الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma -

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Fin d'étude Master

Filière : Informatique

Option : Ingénierie des Médias(IM)

Thème :

Techniques d'Analyse des Séquences Biologique

Encadré Par :

M : LABSIR Rabah

Présenté par :

ABDAOUI Nasreddine

BENCHEIKH Badreddine

Juin 2017

Remerciement

Dieu, merci pour nous avoir donné la force et la volonté de mener à bien ce travail.

Nous tenons à remercier tout particulièrement notre encadreur, Monsieur Rabah Lebsir pour nous avoir fait confiance et nous avoir suggéré ce thème.

Nos remerciements les plus sincères, ainsi que nos gratitudes vont à nos parents ainsi que nos frères et sœurs, qui nous ont accompagné et soutenu tout au long de notre parcours.

Nous remercions les membres de jury de soutenance d'avoir accepté d'évaluer ce modeste travail.

A vous tous nous exprimons notre reconnaissance et notre gratitude car encore une fois c'est grâce à vous tous et à votre abnégation et à votre altruisme qu'on a pu poursuivre nos études.

Résumé

Dans ce mémoire de fin d'étude, nous concentrons sur les méthodes progressives pour l'alignement multiple des séquences biologiques, dans le cas des séquences biologiques à grande échelle.

L'objectif de cette étude est d'avoir un bon compromis entre le temps d'exécution et la qualité de l'alignement pour des masses très importantes de données biologiques à traiter.

Nous proposons une stratégie inspirée du paradigme diviser pour conquérir pour améliorer le temps d'exécution en utilisant un algorithme de clustering rapide pour le regroupement des séquences biologiques, puis un algorithme hybride pour améliorer la qualité de l'alignement des séquences multiple (MSA).

Nous avons testé notre approche sur un processeur multi-core avec un ensemble de benchmarks connus dans la littérature. Les résultats montrent que notre approche donne les meilleurs résultats en termes de temps de calcul par rapport aux techniques les plus utilisées, tout en perdant une légère précision dans tous bechmarks de référence testés.

Le code source de la stratégie proposée est disponible sur demande.

Contenu

Liste des figures

Chapitre 1 : L'alignement multiple des séquences

Chapitre 2 : Notre approche

Chapitre 3 : Implémentation

Liste des tableaux

Chapitre 1 : L'alignement multiple des séquences

Chapitre 2 : Notre approche

Introduction général

L'alignement multiple des séquences (MSA) est une tâche très importante dans la bioinformatique. Il permet de représenter deux ou plusieurs séquences de macromolécules biologiques (ADN, ARN ou protéines) l'une sous l'autre, pour faire ressortir les régions similaires ou homologues.

MSA est utilisé pour des tâches complexes telles que l'analyse des protéines, l'identification des sites fonctionnels dans les séquences génomiques, la prédiction structurelle et fonctionnelle. Malheureusement, faire un alignement multiple précis a été montré NP-Compliqué [1]. Par conséquent, MSA est un problème d'optimisation qui présente une grande complexité en temps et en espace. En conséquence, plusieurs méthodes ont été proposées pour faire face à ce problème. Ils peuvent être regroupés en trois classes [2]

La première classe comprend les méthodes exactes qui utilisent une généralisation de l'algorithme Needleman [3] pour aligner toutes les séquences simultanément. Bien que les méthodes exactes fournissent des solutions optimales, mais leur principal inconvénient est leur complexité en temps et en espace, ce qui devient encore plus critique avec le nombre et la longueur accrus des séquences et devient rapidement inutilisable.

La deuxième classe contient des méthodes basées sur une approche progressive [4]. L'alignement progressif crée un MSA final en combinant les alignements par paires, en commençant par la paire la plus semblable et en progressant vers les relations les plus éloignées. Les méthodes progressives sont simples, rapides et généralement donnent de bonnes qualités d'alignements. Cependant, leur principal inconvénient est le problème des minimums locaux et, par conséquent, ils peuvent conduire à de mauvaises solutions.

La troisième classe se compose de méthodes itératives. L'idée de base est de commencer par un alignement initial et d'affiner de façon itérative grâce à une série d'améliorations appropriées appelées itérations. Le processus est répété jusqu'à la satisfaction de certains critères. Ces méthodes ont été prometteuses mais peuvent être beaucoup plus lentes, ce qui peut les rendre inutilisables pour les données à grande échelle

1. L'alignement multiple de séquences

1.1. 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.[1]

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 1.1*). 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 1.2*). 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.

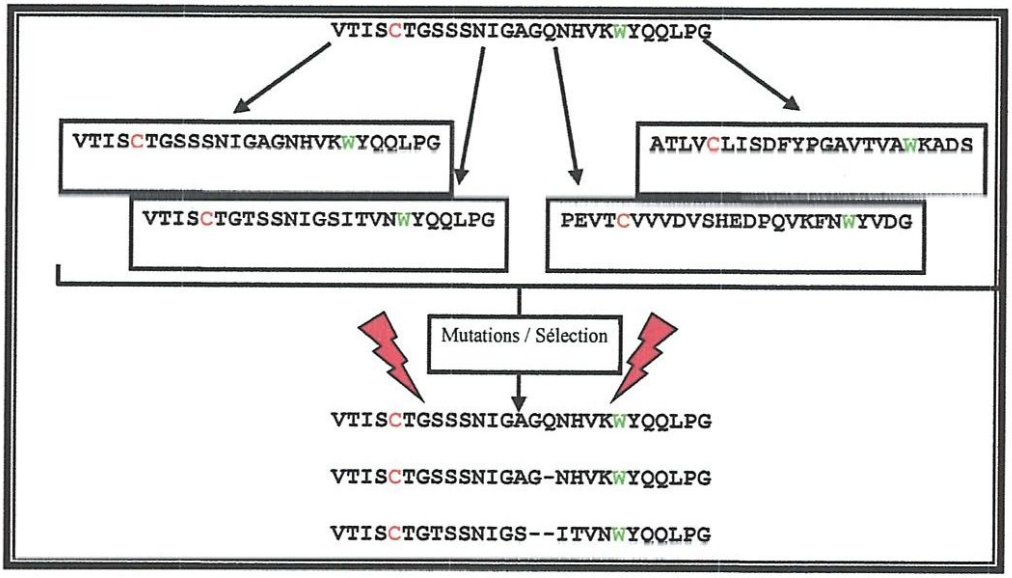


Figure 1.1 : *Alignement multiple : une histoire.*

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 [5]. En effet, pour aligner 3 séquences de taille 1000, il faut 10003 soit 1 Go de mémoire alors que les biologistes ont

[1] La principale source de ce chapitre est [5]

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 objective mathématique capable de mesurer la qualité biologique d'un alignement. En effet, une bonne fonction objective 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, T-COFFEE score, 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 consequence, 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 objective parfaite, le calcul mathématique de l'alignement optimal est une tâche très complexe pour qu'une méthode exacte soit employée. 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 fournissent habituellement un alignement très près de l'optimalité. Néanmoins, elles peuvent seulement manipuler un petit nombre de séquences (< 20) et sont limités à la fonction objective 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. 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.

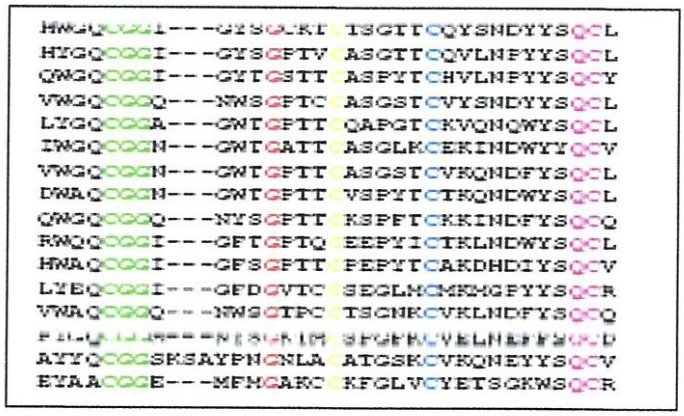


Figure 1.2 : *Site de fixation de la cellulose (extrait de prosite entrée PS00562).*

1.2. Les arbres phylogénétiques

Les arbres phylogénétiques jouent 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 1.3*) 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. La deuxième utilisation des arbres phylogénétiques en MSA est pour calculer les poids des séquences dans la fonction objectif WSP. 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 arbres non enraciné : pas d'ancêtre commun.

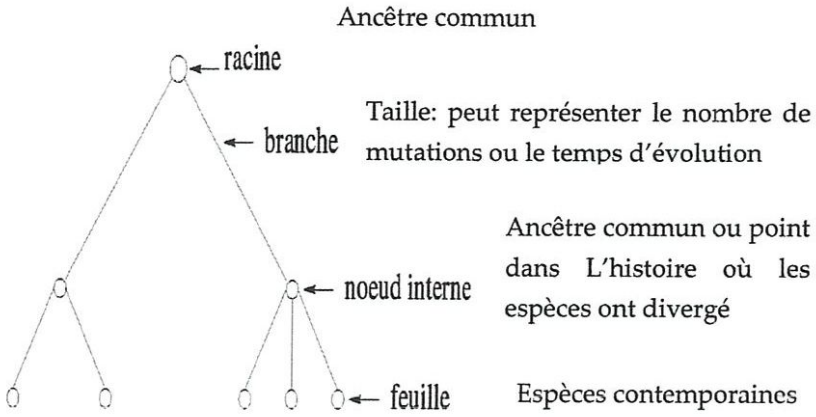


Figure 1.3 : *Un arbre phylogénétique enraciné.*

1.2.1. 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 1.4*).

1.2.2. 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. 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 modèle mathématique (mutation, temps...).

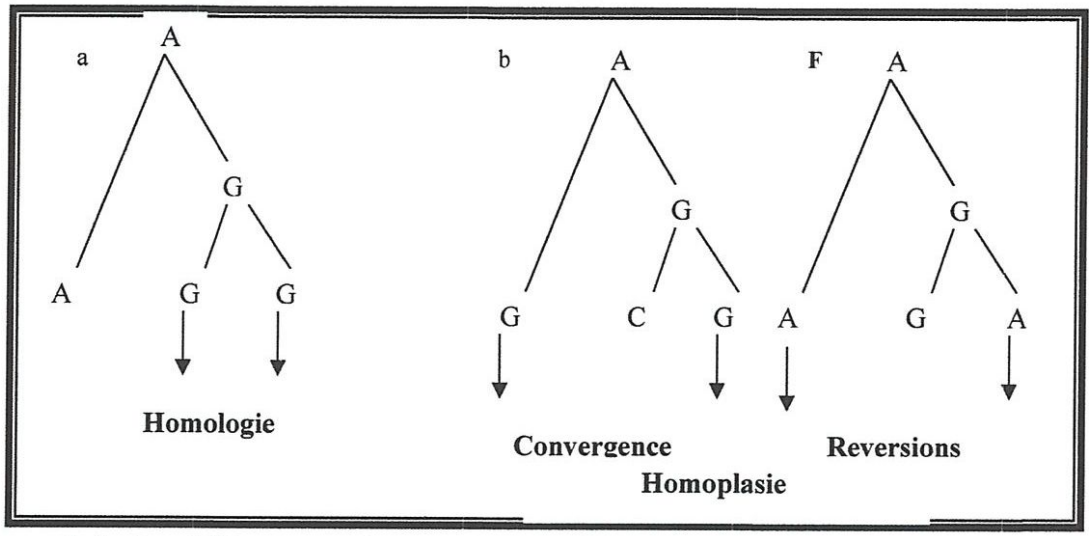


Figure 1.4 : *Les différentes catégories de ressemblances suite à l'évolution d'un caractère. (a): ressemblance due à l'homologie, (b,c): ressemblance due à l'homoplasie suite à une convergence ou à une réversion.*

- **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 [6]. Cependant, L'inconvénient majeur de cette méthode est la sensibilité de la méthode à des taux de mutations différents sur les différentes branches.

