***Réf : MR-INFO-2013- …..***

**Mémoire pour obtenir le**

**Diplôme de Mastère de Recherche en Informatique**

**Parcours : Sciences de l’Informatique**

Présenté et soutenu publiquement le ../../2014

Par

**Mlle. NEFFATI CHAIMA**

|  |
| --- |
| **Algorithme d’Alignement Multiple de Séquences**  **Biologiques basé sur la Technique de Biregroupement.** |
|  |

**Composition du jury**

**Monsieur Prof. Ben Braiek Ezzeddine Président**

**Monsieur Dr. Faouzi Mhamdi Rapporteur**

**Monsieur Prof. Mourad Elloumi Encadrant**

**Monsieur Dr. Wassim Ayadi Invité**

**Année universitaire : 2012-2013**

**Dédicaces**

*Je dédie ce mémoire à …*

*À mon Père Noureddine*

*Aucune dédicace ne saurait exprimer l’amour,*

*l’estime, le dévouement et le respect que j’ai toujours eu pour vous.*

*Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien-être.*

*Ce travail est le fruit de tes sacrifices que tu as consentis pour mon éducation et ma formation.*

*À ma très chère mère Yosra*

*Affable, honorable, aimable : Tu représentes pour moi le symbole de la bonté par excellence, la source de tendresse et l’exemple du dévouement qui n’a pas cessé de m’encourager et de prier pour moi.*

*Ta prière et ta bénédiction m’ont été d’un grand secours pour mener à bien mes études.*

*Aucune dédicace ne saurait être assez éloquente pour exprimer ce que tu mérites pour tous les sacrifices que tu n’as cessé de me donner depuis ma naissance, durant mon enfance et même à l’âge adulte.*

*Puisse Dieu, le tout puissant, te préserver et t’accorder santé, longue vie et bonheur.*

*À mon très cher frère Kais et Sœur Nesrine*

*Mon cher frère et Ma chère sœur qui m’ont le père et la mère, les mots ne suffisent guère pour exprimer l’attachement, l’amour et l’affection que je porte pour vous.*

*Mes anges gardiens et mes fidèles accompagnants dans les moments les plus délicats de cette vie mystérieuse et présents dans tous mes moments d’examens par leur soutien moral et leurs belles surprises sucrées.*

*Je vous dédie ce travail avec tous mes vœux de bonheur, de santé et de réussite.*

*À mon Fiancé Helmi*

*Quand je t’ai connu, j’ai trouvé l’homme de ma vie. Ma vie à ton coté est remplie de belles surprises.*

*Ton soutien moral et matériel, ta gentillesse sans égal, ton profond attachement m'ont permis de réussir mes études.*

*Ton aide, tes conseils et tes encouragements sont inoubliables*

*Que dieu réunisse nos chemins pour un long commun serein et que ce travail soit témoignage de ma reconnaissance.*

*Ma grande famille.*

*Mes chers ami(e)s, et enseignant(e)s.*

*Tous qui ont collaborés de près ou de loin à l’élaboration de ce travail.*

*Que dieu leurs accorde santé et prospérité*.

*Remerciements …*

Ce mémoire de mastère a été réalisé sous la direction de Monsieur Mourad Elloumi, Professeur en Informatique à la Faculté des Sciences Économiques et de Gestion de Tunis et membre de Laboratoire des Technologies de l’Information et Communications et de Génie Electrique (LATICE)

Je tiens, tout d’abord, à remercier mon encadreur, Monsieur Mourad Elloumi, qui m’a offert la précieuse occasion de travailler avec lui et qui, avec un recul à la fois scientifique et constructif, a su me guider dans cette recherche. Je lui exprime par ces quelques lignes ma sincère gratitude.

Ma gratitude s’adresse spécialement à Monsieur Wassim Ayadi, assistant à l'ESSEC de Tunis, pour son encadrement, sa confiance et pour l’aide qu’il m’a prodigué durant ce travail laborieux, ainsi, pour son suivi et son encouragement tout au long de ce mémoire.

Je tiens à présenter tous mes respects et ma gratitude à Monsieur Faouzi Mhamdi, maitre assistant à l’ESSTT, qui m’a offert l’occasion de travailler dans le domaine de la bioinformatique.

Je tiens à remercier également les membres des jurys qui ont accepté de me faire l'honneur de présider et rapporter mon travail. Je tiens à leur exprimer mes remerciements les plus distingués.

Je tiens, au terme de ce travail, à présenter mon vif remerciement à toutes personnes qui ont contribué, de près ou de loin, à son bon déroulement.

Merci à tous…

# 

# Table des matières

|  |  |  |
| --- | --- | --- |
|  | **Table des matières**…………………………………………………………………………………………………………………..……….….… | ……….. |
|  | **Liste des Figures**……………………………………………………………………………………………………………………..……….….… | ……….. |
|  |  |  |
|  | **Liste des tableaux**………………………….…………………………………………………………………………………………….……....…. | ……….. |
|  |  |  |
|  | **Liste des algorithmes**…………………………………………………………………………………………………………………………….... | ……….. |
|  |  |  |
|  | **Abréviations** ………………………………………………………………………………………………………………………………………….… | ……….. |
|  |  |  |
|  | **Introduction Générale**………………………………………………………………………………………………………………………….….. | ……...1 |
|  | Contexte de travail ……………………………………………………………………………………………………………………………….... | ……...1 |
|  | Motivations et contribution.……………………………………………………………………………………………………………….... | ……...2 |
|  | Structure du mémoire………………………………………………………………………………………………………………………….... | ……...4 |
|  |  |  |
|  | **Chapitre 1 Alignement de Séquences Biologiques**…………………………………………………………………………………… | ……...6 |
|  | 1.1 Introduction…………………………………………………………………………………………………………………………………….... | ……...6 |
|  | 1.2 Alignement de Séquences…………………………………………………………………………………………………………..…..… | ………7 |
|  | 1.2.1 Définitions et Notations…………………………………………………………………………………………………………… | ………7 |
|  | 1.2.2 Description d’un alignement…………………………………………………………………………………………………... | ……...9 |
|  | 1.3. Les fonctions d’évaluation……………………………………………………………………………………………………………… | …....10 |
|  | 1.3.1 Les matrices de similarité…………………………………………………………………………………………………….…. | …….10 |
|  | 1.3.2 Les fonctions de scores……………………………………………………………………………………………………...... | …….11 |
|  | 1.3.3 Evaluation des brèches………………………………………………………………………………………………….…..…… | …….12 |
|  | 1.4. Les algorithmes d’alignement…………………………………………………………………………………………..……………… | …….12 |
|  | 1.4. 1.Alignement par paire…………………………………………………………………………………………………………...... | …….13 |
|  | 1.4. 2.Alignement Multiple de Séquences………………………………………………………………………………………... | …….16 |
|  | 1.5. Conclusion………………………………………………………………………………………………………………….......................... | …….23 |
|  |  |  |
|  | **Chapitre 2 Biregroupement de Données Biologiques**……………………………………………………………………………… | …….24 |
|  | 2.1. Introduction……………………………………………………………………………………………………………………………………….. | …….24 |
|  | 2.2. Définition et Notation…………………………………………………………………………………………………………................ | …….25 |
|  | 2.3. Biregroupement…………………………………………………………………………………………………………………………………. | …….26 |
|  | 2.4. Les Bigroupes…………………………………………………………………………………………………………………………………….. | …….27 |
|  | 2.4.1. Les types de bigroupes……………………………………………………………………………………………………………… | …….27 |
|  | 2.4.2. Les structures de bigroupes……………………………………………………………………………………………………… | …….28 |
|  | 2.5. Fonctions d’évaluation………………………………………………………………………………………………………………………. | …….29 |
|  | 2.6. Les algorithmes de biregroupement………………………………………………………………………………………………… | …….31 |
|  | 2.6.1. Les algorithmes systématiques………………………………………………………………………………………………… | ...….31 |
|  | 2.6.2. Les algorithmes stochastiques………………………………………………………………………………………………… | ..…..35 |
|  | 2.6.3. Récapitulatif des algorithmes………………………………………………………………………………………………….. | ..…..37 |
|  | 2.7. Conclusion…………………………………………………………………………………………………………………………………………. | ..…..38 |
|  |  |  |
|  | **Chapitre 3 BicMSA : Algorithme d’Alignement Multiple basé sur le Biregroupement** …………………………… | …….39 |
|  | 3.1. Introduction…………………………………………………………………………………………………………………..………………….. | …….39 |
|  | 3.2. L’algorithme BicMSA…………………………………………………………………………………………………………………………. | …….40 |
|  | 3.2.1. Définitions et notions de base………………………………………………………………………………………….…..… | …….40 |
|  | 3.2.2. Description de l’algorithme BicMSA……………………………………………………………………………………….. | …….42 |
|  | 3.2.3. Exemple illustratif…………………………………………………………………………………………………………………….. | …….53 |
|  | 3.3. Discussion…………………………………………………………………………………………………………………………..…………….. | …….60 |
|  | 3.4. Conclusion……………………………………………………………………………………………………………………………………..…. | ...….63 |
|  |  |  |
|  | **Chapitre 4 Étude Expérimentale**……………………………………………………………………………………………………….…….. | ...….64 |
|  | 4.1. Introduction…………………………………………………………………………………………………………………………….……….. | ...….64 |
|  | 4.2. Les jeux de données utilisés…………………………………………………………………………………………………………….. | ...….63 |
|  | 4.2.1. BALIBASE (Protéines)……………………………………………………………………………………………………………. | ...….65 |
|  | 4.2.2. BRALIBASE (ARN)…………………………………………………………………………………………………………..……… | ...….66 |
|  | 4.2.3. DIRMBASE (ADN)……………………………………………………………………………………………………..………….… | ...….66 |
|  | 4.3. Critères de comparaison………………………………………………………………………………………………………..……..….. | ...….67 |
|  | 4.4. Analyse de sensibilité……………………………………………………………………………………………………………..………... | ...….67 |
|  | 4.4.1 Choix de la taille du bloc………………………………………………………………………………………………………..… | ...….67 |
|  | 4.5. Résultats expérimentaux sur des données réelles………………………………………………………………………….. | ...….69 |
|  | 4.5.1. Protocole des expérimentaux………………………………………………………………………………………………..… | ...….69 |
|  | 4.5.2. Résultats expérimentaux basés sur l’évaluation de protéines …………………………………………....… | ...….72 |
|  | 4.5.3. Résultats expérimentaux basés sur l’évaluation de l’ARN …………………………………………………….. | ...….76 |
|  | 4.5.4. Résultats expérimentaux basés sur l’évaluation de l’ADN …………………………………………………..… | ...….80 |
|  | 4.6. Temps d’exécution…………………………………………………………………………………………………………….…………..… | …....82 |
|  | 4.7. Conclusion………………………………………………………………………………………………………………………………………… | …....83 |
|  |  |  |
|  | **Conclusion Générale** ………………………………………………………………………………………………………………………………. | …….84 |
|  | Perspectives de recherches……………………………………………………………………………………………………………… | …….86 |
|  |  |  |
|  | **Références**……………………………………………………….……………………………………………………………………………………..… | ……….. |
|  |  |  |
|  | **Annexe**……….…………………………………………………….…………………………………………………………………………………….… | ……….. |

# Liste des Figures

|  |  |  |
| --- | --- | --- |
|  | Figure ‎1‑1 Les séquences biologiques …………………………………………………………………………………………………… | ……...7 |
|  | Figure ‎1‑2 Séquence de protéines ……………………………………………………………………………………………………..... | ……...8 |
|  | Figure ‎1‑3 Les mouvements d'alignement ………………………………………………………………………………………..…… | ………9 |
|  | Figure ‎1‑4 Alignement de séquences protéiques.………………………………………………………………………………..… | ………9 |
|  | Figure ‎1‑5 Différents Alignements de deux séquences………………………………………………………………………..... | …......9 |
|  | Figure ‎1‑6 Exemple d’un alignement global….…..…………………………………………………………………………………… | …....12 |
|  | Figure ‎1‑7 Exemple d’un alignement local……………………………………………………………………………………..….…. | …....13 |
|  | Figure ‎1‑8 La matrice M de Needleman & Wunch …………………………………………………………………………........ | …….15 |
|  | Figure ‎1‑9 Alignement global Needleman & Wunch……………………………………………………………………………… | …….15 |
|  | Figure ‎1‑10 Comparaison entre un alignement global et un alignement local………………………..……..…… | …….16 |
|  | Figure ‎1‑11 Les algorithmes MSA……………………………………………………………………………………………………....... | …….17 |
|  | Figure ‎1‑12 Algorithme de CLUSTAL………….……………………………………………………………………………………....... | …….18 |
|  |  |  |
|  | Figure ‎2‑1 Processus ECD………………………………………..……………………………………………………………..………………… | …….24 |
|  | Figure ‎2‑2 La différence entre le regroupement et le biregroupement ……………………………………………….. | …….25 |
|  | Figure ‎2‑3 BlockMSA : Alignement en utilisant l’algorithme de biregroupement BIMAX …….………......... | …….32 |
|  |  |  |
|  | Figure 3‑1 Processus d’algorithme BicMSA……………..………………………………..…………………………………………… | …….43 |
|  | Figure 3‑2 Algorithme BicMSA…………….………………………………………………………………………………………………… | …….53 |
|  | Figure 3‑3 Groupes des séquences d’ADN………….…..…………………………………………………………………………….. | …….53 |
|  | Figure 3‑4 Génération des meilleurs alignements globaux………………..…………………………………………………. | …….54 |
|  | Figure 3‑5 Les meilleurs alignements………………………………..………………………………………………………………….. | …….54 |
|  | Figure 3‑6 Génération de blocs……….………………………………..…………………………………………………………………… | …….55 |
|  | Figure 3‑7 Résultat de BiMine-modifié……………………………..…………………………………………………………………… | …….57 |
|  | Figure 3‑8 Résultat de biregroupement…………………………..………………………………………………………………….... | …….57 |
|  | Figure 3‑9 Assemblage par biregroupement …………………..………………………………………...………………………… | …….58 |
|  | Figure 3‑10 Assemblage des blocs-all-all…………………………..………………………………………………………………….. | …….59 |
|  | Figure 3‑11 Résultat de Raffinement………………………………..……………………………….………………………………….. | …….59 |
|  | Figure 3‑12 Les Groupes de séquences d’ADN à aligner……………..………………………….……………………………. | …….59 |
|  | Figure 3‑13 Résultat final de BicMSA…………………………..……………..…………………….………………………………… | …….59 |
|  | Figure 3‑14 Comparaison de BicMSA avec quelques algorithmes………………………………………………………… | …….60 |
|  |  |  |
|  | Figure ‎4‑1 Fichier FASTA des séquences à aligner..……………………………………………………………………….……. | …....70 |
|  | Figure ‎4‑2 Alignement des ADNs avec BicMSA…………………………………………………………………………….…….. | …....71 |
|  | Figure ‎4‑3 BALIBASE : Comparaison des résultats de la référence 1………………………………………..……… | …....72 |
|  | Figure ‎4‑4 BALIBASE : Comparaison des résultats de la référence 2………………………………………………… | …....73 |
|  | Figure ‎4‑5 BALIBASE : Comparaison des résultats de la référence 3…………………………………………………. | …....74 |
|  | Figure ‎4‑6 BALIBASE : Comparaison des résultats de la référence 4……………………………………….………… | …....75 |
|  | Figure ‎4‑7 BALIBASE : Comparaison des résultats de la référence 5………………………………………………… | …....75 |
|  | Figure ‎4‑8 BALIBASE : Comparaison des résultats avec des alignements à grande brèche……………… | …....76 |
|  | Figure ‎4‑9 BRALIBASE : Résultats des SPS avec les petites familles………………………………………………… | …....77 |
|  | Figure ‎410 BRALIBASE : Résultats des SPS avec les familles moyennes…………………………………………… | …....78 |
|  | Figure ‎4‑11BRALIBASE : Résultats des SPS avec les grandes familles (n=10)………………………………… | …....79 |
|  | Figure ‎4‑12BRALIBASE : Résultats des SPS avec les grandes familles (n=15)………………………………… | …....79 |
|  | Figure ‎4‑13BRALIBASE : Comparaison du BlockMSA avec BicMSA…………………………………………………… | …....79 |
|  | Figure ‎4‑14BRALIBASE : Comparaison avec la moyenne des SPS de toutes les familles ……………………. | …....80 |
|  | Figure ‎4‑15BRALIBASE : DIRMBASE : Résultats des SPS……………………….……………………….…………………… | …....81 |
|  | Figure ‎4‑16 DIRMBASE : Résultats des CS…………………..……………………….……………………….………………………… | …....81 |
|  | Figure ‎4‑17 DIRMBASE : Comparaison des algorithmes en suivant score SPS……………………………………… | …....82 |
|  | Figure ‎4‑18 DIRMBASE : Comparaison des algorithmes en suivant score CS……………………………………… | …....82 |
|  | Figure ‎4‑19 Temps d’exécution des algorithmes pour la base DIRMBASE…………………………………………… | …....82 |
|  | Figure ‎4‑20 Temps d’exécution BlockMSA et BicMSA pour BRALIBASE……………………………………………… | …....83 |
|  |  |  |
|  | Figure A-1 La progression du séquençage des génomes……………………………………………………………………….. | ……..… |
|  | Figure A-2. Les composants génomiques………………………………………………………………………………………………… | ……..… |
|  | Figure A-3 Les composants biologiques…………………………………………………………………………………………………. | ……..… |
|  | Figure A-4 Fichier FASTA des ADNs………………………………………………………………………………………………………. | ……..… |
|  | Figure A-5 Fichier FASTA des ADNs alignées…………………………………………………………………………………………. | ……..… |
|  | Figure A-6 Interface de BicMSA……………………………………………………………………………………………………………….. | ……..… |
|  | Figure A-7 BRALIBASE- Comparaison de la taille des blocs suivant les moyennes des SPs………………… | ……..… |
|  | Figure A-9 BALIBASE- Comparaison de la taille des blocs suivant les valeurs maximales des SPs……… | ……..… |
|  | Figure A-8 BALIBASE- Comparaison de la taille des blocs suivant les moyennes des SPs……...………....... | ……..… |
|  | Figure A-10 DIRMBASE- Comparaison de la taille des blocs suivant les moyennes des SPs……...………... | ……..… |
|  | Figure A-11 DIRMBASE- Comparaison de la taille des blocs suivant les valeurs maximales des SPs…… | ………… |
|  |  |  |
|  |  |  |

# Liste des Tableaux

|  |  |  |
| --- | --- | --- |
|  | Tableau ‎1‑1 Evaluation des brèches……………………………………………………………………………………………………… | …....12 |
|  | Tableau ‎1‑2 Les algorithmes locaux d’alignement par paires….………………………………………………………..... | …....14 |
|  | Tableau ‎1‑3 Les algorithmes globaux d’alignement par paires ………………………………………………………….… | …....14 |
|  | Tableau ‎1‑4 Les algorithmes globaux de MSA.…………………………………………………………………………………..… | …....22 |
|  | Tableau ‎1‑5 Les algorithmes locaux de MSA ………………..…………………………………………………………………..... | …....23 |
|  |  |  |
|  | Tableau ‎2‑1 La matrice M…………………….………………………………………………………………………………………………… | …....26 |
|  | Tableau ‎2‑2 Bigroupes à valeurs constantes….…………………………………………………………………………………..... | …....27 |
|  | Tableau ‎2‑3 Bigroupes à valeurs cohérentes….…………………………………………………………………………………..... | …....28 |
|  | Tableau ‎2‑4 Bigroupes à évaluation cohérente…….……………………………………………………………………………..... | …....28 |
|  | Tableau ‎2‑5 Les structures d’un bigroupe……….….……………………………………………………………………………..... | …....29 |
|  | Tableau ‎2‑6 BiMine : Matrice M avant prétraitement….……………………………………………………………………..... | …....35 |
|  | Tableau ‎2‑7 BiMine : Matrice M après prétraitement………………………………………………………………………..... | …....35 |
|  | Tableau ‎2‑8 Les algorithmes de biregroupement………….…………………………………………………………………..... | …....38 |
|  |  |  |
|  | Tableau 3‑1 La Forme de la Matrice M………………………..……………………………..…………………………………………… | …….44 |
|  | Tableau 3‑2 Résultat d’identification de blocs….…………………………………………………………………………………… | …….56 |
|  | Tableau 3‑3 Bigroupes résultats………………..……….…..…………………………………………………………………………….. | …….58 |
|  | Tableau 3‑4 Matrice résultante………………..……………………………………………………………………………………………. | …….58 |
|  | Tableau 3‑5 Résultat de la matrice Mbloc-all-all……………..………………………………………………………………….... | …….58 |
|  |  |  |
|  | Tableau ‎4‑1 Liste des bases de jeux d’essai..………………………………………………………………………………….……. | …....65 |
|  | Tableau ‎4‑2 BALIBASE – familles utilisées…………..……………………………………………………………………….…….. | …....65 |
|  | Tableau ‎4‑3 BRALIBASE – familles utilisées…………..…………………………………………………………………....……… | …....66 |
|  | Tableau ‎4‑4 DIRMBASE – familles utilisées…………..……………………………………………………………………………… | …....66 |
|  | Tableau ‎4‑5 BRALIBASE – Résultats des blocs……………………………………….……………………………………………. | …....68 |
|  | Tableau ‎4‑6 DIRMBASE – Résultats des blocs……………………………………….………………………………….………… | …....68 |
|  | Tableau ‎4‑7 BALIBASE – Résultats des blocs……………………….……………………………………………………………… | …....69 |
|  | Tableau ‎4‑8 Protocole des expérimentations………………………………………………………………………………….…… | …....70 |
|  | Tableau ‎4‑9 BALIBASE : Comparaison des résultats de la référence 1 ………………………………………….…… | …....72 |
|  | Tableau ‎4-10 BALIBASE : Comparaison des résultats de la référence 2 ……………………………..………….…… | …....73 |
|  | Tableau ‎4‑11 BALIBASE : Comparaison des résultats de la référence 3 ……………………………………………… | …....74 |
|  | Tableau ‎4‑12 BALIBASE : Comparaison des résultats de la référence 4 …………………..………………………… | …....75 |
|  | Tableau ‎4‑13 BALIBASE : Comparaison des résultats de la référence 5 ……………………………………………… | …....75 |
|  | Tableau ‎4‑14 BALIBASE : Récapitulatif des résultats………………………………………………………..…………………. | …....75 |
|  | Tableau ‎4‑15 BRALIBASE : Moyenne des SPS avec les petites familles….……………………….…………………… | …....77 |
|  | Tableau ‎4‑16 BRALIBASE : Max des SPS avec les petites familles….…………….………………….…………………… | …....77 |
|  | Tableau ‎4‑17 BRALIBASE : Moy SPS avec les familles moyennes…………………..….………………………………… | …....78 |
|  | Tableau ‎4‑18 BRALIBASE : Max des SPS avec les familles moyennes…………………..….…………………………… | …....78 |
|  | Tableau ‎4‑19 BRALIBASE : Moy SPS avec les grandes familles…………………….……………………………………… | …....78 |
|  | Tableau ‎4‑20 BRALIBASE : Max SPS avec les grandes familles…………………………………………………………… | …....78 |
|  | Tableau ‎4‑21 BRALIBASE : Comparaison du BicMSA avec BlockMSA….………………………………………………… | …....79 |
|  | Tableau ‎4‑22 BRALIBASE : Comparaison avec la moyenne des SPS de toutes les familles…………………… | …....80 |
|  | Tableau ‎4‑23 DIRMBASE : Résultats des Moy\_SPS………………………….……………….…………………….……………… | …....81 |
|  | Tableau ‎4‑24 DIRMBASE : Résultats des Moy\_CS………………………….………………..…………………….……………… | …....81 |
|  | Tableau ‎4‑25 Temps d’exécution des algorithmes pour la base DIRMBASE………………………….……………… | …....82 |
|  | Tableau ‎4‑26 Temps d’exécution BlockMSA et BicMSA pour BRALIBASE…..……………………….……………… | …....83 |
|  |  |  |
|  | Tableau A-1 Tailles de génomes et nombres de chromosomes………….………………………………………………….. | ……..… |
|  | Tableau A-2. Les acides nucléiques…………….…………………………………………………………………………………………… | ……..… |
|  | Tableau A-3 Liste des 20 acides aminés…………………………………………………………………………………………………. | ……..… |

# Liste des Algorithmes

|  |  |  |
| --- | --- | --- |
|  | Algorithme 3‑1 BicMSA…………………………………..………..………………………………..……………………………… | …….43 |
|  | Algorithme 3‑2 FN\_Génération\_blocs()…………………………………………………………………………………..… | …….45 |
|  | Algorithme 3‑3 FN\_identification\_blocs()………….…..………………………………………………………………... | …….46 |
|  | Algorithme 3‑4 FN\_ Processus\_Ajout/MiseàJour\_2blocs()…..……………..………………………………….…. | …….47 |
|  | Algorithme 3‑5 FN\_ Processus\_Ajout/MiseàJour\_1blocs()…..……………..………………………………….…. | …….48 |
|  | Algorithme 3‑6 FN\_Biregroupement()………………………………..………………………………………………….…. | …….49 |
|  | Algorithme 3‑7 FN\_Assemblage\_Raffinement()………………..……………………………………………………... | …….49 |
|  | Algorithme 3‑8 FN\_Assemblage\_Par\_Biregroupement()..…………………………………………………….…. | …….50 |
|  | Algorithme 3‑9 FN\_AssemblerUnBloc()…………………………..……………………………………………………….. | …….51 |
|  | Algorithme 3‑10 FN\_Raffinement()………………………..……………………………………………………………….... | …….52 |
|  | Algorithme 3‑11 FN\_Nettoyage\_PrefixSeqDecalageEnAvant()……………………..………………………… | …….53 |

# Introduction Générale

## Contexte de travail

La *Bioinformatique* est un domaine en plein essor. Il se situe à l'interface de deux disciplines : la ***biologie*** etl'***informatique*.** Dans les années 90, la *bioinformatique* a été présentée comme une annotation et stockage des calculs issus de différents problèmes d’analyses de séquences biologiques par un ordinateur [Pavy et al., 1999]. Aujourd’hui, la *bioinformatique* évolue en fonction des nouveaux problèmes de la *biologie moderne*. Elle n’est plus présentée comme un simple traitement informatique, mais, elle est considérée comme un domaine multidisciplinaire où travaillent des biologistes, médecins, physiciens, mathématiciens et bioinformaticiens dans le but de résoudre un problème scientifique posé par la biologie [Dardel et Képès, 2006]. Plusieurs travaux de recherche ont été menés comme le traitement des *pathologies* à travers l'étude du *génome* humain [Tran et al., 2007], le ré-séquençage des séquences *moléculaires* [Van Leeuwen et al., 2003], la recherche des *polymorphismes* [Kerr et Churchill, 2001], le diagnostic à partir du *sérum* [Kluger et al., 2003], la détection des gènes présents dans le *génome* [Trang et al., 2005], l'analyse des données d'expression des *gènes*, la reconstruction de *phylogénie* [Fitch et Margoliash, 1967]*,* la prédiction de structures secondaires [Chou et Fasman, 1974] et l’*alignement de séquences* [Needleman et Wunsch, 1970].

L’*alignement de séquences biologiques* est une opération fondamentale en bioinformatique [Needleman et Wunsch, 1970; Notredame, 2002; Van Walle et al., 2005]. Son objectif est d’identifier les zones en concordance et de maximiser la similarité entre les *nucléotides* ou les *acides aminés* dans un ensemble de séquences biologiques afin de mettre en évidence des régions *homologues.* Cette identification estappliquée par la « superposition » physique des séquences biologiques afin de pouvoir les comparer et dévoiler les régions *similaires*, qui, au cours de l’évolution, ont été conservées. Ces régions *similaires* jouent un rôle très important d’un point de vue fonctionnel et structurel. Elles permettent de dévoiler efficacement l’histoire évolutive des séquences ou l’évolution des organismes vivants. Proprement dit, lorsque deux *protéines* sont fortement *similaires*, elles sont forcement *homologues* et elles ont souvent un ancêtre commun avec des fonctions similaires proches comme « *le chat* et *le tigre* ». La détection d’*homologie* permet de donner un certain nombre de prédictions sur les fonctions ou les structures des *protéines* puisque si une *protéine* est *homologue* à et est *homologue* à , forcement, est *homologue* à . Pour ce là, l’alignement est utilisé comme un point de départ pour résoudre d’autres problèmes biologiques comme l'analyse et l'annotation des *macromolécules,* l’identification des sites fonctionnels (*site catalytique*, *zone d'interaction*...) ou structurels, la construction des arbres de *phylogénies* entre un ensemble de *protéines*, ainsi, la comparaison et la combinaison des séquences d’*ADN,* etc*.*

## Motivations et contribution

La comparaison de séquences peut se faire soit de façon globale sur toute la longueur des séquences, soit de façon locale, par la recherche de courtes portions de séquences présentant une ou plusieurs caractéristiques similaires. Auparavant, toutes les recherches ont été basées sur l’*alignement global* en raison de son importance à l’époque. De ce fait, plusieurs méthodes d’alignement ont été développées comme *Needleman et Wunch* [Needleman et Wunsch, 1970], *CLUSTAL*  [Higgins et Sharp, 1988], *MULTALIN* [Corpet, 1988], *SAGA* [Notredame et Higgins, 1996] et *HMMER* [Eddy, 1995]. Cependant, l’*alignement global* dispose des limites avec les bases de données fortement bruitées. En outre, l’écart croissant entre la production et la compréhension de séquences biologiques provoque une nécessitée impérieuse d’élaborer des nouvelles méthodes. Un survol sur la littérature nous a montré que dans plusieurs applications biologiques, l’*alignement local* est plus significatif que l’*alignement global* [Duret et Bucher, 1997 ; Blanchette et  Tompa, 2003 ; Fu et Weng, 2005]. Cette signification est argumentée par la capacité de découvrir des *homologies* dans des bases de données disposant des régions faiblement conservées.

Par ailleurs, l’alignement de séquence est composé par deux types d’alignement : l’*alignement par paire* et l*’alignement multiple de séquences (MSA)*. L’objectif de l’*alignement par paire* est de définir une distance de similarité entre seulement deux séquences. Plusieurs algorithmes ont été implémentés comme *Smith et Waterman* [Smith et Waterman, 1981] *FASTA* [Pearon et Lipman, 1988] et *BLAST* [Altschul et al., 1990]. Ces algorithmes restent impuissants face aux séquences protéiques puisqu’ils se basent, dans leurs processus d’alignement, sur le calcul de score de similarités. En effet, ils produisent, soit des scores faibles ce qui amène à diverger les séquences de la base, soit des scores nuls avec les bases hautement bruitées.

À cet égard, un recours à trouver des nouveaux outils performants et rapides a été élaboré ce qui a donné naissance à une nouvelle discipline : l*’alignement multiple de séquences (MSA)*. L*’alignement multiple de séquences (MSA)* est beaucoup plus complexe que l’*alignement par paires*. Ce problème est *NP-Complet* [Carrillo et Lipman, 1988; Elias, 2006]. Son objectif est de trouver le meilleur alignement multiple entre séquences (avec >2). Le défi de MSA se présente par la recherche des meilleures méthodes d’alignement prouvant la qualité et la rapidité souhaitées. En effet, la méthode qui génère un meilleur résultat procédé par l’*homologie* résout directement le problème d’alignement des séquences et fournit des meilleures prédictions fonctionnelles et structurelles. Pour ce là, l’*alignement local* présente une solution parfaite pour les problèmes d’*alignement multiple*.

Cependant, le *MSA local* a attiré l’attention des biologistes au cours de ces dernières années à cause de la qualité des résultats donnés et de la capacité de découvrir efficacement les similarités structurelles et fonctionnelles dans les sites. Il est fiable et nécessaire pour aligner les séquences divergentes de divers espace [Morgenstern, 1999 ; Pei et Grichin, 2007]. Cette méthode est utilisée dans différentes filières comme l’annotation fonctionnelle et structurelle, la génération des familles de séquences [Wang, 2007], la prédiction et l’organisation du génome, le “*phylogenetic footprinting*” [ Blanchette et  Tompa, 2003 ; Wolfgang et al., 2011] et la construction des arbres phylogéniques à large échelle [Hongtao et Jeremy, 2012]. Plusieurs techniques ont été appliquées dans la résolution de *MSA local.* Parmi ces techniques, nous citons celle du regroupement [Corpet, 1988]. En effet, en appliquant le r*egroupement* [Quackenbush, 2006], nous considérons que toutes les régions peuvent avoir une similarité avec toutes les séquences. Cependant, il existe des régions qui ne se présentent que dans un sous-ensemble de séquences. Ceci permet de conclure que le *regroupement* est trop simpliste pour détecter de tels cas [Prelic *et al.*, 2006]. Ce problème a été résolu par l’intéressante technique, le *biregroupement* [Cheng and Church, 2000; Madeira and Oliveira, 2004]. Elle permet d'identifier des groupes de régions ayant un comportement similaire par rapport à un ensemble de séquences.

Dans cette optique, nous proposons un algorithme d’alignement multiple et local, appelé *BicMSA (****Bic****lustering-based local* ***MSA***). Comme le *MSA local* est souvent destiné à grouper un ensemble de sous-séquences conservées, le choix de la technique de regroupement peut facilement influer la qualité de l'alignement. De ce fait, nous avons eu recourt à utiliser le *biregroupement* [Cheng et Church, 2000; Madeira et Oliveira, 2004] afin de générer des *MSA locaux* pour tous les types de séquences biologiques. Également, le *biregroupement* permet la découverte non-triviale des *sous-séquences* similaires possédant des fonctions biologiques proches [Wang et al., 2002; Sharan et al., 2002; Quackenbush, 2006; Prelic et al., 2006; Divina et Aguilar-Ruiz, 2007; Pontes et al., 2007]. L’utilisation du biregroupement dans la résolution des problèmes de *MSA* est une piste qui n'a pas été bien explorée. Récemment, *Wang et al.* [Wang et al., 2007] ont proposé l'algorithme *BlockMSA* qui est  une première adaptation du *biregroupement* pour la résolution des problèmes d’*alignement multiples*. En effet, notre algorithme se présente parmi les premiers algorithmes adaptant le *biregroupement* dans la résolution des problèmes de *MSA*. En s’inspirant de *BlockMSA*, *BicMSA* trouve un compromis entre la technique de *biregroupement* et l’approche *DAC* (*Divide-and-conquer*). Il permet de résoudre le problème de *MSA local* entre un ensemble de séquences biologiques avec la résolution des sous-problèmes des alignements globaux entre un ensemble de sous-séquences. Le résultat généré est à la fois des groupes de sous-séquences similaires et un alignement local entre toutes les séquences.

## Structure du mémoire

Ce mémoire est organisé de la manière suivante :

Dans le **premier chapitre**, nous présentons un état de l'art sur l’*alignement des séquences biologiques*. Dans la première partie, nous décrivons brièvement le problème d'*alignement des séquences biologiques,* ainsi que les notions nécessaires pour la lecture de ce manuscrit. Dans la deuxième partie, nous élaborons les méthodes et les fonctions d’évaluation des *alignements de séquences biologiques*. Dans la troisième partie, nous présentons l’*alignement par paires* et les algorithmes proposés. Nous terminons ce chapitre par la présentation de l’*alignement multiple de séquences*. Par la suite, nous exposons les approches utilisées ainsi que les algorithmes proposés pour la résolution de *MSA*.

Dans le **deuxième chapitre**, nous présentons un état de l'art sur le *biregroupement des données biologiques*. Dans la première partie, nous présentons un état de l’art sur le *biregroupement des données biologiques* en décrivant son principe et les différents types et structures de *bigroupes*. Dans la deuxième partie, nous décrivons les différentes fonctions utilisées dans l’évaluation des algorithmes de *biregroupement*. Dans la dernière partie, nous élaborons les différentes approches adoptées pour la résolution du problème du *biregroupemen*t, ainsi que les algorithmes proposés.

Dans le **troisième chapitre**, nous présentons notre algorithme d’alignement multiple et local de séquences biologiques, *BicMSA (****Bic****lustering-based local* ***MSA***). Dans la première partie, nous citons les différentes définitions et notations en relation avec notre algorithme. Dans la deuxième partie, nous décrivons notre algorithme suivi d’un exemple illustratif. Dans la troisième partie, nous discutons des points forts de notre algorithme par rapport à l’algorithme *BlockMSA*. Nous terminons ce chapitre par une conclusion.

Dans le **quatrième chapitre**, nous présentons les résultats obtenus par *BicMSA*. Pour ce là, une étude expérimentale a été menée sur différentes bases de jeux d’essais (*ADN*, *ARN*, protéines). Dans la première partie, nous présentons les bases de jeux utilisées et les critères de comparaison appliqués. La deuxième partie consiste à analyser la sensibilité de *BicMSA* afin de fixer les meilleurs paramètres à utiliser pour chaque type de séquences. Dans la troisième partie, nous appliquons une évaluation sur la performance de *BicMSA* ainsi une comparaison avec les algorithmes les plus connus dans la littérature. Dans la dernière partie, nous terminons ce chapitre par une comparaison de temps d’exécution de *BicMSA* par rapport aux algorithmes les plus connus.

# Alignement de Séquences Biologiques

## Introduction

L’*alignement de séquences* consiste à « superposer » physiquement un ensemble de séquences afin de pouvoir les comparer et faire sortir les régions *similaires*, qui au cours de l’évolution ont été conservées. Ces régions jouent un rôle très important d’un point de vue fonctionnel et structurel puisqu’elles permettent de dévoiler efficacement l’histoire évolutive des *séquences* et le processus biologique des organismes vivants. Cependant, cette intéressante technique se focalise sur la recherche des *similarités* indiquant à quel degré les séquences se ressemblent afin d’identifier les *homologies.* En effet, lorsque deux *protéines* sont fortement *similaires*, elles sont forcement *homologues* et elles ont souvent un ancêtre commun avec des fonctions similaires proches comme « *le chat* et *le tigre* ». De ce fait, l’alignement de séquences est judicieux pour l'analyse et l'annotation des macromolécules. En recherchant les *homologies*, la technique d’alignement permet de donner un certain nombre de prédictions sur les fonctions ou les structures des *protéines* puisque si une *protéine* X est homologue à Y et Y est homologue à Z, alors X est homologue à Z. Notamment, cette détection d’*homologie* permet d’identifier les sites fonctionnels (*site catalytique*, *zone d'interaction*...) ou structurels. Par ailleurs, l’alignement est utilisé dans la résolution de différents problèmes biologiques. Parmi ces problèmes, nous notons la prédiction de l’histoire évolutive ou l’évolution des organismes vivants, la récupération des *séquences* d’*ADN* à partir des *sous-séquences* plus courtes et la construction des arbres de *phylogénies* entre un ensemble de *protéines*.

Ce chapitre vise à présenter le problème d’alignement des séquences biologiques. De ce fait, nous introduisons le problème d'une manière générale, en décrivant le principe, le domaine d'utilisation et les types d’alignement existants tels que l’*alignement par paire* ou l’*alignement multiple*. Pour présenter chaque type, nous invoquons les différentes techniques, approches et algorithmes utilisés. Nous finissons ce chapitre par une conclusion.

## Alignement de Séquences Biologiques

### Définitions et Notations

Dans cette section, nous définissons quelques définitions et notations nécessaires pour la lecture de ce manuscrit.

