superposition initiale et qui surgissent après l'application de l'opération d'inversement par la moyenne de Grover. Pour remédier à ce problème, Biron introduit une nouvelle rotation qui sera appliquée après la première itération de l'algorithme classique. Cette rotation inverse la phase de l'état désiré et de tous les états qui se trouvent dans la superposition initiale. Puis on applique l'algorithme ordinaire de Grover. Cet algorithme comprend les étapes suivantes [Biron et al, 98] :

1. Initialisation : constitution d'une superposition arbitraire de taille n.
2. Inverser la phase de l'état qui satisfait l'oracle.
3. Appliquer la matrice de rotation par la moyenne de Grover.
4. Appliquer la rotation de Biron sur tous les états initiaux de la superposition.
5. Appliquer la matrice de rotation par la moyenne de Grover.
6. Si le nombre d'itération est inférieur à ($\sqrt{n}$ -1) alors exécuter la boucle suivante, sinon aller à 7.
    - Inverser la phase l'état qui satisfait l'oracle.
    - Appliquer la matrice de rotation par la moyenne de Grover.
7. Fin

L'algorithme de Grover est très utile dans la résolution de plusieurs problèmes NP-difficile comme le problème du voyageur du commerce.

3.2.12 Réalisation physique de l'ordinateur quantique

La construction à grande échelle des ordinateurs quantiques sera des systèmes quantiques complexes contenant plusieurs parties. La réalisation d'un tel système doit faire face à des difficultés expérimentales et théoriques énormes et non encore résolues. L'un des problèmes les plus difficiles dans le développement des systèmes informatiques quantiques est le maintien de la cohérence du système jusqu'à la fin des calculs. Cette perte de cohérence (décohérence) est due à l'interaction des qubits avec le monde extérieur. Plus le nombre de qubits est grand plus la probabilité qu'un qubit interagisse avec l'environnement. Une découverte essentielle pour réduire le taux d'erreur est l'existence des correcteurs d'erreurs. Le principe utilisé est de coder l'état d'un qubit de manière redondante sur une superposition judicieusement choisi de plusieurs qubits. L'altercation d'un qubit est donc détectée et corrigée sans extraire l'information sur l'état du qubit initial. Ce qui préserve la cohérence du système [Brassard, 96]. Un ordinateur quantique peut être construit en utilisant n'importe quelle petite particule qui peut avoir deux états. Des ordinateurs quantiques peuvent être construits à partir d'atomes qui sont à la fois excités et non excités au même moment. Ils peuvent être construits à partir de photons de lumière qui sont à deux endroits en même moment. Ils peuvent être construits à partir de protons et de neutrons ayant un spin soit positif soit négatif ou les deux en même temps. Les différentes techniques pour la réalisation d'un ordinateur quantique peuvent être trouvées dans [Blais, 02].

3.3 Les algorithmes inspirés du quantique

Vu que les algorithmes quantiques purs sont difficiles à concevoir en raison de la non disponibilité actuellement des machines quantiques. Plusieurs chercheurs s'intéressent à la combinaison des algorithmes classiques aux principes de l'informatique quantique. L'avantage des algorithmes inspirés de la mécanique quantique est qu'ils ne nécessitent pas la présence d'ordinateurs quantiques. Le premier algorithme inspiré du quantique est celui de Moore [Moore, 95] pour le tri de nombres. La complexité de son algorithme est $\sqrt{n}\left\lfloor n + 2\left(\sqrt{n} - 1\right)\right\rfloor$. Cependant, la complexité de l'algorithme classique pour le tri de Knuth est ½n(n-1) . En 2000, Han et Kim [Han et al, 00] ont proposé le premier algorithme génétique inspiré du quantique. En 2001, ils proposent un algorithme quantique évolutionnaire [Han et al ,01]. Cet algorithme combine les idées de l'informatique quantique aux algorithmes évolutionnaires classiques. Les algorithmes inspirés de la mécanique quantique ont donné de bons résultats face à des problèmes d'optimisation combinatoire comme le problème du recalage d'image [Draa et al, 04], [Talbi et al, 04], le problème de sac à dos [Han et al ,01], le problème d'allocation de disque [Kim et al, 03], etc.

3.3.1 La méthodologie des méthodes inspirées du quantique

Les méthodes inspirées du quantique doivent avoir les caractéristiques suivantes [Moore, 95] :

1. Le problème doit être exprimé sous forme numérique.
2. La configuration initiale doit être déterministe.
3. Les critères d'arrêt doivent être définis avec concision.
4. Le problème peut être divisé en sous problème.
5. Le nombre de population doit être défini.
6. Assigner un sous problème pour chaque population.
7. Le calcule dans les différentes populations est effectué en parallèle.
8. Il doit y avoir quelque interaction inter populations. L'interférence doit améliorer la solution ou donner une information pour la population pour trouver la solution.

Les caractéristiques ci-dessus donnent une vue utile pour designer les méthodes inspirées de la mécanique quantique, mais elles n'articulasse pas comment déterminer la configuration initiale. Probablement qu'il existe une relation directe entre le type de problème et la configuration utilisée. La division de problème en sous problèmes est l'élément clé des méthodes inspirées du quantique.

3.3.2 Algorithmes quantiques évolutionnaires

Dans les années 50 et les années 60, plusieurs informaticiens ont indépendamment étudié les systèmes évolutionnaires avec l'idée que l'évolution pourrait être utilisée comme un outil d'optimisation pour les problèmes d'ingénierie. L'idée dans tous ces systèmes était d'évoluer une population de solutions candidates pour un problème donné, en utilisant des opérateurs inspirés par la génétique et la sélection naturelle. Depuis lors, trois domaines principaux ont

évolués : stratégies d'évolution, programmation évolutionnaire, et algorithmes génétiques. De nos jours, ils forment le circuit principal de la zone du calcul évolutionnaire.

Les Algorithmes Génétiques (AGs) ont été inventés par John Hollande dans les années 60 [Holland, 75]. Le but initial de Hollande était d'étudier formellement le phénomène de l'adaptation comme il se produit en nature, afin de développer des concepts permettant d'importer le mécanisme de l'adaptation naturelle dans les systèmes informatiques. Dans le travail de Hollande, les AGs sont présentés comme une abstraction de la biologie de l'évolution, et un cadre théorique pour l'adaptation sous l'AG. L'algorithme génétique de Hollande est une méthode permettant un déplacement d'une population de chromosomes à une nouvelle. Ce changement est effectué en utilisant une sorte de sélection naturelle ainsi que des opérateurs inspirés des opérations génétiques tel que le croisement et la mutation. Chaque chromosome se compose de gènes (bits dans la représentation d'ordinateur). Dans le cadre de l'optimisation combinatoire par les AGs, un chromosome code une solution potentielle du problème à résoudre. Le mécanisme de codage de la solution est vital pour la structure des algorithmes génétiques. Il existe principalement deux types de codage utilisés pour coder les solutions dans les chromosomes : le codage binaire et le codage réel. La nature des variables du problème à coder joue un grand rôle dans le choix du codage approprié. [Whitley, 93]. Traditionnellement, le croisement et les mutations sont implémentés comme suit :

- Croisement : Deux chromosomes sont pris pour produire deux chromosomes enfants. Tous les deux chromosomes pères sont coupés à gauche et à droite (sous chromosomes). La position fendue (point de croisement) est la même pour les deux parents. Donc, chaque enfant obtient la gauche sous chromosome d'un parent et la droite sous chromosome de l'autre parent. Par exemple, si les parent chromosomes sont 11001 et 01011 et le point de croisement est entre les bits 3 et 4 (où des bits sont numérotés de gauche à droite), alors les enfants sont 11011 et 01001 (figure 3.14).
- Mutation : Quand un chromosome est altéré par une mutation, quelques gènes sont aléatoirement choisis pour être modifiés. Les bits correspondants sont renversés de 0 à 1 ou de 1 à 0 (figure 3.15).

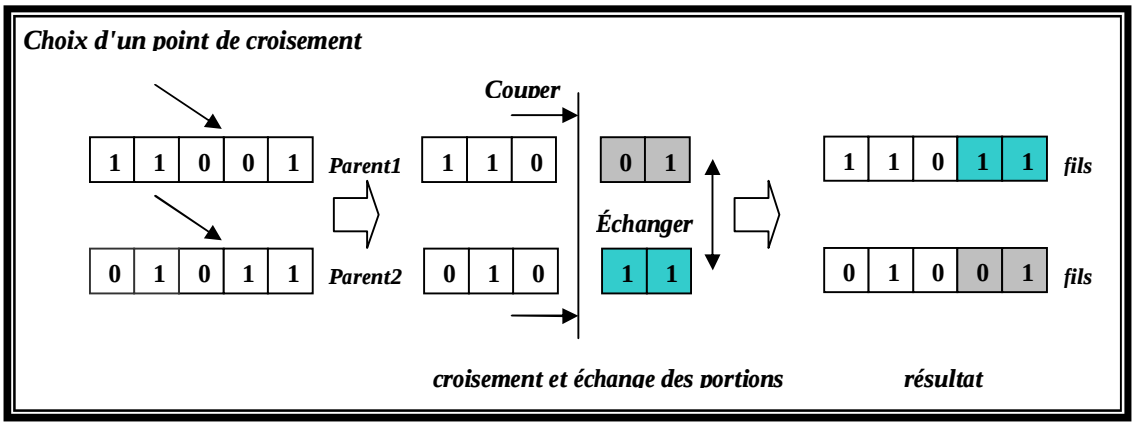


Figure 3.14. Un croisement à un point.

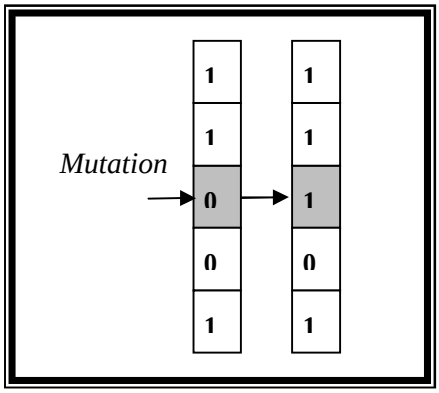


Figure 3.15 : mutation binaire.

Les AGs opèrent par la découverte des chromosomes les plus adaptés, en recombinant leurs régions par des opérations facilement exécutables en parallèle. Ceci a été exploré dans beaucoup de travaux au-dessus de la littérature des algorithmes génétiques. Cependant, les AGs présentes quelques limitations comme la lenteur et la grande taille de la population utilisée pour trouver la meilleure solution. Récemment plusieurs chercheurs étudient l'effet d'ajouter d'autres opérations inspirées d'autres domaines comme l'informatique quantique dans les algorithmes évolutionnaires. Une nouvelle classe d'algorithmes est donc née : les Algorithmes Quantiques Evolutionnaires AQEs.

Définition 3.1 : Un Algorithme Quantique Evolutionnaire (AQE) est un Algorithme Evolutionnaire (AE) qui combine entre les algorithmes évolutionnaires classiques et les concepts et principes de l'informatique quantique, tel que le bit quantique, la superposition d'états, la mesure, etc. un AQE a les mêmes caractéristiques d'un AE classique tel que la représentation des individus, la fonction d'évaluation et la dynamique de la population. Néanmoins, l'AQE utilise une représentation quantique au lieu des représentations usuelles des AEs. En effet, un AQE utilise le bit quantique (qubit) comme une représentation probabiliste, défini comme la plus petite unité d'information [Han et al, 01].

3.3.3 Une représentation quantique des chromosomes

Une nouvelle représentation est adoptée pour les algorithmes quantiques évolutionnaires. Chaque population est constituée d'un ensemble d'individus quantiques. Les individus en AQEs sont représentés par des registres de qubits. Un qubit est la plus petite unité d'information dans un AQE, défini par une paire de nombres (a, b) comme suit :

$$|y\rangle = \begin{pmatrix} a \\ b \end{pmatrix} \text{ Où } |a|^2 + |b|^2 = 1$$

$|a|^2$, $|b|^2$ est la probabilité d'avoir les états '0', '1' respectivement. Selon les principes de l'informatique quantique, un qubit peut être à l'état '1', à l'état '0' ou dans une superposition des deux. Un chromosome quantique est une chaîne de qubits défini par :

$$\begin{pmatrix} a_1\, a_2 \ldots a_m \\ b_1\, b_2 \ldots b_m \end{pmatrix} \text{Où : } |a_i|^2 + |b_i|^2 = 1,\ i=1,2,\ldots,\ m.$$

En comparant avec les représentations classiques des chromosomes, Cette représentation quantique a l'avantage de représenter une superposition linéaire des états. D'une autre façon, elle encode une superposition d'un ensemble de solutions en utilisant un seul registre de qubits. Cependant, la représentation classique ne permet qu'une seule représentation d'une solution. Par exemple : imaginons un problème dont la solution est représentée sur un nombre binaire de trois bits. Soit le système à trois qubits avec trois paires d'amplitudes suivant :

$$\begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} & \sqrt{3}/2 \end{pmatrix}$$

Ce système représente les états $|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle$ et $|111\rangle$.

$$1/4|000\rangle + \sqrt{3}/4|001\rangle - 1/4|010\rangle - \sqrt{3}/4|011\rangle + 1/4|100\rangle + \sqrt{3}/4|101\rangle - 1/4|110\rangle - \sqrt{3}/4|111\rangle$$

La représentation quantique en ket indique que les probabilités de représenter les états précédents sont respectivement : $1/16$, $3/16$, $1/16$, $3/16$, $1/16$, $3/16$, $1/16$ et $3/16$.

On voit clairement que la représentation quantique a une meilleure diversité que les autres représentations classiques. En effet, On a seulement besoin de trois qubits pour représenter les huit états possibles. Contrairement à cette représentation, il faut avoir huit registres classiques de trois bits pour encoder tous les états possibles.

3.3.4 Les opérations quantiques

Comme les algorithmes évolutionnaires classiques, les AQEs utilisent des opérations quantiques afin de diversifier la recherche, et de mener les individus vers les meilleures solutions. Ces opérations doivent satisfaire les conditions de la linéarité et de la réversibilité. Dans [Han et al, 01], une porte quantique est définie comme un opérateur de variation de l'AQE par lequel le qubit mis à jour doit satisfaire la condition de normalisation $|a'|^2 + |b'|^2 = 1$ où a' et b' sont les nouvelles valeurs du qubit mis à jours. Généralement, trois portes sont massivement utilisées dans un AQE [Han et al, 01] :

- *La porte de rotation* : Comme dans l'algorithme de Grover, la porte de rotation est utilisée comme une porte quantique dans l'AQE afin d'augmenter la probabilité de la bonne solution :

$$U(\Delta q_i) = \begin{bmatrix} \cos(\Delta q_i) & -\sin(\Delta q_i) \\ \sin(\Delta q_i) & \cos(\Delta q_i) \end{bmatrix}$$

Où : $\Delta q_i, i = 1,2,\ldots,m$ est un angle de rotation de chaque qubit vers l'état 0 ou 1 dépendant de son signe. Δq_i est un paramètre empirique dépendant du problème d'application. La figure 3.16 montre la courbe polaire de la porte de rotation. La table 3.3 peut être utilisée comme des

paramètres d'angle pour la porte de rotation. La pseudo code détaillé de la porte de rotation est le suivant [Han et al, 01]:

Procedure Update (**q**)
begin
i ← 0
while (i < m) **do**
begin
i ← i + 1
determine Δq_i with the lookup table
obtain (a_i', b_i') from the following :
if (**q** is located in the first/third quadrant)
then $[a_i', b_i']^T = U(\Delta q_i)[a_i, b_i]^T$
else $[a_i', b_i']^T = U(-\Delta q_i)[a_i, b_i]^T$
end
q ← q'
end

Angle	Valeur du bit de référence	β	α
+ $\delta\theta$	**1**	**> 0**	**> 0**
- $\delta\theta$	**0**	**> 0**	**> 0**
- $\delta\theta$	**1**	**< 0**	**> 0**
+ $\delta\theta$	**0**	**< 0**	**> 0**
- $\delta\theta$	**1**	**> 0**	**< 0**
+ $\delta\theta$	**0**	**> 0**	**< 0**
+ $\delta\theta$	**1**	**< 0**	**< 0**
- $\delta\theta$	**0**	**< 0**	**< 0**

Table 3.3. Lookup table.

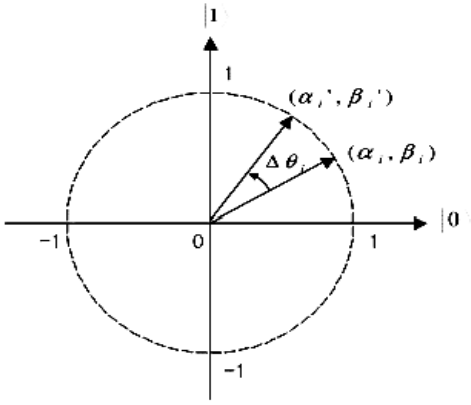


Figure 3.16. Rotation d'un qubit.

- *La porte NOT* : Cette porte opère de la même façon qu'une mutation naturelle. Elle opère sur un qubit en permutant les deux amplitudes de l'état 1(0) pour avoir l'état 0(1) respectivement (figure 3.17). Cette porte est très utile pour s'échapper du problème de l'optimum local.

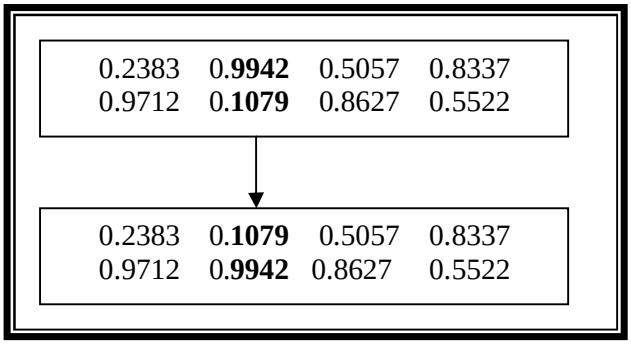


Figure 3.17. La porte NOT.

- *La porte Controlled-NOT* : Cette porte est utilisée pour effectuer des instructions contrôlées. Dans ce cas l'un des deux qubits doit être un bit de contrôle. Si le bit de contrôle est à 1, alors l'opérateur NOT est appliqué à l'autre bit (figure 3.18). cette porte est recommandée dans les problèmes contenant de grandes dépendances entre deux qubits.

$$C_{Not}: \quad |00\rangle \rightarrow |00\rangle \qquad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$
$$|01\rangle \rightarrow |01\rangle$$
$$|10\rangle \rightarrow |11\rangle$$
$$|11\rangle \rightarrow |10\rangle$$

Figure 3.18. La porte Controlled-NOT.

- *La porte de Hadamard* : Cette porte est appropriée pour les algorithmes qui utilisent l'information de phase des qubits aussi bien que l'information amplitude (figure 3.19).

$$U_{WH}|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \qquad U_{WH}|1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)$$

Figure 3.19. La porte de Hadamard.

3.3.5 Algorithmes quantiques génétiques

Afin d'augmenter la vitesse de convergence des AQEs plusieurs portes quantiques inspirées des algorithmes génétiques on été introduites. Un nouveau type des AQEs est donc apparu, il s'agit des Algorithmes Quantiques Génétiques (AQG) [Han et al, 01]. Ces algorithmes se distinguent sur les AQEs par l'introduction des opérateurs génétiques tel que la mutation et le croisement quantique.

Définition 3.2 : Un algorithme quantique génétique est similaire à un algorithme génétique classique. Cependant, il est enrichi par des opérations quantiques tels que l'enchevêtrement, la mesure, l'interférence, etc [Han et al, 00]. Sa structure ressemble aux algorithmes quantiques évolutionnaires.

3.3.6 Opérateurs génétiques quantiques

Dans l'expériences de knapsack, Han et Kim n'ont pas utilisé la mutation et le croisement dans leur AQG. Dans ce problème la mesure et l'interférence sont largement suffisantes pour avoir de bonnes solutions [Han et al, 00]. Cependant, les travaux de [Talbi et al, 04] et [Draa et al, 04] ont montré que les opérations de mutation et de croisement peuvent êtres utiles dans quelques problèmes d'optimisation tel que le recalage d'image.

• *Croisement quantique:* Le croisement quantique a le même principe que le croisement dans les algorithmes génétiques classiques. Cependant, il opère sur des chromosomes quantiques. Les chromosomes sont choisis d'une manière aléatoire. Ensuite, les deux chromosomes quantiques choisis échangent quelque qubits entre eux. L'exemple suivant explique le fonctionnement d'un croisement quantique. Le croisement combine entre les chromosomes quantiques *a* et *b* et produit deux chromosomes quantiques fils *c* et *d* en sortie. L'avantage de croisement quantique est l'exploration de nouvelles solutions [Draa et al, 04]. L'exemple suivant montre un croisement quantique au point 2.

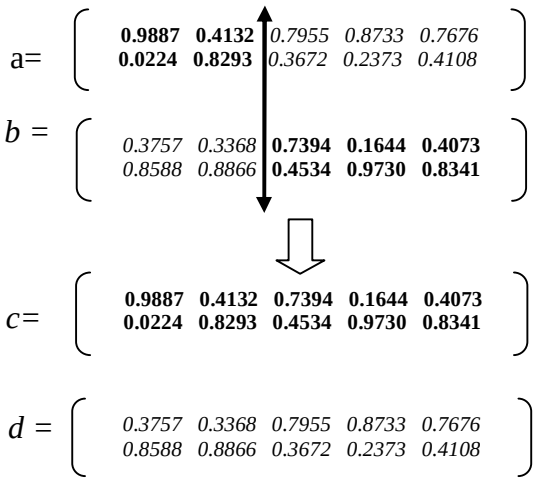


Figure 3.20. Croisement quantique.

• *Mutation quantique:* Comme dans le cas de la mutation classique, la mutation quantique effectue une perturbation sur un qubit. Cette mutation permute les deux composantes de l'amplitude d'un qubit. Elle opère de la manière suivante: si ce qubit est représenté par: $\alpha|0\rangle+\beta|1\rangle$, la mutation génère le qubit $\beta|0\rangle+\alpha|1\rangle$ (figure 3.21). Elle est très utile pour s'échapper des optimums locaux [Draa et al, 04].

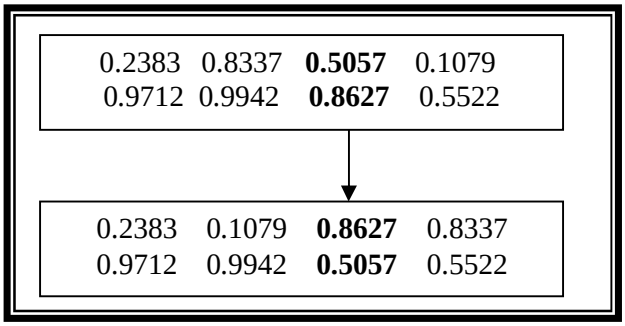


Figure 3.21. Mutation quantique.