- **La méthode NJ (Neighbor-Joining)**

Cette méthode développée par Saitou et Nei [7], 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.

1.3. 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 objective 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 ont été proposées. Nous allons présenter dans la suite les fonctions les plus utilisées dans les algorithmes d'alignement pour évaluer un alignement multiple.

1.3.1. 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.

$$SP = \sum_{i=1}^{n-1} \sum_{j=i}^{n} sc(S_i, S_j) \qquad (1.1)$$

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.

$$\text{sc(Si, Sj)} = \sum_{i=1}^{m} sc(a_i, b_j) - \textit{Pénalité (GAPS)} \qquad (1.2)$$

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

$$\textit{Pénalité (gaps)} = GOP + Ne * GEP \qquad (1.3)$$

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. 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:

$$\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} Wij.score(Si, Sj) \qquad (1.4)$$

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 [8].

1.3.2. La mesure de l'entropie

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

$$Entropie(i) = -\sum_{a} c_{ia} \log(p_{ia}) \qquad (1.5)$$

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*.

$$p_{ia} = c_{ia} / \sum_{a'} c_{ia'} \qquad (1.6)$$

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.

1.3.3. Le score Consensus

Soit A[:][i] une colonne quelconque d'un alignement multiple *A*. La lettre x_i dénote l'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:

$$c(A) = \sum_{i=1}^{l} \sum_{j=1}^{k} d(x_i, A[j][i]) \qquad (1.7)$$

La figure suivante (*Figure 1.5*) 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 1.5 : *Le calcul de score Consensus d'un alignement.*

.

Dans la séquence consensus, X dénote une lettre quelconque.

1.3.4. 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* à la colonne du profile. L'exemple de la figure suivante (*Figure 1.6*) 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 1.6 : *Le calcul de score profil d'un alignement.*

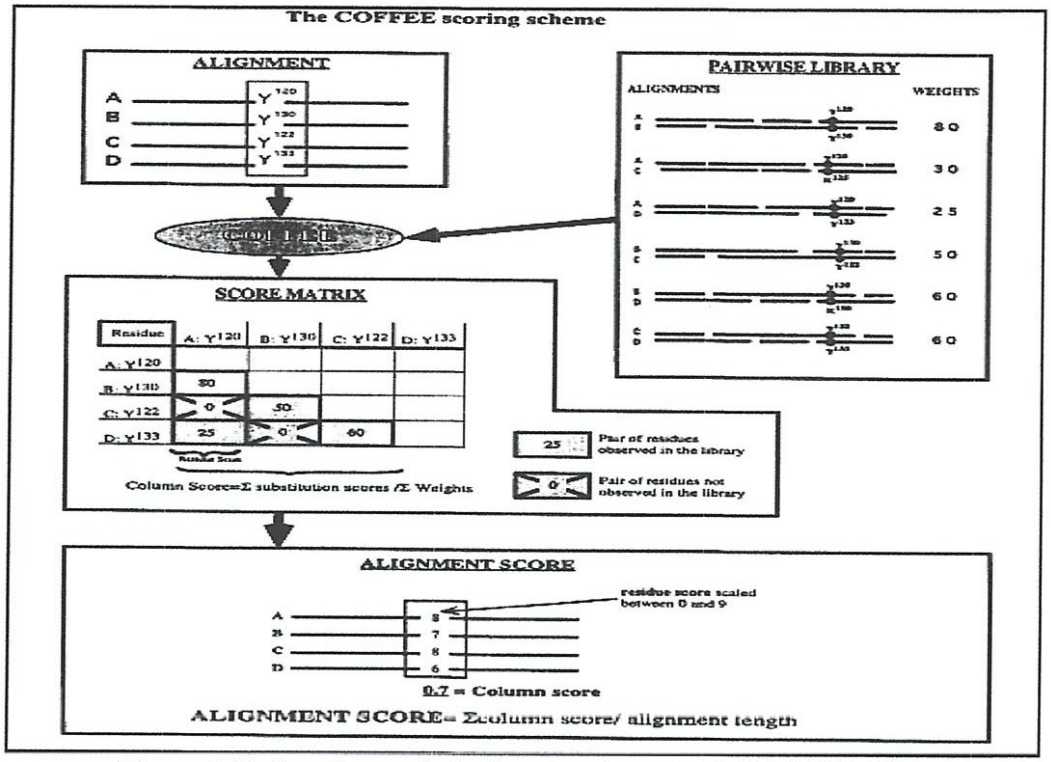


***Figure 1.7** : Les étapes de la fonction de score T-COFFEE* **[9]**.

1.4. Les approches de résolution du problème MSA

Le problème MSA a été démontré NP-difficile. Pour cela, plusieurs méthodes qui utilisent des différentes stratégies ont 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 combine entre les méthodes globales est les méthodes locales comme les méthodes basées segment (le programme DIALIGN) [10].

1.4.1. Les méthodes exactes

On peut généraliser l'algorithme de la programmation dynamique de Needleman-Wunsh 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 1.8*), 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 1.9)* montre un exemple de table de score d'un alignement de trois séquences ainsi que le chemin optimal qui donne l'alignement idéal.

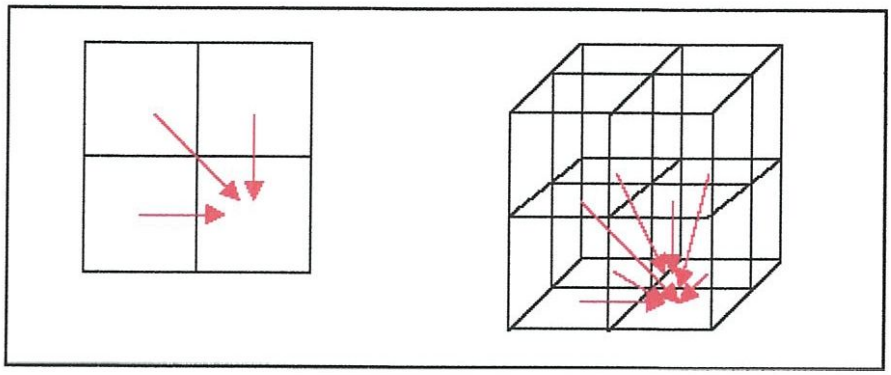


Figure 1.8 : *Tables de score à deux et trois dimensions.*

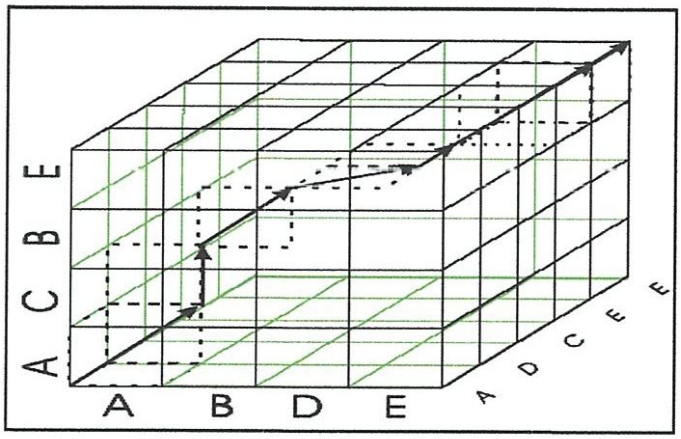


Figure 1.9 : *La trace back dans un alignement de trois séquences.*

Plusieurs heuristiques ont 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 [11]. Plus tard, plusieurs améliorations ont été apportées pour réduire la complexité temporelle et spatiale de MSA. L'idée principale de l'algorithme de Lipman est la suivante: premièrement, le score SP pour n'importe quelle 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 1.10)*, ceci peut réduire le temps de calcul considérablement. Les étapes principales de l'algorithme optimisé de MSA peuvent être trouvées dans [12].

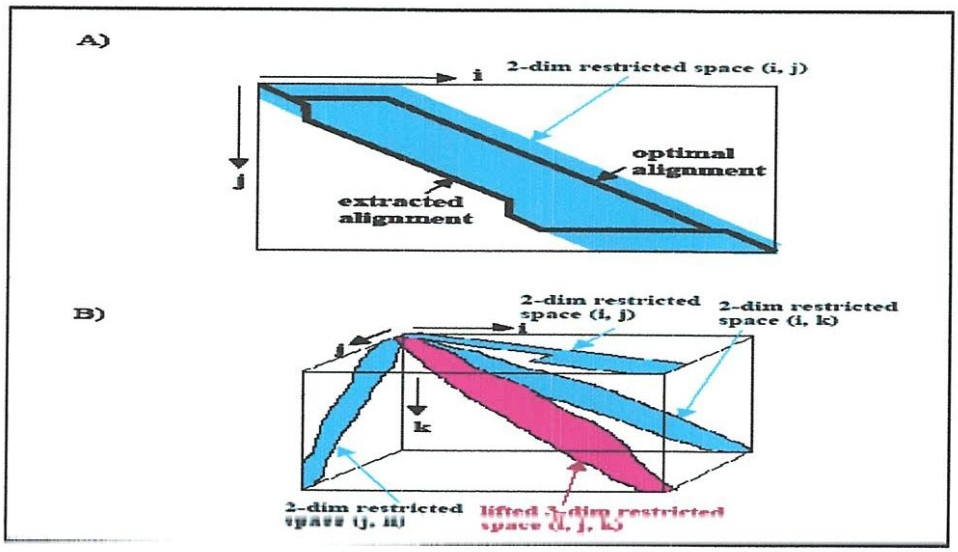


Figure 1.10 : *Un exemple de l'utilisation de l'espace restreint dans le programme MSA.*

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. Récemment, Une autre heuristique qui a maintenu les méthodes exactes encore populaires, est l'algorithme DCA décrit par Stoye. DCA est un algorithme basé sur l'idée "*divide to conquer*" (*Figure 1.11*). 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.

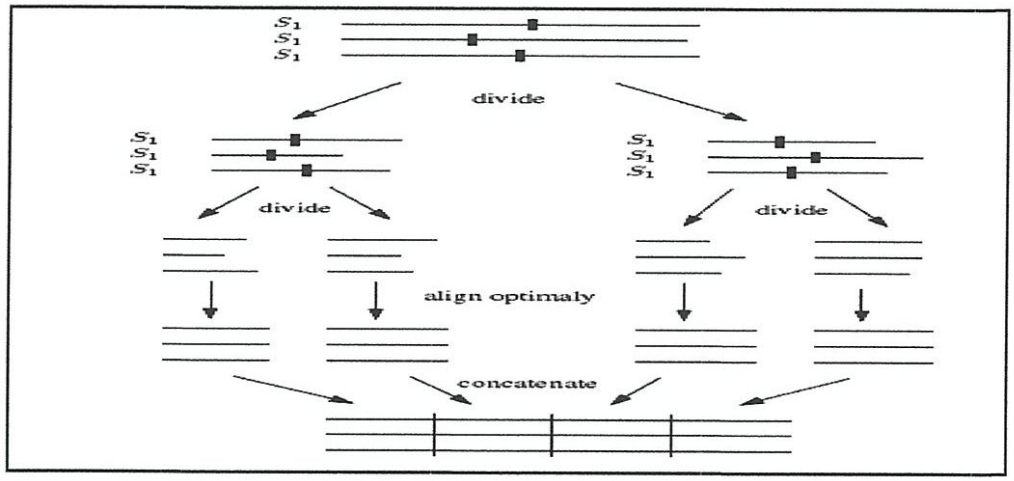


Figure 1.11 : *Les étapes de l'algorithme DCA.*

1.4.2. 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 [13] et plus tard redéfini par Feng [14]. Les méthodes d'alignement multiples les plus employés couramment sont basés sur l'exécution de cet algorithme comme la méthode CLUSTAL W [15] 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. 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 (*Tableau 1.1*). MAFFT utilise un nouvel algorithme pour l'alignement de paires de séquences basé sur la transformation de Fourier.