1. (ADN : Acides Désoxyriboclique) : *l’ADN est un acide désoxyribonucléique de la famille des acides nucléiques. Il se compose par quatre nucléotides (adénine, cytosine, guanine et thymine). C’est un composant nécessaire à la reproduction des organismes et à l’édification de leur fonctionnement [Harry, 2001] (voir Figure ‎1‑1).*
2. *(ARN : Acides Ribonucléioque)* : *l’ARN est obtenu avec la combinaison de deux brins d’ADN (voir Figure ‎1‑1). l’ARN dispose l’uracile au lieu de thymine.*
3. *(Protéines)*: *une protéine est une séquence composée d’un ensemble d’acides aminés. Au total, il existe 20 acides aminés, représentés par un alphabet de 20 lettres (voir Figure* ‎1‑1*).*

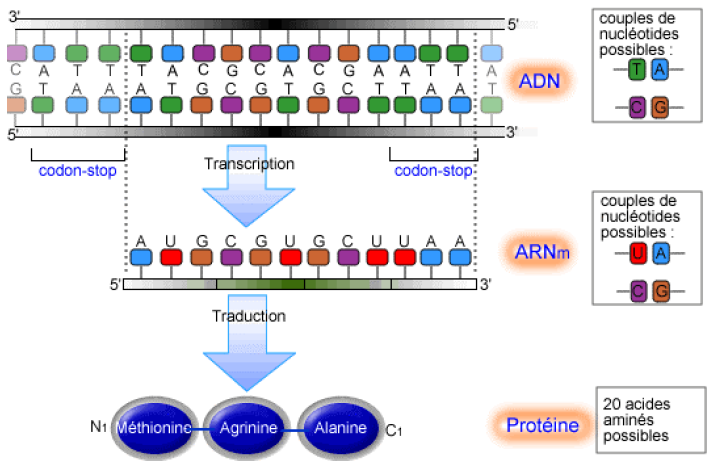


Figure ‎1‑1 Les séquences biologiques [Basseur, 2011]

1. *(Alphabet ADN) : c’est un ensemble de nucléotides représentés comme suit := {A, C, G, T}, avec A : adénine, C : cytosine, G : guanine et T : thymine.*
2. *(Alphabet ARN) :* *c’est un ensemble de nucléotides représentés comme suit : = {A,C,G,U} avec A : adénine, C : cytosine, G : guanine et U : Uracile.*
3. *(Alphabet de protéines) : c’est un ensemble d’acides aminés représentés comme suit  [Gonnick et Wheelis, 1991]: = {A,C,D,E, F, G,H, I,K, L,M,N, P, Q,R, S, T, V,W, Y} : les lettres présentent les acides aminés.*
4. (*Alphabet A)* : *un alphabet est un ensemble de symboles qui constituent une séquence composée soit de nucléotides (ADN et ARN), soit d'acides aminés (protéines). Il est noté (avec ). On définit l’alphabet vide qui contient que le caractère « - ».*
5. *(Séquence): étant donné un alphabet fini, une séquence est une succession ordonnée de caractères (avec). Elle est construite respectivement à partir des alphabets de , et (voir Figure ‎1‑2). On la note par . Le caractère d’une séquence ( avec ) est noté.*
6. *(Longueur d’une séquence)* : *la longueur d’une séquence () notée est le nombre de caractères constituent cette séquence. Par convention, une séquence de longueur nulle est notée .*
7. *(sous-séquence) :* *une sous-séquence est obtenue en récupérant des caractères successive ou pas de (Exemple : GAC est une sous-séquence de AGTACA). On définit aussi une sous-séquence toute partie qui commence à une position et se termine à une position ( avec )*

|  |
| --- |
|  |
| Figure ‎1‑2 Séquence de protéines |

1. *(Préfixe)* : *soit une séquence de longueur . On appelle préfixe de de longueur toute sous-séquence qui commence par et se termine par avec .*
2. *(Suffixe)* : *soit une séquence de longueur . On appelle suffixe de de longueur toute sous-séquence qui commence par une position et se termine par la longueur de la séquence (avec ).*
3. (*Similarité*) *: la similarité est exprimée sous la forme d'un pourcentage calculé à partir de l'alignement des séquences. Elle indique à quel degré les séquences se ressemblent. Le degré de similarité est quantifié par un score. Le résultat de la recherche d'une similarité peut être utilisé pour inférer l'homologie de séquences.*
4. (*Homologie)* : *deux séquences sont homologues si elles ont dans leur évolution un ancêtre commun. L'homologie se mesure par la similarité. Une similarité significative est un signe d'homologie. L’homologie peut concerner des protéines existant chez différentes espèces, ces protéines sont alors qualifiées de protéines orthologues. L’homologie peut aussi concerner des protéines d’une même espèce, ces protéines sont des protéines paralogues.*
5. (*Score)* : *le score permet de quantifier la similarité, soit suivant la conservation totale des mêmes résidus aux mêmes positions des séquences, soit suivant la conservation des mêmes propriétés physico-chimiques aux mêmes positions. C'est le nombre total de "bons appariements" pénalisé par le nombre de mésappariements.*
6. *(Brèche)*: *on appelle brèche (gap en anglais) l’insertion, dans une séquence, d’au moins un caractère égale à « - » ou bien à « .  »(avec)*
7. *(coût brèche)*: *on appelle coût brèche le score accordé à la présence d’une brèche dans la colonne d’un alignement entre n séquences (n). On note par la pénalité de brèche.*
8. *(Mouvements mutationnels)*: *les mouvements mutationnels sont utilisés dans la construction des alignements. Ils peuvent être soit des substitutions en remplaçant un caractère par un autre, soit des insertions en ajoutant un ou plusieurs caractères dans une séquence ou bien des délétions en supprimant un ou plusieurs caractères à partir d’une séquence (voir Figure ‎1‑3).*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | *Substitution* | | | |  | *Insertion* | | | |  | *Délétion* | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |
| Figure ‎1‑3 Les mouvements d'alignement |

### Description d’un alignement

L’alignement permet, d’une manière générale, d’identifier des zones en concordance et de maximiser la coïncidence entre les nucléotides ou les acides aminés dans un ensemble de séquences biologiques (voir Figure ‎1‑4). Il permet de mettre en évidence des zones communes afin de dévoiler des similarités entre les séquences. C’est un problème important issu de la bioinformatique [Needleman et Wunsch, 1970 ; Notredame, 2002 ; Van Walle et al., 2005]. Il est également utilisé comme un point de départ dans la résolution des autres problèmes biologiques.

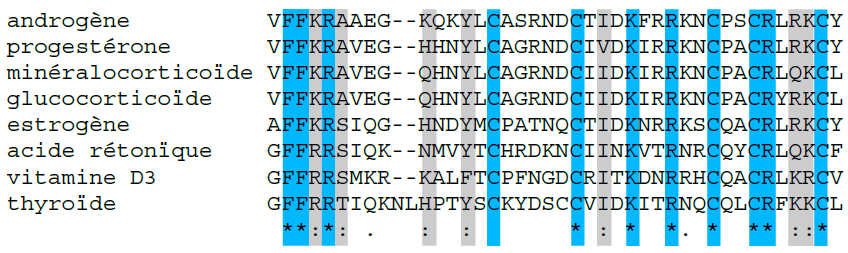


Figure ‎1‑4 Alignement de séquences protéiques

Formellement, l’alignement permet de rechercher la similarité entre séquences (avec) en superposant une séquence avec une autre séquence de telle façon qu’un seul caractère est comparé à un seul caractère pour tout et . Le résultat d’un alignement doit toujours retrouver les séquences initialement utilisées. Par ailleurs, l’alignement de sortie varie en fonction de la méthode d’alignement utilisée. Cette différence se présente avec les méthodes d’utilisation des mouvements mutationnels (voir Figure ‎1‑3). Citant par exemple les deux séquences suivantes dont égale à ‘ATATAACCA’ et  égale à ‘ATAACCAA’, nous pouvons alors trouver plusieurs alignements comme le montre la Figure ‎1‑5. Généralement, pour faciliter la visualisation des alignements, les coupures au sein des séquences alignées sont remplacées par *les brèches.*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Alignement 1** | | | | | | | | | | **A** | **T** | **A** | **T** | **A** | **A** | **C** | **C** | **A** | | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | | **A** | **T** | **A** | **-** | **A** | **-** | **C** | **C** | **A** | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Alignement 2** | | | | | | | | | | **A** | **T** | **A** | **T** | **A** | **A** | **C** | **C** | **A** | | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | | **A** | **-** | **-** | **T** | **A** | **A** | **C** | **C** | **A** | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Alignement 3** | | | | | | | | | | **A** | **T** | **A** | **T** | **A** | **A** | **C** | **C** | **A** | | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | | **-** | **-** | **A** | **T** | **A** | **A** | **C** | **C** | **A** | |

Figure ‎1‑5 Différents Alignements de deux séquences

## Les fonctions d’évaluation

Un alignement optimal correspond à la meilleure méthode qui maximise une *fonction de score*. En effet, aligner des séquences de façon optimale consiste à positionner les brèches de façon à faire correspondre un maximum d’acides aminés ou des nucléotides entre les séquences soit sur la base de l’identité stricte, soit sur la base de la conservation d’une propriété particulière (taille, polarité, réactivité chimique…).

### Les matrices de similarité

La génération d’un alignement nécessite généralement une méthode de construction matricielle pour étudier la similarité de différentes séquences. Parmi ces méthodes, nous pouvons citer la *matrice de points* (*dot-plot*) et la *matrice de substitution*.

### *Matrice de substitution :* Les matrices de substitutions permettent de quantifier les ressemblances entre les acides aminés sans prendre en compte la nature biochimique des acides aminés. La plus simple d’entre elles est la matrice identité, qui répond à la loi du tout ou rien*.* Il existe plusieurs améliorations des matrices de substitution. Parmi ces améliorations, nous citons, les matrices basées sur des arbres construits, soit en utilisant le maximum de *parcimonie* comme *PAM* (*Point Accepted Mutation*) [Dayhoff et al., 1978] et *JTT* [Jones et al., 1992] ou bien en utilisant le maximum de *vraisemblance* comme *WAG* [Whelan et Goldman, 2001]. Ces matrices sont dédiées pour les *alignements globaux*. En deuxième lieu, nous trouvons les matrices basées sur des comparaisons par paires comme *BLOSUM* (*Blocks Substitution Matrices*) [HENIKOFF et al., 1992]. Elles sont dédiées pour les *alignements locaux*.

### *Matrice de point (dot-plot) :* C’est une méthode simple et rapide. Elle utilise une symbolisation avec des points pour décrire la similarité locale. La complexité de son algorithme est de l’ordre de O() [GIBBS et al., 1970].

### Les fonctions de scores

Les fonctions de similarité ont été utilisées afin d’évaluer la qualité ou la signification d’un alignement. Proprement dit, elles permettent de mesurer la capacité d’un algorithme à extraire le maximum des régions similaires. De ce fait, un alignement avec le score maximal peut être considéré comme un alignement optimal. En effet, il existe plusieurs fonctions de score. Parmi les fonctions de score existantes, nous citons :

### *SP* (*Somme des paires*) : c’est la fonction de score la plus utilisée dans l’évaluation des alignements puisqu’elle donne des résultats raisonnables et elle ne possède pas de justifications probabilistes. Elle correspond à la somme de scores de toutes les paires de caractères qui existent dans un alignement [Carroll et al., 2007]. Le score SP est calculé selon l’Équation ‎1‑1. En effet, son défaut majeur est le traitement de chaque séquence comme si elle était directement liée à l’évolution de toutes les autres  séquences au lieu d’un seul ancêtre.

|  |  |
| --- | --- |
|  | Équation ‎1‑1 |
| Avec : (resp. ) est le caractère de la séquence  (resp. ) dans la colonne d’un alignement (avec ), est la longueur de l’alignement et est le score d’un alignement de deux caractères dans la colonne | |

### *WSP :* En se basant sur les limites de *SP, Altschul et al* [Altschul et al., 1990] ont proposé le SP-pondéré *WSP* (*weighted Sum of paire*)pour résoudre partiellement le défaut évoqué en ajoutant un poids à *SP,* noté. On calcule le *WSP* selon l’Équation ‎1‑2.

|  |  |  |
| --- | --- | --- |
|  | Équation ‎1‑2 | |
| Avec : (resp. ) est le poids de la séquence (resp. ) de l’alignement . | |

### Consensus : Pour calculer le consensus d’un alignement [WATERMA et al., 1990], il faut construire tout d’abord la séquence de consensus . En effet, cette séquence est construite par la prise du caractère le plus fréquent dans chaque colonne d’un alignement. Le score de consensus est la somme des distances entre chaque et la (avec et le nombre de séquences).

|  |  |
| --- | --- |
|  | Équation ‎1‑3 |
| Avec : est la distance entre la séquence et la séquence de consensus . | | |

### Entropie : L’entropie définit la fréquence d’apparence de chaque caractère dans chaque colonne d’un alignement. Parmi les outils proposés à base d’entropie, nous citons Al2co [Pei et Grishin, 2001]. En effet, l’entropie d’un alignement multiple de séquence est égale à la somme des entropies de chaque colonne :

|  |  |
| --- | --- |
|  | Équation ‎1‑4 |
| Avec : est le nombre d’occurrences d’un caractère dans la colonne (avec L la longueur d’alignement . | | |

* Autres Scores : Il existe d’autres scores utilisés dans l’évaluation des alignements comme *Coffee* [Notredame et al., 1998], *Normd* [Thompson et al., 2001] et *Confind* [Smagala et al., 2005].

### Evaluation des brèches

L’alignement dans la biologie se base sur les mouvements mutationnels ce qui peut, facilement, pénaliser un alignement par la présence des brèches. Bien qu’une brèche ne soit pas crée naturellement, mais avec un biologiste ou un programme et que sa présence peut pénaliser l’alignement, son évaluation est devenue de plus en plus importante. En effet, il existe plusieurs modèles pour évaluer le coût de brèches. Parmi ces derniers, nous citons le *modèle linéaire* et le *modèle affine* (voir Tableau ‎1‑1).

|  |  |
| --- | --- |
| Modèle de pénalité de brèches |  |
| Modèle linéaire |  |
| Modèle affine |  |
| Modèle logarithmique |  |
| Modèle logarithmique-affine |  |
| Avec : *k* : le nombre des brèches successives et *a, b, c* : des constantes | | |
| **Tableau ‎1‑1 Evaluation des brèches.** | | | |

## Les algorithmes d’alignement

Afin de comparer les séquences biologiques et dévoiler leurs similarités, on utilise différentes techniques : l’*alignement par paire* et l’*alignement multiple*. Par ailleurs, dans chaque technique, la comparaison peut se faire, soit de façon globale pour avoir un *alignement global*, soit de façon locale pour trouver l’*alignement local*.

* *Alignement global*: C’est le fait d’aligner des séquences entières (voir Figure ‎1‑6). Ce type d’alignement donne de bons résultats si les séquences comparées sont fortement reliées. Il est primordial pour les alignements disposants un nombre petit de séquences. Toutefois, l’alignement global peut engendrer des problèmes avec les séquences de tailles très différentes.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alignement Global**:** | **A** | **A** | **G** | **A** | **G** | **T** | **C** | **G** | **G** | **T** |
| **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** |
| **A** | **-** | **G** | **A** | **-** | **T** | **C** | **-** | **G** | **T** |
| Figure ‎1‑6 Exemple d’un alignement global | | | | | | | | | | |

* *Alignement local* : Il permet d’aligner de courtes portions de séquence présentant des similarités entre elles (voir Figure ‎1‑7). Contrairement à un alignement global, l’alignement local est utilisé avec les séquences faiblement et fortement reliées disposant des tailles identiques ou différentes. Il est plus flexible que l’alignement global puisqu’il calcule seulement le score des régions homologues sans pénalisation des régions non-homologues. Ainsi, il donne de bons résultats avec des séquences ayant une forte homologie.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alignement Local**:** | **A** | **A** | **G** | **A** | **G** | **T** | **C** | **G** | **G** | **T** |
| **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** | **|** |
| **-** | **A** | **G** | **A** | **-** | **T** | **C** | **G** | **T** | **-** |
| **Figure ‎1‑7 Exemple d**’**un alignement local** | | | | | | | | | | | |

### Alignement par paire

L’alignement de deux séquences, également appelé *alignement par paires*, permet de définir facilement une similarité entre deux séquences en partant d’une séquence référence pour arriver à un degré de similarité avec une autre séquence de test. En effet, c’est un type particulier du problème d’*alignement multiple de séquences* que nous aborderons dans la section suivante. Il est défini comme un problème polynomial [Mokaddem et Elloumi, 2011].

Formellement, étant donné et deux séquences définies sur un alphabet , et soient et deux séquences définies sur ′ tel que ′ = . On affirme que les séquences et constituent un alignement par paire de deux séquences et si et seulement si :

* + et ont la même longueur :
* *Classification des algorithmes*

Il existe plusieurs algorithmes dédiés pour l’alignement par paire (voir Tableau ‎1‑2, Tableau ‎1‑3). Parmi ces algorithmes, nous citons ceux qui adoptent la programmation dynamique tels que : l’algorithme d’alignement global de *Needleman* et *Wunch* [Needleman et Wunsch, 1970] et l’algorithme d’alignement local de *Smith* et *Waterman* [Smith et Waterman, 1981]. En effet, on dit qu’un algorithme de résolution est basé sur le principe de la programmation dynamique s’il construit son alignement d’une manière récursive en partant des problèmes de plus bas niveau et en se basant sur une matrice de tous les résultats intermédiaires obtenus.

|  |  |  |
| --- | --- | --- |
| Algorithme | Approche | Lien |
| Smith et Waterman [Smith et Waterman, 1981] | Programmation dynamique | - |
| FASTA [Pearon et Lipman, 1988] | approche par grainage | http://ebi.ac.uk/Tools/fasta/ |
| BLAST [Altschul et al., 1990] | approche par grainage | http://ebi.ac.uk/Tools/blast |
| CHAOS [Brudno et al., 2003] | approche par grainage | http://cs.toronto.edu/brudo/chaos/ |
| YASS [Noé et Kucherov, 2005] | approche par grainage | http://bioinfo.lifl.fr/yass/yass.php |

Tableau ‎1‑2 Les algorithmes locaux d'alignement par paires.

|  |  |  |
| --- | --- | --- |
| Algorithme | Approche | Lien |
| Needleman et Wunch  [Needleman et Wunsch, 1970] | Programmation dynamique | http://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE\_TYPE=BlastSearch&PROG\_DEF=blastn&BLAST\_PROG\_DEF=blastn&BLAST\_SPEC=GlobalAln&LINK\_LOC=BlastHomeLink |
| AVID [Bray et al., 2003] | approche par ancrage | http://baboon.math.berkeley.edu/avid\_supplementary/avid.html |
| LAGAN [Brudno et al., 2003] | approche par ancrage | http://genome.lbl.gov/cgi-bin/VistaInput?align\_pgm=lagam&num\_seqs=2 |
| MUMMER [Delcher et al., 2002] | approche par ancrage | http://mummer.sourcegorge.net/ |
| ACANA [Huang et al., 2006] | approche par ancrage | http://biomedempire.org |
| NGILA [Cartwright, 2007] | Programmation dynamique | http://scit.us/projects/ngila/ |

Tableau ‎1‑3 Les algorithmes globaux d'alignement par paires.

* *Algorithme Needleman et Wunch :* L’algorithme de *Needleman-Wunsch* [Needleman et Wunsch, 1970] utilise la technique d’alignement global afin de fournir un score maximal d’un alignement optimal. Etant donné, deux séquences et , un alignement optimal est réalisé progressivement en trois étapes : d’abord, on construit une matrice de substitution M*(n+1)(m+1)* en plaçant verticalement la et horizontalement la et en fixant la valeur de *Coût-brèche* à une constante de départ. L’ensemble de cases M (avec) contient les scores de substitution de la avec la . Puis, on initialise les cases M par la multiplication de et les cases M par la multiplication de. Ensuite, on remplit la matrice M, progressivement, à partir de l'angle supérieur gauche vers le coin inférieur droit de manière à ce que chaque case M contient le maximum entre les valeurs de M, M et M (voir Équation ‎1‑5). Enfin, pour trouver l’alignement optimal avec le score maximum, il suffit de partir de l’élément (), et effectuer le « *chemin inverse* » vers l’élément en regardant à chaque étape à partir de quel case on est parti. S'il s'agissait de l'élément diagonal, alors et sont alignés. S'il est l'élément M, alors est alignée avec une *brèche* et s'il est l'élément M, alors la est alignée avec une *brèche* (voir Figure ‎1‑9). La Figure ‎1‑8 et la Figure ‎1‑9 présentent un exemple d’alignement d’une séquence  et d’une autre séquence en fixant le *Coût-brèche* .

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M | | | | Équation ‎1‑5 | | |
| Avec : est le score de similarité entre le caractère et le caractère proposé par Needleman et Wunsch. | | | | | | | |
|  |  | | | |
| Figure ‎1‑8 La matrice M de Needleman & Wunch [Wang, 2007] | | | | |
|  | | | | |
|  | | **Alignement optimal** | | |
| |  |  |  |  | | --- | --- | --- | --- | | **A** | **A** | **G** | **-** | | **|** | **|** | **|** | **|** | | **-** | **A** | **G** | **C** | | | |  |  |  |  | | --- | --- | --- | --- | | **A** | **A** | **G** | **-** | | **|** | **|** | **|** | **|** | | **A** | **-** | **G** | **C** | |
| Figure ‎1‑9 Alignement global Needleman & Wunch [Wang, 2007] | | | | |

La complexité de l’algorithme de *Needleman* et *Wunch* est de l’ordre de O(). Cet algorithme est sensible aux paramètres de similarité et de pénalité fournis avec la matrice de substitution. En outre, il peut fournir des scores négatifs.

* *Algorithme Smith & Waterman :* En 1981, *Smith* et *Waterman* [Smith et Waterman, 1981] proposent un algorithme d’alignement local afin de résoudre les limites présentés par *Needleman* et *Wunch* [Needleman et Wunsch, 1970]. L’algorithme de *Smith* et *Waterman* calcule les scores positifs en éliminant les scores négatifs et en considérant n’importe quelle case de la matrice M comme un point de départ ou un point d’arrivée. L’idée consiste à supprimer, d’abord, tous les éléments négatifs de la matrice M en les remplaçants par zéro. Puis, il applique une recherche complète sur l’élément maximal dans M afin de le considérer comme un point de départ. Enfin, il trace l’alignement optimal jusqu’à ce qu’il arrive à n’importe quelle case disposant un score égal à zéro. L’algorithme de *Smith et Waterman* donne de bons résultats avec deux protéines de haute divergence. Cependant, avec l’absence des scores négatifs, l’algorithme peut perdre la qualité de l’alignement ce qui engendre la présence d’un alignement global.

### Alignement Multiple de Séquences

L’alignement multiple de séquence, également appelé *MSA*, est un type d’alignement beaucoup plus complexe que l’alignement par paire. Il permet de découvrir les similarités et les régions conservées entre un ensemble de séquences (avec ) [Hubbard et al., 1996; Gusfield, 1997; Durbin et al., 1998]. Son objectif est de trouver le meilleur alignement multiple entre ces séquences.

MSA est considéré comme un problème NP-complet [Carrillo et Lipman, 1988; Elias, 2006]. Il couvre deux sous-problèmes : le *MSA global* et le *MSA local* (voir Figure ‎1‑10). Malgré qu’il dispose un temps de calcul élevé, l’alignement multiple donne plus d’informations, biologiquement, significatives que l’alignement par pair. En effet, l’alignement multiple est une technique préalable et importante pour l’analyse des données biologiques. Elle est utilisée dans plusieurs recherches comme :

* La réalisation des arbres phylogénétiques.
* La caractérisation des familles de séquences.
* La recherche d’homologie dans une base de données.
* La découverte de l’évolution des organismes vivants réalisés par un certain ancêtre au passé.
* L’identification des sites fonctionnels et structurels d’une nouvelle famille de protéines.
* La prédiction des structures 3D [Gusfield, 1997 ; Durbin et al., 1998 ; Higgins et Taylor, 2002]

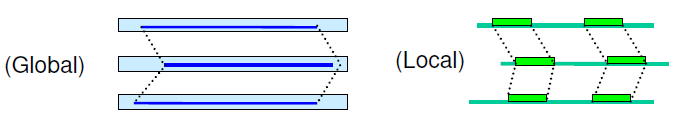


Figure ‎1‑10 Comparaison entre un alignement global et un alignement local [Wang, 2007]

Formellement, étant donné un ensemble de séquence (avec définies sur l’alphabet , un alignement multiple est un ensemble de séquence définies sur l’alphabet ′ ( avec ′ tel que ′ = ) satisfaisant les propriétés suivantes :

* + les séquences disposant la même longueur .
  + aucune colonne n’est constituée uniquement de *brèches*.
  + l’alignement (resp. ) permet toujours de retrouver l’ensemble (resp.

***Classification des algorithmes***

En raison de l’importance biologique de MSA, plusieurs algorithmes ont été élaborés pour résoudre le problème d’alignement (voir Figure ‎1‑11). Étant donné la complexité combinatoire de MSA, les algorithmes exacts se trouvent incapables d’analyser un grand nombre de séquences [Notredame, 2002]. De ce fait, les recherches se sont orientées vers des techniques heuristiques. Plusieurs approches ont été adoptées comme l’*approche progressive*, l’*approche itérative*, l’*approche hybride*, l’*approche diviser-et-conquérir* et l’*approche à base de bloc.*

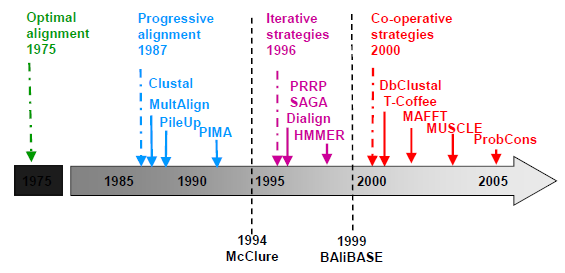


Figure ‎1‑11 Les algorithmes MSA [Lecompte, 2007]

#### L’approche progressive

Cette approche offre une solution progressive en se basant sur des critères précis et sur un ordre bien déterminé. Proprement dit, le principe consiste à générer au départ un alignement entre deux séquences puis ces alignements sont à leurs tours alignés entre eux jusqu’à ce que toutes les séquences soient regroupées. La différence entre les algorithmes dépend de la fonction de score ou de type de la matrice utilisée. Il existe plusieurs techniques utilisées dans la construction des alignements progressifs. Parmi ces techniques, nous citons les *arbres de guidage* [Saitou et Nei, 1987]. L’avantage des algorithmes MSA progressifs réside dans la rapidité des alignements fournit. Toutefois, ils possèdent plusieurs inconvénients : la dépendance entre la qualité des résultats et la qualité des méthodes utilisées (score, ordre…), la création d’une brèche se fait d’une manière définitive « *once a gap, ever a gap* » et la possibilité d’avoir qu’un alignement à deux dimensions.

Le premier algorithme progressif a été développé par *Hogeweg* et *Hesper* [Hogeweg et Hesper, 1984]. En réalisant une phylogénie sur un ensemble de séquences, l’algorithme de *Hogeweg* et *Hesper* applique un alignement par paire sur deux séquences. Puis, il ajoute progressivement les autres jusqu’à ce qu’il aligne toutes les séquences. Cet algorithme a été amélioré en 1990 par *Feng* et *Doolittle* [Feng et Doolittle, 1990]. Il utilise deux notions : la notion de profil pour réaliser les alignements et la notion de « *arbre de guidage*» pour indiquer un ordre convenable pour un tel alignement. Par ailleurs, *alignement par profils* est un alignement intermédiaire dans l’alignement progressif dont une ligne d’un profil est égale à une colonne d’alignement. Il autorise à chaque itération un seul type d’alignement : soit l’alignement séquence par séquence, l’alignement séquence par profil ou l’alignement profil par profil[Heringa, 1999].

Parmi les algorithmes adoptant l’approche progressive tels que :

* *CLUSTAL* est l’algorithme le plus utilisé dans les alignements multiples [Higgins et Sharp, 1988]. Il est souvent utillisé avec les séquences d’ADN ou d’ARN. En appliquant l’approche progressive, l’idée de *CLUSTAL* est de construire un alignement en intégrant progressivement les séquences les plus éloignées à partir des séquences les plus proches. Son principe repose sur la génération des alignements entre toutes les paires possibles. Puis, il utilise un algorithme basique de phylogénie, appelé *UPGMA*, pour construire un arbre de guidage en se basant sur une matrice des scores d’alignement par paire M disposant une taille égale à . Enfin, il aligne les séquences progressivement en respectant l’ordre établis par cet arbre (voir Figure ‎1‑12). Il existe plusieurs améliorations de CLUSTAL. Parmi ces versions les plus connues, nous citons *CLUSTALW* [Thompson et al., 1994], *CLUSTALX* [Thompson et al., 1997], *DBCLUSTAL* [Thompson et al., 2000] *et CLUSTAL Omega* [Sievers et al., 2011]*.* L’inconvénient de *CLUSTAL* réside dans le fait que la qualité de l’alignement peut être affectée par des erreurs faites au début du processus de l’alignement. Ces erreurs ne peuvent être corrigées par la suite. En outre, l’utilisation des arbres de guidage peut influer la précision des alignements [Nelesen et al., 2008], ainsi, la méthode utilisée dans la construction d’arbre n’est pas fiable qu’avec des séquences peu divergentes [Philippe, 2005].

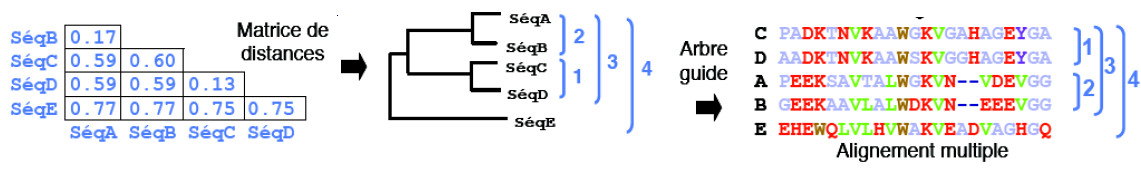


Figure ‎1‑12 Algorithme de *CLUSTAL*

* *T-COFFEE* a été développé par *Notredame et al.* [Notredame et al., 2000]. L’idée principale consiste à combiner les résultats de *CLUSTALW* [Thompson et al., 1994] et *DIALIGN* [Morgenstern, 1999]. Il utilise une bibliothèque disposant tous les alignements locaux (*DIALIGN*) et globaux (*CLUSTALW*) possibles entre les paires de séquences. Cette bibliothèque est utilisée pour guider l’alignement progressif et pour chercher un *MSA* basé sur la consistance des alignements par paire. L’avantage est la production du meilleur alignement, mais son inconvénient est qu’il est trop lent.
* *Autres* : Il existe d’autres algorithmes adoptant l’approche progressive tels que *MAVID* [Bray et Pachter, 2004], *KALIGN* [Lassman et Sonnhammer, 2005], *PRALINE* [Simossis et Heringa, 2005], *SPEM* [Zhou et Zhou, 2005], *PSALIGN* [Sze et al., 2006], *COBALT* [Papadopoulos et Agarwala, 2007], *GRAMALIGN* [Russell et al., 2008], *PicXAA* [Sahraeian et Yoon, 2010] et *IMSA* [Cutello et al., 2011].
* *L’approche itérative*

Les algorithmes itératifs tentent d’améliorer le score de l’alignement global et d’éliminer l’inconvénient majeur des algorithmes progressifs cités précédemment. Son principe consiste à sélectionner des sous-groupes de séquences en utilisant un arbre phylogénétique. Puis, il applique des modifications sur leur alignement afin d’obtenir un nouveau alignement. Enfin, il aligne les séquences d’une manière itérative et globale. Le processus d’alignement s’arrête si la convergence est établie après une telle modification, soit par insertion/délétion d’un ou plusieurs brèches dans l’alignement, soit par l’exclusion des séquences alignées séparément. Son avantage est l’amélioration de la qualité des alignements. Toutefois, elles sont plus coûteuses en termes de temps de calcul par rapport à l’approche progressive. L’algorithme le plus utilisé est *DIALIGN* [Morgenstern, 1999].

* *DIALIGN* [Morgenstern, 1999]génère un alignement multiple (global ou local) à partir des diagonales pondérées en se basant sur des similarités locales et en cherchant un alignement avec une somme de poids maximale. Le score de *DIALIGN* est égal à la somme des scores des diagonales en négligeant les pénalités de gap. En effet, *DIALIGN* donne de bons résultats avec un ensemble de séquences très divergentes ou de longueurs différentes. Il existe des améliorations de *DIALIGN* comme *DIALIGN-T* [Subramanian et al., 2005] *DIALIGN-TX* [Subramanian et al., 2008].
* *Autres* : il existe d’autres algorithmes itératifs basés sur le *modèle de Markov caché (HMM)* *: HMMER* [Eddy, 1995], *SAGA* [Notredame et Higgins, 1996], *SAM* [Karplus et al., 1998], *SARCHMO* [Edgar et Sjolander, 2003], *QOMA* [Zhang et Kahveci, 2007] *et FSA* [Bradley et al., 2009].
* *L’approche hybride*

En se basant sur les inconvénients des approches cités précédemment, plusieurs améliorations ont été élaborées. Nous citons les solutions qui utilisent l’approche progressive en ajoutant une étape de raffinement basée sur l’approche itérative pour produire une nouvelle approche appelée l’*approche hybride*. Parmi ces dernières, nous citons :

* *MAFFT* est un algorithme d’alignement multiple qui utilise la fusion d’une approche progressive avec une approche itérative. Il est réalisé en 2002 par *Katoh et al* [Katoh et al., 2002]. Il est basé sur la *transformation de Fourier*. *MAFFT* se décompose en deux étapes principales. En premier lieu, il utilise l’*approche* *progressive* afin de générer un alignement d’une manière rapide. En deuxième lieu, il améliore cet alignement en appliquant un raffinement basé sur l’*approche* *itérative*. Il permet de produire les meilleurs alignements par rapport aux algorithmes progressifs et itératifs [Nuin et al., 2006; Golubchik et al., 2007; Dessimoz et Gil., 2010; Letsch et al., 2010; Sahraeian et Yoon., 2011; Sievers et al., 2011]. En effet, *Katoh et al* présente sept améliorations de MAFFT [Katoh et al., 2005; Katoh et al., 2009; Katoh et Frith, 2012; Katoh et Standley, 2013]
* *MUSCLE* est réalisé par *Edgar* en 2004 [Edgar, 2004]. Il est dédié pour aligner les séquences protéiques. D’abord, il utilise un alignement progressif en ajoutant une nouvelle fonction de profil appelée *score log-attente*. Puis, il applique un raffinement avec un partitionnement restreint en utilisant un *arbre de dépendance*. L’avantage de *MUSCLE* est la qualité de l’alignement généré. Son inconvénient est le temps d’exécution et la complexité très élevée. Il existe une version rapide de *MUSCLE* qui a été implémente par *Edgar* [Edgar, 2004] appelée *MUSCLE-fast*.
* *MULTALIN* est un algorithme développé par *Corpet* [Corpet, 1988]. Il applique la technique de regroupement hiérarchique des séquences dans son processus d’alignement. Il utilise un alignement par paire pour générer un ensemble de *groupes* (*cluster*). Ces groupes sont utilisés pour construire l’*arbre de guidage*.
* *Autres* : Il existe d’autres algorithmes adoptant l’*approche hybride* tels que *PRRP* [Gotoh, 1996], *MULTI-LAGAN* [Brudno et al., 2003], *PROBCONS* [Do et al., 2005] et *MSAID* [Min et al., 2005].
* *L’approche Diviser-et-conquérir (DAC)*

L’approche DAC est un processus d’alignement simultané [Stoye et al., 1997]. Il est, souvent, réparti en trois étapes présentées comme suit : D’abord, on divise chaque séquence en deux sous-séquences : un préfixe et un suffixe pour obtenir deux nouvelles familles : la famille des préfixes et la famille des suffixes. Puis, on applique ce processus d’une manière récursive jusqu’à obtenir un alignement optimal entre les *sous-séquences* générées. Enfin, le résultat d’alignement de séquences est obtenu avec la concaténation des sous-alignements donnés.

* *BlockMSA*[Wang et al., 2007] : il s’agit d’une première initiative d’utiliser un algorithme de *biregroupement* dans la résolution d’un problème d’alignement multiple. *BlockMSA* est une heuristique basée sur l’*approche DAC*. Elle permet de présenter un problème de *MSA* comme étant un problème de *biregroupement* afin de produire un alignement multiple et local à partir d’un ensemble de séquences biologiques spécifique à l’*ARNs*. Nous détaillons *BlockMSA* dans le chapitre suivant.
* *Autre :* Parmi les algorithmes qui adoptent cette approche, nous citons : *DCA* [Stoye, 1998].
* *L’approche à base de blocs*

C’est une approche d’optimisation globale. Son objectif est d’optimiser le score SP à partir d’un ensemble de blocs afin de rechercher une solution globale et optimale. L’idée consiste à, d’abord, utilise une méthode heuristique pour fournir un ensemble de blocs corresponds aux régions conservées. Puis, elle construit la chaine des blocs et elle recherche un ensemble optimal de blocs disposant la somme maximale des scores de similarité. Enfin, elle présente ces blocs sous forme de régions conservées. L’inconvénient de cette approche est qu’elle consomme beaucoup de temps d’exécution dans la recherche des blocs. Plusieurs algorithmes ont été proposés dans la littérature. Parmi ces algorithmes, nous citons :

* *Block-Maker* [Henikoff et al., 1995] C’est l’algorithme le plus connu dans l’utilisation de l’*approche à base de blocs*. L’algorithme *Block-maker* utilise deux méthodes pour la construction de la chaine des blocs. L’une est basée sur le « *Space-triplet* » [Henikoff et al., 1995]  et l’autre sur « *Gibbs-Sampling* » [Roth et al., 1998; Liu et al., 2001; Higgins et Taylor, 2002]. D’une part, la méthode « *Space-triplet* » applique une recherche exhaustive sur tous les triplets pour trouver une distance maximale entre les sous-séquences. D’autre part, la méthode « *Gibbs-Sampling* » consiste à réaligner, itérativement, un ensemble de séquences jusqu’à maximiser un seuil de probabilité. En effet, la première méthode génère des résultats non précis à cause de la stratégie utilisée tandis que la deuxième peut simplement ignorer certains blocs optimaux. Ainsi, elles disposent les deux méthodes disposent d’un coût très élevé en termes de temps de calcul et de complexité.
* *Autres approches*

Il existe d’autres approches d’alignement. Parmi ces derniers, il y a l’*Approche à base des informations externes (IE)*.En se basant sur l’approche progressive, elle améliore la signification biologique des alignements en utilisant des informations biologiques liées aux séquences ou bien des informations définies par l’utilisateur. Parmi les algorithmes qui adaptent cette approche, nous citons *EXPRESSO* [Armougom et al., 2006] et *PROMALS* [Pei et Grichin, 2007]. Par ailleurs, nous citons l’*approche à base des connaissances phylogéniques* (*Phylogeny-aware*)*.* Cette approche est utilisée dans l’analyse de la phylogénie des séquences afin d’obtenir un alignement biologiquement correct. Parmi les algorithmes utilisant cette approche, nous citons *PRANK* [Löytynoja et Goldman, 2010] et *TRS* [Adam et al., 2013].

