

Implémentation d'une métaheuristique pour l'inférence phylogénétique

Julien HECK



Mémoire présenté sous la direction du **Prof. Jean CARDINAL**
et sous la co-direction du **Prof. Michel MILINKOVITCH**
en vue de l'obtention du grade de Licencié en Informatique
Année académique 2005–2006

Table des matières

Remerciements	iii
Préface	iv
1 Introduction	1
1.1 La reconstruction phylogénétique	1
1.2 Les données traduisant l'évolution	3
1.2.1 L'alignement des séquences moléculaires	4
1.2.2 Le concept de similitude	6
1.3 Historique des méthodes de reconstruction phylogénétique	7
1.3.1 La phénétique	8
1.3.2 Le cladisme	9
1.3.3 La hiérarchie actuelle des méthodes	10
1.4 Les arbres phylogénétiques	11
1.5 Méthodes de reconstruction d'arbres phylogénétiques	12
1.5.1 Méthodes de caractères	13
1.5.2 Méthodes de matrice de distance	15
2 Modélisation du problème	17
2.1 Rappels	17
2.1.1 Les graphes	17
2.1.2 Les arbres	18
2.2 Modélisation des arbres phylogénétiques	19
2.3 Définition du problème	22
2.4 Décomposition de la matrice topologique	27
2.5 Formulation finale	31

3	Stratégie implémentée	33
3.1	Solution du problème	33
3.2	Recherche Tabou	34
3.2.1	Contexte historique	35
3.2.2	Espace de solution et voisinage	36
3.2.3	Tabous	37
3.2.4	Algorithme implémenté	37
3.3	Mouvements	40
3.3.1	LE	40
3.3.2	MLE	41
3.3.3	NNI	41
3.3.4	IELE	43
3.3.5	IEE	43
3.3.6	IE	45
3.4	Evaluation de la solution	46
3.4.1	SUCSESSEURS	46
3.4.2	FASTMTM	49
3.4.3	BINARYEDGES	51
4	Résultats	54
4.1	Données de tests	54
4.2	PAUP	54
4.3	Analyse	57
4.3.1	Recherche Taboue statique	58
4.3.2	Recherche Taboue dynamique	61
4.3.3	Comparaison recherche statique et recherche dynamique	64
4.3.4	Comparaison avec PAUP	66
4.4	Conclusion	68
4.5	Futurs travaux	68
	Bibliographie	70

Remerciements

Je tiens à remercier Jean Cardinal, qui a accepté d'être le directeur de ce mémoire alors que je lui en avais fais la demande assez tardivement (pour ne pas dire à la dernière minute), Michel Milinkovitch pour ce sujet qui je dois dire m'a plus qu'intéressé ainsi que Martine Labbé et Gianluca Bontempi pour avoir accepté de juger ce travail. Je n'oublie certainement pas Daniele Catanzaro pour sa disponibilité et sans qui ce mémoire n'aurait pu voir le jour.

Je tiens aussi à remercier mon équipe de relectrices de choc : Caroline Bollen, Sandrine Bouhy et Maria-Irene Chomel ainsi que leur assistant Benoît Buntinx.

Enfin je remercie mes parents pour leur soutien logistique.

Préface

Depuis le 19^{ème} siècle, des scientifiques de plusieurs disciplines essayent de relever un des plus grand défis de la biologie contemporaine : reconstruire l'histoire évolutive des espèces. Cette reconstruction se base sur les traces laissées par l'évolution, à savoir les caractères hérités d'une espèce à l'autre et les transformations au cours du temps. Tous les composants de vie, par exemple les protéines, ont des histoires spécifiques, ce qui est d'une grande aide pour comprendre leur fonction.

L'évolution des espèces est le plus souvent modélisée par un arbre dont les noeuds externes représentent les espèces contemporaines et les noeuds internes représentent les espèces ancestrales. Cette représentation est appelée *arbre d'évolution*, on parle alors de *phylogénie*.

La reconstruction phylogénétique est essentielle dans la compréhension des mécanismes évolutifs qui ont conduit à la diversité du monde vivant actuel. Il est impensable d'essayer de comprendre ou de localiser des changements de vitesse d'évolution sans une phylogénie préalable.

En 1963, Cavalli-Sforza et Edwards ont affirmé : " . . . si l'évolution d'un groupe de populations d'organismes vivants peut être traitée par des méthodes exactes, on devrait être capable de mesurer la quantité d'évolution qui sépare n'importe quelle paire de populations . . . ". Par la suite, le problème a fait l'objet de tant de recherches que beaucoup d'algorithmes pour déduire les phylogénies ont été développés. Certains algorithmes cherchent à travers toutes les solutions possibles, si nécessaire, pour en trouver une optimale. Mais leur demande en temps de calcul augmente tellement rapidement avec la taille du problème que ces algorithmes ne peuvent généralement pas résoudre des problèmes avec un grand nombre d'espèces. D'autres algorithmes obtiennent des solutions généralement non-

optimales, mais leur demande en temps de calcul est suffisamment raisonnable pour analyser des problèmes avec plusieurs centaines d'espèces.

Après plusieurs décennies d'efforts, le développement de nouveaux algorithmes plus efficaces reste un domaine de recherche actif.

L'objectif de ce mémoire est de développer une méthode basée sur une métaheuristique pour retrouver l'arbre phylogénétique inconnu d'un groupe d'espèces étudiées. Vu qu'il s'agit d'un problème NP-complet, on ne retrouvera jamais l'arbre phylogénétique exact pour un nombre relativement grand d'espèces considérées. Tout l'art de la méthode de reconstruction développée ici consiste à trouver l'arbre phylogénétique le plus vraisemblable possible en un temps raisonnable.

Ce mémoire est organisé en quatre parties distinctes. La première constitue une introduction sur le domaine de la reconstruction phylogénétique. Dans la deuxième partie, nous nous intéresserons à la modélisation du problème que nous allons résoudre par des méthodes élaborées dans la troisième partie. Cette dernière contient les différents apports de ce mémoire. Enfin la quatrième partie regroupera les différents résultats obtenus par ces méthodes ainsi qu'une comparaison avec certaines techniques existantes. Cette dernière partie fera aussi office de conclusion.

Chapitre 1

Introduction

1.1 La reconstruction phylogénétique

Le terme "phylogénie" fut utilisé pour la première fois par E. Haeckel en 1866 pour définir l'enchaînement des espèces animales et végétales au cours du temps, concept que recouvrait jusque là, le terme "généalogie". Ce terme fut ensuite repris dans la dernière édition de *l'Origine des espèces* de C. Darwin, pour désigner "*les liens généalogiques de tous les êtres organisés*".

La *théorie des mécanismes de l'évolution* que proposait C. Darwin dans cet ouvrage avait pour principal postulat la descendance avec modification. Cette théorie dût sa reconnaissance au fait qu'elle donnait une interprétation générale à des phénomènes déjà admis comme l'homologie (héritage d'un ancêtre commun) et l'ordre de la nature (classification des espèces du vivant en différentes catégories). A cette époque fleurissaient déjà de nombreuses généalogies et classifications des espèces animales et végétales.

La *théorie de l'évolution* ajouta une nouvelle dimension aux classifications : la dimension temporelle. Depuis cette époque, les arbres se sont imposés comme le support graphique naturel de la phylogénie, permettant de représenter simultanément les groupements d'espèces et la dimension temporelle.

L'étude des phylogénies s'insère dans le domaine de la biologie comparative, également appelée systématique, dont le but est de proposer une classification des organismes vivants afin de comprendre les causes de leur diversité. Le classement des espèces a d'abord été pratiqué sur base d'observations morphologiques, telles que la présence ou l'absence d'ailes, le nombre de pattes,

etc.

L'utilisation de données moléculaires remonte aux études immunologiques du début du siècle passé avec Nuttall [Nut04], mais les phylogénies moléculaires ne furent vraiment acceptées que dans les années 60, suite à la publication de phylogénies, obtenues depuis des séquences d'ADN ou protéïques, possédant de "bonnes" congruences avec les classifications morphologiques (Zuckerman et Pauling [ZP62], Fitch et Margoliash [FM67]).

La comparaison des espèces, en fonction des différences constatées dans leur génome, connut alors un engouement qui n'a pas cessé d'augmenter depuis, et ce pour une double raison. Premièrement, les séquences étant des enchaînements de constituants chimiques (les nucléotides *A, C, G* et *T*), la comparaison des séquences moléculaires est dépourvue d'ambiguïté, alors que les données morphologiques sont parfois sujettes à diverses interprétations rendant les observations subjectives et donc critiquables. Deuxièmement, les millions de nucléotides, qui composent le génome des organismes, constituent un réservoir de données quasiment inépuisable en comparaison des quelques centaines de caractères morphologiques qu'on utilisait précédemment pour reconstruire des phylogénies.

Dans les années 70 et 80, les techniques de séquençage de l'ADN furent considérablement améliorées ([SN77], [QMB83], [MF87]), augmentant de façon toujours exponentielle la quantité de données exploitables pour des analyses phylogénétiques. Ces données sont pour la grande majorité disponibles dans des banques de données accessibles à la communauté scientifique par l'intermédiaire du réseau Internet (*EMBL, GenBank, Swissprot*, etc).

Par exemple, la base de données génomique *GenBank* contient des informations sur environ 100,000 espèces. Plus de 4 millions d'espèces d'organismes ont été découvertes et y sont répertoriées, et il a été estimé que 10 millions d'espèces doivent encore être découvertes. Placer celles-ci dans ce qu'on appelle *L'Arbre de Vie* est un des problèmes les plus complexes et importants de la biologie.

Le projet de *L'Arbre de Vie* promet donc d'être un programme de recherche substantiel et international, impliquant des milliers de biologistes, d'informaticiens et de mathématiciens. Le but scientifique de ce projet est de comprendre les origines de la vie, l'orientation de son évolution, l'étendue de la biodiversité moderne, ainsi que sa vulnérabilité à des menaces possibles ou existantes. En effet, l'analyse phylogénétique joue un rôle majeur dans la découverte de nouvelles formes de vie. Par exemple, beaucoup de micro-organismes ne peuvent

être cultivés ou étudiés en laboratoire et donc le principal moyen de découverte est d'isoler leur ADN depuis des échantillons récupérés à partir de l'eau ou du sol.

Ces immenses quantités de données ont créé le besoin de disposer d'outils de traitement systématiques, tant pour organiser cette connaissance (stockage) que pour l'exploiter (inférence de phylogénies). C'est ce qui conduisit les informaticiens à s'intéresser au problème de la reconstruction phylogénétique. Ils rejoignaient dans ce domaine les biologistes mais aussi les mathématiciens qui étaient déjà intéressés par l'aspect analyse de données. En effet, depuis les données moléculaires ou morphologiques, on peut facilement évaluer des similitudes ou des dissimilitudes entre les espèces deux à deux. La reconstruction phylogénétique consiste alors en la représentation d'une matrice de dissimilarités (*i.e.*, de *distances évolutives*) par une structure arborée, problème bien connu en mathématique.

1.2 Les données traduisant l'évolution

Les données morphologiques et moléculaires dont on dispose initialement peuvent, la plupart du temps, être considérées comme des *caractères* statistiques discrets pouvant prendre plusieurs valeurs ou *états*. Dans le cas des données moléculaires, les caractères correspondent par exemple aux différents *sites nucléotidiques* de la séquence, dont les états possibles sont *A, C, G* et *T* (représentant respectivement l'adénine, la cytosine, la guanine et la thymine). Dans le cas de données morphologiques, les caractères sont déterminés par l'investigateur et peuvent être soit binaires (présence ou absence de dents) soit multivalués (nombres de dents). Chaque espèce est décrite par les valeurs qu'elle prend pour les différents caractères. D'un point de vue informatique, il s'agit d'une chaîne de caractères sur un alphabet limité (par exemple à 4 lettres pour les 4 bases chimiques de l'ADN ou à 20 lettres pour coder les acides aminés des protéines).

L'évolution des espèces est généralement considérée comme un processus divergeant au cours du temps, *i.e.*, faisant se scinder les espèces en plusieurs lignées à divers points du temps, et traduisant un certain nombre de modifications dans les états pris par les espèces pour les différents caractères. Quand

un changement d'état intervient pour un caractère, on parle de *substitution*. Ce sont ces changements d'état, indépendants d'une espèce à l'autre, qui contribuent à l'éloignement progressif des espèces les unes des autres. Ainsi, pour des données morphologiques, l'évolution se traduit comme l'héritage ou la transformation de certains états de caractères d'un ancêtre à ses descendants. Dans le cas de données moléculaires, la différenciation du génome des espèces résulte non seulement de substitutions (modifications de l'état d'un site), mais aussi d'*insertions* (ajouts de sites supplémentaires) et de *délétions* (suppressions de sites de la séquence). Ces événements peuvent se produire en divers endroits de la séquence, modifiant sa longueur et changeant la place relative des caractères initiaux. Pour pouvoir retrouver d'une espèce à l'autre quels nucléotides correspondent aux mêmes sites (*i.e.*, caractères) ancestraux, nous sommes obligés de passer par une étape d'*alignement* des séquences.