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	$o(N^3L)$

***Tableau 1.1** : La complexité de quelques programmes progressifs.*

- **Clustal : Le meilleur programme progressif pour MSA**

CLUSTALW [8] 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 le 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 de la 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 1.12*, *Figure 1.13*):

1) Alignement de pair 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 1.13*), 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 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.

- 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 tels que les pénalités des gaps et la matrice de substitutions utilisée.

1.4.3. Les méthodes itératives

Les méthodes itératives ont été introduites pour corriger les problèmes des méthodes progressives. Le principe est simple : une solution initiale est obtenue ensuite un ensemble de raffinements est appliqué sur cette solution. Les nouvelles solutions sont évaluées en utilisant une fonction objective. 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 : déterministes et stochastiques.[2]

1.5. L'évaluation des alignements

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

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

Une des méthodes 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

[2] Dans notre travail, on ne s'intéresse qu'aux méthodes progressives

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 [16], PREFAB [17] , etc.

1.5.2. Comparaison de programmes

Thompson et al [8] 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 créé une base d'alignements de références BALIBASE (*Tableau 1.2*). 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. 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.

1.5.3. La base de référence BALiBASE

Pour déterminer l'efficacité des méthodes, BALiBASE a été créée. Cette base contient 142 alignements de 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. 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 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.

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		

Tableau 1.2 : *Le contenu de la base BALiBASE.*

1.5.4. L'évaluation des alignements par BALIBASE

Pour évaluer la qualité d'alignement, Thompson et al ont créé 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 :

$$SPS = \frac{\sum_{i=1}^{l} S_i}{\sum_{i=1}^{l_r} S_{ri}} \qquad (1.8)$$

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

$$S_i = \sum_{j=1}^{N} \sum_{K=1}^{N} P_{ijk} \qquad (1.9)$$

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 :

$$CS = \frac{\sum_{i=1}^{l} C_i}{l} \qquad (1.10)$$

1.5.5. 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 (*Tableau 1.3*). 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 [8].

1.5.6. 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 [8].

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	-	-

Tableau 1.3 : *BALiBASE score des méthodes utilisées dans l'étude* **[16]**

Finalement, on a créé une table (*Tableau 1.4*) qui résume les méthodes récentes et moins récentes utilisées dans le MSA. Dans cette table, les auteurs ont mentionné pour chaque méthode sa classe, la fonction utilisée comme fonction objectif et les avantages et les inconvénients. [18]

Méthode	Classe	Fonction	Avantage	Problème
MSA	Exacte.	Maximiser la somme des paires SP.	La méthode la plus optimale.	Non praticable pour un grand nombre de séquences.
DCA	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)	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	Progressive/ Globale.	T-COFFEE.	• Corrige les erreurs des méthodes progressives. • Plus d'information sur les séquences. • Plus précise que	La méthode progressive la plus lente.

			CLUSTAL.	
PROBCONS	HMM/ Progressive/ Globale.	la transformation probabiliste consistante.	La méthode la plus précise.	Très lente par rapport aux méthodes progressives
MAFFT	Progressive/ Globale	Transformation de Fourier FFT	La méthode la plus rapide.	Moins précise que T-COFFEE.
PRRP [	Itérative/Déterministe/Globale.	WSP	Plus optimal que les méthodes progressives.	Lente.
SAGA	Iterative/ Stochastique/ Global. (AG)	WSP, T_COFFEE…	Le meilleur algorithme basé AG.	• Très lente. • Alignements trouvés ne sont pas forcément optimaux.
HMMT	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]	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.

Tableau 1.4 : *Une table récapitulative des différentes méthodes d'alignement multiple de séquences.*

1.6. 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 réussi largement dans ce domaine.

Comme on a dit dans l'introduction générale nous allons utiliser une nouvelle approche pour traiter le problème de MSA.

2. Notre approche

2.1. Introduction

Toutes les techniques utilisées pour résoudre le problème de MSA sont gourmandes en temps de calcul. Pour résoudre ce problème, deux classes de techniques ont été développées. La première est basée sur l'utilisation du parallélisme grâce à une approche matérielle. Certaines méthodes utilisent des ordinateurs à mémoire partagée et à mémoire distribuée, par ex. ClustalW-MPI [19] et Parallel T-Coffee [20]. Récemment, Church PC [21] a présenté une conception d'algorithmes d'alignement de séquences multiples sur des supercalculateurs à mémoire parallèle et distribuée. D'autre part, les unités de traitement graphique sont utilisées pour accélérer les programmes MSA, ex. GPU-Blast [22], G-MSA [23].

La deuxième classe consiste en l'utilisation du parallélisme par une approche logicielle en utilisant des modèles de programmation parallèles. Dans cette section, quatre techniques ont été proposés:

La première est une approche parallèle dans le calcul de la matrice de score. Elle consiste à calculer la matrice de score en parallèle dans le cas des parties non liées, Zafalon G [24] a montré que sa technique peut apporter une amélioration de 15% en temps d'exécution.

La seconde est une approche Pineline; Agarwal P [25] a proposé une technique avec un pipeline à deux étapes qui peut améliorer la complexité du problème. Ensuite, un nouveau pipeline multi-alignement pour les données de séquençage à haut débit est présenté par Shunping H [26].

Le troisième est une approche parallèle avec un algorithme dynamique. Ces techniques sont basées sur la parallélisation des algorithmes optimisés connus dans le domaine, par exemple, la parallélisation de l'algorithme Needleman & Wunsch par Naveed T [27], la parallélisation de l'algorithme Smith & Waterman de Dohi et al et la parallélisation de La technique optimale pour résoudre le MSA par Manal Helal et al sur l'architecture GPU.

Le quatrième est une approche parallèle des données ;. Cette technique a été proposée par Fahad S & al [28] dans laquelle ils ont proposé la technique k-mer pour faire le regroupement.

Plusieurs techniques de regroupement ont été proposées, telles que Uclort [29], CD-Hit [30], BlastClust [31], etc. Cela a permis de créer plusieurs approches pour MSA telles que Xiangyuan Z & al [32] qui proposait également une approche parallèle basée sur les deux systèmes de clustering UClust et CD-HIT dans l'étape de cluster et MUSCL dans l'étape d'alignement.

2.2. Détail de l'approche

Notre approche comporte quatre étapes principales: Clustering des séquences, alignement de chaque sous-ensemble et génération d'une séquence de consensus pour chacun, trier le consensus par leur longueur et les diviser en sous-groupes en fonction du nombre de noyaux

de processeur et finalement aligner tous les consens générés sur différents processeurs avec notre algorithme progressif pour générer l'alignement complet. Le résumé des différentes étapes de notre approche est illustré à la *Figure 2.1*

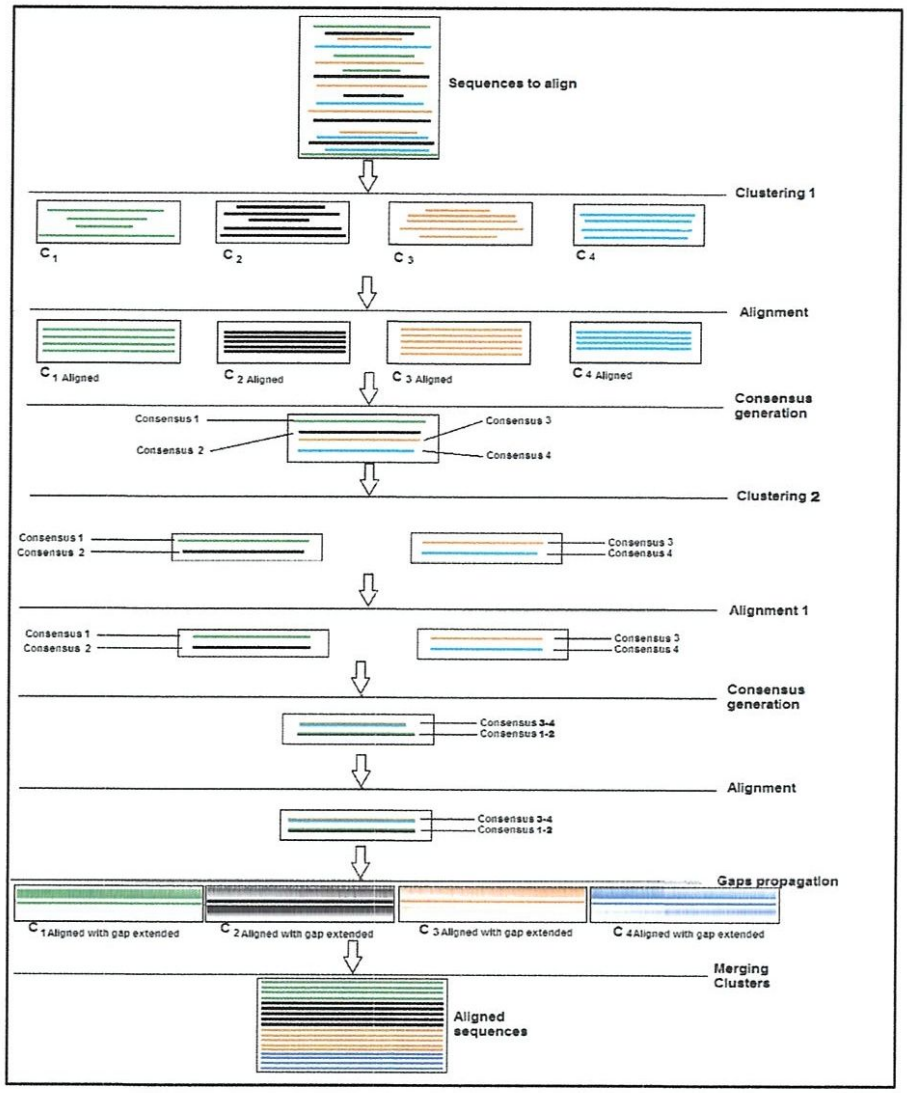


Figure 2.1 : *Résumé des différentes étapes de notre approche.*

2.3. Clustering des séquences

Dans notre étude, nous avons choisi UCLUST parce que c'est un algorithme rapide comparé aux autres. En fait, le regroupement d'un ensemble avec d'autres méthodes nécessiterait des ressources informatiques à grande échelle, tandis que UCLUST est capable de générer des regroupements de haute qualité et peut réduire de manière spectaculaire les ressources requises pour la classification de grands ensembles de séquences et sera donc Valeur pour les biologistes dans un large éventail d'applications [33].

2.4. Alignement des séquences et alignement des consens

Nous avons utilisé à la fois l'alignement local et l'alignement global pour aligner deux séquences de chaque cluster ainsi que les consensus. Les méthodes d'alignement global tentent d'aligner les séquences sur toutes leurs longueurs alors que les algorithmes locaux

recherchent uniquement les motifs les plus conservés [8] [6] L'utilisation d'algorithmes globaux et locaux permet d'éviter la faiblesse des méthodes globales dans les cas de séquences avec une grande extension N / C-terminal ou des insertions internes.

Notre méthode utilise l'algorithme Needleman-Wunsch pour l'alignement global et l'algorithme de Smith-Waterman pour l'alignement local [34]. Dans plus de détails, l'approche proposée peut être décrite comme dans l'Algorithme 1

Algorithm 1. HClustMSA

```
Input : Set of sequences {S1,S2,…,Sn}
Begin
  1-Divide sequences in Clusters using UCLUST Algorithm
  2-Align sequences in each cluster using hybMultiAlign
  3-Generate the consensus for each aligned cluster
  4-Divide the consensus into subsets according to the number of CPU cores
  as well as their length
  5-In parallel, align subsets generated in 4 using hybMultiAlign
  6-Propagate gaps into subsets then into Clusters
  7-Merge Clusters
End
Output: Multiple sequence alignment
```

Algorithm 2. hybMultiAlign

```
Input: Set of sequences {S1,S2,…,Sn}
Begin
  1-Construct the guide tree
  2-Choose the two closest sequences Si and Sj from the guide tree
  3-Compute the difference between the sequence lengths
  'diff=length1/length2' (length1<length2):
  4-If diff<k then
    Use a local pairwise algorithm to align the sequences Si and Sj,
    Else
    Use a global pairwise algorithm to align the sequences Si and Sj,
  5-Propagate gaps
  6-Choose the next unaligned sequence Si and the closest aligned sequence
  Sj to Si, GoTo 3.
End.
```

Output: Multiple sequence alignment

2.5. Temps d'exécution et évolutivité

Pour mesurer le gain fourni par notre approche qui utilise une étape de clustering avant l'alignement, nous avons effectué un ensemble de tests basé sur de grands ensembles de données contenant des ensembles de séquences de référence générés par GenRGenS [35] sur des profils de séquences réelles dérivées de BAliBASE dans lesquelles la longueur d'une séquence varie de 500 à 2000 et le nombre de clusters est fixé à 50 Clusters. Le nombre de séquences et leurs longueurs ont un effet direct sur le temps d'exécution de MSA.