Le Tableau ‎1‑4 et Tableau ‎1‑5 le fournissent un récapitulatif sur les algorithmes locaux et globaux de MSA.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Algorithmes Globaux** | **Approches** | **Liens** |
| **1988** | CLUSTAL [Higgins et Sharp, 1988] | Progressive | http://www.clustal.org/ |
| MULTALIN [Corpet, 1988] | Hybride | http://bioinfo.genotoul.fr/multalin.html |
| **1994** | CLUSTALW [Thompson et al., 1994] | Progressive | http://www.clustal.org/ |
| **1995** | HMMER [Eddy, 1995] | Itérative | http://hmmer.janclia.ord |
| **1996** | PRRN [Gotoh, 1996] | Hybride | http://prrn.hgc.jp |
| SAGA [Notredame et Higgins, 1996] | Itérative | http://www.tcoffe.org/Projects\_home\_page/  saga\_home\_page.html |
|  | CLUSTALX [Thompson et al., 1997] | Progressive | http://www.clustal.org/ |
| **1998** | DCA [Stoye, 1998] | DAC | http://bibiser.techfak.unibielefeld.de/  dca/submission.html |
| **2000** | DBCLUSTAL [Thompson et al., 2000] | Progressive | ftp://ftp-igbmc.u-strasbg.fr/pub/  DbClustal |
| T-COFFEE [Notredame et al., 2000] | Progressive | http://www.tcoffe.org |
| **2003** | MULTI-LAGAN [Brudno et al., 2003] | Hybride | http://lagan.stanford.edu/lagan\_web/  index.shtml |
| SARCHMO [Edgar et Sjolander, 2003] | Itérative | http://phylogenomics.berkeley.edu/q/satchmo/ |
| **2004** | MUSCLE [Edgar, 2004] | Progressive | http://www.drive5.som/muscle |
| MUSCLE-fast [Edgar, 2004] |  |  |
| MAVID [Bray, et Pachte, 2004] | Progressive | http://baboon.math.berkeley.edu/mavid |
| **2005** | KALIGN [Lassman et Sonnhamme, 2005] | Progressive | http://www.ebi.ac.uk/Tools/kalign/ |
| MSAID [Min et al., 2005] | Hybride | - |
| PRALIGN [Simossis et Heringa, 2005] | Progressive | http://www.ibi.vu.nl/programs/  praline |
| PROBCONS [Do et al., 2005] | Hybride | http://probcons.stanford.edu/index.html |
| SPEM [Zhou et Zhou, 2005] | Progressive | http://spark.informatics.iupui.edu/  Softwares-Servuces\_files/spem.htm |
| **2006** | EXPRESSO [Armougom et al., 2006] | Progressive | http://www.tcoffe.org |
| PSALIGN [Sze et al., 2006] | Progressive | http://faculty.cs.tamu.edus/shsze/psalign |
| **2007** | COBALT [Papadopoulos et Agarwala, 2007] | Progressive | ftp://ftp.ncbi.nlm.nih.gov/pub/  agarwala/cobalt |
| **2008** | GRAMALIGN [Russell et al., 2008] | Progressive | http://bioinfo.unl.edu/GramAlign.html |
| **2009** | FSA [Bradley et al., 2009] | Itérative | http://orangutan.math.berkeley.edu/fsa |
| **2010** | PRANK [Löytynoja et Goldman, 2010] | Phylogeny-aware | http://www.ebi.ac.uk/goldman-srv/webprank/ |
| PicXAA (Sahraeian et Yoon, 2010) | progressive | http://www.ece.tamu.edu/∼bjyoon/picxaa/ |
| **2011** | CLUSTAL Omega [Sievers et al., 2011] | Progressive | http://www.clustal.org/ |
| **2013** | MAFFT [Katoh et Standley, 2013] | Hybride | http://align.bmr.kyushu-u.ac.jp/mafft/online/server |
| TRs [Adam et al., 2013] | Phylogeny-aware | - |

Tableau ‎1‑4 Les algorithmes globaux de MSA

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Algorithmes Locaux** | **Approches** | **Liens** |
| **1995** | Block-Maker [Henikoff, et al., 1995] | A based de blocs | http://blocks.fhcrc.org |
| **1998** | SAM [Karplus et al., 1998] | Itérative | - |
| **1999** | DIALIGN [Morgenstern, 1999] | Itérative | http://bibiser.techfak.unibielefeld.de/  dialign/submission.html |
| **2005** | QOMA [Zhang et Kahveci, 2007] | itérative | - |
| DIALIGN-T [Subramanian et al., 2005] | Itérative | http://bibiser.techfak.unibielefeld.de/  dialign/submission.html |
| **2007** | BlockMSA [Wang et al., 2007] | DAC | http://aug.csres.utexas.edu/msa/ |
| PROMALS [Pei et Grichin, 2007] | IE | http://prodata.swmed.edu/promals/promals.php |
| **2008** | DIALIGN-TX [Subramanian et al., 2008] | Itérative | http://bibiser.techfak.unibielefeld.de/  dialign/submission.html |
| **2011** | IMSA [Cutello et al., 2011] | progressive | - |

Tableau ‎1‑5 Les algorithmes locaux de MSA

## Conclusion

Dans ce chapitre, nous avons présenté l’alignement d’une manière générale et plus spécifiquement l’alignement multiple de séquences (*MSA*). Toutefois, le *MSA* est un problème NP-complet, ce qui a donné naissance à la réalisation des heuristiques qui tentent de le résoudre d’une manière optimale. Par ailleurs, plusieurs algorithmes globaux ou bien locaux ont été proposés. Néanmoins qu’ils sont encore incapables de fournir des résultats optimaux et efficaces quelques soit le type de séquences biologiques.

De ce fait, plusieurs innovations ont été réalisées. Parmi ces innovations, nous citons l’algorithme de *Wang et al* [Wang et al., 2007]. Il s’agit d’une seule initiative, à présent, qui utilise le *biregroupement* pour la résolution du problème *MSA*. *Wang et al* [Wang et al., 2007]proposent l’algorithme *BlockMSA* basé sur la technique du *biregroupement*. Cet algorithme fournit des bons résultats avec les séquences d*’ARN.* En appliquantle *biregroupement,* il dévoile facilement et efficacement les régions similaires, ainsi, il produit des alignements possédant des significations biologiques. En s’inspirant de *BlockMSA*, nous présentons, d’abord, dans le chapitre 2 le problème de biregroupement en précisant ses capacités dans la résolution des problèmes d’alignement multiple de séquences biologiques. Ensuite, nous élaborons notre algorithme d’alignement multiple de séquences dans le chapitre 3.

# Biregroupement de Données Biologiques

## Introduction

L’un des principaux défis de la *bioinformatique* est l’extraction de connaissances à partir des bases de données biologiques [Martinez et al., 2006]. Afin d'analyser ces quantités importantes de données, plusieurs techniques d'extraction de données ont été établies. Dans ce cadre, *Fayyad et al.* ont proposé le processus d’extraction des connaissances : *ECD* (***E****xtraction des* ***C****onnaissances à partir des* ***D****onnées*) (*en anglais KDD, Knowledge Discovery in Databases)* [Fayyad et al., 1996] (voir Figure 2-1). C’est un processus multidisciplinaire qui permet la découverte non triviale, à partir de données, des informations implicites précédemment inconnues et potentiellement intéressantes [Frawley et al., 1992]. Plusieurs techniques ont été appliquées à ces données en vue d'en extraire des connaissances pertinentes [Wang *et al.*, 2005].

**Base**

**de**

**Données**

Les connaissances

Figure ‎2‑1 Processus ECD

Parmi les techniques utilisées, nous citons celle du *regroupement* [Quackenbush, 2006]. En effet, en faisant du *regroupement*, nous considérons que tous les individus d'un groupe présentent des comportements similaires sous tous les attributs. Cependant, il existe des individus qui ne présentent des comportements similaires que sous un sous-ensemble d'attributs. Ceci permet de conclure que le regroupement est trop simpliste pour détecter de tels cas [Prelic *et* al., 2006]. Une autre technique plus intéressante (voir Figure ‎2‑2), appelée la technique de *biregroupement* [Cheng et Church, 2000; Madeira et Oliveira, 2004], permet l'identification de groupes d'individus qui présentent des formes d'expression cohérentes à travers des groupes spécifiques d'attributs. Cette intéressante technique de fouille de données a connu un développement considérable durant ces dernières années en raison de ses capacités d’apporter des réponses aux nouveaux défis soulevés par l'analyse de données biologiques tels que les données d'expression de gènes [Ayadi et al., 2009] et récemment l’alignement de séquences [Wang et al., 2007].

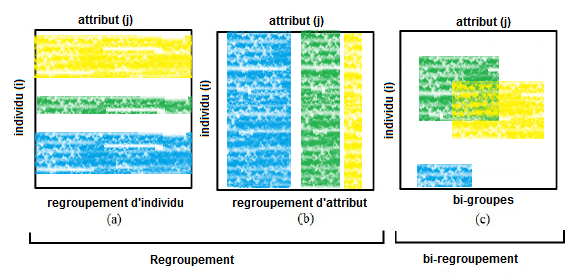


Figure ‎2‑2 La différence entre le regroupement et le biregroupement.

Dans ce chapitre, nous présentons un état de l’art sur le biregroupement en décrivant son principe et les différents types et structures de *bigroupes*. Nous détaillons aussi les différents algorithmes systématiques et stochastiques existants. Nous terminons ce chapitre par une conclusion

## Définitions et Notations

1. *(Individu) : un individu est un ensemble de motifs composés par des caractères appartenant à un alphabet .*
2. *(Attribut) : c’est une variable associée à un individu. Un attribut est « quantitatif » s’il prend des valeurs numériques. Il est « qualitatif » s’il prend des valeurs non-numériques.*
3. *(Pondération) : la pondération est l’intersection d’une ligne « individu » et d’une colonne « attribut » dans une matrice M. Il existe plusieurs types de pondération comme la pondération booléenne et la pondération par fréquence des occurrences* [Sebastiani, 2005].

## Biregroupement

Le *biregroupement* (*biclustering*) consiste à faire simultanément un regroupement de lignes (*individus*) et de colonnes (*attributs*) dans une matrice M afin d’identifier des groupes d’*individus* en cohérence avec des groupes d’*attributs* [Cheng et Church, 2000; Madeira et Oliveira, 2004]. Cette cohérence peut souvent aider à dévoiler leur comportement similaire et leurs fonctions biologiques proches [Wang et al., 2002; Sharan et al., 2002; Quackenbush, 2006; Divina et Aguilar-Ruiz, 2007; Pontes et al., 2007].

Formellement, le biregroupement permet de produire un ensemble de *bigroupes* à partir d’une matrice M ( où est le nombre d’individus et est le nombre d’attributs. Une case M présente la *pondération* du attribut avec le individu (voir Tableau ‎2‑1).

Bigroupe

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | Attribut1 | Attribut 2 | … | Attribut J | … | Attribut M | | Individu1 | M | M | … | M | … | M | | Individu2 | M | M | … | M | … | M | | … | … | … | … | … | … | … | | Individu i | M | M | … | M | … | M | | … | … | … | … | … | … | … | | Individun | M | M | … | M | … | M | |

Tableau ‎2‑1 La matrice M

Par ailleurs, l’objectif majeur de biregroupement est de produire un ensemble de *bigroupes* cohérents, stables et homogènes. La découverte de ces *bigroupes* fournit des résultats qui aident à discerner de nouvelles connaissances comme l’annotation fonctionnelle d’individu [Gibbons et Roth, 2002; Gonçalves, et al., 2009].

Généralement, le problème de biregroupement est NP-difficile [Cheng et Church, 2000; Madeira et Oliveira, 2004]. Il dispose d’un espace de recherche de l’ordre de (avec est l’ensemble d’individus et est l’ensemble d’attributs). Pour résoudre ce problème, plusieurs méthodes heuristiques ont été proposées. Cependant, la majorité d’entre elles ne garantissent pas l’optimalité de solutions. En effet, le problème de biregroupement peut être formulé comme suit [Ayadi et Elloumi, 2011] :

|  |  |
| --- | --- |
|  | Équation ‎2‑1 |

Avec:

f : est une fonction objectif mesurant la qualité, i.e., le degré de cohérence, d'un groupe de bigroupes

BC(M) est l'ensemble de tous les groupes de bigroupes possibles associés à M.

Par ailleurs, le biregroupement a été massivement utilisé dans un certain nombre de domaines tels que la *recherche de marché*s [Gaul et Schader, 1996], la *fouille de texte* [Berkhin et Becher, 2002; Dhillon et al., 2003]*,* le *marketing cible* [Hofmann et Puzicha, 1999; Wang et al., 2002], la *Biologie Moléculaire,* spécifiquement, l’*annotation fonctionnelle* d’individus [Tanay et al., 2002; Gibbons et Roth, 2002; Segal et al., 2003; Gonçalves et al., 2009], l’analyse d’*expression de gènes* [Yong et al., 1998; DiMaggio et al., 2008; Jose et al., 2013; Cheng et al., 2013], la *discrimination* entre familles de protéines [Mhamdi et Elloumi, 2009] et l’alignement multiple de séquences biologiques [Wang et al., 2007].

## Les Bigroupes

Les *bigroupes* représentent un sous-ensemble d’individus ayant un même comportement d’un sous-ensemble d’attributs. Formellement, un bigroupe peut être défini comme suit : Etant donné un ensemble de individus, un ensemble de attributs et M une matrice de données associée à et . Un bigroupe associé à une matrice de données M est un couple tel que et (voir Tableau ‎2‑1). Il est caractérisé par des types et des structures bien définies [Madeira et Oliveira, 2004]. Notant qu’un individu/attribut peut appartenir à aucun ou plusieurs bigroupes.

### Les types de bigroupes

Les différents types de bigroupes sont classés comme suit:

* Les bigroupes à valeurs constantes : Les bigroupes à valeurs constantes présentent les sous-matrices composées par des valeurs identiques dans certaines cases. On distingue trois types (voir *Tableau ‎2‑2*) : l*es bigroupes à une seule valeur constante, les bigroupes à valeurs constantes en lignes et les bigroupes à valeurs constantes en colonnes*.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à une seule valeur constante | | | | | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à valeurs constantes en lignes | | | | | | **1.0** | **1.0** | **1.0** | **1.0** | **1.0** | | **2.0** | **2.0** | **2.0** | **2.0** | **2.0** | | **3.0** | **3.0** | **3.0** | **3.0** | **3.0** | | **4.0** | **4.0** | **4.0** | **4.0** | **4.0** | | **4.0** | **4.0** | **4.0** | **4.0** | **4.0** | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à valeurs constantes en colonnes | | | | | | **1.0** | **2.0** | **3.0** | **4.0** | **5.0** | | **1.0** | **2.0** | **3.0** | **4.0** | **5.0** | | **1.0** | **2.0** | **3.0** | **4.0** | **5.0** | | **1.0** | **2.0** | **3.0** | **4.0** | **5.0** | | **1.0** | **2.0** | **3.0** | **4.0** | **5.0** | |
| **Tableau ‎2‑2 Bigroupes à valeurs constantes** | | |

* Les bigroupes à valeur cohérente : C’est un bigroupe avec un modèle additif (respectivement multiplicatif) (voir Tableau ‎2‑3).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à valeurs cohérentes (addition) | | | | | | 1.0 | 4.0 | 5.0 | 0.0 | 1.5 | | 4.0 | 7.0 | 8.0 | 3.0 | 4.5 | | 3.0 | 6.0 | 7.0 | 2.0 | 3.5 | | 5.0 | 8.0 | 9.0 | 4.0 | 5.5 | | 2.0 | 5.0 | 6.0 | 1.0 | 2.5 | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à valeurs cohérentes (multiplication) | | | | | | 1.0 | 0.5 | 2.0 | 0.2 | 0.8 | | 2.0 | 1.0 | 4.0 | 0.4 | 1.6 | | 3.0 | 1.5 | 6.0 | 0.6 | 2.4 | | 4.0 | 2.0 | 8.0 | 0.8 | 3.2 | | 5.0 | 2.5 | 10.0 | 1.0 | 4.0 | |
| **Tableau ‎2‑3 Bigroupes à valeurs cohérentes** | |

* Les bigroupes à évaluation cohérente : On trouve trois types de bigroupes à évaluation cohérente (voir Tableau ‎2‑4) présentés comme suit [Ben-Dor et al., 2002; Tanay et al., 2002] :
* *Les bigroupes à évaluation cohérente*.
* *Les bigroupes à évaluation cohérente avec une condition de niveaux*.
* *Les bigroupes à évaluations cohérentes avec une condition* *d’orientation*.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à évaluation cohérente (condition des niveaux) | | | | | | S1 | S1 | S1 | S1 | S1 | | S2 | S2 | S2 | S2 | S2 | | S3 | S3 | S3 | S3 | S3 | | S4 | S4 | S4 | S4 | S4 | | S5 | S5 | S5 | S5 | S5 | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Bigroupes à évaluation cohérente | | | | | | 70 | 13 | 19 | 10 | 40 | | 49 | 40 | 49 | 35 | 40 | | 40 | 20 | 27 | 15 | 70 | | 90 | 15 | 20 | 12 | 15 | | 95 | 16 | 14 | 12 | 70 | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Bigroupes à évaluation cohérente (condition d’orientation) | | | | | | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |
| Tableau ‎2‑4 Bigroupes à évaluation cohérente | | |

### Les structures de bigroupes

Les bigroupes extraits à partir des algorithmes de biregroupement peuvent être classés suivant trois structures présentées comme suit [Madeira et Oliveira, 2004] :

* *Bigroupe unique*. (voir Tableau ‎2‑5 /a))
* *Bigroupes avec exclusivité* (voir Tableau ‎2‑5: b), c), d))
* *Bigroupes avec ou sans chevauchement* (voir Tableau ‎2‑5 : e), f), g), h), i))

|  |  |  |
| --- | --- | --- |
| a) Bigroupe unique | b) Bigroupes avec exclusivité de lignes | c) Bigroupes avec exclusivité de colonnes |
|  |  |  |
| d) Bigroupes avec exclusivité de lignes et des colonnes | e) Bigroupes avec une structure d'arbre, sans chevauchement | f) Bigroupes sans chevauchement et sans exclusivité |
|  |  |  |
| g) Bigroupes avec une structure d'échiquier, sans chevauchement | h) Bigroupes avec une structure hiérarchique et chevauchement | i) Bigroupes avec positionnement arbitraire et chevauchement |
|  |  |  |

Tableau ‎2‑5 Les structures d’un bigroupe [Madeira et Oliveira, 2004]

## Fonctions d’évaluation

Les fonctions d’évaluation sont nécessaires pour mesurer le degré de cohérence d’un ensemble de bigroupes fournis par des algorithmes de biregroupement. En effet, il existe plusieurs solutions proposées pour l’évaluation de la qualité de bigroupes. Une étude bibliographique nous a permis d’identifier quelques fonctions d’évaluation présentées comme suit :

* *Fonction MSR :* La *Fonction MSR* (*Mean Squared Residue*) est la plus utilisée dans la littérature [Cheng et Church, 2000]. Elle est réalisée pour mesurer la cohérence de bigroupes. Plus la valeur de MSR est faible, plus les bigroupes deviennent cohérents. Par ailleurs, si un bigroupe dispose une valeur *MSR* inférieure à un certain seuil (voir Équation ‎2‑2) alors il est appelé -bigroupe. Parmi les algorithmes qui utilisent cette fonction on trouve celui de [Yang et al., 2003; Bleuler et al., 2004; Dharan et Nair, 2009; Liu et Wang, 2007; Das et al., 2010].

|  |  |
| --- | --- |
|  | Équation ‎2‑2 |
| Avec : est la moyenne de la totalité du bigroupe, est la moyenne de la colonne et est la moyenne de la ligne . | |

* *Fonction RV&V : Divana et Angiulli* [Angiulli et al., 2008; Divina et Aguilar-Ruiz, 2007] proposent une combinaison entre trois fonctions : *RV, MSR et EV.* Notant qu’un bigroupe disposant une grande *RV (variance de lignes)* est un bigroupe de bonne qualité (voir Équation ‎2‑3), *Divana et Angiulli* utilisent la fonction *RV* pour écarter les bigroupes de moindre qualité, ainsi, ils appliquent la fonction *MSR* [Cheng et Church, 2000] pour détecter les bigroupes de type constant (voir Équation ‎2‑2). En outre, ils exercent la fonction *EV* (*le volume)* pour détecter les bigroupes disposant une taille maximale (voir Équation ‎2‑4).

|  |  |
| --- | --- |
|  | Équation ‎2‑3 Variance de lignes RV |
|  | Équation ‎2‑4 Volume V |

### Fonction ACV : La fonction *ACV* (*Average Correlation Value)* a été proposée par *Teng et Chan* (Teng et Chan, 2008). Plus la valeur *ACV* (voir Équation ‎2‑5) est proche de , plus les bigroupes sont fortement cohérents. Cependant, la performance d’*ACV*se dégrade avec la présence de bruit dans les données. *Cheng et al* [Cheng et al., 2008] proposent d'utiliser les fonctions *ACV* et *MSR* en se basant sur la constatation qu’un bigroupe avec une forte cohérence dispose une petite valeur d’*MSR*et une grande valeur d’*MSR*.

|  |  |
| --- | --- |
|  | Équation ‎2‑5 ACV |
| Avec :  est le coefficient de corrélation de Pearson associé aux lignes d'indices et du bigroupe (I’; J’) [Myers et al., 2003], est le coefficient de corrélation de Pearson associé aux colonnes d'indices et du bigroupe (I’; J’) | |

* Fonction ASR : La fonction *ASR* (*Average Spearman Rho*) a été proposée par *Ayadi et al.* [Ayadi et al., 2009]. Elle est basée sur la *corrélation de Spearman* [Lehmann et D'Abrera, 1998]. Cette fonction est égale à la moyenne de toutes les *corrélations de Spearman* entre les composants d’un bigroupe. Puisqu’elle est basée sur la corrélation de Spearman, elle est donc robuste en cas de présence du bruit. La fonction ASR est définie comme suit :

|  |  |
| --- | --- |
|  | Équation ‎2‑6 ASR |
| Avec :  est la corrélation de Spearman associée aux indices des lignes et du bigroupe (I’, J’), est la corrélation de Spearman associée aux indices des lignes et du bigroupe (I’; J’) | |

## Les algorithmes de biregroupement

Durant les dernières années, le nombre d’algorithmes de biregroupement augmente d’une manière rapide. Cependant, ils se trouvent incapables de résoudre ce problème d’une manière optimale. Pour cela, on a recours à d’autres algorithmes heuristiques pour la construction optimale de bigroupes cohérents [Madeira et Oliveira, 2004].

Il existe deux classes d’algorithmes de biregroupement : les algorithmes systématiques et les algorithmes stochastiques. Dans la section suivante, nous présentons quelques algorithmes de biregroupement, leurs points forts et leurs limites.

### Les algorithmes systématiques

Les algorithmes systématiques de biregroupement adoptent l'une des approches suivantes : l’*approche diviser et conquérir(DAC)*,l’*approche itérative gloutonne (GIS)* et l’*approche énumérative (BE)*.

* *L’approche diviser et conquérir*

Le principe général de l’*approche diviser et conquérir* (***Divide and Conquer (DAC)***) est de diviser d'une manière itérative un problème en sous-problèmes en gardant les mêmes structures que celles du problème initial jusqu'à ce que ces sous-problèmes deviennent assez simples pour être résolus directement. Les résultats sont alors combinés pour créer une solution finale au problème de départ. Proprement dit, cette approche divise itérativement la matrice M de taille en sous-matrices afin de former un ensemble de bigroupes satisfaisants une fonction d’évaluation. L’avantage de cette approche est sa rapidité. Son inconvénient majeur est la possibilité de perdre les bons bigroupes avec le partitionnement avant l’identification [Prelic et al., 2006]. Parmi les algorithmes qui utilisent cette approche, on trouve*Block Clustering* [Hartigan, 1972]et *BiMax* [Prelic et al., 2006].

* *BiMAX (Binary Inclusion-MAXimal biclustering Algorithm)* [Prelic et al., 2006; Barkow et al., 2006] : Il divise une matrice avec un codage binaire en sous-matrices afin de minimiser le nombre de cellules nulles dans les bigroupes. Étant donné la capacité de cet algorithme dans la génération, sans pré-spécification, d’un nombre maximal de bigroupes, *BlockMSA* [Wang et al., 2007] utilise *BiMAX* [Barkow et al., 2006] dans la résolution du problème d’alignement multiple et local d’un ensemble d’*ARN* (voir Figure ‎2‑3). D’abord, il applique un processus d’identification des *blocs* afin d'identifier l'ensemble de régions (*fragments*) possibles conservées. Ces *fragments* ont été construits en divisant itérativement un ensemble d’alignement par paire réalisé par *CLUSTALW* [Thompson et al., 1994]. Ensuite, *BlockMSA* utilise *BiMAX* pour identifier des groupes de *sous-séquences* et les régions conservées parmi eux. Après l’application de biregroupement, cet algorithme applique, pour chaque sous-matrice de bigroupes, une étape de raffinement combinée avec un processus d’assemblage de blocs basé sur un *graphe orienté acyclique DAG (Directed Acyclic Graph)* [Dan, 1997]. En effet, *BiMAX* est dédié pour les matrices de données binaires. Son inconvénient majeur est sa complexité qui est de l’ordre de pour une matrice () avec est le nombre de bigroupes à inclusion maximale calculé par .

|  |  |
| --- | --- |
|  |  |
| Figure ‎2‑3 BlockMSA : Alignement en utilisant l’algorithme de biregroupement BIMAX [Wang et al., 2007] | |

* *L'approche de Recherche Itérative Gloutonne*

L’*approche de recherche itérative Gloutonne* (***G****reedy* ***I****terative* ***S****earch* ***(GIS)***) applique des processus d’ajout/suppression d’une manière itérative sur une matrice M. Son objectif est de produire des bigroupes qui maximisent ou bien minimisent certain critère. Autrement dit, cette approche construit, dans chaque itération, des sous-matrices de données en ajoutant/supprimant une ligne/colonne à la sous-matrice courante. Ces itérations s’arrêtent si aucune ligne ou colonne ne peut être ni ajoutée ni supprimée à aucune sous-matrice. L’approche *GIS* est rapide. Cependant, son inconvénient est qu’elle peut perdre les bons bigroupes avec la suppression avant l’identification. Il existe plusieurs algorithmes adoptants cette approche. Parmi ces derniers, on trouve *Spectral* OPSM[Ben-Dor et al., 2002]*,* [Klugar et al., 2003]*, OP-Clusters* [Liu et Wang, 2003]*, xMOTIFs* [Murali et Kasif, 2003]*,* *FLOC* [Yang et al., 2003]*, MSBE* [Liu et Wang, 2007]*,* δ-biclusters [Cheng et al., 2008]*, Teng et Chan* [Teng et Chan, 2008]et *RWB* [Angiulli et al., 2008].

* OPSM (Order-Preserving Sub-Matrix)[Ben-Dor et al., 2002] : Il considère un bigroupe comme une sous-matrice dont les lignes sont ordonnées par son propre algorithme *Order-Preserving Sub-Matrix*. Le principe de l’algorithme *OPSM* est d’ordonner (suivant une permutation et un ordre croissant) les premières et les dernières valeurs de chaque ligne. Globalement, son objectif est de faire croître itérativement la matrice pour avoir le meilleur *modèle complet[[1]](#footnote-1)*.
* FLOC (FLexible Overlapped biClustering) [Yang et al., 2003] : tout d’abord l’algorithme applique une certaine affectation des lignes (resp. des colonnes) à un bigroupe avec une probabilité *p*. Puis, il effectue une amélioration itérative de la qualité de bigroupes générés.
* δ-biclusters[Cheng et al., 2008]*:* d’abord, cet algorithme applique une procédure de recherche locale en commençant par la suppression itérative de lignes et de colonnes à partir de la matrice initiale jusqu'à ce que la valeur de son *MSR* (*Mean Squared Residue*) soit inférieure à un certain seuil δ. Ensuite, il ajoute, itérativement, des lignes et des colonnes en évitant l'augmentation de sa fonction *MSR*.

* *L'approche énumérative*

L’*approche d’énumération de bigroupes* (***B****icluster* ***E****numeration* ***(BE))*** applique une modélisation d’une matrice M similaire à la modélisation d’un graphe ou d’un arbre de recherche afin d’énumérer tous ses éléments. Son objectif est d’exercer d’une manière implicite ou explicite toutes les solutions possibles pour un problème donné. L’avantage de cette approche est la génération de la solution optimale. Tandis que, son inconvénient est l’importance de la complexité et le taux élevé de temps de calcul. Il existe plusieurs algorithmes adoptants cette approche. On peut citer *SAMBA* [Tanay et al., 2002], δ*-pClusters* [Wang et al., 2002], *Co-Clusters* [Cho et al., 2004], *CE-Tree* [Chen et Chang, 2009], BiMine [Ayadi et al., 2009], *Backtracking Bron-Kerbosch*[Oliver et al., 2012]et *HOCCLUS2*[Pio et al., 2013].

* SAMBA : En suivant l’*approche BE*, *Tanay et al.* [Tanay et al., 2002] transforment une matrice donnée en un graphe bipartie où un bigroupe est représenté par un sous-graphe évalué par une probabilité spécifique. L’objectif principal est de trouver tous les bigroupes ayant une bonne probabilité et de bonne qualité.
* BiMine *: BiMine* [Ayadi et al., 2009]permet de découvrir un ensemble de bigroupes en utilisant une nouvelle structure d'arbre d'énumération de bigroupes, appelée *Bicluster* *Enumeration* *Tree* *(BET)* et une nouvelle fonction d'évaluation basée sur la *corrélation de Spearman*, appelée *Average Spearman Rho (ASR)* [Ayadi et al., 2009]. Son processus de biregroupement se subdivise en trois étapes principales : *Prétraitement*, *Traitement* (*Construction de BET* et *Extraction des bigroupes) et Post-traitement*. D’abord, dans l’étape de *Prétraitement*, il prépare la matrice M en négligeant les pondérations M de moindre qualité. Une valeur de M est considérée négligeable si elle répond à l’Équation ‎2‑7 (voir Tableau ‎2‑6 et Tableau ‎2‑7). Ensuite, notant que *BET* possède comme racine un nœud vide. Chaque nœud du premier niveau est un bigroupe contenant un seul individu et l'ensemble d’attributs sous lequel il s'exprime. Le  nœud fils du nœud père est construit par l'union des individus du père et ceux du oncle en partant de la droite du père et de l'intersection des attributs du père et ceux du oncle. *BiMine* construit la structure *BET* selon une stratégie en largeur d'abord, afin de produire les nœuds fils en combinant un nœud père avec ses frères, puis, selon une stratégie en profondeur d'abord, afin de traiter les fils. Par ailleurs, la construction des meilleurs bigroupes se fait par le traitement des fils (bigroupes cohérents) en appelant la fonction d'évaluation *Average Spearman's Rho (ASR)* [Ayadi et al., 2009]. Il ne garde que les nœuds (*bigroupes*) de bonne qualité si la valeur de sa fonction *ASR* est supérieure ou égale à un certain seuil. Enfin, dans l’étape de post-traitement, *BiMine* applique une procédure de suppression afin d’éliminer tous les bigroupes qui ont moins de deux individus ou moins de deux attributs. Les bigroupes restants sont considérés comme étant une solution pour le problème de biregroupement. L’algorithme *BiMine* est caractérisé par la qualité et la signification biologique de bigroupes fournis.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Équation ‎2‑7 | | | | |
| Avec : | | | | | |
| est la moyenne des valeurs de tous les attributs pour un et est un seuil fixé. (pour un exemple avec , voir le Tableau ‎2‑6 et le Tableau ‎2‑7). [Ayadi et Elloumi, 2011] | | | | | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | Attribut1 | Attribut2 | Attribut3 | Attribut4 | Attribut5 | Attribut6 | | Individu1 | 10 | 20 | 5 | 15 | 40 | 18 | | Individu2 | 20 | 40 | 10 | 30 | 24 | 20 | | Individu3 | 23 | 12 | 8 | 15 | 29 | 15 | | Individu4 | 4 | 8 | 2 | 6 | 5 | 5 | | Individu5 | 15 | 25 | 8 | 12 | 29 | 50 | | | |  |  | |
| **Tableau ‎2‑6 BiMine : Matrice M avant prétraitement [Ayadi et Elloumi, 2011]** | | |  |  | |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | Attribut1 | Attribut2 | Attribut3 | Attribut4 | Attribut5 | Attribut6 | | Individu1 | 10 | 20 | 5 | 15 | 40 | - | | Individu2 | 20 | 40 | 10 | 30 | - | 20 | | Individu3 | - | 12 | 8 | 15 | 29 | 15 | | Individu4 | 4 | 8 | 2 | 6 | - | - | | Individu5 | 15 | - | 8 | 12 | 29 | 50 | | | |  |  | |
| **Tableau ‎2‑7 BiMine : Matrice M aprés prétraitement [Ayadi et Elloumi, 2011]** | | |  |  | |

* CE-Tree [Chen et Chang, 2009] : Pour extraire un ensemble de bigroupes, l’algorithme *CE-Tree* utilise une stratégie globale en profondeur d’abord. L’idée est de commencer par la génération du *Maximum Dimension Sets* (*MDS*) pour chaque paire d’attributs afin de calculer la différence entre deux attributs pour tous les individus. Puis, il applique un tri croissant des valeurs obtenues pour aboutir, finalement, à un regroupement des valeurs les plus proches entre eux.

### Les algorithmes stochastiques

Les algorithmes stochastiques de biregroupement adoptent des approches différentes dans leur processus de biregroupement. Parmi ces approches, nous notons : l’*approche de Voisinage (NS),* l’*approche évolutionnaire (EA)* et l’*approche Hybride (HA).*

* *L'approche de Voisinage*

L’*approche de voisinage* (***N****eighborhood* ***S****earch* ***(NS)***) applique une exploitation locale de tout l’espace de rechercher en suivant une fonction de voisinage. En effet, cette approche applique le même principe que l’*approche gloutonne (GIS*). Autrement dit, l’*approche de voisinage* permet de fixer un point de départ qui peut être un groupe, un bigroupe ou la totalité de la matrice. Par la suite, elle tente d’améliorer une solution obtenue en ajoutant/supprimant quelques individus/attributs pour maximiser/minimiser certaines fonctions. Cependant, pour éviter la présence de l’inconvénient majeur de l’approche *GIS*, elle ajoute la possibilité de récupérer des lignes/colonnes déjà supprimées. Ses avantages sont la qualité de la solution, la capacité d’exploitation et le temps d’exécution réduits par rapport à GIS. Toutefois, elle peut donner une solution non-optimale. Il existe plusieurs algorithmes adoptant cette approche. Parmi ces algorithmes, on note l’algorithme de *Bryan et al.* [Bryan et al., 2006]*,* GRASP[Dharan et Nair, 2009] *et CGRASP* [Das et Idicula, 2010].

* *L'approche évolutionnaire*

L’*approche évolutionnaire* (***E****volutionary* ***A****pproach* ***(EA)***) est basée sur des évolutions naturelles des espèces. Elle utilise un processus itératif de *sélection*[[2]](#footnote-2), *croisement*[[3]](#footnote-3) et *mutation*[[4]](#footnote-4) avec un algorithme évolutionnaire. Proprement dit, l’approche évolutionnaire applique une exploration de toutes les solutions (les individus) générées à partir d’une population (la matrice M). Par la suite, pour minimiser la population générée, elle effectue, pour chaque solution, une évaluation basée sur fonction d’évaluation spécifique. Ce processus s’arrête si un critère d’arrêt est vérifié. L’avantage de cette approche est qu’elle garantit la facilité d’exploration, la qualité de la solution et réduit le temps d’exécution. Cependant, son inconvénient est qu’elle peut aussi tomber sur une solution non-optimale. Il existe plusieurs algorithmes adoptants cette approche. Parmi ces derniers, on trouve *SEBI* [Divina et Aguilar-Ruiz, 2006]*, SMOB* [Divina et Aguilar-Ruiz, 2006] et *PCOBA* [Joung et al., 2012].

* SEBI (Sequential Evolutionary BIclustering) [Divina et Aguilar-Ruiz, 2006]: l’algorithme de *SEBI* utilise un processus itératif afin de découvrir un ensemble de bigroupes à partir d’une population ayant une taille maximale et une valeur de *fonction* *MSR* inférieur à un seuil déjà fixé. L’algorithme utilise les opérations de *mutation*, *sélection* et *croisement*. Le processus se termine si le nombre de bigroupes générés est atteint.
* *L'approche hybride*

L’approche hybride (***H****ybrid* ***A****ppraoch****(HA)***) permet de combiner les deux avantages des approches précédentes. Elle garanti la facilité d’exploration de l’approche évolutionnaire et la capacité d’exploitation de l’approche de voisinage. Elle est fréquemment utilisée avec les problèmes des recherches combinatoires. La combinaison des deux approches précédentes, aide à faciliter l’exploration comme elle garantie la capacité de l’exploitation. Ainsi, cette combinaison peut avoir un bon compromis entre l'exploitation et l'exploration qui sont deux critères pertinents pour la recherche. L’inconvénient majeur de cette approche est qu’elle peut aussi donner une solution non-optimale. Il existe plusieurs algorithmes adoptants l’*approche* *hybride*. Parmi ces derniers on peut citer celui de *Bleuler et al.* [Bleuler et al., 2004]*, celui de Mitra et Banka* [Mitra et Banka, 2006]*, BiHEA* [Gallo et al., 2009] *et MoHEA* [Seridi et al., 2011]*.*

* MoHEA(Multi-objectives Hybrid Evolutionary Algorithm) [Seridi et al., 2011]*:* Cet algorithme génère sa population initiale en utilisant l’algorithme *Cheng et Church* [Cheng et Church, 2000] combiné avec une procédure de recherche locale. Il utilise des opérations (aléatoires ou pré-spécifiées) de croisement en remplaçant les opérations de mutation par l’algorithme de *Cheng et Church* [Cheng et Church, 2000].
* BiHEA(Biclustering-Hybrid Evolutionary Algorithm) [Gallo et al., 2009]: C’est un algorithme hybride et multi-objectif. Il permet de maximiser la taille de bigroupes ainsi que l’homogénéité et la variance des lignes. D’abord, il applique, sur chaque individu, des opérations de mutation avec une faible probabilité. Ensuite, il applique des opérations (aléatoires ou pré-spécifiées) de croisement. Enfin, il utilise une procédure de recherche locale proposée par *Cheng et Church* [Cheng et Church, 2000].

### Récapitulatif des algorithmes

À ce stade, nous avons présenté les différentes approches utilisées pour résoudre le problème de biregroupement et quelques algorithmes reconnus. Le Tableau ‎2‑8 résume les algorithmes de biregroupement les plus connus. En effet, la première colonne présente les algorithmes. La deuxième colonne présente l’approche utilisée. En outre, la troisième colonne fournit la complexité en temps de calcul, ainsi la quatrième colonne décrit les termes utilisés.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Algorithme** | **Approche** | **Complexité** | **Notations** |
| **1972** | Block Clustering [Hartigan, 1972] | DAC | **-** | **n** : représente le nombre de gènes,  m : représente le nombre de conditions,  **l**: représente le nombre des meilleurs modèles d'ordre partiel,  **Cu** : représente le coût du calcul d’EMSR et d’ERV pour un bigroupe,  **pr** : représente une probabilité fournie par l'utilisateur pour permettre à l'algorithme de faire des mouvements aléatoires,  **d**: représente une borne maximale pour le nombre des sommets d'un graphe biparti G dont les deux parties correspondent à l'ensemble de gènes et des conditions  **r** : représente le poids maximum d'un arc dans un graphe biparti G,  **β**: représente le nombre de bigroupes qui ne sont pas inclus dans d'autres bigroupes,  **μ**: représente le nombre maximum de mouvement des gènes/conditions. |
| **2002** | OPSM [Ben-Dor et al., 2002] | GIS |  |
| SAMBA [Tanay et al., 2002] | BE |  |
| δ*-pClusters [Wang et al., 2002]* | BE | **-** |
| **2003** | Spectral [Klugar et al., 2003] | GIS | **-** |
| OP-Clusters [Liu, et Wang, 2003] | GIS | **-** |
| FLOC [Yang et al., 2003] | GIS | **-** |
| xMOTIFs [Murali et Kasif, 2003] | GIS |  |
| **2004** | Bleuler et al. [Bleuler et al., 2004] | HA | **-** |
| Co-Clusters [Cho et al., 2004] | BE | **-** |
| **2006** | SBEI [Divina et Aguilar-Ruiz, 2006] | EA |  |
| SMOB [Divina et Aguilar-Ruiz, 2006] | EA | **-** |
| BiMax [Prelic et al., 2006] | DAC |  |
| Mitra et Banka [Mitra et Banka, 2006] | HA | **-** |
| Bryan et al. [Bryan et al., 2006] | NS | **-** |
| **2007** | MSBE [Liu et Wang, 2007] | GIS |  |
| **2008** | Teng et Chan [Teng et Chan, 2008] | GIS | **-** |
| RWB [Angiulli et al., 2008] | GIS |  |
| δ-biclusters *[Cheng, et al., 2008]* | GIS |  |
| **2009** | BiHEA [Gallo et al., 2009] | HA | **-** |
| CE-Tree [Chen, et Chang, 2009] | BE |  |
| GRASP *[Dharan et Nair, 2009]* | NS |  |
| BiMine [Ayadi et al., 2009] | BE |  |
| **2010** | CGRASP [Das et Idicula, 2010] | NS | **-** |
| **2011** | MoHEA[Seridi et al., 2011] | HA | **-** |
| **2012** | BBK [Oliver et al., 2012] | BE | **-** |
| PCOBA [Joung, et al., 2012] | EA | **-** |
| **2013** | HOCCLUS2 [Pio et al., 2013] | BE | **-** |

Tableau ‎2‑8 Les algorithmes de biregroupement

## Conclusion

Dans ce chapitre, nous avons présenté un état de l’art sur le biregroupement. Notre étude nous a permis de souligner la capacité des algorithmes de biregroupement pour l’analyse des données biologiques, spécifiquement, dans la classification simultanée d’un ensemble d'individus sous un sous-ensemble  d'attributs. Cette technique de fouille de données est très intéressante pour la génération de bigroupes cohérents et pour la découverte locale des comportements similaires entre les individus [Madeira et al., 2004]. Par ailleurs, l'alignement de séquences multiples (MSA) est souvent généré par le regroupement d’un ensemble de sous-séquences fonctionnelles et conservées. En se basant sur cette idée, nous proposons un algorithme d’alignement multiple basé sur la technique du biregroupement.