1.2.1 L'alignement des séquences moléculaires

Lorsque nous envisageons de reconstruire la phylogénie d'un groupe d'espèces sur base de données moléculaires, nous sélectionnons, dans un premier temps, une ou plusieurs portions bien précises de leur génome, comme des gènes. Une deuxième étape, indispensable, consiste à aligner les séquences obtenues pour les différentes espèces, *i.e.* à mettre en correspondance les séquences, nucléotide par nucléotide. Il est en effet fréquent que des bouts d'ADN divers se soient insérés (ou aient été supprimés) dans certaines des séquences, brouillant ainsi les correspondances initiales entre les sites des différentes séquences. Pour que la reconstruction de la phylogénie puisse être menée correctement, il est nécessaire d'identifier les parties des séquences qui correspondent à de tels événements. Prenons par exemple trois séquences :

$$S_1 = \text{AGAATAGCCA}$$

$$S_2 = \text{AGGATAGGA}$$

$$S_3 = \text{AGTATGGA}$$

un alignement possible est :

	0	1	2	3	4	5	6	7	8	9
S_1 :	A	G	A	A	T	A	G	C	C	A
S_2 :	A	G	G	A	T	A	G	G	.	A
S_3 :	A	G	T	A	T	.	G	G	.	A

Cet alignement s'explique ainsi : en position 0, 1, 3, 4, 6 et 9, aucun évènement mutational n'est survenu ; en position 2, il y a eu 2 substitutions ; en position 5, il y a eu délétion ; en position 7, une substitution ; en position 8, une insertion. Cet alignement peut aussi s'interpréter de manière fort différente : nous pouvons aussi supposer qu'en position 8, il y a eu deux délétions. Toutefois, l'interprétation précédente de cette position est la plus parcimonieuse, et c'est celle qu'on retiendra. De la même manière, d'autres alignements que celui proposé ci-dessus sont possibles pour les séquences S_1 , S_2 et S_3 . On tendra toujours vers l'alignement le plus parcimonieux.

D'un point de vue informatique, ce problème s'apparente à la recherche de la "distance d'édition" entre les séquences. La définition de critères permettant d'évaluer la qualité des alignements, comme la recherche du meilleur alignement au sens de ces critères, constitue tout un domaine de recherche, généralement séparé de la recherche phylogénétique. Trouver le meilleur alignement pour un ensemble de n séquences est un problème NP-difficile en général ([21]).

La correspondance, établie entre les nucléotides des différentes séquences lors de l'alignement, définit les caractères (correspondant aux positions) qui servent de données à la reconstruction phylogénétique. Dès lors, la justesse d'une phylogénie proposée sur base de données moléculaires dépend de la qualité de l'alignement. Le risque principal consiste à regrouper, sous un même caractère, des nucléotides provenant initialement de sites différents. Ceci explique que l'alignement soit une étape qui doit être particulièrement soignée, et pour laquelle l'expérience et la connaissance des biologistes sont primordiales. Ceux-ci utilisent généralement divers algorithmes pour proposer un premier alignement acceptable, qu'ils affinent ensuite "à la main", en se servant de logiciels interactifs. Ils veulent avant tout s'assurer que les états mis en correspondance soient homologues, *i.e.*, soient issus des mêmes sites pour toutes les espèces.

Dans les régions stables des séquences (*i.e.*, les régions dans lesquelles on peut proposer un alignement sans insertion-délétion), nous pouvons sans grand risque supposer que les nucléotides, regroupés sous une même position, correspondent bien à un même site de la séquence ancestrale. Aussi ces régions sont-elles privilégiées dans l'alignement : dans le petit alignement donné plus haut, on estimera sans doute que la région 0-4 est bien conservée, tandis que la région 5-9 ne l'est pas et doit être écartée.

Finalement, seuls les caractères pour lesquels l'alignement attribue un état à toutes les espèces sont conservés pour la recherche phylogénétique. C'est ce

qui constitue le paradoxe de l'alignement : plus on étudie d'espèces et moins on dispose de caractères pour inférer la phylogénie, car les caractères définis sur toutes les séquences sont de moins en moins nombreux.

Dans le cadre de ce mémoire, nous ne nous intéresserons qu'à la phase de reconstruction de la phylogénie et non à l'obtention des données qui la soutendent.

1.2.2 Le concept de similitude

Toute reconstruction phylogénétique est basée sur le concept de *similitude* : plus les espèces se ressemblent (par les états qu'elles prennent pour les différents caractères), plus leur degré de parenté est supposé important. On raisonne avant tout par *homologie*, *i.e.*, on suppose que les caractéristiques communes sont héritées d'un ancêtre commun. Par exemple pour un caractère donné, sur base de l'observation d'états *A* ou *C* selon les espèces, on déduira que le caractère était initialement dans l'état *A* puis est devenu *C* (ou inversement) à une certaine date, pour une certaine lignée. Suivant ce raisonnement, toutes les espèces pour lesquelles on observe l'état *C* (ou inversement l'état *A*) sont supposées appartenir à la lignée ayant subi la substitution, et sont supposées former un groupe séparé des autres espèces dans la phylogénie (cfr. figure 1.1).

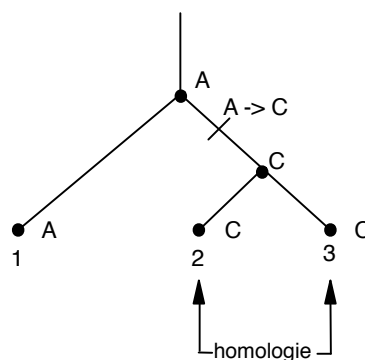


FIG. 1.1 – Ressemblance due à l'homologie

Au fur et à mesure que les séquences évoluent dans le temps, la probabilité qu'une substitution se produise pour un caractère ayant déjà supporté une substitution augmente. Quand cet évènement se produit, le premier changement

d'état peut être occulté (on ne dispose comme donnée que des séquences aux feuilles de la phylogénie et non d'un historique des changements d'états). On parle dans ce cas d'*homoplasie*, ou encore de *substitution cachée*. Les deux types d'homoplasies existantes sont les *réversions* (Figure 1.2-(a)) et les *convergences* (Figure 1.2-(b)).

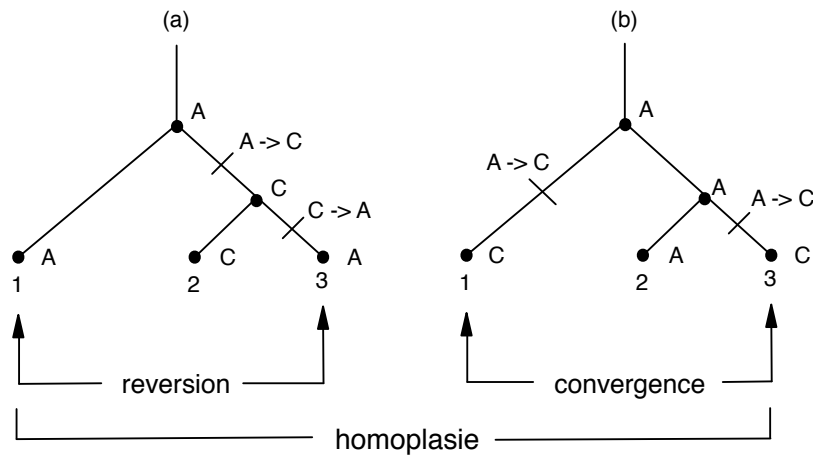


FIG. 1.2 – Ressemblance due à l'homoplasie suite à une réversion (a) ou à une convergence (b)

Les homoplasies contredisent l'hypothèse selon laquelle les similitudes entre espèces indiquent une forte parenté commune, et risquent ainsi de mettre en péril les méthodes de reconstructions phylogénétiques. Par exemple sur la Figure 1.2-(b), toute méthode raisonnable ne disposant que des états observés aux feuilles de la phylogénie pour le caractère représenté aurait faussement regroupé les espèces 1 et 3. Les homoplasies sont très généralement en minorité par rapport aux homologies mais, même en petite quantité, elles peuvent induire des méthodes en erreur.

1.3 Historique des méthodes de reconstruction phylogénétique

Avant les années 60, la reconstruction phylogénétique pouvait difficilement être considérée comme une science à part entière : les méthodes employées étaient le plus souvent obscures, tandis que le choix et l'interprétation des ca-

ractères retenus pour décrire les espèces étaient relativement subjectifs. Tout ceci rendait quasiment impossible la reproductibilité d'une classification par un savant autre que celui qui l'avait produite.

Dans le début des années 60, la connaissance de caractères moléculaires de plus en plus nombreux et peu soumis à la subjectivité, ainsi que l'avènement des premiers ordinateurs, correspondirent au désir des phylogénéticiens de préciser leurs méthodes et de se doter de bases conceptuelles explicites. Désormais, les chercheurs proposaient des méthodes de classification automatique, au sens où une classification devait résulter de l'application d'une certaine *méthode*, caractérisée par un algorithme précis.

De cette clarification méthodologique émergèrent deux écoles de pensée, la *phénétique* et le *cladisme*, chacune possédant ses propres méthodes pour classer les espèces. Ces deux courants étaient opposés par la signification qu'ils donnaient aux classifications des espèces, aussi le débat fut-il essentiellement philosophique et d'autant plus acharné que les arguments théoriques étaient en nombre réduit. En pratique, chacune de ces écoles appréhendait de façon différente le concept de similitude dont nous avons vu précédemment qu'il est à la base de toute classification.

1.3.1 La phénétique

Les principes de la première école furent exposés clairement par Sneath et Sokal dans leur ouvrage "*Principles of Numerical Taxonomy*" [SS63, SS73]. Les phénéticiens prônaient qu'une classification ne devait tendre à regrouper les espèces que dans le but de résumer leur similitudes et dissimilitudes. Pour eux il n'était initialement pas question qu'une classification représente les liens de parenté entre espèces. Ce n'était qu'accessoirement, en utilisant d'autres critères, que des inférences phylogénétiques pouvaient être tirées des regroupements d'espèces résultant de la classification.

Le concept de ressemblance ne pouvait être utilisé scientifiquement que du point de vue des similitudes globales entre espèces, obtenues sur base d'un nombre aussi large que possible de caractères. Face à la critique selon laquelle les homologues et les homoplasies se retrouvaient ainsi mêlées, les adeptes de cette école répondaient que si on disposait de suffisamment de caractères, la "vraie" ressemblance (homologie) prenait le pas sur la "fausse" ressemblance (homoplasie). Simplement résumée, cette approche se fonde sur l'analyse quantitative du plus grand nombre de caractères disponibles suivant le principe de

“ce qui se ressemble s’assemble”.

Construisant des classifications sur la base des similitudes globales, les phénéticiens avaient essentiellement recours à des méthodes phénétiques, *i.e.*, utilisant au départ une matrice de similarités, ou de dissimilarités, entre les espèces deux à deux. Des algorithmes, dits de *cluster analysis*, regroupaient en un *taxon* (une classe) les espèces les plus similaires, puis itéraient le processus jusqu’à obtention d’une classification hiérarchique complète des espèces étudiées. Parmi les méthodes les plus célèbres, citons la méthode du lien moyen (UPGMA) et ses variantes.

1.3.2 Le cladisme

Les cladistes furent opposés aux phénéticiens par l’idée qu’une classification des espèces devait avant tout élucider les relations de parenté entre espèces en retrouvant le scénario évolutif selon lequel elles ont divergé au cours du temps. Cette école de pensée fut fondée par Hennig [Hen50, Hen66] et son ouvrage “*Phylogenetic systematics*”.

Pour les cladistes, la similitude globale seule ne constitue pas une base saine en raison des “fausses” similitudes que sont les homoplasies (convergences et réversions). Seule une analyse qualitative des caractères, les partitionnant en homologies et homoplasies, peut permettre de reconstruire correctement la phylogénie. Toutefois, le concept même d’homologie doit être raffiné en distinguant les états primitifs (plésiomorphes) et les états dérivés (apomorphes) des caractères homologues. Pour les cladistes, seul le partage par plusieurs espèces d’états dérivés (les synapomorphies) indique une parenté commune et peut permettre de retrouver les *clades*, *i.e.*, les groupes d’espèces appartenant à une même lignée. Le concept de similitude se trouve donc ici partitionné en trois catégories, dont une seule est vraiment informative.

Les méthodes de reconstruction employées par les cladistes reposent donc sur l’analyse séparée des caractères. Comme ils s’intéressent à la polarité des caractères (distinction entre état primitif et état dérivé), les classifications proposées sont enracinées. Seulement l’état primitif des caractères n’est pas toujours connu (comme pour les caractères moléculaires), ce qui conduit à envisager tous les états ancestraux possibles pour chaque caractère. Selon l’état ancêtre choisi pour chaque caractère et selon la structure de la classification, le nombre de changements d’état, nécessaires pour expliquer les états observés aux espèces, varie. La solution retenue est celle demandant le moins de changements d’état.

Il s'agit de la méthode du *maximum de parcimonie*, consistant à proposer pour la phylogénie des espèces, le scénario évolutif le plus économe. Pour les cladistes, un avantage non-négligeable de la parcimonie (et d'autres méthodes basées sur les caractères) est d'indiquer les états pris par les espèces ancestrales (les noeuds internes de la phylogénie) pour les différents caractères.

1.3.3 La hiérarchie actuelle des méthodes

Le point de vue adopté vis-à-vis du modèle de l'évolution divise donc les méthodes actuelles de reconstruction phylogénétique en trois catégories :

- les *méthodes cladistes* (ou de caractères), essentiellement des variantes de la méthode du maximum de parcimonie, pour lesquelles aucun modèle explicite de l'évolution n'est posé. Comme le notent Swofford *et al* [SOW96], ceci ne signifie pas que ces méthodes soient libres de toute hypothèse, par exemple, elles nécessitent que les homoplasies soient peu fréquentes en comparaison des homologies.
- les *méthodes de distances*, corrigeant dans un premier temps, selon un modèle de l'évolution explicite, les distances globales observées entre espèces, puis inférant une phylogénie suivant un principe algorithmique basé sur les distances. On peut encore parler ici de méthodes "phénétiques", puisque pour les phylogénéticiens ce terme recouvre toute méthode basée sur une mesure de (di)similitude globale entre espèces. Toutefois, pour les généticiens, ce terme a un sens légèrement différent : il désigne toute méthode basée sur la (di)similitude *observée* entre les espèces et ne recouvre donc pas les méthodes utilisant des distances entre espèces *corrigées* par un modèle de l'évolution. Pour éviter toute confusion, nous utiliserons l'appellation "méthodes de distances".
- les *méthodes probabilistes*, ayant elles aussi recours à un modèle de l'évolution explicite, mais inférant une phylogénie sur base de l'analyse individuelle des caractères et non des distances globales entre espèces. Le modèle mathématique de l'évolution permet d'obtenir les probabilités d'occurrence des différentes substitutions (passage d'un certain état à un autre). Ces probabilités permettent d'évaluer la vraisemblance de chaque phylogénie possible pour les espèces en fonction des caractères observés, et de choisir la phylogénie la plus vraisemblable. Cette approche fut introduite par Felsenstein [Fel81], puis reprise par d'autres auteurs [Sai88,Sai90] ; des variantes étant aussi proposées [Hen91,HPS94].