3.3.7 Les avantages des AQEs et des AQGs

Les algorithmes inspirés du quantique présentent plusieurs avantages. Elles permettent un traitement parallèle de plusieurs données. Contrairement aux AG et AE classiques, le principe de la superposition permet de réduire considérablement la taille de la population dans les AQEs (AQGs) sans perdre d'informations. Les AQEs permettent une balance automatique entre la recherche locale et la recherche globale. Une grande caractéristique des AQEs est que l'historique des individus n'est pas perdu dans les itérations suivantes du à la nature probabiliste. Finalement, L'opération d'interférence permet l'accélération du processus de convergence vers la meilleure solution.

3.4 Conclusion

Il est tout à fait clair que le calcul quantique offre un gain considérable en temps et en espace. L'informatique quantique est caractérisée par des capacités de traitement et de stockage énormes. Cependant, les idées de l'informatique quantique ne sont pas encore exploitables vu le manque de moyens théoriques est surtout pratiques pour la conception d'un tel système. En l'attente de la construction des machines quantiques puissantes, la combinaison entre les approches classiques et l'informatique quantique constitue un domaine prometteur. En effet, les algorithmes quantiques évolutionnaires ont démontré leur efficacité dans plusieurs problèmes d'optimisation combinatoire. Pour cela nous présentons une approche basée sur un algorithme quantique évolutionnaire pour le célèbre problème d'alignement multiple de séquences dans le chapitre suivant.

Chapitre 4

Une approche quantique évolutionnaire pour l'alignement multiple de séquences

"La liberté, c'est la liberté de dire que deux et deux font quatre. Lorsque cela est accordé, le reste suit."
—George Orwell, 1984

4.1 Introduction

Nous avons vu dans ce qui a précédé que l'alignement multiple de séquences est une tâche ardue. En effet, aucune méthode n'a pu résoudre efficacement ce problème. Les méthodes exactes donnent théoriquement des solutions optimales, mais elles sont impraticables dans le cas de grandes familles de séquences. Par ailleurs, les méthodes progressives sont caractérisées par une grande simplicité et rapidité. Elles donnent toutefois des solutions moins optimales. Par contre, les méthodes itératives essayent de donner des solutions optimales dans des délais raisonnables par rapport aux méthodes exactes. Cependant, ce but reste encore loin d'être atteint.

Dans ce travail de magistère, on s'est intéressé à la résolution du problème d'alignement multiple de séquences en investiguant l'apport des approches quantiques évolutionnaires. Pour cela un noyau quantique évolutionnaire a été d'abord défini en première étape. Le développement de ce noyau a nécessité la définition d'une représentation quantique appropriée ainsi qu'une dynamique évolutionnaire opérant sur cette représentation (figure 4.1). Sur la base de ce noyau deux approches ont été proposées. L'approche QEAMSA [Meshoul et al, 05a] et l'approche QUANTALIGN [Meshoul et al, 05b]. La deuxième approche peut être vue comme un enrichissement et une amélioration de la première méthode.

Dans un souci de clarté, ce chapitre est organisé de telle sorte que à explorer ces points successivement après une formulation mathématique du problème traité. La validation des approches proposées est ensuite présentée.

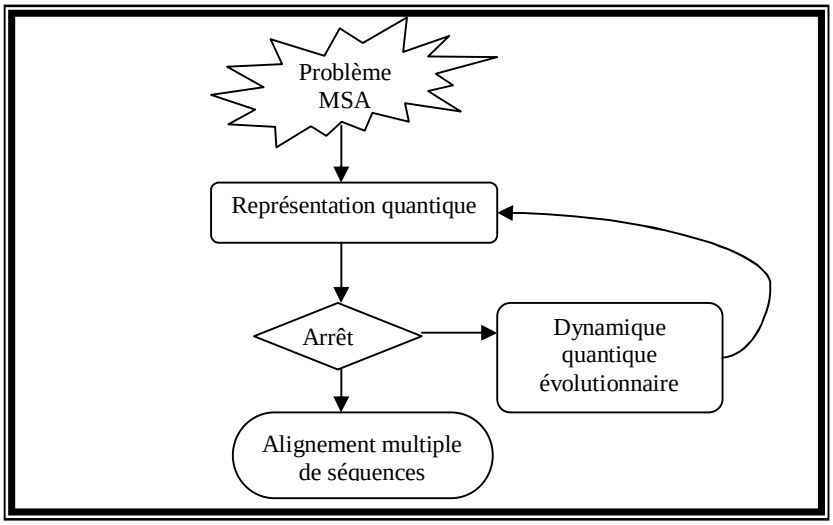


Figure 4.1. Noyau quantique évolutionnaire pour le problème d'alignement multiple de séquences.

4.2 Formulation du problème

L'alignement multiple de séquences consiste à aligner plusieurs séquences dans leur intégralité. Le MSA peut être formulé mathématiquement comme suit [Meshoul et al, 05a]:

Soit $Seqs = \{S_1, S_2, ..., S_n\}$ un ensemble de *n* séquences avec $n \geq 2$. Chaque séquence S_i est une chaîne de lettres définie sur un alphabet *A*. Les longueurs des séquences ne sont pas forcement les mêmes. Le problème MSA peut être défini par la spécification du couple (Ω, C) où Ω est l'ensemble de toutes les solutions possibles. *C* est une transformation de $\Omega \rightarrow R$ nommée score d'alignement. Chaque alignement multiple potentiel est vu comme un ensemble $ALN = \{S'_1, S'_2, ..., S'_n\}$ satisfaisant les conditions suivantes:

- Chaque séquence S'_i est une extension de S_i et elle est définie sur un alphabet $A' = A \cup \{-\}$. Le symbole "–" dénote un gap. Les gaps sont ajoutés à S_i de tel sorte que la suppression des gaps de S'_i donne S_i.
- Pour tous *i, j*: longueur (S'_i)=longueur (S'_j).

4.3 Définition d'un noyau quantique évolutionnaire

La définition de ce noyau repose fondamentalement sur une représentation quantique de l'espace de recherche associé au problème traité ainsi que la dynamique utilisée pour explorer l'espace de recherche en opérant sur la représentation quantique moyennant des opérations quantiques de base.

A. La représentation quantique d'un alignement multiple

Ayant un ensemble de séquences $Seqs = \{S_1, S_2, ..., S_n\}$, un alignement $ALN = \{S'_1, S'_2, ..., S'_n\}$ peut être vue comme une matrice binaire de taille *n, m* (figure 4.2). *n* est le nombre de séquences et *m* est le nombre de colonne dans la matrice d'alignements. Cette matrice satisfait les contraintes suivantes [Meshoul et al, 05a]:

- Ø Chaque ligne *i* représente une séquence S'_i
- Ø La valeur 1 représente un résidu (un acide aminé ou un nucléotide) et la valeur 0 représente un gap.
- Ø Le nombre de 1 dans chaque ligne *i* est égal au nombre de résidus dans la séquence *i*

L'exemple de la figure 4.2 suivante montre la représentation binaire d'un alignement multiple

```
A-CT-PN--H
A-CTTPPNNG
ATCTAP-RRG
ATC--PP--G
```

Représentation binaire →

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Figure 4.2. Une représentation binaire d'un alignement multiple.

Comme les algorithmes évolutionnaires classiques, on a utilisé des chromosomes quantiques pour encoder les alignements possibles. Chaque chromosome quantique contient un ensemble de registres quantiques. Chaque registre représente une séquence comme il est montré dans la figure 4.3 suivante :

$$\left(\begin{array}{c|c|c|c} a_1 & a_2 & \cdots & a_m \\ b_1 & b_2 & & b_m \end{array} \right)$$

Figure 4.3. Un registre quantique représentant une séquence.

Chaque colonne $\begin{pmatrix} a_i \\ b_i \end{pmatrix}$ représente un seul qubit et correspond à un élément de l'alphabet A'. Les a_i et b_i sont des valeurs réelles satisfaisantes la relation $|a_i|^2 + |b_i|^2 = 1$. Pour chaque qubit une valeur binaire est calculée selon les probabilités $|a_i|^2$, $|b_i|^2$ et le nombre de résidus dans chaque séquence. Par conséquence, tous les alignements potentiels peuvent être représentés par une matrice quantique (figure 4.4) contenant une superposition de toutes les configurations possibles.

$$\left[\begin{array}{c} \left(\begin{array}{c|c|c|c} a_{11} & a_{12} & \mathbf{K} & a_{1m} \\ b_{11} & b_{12} & & b_{1m} \end{array} \right) \\ \left(\begin{array}{c|c|c|c} a_{21} & a_{22} & \mathbf{K} & a_{2m} \\ b_{21} & b_{22} & & b_{2m} \end{array} \right) \\ \mathbb{M} \\ \left(\begin{array}{c|c|c|c} a_{n1} & a_{n2} & \mathbf{K} & a_{nm} \\ b_{n1} & b_{n2} & & b_{nm} \end{array} \right) \end{array} \right]$$

Figure 4.4. Représentation quantique de l'alignement multiple de séquences.

Cette matrice quantique peut être vue comme une représentation probabiliste de tous les alignements possibles. Cette représentation est efficace quand une approche évolutionnaire est adaptée car elle contribue à la réduction de la taille de la population des chromosomes.

B. Les opérations quantiques de base

Les opérations quantiques qui sont à la base de la dynamique quantique évolutionnaire proposées sont les suivantes.

1. *L'opération de mesure* : Cette opération transforme par projection la matrice quantique en une matrice binaire. Donc, on aura une solution parmi toutes les solutions présentées dans la superposition. Mais contrairement à la théorie quantique pure, cette mesure ne détruit pas la superposition. Cela a l'avantage de préserver la superposition pour les itérations suivantes sachant qu'on opère sur des machines classiques. La valeur d'un qubit est calculée suivant ses probabilités $|a_i|^2$, $|b_i|^2$ et le nombre de résidus dans chaque

séquence. La figure 4.5 montre une opération de mesure. La matrice binaire est ensuite traduite en une matrice alphabétique en respectant l'ordre d'apparition des bases dans les séquences à aligner (figure 4.6). La matrice alphabétique décrit un alignement possible par exemple l'alignement de l'ensemble de séquences {AGA, AA, AGCA, AA}.

$$\begin{bmatrix} \begin{pmatrix} |0.70 & |0.90 & |-0.90 & |0.77 \\ |0.70 & |0.44 & |0.44 & |0.63 \end{pmatrix} \\ \begin{pmatrix} |0.77 & |0.77 & |0.70 & |0.90 \\ |0.60 & |0.63 & |0.70 & |0.44 \end{pmatrix} \\ \begin{pmatrix} |0.63 & |0.44 & |0.99 & |1.00 \\ |0.77 & |0.90 & |-0.14 & |0.00 \end{pmatrix} \\ \begin{pmatrix} |0.70 & |0.63 & |0.77 & |0.99 \\ |0.70 & |0.77 & |0.63 & |0.14 \end{pmatrix} \end{bmatrix} \xrightarrow{mesure} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Figure 4.5. Projection d'une matrice quantique.

$$\begin{pmatrix} A & - & G & A \\ - & A & - & A \\ A & G & C & A \\ - & - & A & A \end{pmatrix}$$

Figure 4.6. Traduction d'une matrice binaire en un alignement multiple.

2. *L'interférence quantique* : Cette opération amplifie l'amplitude de la meilleure solution et diminue les amplitudes des mauvaises solutions. D'une autre façon, elle augmente la chance de la meilleure solution d'être mesurée par l'opération de mesure. Elle consiste essentiellement à déplacer l'état de chaque qubit dans la direction de la valeur du bit correspondant dans la meilleure solution en cours. L'opération d'interférence permet donc d'intensifier la recherche autour de la meilleure solution. Cette opération peut être accomplie en utilisant une transformation unitaire qui effectue une rotation dont les angles sont en fonction des amplitudes a_i, b_i (Figure 4.7) et de la valeur du bit correspondant dans la solution référence.

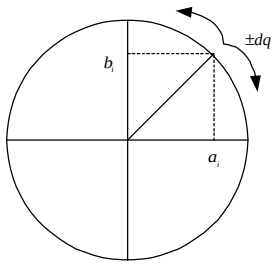


Figure 4.7. Interférence quantique.

Les valeurs de l'angle $\delta\theta$ sont choisies empiriquement de telle manière à éviter une convergence prématurée. La direction de la rotation est déterminée suivant la table 4.1.

3. *La mutation quantique* : Cette opération permet l'exploration de nouvelles solutions et l'augmentation des capacités de diversification du processus de recherche. Elle consiste à l'application d'une perturbation sur un ensemble de qubits de la matrice quantique.

Angle	Reference bit value	a_i	b_i
+δθ	1	> 0	> 0
-δθ	0	> 0	> 0
-δθ	1	> 0	< 0
+δθ	0	> 0	< 0
-δθ	1	< 0	> 0
+δθ	0	< 0	> 0
+δθ	1	< 0	< 0
-δθ	0	< 0	< 0

Table 4.1.Valeurs prises par l'angle de rotation.

4.4 Une approche purement itérative pour résoudre le problème MSA (QEAMSA)

Durant les premières étapes de notre travail notre objectif a été de démontrer la faisabilité des AQEs pour le problème d'alignement multiple de séquences. Il était donc une question de savoir comment exploiter le noyau quantique défini auparavant à travers une dynamique évolutionnaire. La réponse à cette question nous a conduit à définir une première version quantique évolutionnaire pour résoudre le problème traité. L'approche QEAMSA a été proposée à cet effet [Meshoul et al, 05a]. De manière abstraite elle opère comme suit:

Input: A set of sequences *S*

(1) Quantum matrix construction. Let *QM* be this matrix.
(2) Generate an initial alignment *S′*. Let *BM* be the corresponding binary matrix.
(3) Set $S_{best} = S'$ and $C_{best} = C(S')$.

Repeat
(4) Apply an interference operation on *QM* according to the best solution.
(5) Apply a mutation operation on *QM*.
(6) Apply a measurement operation on *QM* to derive a new binary matrix *BM*.
(7) Evaluate the current alignment *S′* corresponding to *BM*
(8) if $C(S_{best}) < C(S')$ then $S_{best} = S'$ and $C_{best} = C(S')$
Until a termination-criterion is reached

Output: S_{best} and $C(S_{best})$.

Nous décrivons maintenant comment notre approche résout le problème d'alignement multiple. QEAMSA commence par une solution initiale aléatoire et applique les opérations mentionnées ci-dessus pour produire d'autres solutions. D'abord dans l'étape initialisation une population de chromosomes quantiques est construite. Tous les chromosomes quantiques sont initialisés avec la même constantes $\frac{1}{\sqrt{2}}$. D'une autre façon, les chromosomes quantiques

représentent des superpositions linéaires de tous les états possibles avec la même probabilité. Dans le cas MSA, un chromosome quantique représente une superposition de tous les alignements possibles (figure 4.4).

$$\left| y_{q_j^0} \right\rangle = \sum_{k=1}^{2^m} \frac{1}{\sqrt{2^m}} \left| X_k \right\rangle \qquad (4.1)$$

Où X_k est le k^{ieme} état présenté par la chaîne binaire $(x_1 x_2 ... x_m)$, où x_i, i = 1,2,…, m. Il représente une superposition de toutes les configurations possibles d'une séquences dans un alignement.

L'étape suivante extrait une matrice binaire *BM(0)* par l'observation (mesure) des états de la matrice quantique *QM*(0), où $BM(0) = \{x_1^0, x_2^0, ..., x_n^0\}$ à la génération *0*. La solution binaire est une chaîne binaire de longueur m. Elle est formée par une suite de '0' et '1' obtenue par la mesure de la matrice *QM* en utilisant les probabilités $|a_i|^2$, $|b_i|^2$ et le nombre de résidus dans chaque séquence. Ensuite, on traduit la solution binaire *BM(0)* en une matrice alphabétique en utilisant l'ensemble de séquences à aligner. L'alignement initial est évalué en utilisant la fonction objectif SP avec une fonction constante pour pénaliser les gaps. A la fin de l'étape d'initialisation, la solution initiale est donc sélectionnée et sauvegardée.

Dans chaque étape de la boucle *"repeat"*, on applique d'abord l'opération d'interférence. Elle permet l'amplification des amplitudes des qubits de la bonne solution et donc sa probabilité d'être mesuré. Deuxièmement, une opération de mutation est appliquée sur le chromosome quantique *QM*. La méthode QEAMSA utilise une opération de mutation simple. C'est une opération "SWAP" (figure 4.8) permettant la permutation de deux qubits choisis aléatoirement [Meshoul et al, 05a].

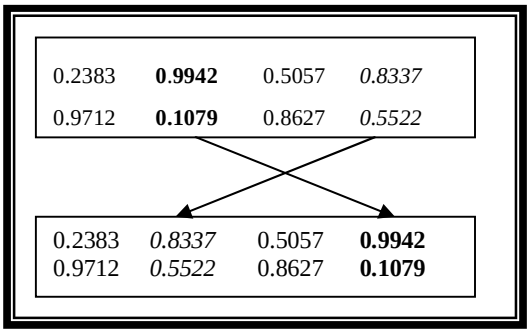


Figure 4.7. Opération "SWAP".

L'opération de mutation est suivie de l'opération de mesure quantique. Cette dernière permet d'avoir une autre matrice binaire *BM(t+1)*. La matrice binaire *BM(t+1)* est ensuite traduite en une matrice alphabétique en utilisant l'ensemble de séquences à aligner. Finalement, on calcule la le score SP de la solution obtenue. Si la solution obtenue est meilleure que celle enregistrée alors on garde cette solution comme la meilleure solution courante sinon elle est rejetée. Le processus est réitéré jusqu'à la satisfaction des conditions finales.

Pour évaluer la performance de notre méthode, des expériences ont été entreprises sur des ensembles de séquences. Nous avons évalué QEAMSA en utilisant des ensembles de séquences nucléiques et protéiques obtenus à partir des bases de données NCBI (http://www.ncbi.nlm.nih.gov/) pour les séquences nucléiques et de la base BALIBASE [Thompson et al, 99a] pour les séquences protéiques. Pour calculer le score d'un alignement, nous avons utilisé la matrice de substitutions représentée sur la table 4.2 pour le cas des séquences d'ADN. Elle prend en compte les substitutions/conservations et également les pénalités de gaps. Pour le cas des séquences de protéines, la matrice BLOSUM30 a été exploitée. Les résultats obtenus sont comparés à ceux donnés par la méthode CLUSTAL en utilisant les mêmes matrices de score.

	A	**C**	**G**	**T**	**-**
A	1.36	-1.60	-0.37	-1.60	-2.00
C	-1.60	1.36	-1.60	-0.37	-2.00
G	-0.37	-1.60	1.36	-1.60	-2.00
T	-1.60	-0.37	-1.60	1.36	-2.00
-	-2.00	-2.00	-2.00	-2.00	0.00

Table 4.2. Matrice de substitution pour les séquences ADN.

Dans la table 4.3, les différents scores obtenus par notre algorithme (QEAMSA) sont comparés à ceux donnés par la méthode CLUSTALX pour quatre ensembles de séquences nucléiques et trois ensembles de séquences de protéines. Il est évident que QEAMSA fournit de meilleurs résultats que CLUSTALX [Meshoul et al, 05a].

Set	**Reference**	**Number and (length) of sequences**	**Align. score using QEAMSA**	**Align. score using Clustal X**
Phyto6	*Phytophthora infestans (Nucleic)*	6 (16-30)	-323.27	-397.81
sap5	*Homo sapiens (Nucleic)*	5 (78-120)	-942.06	-943.29
coli	*E.coli (Nucleic)*	4 (24-32)	84.14	68.67
Phyto12	*Phytophthora infestans (Nucleic)*	11 (10-30)	-1239	-1253.2
1aab_Ref1	*Ref1 (BALIBASE)*	4 (67-79)	338	329
1boa_Ref1	*Ref1 (BALIBASE)*	5 (49-80)	-1740	-2051
1boa_Ref2	*Ref2 (BALIBASE)*	16 (49-80)	-1050	-1087

Table 4.3. La comparaison entre QEAMSA et CLUSTAL X.

Pour étudier le comportement de QEAMSA qui est basé sur une recherche stochastique par rapport à CLUSTALX nous avons enregistré les scores d'alignement fournis au cours de plusieurs exécutions et on les a comparé aux résultats de CLUSTALX. Ces résultats illustrés sur la figure 4.9 montre la capacité de QEAMSA à réaliser des résultats supérieurs à CLUSTALX dans toutes les exécutions.

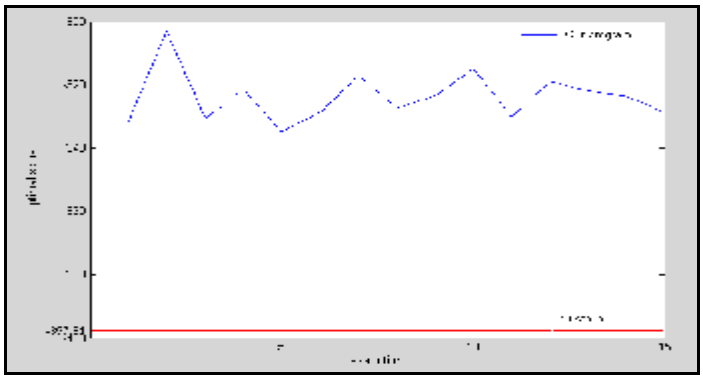


Figure 4.9. Le comportement de QEAMSA w.r.t CLUSTALX.

4.5 Une approche progressive itérative pour l'alignement multiple en utilisant un algorithme quantique évolutionnaire (QUANATALIGN)

Au cours des expériences intensives sur QEMSA, on a constaté que QEAMSA est lent. Il nécessite des heurs et des heures pour aligner de larges ensembles de séquences contenant plus de 16 séquences de taille moyenne égale à 100. Afin, de rendre cette approche praticable, on a pensé a une autre idée. Pourquoi ne pas de combiner les méthodes progressives et les méthodes itératives. L'idée est simple, on utilise une solution crée par une méthode progressive comme une solution initiale dans l'algorithme quantique évolutionnaire [Meshoul et al, 05b]. Ensuite, on applique des raffinements successifs sur cette solution en utilisant une optimisation itérative basée sur un AQE. En effet, démarrer avec une bonne solution initiale va réduire considérablement le temps de convergence vers la solution optimale. Une autre caractéristique de cette nouvelle approche est la possibilité d'optimiser n'importe qu'elle fonction objectif. En effet, plusieurs types de fonctions ont été adoptés: la SP, la WSP et la T_COFFEE (voir chapitre 2). Cependant, on a opté pour la fonction WSP comme fonction objectif standard pour notre méthode. Ce choix est motivé par la qualité des solutions obtenues par WSP par rapport à la fonction SP. En plus, WSP est plus rapide et facile à implémenter par rapport à T-COFFEE. Un autre grand avantage de la méthode proposée nommée QUANTALIGN est l'utilisation d'un ensemble de mutations bien définies afin d'accélérer la vitesse du processus d'optimisation [Meshoul et al, 05b]. En effet, le problème capital dans la plupart d'algorithmes d'alignement est le placement inapproprié des gaps dans les séquences. Par exemple, la figure 4.10 à gauche montre des gaps de la même taille apparaissant à des positions différentes dans l'alignement de séquences. Cela est mauvais du point de vue biologique. En effet, C'est moins probable que des insertions/délétions "indels" de la même taille ce produit à des positions différentes. Dans ce cas, on doit combiner les gaps proches en un seul bloc (figure 4.10 à droite).