# *BicMSA* : Algorithme d’Alignement Multiple basé sur le Biregroupement

## Introduction

L’objectif d’un *alignement multiple de séquences* (*MSA*) est d’identifier le maximum des régions similaires afin de dévoiler les homologies entre les séquences biologiques qu’y appartiennent. Comme nous avons déjà mentionné, ces régions jouent un rôle très important d’un point de vue fonctionnel et structurel puisqu’elles aident efficacement à étudier l’histoire de l’évolution des organismes vivants. Pour ce là, la qualité de résultats obtenus peut notamment influer sur la qualité des interprétations biologiques telles que l’analyse, l’annotation et la prédiction. Par ailleurs, un survol de la littérature nous a montré que dans plusieurs applications biologiques, l’alignement local est plus significatif que l’alignement global. Cette signification est argumentée par la capacité de découvrir des *homologies* dans des bases de données sensibles qui disposent un ensemble de régions hautement ou faiblement conservées. En effet, l'alignement local est souvent destiné à regrouper un ensemble de sous-séquences fonctionnelles et conservées. Cependant, le choix de la technique de regroupement peut facilement influer la qualité de l'alignement.

Dans ce cadre, nous avons eu recourt à utiliser le biregroupement [Cheng et Church, 2000; Madeira et Oliveira, 2004] pour générer des alignements locaux indépendamment de types de séquences biologiques. Ainsi, le biregroupement permet la découverte non-triviale des sous-séquences similaires possédant des fonctions biologiques proches [Wang et al., 2002; Sharan et al., 2002; Quackenbush, 2006; Prelic et al., 200.; Divina et Aguilar-Ruiz, 2007; Pontes et al., 2007]. En effet, l’utilisation du biregroupement dans la résolution des problèmes MSA est une piste qui n'a pas été bien explorée. Récemment, *Wang et al.* [Wang et al., 2007] ont proposé l'algorithme *BlockMSA* qui est  une première adaptation du biregroupement en utilisant l’algorithme *BiMAX* [Barkow et al., 2006] pour la résolution des problèmes d’alignement multiples.

Nous proposons un algorithme d’alignement multiple basé sur le biregroupement, appelé *BicMSA (****Bic****lustering-based local* ***MSA***). Notre algorithme vise à appliquer la technique de biregroupement dans la résolution d’un problème de MSA local. Il exerce les atouts de cette technique de fouille de données dans l’identification maximale des similarités locales entre les séquences biologiques. En effet, l’idée consiste à s’orienter d’une simple recherche des similarités entre les séquences à une recherche des relations entre un ensemble de régions et toutes les séquences à aligner. *BicMSA* utilise l’approche *énumération de bigroupes* pour le *biregroupement* et l’approche *DAC* (*Diviser et conquérir*). ). Il permet de résoudre le problème de *MSA local* entre un ensemble de séquences biologiques avec la résolution des sous-problèmes des alignements globaux entre un ensemble de sous-séquences. Le résultat généré est à la fois des groupes de sous-séquences similaires et un alignement local entre toutes les séquences. En utilisant la technique locale, *BicMSA* fourni des alignements biologiquement significatifs en dévoilant les similarités structurelles et fonctionnelles.

Dans ce chapitre, nous présentons, dans la première partie, les différentes définitions et notations en relation avec notre algorithme. Dans la deuxième partie, nous décrivons notre algorithme suivi d’un exemple illustratif. Dans la troisième partie, nous discutons des points forts de notre algorithme par rapport à l’algorithme *BlockMSA*. Nous terminons ce chapitre par une conclusion.

## L’algorithme *BicMSA*

### Définitions et notions de base

On présente tout d’abord les définitions et les notations relatives à notre algorithme :

1. (fragment) : *C’est une sous-séquence de la séquence (avec ( est le nombre de séquences à aligner). Ses caractères commencent par la position et se termine à la position (avec ). On le note par .*
2. (*paramètre k) : est un paramètre fixé par l’utilisateur, il doit être supérieur à . Il contrôle la taille minimale des fragments qui constituent un*
3. *(longueur-) : La longueur d’un bloc est notée . Elle doit être supérieure ou égale au paramètre k et inférieure à la taille de la séquence (avec ).*
4. *() : C’est un ensemble de régions conservées. C’est le bloc (avec ( est le nombre de blocs générés) appartenant à une séquence . Il est composé par un ou plusieurs fragments de longueur supérieure ou égale à k. Il est noté par*
5. *() : C’est un ensemble de identiques existants dans séquences (avec . Un se compose seulement par des de type un-gapped-blocs. La longueur de est égale soit à k, quand il se compose par des de longueur k (noté , soit à , s’il se compose par des combinaisons de de type bloc-adjacent disposant une longueur (noté .*
6. *(blocs-all-all) : Un est dit blocs-all-all si seulement si tous ses sont identiques et appartiennent à toutes les séquences ( avec ).*
7. *(blocs-1-all) : Un est dit blocs-1-all si seulement si tous ses blocs appartiennent à une seule séquence ( ).*
8. *(bloc-adjacent) : Le bloc-adjacent est la fusion des fragments de longueur qui appartiennent aux mêmes séquences et qui disposent des indices de début adjacents dans toutes ces séquences.*
9. *(un-gapped-blocs) : Un est dit un-gapped si seulement si tous ses blocs necontient pas une brèche « ». Proprement dit, étant donné le caractère appartenant au (avec , il doit satisfaire l’équation suivante :*

|  |  |
| --- | --- |
|  | Équation ‎3‑1 |
|  |  |

1. *(Position bloc) : C’est la position du premier caractère d’un (avec ) dans une séquence non-alignée (avec ). Elle est notée par .*
2. *(Position-maximale) : C’est la position maximale de dans les séquences (voir* *Équation ‎3‑2). On la note par .*

|  |  |  |
| --- | --- | --- |
|  | Équation ‎3‑3 | |
| *Avec :*  est la position de bloc dans la séquences (avec . | | |

1. *(Nombre décalage-) :* *C’est le nombre de décalage entre la position maximale et la position courante (voir Équation ‎3‑4). On le note par .*

|  |  |
| --- | --- |
|  | Équation ‎3‑5 |

*Avec :*  est la position de bloc dans la séquences (avec .

1. *(Position brèche) : C’est la première position d’une brèche dans un fragment d’une séquence (avec . Elle est notée par*
2. (*Position bloc aligné*) *: C’est la position d’un bloc aligné dans une séquence (avec et non pas la séquence originale (avec . Elle est notée par*
3. *(Positions Sup)* : *Ce sont toutes les positions de la matrice M ou bien de la matrice M supérieures à la position*  .
4. *(Positions-Sup-brèche)* : *Ce sont toutes les positions de la matrice M ou bien de la matrice M supérieures à la position   et inférieur à la position  .*
5. *(blocs-non-biregroupés) : Ce sont les blocs qui peuvent être écartés par l’algorithme de biregroupement et qui ne sont pas marqué comme alignés dans l’étape d’alignement par biregroupement.*
6. *(intervalle-brèche) : l’intervalle de brèche est un ensemble de caractèresrécupérés d’une séquence(avec et caractérisés par une valeur de début de brèches notée et une valeur de fin de brèches notée. Ces caractères doivent être seulement des brèches :*

|  |  |
| --- | --- |
|  | -Équation ‎3‑6 |

*Avec :* est la position de caractère dans la )

1. *(meilleure-position) : Une position dans l’intervalle-brèche est considérée comme la meilleure position d’un caractère préfixe (avec ) ou bien d’un caractère suffixe (avec ), si seulement si, un ou plusieurs caractères en préfixes (resp. suffixe) possèdent un score CS inférieur à leur score CS en les décalant à cette position. Elle est notée avec () dans l’intervalle-brèche.*
2. *(alignements-cachés) : Ce sont les alignements qui peuvent être dévoilés avec un décalage d’un ou plusieurs caractère certaine à une meilleure position .*

### Description de l’algorithme *BicMSA*

En s’inspirant de l’algorithme *BlockMSA* [Wang et al., 2007], *BicMSA* *(****Bic****lustering-based local* ***MSA***) utilise un compromis entre l’approche *DAC* « *diviser et conquérir* » et l’approche « *énumération de bigroupes* » dans la résolution du problème de MSA. L’objectif de *BicMSA* est de produire des alignements proches de ce que peut faire un biologiste en cherchant le maximum des blocs cohérents disposant des relations avec le maximum de séquences. De ce fait, l’idée consiste à générer, d’abord, un ensemble des ( est le nombre de blocs générés) disposant une cohérence maximale. Cette cohérence est établie par la division des alignements optimaux entre chaque deux séquences. Puis, *BicMSA* utilise la technique de biregroupement, spécifiquement, l’approche *énumération de bigroupes* afin de maximiser la cohérence totale entre toutes les séquences. Enfin, le résultat est raffiné pour obtenir un alignement multiple et local (*voir Algorithme.3.1*).

|  |
| --- |
| BicMSA (,) |
| Données = Groupe de séquences non alignées (),  Résultats = Groupe de séquences alignées().  FN\_Génération\_blocs(IN , IN , OUT M)  FN\_Biregroupement(IN M, OUT )  FN\_Assemblage\_Raffinement(IN , IN , OUT ).)  -Fin |

Le processus d’alignement se subdivise en trois étapes principales (voir *Figure ‎3‑1*):

Prétraitement : Génération de blocs

Post-traitement : Assemblage

Traitement : Biregroupement

Raffinement

Figure ‎3‑1 Processus de l’algorithme *BicMSA*

*Etape  : Prétraitement (Génération de blocs) :* Cette étape vise à représenter le problème de MSA comme un problème de biregroupement et à produire le maximum des disposant des cohérences maximales. De ce fait, nous générons tous les blocs de longueurs qui appartiennent à au moins séquences tel que. En se basant sur la cohérence maximale des blocs, nous construisons une matrice M contenant l’ensemble de séquences et leurs blocs associés.

*Etape  : Traitement (Biregroupement) :* L’objectif de cette étape est de résoudre un problème de biregroupement indépendamment du problème de MSA. En utilisant la matrice M, elle permet de générer un ensemble de bigroupes cohérents et stables afin maximiser cohérence totale entre toutes les séquences . Dans ce cadre,nous adaptons l’approche *énumération de bigroupes,* spécifiquement, l’arbre *BET* de l’algorithme *BiMine* [Ayadi et al., 2009] afin de regrouper d’une manière optimale les avec les . En effet, les bigroupes générés présentent, d’une part, les relations entre les blocs et les séquences, d’autre part, les relations entre toutes les séquences.

*Etape  : Post-traitement (Assemblage et Raffinement) :* Cette étape vise à assembler les blocs générés dans l’étape de *Prétraitement* en suivant les cohérences données par les bigroupes. Elle permet de produire un alignement local entre les séquences. Par ailleurs, un processus de raffinement est appliqué afin d’améliorer la qualité de l’alignement obtenu.

1. Prétraitement (Génération de blocs)

Cette étape vise à trouver une correspondance entre un ensemble de séquences et leurs *blocs* associés *(voir Algorithme.3.2)*. Elle prend comme entrée un ensemble de séquences et un *paramètre*  qui définit la taille minimale des blocs générés . Lorsque cette étape se termine, *BicMSA* fournit une matrice M dont chaque ligne correspond à un et chaque colonne correspond à une . Si le existe dans la , la case de la matrice M contient alors la *position bloc* de dans la séquence sinon la case vaut (voir Tableau ‎3‑1 ).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | …. |  | …. |  |  |
|  |  | 11 | 12 | …. |  | …. |  |  |
|  |  | 21 | 22 | …. |  | …. |  |  |
|  | …. | …. | …. | …. | …. | …. | …. |  |
|  |  |  |  | …. |  | …. |  |  |
|  | …. | …. | … | …. | ….. | …. | ….. |  |
|  |  | Posm1 |  | …. |  | …. |  |  |
|  | Tableau ‎3‑1 La Forme de la Matrice M. | | | | | | |  |

Cependant, la génération des blocs se fait en deux étapes *(voir Algorithme.3.2)*: la *génération d’une bibliothèque des meilleurs alignements globaux* et l’*identification de blocs.*

|  |
| --- |
| FN\_Génération\_blocs() |
| Données = Groupe de séquences non alignées (),  Résultats = M  FN\_Génération\_Biblio\_Meilleurs\_Alignements\_Globaux()  FN\_Identification\_blocs ()  -Fin |

* *Génération d’une bibliothèque des meilleurs alignements globaux :* Pour identifier les disposant les cohérences maximales, nous commençons par la création d’une bibliothèque des meilleurs alignements globaux entre chaque deux séquences en utilisant un programme externe. Comme nous cherchons toujours à utiliser des solutions optimales afin de garantir la qualité des résultats générés, nous utilisons l’algorithme exact *Needleman et Wunch* [Needleman et Wunsch, 1970]. Le choix de la technique globale est argumenté par sa capacité de fournir le maximum des *fragments* en concordance entre deux séquences et sa qualité de l’alignement par paire par rapport à un alignement local. Par ailleurs, dans chaque génération des alignements entre une séquence et l’ensemble de séquences, nous gardons que les paires (, ) (avec 2 et ) qui disposent le score SP maximal. Le résultat est un ensemble de paires alignées d’une manière optimale et globale.
* *Identification de blocs :* Après la génération de la bibliothèque des meilleurs alignements par paires, cette étape vise, d’une part, à maximiser la cohérence entre l’ensemble de (avec 1) générés, d’autre part à produire la matrice de biregroupement M.

|  |
| --- |
| FN\_identification\_blocs() |
| Données= ,  Résultats = M, M  Tantque (il existe\_une\_paire\_dans\_ (, )) faire  | RécupérerInfo(Nom,Chaine,Nom,Chaine,)  | Pour de 1 à Faire  | | Glisser\_Fênetre\_à\_position\_\_de\_longueur\_(, )  | | Si (FN\_un-gapped(, ))alors  | | | , ←FN\_ChercherPosBlocdansM(, )  | | | Si () alors  | | | | FN\_Proccessus\_Ajout/MiseàJour\_2blocs(, ,,)  | | | Sinon :  | | | | FN\_ Proccessus\_Ajout/MiseàJour\_1blocs(, ,,)  | | | | FN\_Proccessus\_Ajout/MiseàJour\_1blocs(, ,,)  | | | FinSi  | | FinSi  | FinPour  FinTanque  Pour de 1 à faire  | FN\_Eliminer\_blocs-1-all(M)  | FN\_Combiner\_blocs-adjacents(M)  | FN\_Deplacer\_ blocs-all-all (M, M)  FinPour  Fin |

L’idée consiste à glisser une fenêtre de longueur sur l’alignement de chaque paire(, ) (avec 2 et ) afin de générer les *fragments et* les *fragments* . En effet, chaque *fragment* produit un ensemble de caractères qui composent un (resp.) de longueur et chaque deux blocs ( et ) produisent l’ensemble de . Par la suite, nous gardons que les de type *un-gapped-blocs* afin de filtrer les sous-alignements non pénalisé par des brèches. En outre, pour garantir une cohérence maximale entre les deux blocs, nous traitons les suivant leurs similarités. En effet, s’il s’agit d’une similarité totale (les deux blocs sont identiques), nous affectons simultanément le et le (voir *Algorithme.3.4*). Sinon, nous affectons le indépendamment du (voir *Algorithme.3.5*).

Par ailleurs, les affectations des *positions blocs* de à la matrice M se font en suivant un processusd’ajout des ou bien de mise à jour des positions. Dans chaque processus, nous examinons le type et l’existence du chaque bloc dans la matrice M et nous contrôlons l’insertion des brèches avec une fonction appelée *Affectation\_bloc\_Avec\_ Contrôle\_brèche()*. Ces contrôles sont appliqués afin de maximiser la cohérence entre un et l’ensemble de séquences. En effet, la fonction *Affectation\_bloc\_Avec\_ Contrôle\_brèche()* permet de contrôler l’insertion des brèches dans l’étape de *Post-traitement*, également, de contrôler la taille des alignements résultants. Elle autorise une mise à jour ou bien un ajout d’un bloc en calculant le nombre de décalage entre une position courante et une autre position maximale existante dans une ligne de la matrice M. Proprement dit, si un bloc n’existe pas dans la matrice M, nous ajoutons une nouvelle ligne comportant le bloc et sa *position bloc* (). Inversement, ce processus est appliqué lorsque le bloc trouvé dans la ligne est de type *bloc-all-all* ou bien la fonction *Affectation\_bloc\_Avec\_Contrôle\_brèche()* n’autorise pas la mise à jour de la ligne . Ainsi, si le bloc trouvé à la position est de type *bloc-1-all* ou bien la fonction *Affectation\_bloc\_Avec\_Contrôle\_brèche()* autorise la modification de la ligne , nous appliquons une mise à jour à la *position bloc* dans la ligne .

|  |
| --- |
| FN\_ Processus\_Ajout/MiseàJour\_2blocs() |
| Données= , ,,  Résultats = M  Si (== - ) ou (FN\_Bloc-all-all(,) == vrai) alors  | AjouterLigneBloc(,,)  Sinon  | Si (FN\_Bloc-1-all(,) == vrai ) alors  | | MettreaJourPosition(,, )  | Sinon Si (M[][] == 0 OU M[][] == 0) alors  | | FN\_ A*ffectation\_bloc\_Avec\_ Contrôle\_brèche*( , , )  | | FN\_ A*ffectation\_bloc\_Avec\_ Contrôle\_brèche* ( , , )  | FinSi  FinSi |

|  |
| --- |
| Processus\_Ajout/MiseàJour\_1blocs() |
| Données= , ,,  Résultats = M  Si (== - ) ou (FN\_Bloc-all-all(,) == vrai) alors  | AjouterLigneBloc(,)  Sinon  | Si (FN\_Bloc-1-all(,) == vrai ) alors  | | MettreaJourPosition(, )  | Sinon Si (M[][] == 0) alors  | | FN\_ A*ffectation\_bloc\_Avec\_ Contrôle\_brèche* ( , , )  | FinSi  FinSi |

À la fin de l’étape d’identification, un processus de nettoyage est appliqué afin d’éliminer les blocs de type *blocs-1-all*, combiner les blocs de type *bloc-adjacent* et déplacer les blocs de type *blocs-all-all* vers une nouvelle matrice appelée M. En effet, cette matrice possède tous les blocs présents dans toutes les séquences. Les résultats de cette étape sont la matrice M et la matrice M

### Traitement (Biregroupement)

La technique de biregroupement vise à produire un ensemble de bigroupes cohérents, stables et homogènes [Cheng et Church, 2000; Madeira et Oliveira, 2004]. Son utilisation dans la résolution d’un problème d’alignement multiple est argumentée par la capacité de trouver un sous ensemble d’individus (séquences) ayant un même comportement par rapport à un ensemble d’attributs (blocs). En utilisant la matrice M générée dans l’étape précédente, nous appliquons cette technique de fouille de données afin d’extraire des régions similaires et de maximiser des cohérences entre toutes les séquences.

Par ailleurs, la qualité de biregroupement a une grande influence sur la qualité d’alignement. Un survol de la littérature nous a montré que l’approche *énumération de bigroupe*permet detrouver, d’une manière optimale, l’ensemble de meilleurs bigroupes souhaités en suivant une énumération de toutes les solutions possibles pour un problème donné. Il existe plusieurs algorithmes adoptants cette approche. Parmi ces algortithmes, nous avons utilisé une version adaptée de l’algorithme *BiMine* [Ayadi et al., 2009]appelée *BiMine-modifié* (voir *Algorithme.3.6*)*.*

*BiMine* permet de générer tous les meilleurs bigroupes existants en utilisant une nouvelle structure d'arbre, appelée *Bicluster* *Enumeration* *Tree* (*BET*). Son processus de biregroupement se subdivise en trois étapes principales : *Prétraitement*, *Traitement* (*Construction de BET* et *Extraction de bigroupes) et Post-traitement*. En appliquant *BiMine-modifié* sur la matrice de blocs M, nous avons supprimé l’étape de *Prétraitement* à cause de ces opérations de filtrage quinégligent l’existence des blocs dans quelques séquences et en raison de la différence entre les types de données de la matrice M (*BiMine-modifié :* blocs/séquences et *BiMine* : gènes/conditions). Nous avons, ainsi, éliminé l’étape de *Post-traitement* puisqu’elle supprime tous les bigroupes qui ont moins de deux séquences ou moins de deux blocs. En effet, cette étape est ignorée puisque les bigroupes disposant moins de deux séquences sont déjà éliminés en éliminant les de la matrice M (), ainsi, puisque les bigroupes disposant un seul bloc présentent des similarités entre les séquences associées. En outre, nous avons ignoré le contrôle de la fonction *ASR* [Ayadi et al., 2009]dans la construction de l’arbre *BET* à cause de la différence des données biologiques.

En conséquence, *BiMine-modifié* permet de trouver tous les bigroupes quelques soit leur qualité. Il construire l’arbre *BET* suivant deux stratégies : *la stratégie en largeur d'abord* afin de produire les nœuds fils en combinant un nœud père avec ses frères, ainsi, la *stratégie en profondeur d'abord* afin de traiter les fils. Par ailleurs, *BET* possède comme racine un nœud vide. Chaque nœud du premier niveau est un bigroupe contenant une seule séquence et l'ensemble de assosiés. Le nœud fils du nœud père est construit par l'union des séquences du père et ceux du oncle, en partant de la droite du père et de l'intersection des blocs du père et ceux du oncle. Alors, le nœud fils trouvé est considéré comme un nœud père et le processus de construction du sous-arbre ayant comme racine ce nœud continue. La construction de *BET* se termine lorsqu'il n'y a plus de nœuds à traiter. Le résultat généré est un ensemble de bigroupes cohérents .

|  |
| --- |
| FN\_Biregroupement() |
| Données = M  Résultats =  Etapes 1 :  ←*FN\_ BiMine-modifié* (M) ;  Fin |

### Post-traitement (Assemblage et Raffinement)

L’utilisation de biregroupement nous a permet de maximiser la cohérence totale entre les séquences en se basant sur l’ensemble de . Il ne reste que de trouver l’alignement final en utilisant l’ensemble de bigroupes générés dans l’étape précédente. Le processus d’alignement se fait en trois étapes : *Assemblage\_par\_biregroupement*, *Assemblage\_bocs-all-all* et *Raffinement* *(voir Algorithme.3.7)*:

|  |
| --- |
| FN\_Assemblage\_Raffinement() |
| Données = , , M, M  Résultats =  ←FN\_Alignement\_Par\_Biregroupement( , , M);  Alignement\_bloc-all-all(, M;  ←FN\_Raffinement (, );  Fin |

* *Assemblage par biregroupement*

En suivant la cohérence donnée par les bigroupes générés dans l’étape précédente, cette étape vise à assembler tous les blocs sur toutes les séquences afin de produire le premier alignement local *(voir Algorithme.3.8).* L’idée consiste à placer chaque bloc dans les séquences présentées par un bigroupe en appliquant un processus d’ajout des brèches ou bien de décalage autour de la séquence possédant le bloc avec la *position maximale*.

|  |
| --- |
| FN\_Assemblage\_Par\_Biregroupement() |
| Données = , , M  Résultats =  Pour chaque bigroupe Faire  | Pour chaque dans chaque bigroupe Faire  | | Pour chaque dans chaque bigroupe Faire  | | | ← FN\_Chercher\_Max\_PosDansLigne\_de\_);  | | | Si ( ) et () alors  | | | |  | | | |  | | | | FN\_AssemblerUnBloc(M);  | | | | ← vrai  | | | FinSi  | | FinPour  | | Pour chaque séquence contenant des en relation avec l’alignement Faire  | | | Si (( ) alors  | | | | FN\_AssemblerUnBloc(M);  | | | FinSi  | | FinPour  | FinPour  FinPour  Fin |

Proprement dit, nous cherchons pour chaque bloc la *position maximale*  dans les séquences et nous calculons le *nombre de* *décalage*  entre toutes les positions et la position *maximale* trouvée. Lorsque, le nombre de *décalage* est égal à , nous gardons le à la position. Sinon, nous cherchons la position de la brèche dans l’intervalle d’une séquence *(voir Algorithme.3.8* et *Algorithme.3.9)* :

Si la séquence ne contient pas des brèches ( : nous ajoutons des brèches progressivement en décrémentant la valeur jusqu’à ce qu’elle soit égale à . De ce fait, le est aligné à la bonne position de . Dans ce cas, nous appliquons une mise à jour sur toutes les *Positions-Sup* en ajoutant la valeur de . En effet, est une valeur temporelle permettant de stocker le nombre de *décalage* initial de.

Sinon (: nous décalons les caractères dans l’intervalle jusqu’à : soit nous arrivons à la position , ou bien aucune brèche n’est pas trouvée (. Dans chaque décalage, nous incrémentons la valeur , nous décrémentons la valeur et nous cherchons une nouvelle . Si la séquence ne dispose plus de brèches et le n’est pas à la bonne position (), nous retournons à l’étape avec les nouvelles valeurs de et . Lorsque le est aligné à la bonne position de , nous appliquons une mise à jour à toutes les *Positions-Sup-brèche* en les ajoutant la valeur de .

|  |
| --- |
| FN\_AssemblerUnBloc () |
| Données = M  Résultats =  Initialisation : ← -1  - Si () alors  | ←FN\_Rechercher\_Pos\_brèche();  | Tantque ((et () ) faire  | | FN\_décaler\_caracteres\_a\_pos\_brèche(,);  | | FN\_Mettre\_a\_jour\_toutes\_les\_positions\_entre() ;  | |  | |  | | ←FN\_Rechercher\_Pos\_brèche()  | FinTanque  | Si (() and ()) alors  | | Insérer\_brèche\_et\_aligner\_bloc(,)  | | FN\_Mettre\_a\_jour\_toutes\_les\_positions\_sup();  | FinSi  FinSi  Fin |

Pour éviter de perdre l’emplacement des blocs déjà alignés , nous cherchons à réaligner tous les dans les séquences qui disposent une relation avec le situé dans la et dont leur *position bloc aligné* est supérieure à la position . Enfin, nous marquons le comme un afin de dégager facilement les *blocs-non-biregroupés* dans l’étape de « *Raffinement »*.

* *Assemblage des blocs de type bloc-all-all*

Rappelant, que dans l’étape de *génération de blocs,* nous avons déjà filtré les de types *blocs-all-all*. Cette étape vise à les assembler dans les bonnes positions en suivant le même principe cité dans l*’Algorithme.3.8 et l’Algorithme.3.9.* Cependant, ces blocs présentent un alignement local sur toutes les séquences.

* *Raffinement*

L’étape de *Raffinement* vise à améliorer la qualité de l’alignement généré à présent. Etant donné la matrice M et l’alignement réalisé, nous assemblons les *blocs non-biregroupés*, également, les blocs non-alignés et nous cherchons l’ensemble d*’alignements-cachés* (voir *Algorithme.3.10*)*.*

|  |
| --- |
| FN\_Raffinement() |
| Données =  Résultats =  FN\_Nettoyage&AssemblageBlocsNonbiregroupés (  Pour de 1 à n Faire  | //Nettoyage en préfixe  | ;  | FN\_Chercher\_Intervalle\_brèche\_apartir\_de\_deb()  | Tantque ( existe\_de\_brèche && ) faire  | | Si(existe\_de\_brèche) alors  | | | FN\_NettoyagePrefixSeqDecalageEnAvant(,)  | | FinSi  | | FN\_Chercher\_Intervalle\_brèche\_apartir\_de\_deb()  | FinTantque  | //Nettoyage en suffixe  | deb;  | FN\_Chercher\_Intervalle\_brèche\_apartir\_de\_deb()  | Tantque ( existe\_de\_brèche && ) faire  | | Si(existe\_de\_brèche) alors  | | | FN\_NettoyageSuffixeeqDecalageEnArriére(,)  | | FinSi  | | FN\_Chercher\_Intervalle\_brèche\_apartir\_de\_deb()  | FinTantque  FinPour  Fin |

Par ailleurs, les *alignements-cachés* sont dévoilés en utilisant un *nettoyage en suffixe* ou bien un *nettoyage en préfixe*. Étant donné un *intervalle-brèche* dans une séquence, nous appliquons le processus de nettoyage, soit en parcourant l’intervalle de à la pour avoir un *nettoyage en préfixe,* soit en parcourant l’intervalle de au pour avoir un *nettoyage en suffixe*. Le processus de la recherche d’*alignement-caché* s’arrête si tous les *intervalles-brèches* sont traités dans toutes les séquences .

L’idée consiste à rechercher, itérativement, un ensemble d’*intervalles-brèches* dans chaque séquenceafin de décaler un caractère en suffixe (resp. préfixe) disposant un score CS inférieur à un score CS trouvé dans cet intervalle. Egalement, ce processus de recherche et décalage peut être appliqué sur un ensemble de caractères en préfixe (resp. suffixe) qui appartiennent à la même colonne et qui sont identiques à un caractère situé dans la position (resp. ).

Prenant par exemple le *nettoyage en préfixe* (voir *Algorithme.3.11*), nous commençons par la recherche des *intervalles-brèches* caractérisés par une valeurs de et de . Pour chaque *intervalle-brèche* dans chaque , nous appliquons un décalage à la meilleure position , soit d’un caractère ( ) de la colonne , ou bien un ensemble de caractères similaire à dans cette colonne. La position est considérée si, seulement si, un caractère (resp. l’ensemble de caractères) dispose un score CS inférieur à un score CS trouvé dans un *intervalle-brèche.* Le processus de nettoyaged’un *intervalle-brèche* s’arrête si, nous ne trouvons pas des ou bien nous nous trouvons à la .

|  |
| --- |
| FN\_NettoyagePrefixSeqDecalageEnAvant() |
| Données =,  Résultats =  Tantque () Faire  |FN\_Chercher\_\_d\_un\_ou\_plusieurs\_caractéres  (  | Si (\_de\_plusieurs\_caractères ) alors  | | Tantque ( && ) alors  | | | Décaler\_Tous\_Caractères\_de\_la\_colonne\_courante\_à\_  | | | Si ( ) alors  | | | |  | | | |  | | | | FN\_Chercher\_\_d\_un\_ou\_plusieurs\_caractéres  (  | | | Sinon  | | | | Arrêt  | | | FinSi  | | FinTantque  | Sinon Si \_d’un\_caractère\_ alors  | | Tantque ( && ) alors  | | | Décaler\_Caractère\_à\_  | | | Si ( ) alors  | | | |  | | | |  | | | | FN\_Chercher\_\_d\_un\_ou\_plusieurs\_caractéres  (  | | | Sinon  | | | | Arrêt  | | | FinSi  | | FinTantque  | Sinon  FinTantque  Fin |

La Figure 3-2 représente un récapitulative de toutes les fonctions utilisées par *BicMSA* :

**Génération\_bloc()**

Génération\_Biblio()

Identification\_bloc()

**In :**

**Biregroupement()**

BiMine\_Modifié()

**Out :** M

**Out :**

**Out :** Mblocallall

**ALGORITHME : BicMSA**

**Assemblage\_Raffinement()**

Assemblage\_Biregroupement()

Raffinement()

Assemblage\_bloc-all-all

Nettoyage\_bloc\_nonBiregroupé

Nettoyage\_en\_suffixe

Nettoyage\_en\_préfixe

Figure ‎3‑2 ALGORITHME *BicMSA*

### Exemple illustratif

Après la présentation de l’algorithme et l’explication de son principe de fonctionnement, nous décrivons, dans cette section, le processus de *BicMSA* en utilisant un exemple illustratif. De ce fait, nous fixons le *paramètre*  qui contrôle la taille du bloc et l’ensemble de séquences à aligner. En effet, nous avons choisi une collection des séquences d’ADN [Basseur, 2011] déjà alignées avec *CLUSTALW* [Thompson et al., 1994] pour comparer ses résultats avec *BicMSA*. Notant que *CLUSTALW* est l’un des algorithmes les plus célèbres dans la littérature. Il est souvent utilisé dans la résolution des alignements multiples.

Soit le paramètre et les séquences d’*ADN* suivantes tel que  :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Groupe ADN :** |  | **:** | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G |
|  | **:** | C | G | A | G | C | C | A | T | T | G | T | A | G | C | T | A | C | T | G |
|  | **:** | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G |
|  | **:** | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G |

Figure ‎3‑3 Groupes des séquences d’ADNs

1. *Prétraitement (Génération de blocs)*

* *Génération d’une bibliothèque des meilleurs alignements globaux*

- Générer tous les alignements entre chaque deux séquences avec l’algorithme de *Needleman* et *Wunch* [Needleman et Wunsch, 1970] (voir Figure ‎3‑4) en calculant le score SP de chaque alignement (avec ).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | T | G | A | | G | | T | | C | | A | | T | | T | | G | | T | | - | | G | | - | | - | | A | | C | | T | | | | G | | | SP  ↓ | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | - | G | - | | - | | C | | C | | A | | T | | T | | G | | T | | A | | G | | C | | T | | A | | C | | T | | | | G | | |
|  | **1** | **1** | **1** | **-2** | **1** | **-2** | | **-2** | | **-1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **-2** | | **1** | | **-2** | | **-2** | | **1** | | **1** | | **1** | | | | **1** | | | **= 2** | |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | T | G | | A | | G | | T | | C | | A | | T | | T | | G | | - | | T | | G | | A | | C | | T | | G | | SP  ↓ | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | C | C | | A | | - | | T | | T | | G | | T | | A | | G | | C | | T | | A | | C | | C | | T | | G | |
|  | **1** | **1** | **1** | **-1** | **-1** | | **1** | | **-2** | | **1** | | **-1** | | **-1** | | **1** | | **-1** | | **1** | | **-2** | | **1** | | **-1** | | **-1** | | **1** | | **1** | | **1** | | **= 0** | | | |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| : | C | G | A | T | G | | A | | G | | T | | C | | A | | T | | T | | G | | T | | G | | A | | C | | T | | G | | SP  ↓ | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| : | C | G | A | T | G | | A | | G | | T | | C | | A | | C | | T | | G | | T | | G | | A | | C | | T | | G | |
|  | **1** | **1** | **1** | **-1** | **-1** | | **1** | | **-2** | | **1** | | **-1** | | **-1** | | **1** | | **-1** | | **1** | | **-2** | | **1** | | **-1** | | **-1** | | **1** | | **1** | | **= 17** | | | | |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | G | C | C | | A | | T | | T | | G | | T | | A | | G | | C | | T | | A | | - | | C | | T | | G | | SP  ↓ | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | - | C | C | | A | | T | | T | | G | | T | | A | | G | | C | | T | | A | | C | | C | | T | | G | |
|  | **1** | **1** | **1** | **-2** | **1** | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **-2** | | **1** | | **1** | | **1** | | **= 14** | | | | |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | - | G | - | | - | | C | | C | | A | | T | | T | | G | | T | | A | | G | | C | | T | | A | | C | | T | | | | G | | | SP  ↓ | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | T | G | A | | G | | T | | C | | A | | C | | T | | G | | T | | - | | G | | - | | - | | A | | C | | T | | | | G | | |
|  | **1** | **1** | **1** | **-2** | **1** | **-2** | | **-2** | | **-1** | | **1** | | **1** | | **-1** | | **1** | | **1** | | **1** | | **-2** | | **1** | | **-2** | | **-2** | | **1** | | **1** | | **1** | | | | **1** | | | **= -1** | |
| * Alignement de séquences / | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | C | C | A | | T | | T | | G | | T | | A | | G | | C | | T | | A | | C | | C | | T | | G | | SP  ↓ | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C | G | A | T | G | A | | G | | T | | C | | A | | C | | T | | G | | T | | G | | A | | C | | T | | G | |
|  | **1** | **1** | **1** | **-2** | **1** | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **1** | | **-2** | | **1** | | **1** | | **= -1** | | | | |

Figure ‎3‑4 Génération des meilleurs alignements globaux

-Récupérer les couples de séquences ayant le score SP maximal (voir Figure ‎3‑5).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Les Scores SP **:** | | | | | | | |  |  |  |  |  | |  | |  | 2 | 0 | **17** |  | Paire (, ) | | | |  |  | **14** | -1 |  | Paire (, ) | | | | |  |  |  | **-1** |  | Paire (, ) | | | |
| |  |  | | --- | --- | | **Paire** | **Alignement global** | | Paire  (, ) | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | | | Paire  (, ) | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | G | C | C | A | T | T | G | T | A | G | C | T | A | - | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | - | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | | | Paire  (, ) | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | | |

**Figure ‎3‑5 Les meilleurs alignements**

* *Identification de blocs*

Dans cette étape, nous appliquons le processus d’*identification de blocs* sur chaque paire(, ) (avec 2 et ) de la bibliothèque donnée précédemment en initialisant le nombre de (voir Figure ‎3‑6).

|  |  |
| --- | --- |
| **Paire (, )** |  |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  | | CGA | 1 | 1 | 1 | 1 | | GAT | 2 | 0 | 0 | 2 | | ATG | 3 | 0 | 0 | 3 | | TGA | 4 | 0 | 0 | 4 | | GAG | 5 | 0 | 0 | 5 | | AGT | 6 | 0 | 0 | 6 | | GTC | 7 | 0 | 0 | 7 | | TCA | 8 | 0 | 0 | 8 | | CAT | 9 | 6 | 5 | 0 | | CAC | 0 | 0 | 0 | 9 | | ATT | 10 | 7 | 6 | 0 | | ACT | 16 | 0 | 0 | 10 | | TTG | 11 | 8 | 7 | 0 | | CTG | 17 | 17 | 17 | 11 | | TGT | 12 | 9 | 8 | 12 | | GTG | 13 | 0 | 0 | 13 | | TGA | 14 | 0 | 0 | 14 | | GAC | 15 | 0 | 2 | 15 | | CCA | 0 | 5 | 4 | 0 | | GTA | 0 | 10 | 9 | 0 | | TAG | 0 | 11 | 10 | 0 | | AGC | 0 | 12 | 11 | 0 | | GCT | 0 | 13 | 12 | 0 | | CTA | 0 | 14 | 13 | 0 | | CGA | 0 | 0 | 1 | 1 | | ACC | 0 | 0 | 3 | 0 | | CTG | 0 | 0 | 17 | 17 | |
| Alignement (, ) |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | |
| **Paire (, )** |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | G | C | C | A | T | T | G | T | A | G | C | T | A | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14**  gapped-bloc | **15** | **16** | **17** | **18** | **19** | |
| Alignement(, )  x |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | G | C | C | A | T | T | G | T | A | G | C | T | A | - | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | - | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | |
| **Paire (, )** |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | |
| Alignement(, ) |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | |

Figure ‎3‑6 Génération de blocs

En premier lieu, nous générons tous les possibles en suivant l’*Algorithme.3.3*. Nous présentons quelques cas dans le processus d’*identification de blocs* :

|  |
| --- |
| Avec la Paire (, ) : En glissant une fenêtre de longueur sur l’alignement de cette paire afin de produire un ensemble de , nous trouvons par exemple : |
| -Fenêtre avec les *fragments*  :  -Produire l’ensemble tel que le et le :  - Vérifier si le est de type *un-gapped-blocs ? Oui : accepter.*  *-* Vérifiersile et le sont identiques ? oui :  - Vérifier l’existence du bloc dans la matrice M ? non : ajouter une ligne à la matrice M en affectant la position à la case et la position à la case(voir Figure ‎3‑6)  -Fenêtre avec les *fragments*  :  -Produire l’ensemble tel que le et le - :  - Vérifier si le est de type *un-gapped-blocs ? Non: refuser.*  -Fenêtre avec les *fragments*  :  -Produire l’ensemble tel que le et le :  - Vérifier si le est de type *un-gapped-blocs ? Oui : accepter.*  *-* Vérifiersile et le sont identiques ? oui :  - Vérifier l’existence du bloc dans la matrice M ? non : ajouter une ligne à la matrice M en affectant la position à la case et la position à la case |
| Avec la Paire (, ) : En glissant une fenêtre de longueur sur l’alignement de cette paires, nous trouvons par exemple : |
| Fenêtre avec les *fragments*  :  -Produire l’ensemble tel que le et le :  - Vérifier si le est de type *un-gapped-blocs ? Oui : accepter.*  *-* Vérifiersile et le sont identiques ? Non. Gérer les blocs d’une manière isolée. Par exemple pour le  :  - Vérifier l’existence du dans la matrice M ? oui  : appliquer la procédure *Affectation\_bloc\_Avec\_ Contrôle\_brèche().* |

En deuxième lieu, nous appliquons l’étape de filtrage pour (voir Tableau ‎3‑2) :

*-*Éliminer les de type *blocs-1-all*.