1.4 Les arbres phylogénétiques

Dans les études phylogénétiques, les relations d'évolution parmi un groupe d'organismes sont illustrées par les *arbres phylogénétiques* ou phylogénie. Un arbre phylogénétique est un graphe composé de noeuds et de branches, dans lequel seulement une branche connecte n'importe quelle paire de noeuds adjacents. Les noeuds représentent les unités taxonomiques tandis que les branches définissent les relations parmi les unités en terme de généalogie. La configuration de branchements d'un arbre est appelée la *topologie*. La longueur des branches représente généralement le nombre de changements évolutifs effectués dans cette branche. Les unités taxonomiques représentées par des noeuds peuvent être des espèces, des populations, des individus ou des gènes.

Nous pouvons distinguer deux types de noeuds : les noeuds externes (ou feuilles) et les noeuds internes. Les noeuds externes représentent les unités taxonomiques que l'on cherche à comparer et sont attribuées comme étant les unités taxonomiques opérationnelles (UTOs). Les noeuds internes représentent les ancêtres communs.

Les branches d'un arbre, ou *arêtes*, peuvent être différenciées comme étant soit internes, soit externes. Les branches externes sont aussi appelées branches périphériques.

Un arbre est dit additif si les distances entre n'importe quelle paire d'UTOs équivaut à la somme des longueur de toutes les branches les séparant. La distance entre deux UTOs est calculée directement depuis les données moléculaires (comme les séquences ADN), tandis que les longueurs des branches sont évaluées à partir des distances entre les UTOs selon certaines règles.

Les arbres phylogénétiques peuvent être enracinés ou non enracinés. Dans un arbre enraciné, la direction de chaque chemin correspond au temps d'évolution et la racine est l'ancêtre commun de tous les UTOs qui sont étudiés. Dans un arbre non enraciné, on spécifie les relations entre les UTOs mais on ne définit pas de chemin d'évolution. Ces arbres ne font pas de suppositions ou ne requièrent pas de connaissances sur les ancêtres communs.

Par la suite, nous n'allons considérer que les arbres non enracinés. Le nombre d'arbres bifurquants non enracinés (N_U) pour n UTOs ($n \geq 3$) est donné par :

$$N_U = \frac{(2n-5)!}{2^{n-3}(n-3)!} = \prod_{i=3}^n (2i-5) \quad (1.1)$$

Nous voyons que N_U croît exponentiellement en fonction de n . En effet, pour 14 UTOs, il existe approximativement 32 milliards d'arbres différents et 3×10^{84} arbres (*i.e.* plus que le nombre d'atomes dans l'univers connu) pour 55 UTOs. Sachant qu'un seul de ces arbres représente correctement les vraies relations d'évolutions parmi les UTOs, il est généralement très difficile de déduire l'arbre phylogénétique correcte, surtout quand n est relativement grand.

1.5 Méthodes de reconstruction d'arbres phylogénétiques

Comme nous venons de le voir, inférer un arbre phylogénétique est une réelle procédure d'estimation : nous essayons de fournir la meilleure estimation d'une histoire évolutive pour laquelle nous ne disposons que de données incomplètes. Les méthodes de reconstruction d'arbres phylogénétiques sont autant d'estimateurs possibles pour choisir un arbre phylogénétique parmi un nombre exponentiel de possibilités (cfr. équation 1.1).

La plupart des méthodes de reconstruction d'arbres phylogénétiques peuvent être vues comme des procédures d'optimisation, chacune étant caractérisée par deux points :

- le critère d'optimalité (aussi appelé *critère de reconstruction* ou fonction objective) qu'elle considère pour évaluer les différentes phylogénies possibles. Ceci permet ainsi potentiellement de classer ces arbres phylogénétiques par ordre de préférence pour estimer l'arbre phylogénétique inconnu.
- l'algorithme qu'elle emploie pour rechercher un arbre phylogénétique optimal au sens du critère choisi. Bien souvent cet algorithme ne considère pas tous les arbres phylogénétiques possibles, auquel cas il est dit heuristique ; autrement il est dit exact.

Les méthodes de reconstruction phylogénétique sont classées suivant les données utilisées pour la reconstruction, à savoir les caractères ou les distances. Nous allons présenter quelques méthodes caractéristiques de reconstruction phylogénétique pour chacun des deux principes de base.

1.5.1 Méthodes de caractères

Un caractère discret fournit des informations sur une séquence donnée. Par exemple, pour une séquence ADN donnée, une base à une position spécifique de la séquence représente un caractère discret.

...AATTGCATGATGGGGCCCTATTTGCAAAA...

Nous pouvons définir un caractère discret comme une variable indépendante dont les valeurs possibles sont une collection d'état de caractères mutuellement exclusifs. L'hypothèse d'indépendance parmi les caractères est commune à la plupart des méthodes d'analyse basées sur les caractères. Quand nous ne pouvons pas supposer l'indépendance entre caractères, nous sommes forcés de prendre en compte les covariances parmi ceux-ci, ce qui rend les méthodes beaucoup plus compliquées. De plus, la supposition d'indépendance nous permet de traiter chaque position séparativement, permettant aux problèmes d'être sous-divisés en un nombre de plusieurs problèmes plus simples.

Une seconde supposition est la suivante : les caractères doivent être homologues. Par homologue, nous voulons dire qu'un caractère doit être défini de telle façon que tous les états observés sur certains taxons pour un caractère particulier doit avoir été dérivé, peut-être avec modification, d'un état correspondant observé dans un ancêtre commun de ces taxons.

Maximum de parcimonie

Le critère d'optimalité du Maximum de parcimonie (MP) est basé sur le principe que l'estimation la plus plausible de l'histoire des espèces est l'arbre phylogénétique qui correspond au scénario le plus économe en terme de changements évolutifs. D'un point de vue pratique, on va chercher l'arbre qui implique le nombre minimum de changements d'états nécessaires pour expliquer les séquences associées à ses différents noeuds.

Par exemple dans la figure 1.3, on observe une différence entre les séquences S_0 et S_1 , une différence entre S_1 et S_{11} , etc., si bien que la valeur de parcimonie de cet arbre phylogénétique est 6. Dit autrement, 6 changements d'états seulement suffisent à expliquer les différences entre les séquences (par exemple, le premier caractère s'est transformé de A en G entre S_0 et S_1) ; bien sûr d'autres explications sont possibles : entre S_0 et S_1 , le premier caractère peut très bien avoir évolué

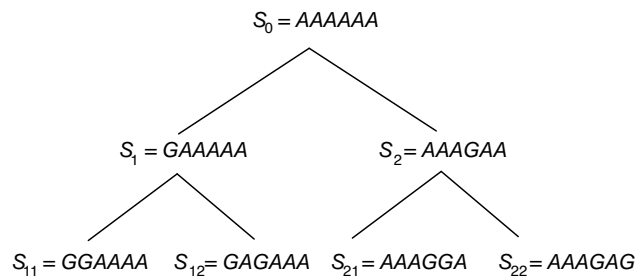


FIG. 1.3 – Arbre phylogénétique d'un ensemble de séquences.

de A en C puis G, demandant ainsi deux changements. Cependant, le principe de parcimonie consiste à toujours choisir l'explication faisant le moins d'hypothèses. On considère donc qu'un seul changement a eu lieu pour le premier caractère et qu'aucun changement n'a affecté les autres caractères entre ces deux séquences, si bien que l'arête séparant S_0 et S_1 se voit attribuer un coût de 1.

Il y a plusieurs variantes du MP qui diffèrent selon les directions permises par le changement d'état des caractères.

Maximum de vraisemblance

Le Maximum de vraisemblance (MV) recherche le modèle évolutif qui produit les données observées avec la plus grande vraisemblance. La vraisemblance est calculée en fonction de la probabilité que le modèle de variation d'un site va être produit par un processus particulier de substitution, à partir d'un arbre particulier et des fréquences de bases observées complètes. La vraisemblance devient la somme des probabilités pour chaque reconstruction possible de substitution sous des processus de substitutions particuliers. Les vraisemblances de tous les sites sont multipliées pour donner une vraisemblance complète de l'arbre (*i.e.*, la probabilité d'observer les données étant donné l'arbre et le processus de substitution).

Dû au fait que le MV utilise d'énormes quantités de temps de calcul, il est généralement infaisable d'effectuer une recherche complète qui optimise simultanément le modèle de substitution et l'arbre pour un ensemble défini de données.

1.5.2 Méthodes de matrice de distance

Les méthodes de matrice de distance ont été introduites par Cavalli-Sforza et Edwards (1967) et par Fitch (1967 ; voir aussi Horne, 1967). L'idée générale ne semble à priori pas fonctionner : on calcule les distances entre chaque paire d'espèces et ensuite on trouve un arbre qui se rapproche le plus possible de l'ensemble de distances observées. Cette méthode omet les informations de plus haut ordre des états de caractères, ce qui réduit la matrice de données à une simple table de paire de distances. On peut penser que cela nous fera perdre tellement de subtilités d'informations que cela ne permettra pas d'effectuer une estimation raisonnable de l'arbre phylogénétique.

Cependant des études de simulations ont montré que la quantité d'information phylogénétique perdue pendant ce procédé est relativement peu élevée.

La meilleure façon d'appréhender la méthode de matrice de distance est de considérer, pour une paire d'espèces donnée, la distance comme une estimation de la longueur de la branche les séparant. Chaque distance déduit le meilleur arbre non enraciné pour cette paire d'espèces. Nous avons donc un large panel d'arbres (estimés) pour deux espèces, et nous essayons de trouver l'arbre à n espèces que ces derniers impliquent. La difficulté en agissant de la sorte est que les distances individuelles entre deux espèces ne seront pas exactement les longueurs des chemins entre ces deux espèces dans un arbre complet de n espèces.

UPGMA

UPGMA (Un-weighted Pair Group Method With Arithmetic Mean) est un algorithme de clusterisation dans lequel la reconstruction se fait pas à pas : on joint les deux séquences les plus proches selon le critère de la plus grande similarité et ainsi de suite entre les séquences restantes et les moyennes des paires jointes. L'inconvénient majeur est la sensibilité de la méthode à des taux de mutations différents sur les différentes branches. UPGMA n'est donc fiable que dans le cas de séquences peu divergentes.

Neighbor Joining

Cette méthode développée par Saitou et Nei (1987) tente de corriger la méthode UPGMA afin d'autoriser un taux de mutation différent sur les branches. Les données initiales permettent de construire une matrice qui donne un arbre en étoile (figure 1.4). Cette matrice de distances est ensuite corrigée afin de prendre en compte la divergence moyenne de chacune des séquences avec les autres. L'arbre est alors reconstruit en reliant les séquences les plus proches dans cette nouvelle matrice. Lorsque deux séquences sont liées, le noeud représentant leur ancêtre commun est ajouté à l'arbre tandis que les deux feuilles sont enlevées. Ce processus convertit l'ancêtre commun en un noeud terminal dans un arbre de taille réduite.

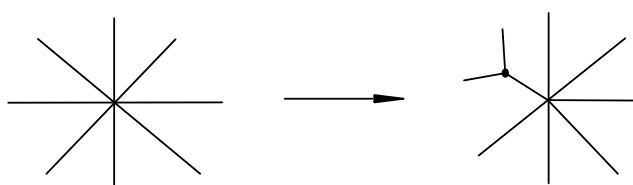


FIG. 1.4 – Décomposition en étoile. Les feuilles les plus similaires sont jointes et une branche est insérée entre elles et l'étoile restante. Le processus est répété jusqu'à avoir un arbre binaire.

Evolution minimum

Le principe de l'évolution minimum est un des critères d'optimalité les plus couramment utilisés pour l'inférence des arbres phylogénétiques (Kidd et Sgaramella 1971, Saitou et Nei 1987, Rzhetsky et Nei 1993, Swofford *et al.* 1996). Le principe de l'EM aspire à trouver la topologie de l'arbre caractérisée par une distance minimale. Introduite au départ par Kidd et Sgaramella-Zonta, ses justifications biologiques sont basées sur le fait que si les distances utilisées sont non-biaisées ou en d'autres termes, qu'il s'agisse estimations plus ou moins exactes des vraies distances, alors l'arbre phylogénétique le plus court est en espérance l'arbre phylogénétique correct si les longueurs d'arêtes sont évaluées aux moindres carrés ([23], [24]).

Chapitre 2

Modélisation du problème

2.1 Rappels

2.1.1 Les graphes

Un graphe G est une paire ordonnée (V, E) en un ensemble non-vidé V de *noeuds* et d'un ensemble E d'*arêtes*, chacune d'elle étant un élément de $\{\{x, y\} : x, y \in V\}$. Nous allons par ailleurs considérer que tous les graphes ont un ensemble fini de noeuds. Une arête qui joint un noeud à lui-même est une *boucle* et les arêtes qui joignent la même paire distincte de sommets sont appelées des *arêtes parallèles*.

Un *graphe simple* est un graphe qui ne contient ni boucle, ni arêtes parallèles. Si $e = \{u, v\}$ est une arête du graphe G , alors u et v sont *adjacents* ou *voisins*, et e est dit *incident* avec u et v . Les noeuds u et v sont les *extrémités* de e .