Afin de montrer ce gain en termes de temps d'exécution, nous avons développé la même approche, avec et sans Clustering comme étape préliminaire avant l'alignement. Nous avons également comparé notre solution avec celles les plus utilisées dans le cas de grands ensembles de données.

L'approche proposée est implémentée dans MATLAB R2014b et nous avons effectué ces ensembles de tests sur un Intel Xeon ES2620 avec 6 cœurs, fonctionnant à 2,00 GHz, avec Caches (L1D- cache 32 Ko, L1I- cache 32Ko, L2- cache 256 Ko, L3-Cache 15 Mo) avec une mémoire DDR3 de 16 Go.

Figure 2.2 montre la différence entre le temps d'exécution entre la même technique avec et sans utilisation du cluster et la *Figure 2.3 : Comparaison du temps d'execution.*montre la différence entre le temps d'exécution entre notre technique et les plus utilisés.

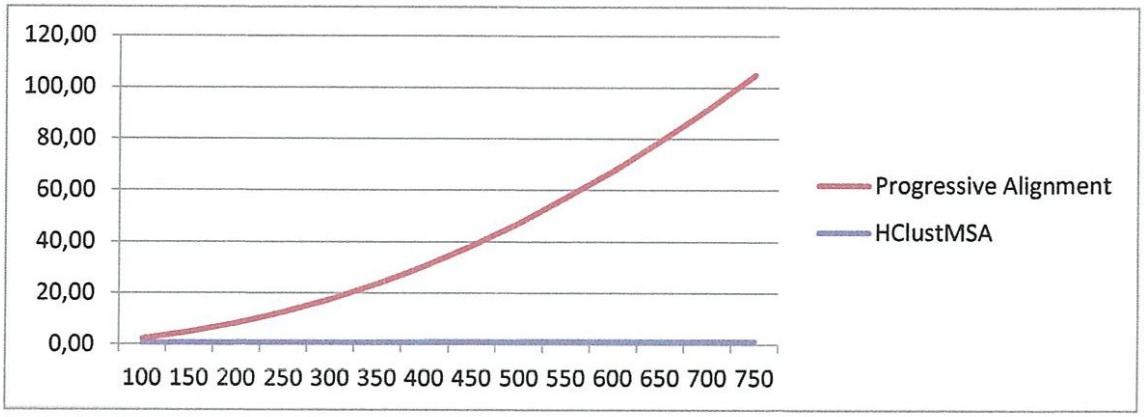
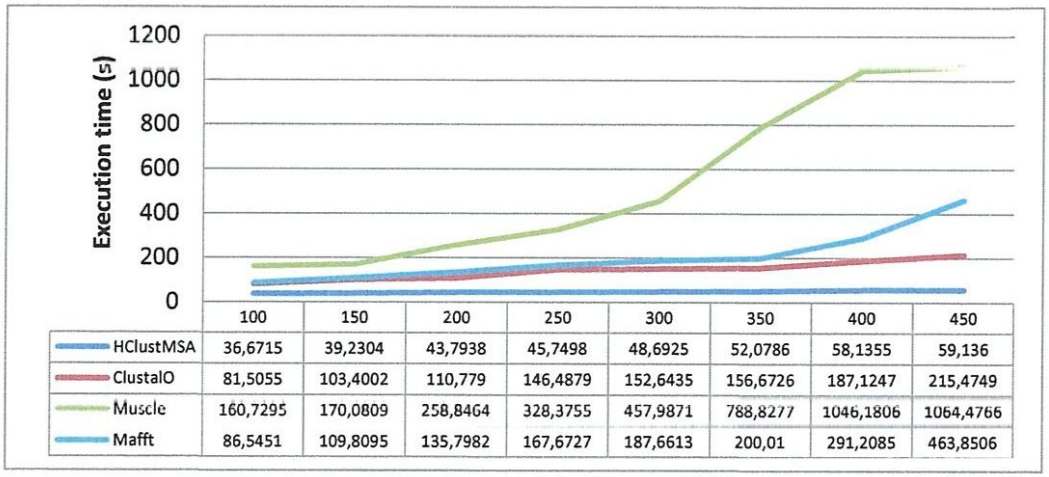


Figure 2.2 : *La différence du temps d'exécution dans le cas de l'utilisation et la no-utilisation du parallélisme*



	100	150	200	250	300	350	400	450
HClustMSA	36,6715	39,2304	43,7938	45,7498	48,6925	52,0786	58,1355	59,136
ClustalO	81,5055	103,4002	110,779	146,4879	152,6435	156,6726	187,1247	215,4749
Muscle	160,7295	170,0809	258,8464	328,3755	457,9871	788,8277	1046,1806	1064,4766
Mafft	86,5451	109,8095	135,7982	167,6727	187,6613	200,01	291,2085	463,8506

Figure 2.3 : *Comparaison du temps d'execution.*

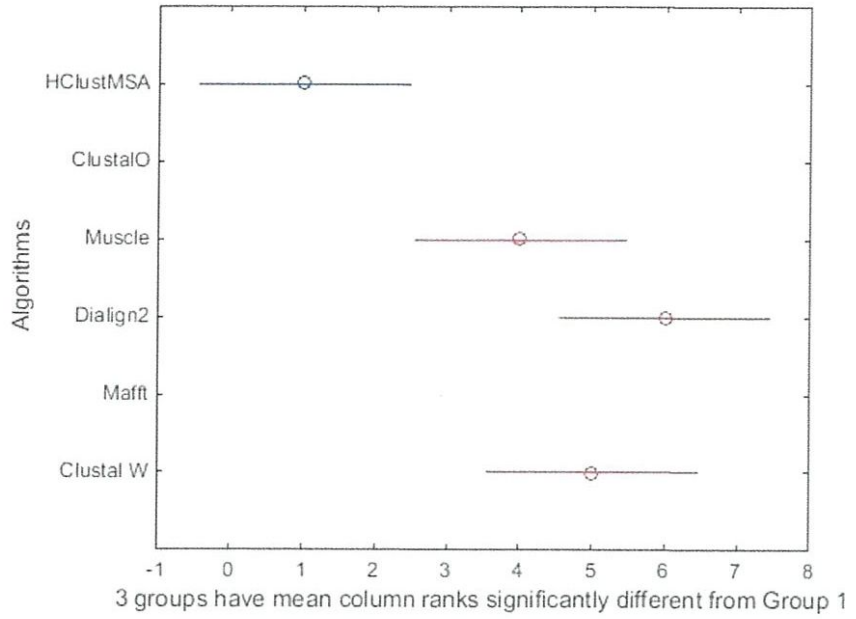


Figure 2.4 : Test de Friedman pour la comparaison des temps d'execution.

Les figures (*Figure 2.2*, *Figure 2.3*,*Figure 2.4 : Test de Friedman pour la comparaison des temps d'execution.*) montrent un gain significatif en termes de temps d'exécution. La première figure montre une grande différence de temps d'exécution si le Clustering est utilisé par rapport à sa non-utilisation. *Figure 2.4 : Test de Friedman pour la comparaison des temps d'execution.*montre que notre solution est très rapide en le comparant avec la plupart des solutions utilisées telles que Clustal Omega, Muscle et Mafft, dans le cas de données à grande échelle où la plus part des autres algorithmes échouent.

2.6. Qualité d'alignement

Pour étudier la qualité de l'alignement, nous avons fait deux comparaisons de notre approche. La première comparaison concerne un algorithme que nous avons développé avec les mêmes paramètres mais qui utilise uniquement l'alignement global ou local et n'utilise pas la technique hybride. Dans la seconde, nous comparons nos résultats avec ceux produits par les principales techniques d'alignement les plus utilisées.

La performance de notre approche a été testée sur une collection de Benchmarks de séquences de protéines comprenant BALIBASE v3, PREFAB v4, OXBENCH et SABER. Nous avons comparé nos résultats avec d'autres algorithmes bien connus tels que CLUSTALW, DIALIGN-TX v1.0.2, MAFFT v6.603 à l'aide d'EINSI, MUSCLE v4.0. [36] et Clustal Omega.

Pour mesurer la qualité de l'alignement, le programme qScore [37] est utilisé. Le programme affiche les scores suivants: Le score PREFAB Q (également connu sous le nom Balibase SPS score ou le Score du développeur) et le Balibase TC (colonne totale).

La suite de benchmark BAliBASE contient des alignements de séquences multiples, organisés en 9 ensembles de référence représentant des problèmes spécifiques de MSA, y compris un

petit nombre de séquences, des distributions phylogénétiques inégales, des extensions N / C-terminales ou des insertions internes, des répétitions, des domaines inversés et des régions transmembranaires [38].

Pour évaluer la performance des trois programmes: MSA avec un alignement global en utilisant clustering, MSA avec un alignement local en utilisant clustering et HCLustMSA, nous avons utilisé plusieurs tests à partir de la base de données d'alignement de référence (BaliBase BB40) contenant de grandes extensions N-C-terminales et des insertions internes. Les résultats de cette expérience sont présentés dans la *Figure 2.5*. Il montre évidemment l'efficacité de la fusion des techniques (globales et locales) pour effectuer l'alignement progressif des séquences multiples.

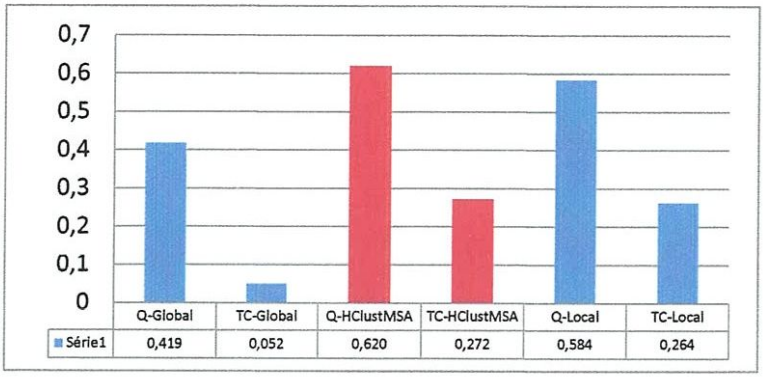


Figure 2.5 : *Comparaison des performances dans le cas des extensions N/C (GlobalClustMSA VS HClustMSA VS Local ClustMSA) sur BB40 de BAliBASE 3.0.*

Nous avons aussi analysé la performance de notre approche en comparant les alignements produits avec ceux obtenus grâce à d'autres techniques d'alignement. Les résultats sont présentés dans le *Tableau 2.1* et le *Tableau 2.2*. En termes de qualité d'alignement, on a observé que notre technique donne des résultats de qualité inférieure par rapport aux autres; Il perd une moyenne de TC de 0,14 dans tous les critères de comparaison testés par rapport à Clustal Omega qui est le programme le plus utilisé dans le cas d'ensembles de données à grande échelle