-Combiner les de type *blocs-adjacents*. Par exemple dans la matrice M : en combinant les *blocs-adjacents* ("CGA" ; "GAT"; "ATG"; "TGA" ; "GAG" ; "AGT" ; "GTC"; "TCA") de la ligne à la qui disposent des adjacents et dans les mêmes séquences et , nous trouvons le nouvel ensemble avec , , et le "GATGAGTCA".

-Déplacer les de type *blocs-all-all* vers la nouvelle matrice M qui possède tous les blocs présents dans toutes les séquences.

|  |  |  |
| --- | --- | --- |
| **Données** | **Traitement** | **Résultats** |
| Matrice initiale :   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Bloc** |  |  |  |  | | CGA | 1 | 1 | 1 | 1 | | GAT | 2 | 0 | 0 | 2 | | ATG | 3 | 0 | 0 | 3 | | TGA | 4 | 0 | 0 | 4 | | GAG | 5 | 0 | 0 | 5 | | AGT | 6 | 0 | 0 | 6 | | GTC | 7 | 0 | 0 | 7 | | TCA | 8 | 0 | 0 | 8 | | CAT | 9 | 6 | 5 | 0 | | CAC | 0 | 0 | 0 | 9 | | ATT | 10 | 7 | 6 | 0 | | ACT | 16 | 0 | 0 | 16 | | TTG | 11 | 8 | 7 | 0 | | CTG | 17 | 17 | 17 | 17 | | TGT | 12 | 9 | 8 | 12 | | GTG | 13 | 0 | 0 | 13 | | TGA | 14 | 0 | 0 | 14 | | GAC | 15 | 0 | 0 | 15 | | CCA | 0 | 5 | 4 | 0 | | GTA | 0 | 10 | 9 | 0 | | TAG | 0 | 11 | 10 | 0 | | AGC | 0 | 12 | 11 | 0 | | GCT | 0 | 13 | 12 | 0 | | CTA | 0 | 14 | 13 | 0 | | GAC | 0 | 0 | 2 | 0 | | ACC | 0 | 0 | 3 | 0 | |  | | | | | | -Supprimer les bloc-1-all | -Matrice de biregroupement M   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **Bloc** |  |  |  |  | | B1 | GATGAGTCA | 2 | 0 | 0 | 2 | | B2 | CAT | 9 | 6 | 5 | 0 | | B3 | ATT | 10 | 7 | 6 | 0 | | B4 | ACT | 16 | 0 | 0 | 16 | | B5 | TTG | 11 | 8 | 7 | 0 | | B6 | GTGAC | 13 | 0 | 0 | 13 | | B7 | CCA | 0 | 5 | 4 | 0 | | B8 | GTAGCTA | 0 | 10 | 9 | 0 | |  | | | | | |   -Matrices de M   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Bloc** |  |  |  |  | | CGA | 1 | 1 | 1 | 1 | | CTG | 17 | 17 | 17 | 11 | | TGT | 12 | 9 | 8 | 12 |  |  |  | | --- | --- | |  | Blocs all-all | |  | Blocs 1-all | |  | Blocs adjacents | |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Bloc** |  |  |  |  | | CGA | 1 | 1 | 1 | 1 | | GAT | 2 | 0 | 0 | 2 | | ATG | 3 | 0 | 0 | 3 | | TGA | 4 | 0 | 0 | 4 | | GAG | 5 | 0 | 0 | 5 | | AGT | 6 | 0 | 0 | 6 | | GTC | 7 | 0 | 0 | 7 | | TCA | 8 | 0 | 0 | 8 | | CAT | 9 | 6 | 5 | 0 | | ATT | 10 | 7 | 6 | 0 | | ACT | 16 | 0 | 0 | 16 | | TTG | 11 | 8 | 7 | 0 | | CTG | 17 | 17 | 17 | 11 | | TGT | 12 | 9 | 8 | 12 | | GTG | 13 | 0 | 0 | 13 | | TGA | 14 | 0 | 0 | 14 | | GAC | 15 | 0 | 0 | 15 | | CCA | 0 | 5 | 4 | 0 | | GTA | 0 | 10 | 9 | 0 | | TAG | 0 | 11 | 10 | 0 | | AGC | 0 | 12 | 11 | 0 | | GCT | 0 | 13 | 12 | 0 | | CTA | 0 | 14 | 13 | 0 | |  | | | | | |
| -Combiner les blocs-adjacents  -Déplacer les blocs-all-all |

Tableau ‎3‑2 Résultat d’identification de blocs

1. Traitement (Biregroupement)

En utilisant la matrice résultante M (voir Figure ‎3‑7), nous appliquons la fonction *BiMine-modifié()* pour produire l’ensemble de bigroupes cohérents. ( voir Figure ‎3‑8), ainsi, pour établir une cohérence totale entre les séquences.

∅

∅

Figure ‎3‑7 Résultat de BiMine-modifié

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-Données** : (Matrice M) | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **N°** |  |  |  |  |  | |  |  |  |  |  |  | | **B1** | GATGAGTCA | 2 | 0 | 0 | 2 | | **B4** | ACT | 16 | 0 | 0 | 16 | | **B6** | GTGAC | 13 | 0 | 0 | 13 | |  |  |  |  |  |  | | **B2** | CAT | 9 | 6 | 5 | 0 | | **B3** | ATT | 10 | 7 | 6 | 0 | | **B5** | TTG | 11 | 8 | 7 | 0 | |  |  |  |  |  |  | | **B2** | CAT | 9 | 6 | 5 | 0 | | **B3** | ATT | 10 | 7 | 6 | 0 | | **B5** | TTG | 11 | 8 | 7 | 0 | | **B7** | CCA | 0 | 5 | 4 | 0 | | **B83** | GTAGCTA | 0 | 10 | 9 | 0 | |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **N°** |  |  |  |  |  | | **B1** | GATGAGTCA | 2 | 0 | 0 | 2 | | **B2** | CAT | 9 | 6 | 5 | 0 | | **B3** | ATT | 10 | 7 | 6 | 0 | | **B4** | ACT | 16 | 0 | 0 | 16 | | **B5** | TTG | 11 | 8 | 7 | 0 | | **B6** | GTGAC | 13 | 0 | 0 | 13 | | **B7** | CCA | 0 | 5 | 4 | 0 | | **B8** | GTAGCTA | 0 | 10 | 9 | 0 | |
| **-Résultats :** () |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **N°** |  |  |  |  |  | | **B1** | GATGAGTCA | 2 | 2 | 0 | 0 | | **B4** | ACT | 16 | 16 | 0 | 0 | | **B6** | GTGAC | 13 | 13 | 0 | 0 | |  |  |  |  |  |  | | **B2** | CAT | 9 | 0 | 6 | 5 | | **B3** | ATT | 10 | 0 | 7 | 6 | | **B5** | TTG | 11 | 0 | 8 | 7 | | **B7** | CCA | 0 | 0 | 5 | 4 | | **B8** | GTAGCTA | 0 | 0 | 10 | 9 | |
|  |
| -**trouvés avec *BiMine-modifié***  ( )  ( )  ( [],[,) |
| Figure ‎3‑8 Résultat de biregroupement | |

1. Assemblage et Raffinement

En appliquant le biregroupement, nous avons trouvé toutes les séquences en concordance avec tous les blocs . Ainsi, le biregroupement nous a dévoilé des sous-alignements entre les sous-séquences. Pour ce là, cette étape vise à assembler les blocs en suivant les cohérences données par les bigroupes. Etant donné (voir Tableau ‎3‑3) et la matrice M (voir Tableau ‎3‑4), le résultat donné par ce processus d’assemblage se présente comme suit :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **bigroupe** | **Séquence** | **blocs** | |  |  |  | |  |  |  | |  | [] | , | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **N°** | **Bloc** | **S1** | **S2** | **S3** | **S4** | | **B1** | GATGAGTCA | 2 | 0 | 0 | 2 | | **B2** | CAT | 9 | 6 | 5 | 0 | | **B3** | ATT | 10 | 7 | 6 | 0 | | **B4** | ACT | 16 | 0 | 0 | 16 | | **B5** | TTG | 11 | 8 | 7 | 0 | | **B6** | GTGAC | 13 | 0 | 0 | 13 | | **B7** | CCA | 0 | 5 | 4 | 0 | | **B8** | GTAGCTA | 0 | 10 | 9 | 0 | |
| Tableau ‎3‑3 Bigroupes résultants | **Tableau ‎3‑4 Matrice résultante** | |

 Comme nous avons déjà mentionné, ce processus est composé par trois étapes :

* *Assemblage par biregroupement* :

En suivant l’*Algorithme.3.6*, nous assemblons les blocs cohérents présentés dans les bigroupes afin de produire des régions alignées localement (voir Figure ‎3‑9).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  |  |  | **Bloc6** | | | | |  |  | |  |  | **Bloc1** | | | | | | | | |  |  |  |  |  | **Bloc4** | | |  | |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | |  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  | **Bloc5** | | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | **Bloc3** | | |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  | **Bloc2** | | |  |  |  |  |  |  |  |  |  |  |  |  | |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | - | - | - | - | |  | C | G | A | G | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | T | G | - | |  | C | G | A | C | - | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  | Bloc5 | | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | Bloc3 | | |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  | Bloc2 | | |  |  |  |  |  |  |  |  |  |  |  |  | |  | C | G | A | G | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | T | G | - | |  | C | G | A | - | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | |  |  |  |  |  | B7 |  |  |  | B7 | |  |  | Bloc8 | | | | | | |  |  |  |  | |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G | - | - | - | - | |  | C | G | A | G | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | T | G | - | |  | C | G | A | - | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G | |  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G | - | - | - | - | |

Figure ‎3‑9 Assemblage par biregroupement

* *Assemblage des blocs de type bloc-all-all*

Rappelant, que dans l’étape de *génération de blocs,* nous avons déjà filtré les de types *blocs-all-all* qui apparaissent dans toutes les séquences (voir Tableau ‎3‑5). Cette étape vise à les assembler dans les bonnes positions (voir Figure ‎3‑10) en suivant le même principe cité dans l*’Algorithme.3.6 et l’Algorithme.3.7.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Bloc** |  |  |  |  |
| **B1** | CGA | 1 | 1 | 1 | 1 |
| **B2** | CTG | 17 | 17 | 17 | 11 |
| **B3** | TGT | 12 | 9 | 8 | 12 |

Tableau ‎3‑5 Résultat de la matrice Mbloc-all-all

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Bloc1 | | |  |  |  |  |  |  |  |  | Bloc3 | | |  |  |  |  |  |  | Bloc2 | | |
|  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | - | - | - | - | C | T | G |
|  | C | G | A | G | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | - | C | T | G |
|  | C | G | A | - | C | - | - | - | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G |
|  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | - | - | - | - | C | T | G |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |

Figure ‎3‑10 Assemblage des blocs-all-all

* *Raffinement*

Dans cette étape, nous cherchons les *blocs-non-biregroupés* et les *alignements-cachés* en utilisant les alignements obtenus dans l’étape précédente (voir Figure ‎3‑10). En effet, en appliquant *BiMine-modifié* sur ce groupe de séquences d’ADN, aucun des blocs générés n’a été écarté. De ce fait, nous nous intéressons qu’à présenter le principe de la recherche des *alignements-cachés* entre les *(voir Algorithme.3.8).*

Comme nous l’avons déjà mentionné, l’étape de raffinement (voir Figure ‎3‑11) améliore la qualité de l’alignement et aide à dévoiler des *alignements-cachés* en appliquant deux sous-étapes : le *nettoyage en suffixe* et le *nettoyage en préfixe*. La Figure ‎3‑11 présente le résultat final après le *Raffinement*.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | - | G | - | - | A | - | C | T | G |
|  |  | C | G | A | - | - | - | G | C | C | A | T | T | G | T | A | G | C | T | A | - | C | T | G |
|  |  | C | G | A | - | - | - | - | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G |
|  |  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | - | G | - | - | A | - | C | T | G |

Figure ‎3‑11 Résultat de Raffinement

Résultat Final :

Le résultat d’alignement donné dans la Figure ‎3‑13est un alignement optimal des séquences d’ADN (voir Figure ‎3‑12). En comparant les résultats de *BicMSA* avec les algorithmes les plus connus dans la littérature, nous avons utilisé le programme de DIALIGN-Tx donné dans le site officiel http://dialign-tx.gobics.de/submission?type=dna, ainsi, les programmes de T-coffee, clustal-omega, clustalx, muscle et mafft présentés dans le site http://www.ebi.ac.uk/Tools/msa/. Nous avons exécuté ces programmes sur les séquences d’ADN en utilisant leur paramètre par défaut. Comme le montre la Figure 3-14, *BicMSA* donne les meilleurs résultats par rapport aux algorithmes globaux et locaux. À ce stade, nous remarquons que l’utilisation de biregroupement dans l’alignement multiple fourni les meilleur résultats.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Groupe ADN non aligné** |  |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | G | A | C | T | G |
|  |  | C | G | A | G | C | C | A | T | T | G | T | A | G | C | T | A | C | T | G |
|  |  | C | G | A | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G |
|  |  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | G | A | C | T | G |

Figure ‎3‑12 Les Groupes de séquences d’ADN à aligner.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Groupe**  **ADN aligné :** |  | C | G | A | T | G | A | G | T | C | A | T | T | G | T | - | G | - | - | A | - | C | T | G |
|  | C | G | A | - | - | - | G | C | C | A | T | T | G | T | A | G | C | T | A | - | C | T | G |
|  | C | G | A | - | - | - | - | C | C | A | T | T | G | T | A | G | C | T | A | C | C | T | G |
|  | C | G | A | T | G | A | G | T | C | A | C | T | G | T | - | G | - | - | A | - | C | T | G |

Figure ‎3‑13 Résultat final de *BicMSA*

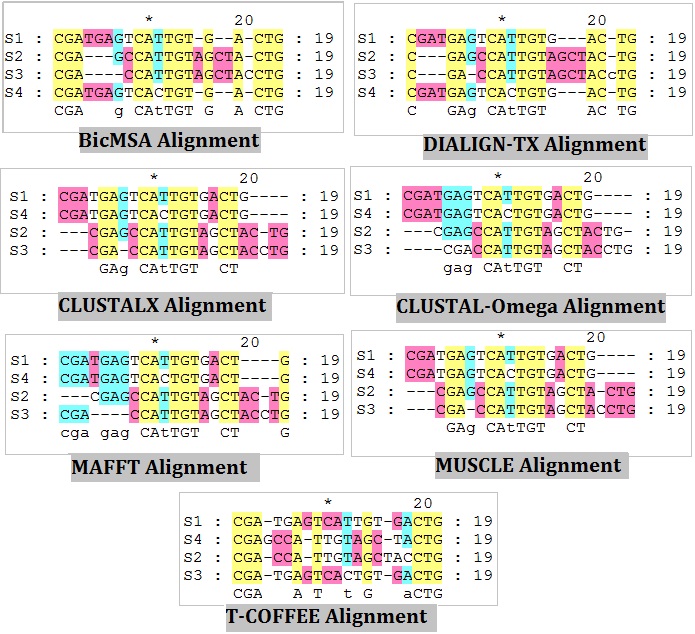


Figure ‎3‑14 Comparaison de BicMSA avec quelques algorithmes (le pourcentage d’identité : jaune=100%, bleu=80%, rose=60%)

## Discussion

L’objectif de *BicMSA* est de produire un alignement multiple et local d’un ensemble de séquences en utilisant la technique de biregroupement. De ce fait, nous nous sommes inspirées de l’algorithme *BlockMSA* [Wang et al., 2007]. Comme nous l’avons déjà mentionné, *BlockMSA* est un algorithme basé sur l’approche *DAC*. Bien que les deux algorithmes adoptent la même technique dans la résolution du problème de *MSA*, ils se différent sur les points suivants :

Génération de la bibliothèque des meilleurs alignements globaux :

*BlockMSA* utilise une approche heuristique et progressive présentée par *CLUSTALW* [Thompson et al., 1994]. En effet, *CLUSTALW* ne résout toujours pas le principal défaut de l'alignement progressif : « la propagation des erreurs ». Proprement dit, la qualité de l’alignement généré peut être affectée par des erreurs réalisées au début du processus d’alignement. Ces erreurs ne peuvent être corrigées par la suite. En outre, l’utilisation des arbres de guidage peut influer la précision des alignements [Nelesen et al., 2008] ainsi la méthode utilisée dans la construction d’arbre n’est fiable pas qu’avec les séquences peu divergentes [Philippe, 2005].Cependant, comme nous cherchons toujours à utiliser des solutions exactes afin de garantir l’optimalité des résultats fournis et de maximiser la cohérence entre les blocs, nous avons utilisé l’algorithme exact *Needleman* et *Wunch* [Needleman et Wunsch, 1970]. C’est un algorithme basé sur la *programmation dynamique*. Il permet de produire des alignements optimaux. Le choix d’une approche exact est établi afin de fournir des blocs possédant des cohérences maximales. Tandis que la technique globale est utilisée à cause de sa capacité de fournir le maximum des fragments en concordance entre deux séquences et à cause de sa qualité d’alignement par paire par rapport à la technique local.

Construction de la matrice M :

Après la génération des meilleurs alignements, *BlockMSA* ne garde que les de type*un-gapped-blocs* en imposant un seuil sur le score d’alignement entre les blocs à assembler. Tandis que, *BicMSA* ne récupère que les de type*un-gapped-blocs* qui possèdent une similarité maximale. Proprement dit, il ne garde que les blocs avec des fragments identiques.

Par ailleurs, les deux algorithmes se divergent dans le type de la matrice M contenant les séquences et leurs blocs associés. Contrairement à *BlockMSA* qui présente une matrice binaire dont ses éléments présentent l’existence ou l’inexistence d’un dans une séquence , *BicMSA* fourni une matrice de positions dont les lignes présentent les et les colonnes présentent les séquences . Chaque prend soit la position de dans la séquence ou bien la valeur , si le n’existe pas dans .

Approche de biregroupement :

*BlockMSA* adopte l’approche *DAC* en utilisant l’algorithme *BiMAX* [Prelic et al., 2006] afin d’identifier les bigroupes cohérents à partir d’une matrice binaire [Wang et al., 2007]. En effet, la qualité de biregroupement a une grande influence sur la qualité d’alignement. De ce fait, il est nécessaire, d’une part, utiliser une approche différente de *DAC* puisqu’elle peut perdre de bons bigroupes avec le partitionnement avant l’identification [Madeira et Oliveira, 2004 ; Prelic et al., 2006]. D’autre part, nous devons appliquer un algorithme différent de *BiMAX* en raison de ces inconvénients. En effet, la qualité des bigroupes générés par *BiMAX* dépend totalement des paramètres initialisés. En outre, *BiMAX* est dédié pour les matrices de données binaires. Et nous pouvons noter aussi que sa complexité est de l’ordre de pour une matrice () avec est le nombre de bigroupes à inclusion maximale calculé par .

Un survol de la littérature nous a montré que l’approche d’*énumération de bigroupe*permet detrouver, d’une manière optimale, un ensemble de meilleurs bigroupes en utilisant une énumération implicite ou explicite de toutes les solutions possibles pour un problème donné [Madeira et Oliveira, 2004]. Il existe plusieurs algorithmes adoptants cette approche. Parmi ces algorithmes, nous avons eu recourt à adapter l’algorithme *BiMine* [Ayadi et al., 2009]*.* En effet, *BiMine* permet de générer tous les meilleurs bigroupes existants en utilisant une nouvelle structure d'arbre, appelée *Bicluster* *Enumeration* *Tree* (BET) [Ayadi et al., 2009]*.* Il dispose trois étapes principales : *Prétraitement*, *Traitement* et *Post-Traitement.*

Par ailleurs, les modifications, apportées pour adapter *BiMine* [Ayadi et al., 2009]à être utiliser dans un problème de MSA local, sont réalisées par une fonction appelée *BiMine-modifié()*. Elles se présentent comme suit :

Modification du *Prétraitement* : Lorsque nous changeons les données de la matrice de *BiMine* de gènes/conditions à blocs/séquences et nous appliquons l’étape de *Prétraitement*, l’algorithme néglige l’existence de certaines similarités entre les séquences. De ce fait, nous avons éliminé la partie de filtrage à cause de la différence entre les données à traiter (*séquences/blocs*) (*gènes/conditions*).

Modification du *Traitement* : Dans le processus de la construction de *BET*, BiMine applique un traitement sur les fils de l’arbre (bigroupes cohérents) en appelant la fonction d'évaluation *Average Spearman's Rho (ASR)* [Ayadi et al., 2009]afin de contrôler la qualité des bigroupes. Cependant, notre objectif est d’extraire toutes les dépendances entre les blocs et les séquences, proprement dit, trouver tous les bigroupes quelle que soit leur qualité. Alors, nous avons ignoré les testes de la fonction *ASR* afin de maximiser la similarité entre les séquences.

Modification du *Post-Traitement :* Après la génération de toutes les feuilles de *BET*, une procédure de suppression intervient pour supprimer tous les bigroupes qui sont inclus dans d'autres ou bien les bigroupes qui ont moins de deux gènes (séquences) ou moins de deux conditions (blocs)[Ayadi et al., 2009]. En appliquant cette étape à la matrice M (séquences/blocs), *BiMine* supprime des relations entre les blocs et les séquences. De ce fait, nous avons ignoré cette étape puisque, en appliquant le nettoyage des de type *blocs-1-all* sur la matrice M, nous ne pouvons pas avoir des bigroupes disposant moins de deux séquences. Ainsi, les bigroupes, possédant un seul bloc, présentent des similarités entre deux ou plusieurs séquences.

Assemblage des blocs

*BlockMSA* et *BicMSA* utilisent deux techniques différentes dans le processus d’assemblage des blocs. En effet, puisque *BlockMSA* utilise une matrice binaire, contrairement à *BicMSA* qui utilise une matrice des positions, il applique un processus récursif d’assemblage des blocs basé sur un *graphe orienté acyclique DAG (Directed Acyclic Graph)* [Dan, 1997]. Tandis que *BicMSA* applique un simple placement des blocs à leurs bonnes positions en utilisant les données de la matrice M.

## Conclusion

Dans ce chapitre, nous avons présenté notre algorithme d’alignement multiple de séquence. *BicMSA (****Bic****lustering-based local* ***MSA***) permet de résoudre un problème de MSA local en s’appuyant sur l’approche *énumération de bigroupes* pour le *biregroupement* et l’approche *DAC (Diviser-et-conquérir)*. Comme montré auparavant, *BicMSA* et *BlockMSA* partagent la même technique de *biregroupement* dans la résolution du problème MSA. Cependant, ils se divergent dans l’utilisation de la méthode de construction de blocs, la matrice M, l’approche et l’algorithme de biregroupement ainsi que d’assemblage des blocs.

Son principe réside en trois étapes principales. En premier lieu (*Prétraitement)*, *BicMSA* divise un ensemble d’alignements par paire optimaux et globaux afin de maximiser la cohérence entre les blocs. En deuxième lieu (*Traitement*), il applique la technique de biregroupement pour augmenter la cohérence entre toutes les séquences. En troisième lieu (*Post-Traitement*), *BicMSA* fournit son alignement local en assemblant les blocs générés suivant les cohérences données par les bigroupes, ainsi un processus de raffinement est appliqué afin d’améliorer la qualité de l’alignement.

À la fin de ce chapitre, nous avons expliqué, avec exemple illustratif, comment *BicMSA* fonctionne, ainsi, nous avons présenté la différence entre *BlockMSA* et *BicMSA* et les améliorations apportées par rapport à cet algorithme.

Dans le chapitre suivant, nous allons mener à une série d'expérimentations sur un ensemble de jeux de données réelles afin de mesurer la performance de *BicMSA* dans la résolution des problèmes d’alignement des séquences biologiques (*ADN*, *ARN* et *protéines*) et de vérifier la signification biologique de ses résultats.

# Étude Expérimentale

## Introduction

Dans ce chapitre, nous effectuons une étude expérimentale sur notre algorithme d’alignement multiple des séquences biologiques basé sur le biregroupement, *BicMSA (****Bic****lustering-based local* ***MSA***). L’objectif de *BicMSA* est de proposer un alignement qui soit le plus proche possible de ce que peut faire un biologiste. Ainsi, il est très important d’évaluer sa performance d’un point de vue biologique. De ce fait, nous comparons les résultats obtenus à ceux obtenus par une sélection d’algorithmes les plus célèbres en utilisant des jeux de données réelles. Dans cette partie expérimentale, nous avons envisagé deux optiques différentes pour la réalisation des tests sur les séquences réelles. La première consiste à mesurer la performance de BicMSA avec les alignements de séquences possédant des caractéristiques biologiques différentes (*ADN*, *ARN*, *protéines*), la seconde consiste à mesurer la qualité des résultats obtenus en les comparant soit avec des alignements de références locaux soit avec des alignements de références globaux.

Dans la première partie, nous présentons les jeux de données utilisés et les critères de comparaison appliqués. La deuxième partie consiste à établir une évaluation des performances de *BicMSA* ainsi qu’une comparaison avec les algorithmes les plus connus dans la littérature. Nous terminons ce chapitre par une conclusion.

## Les jeux de données utilisés

Pour évaluer la performance d’un algorithme MSA, plusieurs alignements de séquences réelles ont été construits manuellement ou automatiquement par des biologistes et stockées dans différentes bases de jeux (voir Tableau ‎4‑1). Cependant, pour tester la capacité de *BicMSA* dans l’alignement de différents types de séquences, nous avons utilisé les bases de jeux les plus connues dans l’évaluation des alignements locaux et globaux : *BALIBASE\_* [Thompson et al., 1999] pour les séquences protéiques, *BRALIBASE\_* [Gardner et Giegerich, 2004] pour les séquences d’*ARN*s et *DIRMBASE\_* [Stoye et al., 1998] pour les séquences d’*ADN*s.

|  |  |
| --- | --- |
| **Base des données** | **Liens** |
| BALIBASE | http://-biod-igbme.u-strasbg.fr/balibase |
| BRALIBASE | http://projects.binf.ku.dk/pgardner/bralibase/ |
| HOMSTRAD | http://-cryst.bioc.cam.ac.uk/homstrad// |
| IRMBASE et DIRMBASE | http://dialign-t.gobics.de/main |
| PREFAB | http://driver5.com/muscle/prefab.htm |
| OXBENCH | http://compbio.dundee.ac.uk/Softare/Oxbench/oxbench.htm |
| SABMARK | http://bioinformatics.vub.ac.be/databases/content.html |

Tableau ‎4‑1 Liste des bases de jeux d’essai [Mokaddem et Elloumi, 2011]

### BALIBASE (Protéines)

*BALIBASE* [Thompson et al., 1999] est la première base des jeux d’essai la plus utilisée dans l’évaluation des alignements multiples et globaux des protéines [Thompson et al., 1999]. La première version comporte un nombre suffisant de jeux d’essai pour pouvoir définir une tendance générale d’un algorithme. Ces jeux sont répartis en catégories suivant la nature des ensembles de séquences, appelées *références*. Chaque référence présente un problème d’alignement différent. Ces alignements prennent en compte différents cas tels que : le nombre de séquences, la longueur des séquences, la similarité entre les séquences, l’alignement des séquences divergentes avec une séquence “*orpheline*”, l’alignement de sous-familles et l’alignement de séquences de longueur différente (insertions/extensions). Par exemple, la référence contient de à séquences (petites, moyennes, longues) disposant un pourcentage de similitude qui peut être soit , ou bien . Tandis que la référence comporte des familles de séquences de tailles très différentes. Ces séquences ne disposent aucune similarité entre elles, mais elles sont alignées avec un, deux ou bien trois séquences *orphelines*. Leurs alignements sont pénalisés par la présence des très grandes brèches. Afin de mesurer la qualité des alignements fournis par *BicMSA*, nous avons sélectionné un ensemble de familles de différentes références disposant des différentes caractéristiques. Cette sélection est établie en fonction des tableaux de comparaison donnés dans le site de BALIBASE (voir Tableau ‎4‑2).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **BALIBASE- Groupes de familles testées** | | | | | | | |
| **∑ Famille** | **Catégorie** | **Caractéristiques des familles** | | | | | **Nombre**  **famille** |
| **Nb Séquences** | **Taille Seq** | **% identité** | **N-insertion** | **C-extensions** |
| **∑ F1** | Ref\_ 1 | ≥ 4 et ≤ 6 |  | < 25% ; < 40% |  |  | 26 |
| **∑ F2** | Ref\_ 2 | ≥ 15 et ≤ 24 |  | < 25% |  |  | 23 |
| **∑ F3** | Ref\_ 3 | ≥ 16 et ≤ 28 |  | ≥ 25% |  | résidus | 16 |
| **∑ F 4** | Ref\_ 4 | ≥ 4 et ≤ 20 |  | - |  | résidus | 16 |
| **∑ F 5** | Ref\_ 5 | ≥ 5 et ≤ 19 |  | - |  |  | 12 |
|  |  |  |  |  | **∑ Nombre de familles testées** | | **93** |

Tableau ‎4‑2 BALIBASE - familles utilisées.

### BRALIBASE (ARN)

La base *BRALIBASE*\_ [Gardner et Giegerich, 2004] a été massivement utilisée dans l’évaluation des algorithmes d’alignement. Elle offre des alignements globaux d’un ensemble d’*ARNs* présentés sous forme des familles. En effet, les familles de testes sont caractérisées par le nombre de séquences (de à séquences) et le pourcentage d’identité (de à ). Dans le but de mesurer les capacités de *BicMSA* avec les séquences d’*ARNs*, nous avons sélectionné les familles les plus utilisées dans l’évaluation des algorithmes avec un nombre de séquences varie entre et . Parmi ces familles, nous citons la famille *THI*, *Yybp-Ykoy*, *Entero\_OriR*, *gcvT*, *SECIS*, *T-box*, *t-RNA* et *S-box* (voir Tableau ‎4‑3).

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BRALIBASE-Groupes de familles testées.** | | | | | | | | | | |
| **Nb Séq** | **Famille** | | | | | | | | **∑ Famille** | **∑ Séquences** |
| **THI** | **Yybp-Ykoy** | **Entero\_OriR** | **gcvT** | **SECIS** | **T-box** | **t-RNA** | **S-box** |
| n=2 | 22 | 17 | 8 | 13 | 2 | 5 | 3 | 0 | 70 | 140 |
| n=7 | 9 | 7 | 10 | 8 | 3 | 0 | 2 | 5 | 44 | 308 |
| n=10 | 17 | 12 | 5 | 3 | 0 | 0 | 4 | 0 | 41 | 300 |
| n=15 | 21 | 9 | 4 | 12 | 0 | 0 | 6 | 0 | 52 | 720 |
|  |  |  |  |  |  |  | **Nb Total** | | **207** | **1468** |

Tableau ‎4‑3 BRALIBASE- familles utilisées.

### DIRMBASE (ADN)

DIREMBASE [Stoye et al., 1998] est la base la plus utilisée dans l’évaluation de la performance des algorithmes d’alignement d’*ADN*. Elle se compose d’un ensemble de séquences réparties en quatre références. Chaque référence se compose de 48 familles d’*ADN*, dont 24 familles contiennent *ROSE-motifs* de longueur 30, alors que les familles restantes contiennent des motifs de différentes longueurs. Pour évaluer le comportement de *BicMSA,* nous avons généré un ensemble d’alignement pour toutes les séquences dans toutes les références de *DIRMBASE* (voir Tableau ‎4‑4).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DIRMBASE-Groupes de séquences testées** | | | | | |
| **Référence** | **Nb Séquences/famille** | | **Nb de famille** | **Taille de Seq/famille** | **Nb total Seq/ Ref** |
| **Référence 1** | **∑F1** | 4 Séq/famille | 16 |  | 320 |
| **∑F2** | 8 Séq/ famille |
| **∑F3** | 16 Séq/ famille |
| **Référence 2** | **∑F1** | 4 Séq/ famille | 16 |  | 320 |
| **∑F2** | 8 Séq/ famille |
| **∑F3** | 16 Séq/ famille |
| **Référence 3** | **∑F1** | 4 Séq/ famille | 16 |  | 320 |
| **∑F2** | 8 Séq/ famille |
| **∑F3** | 16 Séq/ famille |
| **Référence 4** | **∑F1** | 4 Séq/ famille | 16 |  | 320 |
| **∑F2** | 8 Séq/ famille |
| **∑F3** | 16 Séq/ famille |
|  | **∑ Nombre de familles testées** | | **64** | **∑ Nombre de séquences testées** | **1280** |

Tableau ‎4‑4 DIRMBASEE - familles utilisées.

## Critères de comparaison

Afin d’évaluer la qualité ou la signification des alignements, chaque base offre ses propres critères de comparaison. Ces critères se présentent avec des fonctions d’évaluation permettant de comparer des alignements générés par rapport à des alignements de références. Parmi ces fonctions, nous citons :

* *SPS* (*Score Somme-des-Paires*) [Lassmann et Sonnhammer, 2002] : C’est une fonction d’évaluation locale basée sur le principe de *SP* (*Somme-des-Paires*). Elle présente le rapport entre le nombre de paires de caractères correctement alignés et la totalité de nombre des paires de caractères dans les blocs.
* *CS* (*Score des Colonnes*) [Lassmann et Sonnhammer, 2002] : Elle permet d’évaluer, d’une manière globale, un alignement fourni par un algorithme *MSA*. Elle présente un rapport entre le nombre de colonnes correctement alignées et la totalité des nombres de colonnes dans les blocs.
* Autre score : le « *overlap score* » [Lassmann et Sonnhammer, 2002], le « *shift score* » [Cline et al., 2002] et le « *mean opinion score* » (*MOS*) [Lassmann et Sonnhammer, 2005]

Cependant, suite à notre choix de bases de données qui utilisent les mêmes scores d’évaluation (le *SPS* et le *CS*), nous avons évalué *BicMSA* en générant des scores d’alignement avec un programme proposé par *BALIBASE*: *bali\_score.c*. En effet, le score SPS est spécifiquement choisi en raison de sa capacité à évaluer des alignements locaux.

## Analyse de sensibilité

Dans le but de spécifier qu’elles sont les tailles des blocs contribuant le plus à la qualité des résultats d’alignement et ceux les moins influents, nous avons consacré cette section pour étudier la sensibilité de l'algorithme proposé avec la variation de la taille du bloc.

### Choix de la taille du bloc

Cette étape vise à déterminer le meilleur alignement, pour chaque jeu d’essai, que *BicMSA* est en mesure de réaliser. Cela revient, alors, à déterminer la meilleure taille du bloc à fixer. De ce fait, nous avons généré différents alignements pour chaque type de séquences (*ADN*, *ARN* et *protéines*). Ce choix est argumenté par l’aspect biologique différent pour chaque type de séquences. En effet, nous générons les résultats des *SPS* de chaque alignement en variant la taille du bloc de à . Ainsi suivant ces valeurs, nous calculons la moyenne et la valeur maximale des *SPS* pour chaque bloc. Le processus d’identification de la taille du bloc pour chaque type de séquence se présente comme suit :

D’abord, en utilisant la base *BRALIBASE* (spécifiquement les ensembles cités dans le Tableau ‎4‑3), nous avons fixé à peu prés la meilleure taille du bloc. Ainsi, en suivant le Tableau ‎4‑5, nous constatons que le bloc de taille donne les meilleurs alignements.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **BRALIBASE** | | | | | | | | | | | | | | | | |
|  | **Nom** | **NB TEST** |  | **k=2** | **k=3** | **k=4** | **k=5** | **k=6** | **k=7** | **k=8** | **k=9** | **k=10** | **k=11** | **k=12** | **k=13** | **k=14** | **k=15** |
| **Nb Seq=2** | THI | 7 | **Moyenne** | 0,909 | 0,917 | 0,949 | 0,947 | 0,957 | 22:57 | 0,957 | 0,916 | 0,868 | 0,801 | 0,776 | 0,714 | 0,647 | 0,611 |
| yybP-ykoY | 5 | 0,960 | 0,953 | 0,961 | 0,961 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 | 0,963 |
| T-box | 5 | 0,577 | 0,754 | 0,744 | 0,755 | 0,763 | 0,764 | 0,755 | 0,690 | 0,693 | 0,693 | 0,693 | 0,693 | 0,693 | 0,693 |
| SECIS | 2 | 0,502 | 0,619 | 0,579 | 0,606 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 | 0,684 |
| T-RNA | 3 | 0,471 | 0,531 | 0,631 | 0,631 | 0,660 | 0,660 | 0,660 | 0,744 | 0,744 | 0,744 | 0,744 | 0,566 | 0,566 | 0,566 |
| **Nb Seq=7** | THI | 3 | **Moyenne** | 0,250 | 0,284 | 0,333 | 0,490 | 0,588 | 0,604 | 0,467 | 0,389 | 0,360 | 0,351 | 0,351 | 0,353 | 0,349 | 0,369 |
| yybP-ykoY | 3 | 0,141 | 0,156 | 0,192 | 0,175 | 0,305 | 0,341 | 0,368 | 0,362 | 0,362 | 0,359 | 0,349 | 0,338 | 0,338 | 0,338 |
| S\_box | 5 | 0,313 | 0,433 | 0,433 | 0,427 | 0,544 | 0,557 | 0,561 | 0,462 | 0,453 | 0,419 | 0,400 | 0,420 | 0,414 | 0,422 |
| SECIS | 3 | 0,175 | 0,322 | 0,242 | 0,261 | 0,504 | 0,451 | 0,432 | 0,413 | 0,308 | 0,362 | 0,345 | 0,345 | 0,345 | 0,345 |
| T-RNA | 2 | 0,152 | 0,166 | 0,205 | 0,230 | 0,572 | 0,638 | 0,638 | 0,638 | 0,477 | 0,511 | 0,498 | 0,498 | 0,498 | 0,498 |
| **Nb Seq=10** | THI | 5 | **Moyenne** | 0,155 | 0,084 | 0,112 | 0,202 | 0,384 | 0,477 | 0,376 | 0,368 | 0,345 | 0,277 | 0,321 | 0,320 | 0,268 | 0,261 |
| yybP-ykoY | 3 | 0,119 | 0,162 | 0,176 | 0,220 | 0,379 | 0,357 | 0,336 | 0,326 | 0,321 | 0,321 | 0,289 | 0,280 | 0,280 | 0,280 |
| T-RNA | 4 | 0,062 | 0,095 | 0,107 | 0,268 | 0,391 | 0,465 | 0,502 | 0,493 | 0,476 | 0,476 | 0,448 | 0,442 | 0,441 | 0,441 |
| **N b Seq=15** | THI | 21 | **Moyenne** | 0,126 | 0,146 | 0,182 | 0,322 | 0,495 | 0,462 | 0,429 | 0,351 | 0,328 | 0,312 | 0,302 | 0,298 | 0,293 | 0,293 |
| yybP-ykoY | 5 | 0,110 | 0,133 | 0,137 | 0,189 | 0,291 | 0,320 | 0,316 | 0,303 | 0,297 | 0,291 | 0,283 | 0,277 | 0,279 | 0,278 |
| T-RNA | 6 | 0,064 | 0,044 | 0,059 | 0,156 | 0,351 | 0,391 | 0,390 | 0,387 | 0,381 | 0,378 | 0,376 | 0,374 | 0,371 | 0,371 |
| gcvT | 12 | 0,130 | 0,168 | 0,224 | 0,302 | 0,429 | 0,415 | 0,359 | 0,344 | 0,292 | 0,268 | 0,255 | 0,229 | 0,227 | 0,216 |
| Entero\_OriR | 4 | 0,468 | 0,686 | 0,626 | 0,669 | 0,721 | 0,634 | 0,595 | 0,837 | 0,810 | 0,838 | 0,839 | 0,882 | 0,878 | 0,879 |
|  | **Moyenne Par bloc** | | | 0,316 | 0,370 | 0,383 | 0,434 | 0,554 | **0,563** | 0,544 | 0,537 | 0,509 | 0,503 | 0,495 | 0,482 | 0,474 | 0,473 |

**Tableau ‎4‑5 BRALIBASE – Résultats des blocs**

Ensuite, afin de fixer le meilleur paramètre avec les séquences d’*ADN*s, nous avons utilisé tous les alignements de références présentés par *DIRMBASE* (voir Tableau ‎4‑4). Ainsi, en se basant sur le Tableau ‎4‑6, nous constatons que *BicMSA* donne les meilleurs résultats avec des blocs disposant une taille égale à .