Soit v un noeud d'un graphe G . Le *degré* de v , noté $d(v)$, est le nombre d'arêtes dans G qui sont incidentes avec v . Comme chaque arête d'un graphe est incident avec exactement deux noeuds, il suit que, en sommant tous les degrés des noeuds d'un graphe, nous comptons chaque arête exactement deux fois. Cette observation donne l'égalité suivante au Lemme 1 :

Lemme 1 Soit $G = (V, E)$ étant un graphe. Alors :

$$\sum_{v \in V} d(v) = 2|E|$$

Un graphe H est un *sous-graphe* d'un graphe G si $V(H)$ et $E(H)$ sont des sous-ensembles de $V(G)$ et de $E(G)$ respectivement.

Un *chemin* dans un graphe G est une séquence d'arêtes distinctes v_1, v_2, \dots, v_k telle que, pour tout $i \in \{1, 2, 3, \dots, k-1\}$, v_i et v_{i+1} sont adjacents. Si de plus, v_1 et v_k sont adjacents, alors le sous-graphe de G dont l'ensemble de noeuds est $\{v_1, v_2, \dots, v_k\}$ et dont l'ensemble des arêtes est $\{(v_k, v_1)\} \cup \{(v_i, v_{i+1}) : i \in \{1, \dots, k-1\}\}$ est un *cycle*. Un graphe G est dit *connexe* si chaque paire de noeud dans G peut être jointe par un chemin.

2.1.2 Les arbres

Un *arbre* $T = (V, E)$ est un graphe connexe sans cycle (ou *acyclique*). Un noeud de T de degré d'au plus un est appelé une *feuille*. Un noeud de T qui n'est pas une feuille est appelé un *noeud interne*. Une arête de T est *interne* si chacune de ses extrémités est un noeud intérieur. Nous notons l'ensemble de noeuds intérieurs et d'arêtes intérieures de T par \mathring{V} et \mathring{E} respectivement.

Un arbre est *binnaire* si chaque noeud interne est de degré trois. Les arbres binaires sont parfois appelés *cubiques*, *ternaires* ou *trivalents* mais nous allons utiliser le mot "binnaire" pour garder la terminologie proche de la biologie.

Théorème 1 ([26] p.7) Soit $G = (V, E)$ un graphe. Les 3 propositions suivantes sont équivalentes :

1. G est un arbre
2. pour n'importe quels noeuds v_1 et v_2 dans V il existe un chemin unique de v_1 à v_2
3. G est connecté et $|V| = |E| + 1$

En combinant le Lemme 1 et l'équivalence entre la proposition 1 et 3 du Théorème 1, nous avons le Corollaire suivant :

Corollaire 1 Soit T un arbre avec l'ensemble de noeuds V , Nous avons alors :

$$\sum_{v \in V} d(v) = 2|V| - 2$$

Théorème 2 ([26] p.8) Soit $T = (V, E)$ un arbre avec $V \geq 3$, et soit n_1 et n_2 le nombre de noeuds de T de degré un et de degré deux respectivement. Alors

1. $|\mathring{E}| \leq n_1 + n_2 - 3$
2. Supposons que $n_2 = 0$. Alors $|\mathring{E}| = n_1 - 3$ si et seulement si T est binnaire.

2.2 Modélisation des arbres phylogénétiques

Les arbres phylogénétiques fournissent une représentation graphique standard des relations évolutionnistes en biologie. Cependant, d'un point de vue mathématique, il est naturel de considérer une classe d'objet légèrement plus générale appelée "X-trees". Brièvement, un X-tree est un arbre fini dans lequel certains noeuds (incluant les noeuds de degré un ou deux) sont labellés par des sous-ensembles disjoints d'un ensemble X . Un arbre phylogénétique est un X-tree pour lequel seules les feuilles de l'arbre sont labellées et par des sous-ensembles de singletons de X .

Définition 1 Un X-tree \mathcal{T} est une paire ordonnée $(T; \phi)$, où T est un arbre avec un ensemble de noeuds V et $\phi : X \rightarrow V$ est une translation avec la propriété que, pour chaque $v \in V$ de degré au plus deux, $v \in \phi(X)$. Un X-tree est aussi appelé un arbre semi-labellé (sur X). La figure 2.1 montre un X-tree où $X = \{1, 2, \dots, 8\}$.

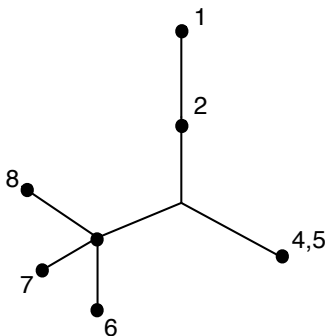


FIG. 2.1 – Un X-tree.

Soit \mathcal{T} un arbre semi-labellé (T, ϕ) . L'arbre T est appelé l'arbre fondamental de \mathcal{T} et la translation ϕ est appelée la translation de label de \mathcal{T} . Le domaine de ϕ est appelé l'ensemble de labels de \mathcal{T} et est noté $\mathcal{L}(\mathcal{T})$.

Définition 2 Un arbre phylogénétique \mathcal{T} est un X-tree $(T; \phi)$ avec la propriété que ϕ est une bijection de X vers l'ensemble des feuilles de T . Si de plus, chaque noeud interne de T est de degré trois, \mathcal{T} est un arbre phylogénétique binaire.

Si $\mathcal{T} = (T; \phi)$ est un arbre phylogénétique, nous pouvons voir X comme étant l'ensemble de feuilles de l'arbre fondamental T , et donc nous dénotons les

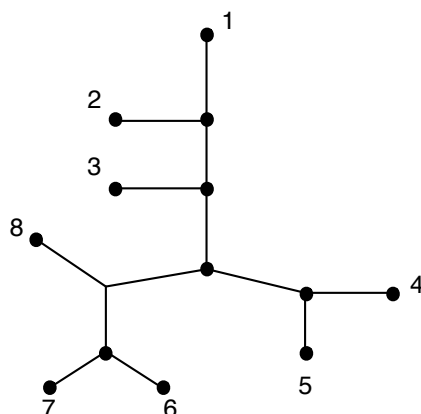


FIG. 2.2 – Un arbre phylogénétique.

feuilles de T par les éléments de X vu que ϕ est implicitement déterminé. La figure 2.2 montre un arbre phylogénétique sur $\{1, 2, \dots, 8\}$.

Deux propriétés intéressantes des arbres phylogénétiques binaires sont établis par les deux Théorèmes suivants. Le premier découle directement du Théorème 2.

Théorème 3 ([26], p.17) Soit \mathcal{T} un arbre phylogénétique binaire et soit $n = |X|$. Alors, pour tout $n \geq 2$, \mathcal{T} possède $2n - 3$ arêtes dont $n - 3$ arêtes internes.

Soit $B(n)$ la collection de tous les arbres phylogénétiques binaires avec comme ensemble de labels $\{1, 2, 3, \dots, n\}$ et soit $b(n) = |B(n)|$. Si $n \in \{1, 2\}$, alors $b(n) = 1$. Pour tout $n \geq 3$, le Théorème suivant fournit une formule très connue pour $b(n)$, un résultat qui date de Schröder (1870).

Théorème 4 ([26], p.17) Pour tout $n \geq 3$,

$$b(n) = 1 \times 3 \times 5 \times \dots \times (2n - 5) = \frac{(2n - 4)!}{(n - 2)! 2^{n-2}}$$

La preuve de ce Théorème se fait par induction sur n . Vu que $b(3) = 1$, le résultat est vérifié pour $n = 3$. Supposons que le résultat se vérifie pour $k = n - 1$, où $k \geq 3$. Soit $\psi : B(k + 1) \rightarrow B(k)$ la translation définie en définissant $\psi(T)$ comme étant l'arbre phylogénétique binaire dans $B(k)$ qui est obtenu depuis \mathcal{T} en effaçant la feuille labellée par $k + 1$ et ses arêtes incidentes, et donc en supprimant l'arête de degré deux résultante. Il suit par le Théorème 3 que

chaque arbre phylogénétique binaire dans $B(k)$ est l'image des $2k - 3$ arbres dans $B(k + 1)$. Donc, par hypothèse d'induction, $b(k + 1) = (2k - 3)b(k)$, et donc

$$b(n) = 1 \times 3 \times 5 \times \cdots \times (2n - 5),$$

comme exigé. Un examen plus approfondi montre que cette dernière expression est aussi égale à

$$\frac{(2n - 4)!}{(n - 2)! 2^{n-2}}$$

Rappelons que m est un entier quelconque, $m!!$ représente le produit $m \times (m - 2) \times (m - 4) \times \cdots \times 1$. En utilisant cette notation, nous avons : $b(n) = (2n - 5)!!$. La fonction $b(n)$ croît très rapidement avec n comme illustré par le fait que $b(10) = 2\,027\,025$ et $b(20) \approx 2 \times 10^{20}$. De plus, en appliquant la formule de Stirling à la deuxième formule pour $b(n)$ au Théorème 4, nous avons les équivalences asymptotiques

$$b(n) \sim \frac{1}{\sqrt{\pi}} 2^{n-2} n! n^{-5/2} \sim \frac{1}{2\sqrt{2}} \left(\frac{2}{e}\right)^n n^{n-2}$$

Le taux de croissance du nombre d'arbres phylogénétiques est un problème important pour la reconstruction de tels arbres à partir de types de données variés. Souvent, nous souhaitons sélectionner un "meilleur" arbre sous certains critères et ceci est clairement impossible en cherchant parmi tous les arbres quand n est relativement grand. Ceci est probablement une des raisons majeures pour développer des nouvelles méthodes mathématiques dont le but est de trouver des arbres optimaux (ou proches de l'optimalité) sans avoir à supporter une recherche à travers tous les arbres possibles.

2.3 Définition du problème

Nous allons maintenant modéliser le problème de l'inférence phylogénétique, un des problèmes les plus importants dans le domaine de la biologie moléculaire, sous le principe de l'Evolution Minimum.

Soit $\mathcal{T} = (T; \phi)$ un arbre phylogénétique binaire avec $T = (V, E)$ où $V = X \cup I$ est l'ensemble des noeuds et E l'ensemble des arêtes d'un arbre. X est, comme défini précédemment, l'ensemble des noeuds externes ou feuilles et I est l'ensemble des noeuds internes.

Un arbre phylogénétique peut être représenté par une matrice d'incidence M où chaque élément m_{ijk} est défini comme suit :

$$m_{ijk} = \begin{cases} 1 & \text{si l'arête } k \text{ existe entre le noeud } i \text{ et le noeud } j \\ 0 & \text{sinon} \end{cases}$$

M représente explicitement la topologie de l'arbre, c'est pour cette raison que nous allons l'appeler par la suite la **matrice topologique**. M est une matrice binaire $n(n-1)/2 \times 2n-3$.

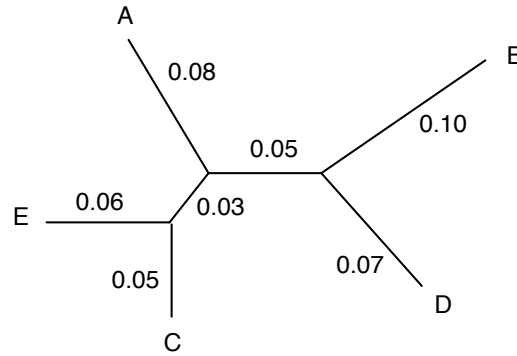
Comme nous venons de le voir, étant donné un ensemble de taxon Γ tel que $|\Gamma| = n$, le nombre possible de topologies d'arbre distinctes est $2n-5!!$ où $n!!$ est la double factorielle de n , ce qui implique que le nombre de matrices topologiques grandit exponentiellement avec le nombre d'espèces considérées.

Pour évaluer la longueur des branches d'un arbre phylogénétique, nous avons vu au chapitre précédent qu'il existait deux grandes familles de méthodes de reconstruction : les méthodes de caractères et les méthodes de matrice de distances. Nous allons plutôt nous intéresser à la deuxième famille.

L'idée fondamentale d'une méthode de matrice de distances est que nous avons une table (matrice) donnée de distances évolutives (D_{ij}), et que n'importe quel arbre particulier dont nous avons estimé les longueurs des branches mène à un ensemble de distances estimées entre les espèces (que nous allons noter d_{ij}).

Définition 3 Soit D la matrice de distance dont on dispose. Le critère de **l'évolution minimum** consiste en la recherche de l'arbre phylogénétique dont les longueurs d'arêtes sont évaluées aux moindres carrés vis-à-vis de D , et qui soit le plus court possible, au sens de la somme des longueurs des arêtes.

Pour trouver la distance estimée entre deux espèces, on va additionner les longueurs des branches entre ces deux espèces.



	A	B	C	D	E
A	0	0.23	0.16	0.20	0.17
B	0.23	0	0.23	0.17	0.24
C	0.16	0.23	0	0.15	0.11
D	0.20	0.17	0.15	0	0.21
E	0.17	0.24	0.11	0.21	0

FIG. 2.3 – Un arbre et les distances que celui-ci prédit. Celles-ci sont générées en additionnant les longueurs des branches entre chaque pair d'espèces.

La mesure utilisée par la méthode des moindres carrés est ([13] p.148) :

$$Q = \sum_{i=1}^n \sum_{j=1}^n w_{ij} (D_{ij} - d_{ij})^2 \quad (2.1)$$

Où les w_{ij} sont les poids qui diffèrent entre les différentes méthodes de moindres carrés. Cavalli-Sforza et Edwards (1967) ont défini les moindres carrés non pondérés où $w_{ij} = 1$. Fitch et Margoliash (1967) ont utilisé $w_{ij} = 1/D_{ij}^2$, et Beyer et al. (1974) ont suggéré $w_{ij} = 1/D_{ij}$.

Pour trouver les longueurs des branches d'une topologie donnée en utilisant les moindres carrés, nous devons minimiser Q . Une façon de faire est de résoudre un ensemble d'équations linéaires. Celle-ci sont obtenues en prenant les dérivées de Q en respectant les longueurs des branches, et en les égalisant à zéro. La solution des équations résultantes va minimiser Q .