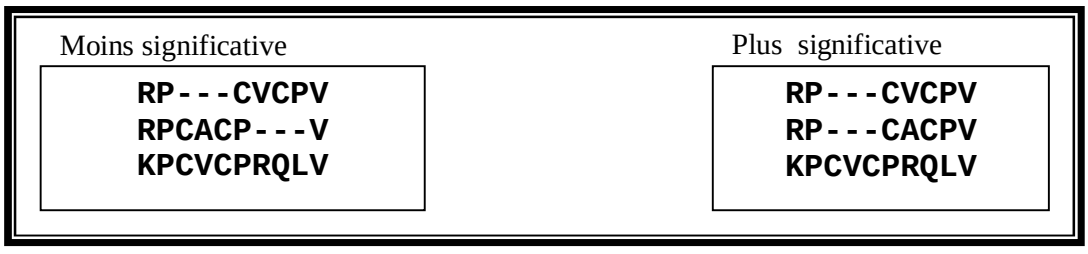


Figure 4.10. Des gaps de même taille dans des positions différentes.

Un autre type de mauvais placement des gaps est la présence des îles de caractères (island) entouré par des gaps (figure 4.11 à gauche).

Moins significative	Plus significative
RPC----CVC----PVC	**RPCCVC--------PVC**

Figure 4.11. La présence des îles dans une séquence d'un alignement.

Par ailleurs, La figure 4.12 décrit un autre exemple de mauvais placement des gaps. Dans la réalité, c'est moins probable que deux "indels" apparaissent très proches l'un de l'autre.

Moins significative	Plus significative
RP-C--CVPV **RPCACPL-PV** **KPCVCPRQLV**	**RPC---CVPV** **RPCACPL-PV** **KPCVCPRQLV**

Figure 4.12. Fusionnement de deux gaps très proches dans la même séquence.

Une bonne méthode itérative d'alignement doit prendre en compte des problèmes ci-dessus. Afin de traiter le problème de mauvais placement de gaps dans notre méthode, on a redéfini et réadapté quelques opérations de mutation. On a crée des opérations de mutation qui opèrent sur des suites et des blocs de qubits. Dans QUATALIGN, sept opérations de mutation ont été créées qu'on peut classifier en trois grandes classes [Meshoul et al, 05b].

- *Mutation d'un seul qubit*: Dans cette catégorie, on a deux opérateurs de mutation, le premier prend un qubit aléatoire correspondant à un résidu et examine ces voisins. Si l'un de ces voisins est un qubit correspondant à un gap alors il permute les deux qubits (figure 4.13). Biologiquement, c'est rare que les gaps se produisent à des distances très proches (figure 4.14.a). L'autre type de mutation est de prendre un qubit aléatoire correspondant à un résidu est de permuter avec un autre qui correspond à un gap (figure 4.14.b).

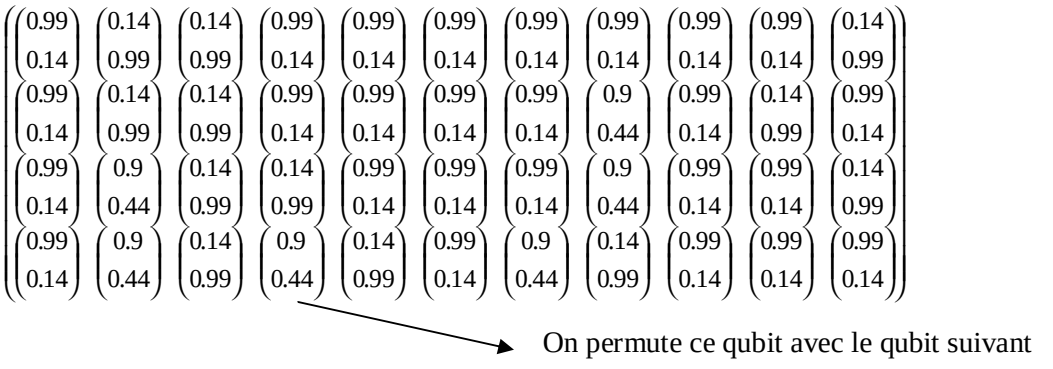


Figure 4.13. Mutation basée qubit.

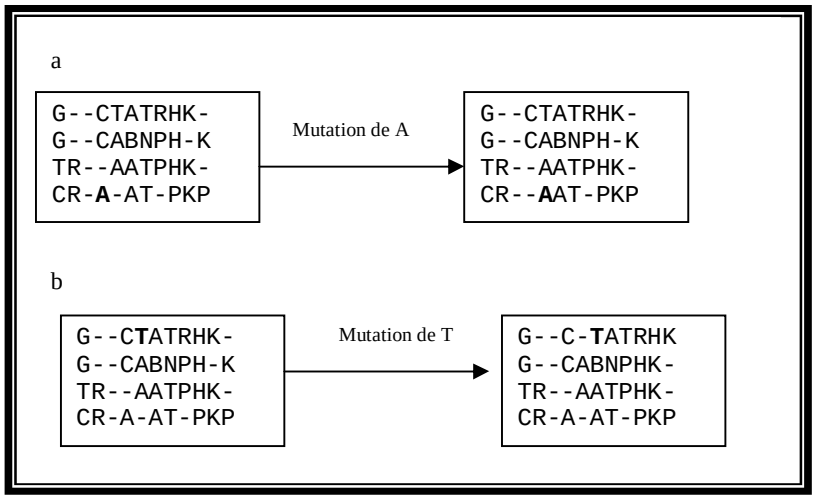


Figure 4.14. L'effet de la mutation basée qubit.

Ø *Mutation d'une suite de qubits*: Ce type de mutation déplace une suite de longueur aléatoire de qubits qui représente une suite de résidus ou de gaps vers la droite ou la gauche. L'effet de cette mutation sur l'alignement est montré dans la figure 4.15.

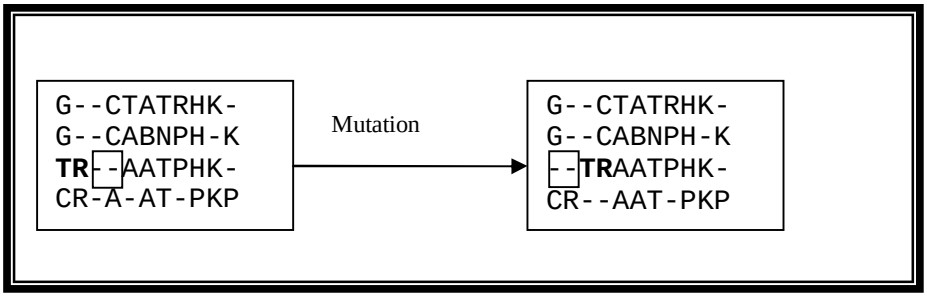


Figure 4.15. L'effet de la mutation d'une suite de résidus.

Ø *Mutation d'un bloc de qubits*: Un bloc de qubits correspondant à un bloc de gaps ou de résidus est déplacé (figure 4.16). L'avantage de ce type de mutation est qu'elle affecte un grand nombre de séquences. Les mutations simples peuvent gaspiller inutilement le temps CPU, parce que à chaque fois on doit évaluer la solution par la fonction objectif.

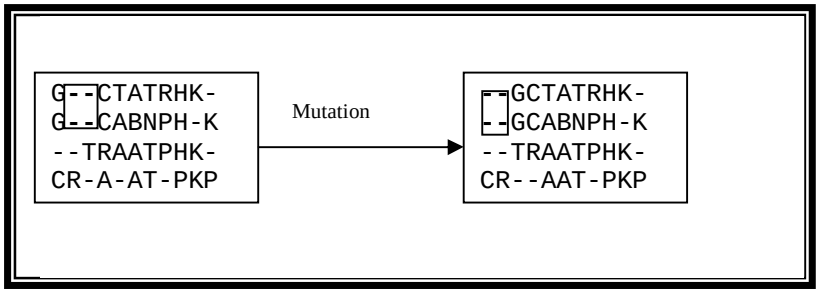


Figure 4.16. L'effet de la *mutation d'un bloc de gaps.*

4.5.1 Structure et fonctionnement de la méthode QUANTALIGN

La méthode QUATALIGN commence d'abord par le calcul d'une solution initiale. Cette dernière peut être calculée directement par la méthode QUANTALIGN (une solution propre à notre méthode), ou bien utiliser une solution créée par une autre méthode progressive

comme par exemple CLUSTAL [Meshoul et al, 05b]. Notre approche calcule la solution initiale en se basant sur l'algorithme de Feng et Dolittle [Feng et al, 87]. Les étapes de la construction de cette solution sont les suivantes :

1. Calculer la matrice de distance de tous les n (n-1)/2 alignements de paires possibles en utilisant un algorithme d'alignement global (Needleman et Wunsch)
2. Construire l'arbre guide en utilisant la matrice de distance obtenue dans l'étape précédente.
3. Aligner chaque séquence selon l'ordre donné par l'arbre guide

L'arbre guide construite dans l'étape 2 de l'algorithme de Feng et Dolittle est utilisé pour inférer les poids des séquences pour une éventuelle utilisation dans la fonction WSP. L'avantage des poids des séquences est pour réduire le score des espèces proches et augmenter celui des séquences divergentes. Les poids dépendant de la distance de la racine de l'arbre mais également des séquences qui ont une branche commune avec d'autres séquences partagent le poids dérivé de la branche partagée. La figure 4.17 montre comment calculer les poids de trois séquences A, B et C.

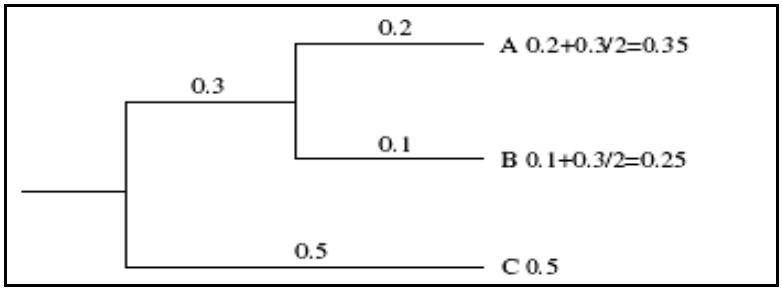


Figure 4.17. Le calcul des poids des séquences à partir de l'arbre phylogénétique.

La méthode QUATALIGN est une méthode flexible. Elle permet l'utilisation de plusieurs types de méthodes pour la construction des arbres phylogénétiques. Apres le calcul de la solution initiale et son score d'alignement, on crée une population de chromosomes quantiques et on transforme la solution alphabétique en solution binaire.

Le noyau de notre approche consiste en l'application cyclique des opérations d'interférence, de mutation et de mesure (figure 4.18). Contrairement à l'approche QEAMSA, dans chaque itération, la méthode QUANTALIGN choisit d'une manière aléatoirement un seul type de mutation parmi les sept opérations de mutations définies dans QUANTALIGN. Comme QEAMSA, à la fin de chaque itération, on évalue l'alignement obtenu à la suite à l'opération de mesure en utilisant l'une des fonctions objectifs supportées par QUANTALIGN (WSP, SP, T-COFFEE). On met à jour la meilleure solution globale dans le cas où la solution trouvée est meilleure que celle enregistrée. Le processus est répété jusqu'à satisfaire des critères d'arrêt.

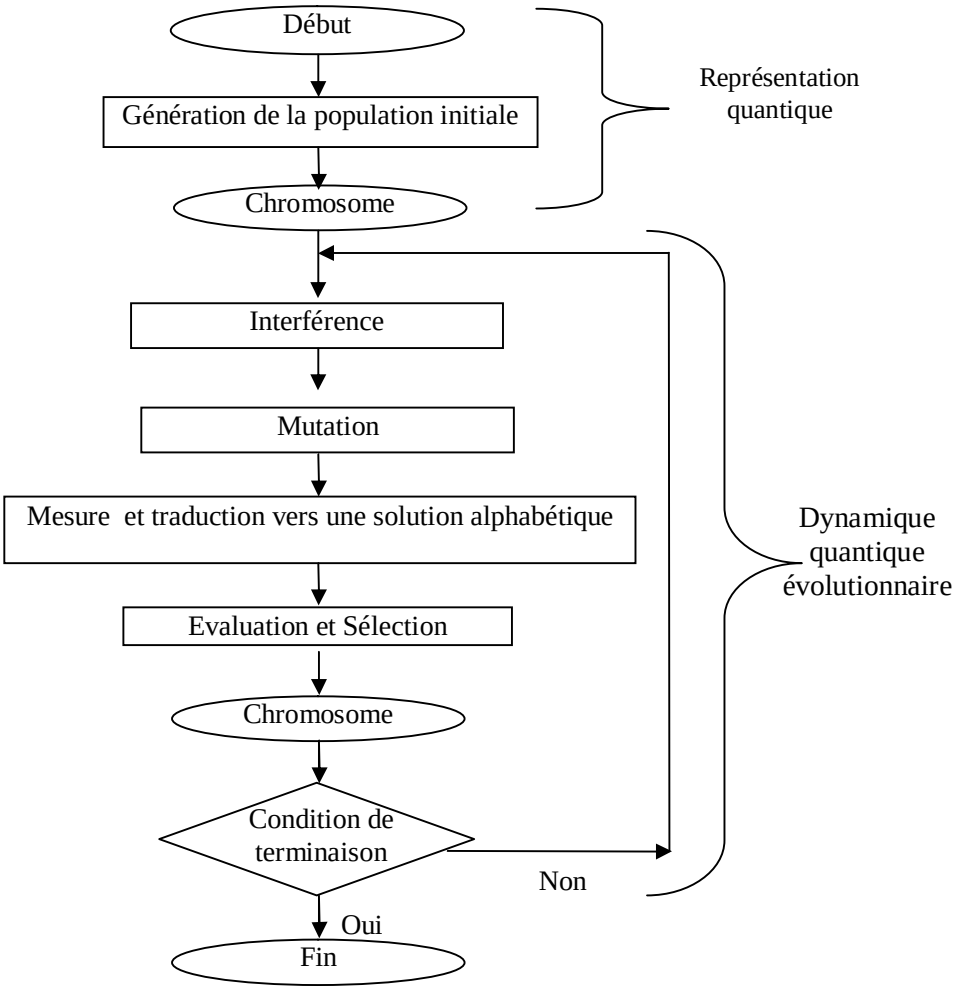


Figure 4.18. La structure générale de QUANTALIGN.

4.5.2 Implémentation et évaluation de la méthode QUANTALIGN

L'outil MATLAB7 a été adopté pour l'implémentation des deux approches décrites précédemment. Le choix de ce langage est motivé par la manipulation facile et efficace des vecteurs et matrices par MATLAB alors que la plupart des données dans QUANTALIGN et QEAMSA sont sous forme matricielle (la matrice quantique, la matrice binaire, la matrice des séquences…). Un autre avantage de MATLAB est sa portabilité, ce qui permet d'implémenter notre approche sur différentes plateformes. Néanmoins, MATLAB peut être lent face aux boucles parce qu'il est un langage interprété. Ce qui augmente donc le temps d'exécution par rapport aux autres langages compilés tels que C.

Pour évaluer la qualité des alignements obtenus par l'approche QUANTALIGN, on a utilisé la base d'alignements de référence BALiBASE [Thompson et al, 99a]. Cette dernière est amplement utilisée et suffisante pour l'évaluation des performances des méthodes d'alignement. Nous avons également utilisé un ensemble de tests de structures ARN obtenus de la base BRALiBASE [Gardner et al ,05]. L'objectif est de déterminer la capacité de QUANTALIGN dans la prédiction des structures ARN.

Pour la validation statistique de nos résultats, on a utilisé le test de Friedman utilisé dans [Thompson et al, 99b] et le test "Wilcoxon signed rank" utilisé dans [Gotoh, 96]. Ces deux tests sont utilisés pour déterminer les différences de performances entre les méthodes d'alignement multiple et trouver donc les méthodes qui réalisent efficacement les alignements d'une référence par rapport à d'autres méthodes. Le test de Friedman est utilisé pour tester si la différence entre les médianes des méthodes d'alignement n'est pas significative. Par ailleurs, le test *Wilcoxon signed rank* teste si la médiane de la différence entre les résultats de deux méthodes est égale à 0. Le résultat de test d'hypothèse, effectué au niveau de signification de 5% ou 1%, est renvoyé dans une valeur booléenne h. si h =0, les deux méthodes ont une distribution symétrique avec la médiane zéro, c.à.d que la différence entre les performances de deux méthodes n'est pas significative. Par ailleurs, Dans le cas où h=1, les deux méthodes d'alignement sont significativement différentes. L'évaluation statistique des résultats des programmes est réalisée par MATLAB. Les figures des tests de Friedman ci-dessous montrent les programmes réussis en lignes continues, les programmes non réussis en lignes discontinues et la référence en ligne pointillée.

4.5.3 Évaluation avec la base de référence BALiBASE

Cette base contient 142 alignements de référence, divisés en cinq ensembles de références. Chaque ensemble contient au moins 12 alignements représentatifs (table 4.4). Les alignements sont manuellement validées pour assurer l'alignement de résidus fonctionnels et autres conservés. Des blocs noyaux sont définis pour chaque alignement en tant des régions qui peuvent être sûrement alignées (voir chapitre2).

Référence	Courte (<100 résidus)	Moyenne (200-300 résidus)	Longue (>400 résidus)
Référence 1: séquences équidistantes de longueurs similaires			
V1 (<25% identité)	7	8	8
V2 (20-40% identité)	10	9	10
V3 (>35% identité)	10	10	8
Référence 2: famille versus orpheline	9	8	7
Référence 3: familles équidistantes divergentes	5	3	5
Référence 4: extension N/C-terminal	12		
Référence 5: insertions	12		

Table4. 4. La base de référence BALiBASE

Pour estimer la qualité biologique des alignements obtenus par QUANTALIGN on a utilisé le programme *BALiBASE-score* et les fichiers annotations qui contiennent les blocs noyaux. Ce programme disponible sur le site de BALiBASE donne deux mesures : SPS et CS. Le SPS (La somme des pairs) donne le pourcentage des pairs correctement alignés. Cependant le CS (le score des colonnes) donne le pourcentage de colonnes correctement alignées (chapitre 2).

Ø Résultats et discussions

A. Référence 1 : un nombre restreint de séquences approximativement équidistantes

Ce test est destiné à tester l'effet de la taille et l'homologie des séquences sur la performance des programmes d'alignement. Dans ce test notre programme est clairement meilleur que tous les autres programmes. Dans l'ensemble V1 qui contient des séquences difficiles à aligner avec moins de 25% d'identité, QUANTALIGN donne de meilleurs résultats. Dans le test de Friedman avec un seuil a=0.05, QUANTALIGN et PRRP sont les seuls programmes réussis dans V1 (figure 4.19). Dans les autres sous ensembles V2 et V3, QUANTALIGN est aussi bon ou meilleur que les autres programmes (table 4.5).

	PRRP	CLUSTAL	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
v1	0,692	0,647	0,592	0,487	0,536	0,504	0,559	0,571	0,730
v2	0,935	0,932	0,920	0,893	0,910	0,909	0,927	0,911	0,935
v3	0,968	0,970	0,962	0,922	0,961	0,955	0,960	0,962	0,969
AVR	0,877	0,864	0,841	0,788	0,821	0,810	0,834	0,832	0,889

Table 4.5. La moyenne des scores des alignements de la référence 1.

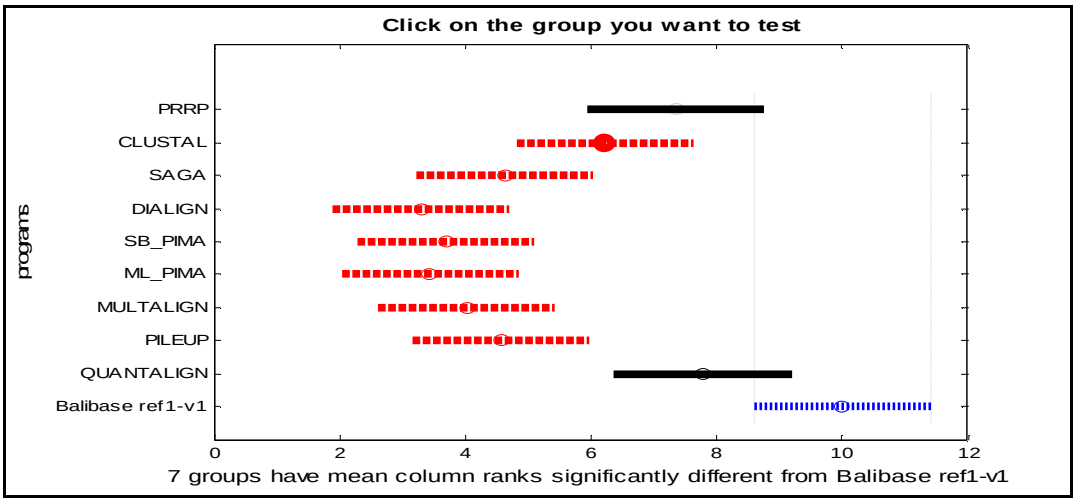


Figure 4.19.Test de Friedman (a=0.05) pour la référence 1.V1

B. Référence 2 : une famille relative avec des ordres divergents et orphelins

Dans ce test on examine la capacité des programmes d'aligner les séquences orphelines divergentes (10-20% identité avec la famille et entre les orphelines) avec une famille hautement liée (> de 25% d'identité) (table 4.6). Dans le test de Friedman (a=0.05) tous les programmes échouent dans ce test par rapport à la référence. Cependant avec un seuil a=0.01, seulement SAGA et CLUSTAL sont les seuls programmes réussis dans ce test. D'un autre coté, la comparaison de notre programme avec le programme CLUSTAL montre qu'il n'existe pas une différence significative (test Friedman a=0.05, figure 4.120). Cependant, les programmes locaux DIALIGN, ML_PIMA et SB_PIMA sont moins bons que CLUSTAL.

PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
0,541	0,583	0,586	0,384	0,379	0,371	0,517	0,429	0,425

Table 4.6. La moyenne des CS scores des alignements de la référence 2

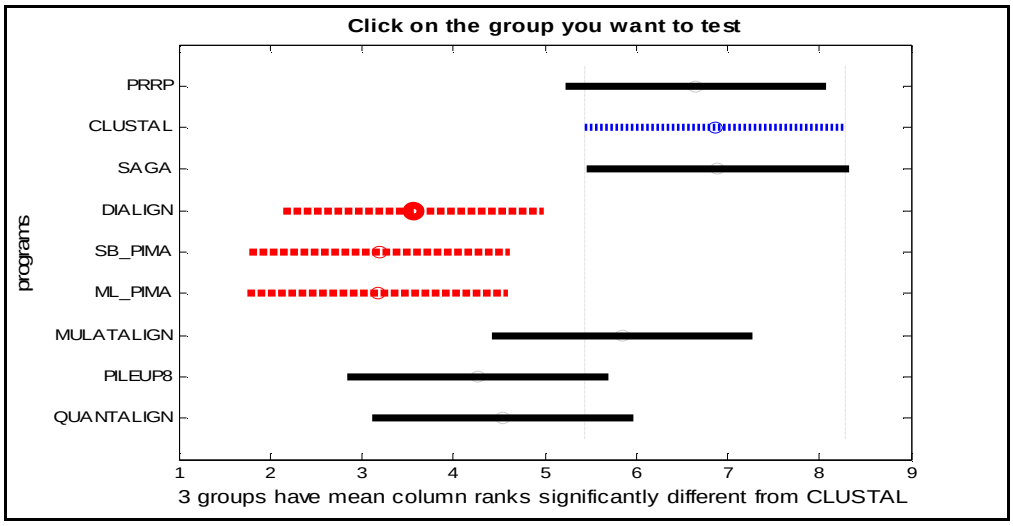


Figure 4.20. Test de Friedman (a=0.05) pour la référence 2. CLUSTAL (dot) par rapport aux autres programmes sans la référence BALIBASE.

C. Référence 3 : familles des ordres relatifs

Ce test est conçu pour évaluer la capacité des programmes à aligner correctement les familles divergentes approximativement équidistantes (<20% d'identité) composées de séquences fortement liées (>25% d'identité) dans un alignement multiple simple. Le CS est employé dans ce test, car c'est un meilleur estimateur de la qualité de l'alignement entre les familles. Le SPS utilisé précédemment est davantage influencé par la qualité de l'alignement dans les familles. Dans ce test les résultats de QUANTALIGN sont comparables avec ceux des meilleurs programmes (table 4.7). Dans cette référence, seulement les programmes itératifs SAGA, PRRP et QUANTALIGN réussissent dans le test de Friedman (0.05) (figure 4.21).

PRRP	**CLUSTALX**	**SAGA**	**DIALIGN**	**SB_PIMA**	**ML_PIMA**	**MULTALIGN**	PILEUP8	QUANTALIGN
0,532	0,446	0,506	0,314	0,267	0,372	0,303	0,323	0,467

Table4.7. La moyenne des CS scores des alignements de la référence 3

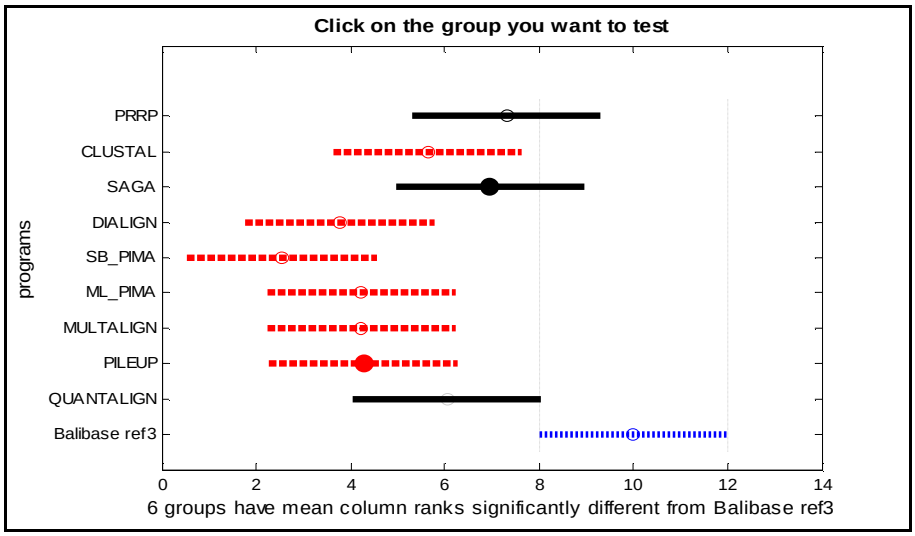


Figure 4.21. Test de Friedman (a=0.05) pour la référence 3.

D. Référence 4 : Prolongements de N/C-Terminal

Contrairement aux tests précédents où les séquences ont des longueurs semblables, la référence 4 contient des séquences avec de grandes extensions de N/C-terminal. L'utilité de ce test est d'étudier si les programmes sont capables d'aligner les blocs de noyaux flanquants les extensions. Aucune grande insertion interne n'est présentée à ce stade. Notre programme et PILEUP8 sont les seuls programmes globaux réussis dans ce test. Cependant, les méthodes itératives PRRP, SAGA échouent dans ce test (Friedman 0.05, figure 4.22). Par ailleurs, les méthodes locales (DIALIGN) sont très réussies dans ce test (table 4.8).

PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
0,323	0,361	0,289	0,853	0,794	0,705	0,292	0,710	0,681

Table 4.8. La moyenne des CS scores des alignements de la référence 4

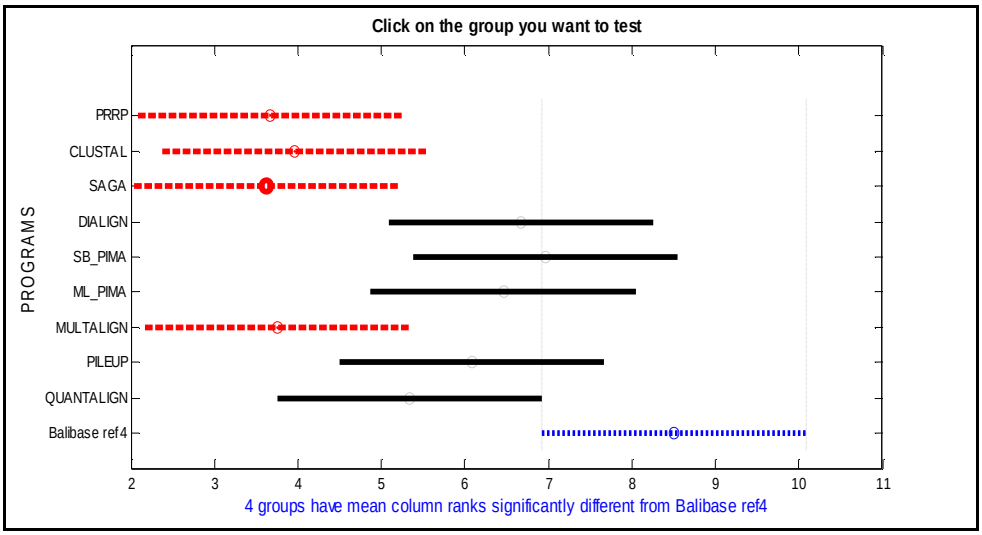


Figure 4.22. Test de Friedman (a=0.05) pour la référence 4.

E. Référence 5 : insertions internes

Ce test contient également des séquences de longueur inégale, mais contrairement à la référence 4, les insertions sont internes aux domaines homologues et pas au N/C-terminal.. Dans ce test notre méthode et DIALIGN sont de loin les meilleurs programmes (table 4.9). Cependant, CLUSTAL et SAGA échouent dans le test de Friedman (figure 4.23)

PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
0,700	0,705	0,642	0,836	0,508	0,572	0,627	0,639	0,807

Table 4.9. La moyenne des CS scores des alignements de référence 5.

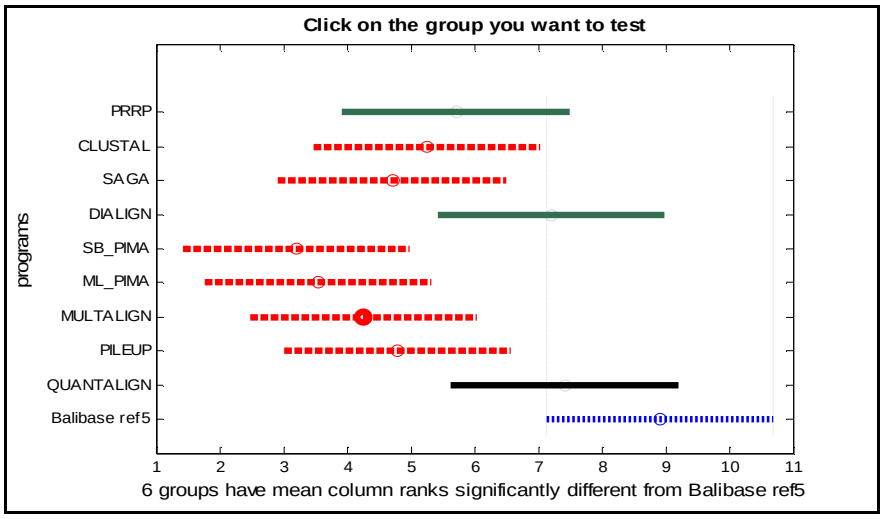


Figure 4.23. Test de Friedman (a=0.05) pour la référence 5.

La table 4.10 montre les différents tests de Wilcoxon de QUANTALIGN par rapport aux autres méthodes. Les résultas obtenus confirment ceux obtenus par les tests de Friedman.

Méthode	PRRP	CLUSTAL	SAGA	DIALIGN
Référence BALiBASE	h	h	h	h
Ref1_V1(0,05)	0	1	1	1
Ref1_V2(0,05)	0	0	0	1
Ref1_V3(0,05)	0	0	1	1
Ref2(0,01)	0	0	1	0
Ref3(0,05)	0	0	0	1
Ref4(0,05)	1	0	1	0
Ref5(0,05)	1	0	1	0

Table 4.10. L'évaluation statistique des résultats de Quantalign en utilisant le test de "Wilconxon signed rank". (h=0: les programmes ont statistiquement les mêmes performances. h=1: les performances des programmes sont différentes).

4.5.4 L'évaluation de notre méthode dans la prédiction structurelle des ARNs

Dans ce test on examine la capacité de notre programme dans la prédiction des structures des ARN. La prédiction structurelle des ARN est un problème non encore résolu en bioinformatique. Plusieurs approches on été suggérées pour résoudre ce problème [Gardner et al ,05]. Malheureusement, aucune approche n'a réussi dans ce domaine. Dans cette étude, on teste l'efficacité de la méthode QUANTALIGN face à d'autres programmes sophistiqués dans la prédiction des structures ARN. Pour cela, on a utilisé plusieurs tests obtenus de la base de benchmarks BRALIBASE contenant des ensembles de tests structures ARN. Ces tests sont divisés en cinq références: Intron, rRNA, SRP, tRNA et U5. Pour l'évaluation des programmes d'alignement dans la prédiction structurelle, on a utilisé deux mesures indépendantes. La première est le fameux SPS score qui mesure dans ce cas la sensitivité de la prédiction structurelle de l'ARN. L'autre mesure la plus importante est l'index de la conservation de la structure (Structure Conservation Index) SCI. Cet index donne la mesure de la structure secondaire conservée dans un alignement. Contrairement au score SPS, cette

mesure est indépendante de l'alignement de référence. Un SCI est proche de zéro s'il n'existe pas des structures ARN communes entre les différentes séquences. Cependant les structures conservées parfaites ont un SCI proche de 1. Par ailleurs, le cas où SCI >1 indique qu'il existe une structure secondaire ARN qui, en outre, est supportée par une préservation compensatoire et/ou conformée de mutations de structures communes. Nous notons que le SCI marque seulement l'exactitude de l'alignement en termes d'information secondaire de structures. Ces deux mesures sont données par le programme RNAz de BRALIBASE. On a comparé nos résultats avec ceux des programmes CLUSTAL, MAFFT, DIALIGN et PROALIGN. Les résultats de ces programmes sont disponibles sur le site de la base BRALIBASE (http://www.binf.ku.dk/users/pgardner/bralibase/).

Ø Résultats et discussions

Les résultats de notre étude sur la capacité de notre approche dans la prédiction structurelle ARN est décrites dans la table 4.11. Les résultats indiquent clairement que QUANTALIGN peut être utile dans la prédiction structurelle. En effet, nos résultats sont comparables aux autres programmes (tables 4.12, 4.13, figures 4.24, 4.25, 4.26, 4.27).

		QUANTALIGN		**CLUSTAL**		**DIALIGN**		**MAFFT**		**PROALIGN**	
		SPS	**SCI**	**SPS**	**SCI**	**SPS**	**SCI**	**SPS**	**SCI**	**SPS**	**SCI**
Intron	aln11	0,61	0,65	0,659	0,69	0,352	0,69	0,554	0,54	0,401	0,69
	aln12	0,592	0,66	0,68	0,69	0,673	0,69	0,632	0,65	0,764	0,66
	aln20	0,653	0,65	0,645	0,66	0,645	0,66	0,504	0	0,624	0,58
	aln25	0,592	0,74	0,62	0,78	0,373	0,71	0,668	0,71	0,396	0,74
	aln51	0,294	0,49	0,449	0,38	0,287	0,26	0,359	0,19	0,493	0,31
	Average	0,548	**0,64**	**0,611**	**0,64**	0,466	0,602	0,543	0,418	0,536	0,596
rRNA	aln12	0,923	0,6	0,978	0,7	0,862	0,37	0,808	0,22	0,952	0,68
	aln20	0,994	1,01	0,979	0,86	0,97	0,87	0,979	0,89	0,981	0,89
	aln25	1	1,02	1	1,05	1	1,05	0,982	0,6	1	1,05
	aln34	1	0,91	1	0,92	1	0,92	0,983	0,9	0,983	0,9
	aln50	0,993	0,98	0,993	0,97	0,988	0,95	0,981	0,98	0,993	0,98
	Average	0,982	**0,9**	**0,99**	**0,9**	0,964	0,832	0,947	0,718	0,982	**0,9**
SRP	aln6	0,809	0,65	0,258	0,75	0,222	0,57	0,194	0,36	0,497	0,64
	aln15	0,272	0,77	0,227	0,85	0,189	0,71	0,184	0,39	0,097	0,82
	aln34	0,274	0,73	0,436	0,77	0,391	0,6	0,312	0,26	0,427	0,68
	aln50	0,812	0,62	0,238	0,55	0,213	0,45	0,194	0,32	0,494	0,55
	aln61	0,366	0,94	0,299	0,55	0,288	1,01	0,272	0,87	0,293	0,97

	Average	**0,507**	**0,742**	0,292	0,69	0,261	0,668	0,231	0,44	0,362	0,732
tRNA	aln1	0,235	0,77	0,505	0,33	0,237	0	0,325	0	0,641	0,62
	aln13	0,656	1,16	0,814	0,86	0,913	0,8	0,748	0,72	0,971	1,16
	aln34	0,454	1,09	0,994	1,09	0,994	1,09	0,994	1,09	0,994	1,09
	aln50	1	1,08	1	1,08	1	1,08	0,942	0,64	1	1,08
	aln60	0,454	1,16	0,975	1,16	0,853	0,79	0,933	0,96	0,967	1,16
	Average	0,56	**1,05**	0,858	0,9	0,799	0,752	0,788	0,682	**0,915**	1,022
U5	aln30	0,186	0,44	0,639	0,26	0,612	0,25	0,66	0,26	0,696	0,52
	aln70	0,434	0,66	0,594	0,77	0,952	0,64	0,885	0,44	0,986	0,75
	aln80	0,965	0,88	0,984	0,85	0,984	0,85	0,954	0,68	0,984	0,85
	aln90	0,714	0,66	0,939	0,71	0,939	0,7	0,82	0,45	0,946	0,7
	aln100	0,53	0,65	0,81	0,65	0,829	0,64	0,829	0,59	0,834	0,62
	Average	0,566	0,66	0,793	0,65	0,863	0,616	0,83	0,484	**0,889**	**0,688**

Table 4.11. Score de SPS et SCI des méthodes, les meilleures moyennes sont en gras.

Les résultats SCI de notre programme sont aussi bons ou meilleurs que ceux obtenus par d'autres méthodes (table 4.12, figure 4.24). Le test de Friedman (a=0.05) indique que notre méthode est meilleure que MAFFT dans la prédiction des structures ARN (figure 4.25). Cependant, la différence entre les résultats des méthodes: QUANTALIGN, DIALIGN, CLUSTAL et PROALIGN n'est pas significatives (figure 4.25).

	QUANTALIGN	CLUSTAL	DIALIGN	MAFFT	PROALIGN
min	0.44	0.26	0	0	0.31
max	1.16	1.16	1.09	1.09	1.16
mean	0.80	0.76	0.69	0.55	0.79
median	0.74	0.77	0.70	0.59	0.74
Std dev	0.21	0.23	0.27	0.30	0.22

Table 4.12. Statistique descriptive des résultats de SCI scores.

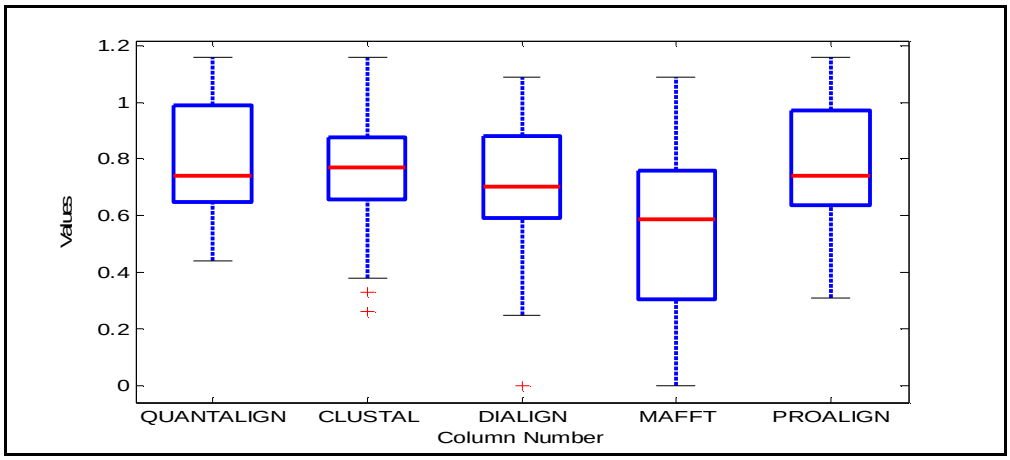


Figure 4.24. Boxplot des programmes (SCI).

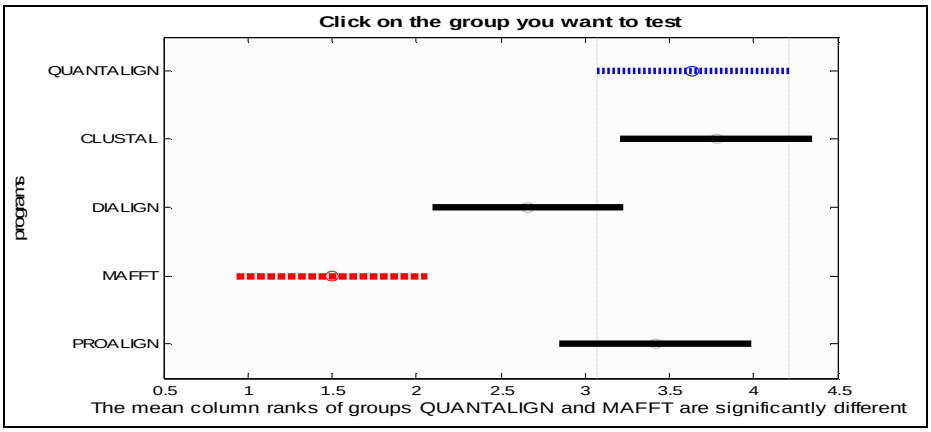


Figure 4.25. Test de Friedman (0.05) pour SCI score.
MAFFT (en dash) est moins bon que QUANTALIGN (en dot) dans ce test.

L'évaluation statistique de l'intégralité des résultats SPS des méthodes avec la référence BRALIBASE (SPS=1) montre qu'aucune méthode n'a réussi dans le test de Friedman à un seuil a=0.01. Pour cela on ôte la référence BRALIBASE du test. Dans ce cas, le test de Friedman est appliqué seulement sur les résultats des programmes pour déterminer les méthodes qui ont de bons résultats par rapport à d'autres méthodes. Les figures 4.26 et 4.27 démontrent que la différence entre les résultats de QUANTALIGN et ceux des autres programmes, n'est pas significative. Cependant, la méthode MAFFT est moins bonne que PROALIGN dans ce test (figure 4.27).

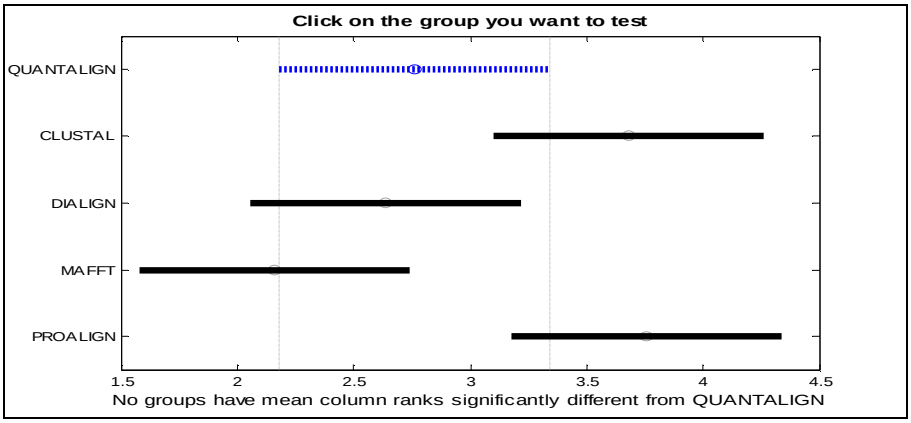


Figure 4.26. Test de Friedman (0.05) pour SPS score. QUANTALIGN (en dot) contre les autres programmes.