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **DIRMBASE** | | | | | | | | | | | | | | |
|  |  | **k=2** | **k=3** | | **k=4** | **k=5** | **k=6** | **k=7** | **k=8** | **k=9** | **k=10** | **k=11** | **k=12** | **k=13** | **k=14** | **k=15** |
| **Réf\_1** | r1-dna-400-4 | 0,097 | 0,060 | | 0,200 | 0,592 | 0,68 | 0,908 | 0,908 | 0,908 | 0,91 | 0,978 | 0,908 | 0,908 | 0,908 | 0,908 |
| r1-dan-400-8 | 0,097 | 0,060 | | 0,200 | 0,565 | 0,809 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 |
| r1-dan-400-16 | 0,060 | 0,050 | | 0,170 | 0,210 | 0,212 | 0,136 | 0,136 | 0,500 | 0,502 | 0,554 | 0,524 | 0,524 | 0,524 | 0,524 |
| **Réf\_2** | r2-dan-500-4 | 0,068 | 0,079 | | 0,108 | 0,207 | 0,412 | 0,469 | 0,469 | 0,472 | 0,492 | 0,876 | 0,876 | 0,876 | 0,876 | 0,876 |
| r2-dan-500-8 | 0,050 | 0,059 | | 0,071 | 0,128 | 0,217 | 0,307 | 0,378 | 0,428 | 0,662 | 0,994 | 0,994 | 0,994 | 0,994 | 0,994 |
| r2-dan-500-16 | 0,050 | 0,050 | | 0,050 | 0,105 | 0,210 | 0,217 | 0,367 | 0,378 | 0,501 | 0,582 | 0,582 | 0,582 | 0,582 | 0,582 |
| **Réf\_3** | r3-dan-500-4 | 0,081 | 0,097 | | 0,170 | 0,210 | 0,412 | 0,469 | 0,469 | 0,472 | 0,492 | 0,951 | 0,941 | 0,941 | 0,941 | 0,941 |
| r3-dan-500-8 | 0,060 | 0,077 | | 0,265 | 0,216 | 0,309 | 0,409 | 0,421 | 0,482 | 0,577 | 0,851 | 0,821 | 0,821 | 0,821 | 0,821 |
| r3-dan-500-16 | 0,022 | 0,027 | | 0,175 | 0,255 | 0,262 | 0,298 | 0,225 | 0,325 | 0,502 | 0,519 | 0,519 | 0,519 | 0,518 | 0,518 |
| **Réf\_4** | r4-dan-600-4 | 0,062 | 0,187 | | 0,229 | 0,412 | 0,412 | 0,457 | 0,467 | 0,478 | 0,502 | 0,796 | 0,796 | 0,706 | 0,706 | 0,706 |
| r4-dan-600-8 | 0,025 | 0,056 | | 0,085 | 0,207 | 0,278 | 0,292 | 0,452 | 0,472 | 0,472 | 0,960 | 0,960 | 0,955 | 0,955 | 0,955 |
| r4-dan-600-16 | 0,022 | 0,051 | | 0,125 | 0,136 | 0,151 | 0,236 | 0,315 | 0,356 | 0,378 | 0,458 | 0,450 | 0,450 | 0,450 | 0,450 |
| **Moyenne des SPs** | | 0,058 | 0,071 | | 0,154 | 0,270 | 0,363 | 0,433 | 0,467 | 0,523 | 0,582 | **0,793** | 0,781 | 0,773 | 0,773 | 0,773 |
| **Max SPs** | | 0,097 | | 0,187 | 0,265 | 0,592 | 0,809 | 1,000 | 1,000 | 1,000 | 1,000 | **1,000** | 1,000 | 1,000 | 1,000 | 1,000 |

**Tableau ‎4‑6 DIRMBASE – Résultats des blocs**

Enfin, nous avons réalisé nos alignements avec les séquences protéiques, spécifiquement avec *BALIBASE* (voir Tableau ‎4‑7). Les valeurs sont générées à partir des familles citées dans le Tableau ‎4‑2. En se basant sur les premiers testes réalisés, nous pouvons avoir le meilleur alignement de séquences protéiques en fixant le paramètre à .

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **BALIBASE** | | | | | | | | | | | | | |
|  | **k=2** | **k=3** | **k=4** | **k=5** | **k=6** | **k=7** | **k=8** | **k=9** | **k=10** | **k=11** | **k=12** | **k=13** | **k=14** | **k=15** |
| **∑ F 1** | 0,662 | 0,875 | 0,888 | 0,709 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 | 0,681 |
| **∑ F2** | 0,237 | 0,255 | 0,990 | 0,478 | 0,579 | 0,769 | 0,861 | 0,861 | 0,880 | 0,984 | 0,984 | 0,984 | 0,990 | 0,990 |
| **∑ F3** | 0,260 | 0,260 | 0,868 | 0,650 | 0,758 | 0,758 | 0,867 | 0,867 | 0,316 | 0,282 | 0,242 | 0,242 | 0,242 | 0,242 |
| **∑ F4** | 0,136 | 0,329 | 0,348 | 0,315 | 0,305 | 0,174 | 0,174 | 0,174 | 0,174 | 0,167 | 0,167 | 0,167 | 0,167 | 0,167 |
| **∑ F5** | 0,256 | 0,256 | 0,428 | 0,366 | 0,302 | 0,314 | 0,266 | 0,231 | 0,226 | 0,240 | 0,222 | 0,214 | 0,210 | 0,210 |
| **Moyenne** | 0,310 | 0,395 | **0,704** | 0,504 | 0,525 | 0,539 | 0,570 | 0,563 | 0,455 | 0,471 | 0,459 | 0,458 | 0,458 | 0,458 |
| **Max SPs** | 0,662 | 0,875 | **0,990** | 0,709 | 0,758 | 0,769 | 0,867 | 0,867 | 0,880 | 0,984 | 0,984 | 0,984 | 0,990 | 0,990 |

**Tableau ‎4‑7 BALIBASE – Résultats des blocs**

## Résultats expérimentaux sur des données réelles

Dans cette section, nous allons présenter les résultats obtenus sur des jeux de données réelles. Les évaluations ont été faites avec les fonctions de comparaison : le *SPS* et le *CS*. Ainsi, les valeurs générées sont les moyennes ou les valeurs maximales des scores obtenus. Elles ont été calculées en prenant en compte le nombre de jeux d’essais dans chaque référence et dans chaque famille. Par ailleurs, nous comparons notre algorithme avec les algorithmes les plus connus dans la littérature tels que : *BlockMSA* [Wang et al., 2007], *PRRP* [Gotoh, 1996], *CLUSTALX* [Thompson et al., 1997], *SAGA* [Notredame et Higgins, 1996], *DIALIGN* [Morgenstern, 1999], *SB\_PIMA* [Smith et Smith, 1992], *ML\_PIMA* [Smith et Smith, 1992], *MULTALIGN* [Corpet, 1988], *HMMT* [Eddy, 1995]. Pour ces algorithmes, nous avons récupéré les scores générés avec leurs paramètres par défaut. Par ailleurs, pour les scores des séquences de protéines et d’*ARN*, nous avons récupéré les résultats de ces algorithmes à partir du site officiel de la base *BALIBASE* [Thompson et al., 1999] et *BRALIBASE* [Gardner et Giegerich, 2004]. Concernant, les séquences d’*ADN,* nous avons reporté les scores publiés dans l’article de *DIALIGN-TX* [Subramanian et al., 2008]. Par ailleurs, pour les résultats de comparaison avec *BlockMSA*, nous avons reporté les scores publiés dans l’article de Wang et al [Wang et al., 2007]. En effet, pour chaque comparaison, nous présentons les scores SPS et les scores CS inférieurs aux scores donnés par *BicMSA* par la couleur jaune.

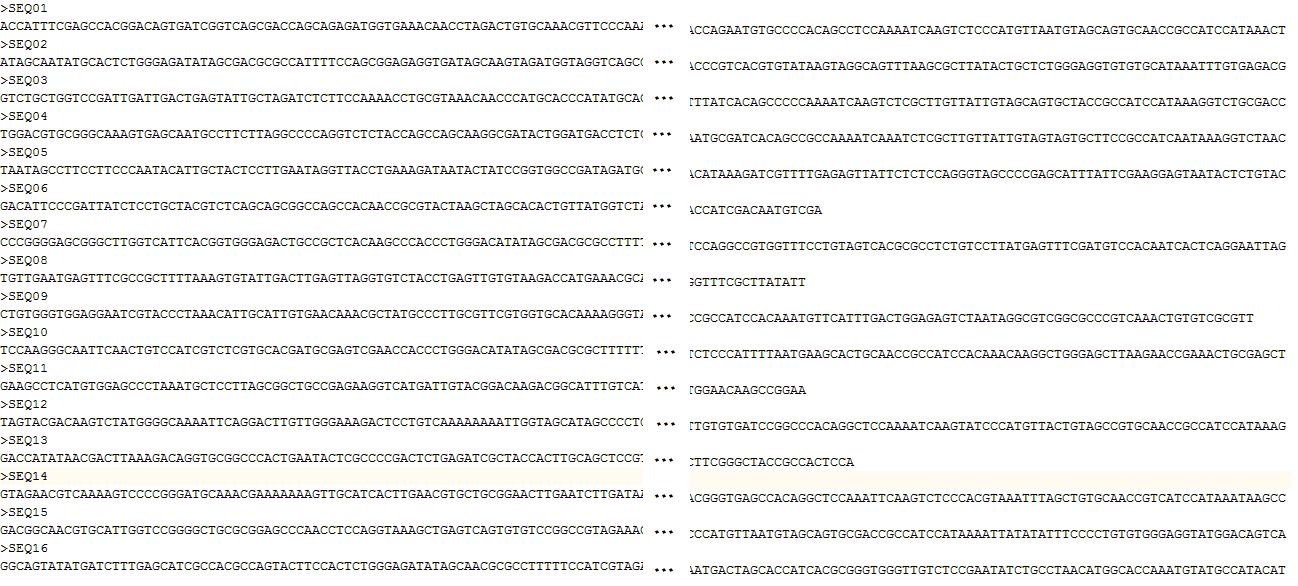
### Protocole des expérimentations

Après plusieurs simulations, le paramètre qui présente la taille du bloc a été fixée à pour aligner les protéines, à pour aligner les *ADN* et à pour aligner les *ARN* (voir le Tableau ‎4‑8). En outre, *BicMSA* prend en entré un fichier de type *FASTA* comportant les séquences à aligner et il génère en sortie un autre fichier *FASTA* contenant les alignements entre ces séquences.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BASE de jeux** | **Type de données** | **Nombre de familles testées** | **Nombre de séquences testées** | **Taille du bloc** |
| BALIBASE | Protéines | 84 | 1381 | **4** |
| BRALIBASE | ARN | 192 | 1468 | **7** |
| DIRMBASE | ADN | 64 | 1280 | **11** |

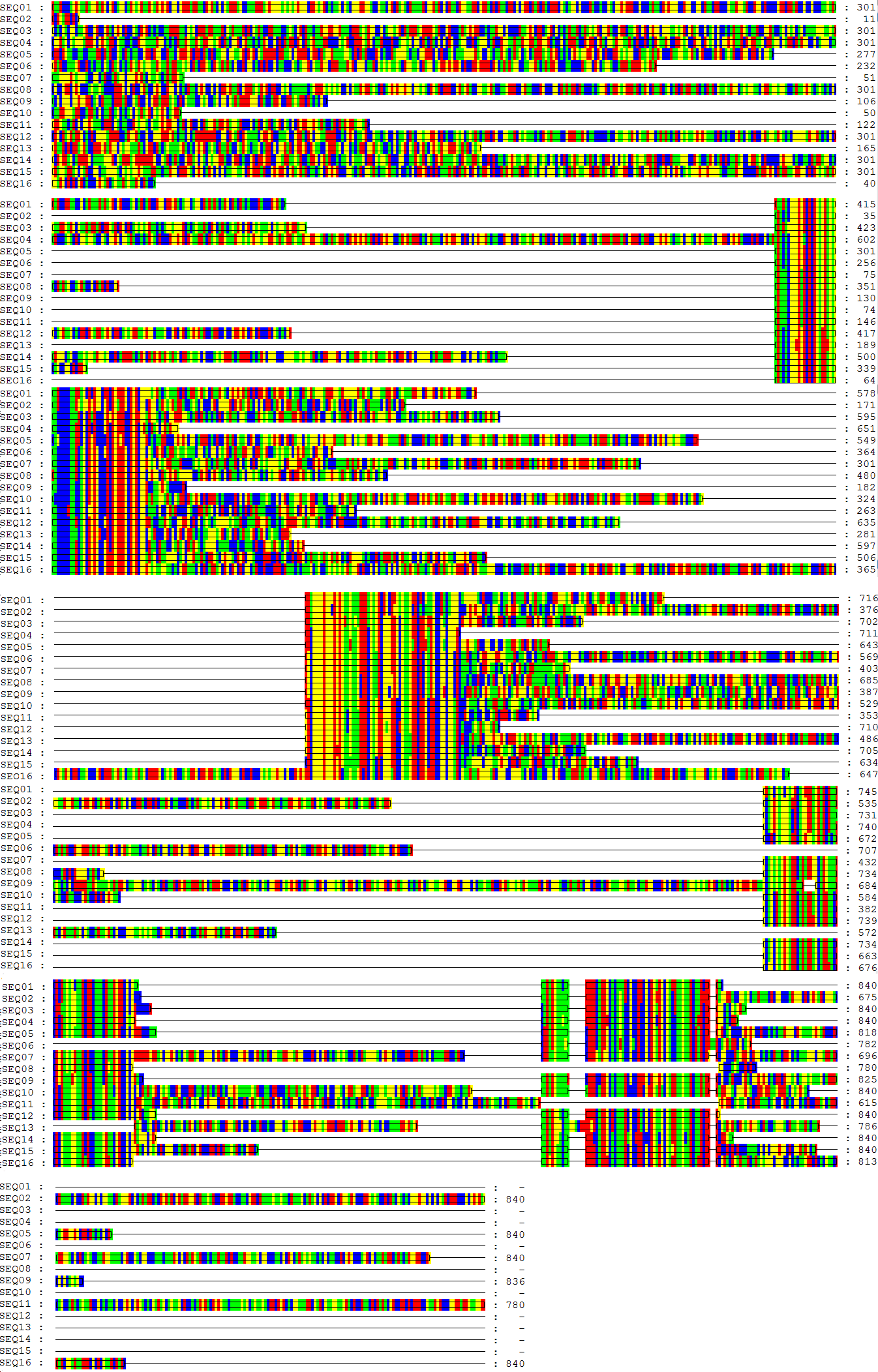
Tableau ‎4‑8 Protocole des expérimentations

Par ailleurs, le format *FASTA* se présente sous la forme d’un fichier texte disposant une extension *.fasta*, *.fa* ou *.aa*. Il est issu de la suite de programmes FASTA [Lipman et Pearson, 1985] Sa simplicité rend le stockage et la lecture des séquences biologiques de nature nucléique ou protéique plus simple à réaliser. Il présente un ensemble de séquence par une suite de caractère appartenant à l’alphabet , et . Chaque séquence se procède par un nom et des commentaires. Ce format présente un standard dans la bioinformatique.

La Figure 4-1 montre un ensemble d’*ADN*s récupérés à partir l’ensemble *∑F3* de la référence 4 dans *DIRMBASE* (voir Tableau ‎4‑4). En effet, c’est la grande famille dans *DIRMBASE*. Elle comporte séquences (avec ).

**Figure ‎4‑1 Fichier FASTA des séquences à aligner**

La Figure 4-2 montre un alignement de cette famille généré avec *BicMSA.* Cette figure présente les *Adénines* en couleur rouge, les *Cytosines* en vert, les *Guanines* en jaune et les nucléotides *Uraciles* en bleu. Les chiffres à droite présentent la position de l’alignement et les noms à gauche présentent les noms de séquences.3 *BicMSA* fournit un alignement disposant une taille égale à .



**Figure ‎4‑2 Alignement des ADNs avec BicMSA**

### Résultats expérimentaux basés sur l’évaluation de protéines

La base *BALIBASE* [Thompson et al., 1999] se compose de références chacune d’entre elles présente un problème d’alignement différent. De ce fait, nous allons faire une étude expérimentale approfondie en générant les résultats des scores SPS pour chaque référence. En effet, pour valider biologiquement *BicMSA*, nous nous focalisons à donner quelques familles testées dans chaque référence en élaborant une comparaison avec les algorithmes les plus célèbres.

* Référence 1 : En utilisant la *référence* , le Tableau ‎4‑9 montre la capacité de *BicMSA* à produire les meilleurs alignements avec des familles de protéine disposant un pourcentage de similarité très faible par rapport à d’autres algorithmes. Citant par exemple, les familles des *ubiquitins* (*ubi*), *BicMSA* réalise un alignement proche de de l’optimal. Il est meilleur que *CLUSTAL* (), *DIALIGN*() et *HMMT* (). En outre, pour la protéine humaine *Myb proto-oncogène* (*idy*), *BicMSA* produit le meilleur alignement () par rapport à *DIALIGN*(), *ML\_PIMA*(), *PILEUP8*() et *MULTAL* ()(voir Figure ‎4‑3).

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **MULTAL** | **HMMT** | **BicMSA** |
| 1idy | 0,606 | 0,705 | 0,342 | 0,018 | 0,145 | 0,062 | 0,566 | 0,080 | 0,080 | 0,138 | 0,108 |
| 1tvxA | 0,378 | 0,438 | 0,278 | 0,306 | 0,344 | 0,344 | 0,228 | 0,344 | 0,244 | 0,108 | 0,114 |
| 1ubi | 0,498 | 0,415 | 0,452 | 0,000 | 0,370 | 0,493 | 0,488 | 0,428 | 0,428 | 0,140 | 0,421 |
| 1havA | 0,656 | 0,446 | 0,411 | 0,130 | 0,300 | 0,466 | 0,419 | 0,536 | 0,040 | 0,133 | 0,318 |
| 1aab | 1,000 | 1,000 | 0,823 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 0,214 | 0,429 |
| 1fjlA | 1,000 | 1,000 | 0,993 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 0,436 | 0,516 |
| 1pfc | 0,975 | 0,988 | 0,994 | 0,894 | 0,927 | 0,927 | 0,975 | 0,964 | 0,861 | 0,350 | 0,495 |
| 1ycc | 0,856 | 0,935 | 0,837 | 0,749 | 0,815 | 0,815 | 0,932 | 0,939 | 0,940 | 0,217 | 0,329 |
| 1csp | 0,943 | 0,993 | 0,993 | 0,980 | 1,000 | 1,000 | 0,987 | 1,000 | 0,625 | 0,776 | 0,671 |
| 1fkj | 0,982 | 0,981 | 0,981 | 0,958 | 0,944 | 0,987 | 0,951 | 0,913 | 0,609 | 0,879 | 0,662 |
| 1fmb | 0,959 | 0,981 | 0,979 | 0,959 | 0,952 | 0,952 | 0,995 | 0,995 | 0,556 | 0,863 | 0,681 |
| glg | 0,982 | 0,941 | 0,972 | 0,959 | 0,978 | 0,968 | 0,954 | 0,977 | 0,943 | 0,689 | 0,917 |
| Tableau ‎4‑9 BALIBASE : Comparaison  des résultats de la référence 1 | | | | | | | | | | | |

|  |  |
| --- | --- |
|  |  |
| Figure ‎4‑3 BALIBASE : Comparaison des résultats de la référence 1 | |

* Référence 2 : Cette référence est caractérisée par la présence des familles hautement bruitées comportant des séquences de tailles très différentes. Ses séquences ne disposent d’aucunes similarités entre elles, mais elles sont alignées avec une séquence *orpheline*. Par ailleurs, le Tableau ‎4‑10 montre que *BicMSA* donne les meilleurs alignements par rapport à la majorité des algorithmes. Par exemple, *BicMSA* génère encore le meilleur alignement () avec les familles des *ubiquitines* (*ubi*) par rapport à *PRRP*(), *DIALIGN*(), *HMMT*(5,3), *SB\_PIMA*(), *ML\_PIMA*(), *MULTALIGN*() et *PILEUP8*(). En outre, pour la protéine  *Homologie Src 2 Domaine* (*y*), *BicMSA* dispose le meilleur score d’alignement () par rapport à *CLUSTALX*(), *SAGA*(), *DIALIGN*(), *HMMT*(0), *SB\_PIMA*(), *ML\_PIMA*(), *MULTALIGN*() et *PILEUP8*(11,4). La Figure ‎4‑4 montre que *BicMSA* n’est pas sensible aux familles hautement bruitées par rapport aux autres algorithmes.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **HMMT** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **BicMSA** |
| 1aboA | 0,256 | 0,650 | 0,489 | 0,384 | 0,724 | 0,391 | 0,220 | 0,528 | 0,000 | 0,255 |
| 1idy | 0,370 | 0,515 | 0,548 | 0,000 | 0,353 | 0,000 | 0,000 | 0,401 | 0,000 | 0,483 |
| 1csy | 0,350 | 0,154 | 0,154 | 0,000 | 0,000 | 0,000 | 0,000 | 0,154 | 0,114 | 0,197 |
| 1ubi | 0,056 | 0,482 | 0,492 | 0,000 | 0,053 | 0,129 | 0,129 | 0,000 | 0,000 | 0,352 |
| 1uky | 0,351 | 0,656 | 0,476 | 0,216 | 0,395 | 0,256 | 0,306 | 0,585 | 0,562 | 0,425 |
| 2hsdA | 0,404 | 0,484 | 0,498 | 0,262 | 0,423 | 0,390 | 0,561 | 0,593 | 0,278 | 0,265 |
| 4enl | 0,668 | 0,375 | 0,739 | 0,122 | 0,213 | 0,096 | 0,092 | 0,384 | 0,224 | 0,319 |
| **Tableau ‎4‑10 BALIBASE : Comparaison  des résultats de la référence 2** | | | | | | | | | | |

|  |  |
| --- | --- |
|  |  |
| Figure ‎4‑4 BALIBASE : Comparaison des résultats de la référence 2 | |

* Référence 3 : Elle est constituée des sous-groupes avec au moins d’identité. Les alignements sont construits en ajoutant des familles homologues aux séquences les plus éloignées de la *référence\_*. En utilisant cette référence, *BicMSA*génère des scores d’alignement proche aux alignements de référence plus que la majorité des algorithmes sélectionnés avec les familles citées dans le Tableau ‎4‑11. Prenant comme exemple la famille des *twitchins* (*wit*), *BicMSA* génère le meilleur SPS par rapport à tous les algorithmes donnés dans le Tableau ‎4‑11. En outre, avec la famille de *kinase uridylate* (*uky*) le score de *BicMSA* () est meilleur que celui de *PRRP*(), *CLUSTALX*(), *DIALIGN*(), *HMMT*(), *SB\_PIMA*(), *ML\_PIMA*(14,8), *MULTALIGN*(2) et *PILEUP8*(8,3). En effet, la Figure ‎4‑5 montre la performance de *BicMSA* par rapport aux autres algorithmes.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **HMMT** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **BicMSA** |
| 1idy | 0,000 | 0,273 | 0,364 | 0,000 | 0,227 | 0,000 | 0,000 | 0,045 | 0,000 | 0,238 |
| 1r69 | 0,905 | 0,524 | 0,524 | 0,524 | 0,000 | 0,000 | 0,905 | 0,000 | 0,000 | 0,109 |
| 1ubi | 0,415 | 0,146 | 0,585 | 0,000 | 0,366 | 0,000 | 0,000 | 0,000 | 0,268 | 0,225 |
| 1wit | 0,742 | 0,565 | 0,484 | 0,500 | 0,323 | 0,645 | 0,323 | 0,242 | 0,210 | 0,867 |
| 1uky | 0,139 | 0,130 | 0,269 | 0,139 | 0,037 | 0,083 | 0,148 | 0,241 | 0,083 | 0,228 |
| 1ajsA | 0,128 | 0,163 | 0,186 | 0,000 | 0,006 | 0,000 | 0,000 | 0,000 | 0,110 | 0,153 |
| 1 pamA | 0,683 | 0,678 | 0,579 | 0,683 | 0,169 | 0,546 | 0,590 | 0,546 | 0,754 | 0,243 |
| 2myr | 0,646 | 0,538 | 0,494 | 0,272 | 0,101 | 0,278 | 0,494 | 0,253 | 0,310 | 0,143 |
| 4enl | 0,736 | 0,547 | 0,672 | 0,050 | 0,050 | 0,393 | 0,438 | 0,652 | 0,498 | 0,403 |
| **Tableau ‎4‑11 BALIBASE : Comparaison  des résultats de la référence 3** | | | | | | | | | | |

|  |  |
| --- | --- |
|  |  |
| Figure ‎4‑5 BALIBASE : Comparaison des résultats de la référence 3 | |

* Référence 4 : Elle est divisée en deux sous-catégories contenant des alignements de séquences allant jusqu'à 20, y compris des extensions terminaux (jusqu'à 400 résidus). Par ailleurs, en récupérant quelques familles testées dans le Tableau ‎4‑12, nous remarquons que la majorité des scores fournis avec les autres algorithmes sont égales à . Citant comme exemple, la famille de *protéine majeure de choc froid* (*csp*) est alignée avec un score de en utilisant *BicMSA*. Par contre, en utilisant les algorithmes PRRP, CLUSTALX, SAGA, *SB\_PIMA*, *ML\_PIMA*, *MULTALIGN,* et *PILEUP8*, cette protéine est alignée avec un score SPS égal à 0%. De ce fait, BicMSA donne de bons résultats par rapport aux CLUSTALX SAGA, MULTALIGN dans l’alignement des séquences avec plusieurs extensions terminales comme la famille de *glycyl-tRNA synthetase*() et *kinase* (voir Figure ‎4‑6).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **BicMSA** |
| 1dynA | 0,000 | 0,000 | 0,000 | 0,600 | 0,600 | 0,600 | 0,000 | 0,000 | 0,060 |
| 1pysA | 0,000 | 0,000 | 0,250 | 0,750 | 1,000 | 1,000 | 0,000 | 0,750 | 0,348 |
| 1ckaA | 1,000 | 0,000 | 0,375 | 1,000 | 1,000 | 0,000 | 0,000 | 1,000 | 0,220 |
| 1csp | 0,000 | 0,000 | 0,000 | 0,889 | 0,000 | 0,000 | 0,000 | 0,000 | 0,329 |
| 1lkl | 1,000 | 1,000 | 0,000 | 1,000 | 1,000 | 1,000 | 0,000 | 1,000 | 0,109 |
| 1vln | 0,000 | 0,879 | 0,606 | 0,545 | 0,636 | 0,576 | 0,636 | 0,848 | 0,284 |
| 2abk | 0,000 | 0,000 | 0,000 | 1,000 | 0,471 | 0,471 | 0,000 | 0,471 | 0,324 |
| kinase1 | 0,000 | 0,000 | 0,000 | 1,000 | 1,000 | 1,000 | 0,000 | 1,000 | 0,174 |
| kinase2 | 0,000 | 0,000 | 0,364 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 0,067 |
| **Tableau ‎4‑12 BALIBASE : Comparaison  des résultats de la référence 4** | | | | | | | | | |

|  |  |
| --- | --- |
|  |  |
| Figure ‎4‑6 BALIBASE : Comparaison des résultats de la référence 4 | |

* Référence 5 : Cette référence est caractérisée par des alignements avec un taux d’insertion des brèches très élevé. En utilisant ces familles, le Tableau ‎4‑13 montre que *BicMSA* fourni toujours le meilleur alignement par rapport à *SB\_PIMA*. En outre pour la famille de *eftu* (), *BicMSA* dispose un score égal à tandis *CLUSTALX*, SAGA, *SB\_PIMA*, *ML\_PIMA* et *MULTALIGN* disposent des score SPS nuls ().

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **BicMSA** |
| 1pysA | 0,429 | 0,429 | 0,429 | 0,762 | 0,190 | 0,762 | 0,429 | 0,190 | 0,249 |
| 1eft | 0,211 | 0,000 | 0,000 | 0,579 | 0,000 | 0,000 | 0,000 | 0,211 | 0,090 |
| 1thm2 | 0,645 | 0,774 | 0,774 | 1,000 | 0,194 | 0,194 | 0,774 | 0,645 | 0,428 |
| **Tableau ‎4‑13 BALIBASE : Comparaison  des résultats de la référence 5** | | | | | | | | | |

|  |  |
| --- | --- |
|  |  |
| Figure ‎4‑7 BALIBASE : Comparaison des résultats de la référence 5 | |

* Récapitulatif des résultats avec BALIBASE : En présentant brièvement les résultats des alignements et les comparaisons avec les algorithmes les plus connus dans la littérature, nous avons bien remarqué que (voir Tableau ‎4‑14) :

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **PRRP** | **CLUSTALX** | **SAGA** | **DIALIGN** | **SB\_PIMA** | **ML\_PIMA** | **MULTALIGN** | **PILEUP8** | **MULTAL** | **HMMT** | **BicMSA** |
|  | **Moyenne** | 0,820 | 0,819 | 0,755 | 0,663 | 0,731 | 0,751 | 0,791 | 0,765 | 0,611 | 0,412 | 0,472 |
| **Max** | 1,000 | 1,000 | 0,994 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | 0,879 | 0,917 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **Moyenne** | 0,351 | 0,474 | 0,485 | 0,141 | 0,180 | 0,187 | 0,378 | 0,168 | - | 0,309 | 0,328 |
| **Max** | 0,668 | 0,656 | 0,739 | 0,384 | 0,391 | 0,561 | 0,593 | 0,562 | - | 0,724 | 0,483 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **Moyenne** | 0,488 | 0,396 | 0,462 | 0,241 | 0,216 | 0,322 | 0,220 | 0,248 | - | 0,142 | 0,290 |
| **Max** | 0,905 | 0,678 | 0,672 | 0,683 | 0,645 | 0,905 | 0,652 | 0,754 | - | 0,366 | 0,867 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **Moyenne** | 0,222 | 0,209 | 0,177 | 0,865 | 0,745 | 0,627 | 0,182 | 0,674 | - | - | 0,213 |
| **Max** | 1,000 | 1,000 | 0,606 | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | - | - | 0,348 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **Moyenne** | 0,428 | 0,401 | 0,401 | 0,780 | 0,128 | 0,319 | 0,401 | 0,349 | - | - | 0,256 |
| **Max** | 0,645 | 0,774 | 0,774 | 1,000 | 0,194 | 0,762 | 0,774 | 0,645 | - | - | 0,428 |
|  | Tableau ‎4‑14 BALIBASE : Récapitulatif des résultats | | | | | | | | | | | |

* En utilisant des petites familles disposant des séquences de même taille avec un pourcentage d’identité acceptable, *BicMSA* fourni les meilleurs alignements par rapport à *HMMT*.
* En utilisant des familles composées par un ensemble de séquences avec une taille très différente et un pourcentage d’identité très faible, *BicMSA* fournit des scores SPS supérieurs à *DIALIGN*, *HMMT*, *SB\_PIMA*, *ML\_PIMA*, *MULTALIGN* et *PILEUP8*. Ainsi, en appliquant le *Raffinement*, *BicMSA* résout le problème de la présence des brèches de grande taille par rapport à ces algorithmes (voir Figure ‎4‑8).
* En utilisant des grandes familles (par rapport à *BALIBASE*) disposant des séquences de tailles très différentes et comportant des extensions faibles*, BicMSA* génère les meilleurs alignements par rapport aux algorithmes les plus connus comm*e CLUSTALX, SAGA, DIALIGN, HMMT, SB\_PIMA, MULTALIGN* e*t PILEUP8*.
* Avec la présence des extensions fortes dans des grandes familles dans *BALIBASE* comportant des séquences de tailles très différentes, *BicMSA* donne les meilleurs scores d’alignement par rapport à *CLUSTALX*, *SAGA* et *MULTALIGN*.
* Avec la présence des *N-insertions* dans des grandes familles disposant d’un pourcentage d’identité variable, *BicMSA* génère les meilleurs alignements par rapport à *SB\_PIMA*.

Par ailleurs, puisqu’il utilise l’étape de *raffinement* dans son processus d’alignement, *BicMSA* ne donne plus des scores nuls (Tableau ‎4‑10, Tableau ‎4‑11, Tableau ‎4‑12, Tableau ‎4‑13) comme *DIALIGN*, *CLUSTAL,…*etc. Dans la section suivante, nous allons étudier la performance de *BicMSA* avec les *ARNs*.

**Figure ‎4‑8 BALIBASE : Comparaison des résultats avec des alignements à grande brèche**

### Résultats expérimentaux basés sur l’évaluation de l’ARN

Pour valider la qualité de l’alignement d’*ARN* fournis par *BicMSA*, nous avons utilisé la base *BRALIBASE*\_ [Gardner et Giegerich, 2004] spécifiquement les familles les plus utilisées dans la validation biologique (voir Tableau ‎4‑3). Ces familles sont caractérisées par le nombre, la taille et le pourcentage d’identité de séquences. Ainsi, pour prouver la capacité de *BicMSA*, nous proposons une décomposition des résultats fournis suivant le nombre de séquences : *petites familles*, *familles moyennes* et *grandes familles*.

* *Les petites familles* : Ces petites familles sont caractérisées par leur forte similarité. En effet, malgré qu’elles disposent un nombre de séquences très petit (), ces familles forment une piège pour les algorithmes d’alignement puisque certaines zones sont répétées plusieurs fois. Comme le montre le Tableau ‎4‑15, *BicMSA* génère les meilleures moyennes des scores *SPS* par rapport à *CLUSTALW*, *PCMA* et *muscle* avec la majorité des familles comme les *THI*, *SECIS*, *yybP-ykoY* et *T-RNA* (voir Figure ‎4‑9). En outre, il a achevé des alignements avec un pourcentage de par rapport aux alignements de références (voir Tableau ‎4‑16). Citant par exemple le groupe de famille T-box, *BicMSA* fourni un pourcentage de tandis que *MAFFT*, *PCMA*, *CLUSTALW* et *muscle* fournissent respectivement , , et (voir Tableau ‎4‑16). Nous constatons alors que *BicMSA* permet d’extraire les similarités entre les séquences d’une manière très proche de l’optimal. Ainsi, il fournit des scores meilleurs que les autres algorithmes. En conséquence, *BicMSA* résout parfaitement le problème des répétitions de zones par rapport à *PCMA* et *CLUSTALW (voir* Figure ‎4‑10*).*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | SECIS | 0,705 | 0,665 | 0,665 | 0,705 | 0,684 | | T-box | 0,786 | 0,742 | 0,738 | 0,810 | 0,786 | | THI | 0,775 | 0,630 | 0,621 | 0,713 | 0,722 | | T-RNA | 0,983 | 0,983 | 0,977 | 0,983 | 0,744 | | yybP-ykoY | 0,626 | 0,554 | 0,574 | 0,636 | 0,591 | | Entero\_OriR | 0,958 | 0,974 | 0,974 | 0,973 | 0,954 | | gcvT | 0,570 | 0,437 | 0,428 | 0,492 | 0,445 |   Tableau ‎4‑15 BRALIBASE : Moyenne des SPS avec les petites familles | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | SECIS | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | | T-box | 0,970 | 0,970 | 0,970 | 0,990 | 1,000 | | THI | 1,000 | 0,970 | 0,970 | 0,970 | 1,000 | | T-RNA | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | | yybP-ykoY | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | | Entero\_OriR | 1,000 | 1,000 | 1,000 | 1,000 | 1,000 | | gcvT | 0,930 | 0,930 | 0,930 | 0,930 | 0,883 |   Tableau ‎4‑16 BRALIBASE : Max des SPS avec les petites familles |
|  |  |
| Figure ‎4‑9 BRALIBASE : Résultats des SPS avec les petites familles | |

* *Les familles moyennes* : Ces familles contiennent au maximum séquences biologiques. En prenant les familles de SECIS (*la structure d'insertion de la sélénocystéine*), *CLUSTALW*, *PCMA* et *muscle* génèrent leurs SPS qui ne dépassent pas tandis que *BicMSA* aligne ces groupes avec un pourcentage égal à par rapport à l’alignement de référence. Par ailleurs, les valeurs maximales atteintes présentées dans le Tableau ‎4‑18prouvent la qualité des résultats fournis par *BicMSA*. En outre, comme le montre la Figure ‎4‑10, *BicMSA* génère les résultats les plus significatifs avec les familles de *SECIS* et *THI*. En se basant sur ces statistiques, *BicMSA* prouve sa performance par rapport aux autres algorithmes spécifiquement avec les alignements des structures tridimensionnelles comme le *SECIS*.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | SECIS | 0,610 | 0,340 | 0,390 | 0,470 | 0,671 | | S-box | 0,830 | 0,810 | 0,810 | 0,830 | 0,726 | | THI | 0,900 | 0,620 | 0,680 | 0,890 | 0,930 | | T-RNA | 0,970 | 0,970 | 0,970 | 0,970 | 0,879 | | yybP-ykoY | 0,560 | 0,570 | 0,570 | 0,530 | 0,463 | | Entero\_OriR | 0,990 | 1,000 | 1,000 | 0,990 | 0,955 | | gcvT | 0,780 | 0,750 | 0,750 | 0,770 | 0,712 | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | SECIS | 0,517 | 0,223 | 0,323 | 0,367 | 0,504 | | S-box | 0,774 | 0,770 | 0,770 | 0,788 | 0,622 | | THI | 0,789 | 0,532 | 0,529 | 0,789 | 0,625 | | T-RNA | 0,930 | 0,790 | 0,890 | 0,940 | 0,704 | | yybP-ykoY | 0,445 | 0,495 | 0,473 | 0,440 | 0,413 | | Entero\_OriR | 0,933 | 0,964 | 0,964 | 0,954 | 0,881 | | gcvT | 0,645 | 0,466 | 0,469 | 0,591 | 0,430 | |
| **Tableau ‎4‑17 BRALIBASE : Moy SPS avec les familles moyennes** | **Tableau ‎4‑18 BRALIBASE : Max des SPS avec les familles moyennes** |
|  |  |
| Figure ‎4‑10 BRALIBASE : Résultats des SPS avec les familles moyennes | |

* *Les grandes familles* : Le nombre de séquences dans ces familles est égal soit à 10 ou bien à 15. Ainsi, le pourcentage d’identité entre les séquences est légèrement acceptable. En effet, pour les familles de grandes tailles dans *BRALIBASE*, *BicMSA* fournit encore les meilleurs alignements par rapport aux autres algorithmes comme *CLUSTALW*, *MAFFT* et *muscle (voir* Figure ‎4‑11et Figure ‎4‑12*).* Citant par exemple la famille des *t-RNA*, le Tableau ‎4‑19 et Tableau ‎4‑20 montrent que *BicMSA* génère le meilleur score SPS () par rapport à *CLUSTALW*(), *MAFFT*(), PCMA () et *muscle*().