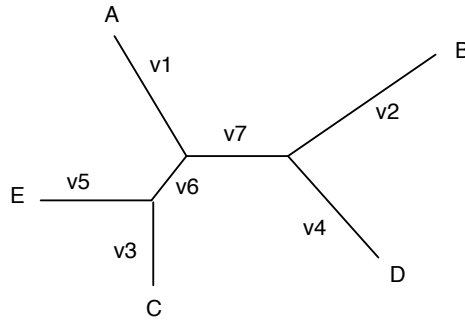


FIG. 2.4 – Le même arbre que la figure précédente, avec les longueurs de branches comme variables.

Dans l'équation 2.1 les d_{ij} sont les sommes des longueurs des branches. La figure 2.4 représente le même arbre mais avec des variables comme longueurs de branches. Si les espèces sont numérotées par ordre alphabétique, d_{14} va être la distance entre A et D, qui est $v_1 + v_7 + v_4$. La distance entre les espèces B et E est $v_{2,5} = v_5 + v_6 + v_7 + v_2$.

Supposons que nous avons numéroté toutes les branches de l'arbre et introduisons un indicateur de variable $m_{ij,k}$ défini de la même façon que les éléments de la matrice topologique. La distance entre i et j vaudra donc

$$d_{ij} = \sum_k m_{ij,k} v_k \quad (2.2)$$

L'équation 2.1 devient donc

$$Q = \sum_{i=1}^n \sum_{j:j \neq i} w_{ij} (D_{ij} - \sum_k m_{ij,k} v_k)^2 \quad (2.3)$$

Si nous dérivons Q par rapport à un v tel que v_k , et égalisons la dérivée à 0, nous avons l'équation

$$\frac{dQ}{dv_k} = -2 \sum_{i=1}^n \sum_{j:j \neq i} w_{ij} m_{ij,k} (D_{ij} - \sum_k m_{ij,k} v_k) = 0 \quad (2.4)$$

Le -2 peut être négligé.

Une façon d'estimer la longueur des branches par les moindres carrés est de résoudre cet ensemble d'équations linéaires. Pour l'arbre des figures 2.3 et 2.4, les équations sont :

$$D_{AB} + D_{AC} + D_{AD} + D_{AE} = 4v_1 + v_2 + v_3 + v_4 + v_5 + 2v_6 + 2v_7$$

$$D_{AB} + D_{BC} + D_{BD} + D_{BE} = v_1 + 4v_2 + v_3 + v_4 + v_5 + 2v_6 + 3v_7$$

$$D_{AC} + D_{BC} + D_{CD} + D_{CE} = v_1 + v_2 + 4v_3 + v_4 + v_5 + 3v_6 + 2v_7$$

$$D_{AD} + D_{BD} + D_{CD} + D_{DE} = v_1 + v_2 + v_3 + 4v_4 + v_5 + 2v_6 + 3v_7$$

$$D_{AE} + D_{BE} + D_{CE} + D_{DE} = v_1 + v_2 + v_3 + v_4 + 4v_5 + 3v_6 + 2v_7$$

$$D_{AC} + D_{AE} + D_{BC} + D_{BE} + D_{CD} + D_{DE} = 2v_1 + 2v_2 + 3v_3 + 2v_4 + 3v_5 + 6v_6 + 4v_7$$

$$D_{AB} + D_{AD} + D_{BC} + D_{CD} + D_{BE} + D_{DE} = 2v_1 + 3v_2 + 2v_3 + 3v_4 + 2v_5 + 4v_6 + 6v_7 \quad (2.5)$$

Supposons maintenant que nous empilons les D_{ij} , en ordre alphabétique, dans un vecteur,

$$\mathbf{d} = \begin{bmatrix} D_{AB} \\ D_{AC} \\ D_{AD} \\ D_{AE} \\ D_{BC} \\ D_{BD} \\ D_{BE} \\ D_{CD} \\ D_{CE} \\ D_{DE} \end{bmatrix}$$

Les coefficients $m_{ij,k}$ sont en fait les $m_{ij,k}$ de la matrice topologique ; chaque ligne correspondant aux D_{ij} dans cette même ligne de \mathbf{d} et contient, comme nous l'avons déjà dit, un 1 si la branche k existe sur le chemin allant de l'espèce i à l'espèce j . Pour les arbres des figures 2.3 et 2.4, nous avons la matrice topologique suivante :

$$\mathbf{M} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Notons que la taille de la matrice est 10 (le nombre de distances) sur 7 (le nombre de branches). Si nous empilons les v_i dans un vecteur, dans l'ordre des i , les équations 2.5 peuvent être exprimées par les notations matricielles suivantes :

$$\mathbf{M}^T \mathbf{d} = (\mathbf{M}^T \mathbf{M}) \mathbf{v} \quad (2.6)$$

En multipliant par la gauche par l'inverse de $(\mathbf{M}^T \mathbf{M})$, nous pouvons résoudre pour les longueurs des branches des moindres carrés l'équation :

$$\mathbf{v} = (\mathbf{M}^T \mathbf{M})^{-1} \mathbf{M}^T \mathbf{d} \quad (2.7)$$

C'est une méthode standard pour exprimer les problèmes des moindres carrés dans une notation matricielle et les résoudre. Quand nous avons les moindres carrés pondérés, avec une matrice diagonale de poids dans le même ordre que les D_{ij} :

$$\mathbf{W} = \begin{bmatrix} w_{AB} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{AC} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_{AD} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{AE} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{BC} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{BD} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_{BE} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{CD} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{CE} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{DE} \end{bmatrix}$$

alors les équations des moindres carrés peuvent s'écrire comme suit :

$$\mathbf{M}^T \mathbf{W} \mathbf{d} = (\mathbf{M}^T \mathbf{W} \mathbf{M}) \mathbf{v} \quad (2.8)$$

et leur solution :

$$\mathbf{v} = (\mathbf{M}^T \mathbf{W} \mathbf{M})^{-1} \mathbf{M}^T \mathbf{W} \mathbf{d} \quad (2.9)$$

Encore une fois, c'est un résultat standard dans la théorie des moindres carrés, qui a été utilisé pour la première fois dans les estimations phylogénétiques de Cavalli-Sforza et Edwards (1967).

On peut imaginer une méthode de moindres carrés de matrice de distance qui, pour chaque topologie d'arbre, forme la matrice $\mathbf{M}^T\mathbf{M}$ (ou $\mathbf{M}^T\mathbf{W}\mathbf{M}$), inverse celle-ci, et obtient l'estimation 2.7 (ou 2.9). Cela peut être réalisé, mais c'est très lourd en ressource de calcul, même si toutes les topologies ne sont pas examinées. L'inversion de la matrice $\mathbf{M}^T\mathbf{W}\mathbf{M}$ prend l'ordre de n^3 opérations pour un arbre de n espèces. En principe, ce calcul doit être réalisé pour chaque topologie d'arbre considérée. Gascuel (1997) et Bryant et Waddell (1998) ont présenté des méthodes plus rapides qui calculent les solutions exactes des équations des moindres carrés des longueurs des branches. Ils se basent sur des travaux antérieurs de Vach (1989), Vach et Degens (1991), Rzhetsky et Nei (1993). Pour un arbre avec n espèces ces méthodes rapides sauvent au moins un facteur de l'ordre de n opérations. Nous y reviendrons dans le chapitre suivant.

2.4 Décomposition de la matrice topologique

Une propriété intéressante de la matrice topologique est sa décomposition en blocs. Nous allons voir que ces blocs ont des informations redondantes les uns envers les autres et que nous n'aurons pas besoin de la matrice topologique complète pour représenter de manière univoque l'arbre phylogénétique que celle-ci représente.

Considérons un ensemble de cinq espèces et un arbre binaire non enraciné correspondant représenté à la figure 2.5. Un tel arbre est représenté univoquement par une matrice binaire \mathbf{M} de taille $n(n-1)/2 \times (2n-3) = 10 \times 7$ montrée à la Table 2.1.

Une telle matrice topologique \mathbf{M} peut être décomposée en blocs ([6]) :

$$\mathbf{M} = \left(\begin{array}{c|c} \mathbf{G} & \mathbf{F} \\ \hline \mathbf{G}^\otimes & \mathbf{F}^\otimes \end{array} \right)$$

où \mathbf{G} est une matrice obtenue par les $n-1$ premières lignes et les n premières colonnes de \mathbf{M} :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

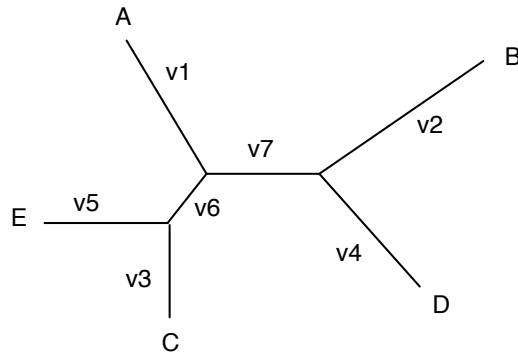


FIG. 2.5 – Cette figure représente un arbre non enraciné pour cinq feuilles (espèces). Les poids v_i sont des variables qui peuvent être calculées à partir de la méthode des moindres carrés.

	v_1	v_2	v_3	v_4	v_5	v_6	v_7
AB	1	1	0	0	0	0	1
AC	1	0	1	0	0	1	0
AD	1	0	0	1	0	0	1
AE	1	0	0	0	1	1	0
BC	0	1	1	0	0	1	1
BD	0	1	0	1	0	0	0
BE	0	1	0	0	1	1	1
CD	0	0	1	1	0	1	1
CE	0	0	1	0	1	0	0
DE	0	0	0	1	1	1	1

TAB. 2.1 – Matrice topologique décrivant l'arbre représenté à la figure 2.5.

\mathbf{F} est une matrice obtenue en prenant les $n - 1$ premières lignes et les colonnes $n + 1$ à $2n - 3$ de \mathbf{M} :

$$\mathbf{F} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

\mathbf{G}^{\otimes} est la matrice obtenue en prenant les lignes n à $n(n-1)/2$ et les n premières colonnes de \mathbf{M} :

$$\mathbf{G}^{\otimes} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

\mathbf{F}^{\otimes} est la matrice obtenue en prenant les lignes n à $n(n-1)/2$ et les colonnes $n + 1$ à $2n - 3$ de \mathbf{M} :

$$\mathbf{F}^{\otimes} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}$$

La matrice \mathbf{G} représente l'assignement d'un sous-ensemble de n arêtes à n feuilles. Nous appelons un tel sous-ensemble un ensemble d'arêtes externes, et le sous-ensemble des $n - 3$ arêtes restantes, l'ensemble des arêtes internes.

La matrice \mathbf{G}^{\otimes} est une information redondante par rapport à \mathbf{G} . Une fois la matrice \mathbf{G} créée, la matrice \mathbf{G}^{\otimes} est univoquement déterminée par les opérations suivantes :

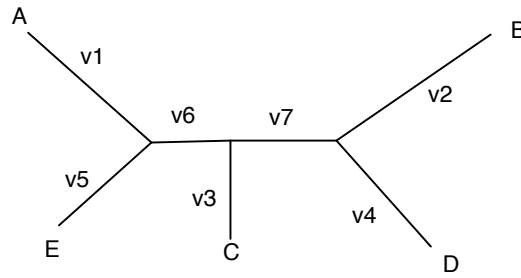


FIG. 2.6 – Une topologie alternative de celle représentée à la figure 2.5

$$\begin{aligned} \mathbf{g}_p^\otimes &= \mathbf{g}_i \otimes \mathbf{g}_j & \forall p = n, \dots, (n(n-1))/2 \\ i &= 1, \dots, n-2 \\ j &= i+1, \dots, n-1 \end{aligned}$$

où \mathbf{g}_p^\otimes est la $p^{\text{ème}}$ ligne de \mathbf{G}^\otimes , \mathbf{g}_i et \mathbf{g}_j sont respectivement les $i^{\text{ème}}$ et $j^{\text{ème}}$ lignes de \mathbf{G} , et \otimes est le ou-exclusif parmi les composantes des vecteurs \mathbf{g}_i et \mathbf{g}_j . Tout comme la matrice \mathbf{G} , \mathbf{G}^\otimes invariante par rapport aux transformations de la topologie interne de l'arbre. Donc \mathbf{G} et \mathbf{G}^\otimes représentent la partie invariante de la matrice topologique \mathbf{M} .

Au contraire, les matrices \mathbf{F} et \mathbf{F}^\otimes sont les parties variables de la matrice topologique \mathbf{M} . \mathbf{F} représente les interactions existant entre les arêtes internes et les arêtes externes, c'est-à-dire : (i) quelle arête interne est liée à quelles arêtes externes particulières, et (ii) comment les arêtes internes sont liées entre elles. En fait, une telle matrice représente la topologie de l'arbre à elle seule. Par exemple, admettons que la matrice \mathbf{F} est modifiée de la façon suivante :

$$\mathbf{F} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{pmatrix}$$

et \mathbf{F} décrit donc la nouvelle topologie de l'arbre représentée à la figure 2.6. De même que pour \mathbf{G}^\otimes , une fois que la matrice \mathbf{F} est créée, la matrice \mathbf{F}^\otimes est

univoquement déterminée par les opérations suivantes :

$$\begin{aligned} \mathbf{f}_p^\otimes &= \mathbf{f}_r \otimes \mathbf{f}_s & \forall p = n, \dots, (n(n-1))/2 \\ r &= 1, \dots, n-2 \\ s &= r+1, \dots, n-1 \end{aligned}$$

où \mathbf{f}_p^\otimes est la $p^{\text{ème}}$ ligne de \mathbf{F}^\otimes , \mathbf{f}_r et \mathbf{f}_s sont respectivement les $r^{\text{ème}}$ et $s^{\text{ème}}$ lignes de \mathbf{F} .