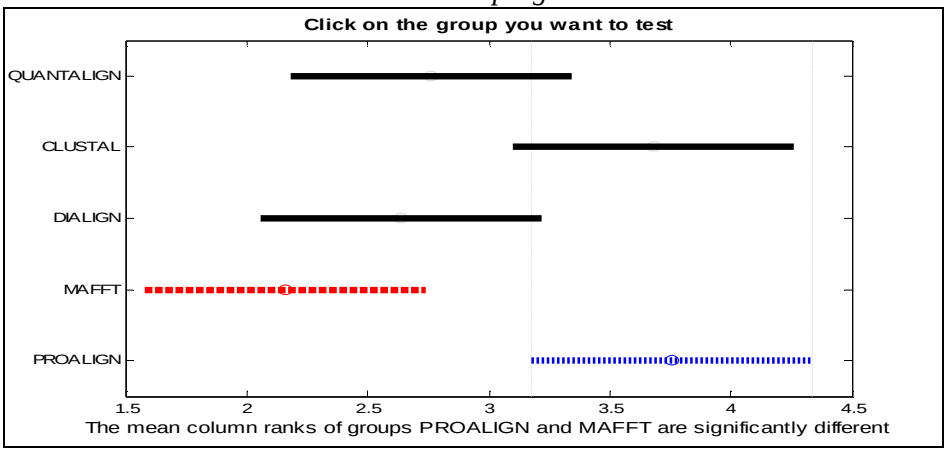


Figure 4.27. Test de Friedman (0.05) pour SPS score. PROALIGN (en dot) contre les autres programmes. MAFFT (en dash) est moins bon.

L'exemple suivant montre la puissance de notre méthode dans le raffinement d'une solution initiale. La figure 4.28 montre que le SCI de la solution initiale est égale à 0.79. Après l'application de QUANTALIGN, le SCI a été amélioré à 1.09 (figure 4.29). L'évaluation des résultats est faite par le programme RNAz de BRALIBASE.

```
############################  RNAz 0.1.1  ############################
 Sequences: 5
 Slice: 1 to 74
 Columns: 74
 Mean pairwise identity:  74.80
 Mean single sequence MFE: -33.76
 Consensus MFE: -26.78
 Energy contribution: -23.90
 Covariance contribution:  -2.88
 Mean z-score:  -2.61
 Structure conservation index (SCI):   0.79
 SVM decision value:   3.13
 SVM RNA-class probability: 0.998533
 Prediction: RNA
########################################################################
>AF070678.1-91_163
GGGCUUGUAGCUCAGCUGGUAGAGCGCCGCCUUUGCAAGGCGGAGGCCCUGGGUCCGAAUCCCAGCAAGUCCA
(((((((..((((........)))).)...(((((((((...)))))))))).(((((.......)))))))))). (-33.50)
>AF083212.1-212_284
GGGCCCAUAGCUCAGUGGUAGAGUGCCUCCUUUGCAAGGAGGAUGCCCUGGGUUCGAAUCCCAGUGGGUCCA
((((((((((((((.((((.....((((((.....))))))..)))))))))).........)))))))). (-35.90)
>X15768.1-211_283
GGGGCCUUAGCUCAGCUGGGAGAGCGCCUGCUUUGCACGCAGGAGGUCAGCGGUUCGAUCCCGCUAGGCUCCA
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))). (-32.40)
>X51423.1-1802_1874
GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCAAGGCAGUGGUCAGCGGUUCGAGUCCGCUUACCUCCA
(((((((.............((.(((((((((.....)))))))))..))(((((.......)))))))))). (-34.70)
>AB003409.1-96_167
GGGGCCAUAGCUCAGCUGGGAGAGCGCCUGCUUUGCACGCAGGAGGUCAGGAGUUCGAUCCUCCUUGGCUCCA
(((((((..((((........))))))(.(((...))).))(((((((((........)).)))))))))). (-32.30)
>consensus
GGGGCCAUAGCUCAGCUGGUAGAGCGCCUCCUUUGCAAGGAGGAGGUCAGGGGUUCGAAUCCCAGUAGG_UCCA
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))). (-26.78 = -
23.90 + -2.88)
```

Figure 4.28. L'évaluation de la solution initiale.

```
############################  RNAz 0.1.1  ############################
 Sequences: 5
 Slice: 1 to 73
 Columns: 73
 Mean pairwise identity:  74.93
 Mean single sequence MFE: -33.76
 Consensus MFE: -36.90
 Energy contribution: -31.62
 Covariance contribution:  -5.28
 Mean z-score:  -2.61
 Structure conservation index (SCI):   1.09
 SVM decision value:   2.56
 SVM RNA-class probability: 0.995308
 Prediction: RNA
########################################################################
>AF070678.1-91_163
GGGCUUGUAGCUCAGCUGGUAGAGCGCCGCCUUUGCAAGGCGGAGGCCCUGGGUCCGAAUCCCAGCAAGUCCA
(((((((..((((........)))).)...(((((((((...)))))))))).(((((.......)))))))))). (-33.50)
>AF083212.1-212_284
GGGCCCAUAGCUCAGUGGUAGAGUGCCUCCUUUGCAAGGAGGAUGCCCUGGGUUCGAAUCCCAGUGGGUCCA
((((((((((((((.((((.....((((((.....))))))..)))))))))).........)))))))). (-35.90)
>X15768.1-211_283
GGGGCCUUAGCUCAGCUGGGAGAGCGCCUGCUUUGCACGCAGGAGGUCAGCGGUUCGAUCCCGCUAGGCUCCA
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))). (-32.40)
>X51423.1-1802_1874
GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCAAGGCAGUGGUCAGCGGUUCGAGUCCGCUUACCUCCA
(((((((.............((.(((((((((.....)))))))))..))(((((.......)))))))))). (-34.70)
>AB003409.1-96_167
GGGGCCAUAGCUCAGCUGGGAGAGCGCCUGCUUUGCACGCAGGAGGUCAGGAGUUCGAUCCUCCUUGGCUCCA
(((((((..((((........))))))(.(((...))).))(((((((((........)).)))))))))).(-32.30)
>consensus
GGGGCCAUAGCUCAGCUGGUAGAGCGCCUCCUUUGCAAGGAGGAGGUCAGGGGUUCGAAUCCCCUUGGCUCCA
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))). (-36.90 = -
31.62 + -5.28)
```

Figure 4.29. L'évaluation de la solution obtenue après l'application de notre programme.

4.6 L'évaluation de QEAMSA avec la fonction WSP et une fonction affine de gaps

Afin d'optimiser QEMSA on a introduit la fonction WSP pour évaluer la fitness des solutions obtenues et une fonction affine pour pénaliser les gaps. Nous avons également introduis la mutation décrite dans l'approche QUANTALIGN. Nous avons testé l'efficacité de QEAMSA face à des familles de séquences prises de la base BALiBASE. Pour des petits ensembles de 4 à 5 séquences de taille entre 49 et 486, QEAMSA donne des solutions aussi bonnes que celles obtenues par QUANTALIGN (table 4.13 et figure 4.30). Cependant, les larges ensemble de séquences ou les ensembles de séquences difficiles à aligner nécessitent un temps CPU énorme pour donner de bons alignements (table 4.14). En effet, QEAMSA nécessite un grand nombre d'itérations par rapport à QUATALIGN afin de converger vers la bonne solution (table 4.14). Donc, QEAMSA est moins recommandé que QUATALIGN dans les familles de séquences de grandes tailles ou difficiles à aligner.

Data set	**La meilleure solution**		**La solution Initial**		**Le Pourcentage d'amélioration**
	SPS	CS	SPS	CS	SPS %
1aboa_ref1	0.574	0.303	0.077	0.000	645
1aab_ref1	1.000	1.000	0.000	0.000	-
glg_ref1	0.947	0.894	0.075	0.000	1162
1ad2_ref1	0.923	0.859	0.026	0.000	3450
1ubi_ref2	0.688	0.000	0.205	0.000	335
2abk_ref4	0.400	0.000	0.067	0.000	597

Table 4.13. Les résultats de QEAMSA en utilisant la fonction WSP et la fonction affine pour gaps.

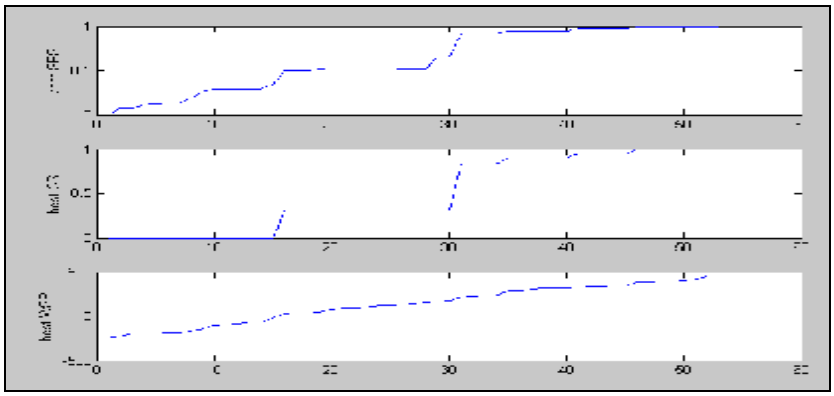


Figure 4.30. La progression des courbes de scores: SPS, CS (BALiBASE) et la fonction objectif WSPS pour le test 1aab_ref1.

Méthode	**QUANTALIGN (3000 itérations)**		**QEAMSA (5000 itérations)**	
Score (Balibase)	SPS	CS	SPS	CS
1idy_ref1	0.448	0.169	0.241	0.077
1tgxa_ref1	0.748	0.640	0.640	0.467
1hava_ref1	0.342	0.000	0.054	0.000

Kinase_ref1	0.719	0.457	0.464	0.235
2trx_ref1	0.558	0.382	0.444	0.195
1gdoa_ref1	0.859	0.734	0.499	0.294

Table 4.14. Comparaison entre QUANTALIGN et QEAMSA en utilisant des tests contenant des familles de séquences difficiles à aligner.

4.7 La comparaison entre notre algorithme évolutionnaire quantique et celui de Han et kim

Notre approche est généralement basé sur le modèle standard d'algorithme évolutionnaire quantique proposé initialement par Han et Kim [Han et al, 01]. Cependant, on a apporté plusieurs améliorations à cet algorithme pour le rendre plus adapté, rapide et efficace pour le problème d'alignement multiple de séquences. Dans l'expérience du knapsack, Han et Kim ont utilisé seulement l'interférence et la mesure quantique pour créer la diversité. Cependant, se limiter seulement à ces deux opérations dans le cas de l'alignement multiple est dérisoire. Nous avons utilisé plusieurs types de mutations bien définies afin d'explorer d'autres solutions et bien guider la recherche. Un apport très important qui a permet de réduire efficacement le temps de convergence est la possibilité de faire un retours en arrière vers la matrice quantique précédente dans le cas d'une mauvaise solution. Vu que les opérations quantiques sont réversibles alors on peut annuler un changement effectué par une mutation qui a donné une mauvaise solution. Cet apport a l'avantage d'accélérer d'interférence vers la meilleure solution courante. Par exemple, supposant que les scores de toutes les solutions sont représentés par la courbe de la figure 4.31 et la meilleure solution courante est B. Apres la mutation de la matrice quantique QM(t) on aura une nouvelle matrice quantique QM(t+1). La mesure de cette dernière et l'évaluation de la solution correspondante donnent une mauvaise solution par rapport à celle obtenue par QM(t) (figure 4.31). Cependant, dans chaque étape on doit interférer vers la bonne solution courante (B). On aura un perte de temps considérable pour passer de la configuration QM(t+1) a la matrice quantique QM(B) correspondante à la meilleure solution courante B. Pour cela, on propose d'annuler le changement fait sur la matrice QM est revenir donc à la configuration QM(t). Ce rebroussement de chemin va réduit considérablement le temps de convergence vers la meilleure solution et évite de faire et répéter vainement des recherches non significatives. On maintien donc la recherche dans le voisinage de la solution courante.

Pour démonter la robustesse de cette approche, nous avons implémenté l'approche utilisée par Han et Kim dans leur expérience du knapsack [Han et al, 01], pour le problème d'alignement multiple. Les résultats obtenus par cette méthode sont comparés à ceux produits par notre approche. Dans cette expérience on a utilisé une solution non alignée pour voir l'efficacité de chaque méthode. En plus, on a utilisé les mêmes paramètres de l'algorithme évolutionnaire quantique et les paramètres d'alignement multiple pour les deux méthodes. Dans cette expérience on a fixé le nombre d'itérations à 5000. Pour chaque test on a pris la moyenne de cinq exécutions consécutives.

Les résultats obtenus au cours de cette expérience (table 4.15) montrent clairement que notre méthode est largement supérieure à celle proposée par Han et Kim dans le problème du

knapsack. Dans les deux derniers tests contenant des séquences de taille supérieure à 260 avec une moyenne d'identité entre les séquences est inférieure à 32%, l'approche de Han et Kim n'a pu améliorer la solution initiale d'un cran (table 4.15).

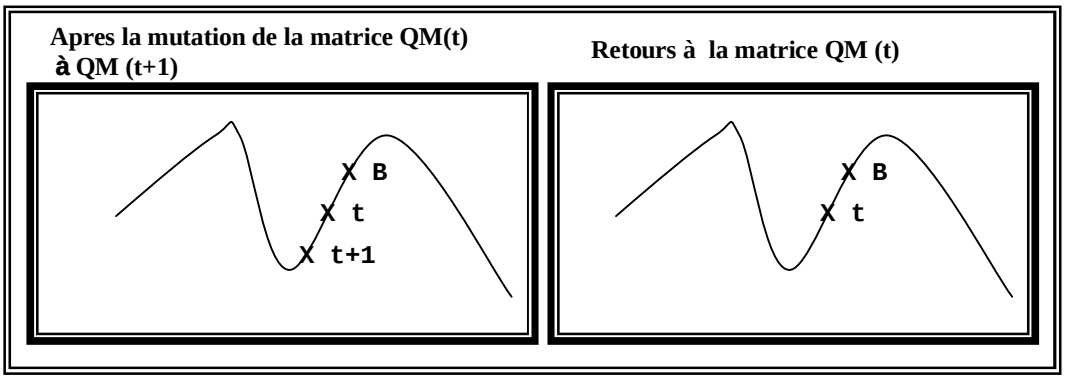


Figure 4.31. L'avantage de rebroussement de chemin dans le cas d'une mauvaise solution.

	Solution initiale		Notre approche		L'approche de HAN	
Score BALiBASE	SPS	CS	SPS	CS	SPS	CS
2trx_ref1	0.193	0.000	**0.449**	**0,245**	0,221	0,000
1aab_ref1	0.000	0.000	**0.630**	**0.419**	0,222	0,000
1idy_ref1	0.038	0.000	**0.258**	**0,054**	0,154	0,000
1ar5a_ref1	0.061	0.000	**0,615**	**0,380**	0,326	0,084
Kinase_ref1	0.240	0.139	**0,341**	**0,139**	0,240	0,139
Glg_ref1	0.075	0.000	**0,299**	**0,064**	0,075	0,000

Table 4.15. La comparaison entre notre approche et l'approche de Han et Kim pour le MSA.

4.8 L'effet de l'augmentation de la taille de la population

Au cours de toutes les expériences précédentes, la population de chromosomes de l'algorithme évolutionnaire quantique contient seulement un seul chromosome quantique. La question qui se pose : l'augmentation du nombre de chromosomes quantiques va-t-elle diminuer le nombre d'itérations nécessaire pour avoir un bon alignement? Pour répondre à cette question, on a effectuer l'expérience suivante: on a comparé les résultats obtenus en utilisant un seul chromosome avec ceux obtenus par l'utilisation de quatre chromosomes. Dans cette expérience on a fixé le nombre d'itérations à 2000. Pour chaque test on a pris la moyenne de cinq exécutions consécutives. L'expérience à montré (table 4.16) que les résultats trouvés par l'approche utilisant 4 chromosomes quantiques est nettement meilleurs que ceux obtenus par un seul chromosome quantiques. Cette expérience prouve que le temps d'exécution de notre approche peut être considérablement réduit en utilisant une implémentation parallèle de notre approche.

Taille de la population	**4 chromosomes**		**Un seul chromosome**	
Score BALiBASE	SPS	CS	SPS	CS
2trx_ref1	**0,388**	**0,127**	0,360	0,094
1aab_ref1	**0,604**	**0,319**	0,392	0,175
1idy_ref1	**0,274**	**0,077**	0,192	0.000
1ar5a_ref1	**0,719**	**0,541**	0,640	0,372
Kinase_ref1	**0,377**	**0,185**	0,352	0,158

Table 4.16. L'effet de la taille de la population sur la précision de la solution (La moyenne de cinq exécutions consécutives pour 2000 itérations).

4.9 Le rôle de l'interférence dans le processus d'optimisation par AQE

Dans cette expérience, on démontre l'impact de l'opération d'interférence dans le processus d'optimisation par les algorithmes quantiques évolutionnaires. La table 4.17 suivante montre les résultats obtenus en utilisant QEAMSA avec interférence et sans interférence.

	Solution initiale		**Avec interférence**		**Sans interférence**	
Score BALiBASE	SPS	CS	SPS	CS	SPS	CS
2trx_ref1	0.193	0.000	0.444	0.195	0,193	0,000
1hava_ref1	0.033	0.000	0.054	0.000	0.033	0.000
1idy_ref1	0.038	0.000	0.241	0.077	0,038	0,000
1gdoa_ref1	0.189	0.000	0.499	0.294	0.220	0.110
Kinase_ref1	0.240	0.067	0.464	0.235	0,240	0,139

Table 4.17. Le rôle de l'interférence dans le processus d'optimisation.

4.10 La complexité de notre approche

Pour aligner un ensemble de *m* séquence dont la taille de la plus grande séquence est *n*, la complexité de notre programme est dépendante de plusieurs facteurs. Les étapes qui consomment considérablement le temps CPU sont les suivantes:

1. La complexité pour calculer une solution initiale est égale à $m*(m-1)/2*n^2$ où n^2 est la complexité de l'algorithme de Needleman pour l'alignement de paires de séquences et $m*(m-1)/2$ et le nombre de toutes les paires possibles de séquences.
2. La complexité de l'évaluation de la fonction objectif WSPS est $m*(m-1)/2*n$.
3. La complexité des opérations d'interférences, de mesure et de traduction est approximativement égale à $m*n$.
4. La complexité d'une boucle de k itérations est égale à la somme des complexités de (2) et (3) multipliée par k : $k*(m*(m-1)/2*n+m*n)$.

Donc la complexité totale est approximativement égale à la somme des complexités précédentes :

$$complexité = k * m^2 * n + k * m * n + m^2 * n^2 \qquad (4.2)$$

Où m : nombre de séquences, n : la taille de la plus grande séquence, et *k:* le nombre d'itérations de l'algorithme quantique évolutionnaire.

Donc le temps CPU de nos résultats est dépendant de la taille des séquences, le nombre de séquences et le nombre d'itérations. Ce dernier est très dépendant de la dureté de l'ensemble de séquences à aligner. En effet on peut aligner un ensemble de 5 séquences en 2 minute parce qu'il contient des séquences faciles à aligner or on a besoin de 10 minutes ou plus pour aligner un autre ensemble de 5 séquences vu la dureté de ses séquences. Concernant la complexité spatiale des approches développées, QUANTALIGN et QEAMSA ne sont pas gourmande en matière de ressource mémoire. Tous les résultats précédents sont obtenus sur une machine avec 128 MO de mémoire. Dans toutes les expériences, même avec les larges tests comme les tests Kinase2_ref4 (17 séquences, taille de l'alignement >1000 et une moyenne d'identité de 20%), 1Pama_ref3 (19 séquences, taille de l'alignement >600 et une moyenne d'identité de 29%), le problème d'espace mémoire ne s'est pas posé. En effet, MATLAB nécessite simplement moins de 64 MO pour qu'il fonctionne correctement pour ce problème.

4.11 Conclusions et perspectives

Pour résoudre le problème d'alignement multiple de séquences, on a proposé deux approches basées AQE. La première est une méthode purement itérative nommée QEAMSA. Cependant, cette approche est lente parce qu'elle commence par une solution complètement non alignée. Ce problème a été surmonté en introduisant la méthode QUANTALIGN. Les expériences montrent clairement que QUANTALIGN peut donner un alignement global aussi bon ou meilleur que les méthodes existantes. Mais, le temps reste encore élevé par rapport aux méthodes progressives. Ce problème peut être surmonté en implémentant nos approches sur des machines parallèles. En effet, les AQEs ont démontré leur efficacité dans les architectures parallèles. Une deuxième idée pour accélérer QEAMSA et QUANTALIGN est l'intensification du processus de recherche dans les régions pauvrement alignées dans un alignement (figure 4.32). Finalement, on peut diviser l'alignement en deux blocs et appliquer notre programme sur chaque bloc.

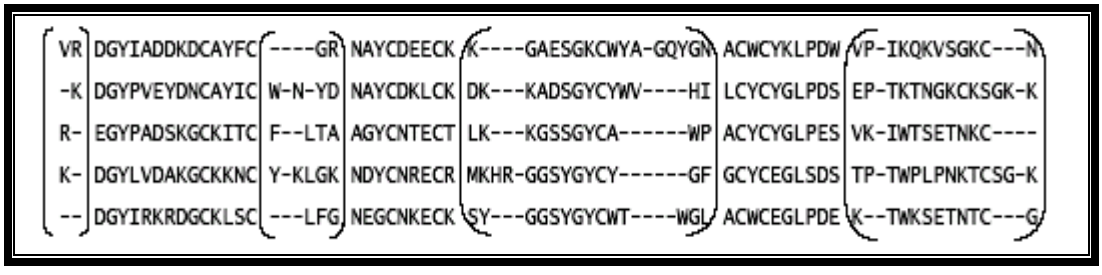


Figure 4.32. Une solution d'élite isolant des régions bien alignées des régions mal alignées. Les parties des séquences à l'intérieur des parenthèses indiquent les régions attrayantes qui sont mal alignées.