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | **n=10** | THI | 0,760 | 0,570 | 0,590 | 0,740 | 0,575 | | T-RNA | 0,950 | 0,800 | 0,830 | 0,950 | 0,559 | | yybP-ykoY | 0,600 | 0,510 | 0,490 | 0,590 | 0,522 | | **n=15** | THI | 0,580 | 0,810 | 0,557 | 0,593 | 0,783 | | T-RNA | 0,413 | 0,836 | 0,620 | 0,706 | 0,858 | | yybP-ykoY | 0,330 | 0,600 | 0,460 | 0,408 | 0,512 | | Entero\_OriR | 0,896 | 0,955 | 0,980 | 0,980 | 0,968 |   **Tableau ‎4‑19 BRALIBASE : Moy SPS avec les grandes familles** | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | **n=10** | THI | 0,686 | 0,518 | 0,530 | 0,701 | 0,465 | | T-RNA | 0,888 | 0,655 | 0,720 | 0,875 | 0,528 | | yybP-ykoY | 0,600 | 0,510 | 0,490 | 0,590 | 0,522 | | **n=15** | THI | 0,646 | 0,850 | 0,590 | 0,660 | 0,810 | | T-RNA | 0,447 | 0,920 | 0,670 | 0,880 | 0,930 | | yybP-ykoY | 0,352 | 0,660 | 0,530 | 0,540 | 0,620 | | Entero\_OriR | 0,924 | 0,970 | 0,990 | 0,990 | 0,980 |   **Tableau ‎4‑20 BRALIBASE : Max SPS avec les grandes familles** |
|  |  |
| Figure ‎4‑11 BRALIBASE : Résultats des SPS avec les grandes familles (*n*=10) | |
|  |  |
| Figure ‎4‑12 BRALIBASE : Résultats des SPS avec les grandes familles (*n*=15) | |

* Comparaison avec BlockMSA

*Wang* et *al* [Wang et al., 2007] utilisent dans la validation du *BlockMSA* des familles extraites à partir de différentes base des *ARNs*. En utilisant la base *BRALIBASE*, ils ne fournissent que les scores des familles de *THI*, *Glycine riboswitch* et *Yybp-Ykoy*. De ce fait, nous avons récupéré ces familles afin de générer les moyennes de nos scores SPS. En utilisant les résultats de *BlockMSA* présentés dans l’article de *Wang et al* [Wang et al., 2007] (voir Tableau ‎4‑21), nous remarquons qu’avec les petites familles disposant un grand nombre de zones de répétitions, *BicMSA* donne des meilleurs résultats () par rapport à *BlockMSA*. Ainsi, en utilisant les grandes familles, *BicMSA* fournis des scores moyens et proches à ceux de *BlockMSA* (voir Figure ‎4‑13).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | **SPS** | | | |  | **n=7** | **n=10** | **n=15** | | BlockMSA | 0.670 | 0,676 | 0,748 | | BicMSA | 0,787 | 0,469 | 0,392 |   Tableau ‎4‑21 BRALIBASE : Comparaison du BicMSA avec BlockMSA |  |
|  | Figure ‎4‑13 BRALIBASE : Comparaison du BlockMSA avec BicMSA |

* Récapitulatif des résultats avec BRALIBASE : En se basant sur ces statistiques, nous remarquons que *BicMSA* donne le meilleur pourcentage des scores par rapport à la majorité des algorithmes comme *PCMA* et *CLUSTALW*. En effet, ces algorithmes se trouvent incapable à aligner les familles de *SECIS*, *THI* et *t-RNA* par rapport à *BicMSA* (voir Figure ‎4‑14). Ils fournissent des moyennes de scores SPS inférieures à celle de *BicMSA*. En outre, avec les familles de grandes tailles par rapport à la base choisie, notre algorithme aligne les séquences de la manière la plus proche de l’optimale. Ainsi, nous constatons que si le nombre de séquences augmente, le meilleur score est trouvé avec *BicMSA* (voir Tableau ‎4‑22). En outre, en récupérant les *SPS\_max* atteints, nous notons que *BicMSA* réalise des alignements optimaux par rapport aux autres algorithmes. Afin de continuer à mesurer la performance de BicMSA dans l’alignement des séquences biologiques, nous présentons, dans la section suivante, une étude expérimentale sur les séquences d’*ADN*.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **MAFFT** | **PCMA** | **CLUSTALW** | **muscle** | **BicMSA** | | **n=2** | 0,731 | 0,640 | 0,640 | 0,702 | 0,671 | | **n=7** | 0,733 | 0,633 | 0,642 | 0,719 | 0,613 | | **n=10** | 0,663 | 0,516 | 0,524 | 0,660 | 0,432 | | **n=15** | 0,532 | 0,790 | 0,646 | 0,663 | 0,769 | |  |
| Tableau ‎4‑22 BRALIBASE : Comparaison avec la moyenne des SPS de toutes les familles | **Figure ‎4‑14 BRALIBASE : Comparaison avec la moyenne des SPS de toutes les familles** |

### Résultats expérimentaux basés sur l’évaluation de l’ADN

La base *DIRMBASE* aide à valider la performance des algorithmes d’alignement d’un point de vue taille de séquences. Elle possède des familles d’*ADN* contenant un nombre suffisant de séquences (ne dépasse pas 16) avec des tailles énormes (entre 400 et 600 nucléotides). En appliquant *BicMSA*, nous citons dans le Tableau ‎4‑23 les moyennes de scores *SPS* et dans le Tableau ‎4‑24, les moyennes de scores CS. En effet, comme le montre la Figure ‎4‑15 et la Figure ‎4‑16, *BicMSA* génère avec les petites (*ref\_*) et les moyennes (*ref\_*) séquences des score SPS meilleurs que *CLUSTALW2*, *T-COFFEE5.56* et *ProBCONSRNA1.10* ainsi des scores CS meilleurs que *DIALIGN-T0.2.2*, *CLUSTALW2*, T*-COFFEE 5.56*, *POA V2*, *MAFFT 6.240 L-INSI*, *Muscle3.7* et *PROBCONSRNA1.10*. En outre, pour les séquences de grandes tailles (), *BicMSA* dispose des meilleurs alignements proches de l’optimal que *CLUSTALW2*, *T-COFFEE5.56*, *POAV2*, *ProBCONSRNA1.10*, *MAFFT6.240L-INSI* et *Muscle3.7*. Par ailleurs, en calculant la moyenne totale des scores obtenus, nous remarquons alors que *BicMSA* produit la bonne qualité d’alignement (SPS ;CS) par rapport à *DIALIGN-T0.2.2* (CS), *CLUSTALW2* (SPS;CS ), *T-COFFEE5.56* (SPS ; CS ), *POAV2* (SPS; CS=), *MAFFT6.240L-INSI* (SPS), *Muscle3.7* (CS) et *PROBCONSRNA1.10* (SPS ; CS). Pour ce là, nous constatons que *BicMSA* est plus performant avec les séquences de grande taille que les algorithmes comparés, spécifiquement *DIALIGN-T-0.2.2* qui est une référence dans l’alignement des ADNs (voir Figure ‎4‑17 et Figure ‎4‑18).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **REF1** | **REF2** | **REF3** | **REF4** | **Totale** | | DIALIGN-TX | 94,38 | 92,85 | 95,44 | 95,70 | 94,59 | | DIALIGN-T 0,2,2 | 64,00 | 61,22 | 64,96 | 65,24 | 63,85 | | DIALIGN 2,2 | 92,61 | 91,10 | 94,62 | 94,13 | 93,12 | | CLUSTAL W2 | 6,79 | 8,27 | 18,51 | 29,09 | 15,66 | | T-COFFEE 5,56 | 14,71 | 18,88 | 32,08 | 43,39 | 27,62 | | POA V2 | 32,03 | 27,40 | 28,78 | 32,18 | 30,10 | | MAFFT 6,240 L-INSi | 52,40 | 48,81 | 49,77 | 57,47 | 52,36 | | MAFFT 6,240 E-INSi | 92,42 | 84,15 | 87,91 | 89,36 | 88,46 | | MUSCLE 3,7 | 48,17 | 54,40 | 56,57 | 60,24 | 56,84 | | PROBCONSRNA 1,10 | 13,00 | 12,94 | 20,28 | 32,56 | 19,69 | | BicMSA | 47,20 | 47,68 | 42,37 | 43,61 | 45,22 | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **REF1** | **REF2** | **REF3** | **REF4** | **Totale** | | DIALIGN-TX | 74,39 | 69,03 | 71,57 | 75,11 | 72,52 | | DIALIGN-T 0,2,2 | 29,60 | 28,63 | 35,51 | 35,85 | 32,40 | | DIALIGN 2,2 | 69,95 | 68,19 | 71,25 | 72,48 | 70,47 | | CLUSTAL W2 | 0,00 | 0,00 | 2,19 | 4,99 | 1,80 | | T-COFFEE 5,56 | 0,00 | 0,18 | 4,01 | 8,44 | 3,16 | | POA V2 | 5,63 | 7,32 | 4,12 | 6,81 | 5,97 | | MAFFT 6,240 L-INSi | 21,45 | 11,93 | 16,02 | 22,30 | 17,93 | | MAFFT 6,240 E-INSi | 40,28 | 41,99 | 45,77 | 51,01 | 44,76 | | MUSCLE 3,7 | 14,18 | 16,18 | 19,62 | 30,43 | 20,10 | | PROBCONSRNA 1,10 | 0,73 | 0,05 | 1,34 | 4,31 | 1,61 | | BicMSA | 32,78 | 38,30 | 30,53 | 32,22 | 33,46 | |
| Tableau ‎4‑23 DIRMBASE : Résultats des Moy\_SPS | Tableau ‎4‑24 DIRMBASE : Résultats des Moy CS |

Figure ‎4‑15 DIRMBASE : Résultats des SPS

Figure ‎4‑16 DIRMBASE : Résultats des CS

|  |  |
| --- | --- |
|  |  |
| **Figure ‎4‑17 DIRMBASE : Comparaison des algorithmes en suivant score SPS** | **Figure ‎4‑18 DIRMBASE : Comparaison des algorithmes en suivant score CS** |

## Temps d’exécution

*BicMSA* est implémenté en utilisant le langage java. Tous les autres programmes sont écrits en *C*. La machine utilisée dans la réalisation des tests est caractérisée par un processeur de type *Dual-Core CPU T4400@2.22Ghz* et une capacité de mémoire installée (*RAM*) égale à .

*BicMSA* aligne les séquences de la base *DIRMBASE* plus rapidement () que *T-COFFEE5.56* () (voir Tableau ‎4‑25 et Figure ‎4‑19) et les séquences de la base *BRALIBASE* plus rapidement que *BlockMSA* (voir Figure ‎4‑20 et Tableau ‎4‑26). Par ailleurs, la complexité en temps de calcul de *BicMSA* est fortement liée à la complexité en temps de calcul de *BiMine-modifié()* et de *Needleman et Wunch* [Needleman et Wunsch, 1970]. Actuellement, le biregroupement est encore un domaine de recherche actif. Pour ce là, la vitesse de *BicMSA* peut être également amélioré.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | **Moyenne Temps d’exécution** | | DIALIGN-TX 1.0 | 9.84s | | DIALIGN-T 0.2.2 | 2.31s | | DIALIGN 2.2 | 4.82s | | CLUSTAL W2 | 1.36s | | T-COFFEE 5.56 | 365.88s | | POA V2 | 1.20s | | MAFFT 6.240 L-INSi | 5.33s | | MAFFT 6.240 E-INSi | 8.39s | | MUSCLE 3.7 | 4.87s | | PROBCONS(RNA)(1.10) | 18.54s | | BicMSA | 40,47s | | |  |
| Tableau ‎4‑25 Temps d’exécution des algorithmes pour la base DIRMBASE. | | Figure ‎4‑19 Temps d’exécution des algorithmes pour la base DIRMBASE. |
| |  |  |  |  | | --- | --- | --- | --- | |  | **Temps en seconde** | | | | Program | n=7 | n=10 | n=15 | | BlockMSA | 0,507 | 0,653 | 0,855 | | BicMSA | 0,487 | 0,627 | 0,709 |   Tableau ‎4‑26 Temps d’exécution BlockMSA et BicMSA pour BRALIBASE |  | |
| Figure ‎4‑20 Temps d’exécution BlockMSA et BicMSA pour BRALIBASE | | |

## Conclusion

Dans ce chapitre, nous avons réalisé une étude expérimentale de notre algorithme d’alignement multiple des séquences biologiques. Nous avons effectué différents tests sur des bases de données réelles pour évaluer la performance de *BicMSA*. Pour ce faire, une étude comparative a été appliquée sur des séquences biologiques afin de mesurer la qualité des alignements générés par *BicMSA.* Dans la réalisation de ces comparaisons, nous avons utilisé les critères SPS et *CS*. Nous avons également utilisé des familles de séquences de grande taille disposant souvent d’une grande variété dans la longueur, le pourcentage d’identité et le contenu. En ce qui concerne les comparaisons des algorithmes de *MSA*, *BicMSA* génère des résultats comparables ou meilleurs que ceux obtenus par les algorithmes les plus connus dans la littérature. Ces résultats prouvent la performance et la qualité de *BicMSA* par rapport à certains algorithmes comme *CLUSTALW*, *DIALIGN*, *MAFFT*, *MUSCLE*. Par ailleurs, nous constatons que l’utilisation des techniques de biregroupement améliore la capacité de *BicMSA* dans l’identification des régions hautement ou faiblement conservées. Proprement dit, cette technique de fouille des données cherche à maximiser la cohérence totale entre les *blocs* respectivement les séquences et donc maximiser la découverte des homologies et des structures fonctionnelles et structurelles des séquences (*ADN*, *ARN* et *protéines*). Cependant, nos résultats de l'alignement dépendent de la performance de l’algorithme de biregroupement. En conséquence, *BicMSA* donne de bons résultats pour n’importe quel type de séquence (*ADN*, *protéines*…etc.).

# Conclusion Générale

Les progrès dans les projets de séquençage génomique ont conduit à une explosion des données biologiques. Ce qui a déclaré une nécessité impérieuse d’avoir des nouveaux outils performants et efficaces.

Ce mémoire s’articule autour de deux domaines : *l’alignement multiple des séquences biologiques* et le *biregroupement de données biologiques*. En effet, l'un des principaux défis de *MSA-local* est de trouver une technique performante qui permet de regrouper un ensemble de régions conservées dans des séquences biologiques et de fournir des résultats identiques à ceux que peut faire un biologiste. Un grand nombre de techniques ont été développées afin d’extraire les homologies entre les séquences [Henikoff et al., 1995 ; Karplus et al., 1998 ; Zhang et Kahveci, 2005 ; Subramanian et al., 2005 ; Pei et Grichin, 2007 ; Subramanian et al., 2008 ; Bradley et al., 2009 ; Cutello et al., 2011]. Parmi les techniques utilisées, nous citons le *biregroupement* [Cheng and Church, 2000; Madeira and Oliveira, 2004]. Il permet de dévoiler des groupes de sous-séquences (*blocs*) ayant un comportement similaires par rapport à des groupes spécifiques de séquences. En effet, l’utilisation du biregroupement est une piste qui n'a pas été bien explorée. Récemment, *Wang et al.* [Wang et al., 2007] ont proposé l'algorithme *BlockMSA* qui est une première adaptation du *biregroupement* pour la résolution des problèmes d’alignement multiples.

Dans ce cadre, nous avons développé un algorithme d’alignement multiple et local basé sur le *biregroupement*, appelé *BicMSA (****Bic****lustering-based local* ***MSA***). En effet, notre algorithme se présente parmi les premiers algorithmes adaptant le biregroupement pour un problème d’alignement. Puisque le problème de *MSA* est NP-complet [Carrillo et Lipman, 1988; Elias, 2006], l’objectif de *BicMSA* est de maximiser la similarité entre un ensemble de séquences et de trouver les alignements multiples les plus proches de ce que peut faire un biologiste. Le processus de *BicMSA* se compose de trois étapes principales : *Prétraitement, Traitement* et *Post-traitement*. En utilisant l’approche *DAC*, l’étape de *Prétraitement* consiste à produire, d’une manière optimale et globale, un ensemble d’alignements par paires entre les séquences afin de trouver le maximum des *blocs* disposant des cohérences maximales. Beaucoup de ces *blocs* sont liés les uns aux autres pour fournir des sous-séquences hautement conservées. Par la suite, l’étape de *Traitement* consiste à utiliser la technique de *biregroupement* dans la résolution de *MSA local*. De ce fait, *BicMSA* applique l’approche d’*énumération de bigroupes* afin de maximiser la cohérence totale entre les séquences. Quant à l’étape de *Post-traitement*, elle consiste à assembler les blocs générés dans l’étape de *Prétraitement* en utilisant les bigroupes cohérents donnés par l’étape de *Traitement*. Ainsi, une sous-étape de *raffinement* est appliquée afin d’améliorer la qualité de l’alignement dévoilé. Une fois ce processus est appliqué, *BicMSA* génère à la fois des groupes de sous-séquences similaires et un alignement multiple et local entre toutes les séquences biologiques. Par ailleurs, l’utilisation *MSA-local* donne une signification biologique aux résultats donnés par *BicMSA*. Pour prouver cette signification, nous avons effectué une étude expérimentale sur un ensemble de données réelles disposant souvent une grande variété dans la longueur, le pourcentage d’identité et le contenu.

Plus précisément, *BicMSA* a été testé sur des familles de protéines extraites de la base *BALIBASE* [Thompson et al., 1999], sur des familles d’ARN récupérées à partir de la base *BRALIBASE\_* [Gardner et Giegerich, 2004] et sur toutes les séquences d*’ADN* présentées dans la base *DIRMBASE* [Stoye et al., 1998]. L’objectif de cette étude expérimentale est d’évaluer la performance de *BicMSA* par rapport aux alignements de références et aux algorithmes les plus célèbres dans la littérature. D’abord, en utilisant la base *BALIBASE* (protéines), *BicMSA* fournit des moyennes de scores supérieurs à ceux obtenus avec *HMMT*, *MULTAL*, *PILEUP8*, *SB\_PIMA*, *ML\_PIMA* et *DIALIGN*. En outre, il génère des résultats comparables et proches de ceux réalisés par *PRRP*, *MULTALIGN*, *CLUSTALX* et *SAGA*. Par ailleurs, contrairement à *BicMSA*, plusieurs algorithmes ont généré, avec certaines familles, des scores SPS nuls comme *DIALIGN*, *CLUSTALX*, *SB\_PIMA*, *ML\_PIMA*, *MULTALIGN* et *MULTAL*. De ce fait, avec les protéines, *BicMSA* est plus performant par rapport à ces algorithmes. En outre, en regardant l’écart entre quelques résultats obtenus par *MULTALIGN* qui utilise la technique de regroupement dans la résolution de MSA et ceux de *BicMSA*, nous avons bien remarqué que l’utilisation de biregroupement est plus efficace pour l’extraction maximale des régions conservées. Ensuite, en utilisant la base *DIRMBASE* (ADN), *BicMSA* génère des alignements avec des scores *SPS* et *CS* proches de l’optimale que *PROBCONSRNA 1.10*, *MUSCLE 3.7*, *MAFFT6.240 L-INSi*, *POA V2*, *T-COFFEE 5.56*, *CLUSTALW* et *DIALIGN-TX*. Ainsi, il fournit des scores comparables à *DIALIGN-T 0.2.2*, *DIALIGN 2,2* et *MAFFT 6,240 E-INSi*. En appliquant les évaluations locales (avec le critère SPS) et les évaluations globales (avec le critère CS), nous constatons que *BicMSA* est efficace pour les alignements des séquences d’ADNs par rapport à certains algorithmes de références. Enfin, en utilisant la base *BRALIBASE\_* (*ARN*), *BicMSA* génère des alignements meilleurs que *MAFFT*, *CLUSTALW* et *Muscle*, spécifiquement avec les séquences de grande taille (), ainsi, il fournit des scores SPS comparables à ceux de *PCMA* et *BlockMSA*. En outre en utilisant des familles de petite taille (), *BicMSA* fourni des alignements proches et identique de l’optimal que *PCMA*, *CLUSTALW* et *BlockMSA*. Proprement dit, l’algorithme *BicMSA* peut être utilisé dans l’*alignement par paire* et il donne des résultats performants et efficaces que ces derniers algorithmes. Cependant, nous notons que *BicMSA* est plus rapide que *BlockMSA*. Par ailleurs*,* néanmoins que *BicMSA* est un algorithme local, il permetde dévoiler efficacement les alignements globaux présentés dans les bases de références *BALIBASE* et *BRALIBASE* meilleurs que certains algorithmes globaux comme *CLUSTALX*. Ainsi, il génère des alignements jusqu’à 100% proches de l’optimal avec certaines familles d’*ADN*, d’*ARN* et de protéines. En conséquence, *BicMSA* donne des bons résultats avec les *ADNs* et les *protéines*, ainsi, des résultats raisonnables avec les *ARN*s.

## Perspectives de recherches

Dans le but d’améliorer encore plus *BicMSA*, nous pensons qu’il est certainement intéressant de chercher à modifier la méthode de *génération de blocs (Prétraitement)*. De ce fait, au lieu de garder les *blocs* identiques, nous avons imposé un seuil sur l’identification des similarités entre ces blocs. En développant la deuxième version de *BicMSA,* nous avons bien remarqué que les résultats d’alignement s’améliorent spécifiquement avec les protéines. Ainsi, nous prévoyons à introduire d’autres contraintes sur la génération des *blocs* telle que l’autorisation des *blocs* disposant des brèches (*gapped-bloc)*. Ainsi, nous pensons à déterminer les résultats de ces modifications pour les autres séquences biologiques.

Egalement, d’autres modifications semblent très intéressantes à appliquer et elles sont en cours de réalisation. Citant la génération de la bibliothèque des meilleurs alignements globaux, l’alignement global peut engendrer des problèmes avec les séquences de tailles très différentes. Nous prévoyons alors de changer la méthode d’alignement global à une méthode d’alignement local. Ainsi, nous pouvons améliorer le temps d’exécution de *BicMSA* en changeant l’algorithme exact *Needleman et Wunch* [Needleman et Wunch, 1970].

Par ailleurs, analyser un algorithme revient à prévoir les ressources (i.e. la quantité de mémoire) nécessaires à cet algorithme et à mesurer son temps d’exécution. De ce fait, dans un futur proche, nous envisageons de calculer la complexité de *BicMSA.*

En outre, nos résultats de l'alignement dépendent de la performance de l’algorithme de biregroupement. Actuellement, le biregroupement est encore un axe actif de recherche. Pour ce là, nous prévoyons que lorsque nous changeons la technique de biregroupement, la performance de *BicMSA* va, notamment, s'améliorer.