2.5 Formulation finale

Formalisons le principe de l'EM : soit Γ l'ensemble des espèces (taxons) à analyser, $|\Gamma| = n$, et avec $\mathbf{D} = \{d_{ij}\}$ la matrice symétrique $n \times n$ des distances évolutives.

Ainsi le principe de l'EM peut se formuler en terme de problème d'optimisation comme suit :

$$\min_{\mathbf{M} \in \mathcal{M}, \mathbf{v} \in \mathcal{V}} \|f(\mathbf{D}, \mathbf{M}, \mathbf{v})\| \quad (2.10)$$

La matrice \mathbf{M} est la matrice topologique, et par après appelée *solution du problème* ; \mathbf{v} est le vecteur dont les éléments représentent les poids associés aux $2n - 3$ arêtes de l'arbre ; \mathcal{M} et \mathcal{V} sont respectivement l'ensemble de toutes les matrices topologiques (par après appelé *l'espace de solutions*), et l'ensemble de tous les poids associés au vecteur \mathbf{v} . Finalement, $\|f(\mathbf{D}, \mathbf{M}, \mathbf{v})\|$ est la norme L^1 du vecteur résultant $f(\mathbf{D}, \mathbf{M}, \mathbf{v})$.

L'espace de solution \mathcal{M} peut être défini de la façon suivante :

$$\mathbf{M} \in \mathcal{M} \quad (2.11)$$

$$\text{ssi } m_{p1} = 1 \quad p = 1, \dots, n-1 \quad (2.12)$$

$$m_{p(p+1)} = 1 \quad p = 1, \dots, n-1 \quad (2.13)$$

$$m_{(n(r-1) - \frac{r(r+1)}{2} + s), e} \leq m_{r-1, e} + m_{s-1, e} \quad (2.14)$$

$$m_{(n(r-1) - \frac{r(r+1)}{2} + s), e} \leq 2 - m_{r-1, e} - m_{s-1, e} \quad (2.15)$$

$$m_{(n(r-1) - \frac{r(r+1)}{2} + s), e} \geq m_{r-1, e} - m_{s-1, e} \quad (2.16)$$

$$m_{(n(r-1) - \frac{r(r+1)}{2} + s), e} \geq -m_{r-1, e} + m_{s-1, e} \quad (2.17)$$

$$r = 2, \dots, n-1$$

$$s = r+1, \dots, n$$

$$e = 1, \dots, 2n-3$$

$$m_{A_i, s} + m_{A_j, s} + m_{A_i, r} - m_{A_j, r} \leq 2 \quad (2.18)$$

$$m_{A_i, s} + m_{A_j, s} - m_{A_i, r} + m_{A_j, r} \leq 2 \quad (2.19)$$

$$i, j = 1, \dots, n-1$$

$$r, s = n+1, \dots, 2n-3$$

$$r < s$$

$$\sum_{i \in V \setminus A} 2 \leq m_{A_i, r} \leq n-2 \quad \forall r = n, \dots, 2n-3 \quad (2.20)$$

$$\sum_{i \in V \setminus A} m_{A_i, s} \leq \sum_{i \in V \setminus A} m_{A_i, r} - 1 + (n-1)(2 - m_{A_j, r} - m_{A_j, s}) \quad (2.21)$$

où A est la première feuille.

Le problème 2.10 a été démontré comme étant NP-difficile par Day en 1987 et avec l'équation 2.6, nous avons :

$$f(\mathbf{D}, \mathbf{M}, \mathbf{v}) = (\mathbf{M}\mathbf{v} - \mathbf{d})^t (\mathbf{M}\mathbf{v} - \mathbf{d}) \quad (2.22)$$

Ce qui nous ramène, avec l'équation 2.7, au problème d'optimisation suivant :

$$\min \|\mathbf{v}\| = \|(\mathbf{M}^t \mathbf{M})^{-1} \mathbf{M}^t \mathbf{d}\| \quad (2.23)$$

$$\text{s.t. } \mathbf{M} \in \mathcal{M} \quad (2.24)$$

Chapitre 3

Stratégie implémentée

3.1 Solution du problème

Comme nous l'avons vu au chapitre précédent, nous allons essayer de trouver l'arbre phylogénétique de poids minimum à partir d'une matrice de distances entre les espèces étudiées, et donc la matrice topologique qui représente un tel arbre.

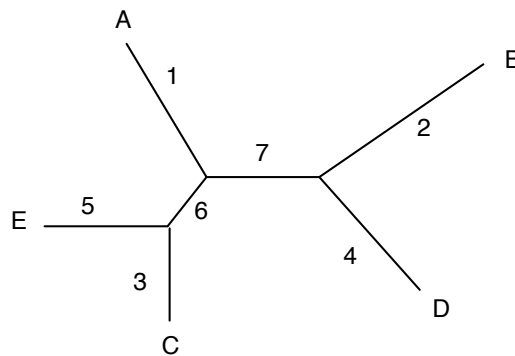
Les méthodes que nous allons développer ici vont utiliser la propriété de décomposition de cette dernière vu au chapitre 2.4. Cette décomposition de matrice en blocs est intéressante car plutôt que de rechercher la meilleure matrice topologique complète \mathbf{M} qui minimise les moindres carrés de la longueur totale de l'arbre, nous pouvons nous contenter de ne chercher que la meilleure matrice \mathbf{F} . Les méthodes que nous allons développer pour résoudre notre problème, et c'est là leur originalité, seront basées exclusivement sur cette dernière constatation. Par la suite, nous considérerons ce bloc \mathbf{F} comme étant la matrice topologique complète, les autres blocs étant inutiles avec notre implémentation.

La matrice topologique par après se représentera comme suit :

- les arêtes seront numérotées de 1 à $2n - 3$
- les arêtes externes seront numérotées de 1 à n
- les arêtes internes seront numérotées de $n + 1$ à $2n - 3$
- les feuilles seront labellées par les lettres A, B, C, \dots et seront liées à l'arête externe $1, 2, 3, \dots, n$ correspondante.
- les lignes de la matrice correspondront aux feuilles donc aussi aux arêtes externes correspondantes

- les colonnes de la matrice correspondront aux arêtes internes et seront numérotées en conséquence

Reprenons l'exemple du chapitre 2.4, nous aurons dès lors l'arbre phylogénétique et la matrice topologique repris à la figure 3.1



	6	7
B	0	1
C	1	0
D	0	1
E	1	0

FIG. 3.1 – Arbre phylogénétique avec la matrice topologique associée comme nous allons la considérer par la suite.

3.2 Recherche Tabou

La recherche avec tabous (parfois appelée simplement recherche tabou) a été présentée par Fred Glover dans un article paru en 1986 [Glover 86] mais reprenant de nombreuses idées proposées antérieurement dans les années 60. Les deux articles simplement intitulés *Tabu Search* [Glover 89, Glover 90] proposent la plupart des principes de la recherche avec tabous telle qu'elle est décrite actuellement. Certains de ces principes ont mis du temps avant de s'imposer dans la communauté scientifique. En effet, dans la première moitié des années 90, la plupart des implémentations de la recherche avec tabous ne faisaient appel qu'à un sous-ensemble très restreint des principes de la technique, souvent limité à une *liste de tabous* et à une *condition d'aspiration* élémentaires.

Le principe de base de la recherche tabou est de poursuivre la recherche locale au-delà d'un optimum local en permettant des mouvements qui ne permettent pas nécessairement d'améliorer la solution courante. Pour se prémunir de retourner dans le (ou les) même optimum local, il faut interdire à la recherche de pouvoir y retourner. En d'autres termes, des solutions doivent être rendues taboues. Pour cela, on utilise des mémoires, appelées *liste taboue*, qui enregistre l'historique récent de la recherche.

3.2.1 Contexte historique

Avant d'introduire les concepts de base de la recherche taboue, nous allons avoir un bref aperçu de sa genèse et sa relation avec des recherches antérieures.

Les heuristiques, i.e., des techniques d'approximation de solutions, ont été utilisées depuis les origines de la recherche opérationnelle pour faire face à des problèmes combinatoires difficiles. Avec le développement de la théorie de la complexité au début des années 70, il devint clair que, vu que la plupart de ces problèmes étaient probablement *NP – difficiles*, il y avait peu d'espoir de jamais trouver des procédures efficaces pour obtenir des solutions exactes pour ces problèmes. Cette prise de conscience a souligné le rôle des heuristiques pour résoudre des problèmes d'optimisation qui étaient rencontrés dans des applications de la vie de tous les jours, et qui avaient besoin d'être résolus, qu'ils soient *NP – difficiles* ou non. Bien que nombreuses furent les différentes approches qui ont été proposées et expérimentées, la plus populaire était basée sur des techniques d'améliorations de la Recherche Locale. La Recherche Locale peut être plus ou moins résumée par une procédure de recherche itérative qui, partant d'une solution initiale, améliore progressivement celle-ci en appliquant une série de modifications locales (ou *mouvements*). A chaque itération, la recherche se déplace vers une autre solution améliorée qui diffère seulement légèrement de la solution courante. La recherche se termine quand elle rencontre un optimum local en fonction des transformations considérées, ce qui est une faiblesse de la méthode : sauf avec beaucoup de chance, cet optimum local est souvent une assez médiocre solution. Dans la Recherche Locale, la qualité de la solution obtenue et le temps de calcul dépendent généralement très fort de la "richesse" de l'ensemble de transformations considérées à chaque étape itération de l'heuristique.

En 1983, le monde de l'optimisation combinatoire fut bouleversé par l'apparition d'un article dans lequel les auteurs (Kirkpatrick, Gelatt et Vecchi) ont décrit

une nouvelle approche d'heuristique appelée *Recuit Simulé* qui pouvait converger vers une solution optimale d'un problème combinatoire, bien que cela se fasse en un temps de calcul infini. Basé sur une analogie avec les mécaniques statistiques, le Recuit Simulé pouvait être interprété comme une forme de marche aléatoire contrôlée dans l'espace de solutions possibles. L'émergence du Recuit Simulé appelait à regarder une autre forme d'approche pour résoudre les problèmes d'optimisation combinatoires et suscita un intérêt certain dans la communauté de recherche. Dans les années qui suivirent, beaucoup d'autres approches, la plupart basées sur des analogies avec des phénomènes naturels, ont été proposées (Recherche Taboue, Systèmes de Colonies de Fourmis, ...) et, comme d'autres plus anciennes, comme les *Algorithmes Génétiques* (Holland, 1975), ces méthodes ont acquis une certaine popularité. Maintenant appelées sous le nom de *Meta-Heuristiques* (un terme qui a été inventé à l'origine par Glover en 1986), ces méthodes sont devenues depuis les 15 dernières années la pierre angulaire des approches heuristiques pour résoudre les problèmes d'optimisation combinatoire.

3.2.2 Espace de solution et voisinage

Comme nous venons de le mentionner, la recherche tabou est une extension des méthodes de recherche locale classiques. En fait, la recherche tabou basique peut être vue simplement comme la combinaison de la recherche locale avec une mémoire à court terme. Cela induit que les deux premiers éléments basiques de n'importe quel heuristique de recherche tabou sont la définition de son **espace de solution** et de son **voisinage**.

Définition 4 *L'espace de solution d'une recherche locale ou d'un heuristique d'une recherche tabou est simplement l'espace de toutes les solutions possibles qui peuvent être considérées (visitées) durant la recherche.*

La définition du voisinage est étroitement lié à celle de l'espace de solution. A chaque itération de la recherche locale ou de la recherche tabou, la transformation locale qui peut être appliquée à la solution courante, notée S , définit un ensemble de solutions voisines dans l'espace de solution, noté $N(S)$ (le voisinage de S).

Comme nous l'avons déjà mentionné, l'espace de solution utilisé dans notre problème est l'ensemble de toutes les matrices topologiques qui satisfont les conditions mentionnées au chapitre 2.3. En ce qui concerne le voisinage, nous

allons définir une série de transformations ou *mouvements* pour passer d'une solution à une autre au chapitre 3.3.

3.2.3 Tabous

Les tabous sont un des éléments distincts de la recherche tabou quand on compare celle-ci à la recherche locale. Comme nous l'avons déjà mentionné, les tabous sont utilisés pour prévenir des cycles quand on quitte un optimum local en effectuant des mouvements qui n'améliorent pas la solution. Quand cette situation arrive, quelque chose doit être fait pour éviter que la recherche ne revienne là où elle était au départ. Ceci est réalisé en déclarant *tabou* (non-permis) les mouvements inverses de ceux qui viennent d'être effectués. Les tabous sont aussi utiles pour aider la recherche à visiter des portions non-visitées de l'espace de recherche et donc améliorer l'exploration.

Les tabous sont stockés dans une mémoire à court terme de la recherche (la liste tabou) et généralement, seulement une quantité fixée et limitée d'information y est enregistrée. Dans n'importe quel contexte donné, il y a plusieurs possibilités pour les informations spécifiques à stocker. On pourrait par exemple, enregistrer des solutions complètes, mais cela nécessiterait beaucoup de mémoire et il serait coûteux de vérifier si un mouvement potentiel est tabou ou non. Les tabous les plus fréquemment utilisés consistent à enregistrer les dernières transformations effectuées sur la solution courante et d'interdire les mouvements inverses. D'autres possibilités sont basées sur des caractéristiques clés des solutions elles-mêmes ou des mouvements.

3.2.4 Algorithme implémenté

La Figure 3.2 donne un aperçu de l'implémentation de la recherche tabou utilisée pour notre problème.