Conclusion générale

Au cours de ce travail de magistère, on a proposé de nouvelles approches itératives pour résoudre le célèbre problème de la bioinformatique: l'alignement multiple de séquences, en exploitant d'autres domaines d'optimisation récents. Les deux approches développées sont basées sur un noyau quantique évolutionnaire développé spécifiquement pour le problème d'alignement multiple de séquences. En effet, nous avons opté une représentation quantique pour coder les solutions possibles d'un MSA. L'avantage de cette représentation est la possibilité de représenter tous les alignements possibles en utilisant seulement un seul chromosome. Ce qui réduit donc la taille de la population de chromosomes utilisée par rapport aux algorithmes évolutionnaires classiques (SAGA utilise une population de taille 100). L'autre caractéristique des approches développées est l'utilisation d'une dynamique évolutionnaire quantique permettant une stratégie efficace pour la recherche de la bonne solution. En effet, cette dynamique permet un meilleur équilibre entre la capacité d'exploration et d'exploitation de l'espace de recherche afin d'avoir des solutions de bonne qualité. Cependant, les deux approches se différent dans plusieurs détailles. Dans un premier temps, on a proposé une approche purement itérative nommée "QEAMSA" [Meshoul et al, 05a]. Le processus d'optimisation démarre avec une solution complètement non alignée (aléatoire). La mutation dans cette approche est simple. Elle consiste à permuter aléatoirement quelques qubits du chromosome quantique en utilisant l'opération quantique "SWAP". Les résultats obtenus par cette approche sont meilleurs que ceux obtenus par la méthode CLUSTAL X [Meshoul et al, 05a]. Néanmoins, cette approche nécessite beaucoup de temps pour converger vers la bonne solution. Une amélioration de cette approche a conduit à la proposition d'une autre approche nommé "QUANTALIGN" afin de surmonter les problèmes de la méthode précédente. D'abord, on a remarqué que la solution initiale dans les méthodes itératives joue un grand rôle dans le processus d'optimisation. Le choix d'une bonne solution initiale va accélérer considérablement la vitesse de convergence vers la solution optimale [Meshoul et al, 05b]. Le deuxième grand apport de la deuxième approche réside dans l'opération de mutation. En effet, contrairement aux autres problèmes d'optimisation, la mutation dans le cas de l'alignement multiple doit être opéré d'une manière "intelligente". On a remarqué que la mutation simple utilisée dans la première approche gaspille vainement le temps CPU. On a donc utilisé des mutations qui opèrent sur des suites ou des blocs de qubits [Meshoul et al, 05b]. L'avantage de ce type de mutation est qu'elle affecte un grand nombre de séquences ce qui réduit efficacement le temps CPU. Troisièmement, la méthode QUANTALIGN est un outil puissant et flexible permettant l'optimisation de n'importe qu'elle fonction objective. En effet, Nous avons testé cette approche avec trois fonctions objectives différentes et les résultats étaient très satisfaisants. Finalement, on a utilisé une fonction affine

pour pénaliser les gaps parce qu'elle est biologiquement plus significative que la fonction de pénalité constante utilisée dans l'approche QEAMSA. Pour l'évaluation biologique des résultats obtenus par QUANTALIGN, nous avons testé notre approche avec la base d'alignements de références BAliBASE [Thompson et al, 99a] qui contient des alignements de séquences protéiques. Ces alignements sont conçus à la main et du point de vue biologique sont valides. Ce test est le plus utilisé dans l'alignement multiple pour déterminer la robustesse et l'efficacité d'une méthode. En plus, nos résultats d'alignement sont ensuite comparés à ceux d'autres programmes d'alignement multiple. Finalement, on a testé également notre méthode avec des ensembles de séquences ARN pris de la base BRALIBASE [Gardner et al, 05]. L'utilité de ce test est de déterminer la capacité de notre la méthode QUANTALIGN dans la prédiction des structures ARN. Nos résultats montre que QUANTALIGN peut donner un alignement global aussi bon ou meilleur que les méthodes existantes.

Comme les biologistes veulent des solutions optimales dans des délais raisonnables, la deuxième approche QUANTALIGN est amplement recommandée parce qu'elle offre de bons résultats dans des délais raisonnables.

Comme perspectives, on envisage d'utiliser d'utiliser le noyau développé avec un modèle HMM pour l'alignement multiple de séquences. On envisage également d'utiliser les AQEs dans *l'alignement structurel de séquences*. Par ailleurs, les AQEs pourraient être très utiles dans d'autres problèmes bioinformatiques tel que la construction des arbres phylogénétiques. En effet, ce problème est encore non résolu. Il est caractérisé par une grande complexité spatiotemporelle. Un autre problème très compliqué est l'assemblage de fragments d'ADN. Ce problème consiste à reconstituer une "super séquence" à partir d'un ensemble de séquences. L'assemblage de fragments d'ADN constitue la base des grands projets de séquençage du génome humain. Cependant, trouver une super séquence optimale (la plus petite super séquence qui englobe tous les fragments d'ADN) est une tâche NP-difficile.

Reference 1 - Small number of colinear sequences - Short, V1

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1ABOA	0,56	0,687	0,529	0,359	0,312	0,312	0,703	0,521	0,526	0,181	0,762
1IDY	0,606	0,705	0,342	0,018	0,145	0,062	0,566	0,08	0,08	0,138	0,844
1R69	0,837	0,481	0,55	0,406	0,681	0,366	0,325	0,562	0,225	0,1	0,7
1TVXA	0,378	0,438	0,278	0,306	0,344	0,344	0,228	0,344	0,244	0,108	0,5
1UBI	0,498	0,415	0,452	0	0,37	0,493	0,488	0,428	0,428	0,14	0,78
1WIT	0,991	0,982	0,899	0,851	0,963	0,444	0,842	0,773	0,763	0,549	0,951
2TRX	0,494	0,754	0,801	0,728	0,451	0,496	0,5	0,453	0,235	0,292	0,73

Reference 1 - Small number of colinear sequences - Medium, V1

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1bbt3	0,907	0,706	0,652	0,45	0,26	0,26	0,582	0,43	0,16	0,128	0,814
1sbp	0,613	0,674	0,543	0,453	0,54	0,456	0,58	0,547	0,537	0,144	0,617
1havA	0,656	0,446	0,411	0,13	0,3	0,466	0,419	0,536	0,04	0,133	0,582
1uky	0,676	0,724	0,672	0,566	0,684	0,684	0,685	0,571	0,46	0,084	0,693
2hsdA	0,885	0,691	0,771	0,679	0,47	0,47	0,47	0,646	0,463	0,117	0,821
2pia	0,78	0,89	0,69	0,66	0,74	0,74	0,76	0,85	0,56	0,057	0,87
3grs	0,447	0,635	0,689	0,327	0,416	0,416	0,38	0,446	0,347	0,056	0,677
kinase	0,891	0,736	0,862	0,764	0,733	0,703	0,643	0,73	0,65	0,277	0,839

Reference 1 - Small number of colinear sequences - Long, V1

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1ajsA	0,636	0,571	0,531	0,142	0,486	0,471	0,424	0,556	0,364	0,105	0,635
1cpt	0,862	0,827	0,78	0,829	0,792	0,792	0,835	0,865	0,777	0,156	0,838
1lvl	0,694	0,632	0,619	0,699	0,559	0,559	0,57	0,587	0,572	0,064	0,603
1pamA	0,724	0,582	0,596	0,694	0,513	0,549	0,596	0,641	0,204	0,034	0,705
1ped	0,846	0,812	0,748	0,743	0,756	0,756	0,727	0,723	0,73	0,032	0,813
2myr	0,506	0,381	0,285	0,284	0,67	0,613	0,422	0,621	0,547	0,05	0,532
4enl	0,677	0,706	0,414	0,529	0,701	0,701	0,754	0,669	0,787	0,044	0,782
gal4	0,742	0,407	0,5	0,581	0,445	0,445	0,368	0,543	0,406	0,055	0,713

Reference 1 - Small number of colinear sequences - Short, V2

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1aab	1	1	0,823	1	1	1	1	1	1	0,214	1
1fjla	1	1	0,993	1	1	1	1	1	1	0,436	1
1hfh	0,933	0,917	0,945	0,41	0,868	0,932	0,936	0,848	0,883	0,261	0,933
1hpi	0,918	0,861	0,916	0,785	0,909	0,909	0,89	0,859	0,852	0,962	0,957
1csy	0,949	1	0,969	0,98	0,976	0,968	0,981	0,932	0,935	0,779	0,975
1pfc	0,975	0,988	0,994	0,894	0,927	0,927	0,975	0,964	0,861	0,35	0,986
1tgxA	0,895	0,806	0,873	0,871	0,782	0,768	0,819	0,718	0,651	0,556	0,823
1ycc	0,856	0,935	0,837	0,749	0,815	0,815	0,932	0,939	0,94	0,217	0,858
3cyr	0,907	0,805	0,908	0,75	0,887	0,887	0,858	0,854	0,841	0,454	0,93
451c	0,681	0,719	0,662	0,729	0,541	0,552	0,683	0,668	0,713	0,346	0,717

Reference 1 - Small number of colinear sequences - Medium, V2

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1ad2	0,943	0,984	0,917	0,96	0,934	0,934	0,96	0,96	0,975	0,341	0,985
1aym3	0,976	0,969	0,955	0,962	0,976	0,958	0,969	0,952	0,969	0,567	0,966
1gdoA	0,926	0,929	0,907	0,81	0,911	0,911	0,935	0,882	0,84	0,604	0,927
1ldg	1	0,963	0,989	0,966	0,996	0,996	0,989	0,958	0,956	0,326	0,998

1mrj	1	1	0,949	0,977	1	1	0,994	1	1	0,561	0,994
1pgtA	0,969	1	0,982	0,996	0,993	1	0,937	0,975	1	0,545	0,965
1pii	0,883	0,864	0,896	0,89	0,832	0,788	0,884	0,854	0,859	0,193	0,888
1ton	0,965	0,935	0,945	0,867	0,904	0,844	0,899	0,826	0,749	0,512	0,928
2cba	0,946	0,926	0,946	0,941	0,928	0,935	0,855	0,846	0,859	0,176	0,913

Reference 1 - Small number of colinear sequences - Long, V2

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1ac5	0,94	0,992	0,911	0,926	0,941	0,941	0,942	0,898	0,896	0,236	0,911
1bgl	0,979	0,959	0,972	0,959	0,934	0,929	0,959	0,958	0,977	0,365	0,962
1dlc	0,977	0,961	0,916	0,897	0,952	0,952	0,98	0,984	0,926	0,427	0,933
1eft	0,934	0,911	0,91	0,925	0,926	0,926	0,929	0,913	0,895	0,289	0,933
1fieA	0,965	0,944	0,947	0,979	0,905	0,926	0,958	0,958	0,939	0,637	0,968
1gowA	0,749	0,898	0,814	0,902	0,872	0,872	0,878	0,921	0,927	0,396	0,907
1pkm	0,942	0,921	0,955	0,927	0,907	0,911	0,946	0,931	0,851	0,748	0,932
1sesA	0,985	0,968	0,954	0,968	0,899	0,899	0,973	0,963	0,882	0,642	0,981
2ack	0,909	0,907	0,904	0,882	0,907	0,88	0,848	0,87	0,766	0,524	0,866
arp	0,956	0,945	0,933	0,93	0,914	0,927	0,938	0,925	0,927	0,293	0,939
glg	0,982	0,941	0,972	0,959	0,978	0,968	0,954	0,977	0,943	0,689	0,978

Reference 1 - Small number of colinear sequences - Short, V3

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1aho	0,99	0,971	1	1	1	1	0,913	0,938	0,938	0,789	1
1csp	0,943	0,993	0,993	0,98	1	1	0,987	1	0,625	0,776	0,993
1dox	0,887	0,919	0,879	0,859	0,868	0,868	0,799	0,812	0,48	0,806	0,866
1fkj	0,982	0,981	0,981	0,958	0,944	0,987	0,951	0,913	0,609	0,879	0,982
1fmb	0,959	0,981	0,979	0,959	0,952	0,952	0,995	0,995	0,556	0,863	0,984
1krn	1	1	0,993	1	0,986	0,986	0,993	1	1	0,944	1
1plc	0,979	0,934	0,958	0,931	0,904	0,875	0,964	0,958	0,946	0,797	0,968
2fxb	0,945	0,945	0,951	0,945	0,945	0,945	0,945	0,945	0,945	0,93	0,943
2mhr	0,98	0,985	0,952	0,951	0,975	0,908	0,962	0,965	0,961	0,803	0,982
9rnt	0,965	0,974	0,965	0,864	0,97	0,961	0,965	0,97	0,974	0,832	0,957

Reference 1 - Small numberof colinear sequences - Medium, V3

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1amk	0,986	0,989	0,997	0,993	0,987	0,987	0,991	0,993	0,992	0,941	0,996
1ar5A	0,971	0,994	0,971	0,92	0,994	0,963	0,99	0,98	0,98	0,903	**0,995**
1ezm	0,941	0,948	0,932	0,911	0,956	0,956	0,958	0,948	0,61	0,851	0,951
1led	0,969	0,946	0,923	0,516	0,987	0,93	0,94	0,933	0,882	0,761	0,966
1ppn	0,973	0,989	0,983	0,648	0,962	0,979	0,973	0,978	0,973	0,856	0,962
1pysA	0,931	0,922	0,906	0,936	0,935	0,935	0,926	0,931	0,917	0,616	0,929
1thm	0,956	0,961	0,956	0,946	0,971	0,971	0,966	0,961	0,936	0,697	0,963
1tis	0,971	0,977	0,953	0,971	0,951	0,965	0,942	0,985	0,971	0,846	0,956
1zin	0,988	0,977	0,977	0,966	0,916	0,916	0,954	0,961	0,927	0,894	0,981
5ptp	0,977	0,966	0,94	0,888	0,966	0,96	0,954	0,972	0,953	0,742	0,969

Reference 1 - Small number of colinear sequences - Long, V3

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	MULTAL	HMMT	QUANTALIGN
1ad3	0,972	0,968	0,967	0,963	0,962	0,962	0,979	0,97	0,971	0,738	0,98
1gpb	0,988	0,986	0,982	0,958	0,964	0,964	0,987	0,983	0,964	0,65	0,982
1gtr	0,992	0,986	0,995	0,994	0,961	0,961	0,976	0,995	0,99	0,796	0,989

1lcf	0,98	0,981	0,967	0,947	0,962	0,964	0,979	0,971	0,968	0,854	0,978
1rthA	0,961	0,977	0,96	0,958	0,962	0,952	0,966	0,982	0,956	0,856	0,959
1taq	0,963	0,963	0,931	0,889	0,961	0,934	0,944	0,942	0,88	0,691	0,953
3pmg	0,987	0,995	0,989	0,985	0,985	0,985	0,994	0,994	0,993	0,844	0,991
actin	0,979	0,965	0,965	0,968	0,968	0,98	0,975	0,969	0,962	0,788	0,97
MOY Ref1	**0,877**	**0,864**	**0,841**	**0,788**	**0,821**	**0,810**	**0,834**	**0,832**	**0,763**	**0,487**	**0,889**

Reference 2 - 1 Orphan

	PRRP	CLUSTALX	SAGA	DIALIGN	HMMT	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
1aboA	0,256	0,65	0,489	0,384	0,724	0,391	0,22	0,528	0	0,811
1idy	0,37	0,515	0,548	0	0,353	0	0	0,401	0	0,2
1csy	0,35	0,154	0,154	0	0	0	0	0,154	0,114	0,125
1r69	0,675	0,675	0,475	0,675	0	0,675	0,675	0,675	0,45	0,739
1tvxA	0,207	0,552	0,448	0	0,276	0,241	0,241	0,138	0,345	0
1tgxA	0,695	0,727	0,773	0,63	0,622	0,678	0,543	0,696	0,318	0,391
1ubi	0,056	0,482	0,492	0	0,053	0,129	0,129	0	0	0,481
1wit	0,76	0,557	0,694	0,724	0,641	0,469	0,463	0,5	0,476	0,679
2trx	0,87	0,87	0,87	0,734	0,739	0,85	0,702	0,87	0,87	0,652
1sbp	0,231	0,217	0,374	0,043	0,214	0,043	0,054	0,186	0,177	0,057
1havA	0,52	0,48	0,448	0	0,194	0,259	0,238	0,5	0,493	0,094
1uky	0,351	0,656	0,476	0,216	0,395	0,256	0,306	0,585	0,562	0,414
2hsdA	0,404	0,484	0,498	0,262	0,423	0,39	0,561	0,593	0,278	0,412
2pia	0,767	0,752	0,763	0,612	0,647	0,73	0,695	0,765	0,766	0,674
3grs	0,363	0,192	0,282	0,35	0,141	0,183	0,211	0,192	0,159	0,394
kinase	0,896	0,848	0,867	0,692	0,749	0,755	0,651	0,83	0,799	0,443
1ajsA	0,227	0,324	0,311	0	0,242	0	0	0,311	0,227	0,337
1cpt	0,821	0,66	0,776	0,425	0,388	0,184	0,277	0,777	0,688	0,715
1lvl	0,772	0,746	0,726	0,783	0,539	0,62	0,688	0,614	0,678	0,338
1pamA	0,711	0,761	0,623	0,576	0,53	0,393	0,386	0,566	0,702	0,691
1ped	0,881	0,834	0,835	0,773	0,696	0,651	0,647	0,741	0,749	0,235
2myr	0,582	0,904	0,825	0,84	0,443	0,727	0,75	0,894	0,786	0,389
4enl	0,668	0,375	0,739	0,122	0,213	0,096	0,092	0,384	0,224	0,511
MOY Ref2	**0,541**	**0,583**	**0,586**	**0,384**	**0,401**	**0,379**	**0,371**	**0,517**	**0,429**	**0,425**

Reference 3 - Sub-groups of sequences

	PRRP	CLUSTALX	SAGA	DIALIGN	HMMT	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
1idy	0	0,273	0,364	0	0,227	0	0	0,045	0	0
1r69	0,905	0,524	0,524	0,524	0	0	0,905	0	0	0,762
1ubi	0,415	0,146	0,585	0	0,366	0	0	0	0,268	0,415
1wit	0,742	0,565	0,484	0,5	0,323	0,645	0,323	0,242	0,21	0,645
1uky	0,139	0,13	0,269	0,139	0,037	0,083	0,148	0,241	0,083	0,185
kinase	0,783	0,72	0,758	0,65	0,478	0,541	0,682	0,688	0,599	0,63
1ajsA	0,128	0,163	0,186	0	0,006	0	0	0	0,11	0,179
1 pamA	0,683	0,678	0,579	0,683	0,169	0,546	0,59	0,546	0,754	0,705
1ped	0,679	0,627	0,646	0,641	0,172	0,45	0,507	0,665	0,722	0,636
2myr	0,646	0,538	0,494	0,272	0,101	0,278	0,494	0,253	0,31	0,424
4enl	0,736	0,547	0,672	0,05	0,05	0,393	0,438	0,652	0,498	0,552
MOY Ref3	**0,532**	**0,446**	**0,506**	**0,314**	**0,175**	**0,267**	**0,372**	**0,303**	**0,323**	**0,467**

Reference 4 - N/C terminal extensions

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
1dynA	0	0	0	0,6	0,6	0,6	0	0	0
1pysA	0	0	0,25	0,75	1	1	0	0,75	0,25
1ckaA	1	0	0,375	1	1	0	0	1	1
1csp	0	0	0	0,889	0	0	0	0	0,889
1lkl	1	1	0	1	1	1	0	1	1
1mfa	0,385	1	0,385	1	0,846	1	0,385	1	0,385
1pfc	1	0,969	1	0,719	1	1	1	1	1
1vln	0	0,879	0,606	0,545	0,636	0,576	0,636	0,848	0,545
1ycc	0,485	0,485	0,485	0,727	0,97	0,818	0,485	0,455	0,485
2abk	0	0	0	1	0,471	0,471	0	0,471	1
kinase1	0	0	0	1	1	1	0	1	0,896
kinase2	0	0	0,364	1	1	1	1	1	0,727
MOY Ref 4	0,323	0,361	0,289	0,853	0,794	0,705	0,292	0,710	0,681

Reference 5 - internal insertions

	PRRP	CLUSTALX	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	QUANTALIGN
1pysA	0,429	0,429	0,429	0,762	0,19	0,762	0,429	0,19	0,571
1eft	0,211	0	0	0,579	0	0	0	0,211	0,316
1ivy	1	0,735	0,735	1	1	0,882	0,735	1	1
1qpg	1	1	0,521	1	1	1	1	1	1
1thm1	0,765	0,412	0,765	0,765	0,765	0,412	0,412	0,765	0,765
1thm2	0,645	0,774	0,774	1	0,194	0,194	0,774	0,645	0,775
2cba	0,867	0,717	0,767	1	0,533	0,767	0,55	0,6	1
S51	0,662	0,938	0,831	0,646	0,338	0,631	0,646	0,646	0,646
S52	1	1	1	1	0,515	0,515	1	0,515	1
kinase1	1	0,806	0,484	0,806	0,677	0,677	1	0,677	1
kinase2	0,489	1	0,667	0,667	0,556	0,444	0,333	0,689	0,8
kinase3	0,333	0,646	0,729	0,812	0,333	0,583	0,646	0,729	0,813
MOY Ref 5	0,700	0,705	0,642	0,836	0,508	0,572	0,627	0,639	0,807

Annexe 1 Théorie de la complexité

La matière de cette annexe est obtenue à partir du site:
"http://fr.wikipedia.org/wiki/Théorie de la complexité"

1. Introduction

La théorie de la complexité s'intéresse à l'étude formelle de la *difficulté* des problèmes en informatique. La question est de savoir si un problème peut être résolus efficacement ou pas en se basant sur une estimation théorique des temps de calcul et des besoins en mémoire informatique.

En théorie de la complexité, un problème est vu comme un ensemble de données en entrée, et une question sur ces données pouvant demander éventuellement un calcul. La théorie de la complexité traite les problèmes de décision binaire. On étend également la notion de complexité aux problèmes d'optimisation. En effet il est facile de transformer un problème d'optimisation en problème de décision. Si par exemple on cherche à optimiser une valeur *n* on traite le problème de décision qui consiste à comparer *n* à un certain *k*. En traitant plusieurs valeurs de *k* on peut déterminer une valeur optimale. On confondra souvent un problème d'optimisation et son problème de décision associé. La théorie de la complexité classifie les problèmes en fonction de la complexité des algorithmes qui existent pour les résoudre. Parmi les classes les plus courantes, on distingue:

- Classe **L** : un problème de décision qui peut être résolu par un algorithme déterministe en espace *logarithmique* par rapport à la taille de l'instance est dans L.
- Classe **NL** : cette classe correspond à la précédente mais pour un algorithme non-déterministe.
- Classe **P** : un problème de décision est dans P s'il peut être décidé par un algorithme déterministe en un temps *polynomial* par rapport à la taille de l'instance. On qualifie alors le problème de polynomial.
- Classe **NP** : c'est la classe des problèmes de décision pour lesquels la réponse *oui* peut être décidée par un algorithme non-déterministe en un temps polynomial par rapport à la taille de l'instance.
- Classe **Co-NP** : nom parfois donné pour l'équivalent de la classe NP avec la réponse *non*.
- Classe **PSPACE** : les problèmes décidables par un algorithme déterministe en espace polynomial par rapport à la taille de son instance.
- Classe **NSPACE** ou **NPSPACE** : les problèmes décidables par un algorithme non-déterministe en espace polynomial par rapport à la taille de son instance.
- Classe **EXPTIME** : les problèmes décidables par un algorithme déterministe en temps exponentiel par rapport à la taille de son instance.