	BaliBase V3	Bali3 pdb	Bali3 pdbM	Sabre	Sabrem	Prefab4	Oxm
HClustMSA	0,715 /0,367	0,609 /0,422	0,804 /0,658	0,446 /0,264	0,559 /0,362	0,537 /0,537	0,988 /0,712
Clustal Omega	0,840 /0,562	0,511/ 0,589	0,569 0,759	**0,550 /0,355**	0,623 /0,451	**0,670 / 0,670**	0,895 / 0,891
Clustal W	0,78 /0,43	0,671 /0,48	0,786 /0,596	0,315 /0,151	0,59 /0,4	0,586 /0,586	**1** /0,875
Dialign	0,786 /0,456	0,734 /0,546	0,824 /0,696	0,299 /0,123	0,568 /0,369	0,517 /0,517	0,99 /0,738
Mafft	0,872 /0,602	0,796 /0,606	0,832 /0,633	0,377 /0,182	0,646 /0,456	0,569 /0,569	0,98 /0,787
Muscle 4	**0,887 /0,633**	**0,829 /0,666**	**0,944 /0,879**	0,414 /0,215	**0,676 /0,488**	0,61 /0,61	0,992 **/0,893**

Tableau 2.1 : *Comparaison des performances sur les différents Bechmarks.*

	BB 11	BB 12	BB 20	BB 30	BB 40	BB 50	BBS 11	BBS 12	BBS 20	BBS 30	BBS 50	Average
HClustMSA Q/TC	0,435 /0,219	0,825 /0,648	0,801 /0,228	0,611 /0,172	0,620 /0,272	0,615 /0,262	0,645 /0,407	0,882 /0,733	0,898 /0,410	0,802 /0,385	0,729 /0,300	0,715 /0,367
Clustal Omega	0,590 / 0,362	0,906 / 0,794	0,912 /0,453	0,863 /0,579	0,901 /0,583	0,862 /0,537	0,626 /0,393	0,904 /0,784	0,937 /0,495	0,866 /0,601	0,853 /0,495	0,840 /0,562
Clustal W Q/TC	0,501 /0,23	0,865 /0,717	0,852 /0,222	0,725 /0,276	0,789 /0,398	0,742 /0,312	0,664 /0,421	0,903 /0,795	0,924 /0,456	0,818 /0,488	0,798 /0,422	0,78 /0,431
Dialign Q/TC	0,505 /0,268	0,882 /0,757	0,878 /0,308	0,761 /0,389	0,834 /0,452	0,822 /0,471	0,576 /0,366	0,884 /0,763	0,902 /0,386	0,775 /0,409	0,822 /0,442	0,786 /0,456
Mafft Q/TC	0,66 /0,44	0,936 /0,839	0,926 /0,452	0,861 /0,592	0,914 /0,575	0,899 /0,599	0,724 /0,522	0,938 /0,847	0,945 /0,525	0,889 /0,634	0,903 /0,6	0,872 /0,602
Muscle 4 Q/TC	0,683 /0,441	0,945 /0,862	0,928 /0,479	0,875 /0,623	0,925 /0,604	0,894 /0,593	0,792 /0,591	0,951 /0,876	0,96 /0,635	0,894 /0,648	0,904 /0,611	0,886 /0,633

Tableau 2.2 : *Comparaison des performances sur BAliBase 3.0*

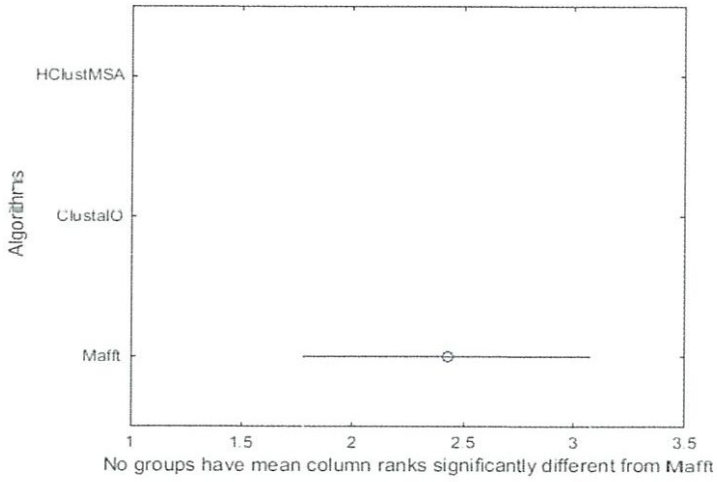


Figure 2.6 : *Test Friedman (Q Score).*

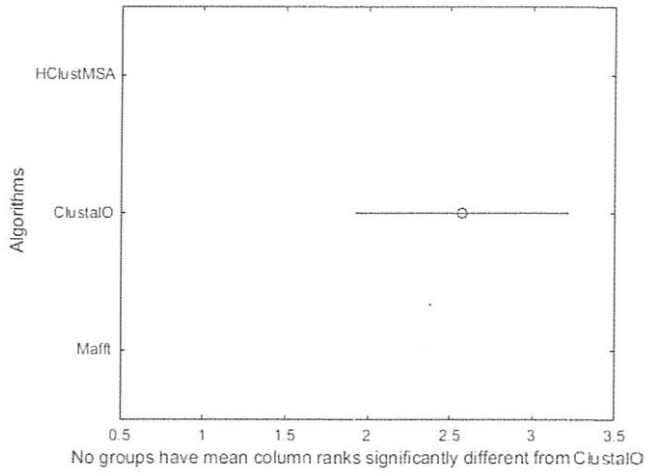


Figure 2.7 : *Test Friedman (TC Score).*

2.7. Discussions

Selon la *Figure 2.2*, *Figure 2.3 : Comparaison du temps d'execution.*et le diagramme de Friedman dans la *Figure 2.4*, les algorithmes Dialign2, Clustal W et Muscle échouent dans le cas de grandes données. Le temps de calcul pour les autres algorithmes (HClustMSA, Clustal Omega et Mafft) est acceptable.

En termes de qualité d'alignement et selon le et le *Tableau 2.2* et le test de Friedman dans la *Figure 2.6* et *Figure 2.7*, et après l'élimination des autres algorithmes qui échouent dans le cas des grandes données, et après avoir comparé notre approche avec les autres (Clustal Omega et Mafft), on constate que malgré la petite perte de qualité d'alignement, notre solution reste toujours dans le même groupe que les autres algorithmes les plus utilisés dans le domaine.

3. Implementation

3.1. Introduction

Dans ce chapitre, nous allons présenter l'implémentation de notre application d'alignement multiple de séquences. Notre application fait l'alignement multiple et calcule la qualité de l'alignement en utilisant un fichier de référence.

3.2. Objectif de notre application

L'objectif principale de notre projet est de construire une application qui permet d'aligner un ensemble de séquences biologique ADN ou protéine en essayant d'avoir un bon compromis Temps /qualité d'alignement. Les différents objectifs visés par un alignement multiple de séquences sont:

- Trouver les parties homologues
- Identification de résidus importants (conservés)
- Extraction de motifs communs
- Génération de séquences consensus

3.3. Présentation de langage de programmation

3.3.1 Matlab

Nous avons choisi l'environnement de développement Matlab R2014b qui est un langage de développement informatique particulièrement dédié aux applications scientifiques.

La plate-forme MATLAB est optimisée pour résoudre les problèmes scientifiques et techniques. Le langage MATLAB, basé sur les matrices, est un moyen simple pour exprimer les mathématiques computationnelles. Les graphiques intégrés permettent de visualiser facilement les données afin d'en dégager des informations. Grâce à sa bibliothèque et sa boîte à outils prédéfinie, ce qui encourage l'expérimentation, l'exploration et la découverte. Les outils et les fonctionnalités MATLAB sont tous testés rigoureusement. Ils sont conçus pour fonctionner conjointement.

3.3.2 Outils utilisés

Nous avons utilisés un ensemble d'outils tel que UClust, qscore :

- **Uclust :** c'est un outil d'analyse de séquences avec des milliers d'utilisateurs dans le monde entier. UCLust propose des algorithmes de recherche et de clustering [33].
- **Qscore :** c'est un programme qui compare deux alignements de séquences multiples: un alignement à évaluer et un deuxième alignement qui est censé être correct (l'alignement "référence") [37].

Afin d'évaluer notre solution, on l'a comparé avec les solutions des algorithmes les plus utilisés, donc, dans notre application, on a intégré les algorithmes suivants :Cluslatlw, ClustalO, Muscle, Dialigne, Maftt.

3.3.3 Format fasta

C'est un format qui permet de représenter un ou plusieurs séquences (nucléiques ou protéiques). Une ligne qui commence par le symbol '>' caractérise le début d'une nouvelle séquence. Le symbol '>' est suivi d'un identifiant de séquence et de commentaires éventuels. Les lignes suivantes constituent la séquence (jusqu'à ce qu'une nouvelle ligne commence par '>' ou la fin de fichier).

Exemple (*Figure 3.1*)

```
>1|chr12|64798729-64798930|354.27082|-1.0|1127
CTGGCTGGGCGGACCGGGTGGGGGTGGGTACGAGCCGGGGCCGCCGCCGAGGAGCGCGT
TTGGTGTTTCATCACCCGAATTGCCACGAGGCTTCCTTTAGGGGAGGGATCGGGGGAGG
GGGTCGGCATCGCCTGTGGTTCCGAAGCCCGTTAG
>2|chr12|57848784-57848985|336.02993|-1.0|635
CCACCTGGCTCATAAGGCGTTCCCTCCCCCAAGTCCCAGACCTTGGGGACTGAGCATGT
TGCGTGTCCACATTGCACCCCCCCACCCCCATACCCCTACTTCAGGCCCAGTCACCATG
TGGGGAGGAGGACCTCCACCCCCTGCAGGGGCCTG
```

Figure 3.1 : *Format fasta.*

C'est l'un des standards les plus utilisés en bioinformatique et la plupart des logiciels reconnaissent ce format.

3.4. Interface

La figure suivante (*Figure 3.2*) présente l'interface principale de notre application.

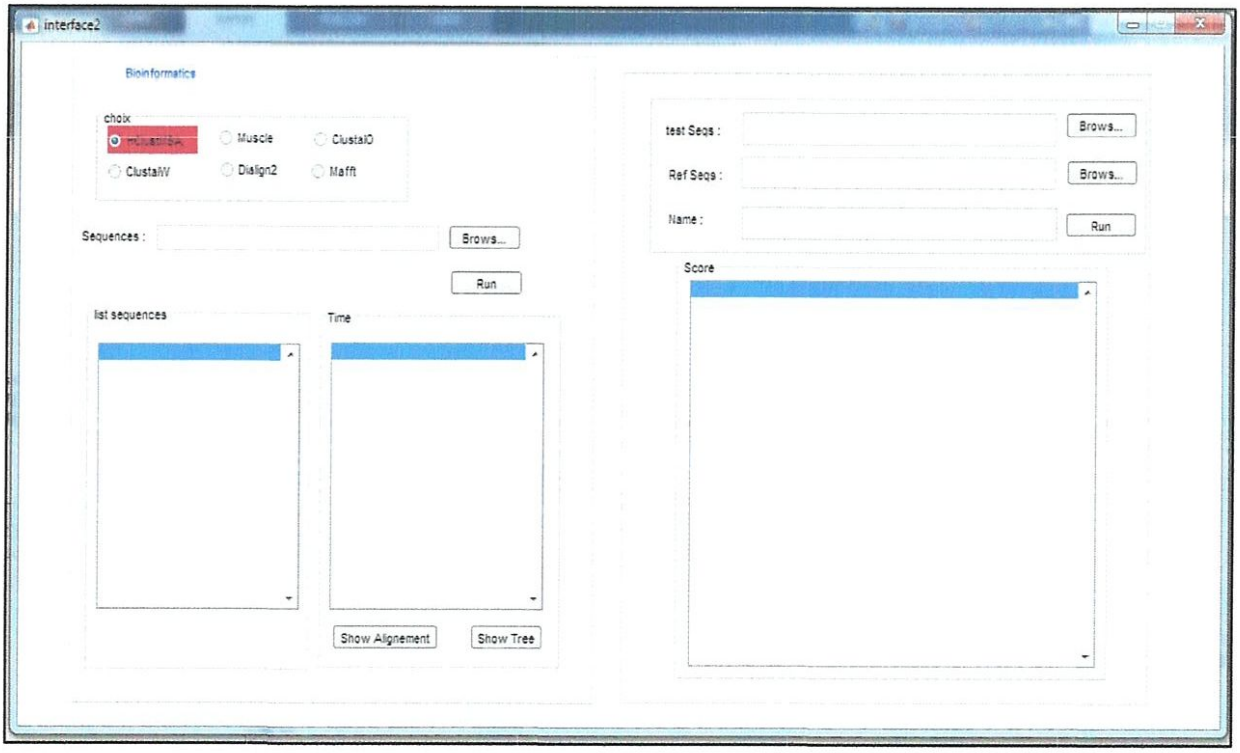


Figure 3.2 : *Interface Principale.*

3.4.1. L'alignement multiple de séquences

Avant de faire un MSA, on doit choisir l'algorithme d'alignement, coloré en rouge, HClustMSA est notre algorithme, sélectionné par défaut.