Une solution s est en fait une matrice topologique F d'un arbre phylogénétique considéré. Comme nous l'avons spécifié précédemment, une matrice topologique est appelée solution du problème. a et b sont des indices de lignes et/ou de colonnes de cette matrice. Ces indices sont utilisés par les différents *Mouvements* (voir ci-après) pour déterminer quels changements sont à effectuer dans la matrice pour obtenir une nouvelle solution s' . a et b ne peuvent pas être choisis de manière tout à fait aléatoire. En effet, en fonction du mouvement, certaines conditions sont à respecter pour effectuer le dit mouvement, nous y reviendrons

TABU SEARCH

Initialisation des structures mémoires

 $s_0 \leftarrow$ Générer Solution initiale $s_{best} \leftarrow s_0$ Tant que (*condition de terminaison* n'est pas atteinte) $NewTabuListLenght()$ Choisir aléatoirement a et b tels que $(a, b) \notin TabuList$ Si a et b satisfont les conditions du *Mouvement* $s \leftarrow$ *Mouvement*(s_0, a, b) $TabuList \leftarrow (a, b)$ Si ($f(s) < f(s_{best})$)

Mise à jour des structures mémoires

 $s_{best} \leftarrow s$ $s_0 \leftarrow s$

Sinon

 $s \leftarrow s_0$

FIG. 3.2 – Tabu Search.

au Chapitre 3.3.

Toujours en ce qui concerne les mouvements, ceux-ci peuvent être, comme nous l'avons expliqué précédemment, rendus tabous. Par exemple si on effectue un mouvement avec la paire (a, b) , celle-ci sera rendue taboue pendant un nombre t d'itérations. Une valeur de t fixe ne produit pas une recherche très robuste, car des cycles peuvent apparaître, même pour des t assez grands. Pour faire face à ce problème, il est proposé dans [Taillard91] de tirer t aléatoirement, uniformément entre $[0, 9n]$ et $[1, 1n + 4]$ ([11]). En effet, une durée d'interdiction égale à la taille du problème, ou légèrement plus grande pour les petits exemples, semble assez bonne, expérimentalement. D'où l'idée de tirer la valeur de t de façon dynamique en cours de recherche.

En pratique, pour implanter le mécanisme d'interdiction, donc la *TabuList*, on utilisera une matrice TL dont l'entrée tl_{ab} donnera le numéro de l'itération à laquelle la paire (a, b) peut à nouveau être utilisée par un mouvement. Ce numéro correspond en fait au numéro de la dernière itération à laquelle nous avons utilisée la dite paire auquel nous avons rajouté la nouvelle taille de la *TabuList* retournée par la fonction $NewTabuListLenght()$. Ainsi, le mouvement (a, b) sera interdit si les deux entrées tl_{ab} et tl_{ba} contiennent des valeurs supérieures au

numéro de l'itération courante.

La *condition de terminaison* peut être définie de plusieurs façons. On peut par exemple imposer un nombre fini d'itérations de l'algorithme ou une durée maximale du temps d'exécution. Enfin, f est la fonction objectif à minimiser. C'est cette fonction qui va évaluer les solutions trouvées par la recherche taboue. Nous y reviendrons au Chapitre 3.4.

Finalement, en ce qui concerne la génération de la solution de départ, nous allons utiliser l'algorithme du Neighbor-Joining dont le principe a été énoncé au Chapitre 1. Celui-ci peut se résumer de la manière suivante ([13], p.167) :

1. Pour chaque taxon, calculer $u_i = \sum_{j:j \neq i}^n D_{ij} / (n-2)$. Notons que le dénominateur n'est pas le nombre d'éléments sommé.
2. Choisir i et j tel que $D_{ij} - u_i - u_j$ est minimal.
3. Joindre les éléments i et j . Calculer la longueur des branches de i vers le nouveau noeud (v_i) et de j vers le nouveau noeud (v_j) tel que :

$$v_i = \frac{1}{2}D_{ij} + \frac{1}{2}(u_i - u_j)$$

$$v_j = \frac{1}{2}D_{ij} + \frac{1}{2}(u_j - u_i)$$

4. Calculer les distances entre le nouveau noeud (ij) et chaque taxon restant tel que :

$$D_{(ij),k} = (D_{ik} + D_{jk} - D_{ij})/2$$

5. Enlever les taxons i et j des tables et les remplacer par le nouveau noeud, (i, j) , qui est dorénavant traité comme un taxon à part entière.
6. Si plus de deux noeuds de départ subsistent, retourner à l'étape 1. Sinon, connecter les deux derniers noeuds, disons l et m par une branche de longueur D_{lm} .

3.3 Mouvements

A la base des recherches locales, il y a la définition de l'ensemble $N(S)$ des solutions voisines de S , mais d'un point de vue pratique, on aura intérêt à considérer, plutôt que l'ensemble $N(S)$, l'ensemble des modifications que l'on peut apporter à S .

Définition 5 On appelle *mouvement* une modification apportée à une solution.

L'ensemble $N(s)$ des solutions voisines de la solution s s'exprimera comme l'ensemble des solutions admissibles que l'on peut obtenir en appliquant à la solution s un mouvement m appartenant à un ensemble de mouvements MV . L'application de mv à s sera notée $s \oplus mv$ et on aura l'équivalence $N(s) = \{s' \mid s' = s \oplus mv, mv \in MV\}$. Exprimer le voisinage en terme de mouvement permet, lorsque c'est possible, de caractériser l'ensemble M beaucoup plus facilement.

La définition du voisinage influence directement le temps de calcul nécessaire pour décrire l'espace de solutions, et indirectement la qualité des solutions trouvées. Une bonne définition de voisinage est souvent un compromis entre ces deux facteurs. Comprendre la nature combinatoire de ce problème est fondamentale pour modéliser des opérateurs de voisinages ou mouvements.

Nous allons définir les mouvements utilisés dans notre implémentation.

3.3.1 LE

L'algorithme LE, pour Leaf Exchange, permet d'échanger deux arêtes externes de l'arbre initial, quelle qu'elles soient. L'algorithme de LE est formalisé comme représenté à la Figure 3.3.

LE(F, a, b)

Pour chaque arête interne i

temp $\leftarrow F[a][i]$

$F[a][i] \leftarrow F[b][i]$

$F[b][i] \leftarrow$ temp

FIG. 3.3 – Algorithme LE.

3.3.2 MLE

L'algorithme MLE, pour Multi Leaf Exchange, est très similaire à LE . En effet, cet algorithme permet d'échanger trois arêtes externes de l'arbre initial entre elles. L'algorithme MLE est formalisé comme représenté à la Figure 3.4.

MLE(F, a, b, c)

Pour chaque arête interne i

temp $\leftarrow F[a][i]$

$F[a][i] \leftarrow F[b][i]$

$F[b][i] \leftarrow F[c][i]$

$F[c][i] \leftarrow$ temp

FIG. 3.4 – Algorithme MLE.

3.3.3 NNI

L'algorithme NNI, pour Nearest Neighbor Interchange, échange les sous-arbres d'un arbre initial, séparés par trois arêtes. L'arête du milieu x est toujours

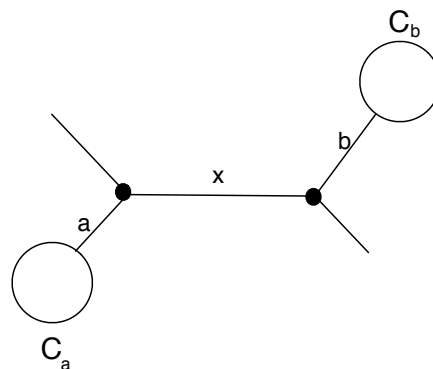


FIG. 3.5 – Les sous-arbres C_a et C_b peuvent être échangés par NNI.

une arête interne tandis que les deux autres arêtes a et b peuvent être soit interne, soit externe. Ce qui fait que NNI est caractérisé par 3 cas :

1. le cas où a et b sont internes.
2. le cas où soit a soit b est externe et l'autre interne.
3. le cas où a et b sont toutes les deux externes.

NNI est formalisé par les algorithmes représentés aux Figures 3.6 et 3.7.

```

NNI( $F, x, a, b$ )
Pour chaque arête externe  $i$ 
  Si  $F[i][a] == 1$ 
    Si  $F[i][x] == 0$ 
       $F[i][x] \leftarrow 1$ 
    Sinon
       $F[i][x] \leftarrow 0$ 
  Sinon si  $F[i][b] == 1$ 
    Si  $F[i][x] == 0$ 
       $F[i][x] \leftarrow 1$ 
    Sinon
       $F[i][x] \leftarrow 0$ 

```

FIG. 3.6 – NNI pour le cas où a et b sont des arêtes internes.

```

NNI2( $F, x, a, b$ )
Pour chaque arête externe  $i$ 
  Si  $i \neq b$ 
    Si  $F[i][a] == 1$ 
      Si  $F[i][x] == 1$ 
         $F[i][x] \leftarrow 0$ 
      Sinon
         $F[i][x] \leftarrow 1$ 
  Sinon
    Si  $F[i][x] == 0$ 
       $F[i][x] \leftarrow 1$ 
    Sinon
       $F[i][x] \leftarrow 0$ 

```

FIG. 3.7 – NNI pour le cas où a est une arête interne et b une arête externe.

où F est la matrice topologique associée à l'arbre initial. x, a et b sont les arêtes comme représentées à la figure 3.5. Pour le cas où a et b sont des arêtes externes, on utilisera l'algorithme LE en ne tenant pas compte de x .

3.3.4 IELE

L'algorithme IELE, pour Internal Edge and Leaf Exchange, permet d'échanger une arête externe avec une arête interne n'appartenant pas au même sous-arbre. Cette dernière condition se formalise de la façon suivante :

Corollaire 2 *Soit une arête interne b , une arête externe a et F la matrice topologique de l'arbre associé. a et b peuvent être échangées par l'algorithme IELE ssi $F[a][b] == 0$.*

L'algorithme IELE se formalise comme représenté à la Figure 3.8.

```

IELE( $F, a, b$ )
/*  $a$  est l'arête externe et  $b$  l'arête interne */
/*  $a$  et  $b$  sont tels que  $F[a][b] == 0$  */

```

```

Pour chaque arête externe  $i$ 
  Si  $F[i][b] == 1$ 
    Insérer  $i$  dans la liste  $L$ 

```

```

Pour chaque arête interne  $j$ 
  Si  $j \neq b$ 
     $ok \leftarrow \top$ 
    choisir aléatoirement  $x \in L$ 
    Pour chaque  $i \in L$  et  $i \neq x$ 
      Si  $F[x][j] \neq F[i][j]$ 
         $ok \leftarrow \perp$ 
    Si  $F[x][j] \neq F[a][j] \wedge ok == \top$ 
       $temp \leftarrow F[x][j]$ 
      Pour chaque  $i \in L$ 
         $F[i][j] \leftarrow F[a][j]$ 
       $X[a][j] \leftarrow temp$ 

```

FIG. 3.8 – L'algorithme IELE échange une arête externe a avec une arête interne b .

3.3.5 IEE

L'algorithme IEE, pour Internal Edge Exchange, permet d'échanger deux arêtes internes n'appartenant pas au même sous-arbre. Cette dernière condition se formalise de la façon suivante :

Corollaire 3 Soit a et b deux arêtes internes et X la matrice topologique de l'arbre associé. a et b peuvent être échangées par l'algorithme IEE ssi $\nexists i$ tel que $F[i][a] == 1 \wedge F[i][b] == 1$.

L'algorithme IEE se formalise comme représenté à la Figure 3.9.

IEE(F, a, b)

Pour chaque arête externe i

 Si $F[i][a] == 1$

 Insérer i dans la liste LF

 Sinon si $F[i][b] == 1$

 Insérer i dans la liste LF

Pour chaque arête interne j

$i \leftarrow 1^{\text{er}}$ élément de LF

 Si $F[i][j] == F[i][a]$

$ok \leftarrow \top$

 Pour chaque $i \in LF$

 Si $F[i][j] \neq F[i][a]$

$ok \leftarrow \perp$

 Si $ok == \top$

 Insérer j dans la liste LA

 Sinon si $F[i][j] == F[i][b]$

$ok \leftarrow \top$

 Pour chaque $i \in LF$

 Si $F[i][j] \neq F[i][b]$

$ok \leftarrow \perp$

 Si $ok == \top$

 Insérer j dans la liste LA

Pour chaque $j \in LA$

 Pour chaque $i \in LF$

 Si $F[i][j] == 1$

$F[i][j] \leftarrow 0$

 Sinon $F[i][j] \leftarrow 1$

FIG. 3.9 – Algorithme IEE.

3.3.6 IE

Le mouvement le plus intéressant est sans doute la combinaison de l'algorithme IELE avec l'algorithme IEE ou IE pour Internal Exchange (ça ne veut rien dire mais est-ce vraiment important?). Sur les deux arêtes a et b que l'algorithme reçoit en paramètre, l'une est obligatoirement une arête interne, l'autre étant interne ou externe. Cette combinaison est formalisée à la Figure 3.10.

$\text{IE}(F, a, b)$

Si a est une arête externe

 Si $F[a][b] == 0$

 IELE(a, b)

Sinon si a et $b \notin$ au même sous-arbre

 IEE(a, b)

FIG. 3.10 – Algorithme IE avec b obligatoirement interne.

3.4 Evaluation de la solution

La méthode utilisée pour l'évaluation d'une solution est inspirée de la méthode développée par Bryant et Waddell ([4]). Précédemment, nous avons utilisé la décomposition de Cholesky pour ce faire, mais les résultats étaient tellement catastrophiques au niveau de la rapidité de calcul que nous n'allons considérer que la méthode présentée ici.

La variante de la méthode de Bryant et Waddell élaborée ici repose, comme le reste de notre implémentation, sur l'utilisation exclusive de la matrice F . Nous allons à présent introduire quelques notions théoriques avant de développer cette méthode.

Soit un ensemble L de N espèces considérées, M la matrice topologique de l'arbre associé, d l'ensemble des distances entre les espèces et b le vecteur colonne des longueurs des arêtes de l'arbre.

Définition 6 Une coupe $A|B$ est une partition d'un ensemble de taxons en deux parties, A et B .

Définition 7 Chaque arête e d'un arbre correspond à une coupe unique. Cette coupe est appelée la coupe correspondant à l'arête e .