2. Problème NP-Complet (NP-Diffcile)

Soit **C** une classe de complexité comme les classes décrites précédemment. On dit qu'un problème est **C-complet** si

- il est dans **C**
- il est **C-difficile** (ou **C-dur**)

Un problème est **C-difficile** (ou **C-dur**) si ce problème est plus dur que tous problèmes dans **C**. Formellement on définit une notion de réduction : soient *p* et *q* deux problèmes, *p* se réduit à *q* si *p* est une instance de *q*. Et donc *p* est **C-difficile** (ou **C-dur**) si pour tout problème *q* de **C**, *q* se réduit à *p*.
Les problèmes complets les plus étudiés sont ce la classe **NP**. En effet, la plus part de problèmes intéressants sont NP-complets et que l'on ne sait pas résoudre un problème NP-complet efficacement à cause du non déterminisme. La classe de complexité est théoriquement réservée à des problèmes de décisions. On parlera de problème NP-difficile pour les problèmes d'optimisation sachant que pour ces problèmes d'optimisation on peut construire facilement un problème qui lui est associé et il est dans NP et qui est donc NP-complet. Parmi les problèmes NP-Complets les plus célèbres on distingue le problème du voyageur de commerce, problèmes de colorations de graphes, problème de couverture de sommets dans un graphe, le problème d'alignement multiple de séquences.

Annexe 3: Glossaire: Bioinformatique

(Obtenu à partir de http://www.infobiogen.fr/services/deambulum/fr/bioinfo_glos.html)

A

Acide aminé (= Résidu) (**amino acid**).
Monomère constitutif des protéines. Il en existe 20, codés par un système à trois nucléotides (codons), dans l'ADN et l'ARN.

Acides nucléiques (**Nucleic acids**).
Macromolécules jouant un rôle fondamental dans le stockage, le maintien et le transfert de l'information génétique. Ce sont des polynucléotides (polymères de nucléotides), reliés par des liaisons phosphodiester. Les molécules linéaires présentent une extrémité 5' P (en raison du nucléotide dont une fonction alcool, en position 5' du sucre, et estérifiée par un acide phosphorique) et une extrémité 3' OH. Voir ADN et ARN.

ADN (Acide DésoxyriboNucléique) (**DNA (DesoxyriboNucleic Acid)**).
L'ADN est la forme de stockage de l'information génétique du génome de pratiquement tous les êtres vivants. Cette information est représentée sur le chromosome par une suite linéaire de gènes, séparés par des régions intergéniques. L'**ADN**, macromolécule biologique formée de désoxyribonucléotides, est un des constituants des chromosomes. Elle forme des pelotes microscopiques qui, chez les organismes eucaryotes, sont localisés dans le noyau des cellules. Déroulés, les molécules d'ADN s'étirent en un très long fil, constitué par un enchaînement (séquence) précis d'unités élémentaires que sont les nucléotides. La structure originale de l'ADN, formée de deux brins complémentaires enroulés en hélice (double hélice), lui permet de se dupliquer en deux molécules identiques entre elles et identiques à la molécule mère lors du phénomène de réplication. L'ADN est une séquence d'acide nucléique linéaire ou circulaire, et bicaténaire (la double hélice unit deux brins polynucléotidiques antiparallèles et complémentaires). On distingue :

- l'ADN nucléaire : ADN du noyau
- l'ADN mitochondrial : ADN du génome mitochondrial
- l'ADN chloroplastique : ADN du génome chloroplastique

Alignement (**alignement**).
Processus par lequel deux séquences sont comparées afin d'obtenir le plus de correspondances (identités ou substitutions conservatives) possibles entre les lettres qui les composent.

- Alignement global (**global alignement**) : alignement des deux séquences sur toute leur longueur.
- Alignement local (local alignement) : alignement des deux séquences seulement sur une portion de leur longueur. (Ex: programmes Fasta et Blast)
- Alignement optimal (optimal alignement) : alignement de deux séquences de façon à obtenir le plus haut score possible. (Cf: algorithme Needleman et Wunsch)
- Alignement multiple (multiple alignement) : alignement global de trois ou plus de trois séquences. (Ex: programme Clustalw)

Annotation (**annotation**).
L'annotation du génome consiste à prédire et localiser l'ensemble des séquences codantes (gènes) du génome et à déterminer et identifier leur structure (annotation syntaxique), leur fonction (annotation fonctionnelle) ainsi que les relations entre les entités biologiques relatives au génome (annotation relationnelle). L'information résultante enrichit les bases de données biologiques. Voir aussi Ontologie.

Appariement (**Base pairing - Match**).

a. Appariement de bases : Formation de liaisons hydrogène entre des paires de bases complémentaires. L'appariement des bases est la clef des transferts d'information fondamentaux tels que la réplication et la transcription et permet la renaturation de l'ADN, l'hybridation ADN-ARN, l'utilisation de sondes.

b. Appariement de chromosomes homologues : opération s'effectuant au cours de la métaphase lors de la méïose et durant laquelle les deux chromosomes homologues vont échanger du matériel génétique.

c. Appariement (dans une homologie : identité entre deux caractères en vis à vis dans une homologie entre deux séquences (Voir le contraire : mésappariement).

ARN (Acide RiboNucléique) (**RNA = RiboNucleic Acid**).
L'ARN est une macromolécule biologique formée de ribonucléotides permettant de transférer et de traiter l'information dans la cellule. L'ARN est une séquence d'acide nucléique linéaire, simple brin (=monocaténaire). On distingue les ARN messagers, ARN de transfert, les ARN ribosomaux, les ARN nucléaires et les ARN cytoplasmiques.

ARN messager (ARNm - Transcrit) (**messenger RNA = mRNA**).
L'ARNm, transcrit à partir d'un gène protéique, est un ARN destiné à être traduit en protéine dans le cytoplasme.

ARN ribosomal (**ribosomal RNA = rRNA**).
L'ARNr est un des deux constituants des ribosomes. Molécule fonctionnelle mais non traduite et impliquée dans la traduction.

ARN polymérase (**RNA polymerase**).

Enzyme catalysant la synthèse d'ARN à partir d'ADN ou d'ARN. Chez les Procaryotes, l'ARN polymérase ADN dépendante est unique. Chez les Eucaryotes, on connaît trois ARN polymérases ADN dépendantes, spécialisées dans la transcription des différents groupes de gènes cellulaires (ARN polymérase I ou A pour les ARN ribosomiques, ARN polymérase II ou B pour les ARN messagers, ARN polymérase III ou C pour les petits ARN). Certains virus à ARN ont une polymérase ARN dépendante appelée aussi réplicase.

B

Bioinformatique (**bioinformatics**).
Discipline fondée sur les acquis de la biologie, des mathématiques et de l'informatique. Elle propose des méthodes et des logiciels qui permettent de gérer, d'organiser, de comparer, d'analyser, d'explorer l'information génétique et génomique stockée dans les bases de données afin de prédire et produire des connaissances nouvelles dans le domaine ainsi qu'élaborer de nouveaux concepts.
Bioinformatique fonctionnelle : ensemble des techniques bioinformatiques destinées à déterminer la fonction d'un gène à partir de sa séquence. (cf : Génomique fonctionnelle)

C

Chaîne de Markov (= modèles de Markov) (**Hidden Markov Models**).
Progression aléatoire estimant la probabilité de transition à chaque étape et utilisant la technique d'apprentissage. Méthode prévisionnelle algorithmique utilisée en bioinformatique pour prédire la jonction intron-exon, la conformation d'une protéine ...

Chromosome

- en génétique : ensemble d'éléments d'information liés entre eux dans une même molécule d'ADN.
- en biologie cellulaire : le chromosome est une structure cytologique, typiquement eucaryotiques, résultant d'une hypercondensation de la chromatine, permettant la répartion du matériel génétique entre les cellules filles lors de la mitose ou de la méïose. Chromosome vient de "chromos", couleur : allusion à la capacité de ces organites de fixer les colorants (Voir FISH). Les chromosomes ne sont visibles, en général, que durant la division cellulaire.
- Chromosomes homologues (**homologous chromosome**) : Paire de chromosomes (l'un vient du père, l'autre de la mère). Ils comportent les mêmes loci mais pas nécessairement équipés des mêmes allèles.

Code génétique (**genetic code**).
Le code génétique est le système de correspondance (code) permettant au message génétique (acides nucléiques, alphabet de 4 lettres) d'être traduit en protéine (alphabet de 20 lettres) par une cellule. A chaque séquence de trois bases consécutifs (codon) portées par l'ARN messager, correspond un acide aminé donné et un seul. C'est le code génétique qui permet donc la traduction des messages codés dans le génome en protéines ayant des fonctions bien précises. Il y a 64 combinaisons codon-acide aminé possibles pour 20 acides aminés seulement : un même acide aminé peut donc être codé par plusieurs codons différents (codons synonymes) : on parle de "code dégénéré".

Codon (**codon**).
Combinaison de trois nucléotides (triplet) codant pour un acide aminé.

- codon d'initiation (**initiation codon = start codon**) : triplet qui signale le début du message génétique sur un ARNm., c'est à dire le début de la traduction de l'ARNm en protéine. Le codon d'initiation est le plus souvent AUG, qui code pour une méthionine.
- codon stop = codon non-sens (stop codon) : triplet ne codant pour aucun acide aminé et signalant de ce fait la fin d'un message génétique sur un ARNm. Les codons stop sont le plus souvent UAA, UAG, UGA.

Complémentarité (**complementarity**).

- Bases complémentaires (complementary bases) : capacité des bases organiques de nucléotides à s'apparier par des liaisons hydrogène : deux liaisons hydrogène pour les paires de bases A-T (adénine-thymine)ou A-U (adénine-uracile) et trois liaisons pour G-C (cytosine-guanine). Cela confère la structure double hélice à l'ADN.
- Séquences complémentaires (complementary sequences) : séquence d'acides nucléiques constituée d'une succession de bases complémentaires à celles d'une autre séquence (et donc capable de s'hybrider avec elle).

Consensus (**consensus**).
Séquences nucléiques ou protéiques, impliquées généralement dans des fonctions semblables, et ne présentant que quelques variations mineures. La séquence consensus est établie, après alignement multiple de ces séquences homologues, en notant le monomère le plus fréquent en chaque position.

Conservation (**conservative residue**).
Remplacement d'un acide aminé par un autre ayant les mêmes caractéristiques physico-chimiques.

Crossing-over (= enjambement chromosomique) (**crossing-over**).
Le crossing-over est un mécanisme chromosomique se produisant au cours de la méïose et permettant la recombinaison de molécules d'ADN homologues appartenant à deux chromatides non soeur d'un bivalent avec échange réciproque : les paires de chromosomes homologues s'associent, croisent puis échangent certains segments de leur matériel génétique, avant de se séparer.

D

Délétion (**deletion**).
Mutation entrainant la perte d'éléments d'information (allant du nucléotide au fragment de chromosome).
Dogme central de la biologie moléculaire (**Central dogma**).
Le dogme central a été introduit pour la première fois par Francis Crick, puis établi par Jacques Monod, François Jacob et André Wolf en 1965. Il assied les mécanismes de stockage de l'information biologique, de réplication et de l'expression génique (transcription, traduction).
Duplication (**duplication**).
Doublement d'éléments génétiques. Soit de tout le génome, par réplication de l'ADN, soit de portions plus restreintes : duplication de gènes ou de segments de chromosomes pouvant conduire à des anomalies chromosomiques.

E

Eucaryote (**eukaryote**).
Organisme vivant dont les cellules possèdent un noyau au sein duquel est isolé le génome nucléaire. Celui-ci est représenté par un nombre de chromosomes constant pour une espèce donnée.

F

Fonction d'un gène (**gene function**).
La fonction d'un gène correspond au phénotype (caractéristiques moléculaires, biologiques et physiologiques) auquel il conduit.

G

Gap (Brèche).
Espace introduit artificiellement dans un alignement pour contre balancer (et matérialiser) une insertion dans une autre séquence, cela afin d'optimiser l'alignement entre ces deux séquences.
- Indel : un évènement d'insertion ou de délétion.

Gène (**gene**).
Le gène est un segment d'ADN ou d'ARN (certains virus) situé à un endroit bien précis (locus) sur un chromosome et porteur d'une information génétique. On distingue trois types de gènes (au niveau fonctionnel) : les gènes protéiques, les gènes spécifiant des ARN non traduits et les gènes régulateurs.
- gène structural (**structural gene**) : Les gènes structuraux regroupent les gènes protéiques et les gènes spécifiant les ARN non traduits.
- gène protéique (**protein gene**) : gène codant pour une protéine.
- gène régulateur (**regulatory gene**) : gène non transcrits et dont la fonction est de spécifier des sites d'initiation et de terminaison de la réplication (gènes réplicateurs), des sites d'attachement nécessaires à la ségrégation des chromosomes (gènes ségrégateurs) ou des sites de reconnaissance pour les enzymes de recombinaison (gènes recombinateurs).
- gène de régulation = gène de contrôle () : gène dont le produit (protéine de régulation) module l'expression (contrôler le taux d'expression) d'un ou de plusieurs autres gènes par sa fixation sélective sur un site de régulation ou par interaction protéine-protéine.
- gènes orthologues (**orthologs**) : gènes d'espèces différentes dont les séquences sont homologues, dérivent d'un même gène ancestral et ont divergés à la suite d'un évènement de spéciation. Peuvent ou non avoir la même fonction. (voir The Concepts of Orthology and Paralogy).
- gènes paralogues (**paralogs**) : gènes d'une même espèce dont les séquences sont homologues et résultent de la duplication d'un même gène ancestral.

Génétique (**genetics**).
Science de l'hérédité. La génétique étudie les caractères héréditaires des individus, leur transmission au fil des générations et leurs variations (mutations). C'est l'étude de cette transmission héréditaire qui a permis l'établissement des lois de Mendel.
- Information génétique (genetic information) : ensemble des instructions héréditaires qui contribuent à la réalisation du phénotype. Ces instructions codent d'une part les protéines ainsi que les ARN transcrits mais non traduits (gènes de structure) et interviennent sur le contrôle de leur expression (gènes de régulation, séquences de régulation).

Génie génétique (**genetic engineering**).
Ensemble de techniques permettant de modifier le patrimoine héréditaire d'une cellule par la manipulation de gènes in vitro.
Génome (**genome**).
Ensemble du matériel génétique (patrimoine héréditaire) d'un individu ou d'une espèce. Il est constitué de molécules d'acides nucléiques (ADN ou ARN). Les gènes, c'est-à-dire les parties d'ADN porteuses d'une information génétique, ne constituent qu'une partie du génome. Chez les virus, le génome viral est contenu dans une molécule d'ADN simple ou double brin ou une molécule d'ARN.Chez les procaryotes, on distingue :
- le génome chromosomique (une molécule d'ADN circulaire).
- le génome extrachromosomique (plasmides, épisomes).

Chez les eucaryotes, on distingue :

- le génome nucléaire (du noyau)
- le génome mitochondrial (des mitochondries), transmis par la mère.
- le génome chloroplastique (des chloroplastes), chez les plantes.

Génomique (**genomics**).
Science qui étudie les génomes. La génomique est l'étude et l'analyse exhaustive et multidisciplinaire des génomes. Elle vise à dresser l'inventaire de l'ensemble des gènes d'un organisme à les localiser sur les chromosomes et à caractériser leur séquence ainsi qu'à étudier leur fonction. La génomique se compose de deux volets complémentaires. Le premier volet correspond à l'analyse structurale (structure physique et organisation) du génome et du protéome : elle s'appuie sur l'analyse des séquences (issue du séquençage) et sur les données de la génomique structurale. Le second volet correspond à la génomique fonctionnelle (ou post-génomique), qui étudie la fonction des gènes, leur mode de régulation et leurs interactions. La génomique, dans son acceptation la plus large, englobe l'analyse de la structure physique du génome et du protéome (qui s'appuie sur l'analyse des séquences et les données de la génomique structurale) et la génomique fonctionnelle (ou post-génomique).
Génomique fonctionnelle ("post-génomique") (**functional genomics**).
Etude et analyse directe du transcriptome et du protéome : elle vise à déterminer la fonction des gènes à partir de leurs produits d'expression (ARN et protéines), ainsi qu'à étudier leur mode de régulation et leurs interactions.
Génomique structurale (**genomics**).
Désigne l'étude du protéome (ensemble des protéines identifiées à partir des génomes) du point de vue de la structure tridimensionnelle : étude des familles structurales protéiques, des repliements structuraux et de la relation structure-fonction.

H

Hérédité (**heredity**).
Transmission de l'information génétique d'une génération à une autre. "La génétique est la science de l'hérédité."

- hérédité mendélienne (= hérédité monogénique - hérédité monofactorielle) : transmission des maladies génétiques dues à une mutation dans un seul gène.

Homologie (**homology**).
Ressemblance héritée d'un même ancêtre commun. (Deux êtres vivants qui partagent un même caractère homologue l'ont hérité d'un même ancêtre commun chez qui ce caractère est apparu)

- Molécule, organite, organe ayant la même structure et (ou) fonction qu'un autre. L'identité n'est pas forcément parfaite. Voir chromosomes homologues.
- Deux organes sont dits homologues lorsqu'ils présentent le même plan d'organisation et les mêmes relations avec les organes voisins.
- Deux séquences sont dites homologues si elles ont un ancêtre commun. En pratique, l'homologie est mise en évidence en recherchant des similitudes entre les séquences.

Horloge moléculaire : Hypothèse selon laquelle une certaine molécule évoluerait de façon constante au cours du temps. Selon cette théorie, des vitesses évolutives différentes sont possibles pour différentes molécules. Si l'on admet cette théorie, et que l'on connaît le taux d'accumulation des mutations, il est possible d'estimer le temps de divergences d'espèces en comparant leur diversité moléculaire.

I

Identité (**identity**). Acides aminés ou nucléotides invariants entre deux séquences.
Insert (**insert**). Séquence d'ADN étranger introduite dans une molécule d'ADN donnée.
Insertion (**insertion**).

1. Mutation par adjonction d'un nucléotide au sein d'une séquence d'ADN.
2. Introduction d'une séquence d'ADN à l'intérieur d'une autre, par recombinaison in vivo ou in vitro. Le terme d'intégration est parfois utilisé : intégration d'un génome viral dans le génome de la cellule hôte, intégration d'un épisome dans le chromosome bactérien.

M

Matrice de substitution (**substitution matrix**). (Matrice 20x20).
Elle contient les coûts de substitution d'un acide aminé par un autre à partir de laquelle sera extraite la valeur du score élémentaire permettant de quantifier la similarité entre deux protéines. Elle permet de tenir compte des similitudes variables entre acides aminés et donc de pondérer différemment chacune des substitutions possibles.

- matrice unitaire (**unitary matrix - identity matrix**) : seuls les caractères identiques de la matrices recoivent le score de 1, les autres ont un score de 0. Cette matrice permet notamment de désigner le système de scores appliqué aux séquences nucléiques.

Mésappariement (**mismatch = misparing**).
Non correspondance entre deux bases en vis à vis dans le cadre d'une recherche d'homologie ou d'alignement entre deux séquences. (Voir le contraire : appariement). Un mésappariement peut être :

- soit la substitution d'un caractère par un autre : elle correspond à une mutation.
- soit l'introduction d'un "gap", permettant d'optimiser l'alignement entre les deux séquences.

Mesure de similarité : C'est une fonction qui associe une valeur numérique à une paire de séquences (plus cette valeur est élevée plus la similarité est importante).
Mesure de distances : C'est une fonction qui associe une valeur numérique positive ou nulle à une paire de séquences (plus la distance est petite, plus la similarité est importante).
Motif (**motif**).
Segment court et conservé d'une séquence nucléique ou protéique. Les motifs sont fréquemment des parties hautement conservées des domaines.
Mutagénèse dirigée (**site specific mutagenesis = site directed mutagenesis**).
Introduction d'une mutation précise dans un fragment d'ADN cloné, suivie de la réinsertion de la séquence mutée dans le gène original en remplacement de l'ADN sauvage correspondant.
Mutation (**mutation**).
Toute modification du génotype par altération de la séquence d'un fragment d'ADN allant de la modification d'une seule paire de nucléotides au réarrangement ou à la perte d'un morceau de chromosome détectable à l'observation cytologique. Si elle n'est pas létale, elle est héréditaire.

- mutation ponctuelle (**point mutation**) : mutation portant sur une seule base (substitution, insertion ou délétion)
- mutation de chromosome (**mutation of chromosome**) : délétion, insertion, translocation, inversion, duplication.

N

Neigbor−Joining : Méthode de reconstruction d'arbres phylogénétique basée sur les distances et autorisant des taux de mutations différents sur les branches.
Nucléotide (**nucleotid**).
Base azotée. Unité de construction des acides nucléiques, résultant de l'addition d'un sucre (ribose pour l'ARN et désoxyribose pour l'ADN), d'un groupement phosphate et d'une base azotée. Il existe quatre nucléotides différents pour l'ADN : adénine (A), thymine (T), guanine (G), cytosine (C) et quatre nucléotides différents pour l'ARN : uracile (U), guanine (G), cytosine (C), adénine (A). C'est la succession des bases résultant de l'enchaînement des nucléotides dans l'acide nucléique qui constitue le message génétique. Ces quatre molécules élémentaires ont la particularité de s'unir deux à deux par complémentarité. La taille d'une séquence d'ADN est donnée en pb (paires de bases), celle de l'ARN en **nt** (nucléotides).

O

Orthologue Voir Gène orthologue.
P

Paralogue Voir Gène paralogue.
Polymérase (**polymerase**).
Enzyme capable d'enchaîner des nucléotides en polymères d'ADN ou d'ARN (ARN polymérase, ADN polymérase).
Procaryote (**prokaryote**).
Organisme (bactéries, cyanophycées) dont la cellule ne possède pas de noyau, contrairement aux eucaryotes. Le génome est constitué d'un unique chromosome circulaire. Il peut aussi coexister des structures extrachromosomiques, sous forme de molécules d'ADN circulaires libres dans le cytoplasme (plasmides), qui se répliquent de façon indépendante.
Protéine (**protein**).
La protéine est un produit du gène issu de la synthèse protéique via le code génétique. Les protéines sont des macromolécules constituées de longues chaînes d'acides aminés (de 50 à 30000 acides aminés, la moyenne étant d'environ 400) qui se replient sur elles-même et adoptent des conformations très spécifiques dans l'espace. L'ensemble des protéines codées sur le génome (= le protéome) peut être ainsi considéré comme une collection de repliements 3D suffisants pour assurer les principales fonctions cellulaires, comme le métabolisme, la réplication ou la gestion de l'information.

R

Réplication (**replication**).
La réplication est le mécanisme de synthèse de l'ADN permettant de transmettre l'information génétique d'une cellule ou d'un organisme à sa descendance. Ce mécanisme repose sur la complémentarité des bases. Chaque molécule fille d'ADN est constituée d'un brin ancien qui sert de matrice à un brin néosynthétisé. Ceci conduit à la duplication des molécules d'ADN de tout le génome.