Lorsqu'on clique sur le bouton Brows, on peut parcourir les bases des séquences et choisir le fichier FASTA contenant les séquences à aligner. Les figures suivantes (*Figure 3.3*, *Figure 3.4*) affichent un exemple des séquences choisies par l'utilisateur.

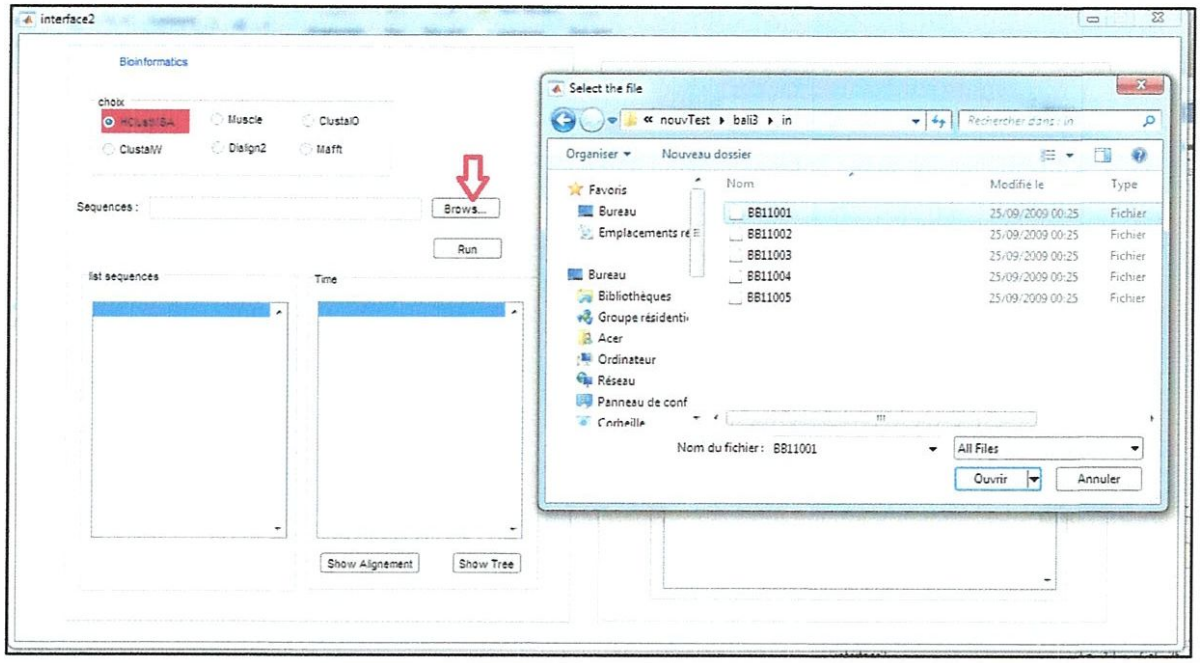


Figure 3.3 : *Sélectionner fichier de séquences.*

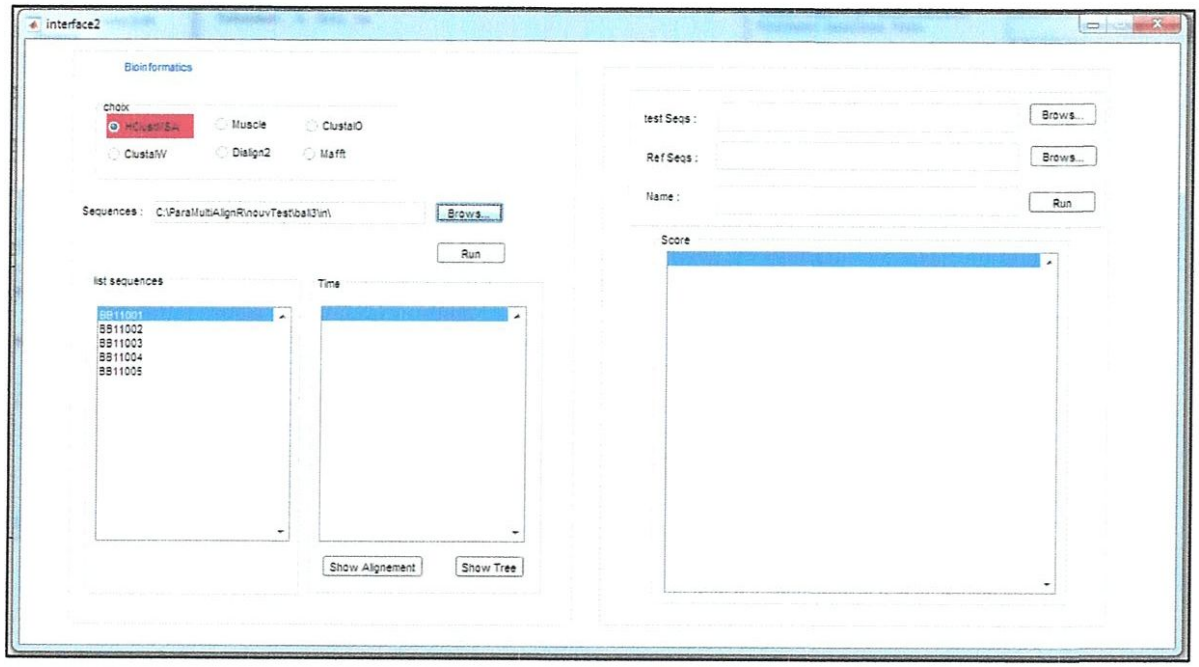


Figure 3.4 : *Fichier Sélectionner.*

Après on clique sur le bouton Run pour lancer l'alignement (*Figure 3.5*).

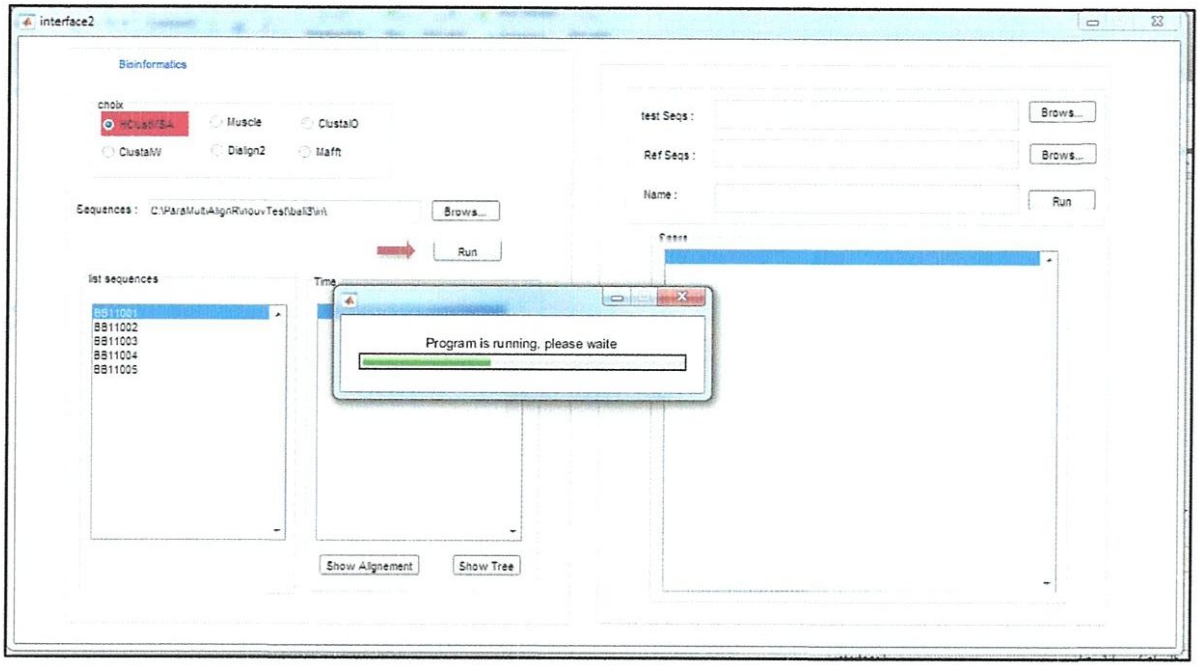


Figure 3.5 : *Fichiers de Séquences en cours de l'alignement.*

Une fois l'alignement est terminé le temps de l'alignement multiple de chaque fichier est affiché et enregistré dans un fichier csv (time.csv) (*Figure 3.6*, *Figure 3.7*).