# Références

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [Adam et al., 2013] | | |  | | | | | |
|  | M. Adam, Szalkowski and M. Anisimova. Graph-based modeling of tandem repeats improves global multiple sequence alignment. *Nucleic Acids Research,* 41(17): e162, 2013. | | | | | | | |
| [Altschul et al., 1990] | | | |  | | | | |
|  | S. Altschul, W. Gish, W. Miller, E. W. Myers et D. J. Lipman, Basic local alignment search tool. *Journal of Molecular Biology,* 215(3) *:* 403-410,1990. | | | | | | | |
| [Angiulli et al., 2008] | | | |  | | | | |
|  | F. Angiulli, E. Cesario and C. Pizzuti. Random walk biclustering for microarray data. *Journal of Information Sciences,* pages 1479-1497, 2008. | | | | | | | |
| [Armougom et al., 2006] | | | |  | | | | |
|  | F. Armougom, S. Moretti, O. Poirot, S. Audic, Dumas, B. Schaeli, V. Keduas and C. Notredame. Expresso: Automatic incorporation of structural information in multiple sequence alignments using 3D-Coffe. *Nucleic Acids Res,* 34 : pages 604-608, 2006. | | | | | | | |
| [Ayadi et Elloumi, 2001] | | | |  | | | | |
|  | W. Ayadi and M. Elloumi. *Algorithms in Computational Molecular Biology : Techniques, Approaches and Applications*, chapter Biclustering of Microarray Data, pages 651-664.  Wiley Book Series on Bioinformatics : Computational Techniques and Engineering.  Wiley-Blackwell, John Wiley & Sons Ltd., New Jersey, USA (Publish), 2011. | | | | | | | |
| [Ayadi et al., 2009] | | | |  | | | | |
|  | W. Ayadi, M. Elloumi and J. K. Hao. A biclustering algorithm based on a bicluster enumeration tree : application to dna microarray data. *BioData Mining,* 2(1) : 9, 2009. | | | | | | | |
| [Barkow et al., 2006] | | | |  | | | | |
|  | S. Barkow, S. Bleuler, A. Prelic, Zimmermann and E. Zitzler. Bicat : a biclustering analysis toolbox. *Bioinformatics,* 22(10) : 1282-1283, 2006. | | | | | | | |
| [Basseur, 2011] | | |  | | | | | |
|  | M. Basseur. Introduction à la bio-informatique. 2011. | | | | | | | |
| [Ben-Dor et al., 2002] | | | |  | | | | |
|  | A. Ben-Dor, B. Chor, R. Karp and Z. Yakhini. Discovering local structure in gene expression data : the order-preserving submatrix problem. In *RECOMB '02 :* *Proceedings of the Sixth Annual International Conference on Computational Biology*, pages 49-57, New York, NY, USA, 2002. ACM.2002. | | | | | | | |
| [Berkhin et al., 2002] | | |  | | | | | |
|  | P. Berkhin and J. D. Becher. Learning Simple Relations: Theory and Applications. In *Proceedings* of the The *2nd SIAM International Conference on Data Mining*, pages 420-436, 2002. | | | | | | | |
| [Blanchette et  Tompa, 2003] | | |  | | | | | |
|  | Mathieu Blanchette and Martin Tompa. FootPrinter: a program designed for phylogenetic footprinting. Nucleic Acids Res, 31(13): 3840–3842, 2003. | | | | | | | |
| [Bleuler et al., 2004] | | |  | | | | | |
|  | S. Bleuler, A. Prelic and E. Zitzler. An ea framework for biclustering of gene expression data. In *Proceedings of Congress on Evolutionary Computation,* pages 166-173, 2004. | | | | | | | |
| [Bradley et al., 2009] | | |  | | | | | |
|  | R. Bradley, A. Robert, M. Smoot, S. Juvekar, J. Do, C. Dewey, I. Holmes and L. Pachter. Fast statistical alignment. *PLoS Computational Biology,* 5(5), e1000392, 1382009. | | | | | | | |
| [Bray et al., 2003] | | |  | | | | | |
|  | N. Bray, I. Dubchark and L. Pachter. AVID : A Global Alignment Program. *Genome Res*, 13(1):97-102, 2003 | | | | | | | |
| [Bray et Pachter, 2004] | | |  | | | | | |
|  | N. Bray and L. Pachter. MAVID : Constrained ancestral alignment of multiple sequences. *Genome Res,* vol. 14, pages 693-699, 2004. | | | | | | | |
| [Brudno et al., 2003] | | |  | | | | | |
|  | M. Brudno, M. Chapman, B. Gottgens, S. Batzoglou and B. Morgenstern. Fast and sensitive multiple alignment of large genomic sequences. *BMC Bioinformatics,* 4:66, 2003. | | | | | | | |
| [Brudno et al., 2003] | | |  | | | | | |
|  | M. Brudno, C. Do, G. Cooper, M. Kim, E. Davydov, E. Green, A. Sidow and S. Batzoglou. LAGAN and Multi-LAGAN: Efficient tools for large scale multiple alignment of genomic DNA. *Genome* Res,13: 721-731, 2003. | | | | | | | |
| [Bryan et al., 2006] | | |  | | | | | |
|  | K. Bryan, Cunningham and N. Bolshakova. Application of simulated annealing to the biclustering of gene expression data. In *IEEE Transactions on Information Technology on Biomedicine,* 10(13) : 519-525, 2006. | | | | | | | |
| [Carrillo et Lipman, 1988] | | | | |  | | | |
|  | H. Carrillo and D.J. Lipman. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.,*  48(15): 1073-1082, 1988. | | | | | | | |
| [Carroll et al., 2007] | | |  | | | | | |
|  | H. Carroll, W. Beckstead, T. O'Cannor, M. Ebbert, M. Clement, Q. Snell and D. Mc-Clellan. DNA reference alignment benchmarks based on tertiary structure of encoded proteins. *Bioinformatics,*  23(119) : 2648-2649, 2007. | | | | | | | |
| [Cartwright et Ngila, 2007] | | | | |  | | | |
|  | R. Cartwright and Ngila: global pairwise alignments with logarithmic and affine gap costs. *Bioinformatics,* 23(111): 1427-1428, 2007. | | | | | | | |
| [Chen et Chang, 2009] | | |  | | | | | |
|  | J. R. Chen and Y. I. Chang. A condition-enumeration tree method for mining biclusters from dna microarray data sets. *BioSystems,* 97: 44-59, 2009. | | | | | | | |
| [Cheng et al., 2013] | | |  | | | | | |
|  | K. O. Cheng, N.F. Law and W. C. Siu. Use of biclustering for missing value imputation in gene expression data. *Artificial Intelligence Research,* 2(12) : 2013. | | | | | | | |
| [Cheng et al., 2008] | | |  | | | | | |
|  | K. O. Cheng, N.F. Law, W. C. Siu and A. W. Liew. Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. *BMC Bioinformatics,*  9(210) : 1282-1283, 2008. | | | | | | | |
| [Cheng et Church, 2000] | | | |  | | | | |
|  | Y. Cheng and G. M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93-103. AAAI Press, 2000. | | | | | | | |
| [Cho et al., 2002] | | |  | | | | | |
|  | H. Cho, I. S. Dhillon, Y. Guan and S. Sra. Minimum Sum-Squared Residue Coclustering of Gene Expression. In Data. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, Pages 131, 2004. | | | | | | | |
| [Cline et al., 2002] | | |  | | | | | |
|  | M. Cline, R. Hughey and K. Karplus. Predicting reliable regions in protein sequence alignments. *Bioinformatics,* 18(2) : 306-314, 2002. | | | | | | | |
| [Corpet, 1988] | | |  | | | | | |
|  | F. Corpet. Multiple sequence alignment with hierarchical clustering. *Nucleic Acids Res,* 22(16) pages 10881-10890, 1988. | | | | | | | |
| [Cutello et al., 2011] | | |  | | | | | |
|  | V. Cutello, G. Nicosia, M. Pavone and I. Prizzi. Protein multiple sequence alignment by hybrid bio-inspired algorithms. *Nucl. Acids Res.,* 39(6), 2011. | | | | | | | |
| [Dan, 1997] | | |  | | | | | |
|  | G. Dan. Algorithms On Strings, Trees, and Sequences. *Cambridge University Press,* 1997. | | | | | | | |
| [Das et Idicula, 2010] | | |  | | | | | |
|  | S. Das and S. M. Idicula. Application of cardinality based grasp to the biclustering ofgene expression data. *International Journal of Computer Applications,* 1(18) : 44-51, 2010. | | | | | | | |
| [Dayhoff et al., 1978] | | |  | | | | | |
|  | Dayhoff, Schwartz and Orcutt. A model of evolutionary change in proteins, matrixes for detecting distant relationshipsdans. *Atlas of protein sequence and structure,* 5: 345 - 358, 1978. | | | | | | | |
| [Delcher et al., 2002] | | |  | | | | | |
|  | A. Delcher, A. Phillippy, J. Carlton and S. Salzberg. Fast algorithms for large-scale Genome alignment and comparison. *Nucleic Acids Res,* 30(11) : 2478-2483, 2002. | | | | | | | |
| [Dessimoz et Gil, 2010] | | |  | | | | | |
|  | C. Dessimoz and M. Gil. Phylogenetic assessment of alignments revealsneglected tree signal in gaps. . *Genome Biol.,* 11 : 37, 2010. | | | | | | | |
| [Dharan et Nair, 2009] | | |  | | | | | |
|  | A. Dharan and A. Nair. Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. *BMC Bioinformatics,*  10 : 27, 2009. | | | | | | | |
| [Dhillon et al., 2003] | | |  | | | | | |
|  | I. S. Dhillon, S. Mallela and D. S. Modha. Information-Theoretical Coclustering. In *9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD’03)*, page 89-98, 2003. | | | | | | | |
| [DiMaggio et al., 2008] | | |  | | | | | |
|  | A. DiMaggio, S. R. McAllister, C. A. Floudas, X. J. Feng, J. D. Rabinowitz and H. A. Rabitz. Biclustering via Optimal Re-Ordering of Data Matrices in Systems Biology: Rigorous Methods and Comparative Studies. *BMC Bioinformatics,* 9(458), 2008. | | | | | | | |
| [Divina et Aguilar-Ruiz, 2007] | | | | | | | |  |
|  | F. Divina and J. S. Aguilar-Ruiz. A multi-objective approach to discover biclusters in microarray data. In *GECCO '07 : Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, New York, USA., pages 385–392, 2007. | | | | | | | |
| [Divina et Aguilar-Ruiz, 2006] | | | | | |  | | |
|  | F. Divina and J. S. Aguilar-Ruiz. Biclustering of expression data with evolutionary computation. *IEEE Transactions on Knowledge & Data Engineering,* 18(5) :590-602, 2006. | | | | | | | |
| [Do et al., 2005] | | |  | | | | | |
|  | C. Do, M. Mahabhashyam, M. Brudno and S. Batzoglou. PROBCONS: Probabilistic consistency-based multiple sequence alignment. *Genome Res,* 15 : 330-340, 2005. | | | | | | | |
| [Durbin et al., 1998] | | |  | | | | | |
|  | Durbin, R; Edy, S.; Krogh, A. and Mitchison, G. Biological Sequence Analysis. *Cambridge University Press,* 1998. | | | | | | | |
| [Duret et Bucher, 1997] | | |  | | | | | |
|  | L. Duret and P. Bucher. Searching for regulatory elements in human noncoding sequences. Curr. Op. Struct. Biol., 7: 399–405, 1997. | | | | | | | |
| [Eddy, 1995] | | |  | | | | | |
|  | S. Eddy. Multiple alignment using hidden markov models. In *Proceedings on the International Conference on Intelligent Systems for Molecular Biology*, Cambridge,UK, pages 114-120, 1995. | | | | | | | |
| [Edgar et Sjolander, 2003] | | | |  | | | | |
|  | R. Edgar and K. Sjolander. SATCHMO: sequence alignment and tree construction using hidden Markov models. *Bioinformatics,* 19 (11): 1404-1411, 2003. | | | | | | | |
| [Edgar, 2004] | | |  | | | | | |
|  | R. Edgar. MUSCLE: multiple sequence alignment with high accuracy high throughput. *Nucleic Acids Res,* 32(5) : 1792-1797, 2004. | | | | | | | |
| [Edgar, 2004] | | |  | | | | | |
|  | R. Edgar. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics,* 5(1): 113, 2004. | | | | | | | |
| [Elias, 2006] | |  | | | | | | |
|  | I. Elias. Settling the intractability of multiple Alignment. *J. Computat. Biol.,*  13(7) : 1323-1339, 2006. | | | | | | | |
| [Fayyad et al., 1996] | | |  | | | | | |
|  | U. Fayyad, P.-s. Gregory and pagesSmyth. From Data Mining to Knowledge Discovery in Databases. *American Association for Artificial Intelligence, AI MAGAZINE,* pages 37-39, 1996. | | | | | | | |
| [Feng et Doolittle, 1990] | | |  | | | | | |
|  | D. Feng and R. Doolittle. Progressive alignment and phylogenetic tree construction of protein sequences. *Methods in Enzymology,* 183 : 375–387, 1990. | | | | | | | |
| [Frawley et al., 1992] | | |  | | | | | |
|  | J. Frawley, G. William, Piatetsky-Shapiro and C. Matheus. Knowledge Discovery in DataBases : An Overview. *AI Magazine,* pages 60, 1992. | | | | | | | |
| [Fu et Weng, 2005] | | |  | | | | | |
|  | Y. Fu , Z. Weng. Improvement of TRANSFAC matrices using multiple local alignment of transcription factor binding site sequences. Genome Inform.,16(1):68-72, 2005. | | | | | | | |
| [Gallo et al. 2009] | | |  | | | | | |
|  | C. A. Gallo, A. I. J. And A. Carballido. BiHEA : A hybrid evolutionary approach for microarray biclustering. In *Proceeding BSB '09 Proceedings of the 4th Brazilian Symposium on Bioinformatics,* pages 36-47, 2009. | | | | | | | |
| [Gardner et Giegerich, 2004] | | | | | |  | | |
|  | P. Gardner and R. Giegerich. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics,*  5(1) : 140, 2004. | | | | | | | |
| [Gaul et Schader, 1996] | | |  | | | | | |
|  | W. Gaul and M. Schader. A New Algorithm for Two-Mode Clustering, Data Analysis and Information Systems. *Springer,* pages 15-23, 1996. | | | | | | | |
| [Gibbons et Roth, 2002] | | |  | | | | | |
|  | F. D. Gibbons and F. Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Research,* 12(10) : 1574-1581, 2002. | | | | | | | | |
| [GIBBS et MCINTYRE, 1970] | | | | | |  | | |
|  | A. J. GIBBS and G. A. MCINTYRE. The Diagram, a Method for Comparing Sequences. *European Journal of Biochemistry,* 16 : 11, 4 May 1970. | | | | | | | |
| [Golubchik et al., 2007] | | |  | | | | | |
|  | T. Golubchik, M. Wise, S. Easteal and L. Jermiin. Mind the gaps: evidence of bias in estimates of multiple sequence alignments. *Mol. Biol. Evol.,* 24 : 2433–2442, 2007. | | | | | | | |
| [Gonçalves et al., 2009] | | |  | | | | | |
|  | J. Gonçalves, S. C. Madeira and A. L. Oliveira. Biggests: integrated environment for biclustering analysis of time series gene expression data. 2(124), 2009. | | | | | | | |
| [Gonnick et Wheelis, 1991] | | | |  | | | | |
|  | L. Gonnick and M. Wheelis, The cartoon guide to genetics, Harper Perennial, 1991. | | | | | | | |
| [Gotoh, 1996] | | |  | | | | | |
|  | O. Gotoh. Signaficant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *J Mol Biol.,* 4(264) : 823-838, 1996. | | | | | | | |
| [Gusfield, 1997] | | |  | | | | | |
|  | D. Gusfield. Algorithms On Strings, Trees, and Sequences. *Cambridge University Press,* 1997. | | | | | | | |
| [Hongtao et Jeremy, 2012] | | |  | | | | | |
|  | Hongtao Sun and Jeremy D. Buhler. PhyLAT: a phylogenetic local alignment tool. Bioinformatics , 28 (10) : 1336-1344, 2012. | | | | | | | |
| [Harry, 2001] | | |  | | | | | |
|  | M. Harry. Génétique moléculaire et évolutive. 2001. | | | | | | | |
| [Henikoff et Henikoff, 1992] | | | | |  | | | |
|  | S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Biochemistry,*  89: 10915-10919, 28 August 1992. | | | | | | | |
| [Henikoff et al.,1995] | | |  | | | | | |
|  | S. Henikoff, J. G. Henikoff, W. J. Alford and S. Pie. Automated Construction and Graphical Presentation of Protein Blocks from Unaligned Sequences. *Gene-COMBIS,* pages 17-26, 1995. | | | | | | | |
| [Heringa, 1999] | | |  | | | | | |
|  | J. Heringa. Two strategies for sequence comparison: profile-preprocessed and secondary structure-induced multiple alignment. *Comput. Chem.,* 23 : 341-364, 1999. | | | | | | | |
| [Heringa,1972] | | |  | | | | | |
|  | J. Hartigan. Direct Clustering of a Data Matrix. *J. Am.Statistical Assoc. (JASA),* 67 (337) : 123-129, 1972. | | | | | | | |
| [Higgins et Taylor,2002] | | |  | | | | | |
|  | D.G. Higgins and W. Taylor. Bioinformatics: Sequence, Structure and Databanks. *Oxford University Press,* 2002. | | | | | | | |
| [Higgins et M,1988] | | |  | | | | | |
|  | D. G. Higgins and M. Sharpages. CLUSTAL: A package for performing multiple sequence alignment on a microcomputer. *Gene ,* 73(11):237–244, 1988. | | | | | | | |
| [Hofmann et Puzicha,1999] | | | |  | | | | |
|  | T. Hofmann and J. Puzicha. Latent Class Models for Collaborative Filtering. In Proceedings of th 16th *International Joint Conference on Artificial Intelligence*(1JCA1-99), pages 688-693, 1999. | | | | | | | |
| [Hogeweg et Hesper,1984] | | |  | | | | | |
|  | 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. | | | | | | | |
| [Huang et al., 2006] | | |  | | | | | |
|  | W. Huang, D. Umbach and L. Li. Accurate anchoring alignment of divergent sequences. *Bioinformatics,* 22(11): 29-34, 2006. | | | | | | | |
| [Hubbard et al., 1996] | | |  | | | | | |
|  | T. Hubbard, A. Lesk and A. Tramontano. Gathering them into the fold. *Nature Structural Biology.,* pages 313, 1996. | | | | | | | |
| [Jones et al., 1992] | | |  | | | | | |
|  | D. Jones, W. Taylor and J. Thornton. The rapid generation of mutation data matrices from protein sequences. *Comput Applic Biosci.,* 8, pages275–282, 1992. | | | | | | | |
| [Jose et al., 2013] | | |  | | | | | |
|  | L. F. Jose, I. Iñaki, L. Pedro and C. Borja. A new measure for gene expression biclustering based on non-parametric correlation. *Computer Methods and Programs in Biomedicine*., 112(3): 367-397, 30 September 2013. | | | | | | | |
| [Joung et al., 2013] | | |  | | | | | |
|  | J.-G. Joung, S.-J. Kim, S.-Y. Shin and B.-T. Zhang. A probabilistic coevolutionary biclustering algorithm for discovering coherent patterns in gene expression dataset. *BMC Bioinformatics,* 13(Suppl 17):S12*,* 2012. | | | | | | | |
| [Karplus et al., 1998] | | |  | | | | | |
|  | K. Karplus, C. Barrett and R. Hughey. Hidden markov models for detecting remote protein homologies. *Bioinformatics,* 14 (10) : 846-856, 1998. | | | | | | | |
| [Katoh et al., 2002] | | |  | | | | | |
|  | K. Katoh, K. Misawa, K. Kuma and T. Miyata. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Res.,* 30 :3059–3066, 2002. | | | | | | | |
| [Katoh et al., 2005] | | |  | | | | | |
|  | K. Katoh, K. Kuma, H. Toh and T. Miyata. MAFFT version 5: Improvement in accuracy of multiple sequence alignment. *Nucleic Aids Res.,*  2(33) : 511-518, 2005. | | | | | | | |
| [Katoh et al., 2009] | | |  | | | | | |
|  | K. Katoh, G. Asimenos and H. Toh. Multiple alignment of DNA sequences with MAFFT. *Methods Mol Biol.,* 537 : 39–64, 2009. | | | | | | | |
| [Katoh et Frith, 2012] | | |  | | | | | |
|  | K. Katoh and M. Frith. Adding unaligned sequences into an existing alignment using MAFFT and LAST. *Bioinformatics,*  28 : 3144–3146, 2012. | | | | | | | |
| [Katoh et Standley, 2013] | | |  | | | | | |
|  | K. Katoh and D. Standley. MAFFT: iterative refinement and additional methods*. Methods in Molecular Biology* 1079(2014) :131-146*,* 2013. | | | | | | | |
| [Klugar et al., 2003] | | |  | | | | | |
|  | Y. Klugar, R. Basri, J. Chang and M. Gerstein. Spectral biclustering of microarray data : coclustering genes and conditions. *Genome Research,* 13 (4):703, 2003. | | | | | | | |
| [Lassman et Sonnhammer, 2005] | | | | | | | |  |
|  | T. Lassman and L. Sonnhammer. Automatic assessment of alignment quality. *Nucleic Acids Res.,* pages 7122-7128, 2005. | | | | | | | |
| [Lassmann et Sonnhammer, 2002] | | | | | | | |  |
|  | T. Lassmann and E. Sonnhammer. Quality Assessment of Multiple Alignment Programs. *FEBS Letters,* pages 126-130, 2002. | | | | | | | |
| [Lassmann et Sonnhammer, 2002] | | | | | | | |  |
|  | T. Lassman and L. Sonnhammer. Quality assessment of multiple alignment programs. *FEBS Lett.,* pages 126-130, 2002. | | | | | | | |
| [Lassmann et Sonnhammer, 2005] | | | | | | | |  |
|  | T. Lassman and L. Sonnhammer. KALIGN: An accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics,*  6 : 298, 2005. | | | | | | | |
| [Lecompte, 2007] | | |  | | | | | |
|  | O. Lecompte, Analyse de Séquences Macromoléculaires II, Laboratoire de Bioinformatique et Génomique Intégratives – IGBMC, pages 12, 2007. | | | | | | | |
| [Lehmann et al., 1998] | | |  | | | | | |
|  | L. Lehmann, D'Abrera and H. J. M. Nonparametrics : Statistical methods based on ranks. *NJ : Prentice-Hall,* pages 292-323, 1998. | | | | | | | |
| [Letsch et al., 2010] | | |  | | | | | |
|  | H. Letsch, Kuck, R. Stocsits and B. Misof. The impact of rRNA secondary structure consideration in alignment and tree reconstruction: simulated data and a case study on the phylogeny of hexapods. *Mol Biol Evol.*  27 : 2507–2521, 2010. | | | | | | | |
| [Lipman et Pearson, 1985] | | |  | | | | | |
|  | Lipman DJ. and Pearson WR. Rapid and sensitive protein similarity searches. Science, 227(4693):‎ 1435-41, 1985. | | | | | | | |
| [Liu et Wang, 2007] | | |  | | | | | |
|  | X. Liu and L. Wang. Computing the maximum similarity bi-clusters of gene expression data. *Bioinformatics,* 23(1): 50-56, 2007. | | | | | | | |
| [Liu et al., 2001] | | |  | | | | | |
|  | X. D. Liu, D. Brutlag and J. Liu. BioProspector: Discovering Conserved DNA Motifs In Upstream Regulatory Regions of Co-expressed Genes. *Pacific Symposium on Biocomputing,* pages 127-138, 2001. | | | | | | | |
| [Liu et Wang., 2003] | | |  | | | | | |
|  | J. Liu and W. Wang. OP-Cluster: Clustering by Tendency in High Dimensional Space. In *Data Mining*, 2003. | | | | | | | |
| [Löytynoja et Goldman, 2010] | | | | | | |  | |
|  | A. Löytynoja and N. Goldman. webPRANK: a phylogeny-aware multiple sequence aligner with interactive alignment browser. *BMC Bioinformatics,* 11(1):579, 2010. | | | | | | | |
| [Madeira et Oliveira, 2004] | | |  | | | | | |
|  | S. C. Madeira and A. L. Oliveira. Biclustering Algorithms for Biological Data Analysis: A Survey. *IEEE Transactions on Computational Biology and Bioinformatics ,* 1(1) :24-45, 2004. | | | | | | | |
| [Martinez et al., 2006] | | |  | | | | | |
|  | R. Martinez, N. Pasquier, C. Pasquier and M. Collard. Analyse des groupes de gènes co-exprimés (AGGC) : un outil automatique pour l'interprétation des expériences de biopuces. In actes de la *13ème* Rencontres de la Société Francophone de Classification (SFC 2006). Metz, France; 2006. | | | | | | | |
| [Matthew, 2013] | | |  | | | | | |
|  | Matthew. Biclustering.pdf. 2010. [En ligne]. Available: http://www.cs.princeton.edu/courses/archive/spr05/cos598E/Biclustering.pdf. [Accès le 03 Mars 2013]. | | | | | | | |
| [Mhamdi et Elloumi, 2009] | | |  | | | | | |
|  | F. Mhamdi and M. Elloumi. Algorithms for Proteins Biclustering. *the 2009 IEEE International Workshop on Bioinformatics and Life Science Modeling and Computing (BLSMC’09),* (Bradford, UK) : (Mai 2009). | | | | | | | |
| [Min et al., 2005] | | |  | | | | | |
|  | Z. Min, F. Weiwu, Z. Junhua and C. Zhongxian. MSAID:multiple sequence alignment based on a measure of information discrepancy. *Comput Biol Chem.,* 29:175-181, 2005. | | | | | | | |
| [Mitra et Banka, 2006] | | |  | | | | | |
|  | S. Mitra and H. Banka. Multi-objective evolutionary biclustering of gene expression data. *Pattern Recognition.,* 39(12) : 2464-2477, 2006. | | | | | | | |
| [Mokaddem et Elloumi, 2011] | | | | | |  | | |
|  | A. Mokaddem and M. Elloumi. Algorithms for the alignment of biological sequences, in Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications (eds M. Elloumi and A. Y. Zomaya), John Wiley & Sons, Inc., Hoboken, NJ, USA. doi: 10.1002/9780470892107.ch12, Pages 241-260, 2011. | | | | | | | |
| [Morgenstern, 1999] | | |  | | | | | |
|  | B. Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics,* 15(3) : 211-218, 1999. | | | | | | | |
| [Murali et al., 2003] | | |  | | | | | |
|  | T. Murali, Kasif and S. Extracting Conserved Gene Expression Motifs from Gene Expression Data. *Proc. Pacific SympagesBiocomputing,* 8 : 77-88, 2003. | | | | | | | |
| [Myers et Arnold, 2003] | | |  | | | | | |
|  | J. Myers and D. Arnold. Research design and statistical analysis. 2003. | | | | | | | |
| [Needleman et Wunsch, 1970] | | | | |  | | | |
|  | Needleman and S. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J.Mol. Biol.,* 48(3) : 53-443, 1970. | | | | | | | |
| [Nelesen et al., 2008] | | |  | | | | | |
|  | S. Nelesen, K. Liu, D. Zhao and C. W. T. Linder. The effect of the guide tree on multiple sequence alignments and subsequent phylogenetic analyses. *Pacific Symposium on Biocomputing* 13:25-36., 2008. | | | | | | | |
| [Noé et Kucherov, 2005] | | |  | | | | | |
|  | L. Noé and G. Kucherov. YASS : enhancing the sensitivity of DNA similarity search. *Nucleic Acids Res.,* 33(2) : 540-543, 2005. | | | | | | | |
| [Notredame et al., 1998] | | |  | | | | | |
|  | C. Notredame, L. Holm and D. Higgins. COFFEE : an objective function for multiple sequence alignments. *Bioinformatics,* 14(5) : 407-422, 1998. | | | | | | | |
| [Notredame et al., 2000] | | |  | | | | | |
|  | C. Notredame, D. Higgins and J. Heringa. T-COFFEE: A novel method for multiple sequence alignments. *J. Mol. Biol.,* pages 205-217, 2000. | | | | | | | |
| [Notredame , 2002] | | |  | | | | | |
|  | C. Notredame. Recent progresses in multiple sequence alignment : a survey. *. Pharma-cogenomics,*  3 :12, 2002. | | | | | | | |
| [Notredame et Higgins, 1996] | | | | | |  | | |
|  | C. Notredame and D. Higgins. SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.,* pages 1515-1524, 1996. | | | | | | | |
| [Nuin et al., 2006] | | |  | | | | | |
|  | Nuin, Z. Wang and E. Tillier. The accuracy of several multiple sequence alignment programs for proteins. *BMC Bioinformatics ,* 7: 471, 2006. | | | | | | | |
| [Oliver et al., 2012] | | |  | | | | | |
|  | V. Oliver, S. Bleuler and W. Gruissem. Exact biclustering algorithm for the analysis of large gene expression data sets. *BMC Bioinformatics ,* 13(18):A10,2012. | | | | | | | |
| [Papadopoulos et Agarwala, 2007] | | | | | | | |  |
|  | J. Papadopoulos and R. Agarwala. COBALT : constraint-based alignment tool for multiple protein sequences. *Bioinformatics,* 23 (9): 1073-1079, 2007. | | | | | | | |
| [PAVY et al., 1999] | | |  | | | | | |
|  | N. PAVY, C. MATHÉ, S. ROMBAUTS, P. ROUZÉ. Genomics and bio-computing. OCL-Oléagineux Corps Gras Lipides, Opgenomen in: SCI, 6 (2):148-154, 1999. | | | | | | | |
| [Pearon and Lipman, 1988] | | |  | | | | | |
|  | W. Pearon and D. Lipman, *Improves tools for biological sequence comparison.* pages 2444-2448, U.S.A., 1988. | | | | | | | |
| [Pei et Grichin, 2001] | | |  | | | | | |
|  | J. Pei and N. Grishin. Al2co : Calculation of positional conservation in a protien sequence alignment. 17(8) : 700-721, 2001. | | | | | | | |
| [Pei et Grichin, 2007] | | |  | | | | | |
|  | J. Pei and N. Grichin. PROMAILS: Towards accurate multiple sequence alignments of distantly related proteins. *Bioinformatics,* 7(23) : 802-808, 2007. | | | | | | | |
| [Philippe, 2005] | | |  | | | | | |
|  | G. Philippe. upgma. 30 06 2005. [En ligne]. Available: http://philippe.gambette.free.fr/SCOL/NotesBioinfo/node37.html#upgma. [Accès le 12 Juin 2013]. | | | | | | | |
| [Pio et al., 2013] | | |  | | | | | |
|  | G. Pio, M. Ceci, D. D'Elia, C. Loglisci and D. Malerba. A Novel Biclustering Algorithm for the Discovery of Meaningful Biological Correlations between microRNAs and their Target Genes. *BMC Bioinformatics ,* 22, 2013. | | | | | | | |
| [Pontes et al., 2007] | | |  | | | | | |
|  | B. Pontes, F. Divina, R. Giraldez and J. Ruiz. Virtual error : A new measure for evolutionary biclustering. *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics,* pages 217-226, 2007. | | | | | | | |
| [Prelic et al., 2006] | | |  | | | | | |
|  | A. Prelic, S. Bleuler, Zimmermann, Buhlmann, W. Gruissem, L. Hennig, L. Thiele and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data,. *. Bioinformatics,* 22(9) : 1122-1129, 2006. | | | | | | | |
| [Quackenbush, 2006] | | |  | | | | | |
|  | J. Quackenbush. Microarray analysis and tumors classification. *The New ENGLAND Journal of MEDECINE,* 354(23) : 2463-2472, 2006. | | | | | | | |
| [Roth et al., 1998] | | |  | | | | | |
|  | F. Roth, J. Hughes, Estep and G. Chur. Finding DNA Regulatory Motifs Within Unaligned Noncoding Sequences Clustered By Wholegenome mRNA Quantitation. *Nat. Biotechnol.,* pages 939-945, 1998. | | | | | | | |
| [Russell et al., 2008] | | |  | | | | | |
|  | D. Russell, H. Out and K. Sayood. Grammar-based distance in progressive multiple sequence alignment. *BMC Bioinformatics*, vol.9 2008. | | | | | | | |
| [Sahraeian et Yoon, 2010] | | |  | | | | | |
|  | S. M. E. Sahraeian and B.-J. Yoon. PicXAA: greedy probabilistic construction of maximum expected accuracy alignment of multiple sequences. *Nucl. Acids Res.,* 38(15) : 4917-4928, 2010. | | | | | | | |
| [Sahraeian et Yoon, 2011] | | |  | | | | | |
|  | S. Sahraeian and B. Yoon. PicXAA-R: efficient structural alignment of multiple RNA sequences using a greedy approach. *BMC Bioinformatics ,* 12 : 38, 2011. | | | | | | | |
| [Saitou et Nei, 1987] | | |  | | | | | |
|  | N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.,* 4 : 406-125, 1987. | | | | | | | |
| [Sebastiani, 2005] | | |  | | | | | |
|  | F. Sebastiani. Text Mining and its Applications to Intelligence,CRM and Knowledge Managemen. *WIT Press,* pages 109–129, 2005. | | | | | | | |
| [Segal et al., 2003] | | |  | | | | | |
|  | E. Segal, A. Battle and D. Koller. Decomposing Gene Expression into Cellular Processes. In *Pacific Symptom Biocomputing*, 8:89-100, 2003. | | | | | | | |
| [Seridi et al., 2011] | | |  | | | | | |
|  | K. Seridi, L. Jourdan and G. Talbi. Multi-objective evolutionary algorithm for biclustering in microarrays data. *IEEE Congress of Evolutionary Computation,* pages 2593-2599, 2011. | | | | | | | |
| [Sharan et al., 2002] | | |  | | | | | |
|  | R. Sharan, R. Elkon and R. Shamir. Cluster analysis and its applications to gene expression data. *Ernst Schering Workshop on Bioinformatics and Genome Analysis,* pages 83-108, 2002. | | | | | | | |
| [Sievers et al., 2011] | | |  | | | | | |
|  | F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, L. Weizhong, R. Lopez, H. McWilliam, M. Remmert, J. Söding, J. D. Thompson and D. G. Higgins. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Molecular Systems Biology,* 539(7), Octobe 2011. | | | | | | | |
| [Simossis et Heringa, 2005] | | | | |  | | | |
|  | V. Simossis and J. Heringa. PRALINE: A multiple sequence alignment toolbox that integrates homology-extended and secondary structure information. *Nucleic Acids Res.,* pages 289-294, 2005. | | | | | | | |
| [Smagala et al., 2005] | | |  | | | | | |
|  | J. Smagala, E. Dawson, M. Mehlmann, M. Townsend, R. Kuchta and K. Rowlen. Confind : a robust tool for conserved sequence identification. *Bioinformatics,* pages 4420-4422, 2005. | | | | | | | |
| [Smith et Waterman, 1981] | | |  | | | | | |
|  | T. F. Smith and M. S. Waterman. Identification of Common Molecular.Subsequences. *Journal of Molecular Biology,* 147(1):195 - 197, 1981. | | | | | | | |
| [Smith et Smith, 1992] | | |  | | | | | |
|  | R. Smith and T. Smith. Pattern-Induced Multi-sequence Alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for comparative protein modelling. *Protein Engineering,* 5 : 35-41, 1992. | | | | | | | |
| [Stoye et Meyer, 1998] | | |  | | | | | |
|  | J. Stoye, D. Evers and F. Meyer. ROSE: Generating sequence families. *Bioinformatics,* 14 : 157-163, 1998. | | | | | | | |
| [Stoye et al., 1997] | | |  | | | | | |
|  | J. Stoye, V. Moulton et A. Dress. Dca : an efficient implementation of the divideand-conquer approach to simultaneous multiple sequence alignment. *Comput. Apll. Biosci.,* 6(13): 625–626, 1997. | | | | | | | |
| [Stoye, 1998] | | |  | | | | | |
|  | J. Stoye. Multiple sequence alignment with the divide-and-conquer method. Gene. 1998, 12;211(2):GC45-56 | | | | | | | |
| [Subramanian et al., 2005] | | |  | | | | | |
|  | A. Subramanian, J. Weyer-Menkhoff, M. Kaufmann and B. Morgenstern. DIALIGN-T: an improved algorithm for segment-based multiple sequence alignment. *BMC Bioinformatics,* 6 :66, 2005. | | | | | | | |
| [Subramanian et al., 2008] | | |  | | | | | |
|  | A. Subramanian, M. Kaufmann and B. Morgenstern. DIALIGN-TX: Greedy and progressive approaches for segment-based multiple sequence alignment. *Algorithms for Molecular Biology,* 3: 6, 2008. | | | | | | | |
| [Sze et al., 2006] | | |  | | | | | |
|  | S. Sze, Y. Lu and Q. Yang. A ploynomail time solvable formulation of multiple sequence alignment. *J. Comput. Bio.,* pages 309-319, 2006. | | | | | | | |
| [Tanay et al., 2002] | | |  | | | | | |
|  | A. Tanay, R. Sharan and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Boinformatics,*  18:136-144, 2002. | | | | | | | |
| [Teng et Chan, 2008] | | |  | | | | | |
|  | L. Teng and L. Chan. Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. *Journal of Signal Processing Systems,*  50(3) : 267-280, 2008. | | | | | | | |
| [Thompson et al., 1999] | | |  | | | | | |
|  | J. Thompson, F. Plewniak and O. Poch. Balibase : A benchmark alignments database for the evaluation of multiple sequence alignment programs. *Bioinformatics,*  15: 87–88, 1999. | | | | | | | |
| [Thompson et al., 2005] | | |  | | | | | |
|  | J. Thompson, Koehl, R. Ripp and O. Poch. Balibase 3.0: Latest developments of the multiple sequence alignment benchmark. PROTEINS : Structure, Function, and Bioinformatics. *Bioinformatics,*  61: 127–136, 2005. | | | | | | | |
| [Thompson et al., 2001] | | |  | | | | | |
|  | J. Thompson, F. Plewniak, R. Ripp, J. Thierry and O. Poch. Towards a reliable objective function for multiple sequence alignments. *J .Mol. Biol.,* pages 937-951, 2001. | | | | | | | |
| [Thompson et al., 1994] | | |  | | | | | |
|  | J. Thompson, D. Higgins and T. Gibson. CLUSTALW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalities and weight matrix choice. *Nucleic Acids Res,*  22(22) : 4673-4680, 1994. | | | | | | | |
| [Thompson et al., 1997] | | |  | | | | | |
|  | J. Thompson, T. Gibson, F. Plewniak, F. Jeanmougin et D. Higgins. The CLUSTALX windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Research,*  25(24) : 4876–4882, 1997. | | | | | | | |
| [Thompson et al., 2000] | | |  | | | | | |
|  | J. Thompson, F. Plewniak, J. Thierry and O. Poch. DBCLUSTAL : rapid and reliable global multiple alignments of protien sequences detected by database searches. *Nucleic Acids Res.,* 28(15): 2919-2926, 2000. | | | | | | | |
| [Van Mechelen et al., 2004] | | | | |  | | | |
|  | I. Van Mechelen, H. H. Bock and De Boeck. Two-Mode Clustering Methods:A Structured Overview. *Statistical Methods in Medical Research J.,* vol. 13, n° 5, pages 94-363, 2004. | | | | | | | |
| [Van Walle et al., 2005] | | |  | | | | | |
|  | I. Van Walle, I. Lasters and L. Wyns. SABMARK: a benchmark for sequence alignment that covers the entire know fold space. *Bioinformatics,* pages 1267-1268, 2005. | | | | | | | |
| [Wang et al., 2007] | | |  | | | | | |
|  | S. Wang, R. Gutell and Miranker. Biclustering as a method for RNA local multiple sequence alignment. *Bioinformatics,*  23(24) : 3289-3296, 2007. | | | | | | | |
| [Wang et al., 2002] | | |  | | | | | |
|  | H. Wang, W. Wang, J. Yang and S. Yu. Clustering by pattern similarity in large data sets. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, ACM, 394-405, 2002. | | | | | | | |
| [Wang , 2007] | | |  | | | | | |
|  | S. Wang, *On Multiple Sequence Alignment,* The University of Texas at Austin, pages 1-122, 2007. | | | | | | | |
| [Watson et Crick, 1953] | | |  | | | | | |
|  | Watson James et Crick. Francis Molecular structure of nucleic acids. Nature, 4356 : 737–738, 1953. | | | | | | | |
| [WATERMA et al., 1990] | | |  | | | | | |
|  | M. S. WATERMA, S. MICHAEL and R. JONES. Consensus Methods for DNA and Protein Sequence Alignment. vol. 183, 1990. | | | | | | | |
| [Whelan et Goldman, 2001] | | | | | |  | | |
|  | S. Whelan and N. Goldman. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Molecular Biology and Evolution,* n° 18, pages 691-699, 2001. | | | | | | | |
| [Wolfgang et al., 2011] | | |  | | | | | |
|  | Otto Wolfgang, F. S. Peter, J. P. Sonja. Phylogenetic Footprinting and Consistent Sets of Local Aligments. In CPM'11 Proceedings of the 22nd annual conference on Combinatorial pattern matching. Springer-Verlag Berlin, Heidelberg, pages 118-131, 2011. | | | | | | | |
| [Yang et al, 2003] | | |  | | | | | |
|  | J. Yang, H. Wang, W. Wang and Yu. Enhanced biclustering on expression data. In *BIBE '03,* pages 321-327, 2003. | | | | | | | |
| [Yong et al, 1998] | | |  | | | | | |
|  | V. Yong, S. Chabot, Q. Stuve and G. Williams. Interferon Beta in the Treatment of Multiple Sclerosis: Mechanisms of Action. *Neurology J,* n°51, pages 682-689, 1998. | | | | | | | |
| [Zhang et Kahveci, 2005] | | |  | | | | | |
|  | X. Zhang and T. Kahveci. QOMA: quasi-optimal multiple alignment of protein sequence. *Bioinformatics ,*  23 (2): 162-168, 2007. | | | | | | | |
| [Zhou et Zhou, 2005] | | |  | | | | | |
|  | H. Zhou and Y. Zhou. SPEM: Imroving multiple sequence alignment with sequence profiles and predicted secondary structure. *Bioinformatics,* pages 3615-3621, 2005. | | | | | | | |

# Annexe

## La Bioinformatique : Naissance

Au début du *XXème*siècle et après la présentation des 3 lois de l’*hérédité* par *Mendel* en *1866*, les biologistes ont utilisé l’observation *microscopique* des noyaux de cellules afin de découvrir, pour la première fois, les *chromosomes* et les *gènes*. Vers la fin de la guerre mondiale que *Waston et Crick* ont présenté le *modèle en double hélice* de la structure d’*ADN* [Watson et Crick, 1953]. Cependant, les chercheurs ont commencé à s’intéresser de plus en plus à la *génomique* ce qui a donné naissance à une nouvelle discipline : le *génie génétique*. En *1989*, le *séquençage de génomes* a été évolué pour arriver à son achèvement complet en *2003*. Par la suite, les informations sur la structure des séquences sont de plus en plus stockées et accumulées dans des bases de données ce qui déclare une nécessité d’avoir des outils informatisés pour l’acquisition, le stockage, la diffusion et l’interprétation automatiques des données biologiques. En conséquence, l’informatique joue un rôle central au cœur de la biologie ce qui a donné une naissance à la *bioinformatique*.

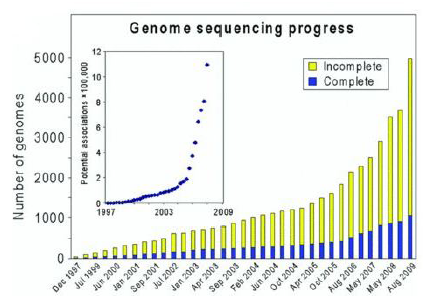


Figure A-21 La progression du séquençage des génomes. [Hanson et al., 2010]

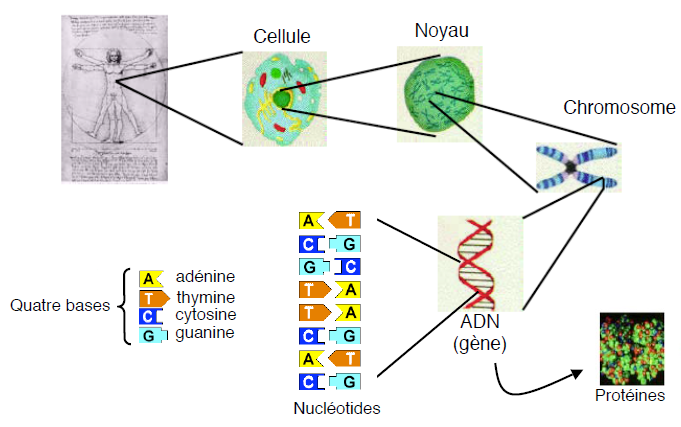
## La Bioinformatique : Définitions et Notations

Dans cette section, nous définissons quelques termes utilisés dans la Bioinformatique. Tout d’abord, la Figure A-2 montre un ensemble d’images de différents composants biologiques. En effet, chaque diagramme représente une image grossie d’un facteur de 10 de la précédente : commençant de doigt vers la peau puis les cellules de la peau ensuite la structure des cellules et la structure d’une *mitochondrie*[[5]](#footnote-5) ainsi la structure d’un *ribosome*[[6]](#footnote-6) pour arriver à une structure de deux protéines dont ils se composent d’un ensemble d’atomes.



Figure A-2. Les composants génomiques [Basseur, 2011]

En se basant sur ces diagrammes et sur la figure précédente (voir Figure A-3), nous présentons les définitions suivantes :



**Figure A-3 Les composants biologiques**

* *Définition (Macromolécules/molécule polymère) : une macromolécule généralement est considérée comme une très grande molécule puisqu’elle possède une masse moléculaire relativement élevée.*
* *Définition (Chromosomes) : L’information génétique est contenue dans les chromosomes. Chaque cellule d’un être humain comporte 23 paires de chromosomes (nombre variant pour chaque espèce animale).*
* *Définition (Cellules) : On peut considérer la cellule comme une usine pour produire des objets à partir d’un plan (exemple : les protéines).*
* *Définition (Nucléotides) : Les nucléotides sont constitués à partir d’une base azotée (Adénine, Thymine, Guanine, Cytosine) d’un sucre et d’un groupement phosphate. Par convention, on utilise la première lettre de chacune des bases pour appeler les nucléotides.*
* *Définition (Gènes) : Le gène est un composant d’un génome ou une portion d'un chromosome. Il code les composants de bases de la vie comme les protéines.*
* *Définition (Génétique) : Elle permet d'étudier les caractères héréditaires des individus, leur transmission au fil des générations et leurs variations ou mutations (Ramsay, 1998).*
* *Définition (Génome) : Le génome est un ensemble des bases réparties en 23 segments linéaires comme les chromosomes. Le génome d’une bactérie est composé de 4.6 Millions de bases et celui de l’homme est composé d’environ 3 Milliards de bases [3 Go! !]. Le tableau suivant donne quelques exemples de taille de génomes et de nombres de chromosomes suivant les espèces (voir Tableau A-1). On remarque que les tailles sont très variables d’une espèce à une autre.*

|  |  |  |
| --- | --- | --- |
| **Organismes** | **Taille du génome en Mb** | **Nombre de chromosomes** |
| Procaryotes | | |
| Mycoplasma genitalium | 0,6 | 1 |
| Bacillus subtilis | 4,2 | 1 |
| Escherichia coli | 4,7 | 1 |
| Eucaryotes | | |
| Arabidopsis thaliana (plante) | 125 | 5 paires |
| Drosophila melanogaster (mouche) | 180 | 4 paires |
| Zea mays (maÏs) | 2 500 | 5 paires |
| Mus musculus (souris) | 3 000 20 paires | 3 000 20 paires |
| Homo sapiens (homme) | 3 200 | 23 paires |
| blé tendre | 16 000 | 21 paires |
| Amoeba dubia (amibe) | 675 000 | 17 paires |

Tableau A-1 Tailles de génomes et nombres de chromosomes.

* *Définition (Génomique) : La génomique est la science qui étudie la structure, le contenu et l’évolution des génomes suivant des informations génétiques. En effet, l’information génétique est principalement représentée sur les chromosomes par une suite de gènes séparés par des régions intergéniques.*
* *Définition (Codes Acides nucléiques et Acides Aminés) : Les codes des acides nucléiques se présentent dans le Tableau A-2, tandis que les codes des acides aminés (24 codes pour les acides aminés) se présentent dans le Tableau A-3.*

| **Code des acides nucléiques** | **Signification** |
| --- | --- |
| A | [Adénine](http://fr.wikipedia.org/wiki/Ad%C3%A9nine) |
| C | [Cytosine](http://fr.wikipedia.org/wiki/Cytosine) |
| G | [Guanine](http://fr.wikipedia.org/wiki/Guanine) |
| T | [Thymine](http://fr.wikipedia.org/wiki/Thymine) |
| U | [Uracile](http://fr.wikipedia.org/wiki/Uracile) |
| R | [puRine](http://fr.wikipedia.org/wiki/Purine)(A or G) |
| Y | [pYrimidines](http://fr.wikipedia.org/wiki/Pyrimidine)(C, T or U) |
| - | brèche |
| Tableau A-2 Les acides nucléiques | |

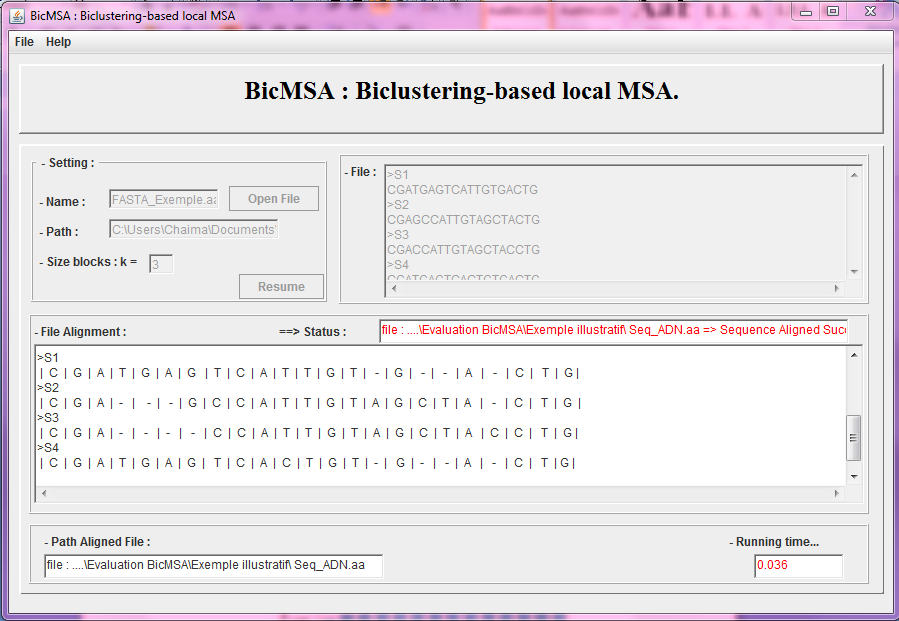
|  |  |  |  |
| --- | --- | --- | --- |
| **Nom** | **Code** | **Nom** | **Code** |
| Alanine | A | Leucine | L |
| Arginine | R | Lysine | K |
| Asparagine | N | Méthionine | M |
| Aspartate | D | Phénylalanine | F |
| Cystéine | C | Proline | P |
| Glutamate | E | Sérine | S |
| Glutamine | Q | Thréonine | T |
| Glycine | G | Tryptophane | W |
| Histidine | H | Tyrosine | Y |
| Isoleucine | I | Valine | V |

**Tableau A-3 Liste des 20 acides aminés.**

## *BicMSA* : Interface

En utilisant le langage java, nous avons développé notre algorithme *BicMSA*. L’interface de *BicMSA* se présente par la Figure A-6. Le programme prend en entré un fichier FASTA dont les lignes contiennent, soit le nom de la séquence, soit la séquence à aligner (voir Figure A-4). En ajoutant la taille du bloc, *BicMSA* fourni un fichier FASTA contenant les séquences alignées (voir Figure A-5). Ainsi, il produit le temps d’exécution pour chaque alignement.

|  |  |
| --- | --- |
|  |  |
| **Figure A-4 Fichier FASTA des ADNs** | **Figure A-5 Fichier FASTA des ADNs alignées** |



**Figure A-6 Interface de BicMSA**

## *BicMSA* : Choix de la taille du bloc

Nous présentons dans cette section les différentes figures présentant la taille à fixer pour aligner les séquences d’ADN, d’ARN et de protéine.

**Figure A-7 BRALIBASE- Comparaison de la taille des blocs suivant les moyennes des SPs.**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | |
| Figure A-8 BALIBASE- Comparaison de la taille des blocs suivant les moyennes des SPs. |  | Figure A-9 BALIBASE- Comparaison de la taille des blocs suivant les valeurs maximales des SPs. | |
|  | |  |  |
| Figure A-10 DIRMBASE- Comparaison de la taille des blocs suivant les moyennes des SPs. | |  | Figure A-11 DIRMBASE- Comparaison de la taille des blocs suivant les valeurs maximales des SPs. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithme d’Alignement Multiple de Séquences Biologiques basé sur la Technique de Biregroupement | | | | |
| **Résumé**  Dans ce mémoire, nous développons un algorithme d’*alignement multiple*, appelé *BicMSA (****Bic****lustering-based local* ***MSA***). Le problème de *MSA-local* est *NP-Complet.* Il est souvent destiné à grouper un ensemble de sous-séquences conservées. Cependant, le choix de la technique de regroupement peut facilement influer la qualité de l'alignement. De ce fait, nous avons eu recourt à adapter le *biregroupement* pour générer des *MSA locaux*. À présent, l’utilisation du *biregroupement* dans la résolution des problèmes de *MSA* est une piste qui n'a pas été bien explorée. Notre algorithme se présente parmi les premiers algorithmes adaptant le *biregroupement* dans la résolution de *MSA*. *BicMSA* trouve un compromis entre l’approche *énumération de bigroupe*s et l’approche *Diviser-et-conquérir* afin de générer à la fois des groupes de séquences s*imilaires* et un *MSA-local* entre toutes les séquences. Cette mémoire est répartie en quatre parties : Dans la première partie, nous présentons un état de l’art sur l’*alignement des séquences biologiques*. Dans la deuxième partie, nous présentons un état de l’art sur le *biregroupement des données biologiques*. Dans la troisième partie, nous présentons notre algorithme, *BicMSA*. Dans la quatrième partie, nous effectuons une étude expérimentale sur les différents types de séquences (*ADNs*, *ARNs* et *protéines).* Cette étude a permis de montrer que notre algorithme est performant et arrive à générer des alignements proches de l’optimal que certains algorithmes de *MSA*. | | | | |
|  | **Mots clés :** Alignement Multiple, Alignement par Paires, MSA-local, Fonctions d’évaluation, SPS, CS, approche Diviser-et-Conquérir, BALIBASE, BRALIBASE, DIRMBASE, Biregroupement, Algorithme Systématique, Approche Énumération de bigroupes, MSA-local par Biregroupement, ADN, ARN, protéines. | | |  |
|  | |  |  | |
| Algorithm Multiple Sequence Alignment based on Biclustering | | | | |
| **Abstract**  In this report of master's degree, we developed a new *MSA* algorithm, a *biclustering-based MSA*. A challenge in *MSA* is that the alignment of sequences is often by grouping a set of conserved functional subsequences. However, the choice of the method of grouping can easily impact the quality of the alignment. Thus, we defined a representation of the *MSA-problem* enabling the application of *biclustering* algorithm. At the moment, the use of the *biclustering* in solving the *MSA-problems* is a method which was not explored well. Our algorithm appears among the first algorithms adapting the *biclustering* for *MSA-problems*. We developed a computer program for *local-MSA*, *BicMSA* (***Bic****lustering-based local* ***MSA***), that combined *biclusters enumeration* approach with *divide-and-conquer* to generate a *local-MSA*. The net result is both a multiple sequence alignment and a hierarchical clustering of the sequences. This report is divided into four parts: In the first part, we present a state of the art on the *alignment of biological sequence*. In the second part, we present a state of the art on the *biclustering of biological data*. In the third part, we present our new *MSA* algorithm based on *biclustering* . Finally, in the fourth section, we conduct an experimental study on the different biological sequence (*AND*, *ARN* and *proteins*). *BicMSA* was compared with a suite of leading *MSA* programs. With respect to quantitative measures and biological validation of *MSA*, *BicMSA* scores comparable to or better than the other leading *MSA* programs. | | | | |
|  | **Keywords :** Multiple Sequence Alignment, Parwise Alignment, local-MSA, Evaluation Functions, SPS, CS, Divide-and-Conquer approach, BALIBASE, BRALIBASE, DIRMBASE, Biclustering, Systematic Algorithm, Biclusters Enumeration Approach, local-MSA with Biclustering, ADN, ARN, proteins. | | |  |

1. ***Modèle complet*** : C’est une sous-matrice composée d’un ensemble de valeurs strictement croissantes et d’un ensemble de lignes ordonnées en suivant la permutation *π*. [↑](#footnote-ref-1)
2. *Sélection :* ce processus est analogue à un processus de sélection naturelle, les individus les plus adaptés gagnent la compétition de la reproduction tandis que les moins adaptés meurent avant la reproduction, ce qui améliore globalement l'adaptation. [↑](#footnote-ref-2)
3. *Le croisement :* c’est la transposition informatique du mécanisme de production de chromosomes qui héritent partiellement des caractéristiques des parents. Son rôle fondamental est de permettre la *recombinaison* des informations présentes dans le patrimoine génétique de la population. [↑](#footnote-ref-3)
4. *Une* mutation*:* c’est une modification de l'information génétique dans le génome d'une cellule ou d'un virus. C'est donc une modification de la séquence de l'ADN, ou bien de l'ARN. C'est l'une des causes principales de l'évolution des espèces. [↑](#footnote-ref-4)
5. Une mitochondrie (du grec mitos, fil et chondros, grain) est un organite à l'intérieur d'une cellule eucaryote, dont la taille est de l'ordre du micromètre. [↑](#footnote-ref-5)
6. Le ribosome est une machine essentielle de la cellule qui assure la synthèse protéique à partir de l'information génétique. [↑](#footnote-ref-6)