S

Score (**score**).

Un score global permet de quantifier l'homologie : il résulte de la somme des scores élémentaires calculés sur chacune des positions en vis à vis des deux séquences dans leur appariement optimal. D'une manière simple, c'est le nombre total de "bons appariements" pénalisé par le nombre de mésappariements.

- Ø score élémentaire : dans le cas des séquences nucléiques, la valeur du score élémentaire est de 1 ou 0 : les deux bases sont identiques (bon appariement) ou différentes (mauvais appariement). Dans le cas des séquences protéiques, cette valeur est extraite d'une matrice de substitution.

Séquence (**sequence**). Succession de monomères dans un polymère. L'orientation de la séquence est définie par la synthèse du polymère.

- Ø Les séquences nucléiques (ADN ou ARN) sont des polynucléotides (polymères de nucléotides, reliés par des liaisons "phosphodiester". Les molécules linéaires présentent une extrémité dite 5'P (en raison du nucléotide dont une fonction alcool, en position 5' du sucre, est estérifiée par un acide phosphorique) et une extrémité 3'OH. Orientation de 5' en 3'.
- Ø Les séquence protéiques (= protéines) sont des polymères d'acides aminés. Orientation NH2 terminal à COOH terminal.

Séquençage (**sequencing**).
Détermination de l'ordre linéaire des composants d'une macromolécule (les acides aminés d'une protéine, les nucléotides d'un acide nucléique, etc.). Le séquençage de l'ADN ("décryptage" du génome) s'effectue selon le protocole enzymatique de Sanger.

- Ø Séquençage d'étiquettes (**signature sequencing**) : pour identifier un gène, on n'utilise que la séquence d'un petit fragment, ou étiquette (tag), correspondant à la signature des gènes.

Similarité (Similitude) (**similarity**).
Mesure de la ressemblance entre séquences protéiques ou nucléiques. Le degré de similarité entre deux séquences, quantifiée par un score, est basée sur le pourcentage d'identités et/ou de substitutions conservatives des séquences. Une bonne similarité pourra ainsi conduire à une homologie.

Substitution (**substitution**).
Mutation ponctuelle consistant au remplacement d'un nucléotide par un autre dans une séquence nucléique ou d'un acide aminé par un autre dans une protéine. Les matrices de substitution permettent de rendre compte des similitudes variables entre acides aminés, donc des conséquences inégales (coûts évolutifs différents) des mutations (substitution) d'un acide aminé par un autre.

- Ø Substitution conservative (=conservation) (conservative substitution-conservation) : changement à une position spécifique d'une séquence nucléique ou protéique qui préserve les propriétés physico-chimiques du résidu originel.

T

Traduction (synthèse protéique) (**translation (protein synthesis)**).
Processus permettant la synthèse d'une chaîne polypeptidique (protéine) à partir d'un brin d'ARN messager. La traduction a lieu au niveau des ribosomes.

Transcription (**transcription**).
La transcription est la synthèse d'une molécule d'ARN complémentaire (ARN messager) à une séquence d'ADN. La transcription est initiée par une ARN polymérase.Les ARNm sont traduits tels quels chez les procaryotes. Chez les eucaryotes, l'ARNm subit une maturation avant la traduction.

Transition (**transition**).
Substitution (mutation) d'une purine (Adénine, Guanine) par une pyrimidine (Cytosine, Thymine) ou inversement. Voir Transversion.

Transversion (**transversion**).
Substitution (mutation) d'une purine (Adénine, Guanine) par une purine ou d'une pyrimidine (Cytosine, Thymine) par une pyrimidine. Voir Transition.

U

UPGMA (Unweight Pair Group Method with Arithmetic mean) : Méthode de reconstruction d'arbres phylogénétiques basée sur les distances, rapide mais très sensible à des taux de mutations différents sur les branches de l'arbre.

Résumé

Dans ce travail de magistère, nous traitons un problème connu en bioinformatique : l'alignement multiple de séquences (MSA). L'alignement multiple de séquences est une tâche fondamentale car il constitue une plateforme essentielle pour plusieurs autres tâches telles que la construction des arbres phylogéniques et la prédiction structurelle et fonctionnelle des protéines, etc. Par ailleurs, c'est un problème d'optimisation caractérisé par une grande complexité temporelle et spatiale. Pour sa résolution, nous avons adopté une approche quantique évolutionnaire dont le développement a conduit à la proposition de deux méthodes: QEAMSA et QUANTALIGN. Ces dernières reposent sur un noyau de base défini par une représentation quantique appropriée et une dynamique quantique évolutionnaire adaptée. Elles se distinguent des autres méthodes évolutionnaires par une taille de population réduite et un nombre raisonnable d'itérations pour trouver les meilleurs alignements grâce aux principes de l'informatique quantique tels que la superposition d'états, l'interférence et la mutation. Une autre caractéristique de ces méthodes est leur capacité d'optimiser n'importe quelle fonction objectif. Les études expérimentales intensives prouvent que le programme QUANTALIGN produit des alignements qui, hormis la préservation des caractéristiques fonctionnelles et structurelles, sont comparables voir meilleures que d'autres méthodes populaires de l'alignement multiple de séquences.

Bibliographie

[Adleman, 94]: L. Adleman. "Molecular computation of solutions of combinatorial problems". Science 266:pp. 1021-1024, 1994.

[Altschul et al, 90]: S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. "Basic local alignment search tool". J Mol Biol, 10: pp. 215:403, 1990.

[Altschul, 91]: S. Altschul. "Amino acid substitution matrices from an information theoretic perspective". J. Mol. Biol 219:pp. 555 -665, 1991.

[Altschul et al, 89]: S F. Altschul, R J. Carroll, and D J. Lipman. "Weights for data related by a tree", J. Mol.Biol, 207:pp. 647-653, 1989.

[Barton et al, 87]: G J. Barton, M J E. Sternberg. "A strategy for the rapid multiple alignment of protein sequences: confidence levels from tertiary structure comparisons". J. Mol. Biol 198:pp. 327-337, 1987.

[Berger et al, 91]: M P. Berger, P J. Munson. "A novel randomized iterative strategy for aligning multiple protein sequences". Comput. Appl. Biosci 7: pp. 479-484, 1991.

[Berry, 97]: V. Berry: "Méthodes et algorithmes pour reconstruire les arbres de l'évolution". Thèse de Doctorat, Université des sciences et techniques du Languedoc, 1997.

[Biron et al, 98]: D. Biron, O. Biham, E. Biham, M. Grassl and D A. Lidar. "Generalized Grover Search Algorithm for Arbitrary Initial Amplitude Distribution". Los Alamos Physics Preprint Archive, /quant-ph/9801066, 1998.

[Blais et al, 00]: A. Blais, A.M. Zagoskin. "Operation of universal gates in a solid-state quantum computer based on clean Josephson junctions between d-wave superconductors". Phys. Rev. A, 61:042308, 2000.

[Blais, 02]: A.Blais. "Algorithmes et architectures pour ordinateurs quantiques supraconducteurs". Thèse de doctorat, faculté des science, université de sherbrooke, canada, 2002.

[Brassard, 96]: G. Brassard. "New Trends in Quantum Computing". arxive.org, quant-ph/9602014 19, 1996.

[Brundno et al, 04]: D C. Brudno M. Batzoglou. "ProbCons: probabilistic consistency-based multiple alignment of amino acid sequences". Proceedings Nineteenth National Conference on Artificial Intelligence, 703-708, 2004.

[Dayhoff et al, 78]: M O. Dayhoff, R.M. Schwartz and B C. Orcutt. "A model of evolutionary change in proteins, Atlas of protein sequence and structure". National Biomedical Research Foundation. 5: pp. 345-352, 1978.

[Deutsch, 85]: D. Deutsch. "Quantum theory, the Church-Turing principle and the universal quantum computer". Proceedings of the Royal Society of London Ser. AA400, pp. 97:117, 1985.

[Dirac, 59]: P. Dirac, "The Principles of Quantum Mechanics", Oxford University Press, (4th ed.), 1958.

[Draa et al, 04]: A. Draa H. Talbi, M. Batouche. "Un algorithme génétique quantique pour le recalage d'images multi-sources". Proceeding du 4ieme séminaire national en informatique, (SNIB'04), pp. 205:216, 2004.

[Durbin et al, 98]: R. Durbin, A. Krogh, A., and G. Mitchison. "Biological Sequence Analysis". Cambridge University Press. Cambridge, UK 1998.

[Eddy, 95]: S R. Eddy. "Multiple alignment using hidden Markov models". Third international conference on intelligent systems for molecular biology (ISMB). Cambridge England: Menlo Park CA: AAAI Press, 1995.

[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.

[Feng et al, 87]: D. Feng and R. Doolittle, "Progressive sequence alignment as a prerequisite to correct phylogenetic trees". J. Mol. Evol., 25:pp. 351-360, 1987.

[Fitch, 66]: W M. Fitch. "An improved method of testing for evolutionary homology". J. Mol. Biol., 16: pp. 9-16, 1966.

[Fitch et al, 67]: W M. Fitch and E. Margoliash. "Construction of Phylogenetic Trees". Science, Vol.155, No. 20, pp. 279-284, 1967.

[Gardner et al, 05]: P. Gardner, A. Wilm, S. Washietl. "A benchmark of multiple sequence alignment programs upon structural RNAs". Nucleic Acids Research, Vol. 33, No. 8, pp. 2433–2439, 2005.

[Gonnet et al, 92]: G. H., Gonnet, M.A Cohen, and S.A Benner. "Exhaustive matching of the entire protein sequence database". Science, 256:pp. 1443-1445, 1992.

[Gotoh, 96]: O. Gotoh. "Significant Improvement in Accuracy of Multiple Protein Sequence Alignments by Iterative Refinements as Assessed by Reference to Structural Alignments". J. Mol. Biol.Vol. 264, No. 4, 823-838, 1996.

[Grover, 96]: L K. Grover, "A fast quantum mechanical algorithm for database search". The Twenty-Eighth Annual ACM Symposium on the Theory of Computing, pp. 212–219, 1996.

[Gupta et al, 95]: S K. Gupta, J. Kececioglu and A A. Schäffer. "Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment". J. Comput. Biol, 2: pp. 459-472, 1995.

[Han et al, 00]: K. Han and J. KIM, "Genetic Quantum Algorithm and its Application to Combinatorial Optimization Problem," In Proceedings of the 2000 Congress on Evolutionary Computation, IEEE Press, pp. 1354-1360, 2000.

[Han et al, 01]: K. Han and J. KIM , "Analysis of Quantum-inspired Evolutionary Algorithm," In Proccedings of the 2001 International Conference on Artificial Intelligence, CSREA Press, Vol. 2, pp. 727-730, June 2001.

[Henikoff et al, 92]: S. Henikoff, and J.G. Henikoff. "Amino acid substitution matrices from protein blocks". Proc. Natl. Acad. Sci. 89:pp. 10915-10919, 1992.

[Henikoff et al, 93]: S. Henikoff, and J.G. Henikoff. "Performance evaluation of amino acid substitution matrices", Proteins, 17:pp. 49- 61, 1993.

[Hogeweg et al, 84]: P .Hogeweg and B. Hesper. "The alignment of sets of sequences and the construction of phylogenetic trees. An integrated method". J. Mol. Evol. 20, 175-186, 1984.

[Holland, 75]: J H. Holland. "Adaptation in Natural and Artificial Systems", Ann Arbor: The University of Michigan Press, 1975.

[Jiang et al, 94]: T.Jiang, E. Lawler, and L. Wang. "Aligning Sequences via an Evolutionary Tree". In the Proceedings of the Annual ACM Symposium on Theory of Computing, pp. 760-769, 1994.

[Johnson et al, 93]: M S. Johnson and J P. Overington. "A structural basis for sequence comparisons. An evaluation of scoring methodologies". J. Mol. Biol. 233:pp. 716-738, 1993.

[Jones, 99]: D T. Jones. "Protein secondary structure prediction based on position-specific scoring matrices". J Mol Biol 292(2):pp. 195-202, 1999.

[Jourdon et al, 03]: L. Jourdon. " Metaheuristiques Pour L'extraction De Connaissances : Application A La Génomique". Thèse doctorat, Université Des Sciences Et Technologies De Lille, U.F.R. D'I.E.E.A. No.3368, 2003.

[Jovanovich, 04]: O. Jovanovich, "Introduction to Computational Biological", ICB, 2004.

[karlin et al, 93]: S. Karlin and S F. Altschul." Applications and statistics for multiple high-scoring segments in molecular sequences". Proc. Natl. Acad. Sci USA, 90: pp. 5873- 5877, 1993.

[Katoh et al, 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 Research, Vol. 30, No. 14, pp. 3059-3066, 2002.

[Kim et al, 94]: J. Kim, S. Pramanik and M.J. Chung, "Multiple sequence alignment using simulated annealing". Computer applications in bioscience, 10: pp. 419-426, 1994.

[Kim et al, 03]: K. Kim, j. Hwang, k. Han, J. Kim, "A Quantum-inspired Evolutionary Computing Algorithm for Disk Allocation Method," IEICE Transactions on Information and Systems, IEICE Press, Vol. E86-D, No. 3, pp. 645-649, 2003.

[Lambert et al, 03]: C. Lambert, J. Van Campenhout, X. DeBolle and E. Depiereux," Review of Common Sequence Alignment Methods: Clues to Enhance Reliability", Current Genomics, 4: pp. 131-146, 2003.

[Le Bellac ,03]: M. Le Bellac, "Introduction à l'informatique quantique", Cours donné à l'Ecole Supérieure de Sciences Informatiques (ESSI), Octobre 2003.

[Lipman et al, 88]: D. Lipman and W. Pearson. "Improved tools for biological sequence comparison". Proc. Natl. Academy Science, 85:pp. 2444-2448, 1988.

[Lipman et al, 89]: D J. Lipman, S F. Altschul, and J D. Kececioglu. "A tool for multiple sequence alignment". Proc. Nat. Acad. Sci. USA 86: pp. 4412-4415, 1989.

[Luscombe et al, 01]: N.M. Luscombe, D. Greenbaum and M. Gerstein. "What is bioinformatics? An introduction and overview". Yearbook of Medical Informatics, pp. 83:100, 2001.

[McLachlan et al, 72]: A D. McLachlan. "Repeating sequences and gene duplication in proteins". J. Mol. Biol., 64: pp. 417-437, 1972.

[Melcher, 99]: U. Melcher, "Molecular Genetics", http://opbs.okstate.edu/emelcher/UM.html,1999.

[Meshoul et al, 05a]: S. Meshoul, A. Layeb, M. Batouche and H.Talbi. "QEAMSA: Multiple Sequence Alignment using Quantum Evolutionary algorithm", to appear in proceeding of ACIIT2005, 2005.

[Meshoul et al, 05b]: S. Meshoul, A. Layeb and M. Batouche. "A Quantum Evolutionary Framework for Multiple Sequence Alignment" to appear in proceeding of EPIA LNAI 2005, 2005.

[Moore, 95]: M. MOORE, "Quantum-inspired computing ", department of computer science, old library, university of Exeter, UK, 1995.

[Morgenstern, 04]: B.Morgenstern. "DIALIGN: Multiple DNA and protein sequence alignment". BiBiServ. Nucleic Acids Research, 32:W33-W36, 2004.

[Mount, 01]: D W. Mount. "Bioinformatics: sequence and genome analysis". Cold spring, Harbor laboratory press, cold spring Harbor, NY, 2001.

[Needleman et al, 70]: S B. Needleman and C D. Wunsch. "A general method applicable to the search for similarities in the amino acid sequence of two proteins". J Mol Biol, 48(3):pp. 443.53, Mars 1970.

[Nguyen et al, 02]: H. Nguyen, I.Yoshihara. "Aligning Multiple Protein Sequences by Parallel Hybrid Genetic Algorithm". Genome Informatics 13: 123–132, 2002.

[Nicolas et al, 02]: H B. Nicholas, A J Ropelewski, and D W Deerfield. "Strategies for multiple sequence alignment". Biotechniques, 32: 572-578.2002.

[Notredame , 02]: C.Notredame, "Recent progresses in MSA: a survey". pharmacogenomic, 3:pp. 1–14,2002.

[Notredame et al, 96]: C. Notredame and D G. Higgins. "SAGA: sequence alignment by genetic algorithm". Nucleic Acids Research, Vol. 24, No. 8, pp. 1515–1524, 1996.

[Notredame et al, 98]: C. Notredame. L. Holm and D.G. "COFFEE: An objective functions for multiple sequence alignments". Bioinformatics, Vol. 14, No. 5, pp. 407-22. 1998

[Notredame et al, 00]: C. Notredame, D. Higgins J. Heringa. "T-Coffee: a novel algorithm for multiple sequence alignment". J Mol Biol, 302:pp. 205-21, 2000.

[Pedersen, 00]: C N S. Pedersen. "Algorithms in Computational Biology", BRICS DS-00-4, ISSN 1396-7002, 2000.

[Pearson, 96]: W R. Pearson. "Effective protein sequence comparison". Methods Enzymol. 266: pp. 227-258, 1996.

[Preskill, 98]: J. Preskill. "Quantum Information and Computation". Lecture Notes for Physics 229, pp. 1:321, California institute of technology, 1998.

[Quinkal, 03]: I. Quinkal, "Quelques termes-clef de biologie moléculaire et leur définition". INRIA Rhône-Alpes, 2003.

[Raghava et al, 03]: G. Raghava, SMJ. Searle, P C. Audley, J D. Barber and G J. Barton. "OXBench: A benchmark for evaluation of protein multiple sequence alignment accuracy". BMC Bioinformatics, 10.1186/1471-2105-4-47, 2003.

[Reinert, 03]: K. Reinert. "Introduction to Multiple Sequence Alignment". Algorithmische Bioinformatik, WS 03, 10. November 2003.

[Riaz et al, 04]: T. Riaz, Y. Wang, and K.B Li, "Multiple sequence alignment using Tabu Search". Proc. 2nd Asia-Pacific Bioinformatics Conference (APBC), pp. 223-232. 2004.

[Rieffel et al, 00]: E. Rieffel and W. Polak. "An introduction to quantum computing for non-physicists". arxive.org, quant-ph/9809016 v2, 2000.

[Saitou et al, 87]: N. Saitou, and M. Nei. "The neighbor-joining method: a new method for reconstructing phylogenetic trees". Mol. Biol. Evol., 4:pp. 406-425, 1987.

[Setubal et al, 97]: J. Setubal and J. Meidanis: "Introduction to Computaional Molecular Biology". International Thompson Publishing Company, first edition, ISBN 0-534-95262-3 QH506.S49, 1997.

[Shor, 94]: P W. SHOR, "Algorithms for quantum computation: Discrete log and factoring", In Proceedings of the 35th Annual Symposium on Foundations of Computer Science pp. 124–134, 1994.

[Shor, 98]: P W. Shor," Quantum Computing", documenta mathematica, extra volume ICM –I-,pp. 467:484,1998.

[Simon, 94]: D. Simon. "On the Power of Quantum Computation". Proc. 35th Annual Symposium on Foundationsof Computer Science, 1994, pp. 116-123.

[Smith et al, 81]: T F. Smith and M S. Waterman. "Comparative biosequence metrics". J. Mol. Evol. 18:pp. 38-46. 1981.

[Steven et al, 98]: L. Steven L. Salzberg, D B. Searls, and S. Kasif,. "Computational Methods in Molecular Biology". Elsevier, 1998.

[Stoye et al, 97]: J. Stoye, V. Moulton A W. Dress " DCA: an efficient implementation of the divide and conquer approach to simultaneous multiple sequence alignment". Comput. Appl. Biosci. Vol. 13, No 6, pp. 625-631, 1997.

[Subramanian et al., 05]: A R. Subramanian, J W. Menkhoff, M. Kaufmann and B. Morgenstern. "DIALIGN-T: An improved algorithm for segment-based multiple sequence alignment". BMC Bioinformatics, doi: 10.1186/1471-2105-6-66, 2005.

[Talbi et al, 04]: H.Talbi, A. Draa, M. Batouche," a quantum-inspired genetic algorithm for multi source affine image registration", ICIAR'2004, LNCS 3211, pp. 147-154, 2004.

[Taylor, 88]: WR. Taylor "A flexible method to align large numbers of biological sequences". J. Mol. Evol. 28, pp. 161-169, 1988.

[Thomsen et al, 02]: R. Thomsen, G. Fogel, T. Krink. " Clustal Alignment Improver using Evolutionary Algorithms". Proceedings of the Fourth Congress on Evolutionary Computation , vol. 1, pp. 121-126 , 2002

[Thompson et al, 99a]: J D. Thompson, F. Plewniak, and O. Poch. "BAliBASE: A benchmark alignment database for the evaluation of multiple alignment programs". Bioinformatics, 15:pp. 87:88, 1999.

[Thompson et al, 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. 4673-80.1994.

[Thompson et al, 99b]: J D. Thompson, F. Plewniak, and O. Poch, "comprehensive comparison of multiple sequence alignment programs". Nucleic Acids Research, pp: 2682-2690, 1999.

[Van Walle et al, 04]: I. Van Walle, I. Lasters, and L. Wyns. "Align-m: a new algorithm for multiple alignment of highly divergent sequences". Bioinformatics, Vol. 20, No. 9:pp. 1428–35, Jun 2004.

[Vingron et al, 94]: M. Vingron, and M.S Waterman. "Sequence alignment and penalty choice: review of concepts, case studies and implications", J. Mol .Biol. 235:pp. 1-2, 1994.

[Vogt et al, 95]: G. Vogt et al. "an assessment of amine acid exchange matrices in aligning protein sequences: the twilight zone revisited" j.mol.bio 249: pp. 816-831, 1995.

[Wang et al, 94]: L. Wang and T. Jiang, "on the complexity of multiple sequence alignment". J. Comput. Biol., 1:pp. 337-348 1994.

[Waterman, 86]: M S. Waterman, "Multiple sequence alignment by consensus". Nucleic Acids Res, 14:pp. 9095-9102, 1986.

[Wilhelm, 00]: R. Wilhelm. "Informatics: 10 Years Back - 10 Years Ahead". (Ed.), Springer, p. 341-355, ISBN 3-540-41635-8, 2000

[Whitley, 93]: D. Whitley. "Genetic Algorithm Tutorial". Technical Report CS-93-103. Department of Computer Science Colorado State University, 1993.

[Zachariah et al, 05]: M A. Zachariah, G E. Crooks, S R. Holbrook, and S E. Brenner. "A Generalized Affine Gap Model Significantly Improves Protein Sequence Alignment Accuracy". PROTEINS: Structure, Function, and Bioinformatics 58:pp. 329–338, 2005.

Liste des figures

Liste des tables