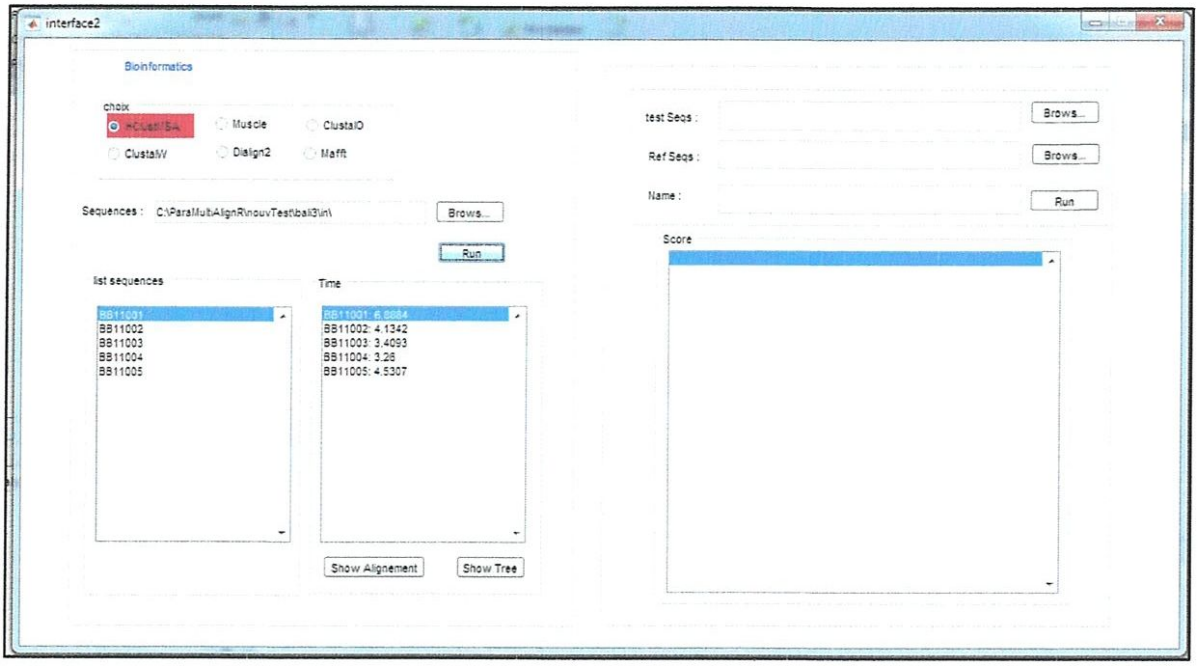


Figure 3.6 : *L'alignement est terminé.*

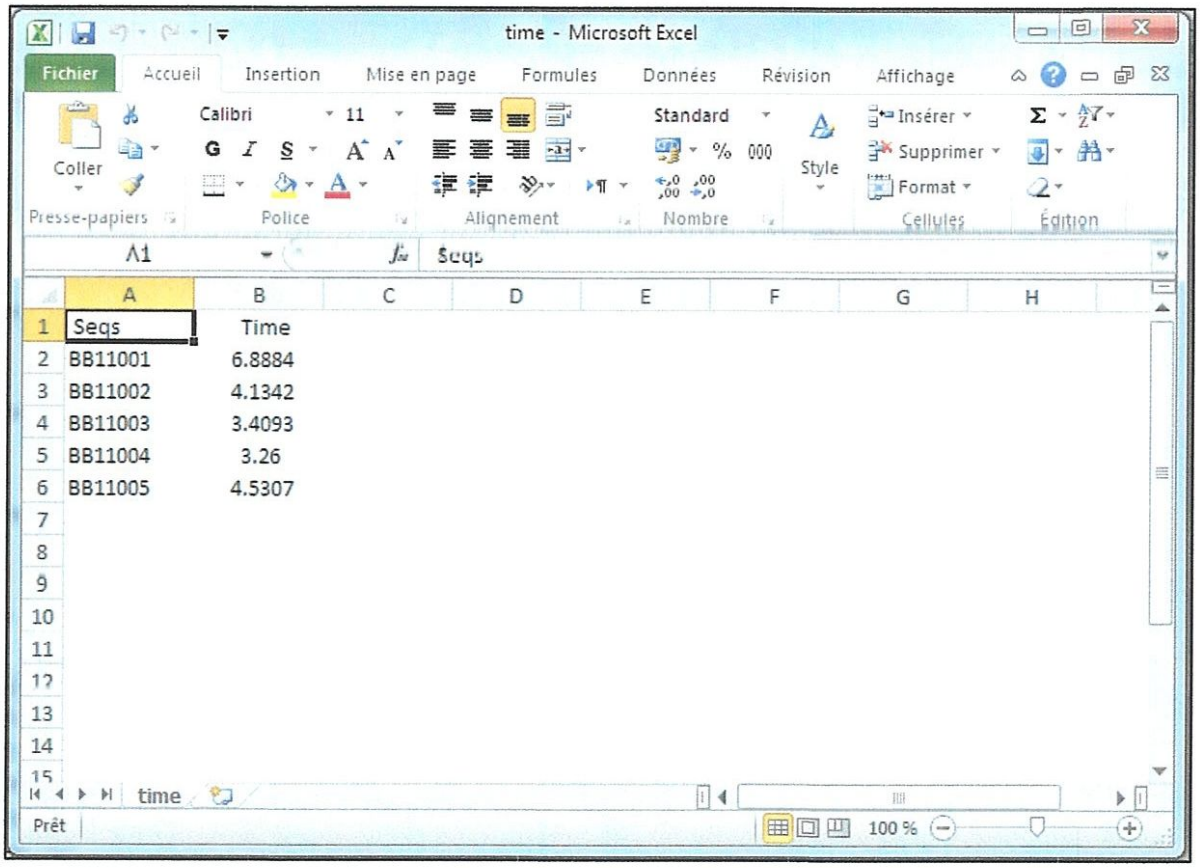


Figure 3.7 : *Temps d'alignement multiple de séquences.*

Pour afficher les séquences alignées, on choisit le fichier contenant les séquences déjà alignées et on clique sur le bouton Show Alignement (*Figure 3.8*).

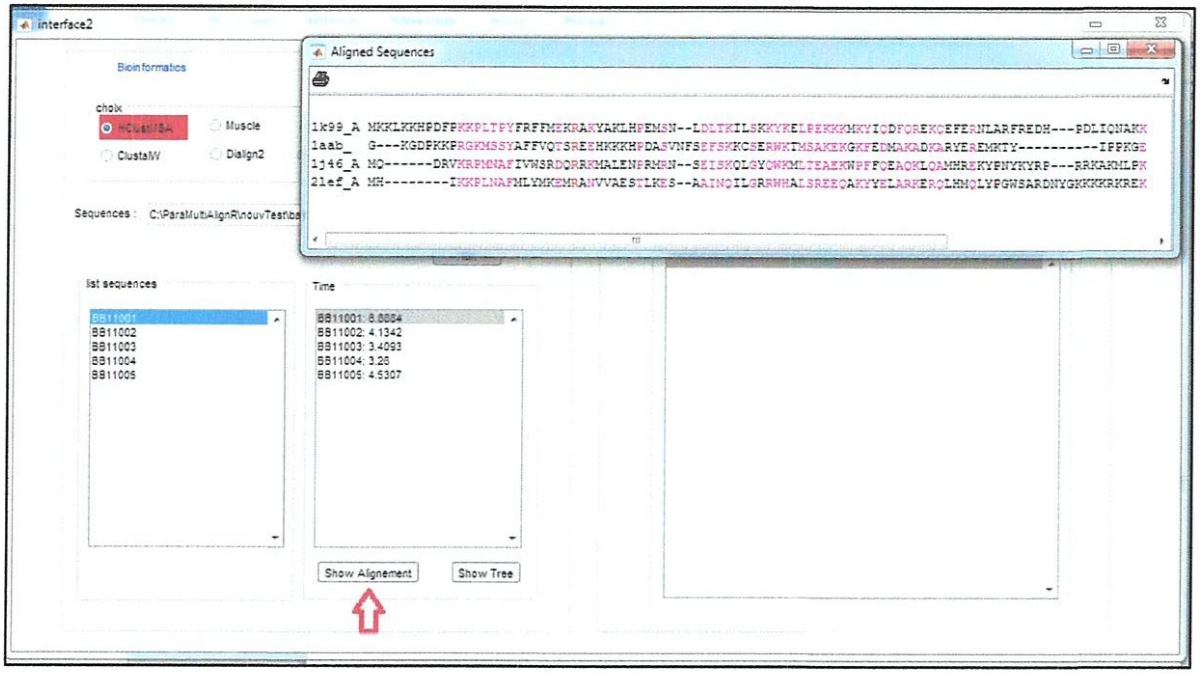


Figure 3.8 : *Séquences alignés.*

3.4.2. La qualité d'alignement multiple de séquences

Le bouton Brows, permet de parcourir les bases des séquences alignées par notre application puis on choisit n'importe quelle base pour la comparer avec sa base de références et ainsi calculer la qualité d'alignement. La figure suivante (*Figure 3.9*) affiche un exemple de fichier séquences(Fasta) choisi.

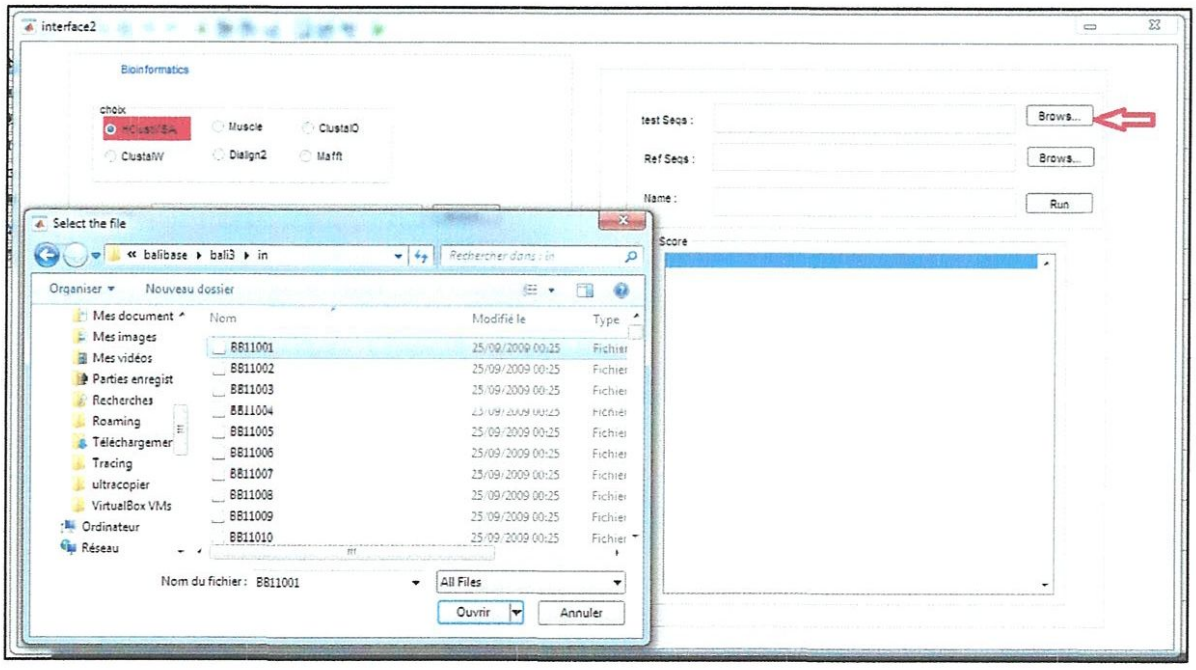


Figure 3.9 : *Fichiers d'alignement test.*

Après on choisit les séquences de référence alignées par les biologistes (*Figure 3.10*) puis on clique sur le bouton Run (*Figure 3.11*).

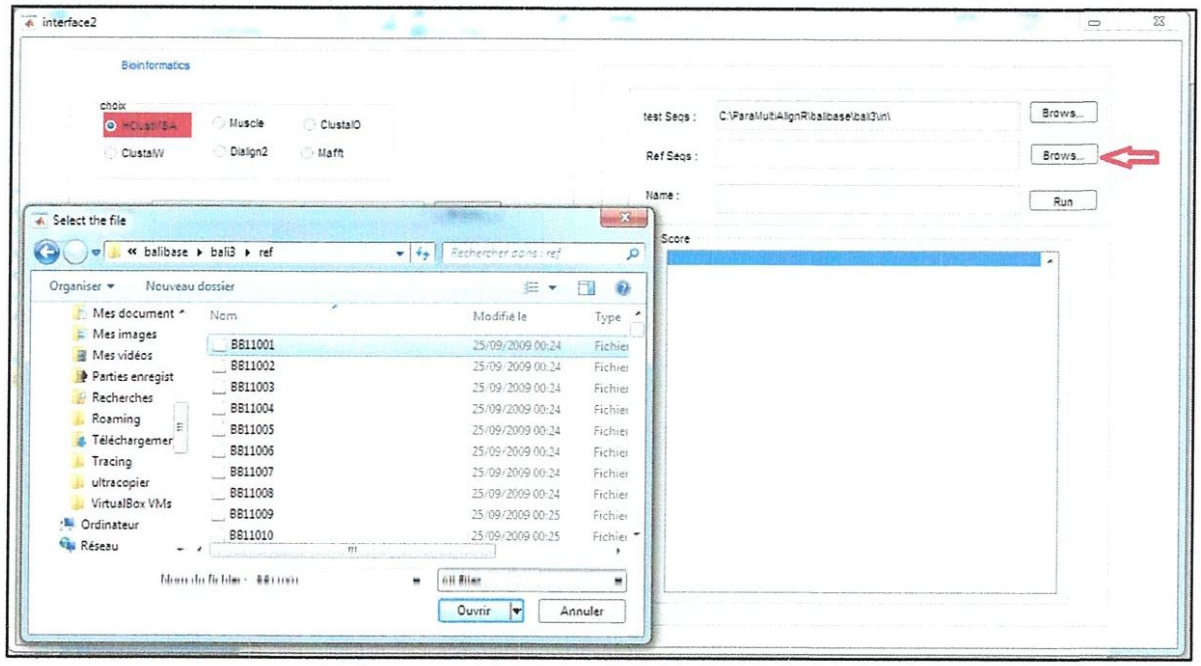


Figure 3.10 : *Fichier d'alignement référence.*

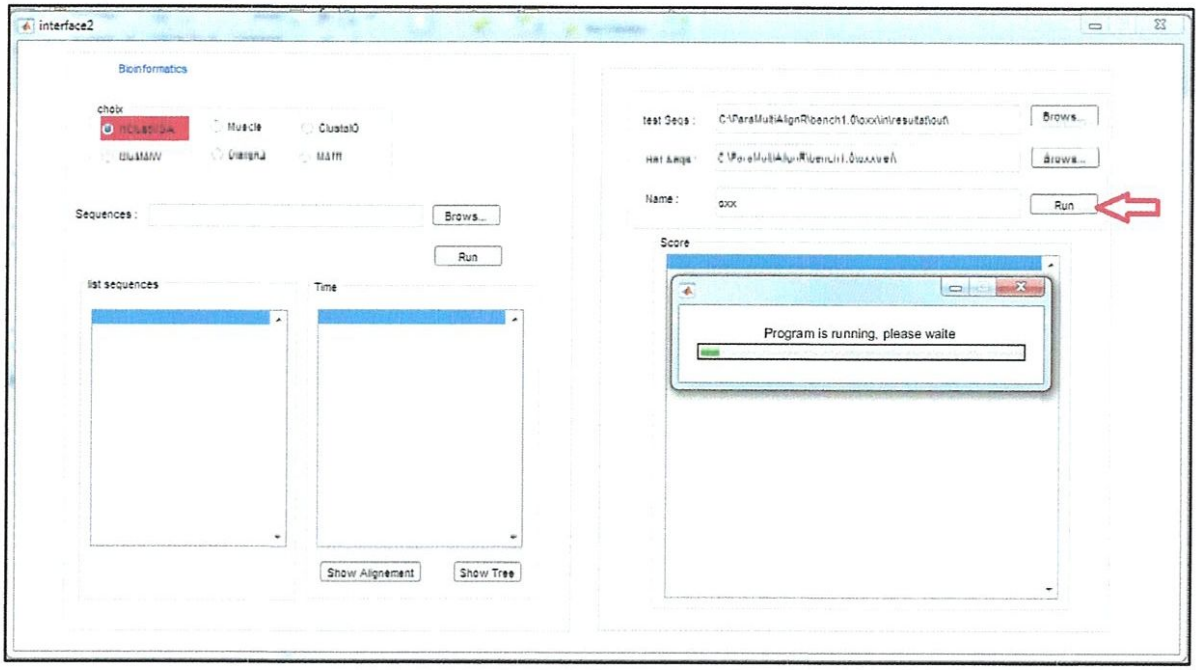
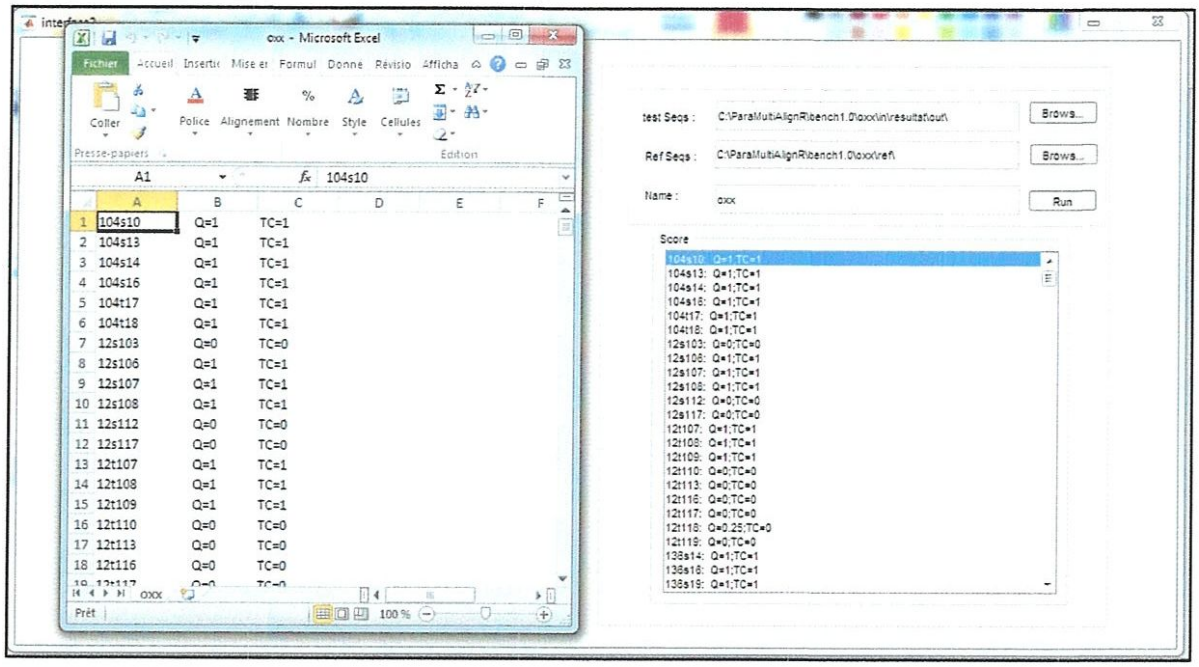


Figure 3.11 : *Calcule de qualité.*

Afin de terminer les calculs de la qualité d'alignement, les résultats seront affichés et enregistrés dans un fichier csv (*Figure 3.12*).



	A	B	C
1	104s10	Q=1	TC=1
2	104s13	Q=1	TC=1
3	104s14	Q=1	TC=1
4	104s16	Q=1	TC=1
5	104t17	Q=1	TC=1
6	104t18	Q=1	TC=1
7	12s103	Q=0	TC=0
8	12s106	Q=1	TC=1
9	12s107	Q=1	TC=1
10	12s108	Q=1	TC=1
11	12s112	Q=0	TC=0
12	12s117	Q=0	TC=0
13	12t107	Q=1	TC=1
14	12t108	Q=1	TC=1
15	12t109	Q=1	TC=1
16	12t110	Q=0	TC=0
17	12t113	Q=0	TC=0
18	12t116	Q=0	TC=0

Figure 3.12 : *Qualité d'alignement.*

3.5. Conclusion

Dans ce chapitre nous avons présenté notre outil d'alignement de séquences biologiques. Notre outil peut aligner des séquences biologiques en utilisant plusieurs algorithmes. Par défaut, il utilise notre propre algorithme.

Comme deuxième fonctionnalité, notre outil fait la comparaison entre les différentes approches en alignant des Benchmarks dédiés à la comparaison entre les différents outils et en calculant la qualité pour chaque outil.

Conclusion générale

Dans ce travail, nous avons présenté une nouvelle technique d'optimisation pour l'alignement des séquences multiples en bioinformatique. Dans notre technique, l'alignement implique deux étapes principales: le premier est le regroupement de séquences en sous-ensembles, et cela a une grande amélioration dans le temps d'exécution dans le cas de séries de données à grande échelle, et permet également l'utilisation du parallélisme dans le cas des ordinateurs multi-cœur, ce qui a permis d'éviter l'échec apparut dans la plus part des méthodes MSA pour aligner un grand nombre de séquences. La deuxième stratégie consiste à fusionner les algorithmes (local / global) afin d'améliorer la précision de la méthode progressive. La grande caractéristique de notre approche est sa simplicité et sa capacité à fournir une plat-forme extensible pour améliorer d'autres programmes d'alignement.

Notre travail a abouti à la création d'un outil d'alignement de séquence utilisant différents algorithmes.

4415, 1989.

[13] Hogeweg, P., & Hesper, B, "The alignment of sets of sequences and the construction of phyletic trees: an integrated method," *Journal of molecular evolution,* vol. 20, no. 2, pp. 175-186, 1984.

[14] Feng, D. F., & Doolittle, R. F, "Progressive sequence alignment as a prerequisitetto correct phylogenetic trees," *Journal of molecular evolution,* vol. 25, no. 4, pp. 351-360, 1987.

[15] Thompson, J. D., Higgins, D. G., & Gibson, T. J, "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic acids research,* vol. 22, no. 22, pp. 4673-4680, 1994.

[16] Thompson, J. D., Plewniak, F., & Poch, O, "BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs," *Bioinformatics,* vol. 15, no. 1, pp. 87-88, 1999.

[17] Edgar, R. C, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research,* vol. 32, no. 5, pp. 1792-1797, 2004.

[18] Abdesslem, L., MESHOUL, S., BOUFAIDA, P. Z., NAIMI, P. D., SAIDOUNI, D., & BENSLAMA, A, "Approche quantique évolutionnaire pour l'alignement multiple de séquences en bioinformatique," 2016.

[19] Li, K.-B., "ClustalW-MPI: ClustalW analysis using distributed and parallel computing. Bioinformatics," vol. 19, no. 12, pp. 1585-1586, 2003.

[20] Zola, J., et al, "PARALLEL-TCOFFEE: A parallel multiple sequence aligner. ISCA PDCS," vol. 7, pp. 248-253, 2007.

[21] Church, P.C., et al, "Design of multiple sequence alignment algorithms on parallel, distributed memory supercomputers. in Engineering in Medicine and Biology Society, EMBC," in *Annual International Conference of the IEEE*, 2011.

[22] Vouzis, P.D. and N.V, "Sahinidis, GPU-BLAST: using graphics processors to accelerate protein sequence alignment. Bioinformatics," vol. 27, no. 2, pp. 182-188, 2011.

[23] Blazewicz, J., et al, " G-MSA—A GPU-based, fast and accurate algorithm for multiple sequence alignment," *Journal of Parallel and Distributed Computing,* vol. 73, no. 1, pp. 32-41, 2013.

[24] Zafalon, G.F., et al, "Improvements in the score matrix calculation method using parallel score estimating algorithm," 2013.

[25] Agarwal, P. and S, "Solving sequence alignment problem using pipeline approach," *Bharati Vidyapeeth's Institute of Computer Applications and Management,* p. 107, 2009.

[26] Huang, S., et al, "A novel multi-alignment pipeline for high-throughput sequencing data," *Database,* vol. 2014, p. bau057, 2014.

[27] Naveed, T., I.S. Siddiqui, and S. Ahmed, "Parallel needleman-wunsch algorithm for grid. in Proceedings of the PAK-US International Symposium on High Capacity Optical Networks and Enabling Technologies (HONET 2005)," *Islamabad,* 2005.

[28] Saeed, F. and A. Khokhar, "A domain decomposition strategy for alignment of multiple biological sequences on multiprocessor platforms," *Journal of Parallel and Distributed Computing,* vol. 69, no. 7, pp. 666-677, 2009.

[29] Edgar, R.C, "Search and clustering orders of magnitude faster than BLAST," *Bioinformatics,* vol. 26, no. 19, pp. 2460-2461, 2010.

[30] Li, W. and A. Godzik, Cd-hit, "a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics,* vol. 22, no. 13, pp. 1658-1659, 2006.

[31] Dondoshansky, I. and Y. Wolf, "Blastclust (ncbi software development toolkit). NCBI, Bethesda, Md," 2002.

[32] Zhu, X., K. Li, and A. Salah, " A data parallel strategy for aligning multiple biological sequences on multi-core computers," *Computers in biology and medicine,* vol. 43, no. 4, pp. 350-361, 2013.

[33] Edgar, R.C, "Search and clustering orders of magnitude faster than BLAST," *Bioinformatics,* vol. 26, no. 19, pp. 2460-2461, 2010.

[34] Layeb, A., S. Meshoul, and M. Batouche, "A Hybrid method for effective multiple sequence alignment. in Computers and Communications," 2009.

[35] Ponty, Y., M. Termier, and A. Denise, "GenRGenS: software for generating random genomic sequences and structures," *Bioinformatics,* vol. 22, no. 12, pp. 1534-1535, 2006.

[36] Edgar, R.C, "A collection of protein sequence alignment benchmarks," 2017. [Online]. Available: http://www.drive5.com/bench/.

[37] Edgar, R.C., "A quality scoring program," 2017. [Online]. Available: http://www.drive5.com/qscore/.

[38] Thompson, J.D., et al., "A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives," *PloS one,* vol. 6, no. 3, p. e18093, 2011.