N'importe quelle coupe à une **mesure de coupe**, une distance sur l'arbre où deux taxons sont de distances 1 s'ils sont des différents côtés de la coupe et de distance 0 s'ils sont du même côté de la coupe. Les colonnes de la matrice topologique M de l'arbre sont exactement les mesures de coupe associées aux mesures de coupe dans l'arbre. La colonne k de la matrice est la mesure de coupe pour la coupe correspondant à l'arête e_k .

Pour rappel, le problème consiste à minimiser \mathbf{v} tel que :

$$\mathbf{v} = (\mathbf{M}^t \mathbf{M})^{-1} \mathbf{M}^t \mathbf{d}$$

3.4.1 SUCCESSEURS

Le premier algorithme que nous allons développer sert à trouver les successeurs directs d'une arête interne, en admettant que la première feuille, A , est la racine fictive.

Comme nous l'avons vu à la section 3.1, les colonnes de la matrice topologique définissent chacune l'arête interne associée. L'algorithme se base sur le théorème suivant :

Corollaire 4 Soit une arête interne x et F la matrice topologique de l'arbre associé. L'arête interne i est un successeur de x ssi $\forall j$ tel que $F[j][i] == 1$, alors $F[j][x] == 1$.

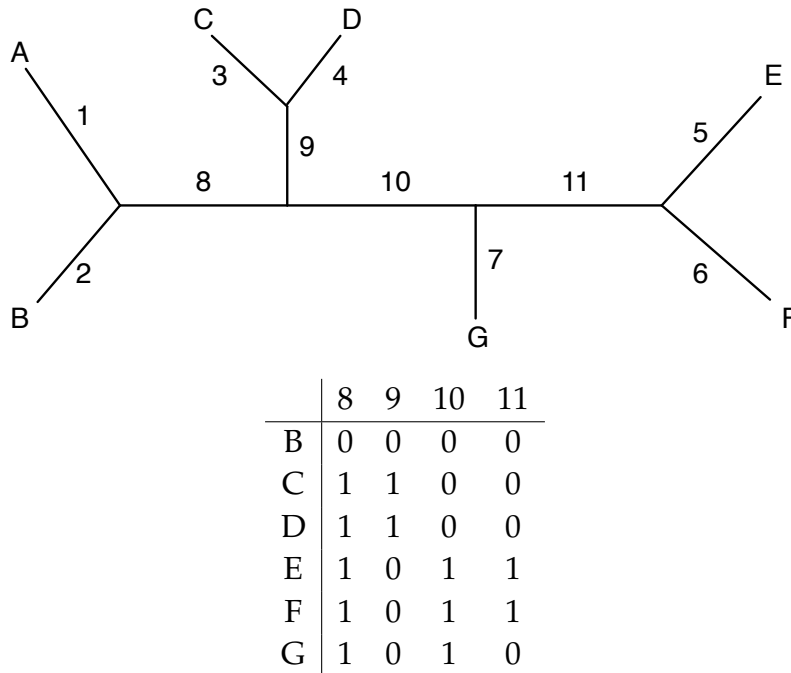


FIG. 3.11 – Un arbre avec 7 taxons et la matrice topologique associée.

Illustrons cette propriété par un exemple. Considérons l'arbre représenté à la figure 3.11 et la matrice topologique associée. En reprenant la propriété énoncée ci-dessus, nous voyons aisément dans la matrice topologique que les arêtes internes qui succèdent à l'arête 8, sont les arêtes 9, 10 et 11. En effet, la colonne 8 possède le plus de "1" et chacune des autres arêtes internes à des "1" sur les mêmes lignes qu'elle.

De même, nous voyons que l'arête 11 succède elle-même à l'arête 10.

Cependant, trouver les successeurs directs, c'est-à-dire ceux qui ont un noeud commun avec l'arête considérée, est un peu plus délicat. Il faut pour cela considérer 3 cas :

1. le cas où un successeur est une arête interne et l'autre une arête externe.
2. le cas où les deux successeurs sont toutes les deux une arête interne.
3. le cas où les deux successeurs sont toutes les deux une arête externe.

L'algorithme `SUCCESEURS` suivant prend en compte ces 3 cas et retourne les deux successeurs directs d'une arête interne passée en paramètre, que ceux-ci soient des arêtes internes ou externes :

SUCCESEURS(F, x)

Pour toutes les arêtes internes i

$compteur \leftarrow 0$

 Si i est un successeur de x

$compteur \leftarrow$ nombre de différences de "1" entre x et i

 Heap.Push($i, compteur$)

 /* le Heap est trié sur $compteur$ de façon croissante */

Si Heap n'est pas vide

 (a, dif) \leftarrow Heap.Pop()

 Si ($dif == 1$)

$cas \leftarrow 1$

$b \leftarrow$ n° de ligne où x et a différent

 Sinon

$cas \leftarrow 2$

 Tant qu'on a pas trouvé le 2^{ème} successeur direct

 (b, dif) \leftarrow Heap.Pop()

$test \leftarrow$ TRUE

 Pour chaque ligne j où la colonne $a \neq$ la colonne x

 Si la colonne $b \neq$ la colonne x à la ligne j

$test \leftarrow$ FALSE

 Si $test ==$ TRUE

b est le 2^{ème} successeur direct

 Sinon

$cas \leftarrow 3$

a et b sont les n° de lignes où la colonne $x == 1$

 return (a, b, cas)

FIG. 3.12 – Cet algorithme retourne les successeurs directs d'une arête interne x passée en paramètre. M est la matrice topologique de l'arbre associé.

3.4.2 FASTMTM

Le second algorithme permet de calculer la multiplication d'un vecteur par la transposée de la matrice topologique d'un arbre beaucoup plus rapidement que la multiplication matricielle standard. C'est aussi le premier algorithme faisant partie de la méthode de Bryant et Waddell.

Les colonnes de X sont les mesures de coupe $\delta_1, \delta_2, \dots, \delta_k$ correspondant aux arêtes de l'arbre, donc les éléments de $\mathbf{M}^t \mathbf{d}$ sont les quantités $\delta_1^t d, \delta_2^t d, \dots, \delta_k^t d$.

La première étape dans cette méthode est le calcul de $\delta_i^t d$ pour toutes les mesures de coupe δ_i qui correspondent aux arêtes externes de l'arbre. Si e_i est une arête externe adjacente à, par exemple, le taxon x , alors

$$\delta_i^t d = \sum_{y \in L-x} d_{xy} \quad (3.1)$$

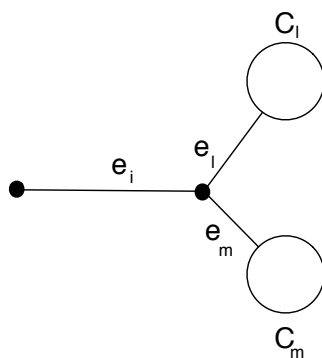


FIG. 3.13 – Forme sous laquelle une arête externe e_i d'un arbre binaire peut être représentée : les sous-arbres connectés à e_i sont représentés par des cercles.

Pour ce qui est des arêtes internes, établissons d'abord le théorème suivant :

Théorème 5 Soit e_i une arête interne de l'arbre et e_j et e_k les arêtes adjacentes au même noeud que e_i . Soit C_j , C_k et C_i les sous-arbres correspondants (voir Figure 3.14, notons que $C_i = C_j \cup C_k$). Nous avons donc

$$\delta_i^t \mathbf{d} = \delta_j^t \mathbf{d} + \delta_k^t \mathbf{d} - 2 \sum_{x \in C_j, y \in C_k} d_{xy} \quad (3.2)$$

Pour utiliser l'équation 3.2 pour calculer $\delta_i^t \mathbf{d}$ pour une arête e_i , nous avons uniquement besoin d'avoir calculé $\delta_j^t \mathbf{d}$ pour toutes les arêtes e_j adjacentes à

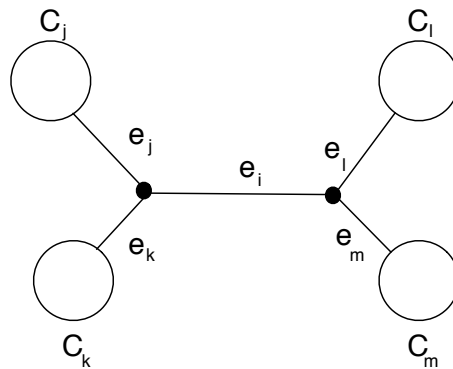


FIG. 3.14 – Forme sous laquelle une arête interne e_i d'un arbre binaire peut être représentée : les sous-arbres connectés à e_i sont représentés par des cercles.

un noeud de e_i . Donc, si nous commençons par calculer les arêtes externes, nous pouvons nous diriger petit à petit vers l'intérieur de l'arbre en appliquant répétitivement l'équation 3.2, jusqu'à ce que toutes les valeurs $\delta_i^t \mathbf{d}$ aient été calculées.

Cette méthode se formalise par l'algorithme FastMTM (raccourci pour "fast multiplication by topological matrix") décrit comme suit :

```

FASTMTM( $F, V, S$ )
remise à 0 des  $n - 3$  dernières entrées de  $V$ 
/*  $n$  étant le nombre de taxons analysés */
Pour chaque arête interne  $i$ 
     $compteur \leftarrow$  nombre de "1" dans la colonne  $i$ 
    Heap.Push( $i, compteur$ )
    /* le Heap est trié sur  $compteur$  de façon croissante */

Tant que Heap n'est pas vide
     $x \leftarrow$  Heap.Pop()
    /* On ne récupère que le n° de colonne */
     $a, b, cas \leftarrow$  SUCESSEURS( $F, x$ )
    Mettre  $S$  à jour
    /*  $S$  est une matrice qui retient les successeurs directs de chaque arête interne */
    Switch( $cas$ )
        cas 1 :  $dis \leftarrow \sum d_{ib} \quad \forall i \text{ tel que } F[i][a] == 1$ 
        cas 2 :  $dis \leftarrow \sum d_{ij} \quad \begin{cases} \forall i \text{ tel que } F[i][a] == 1 \\ \forall j \text{ tel que } F[j][b] == 1 \end{cases}$ 
        cas 3 :  $dis \leftarrow d_{ab}$ 
     $V[x] \leftarrow V[a] + V[b] - 2 \times dis$ 
return  $V$ 

```

FIG. 3.15 – Version modifiée de l'algorithme FASTMTM de Bryant et de Waddell.

3.4.3 BINARYEDGES

Le deuxième algorithme de la méthode de Bryant et Waddell repose sur le fait que la longueur d'une arête e_i peut être exprimé en terme de $\delta_i^t \mathbf{d}$, $\{\delta_{jl}^t \mathbf{d} : e_{jl} \text{ adjacent à } e_i\}$ et du nombre de taxons dans les sous-arbres correspondants (Vach 1989, Bryant 1997).

Nous devons considérer deux cas pour les formules des longueurs des arêtes : les arêtes internes et les arêtes externes. Soit e_i une arête interne dans un arbre binaire T à N taxons. Nous pouvons dessiner T dans la forme de la figure 3.14. Les arêtes adjacentes à e_i sont notées e_j, e_k, e_l et e_m . Soit N_j, N_k, N_l , et N_m les nombres de taxons dans chacun des sous-arbres correspondant aux arêtes e_j, e_k, e_l et e_m . La longueur d'arête optimale \mathbf{v}_i pour e_i est donné par :

$$\begin{aligned}
\mathbf{v}_i = & \left[\left(\frac{N}{N_j} + \frac{N}{N_k} + \frac{N}{N_l} + \frac{N}{N_m} - 4 \right) \delta_i^t \mathbf{d} \right. \\
& + \frac{N_j + N_k}{N_j N_k} ((2N_k - N) \delta_j^t \mathbf{d} + (2N_j - N) \delta_k^t \mathbf{d}) \\
& + \left. \frac{N_l + N_m}{N_l N_m} ((2N_m - N) \delta_l^t \mathbf{d} + (2N_l - N) \delta_m^t \mathbf{d}) \right] \\
& \times \frac{1}{4(N_j + N_k)(N_l + N_m)} \tag{3.3}
\end{aligned}$$

La formule pour les arêtes externes est plus simple. Soit e_i une arête externe dans un arbre binaire T . Nous pouvons dessiner T comme à la figure 3.13. Soit e_j et e_k les arêtes adjacentes, et soit N_j et N_k le nombre de taxons dans les sous-arbres correspondants. La longueur d'arête optimale pour e_i est donnée par :

$$\mathbf{v}_i = \frac{1}{4(N_j N_k)} \left[(1 + N_j + N_k) \delta_i^t \mathbf{d} - (1 + N_j - N_k) \delta_j^t \mathbf{d} - (1 - N_j + N_k) \delta_k^t \mathbf{d} \right] \tag{3.4}$$

Avec les équations 3.3 et 3.4, l'algorithme donne donc :

```

BINARYEDGES( $F, P, V, S$ )
init  $P$ 
init  $S$ 
 $V \leftarrow \text{FASTMTM}(F, V, S)$ 
/*  $S$  a été mis à jour par SUCCESEURS*/

Pour chaque arête interne  $i$ 
   $l \leftarrow S[0][i]$  (1er successeur direct de  $i$ )
   $m \leftarrow S[1][i]$  (2ème successeur direct de  $i$ )
  Trouver dans  $S$  la colonne  $j$  où  $i$  apparaît
  /* on cherche le prédécesseur de  $i$  */
  Soit  $k$  tel que  $\begin{cases} k = S[0][j] \text{ si } i = S[1][j] \\ k = S[1][j] \text{ sinon} \end{cases}$ 
  Appliquer l'équation 3.3 telle que :
     $N_j = N$  - nombre de "1" dans la colonne  $j$  de  $X$ 
     $N_k =$  nombre de "1" dans la colonne  $k$  de  $F$  ou 1 si  $k$  est une arête externe
     $N_l =$  nombre de "1" dans la colonne  $l$  de  $F$  ou 1 si  $l$  est une arête externe
     $N_m =$  nombre de "1" dans la colonne  $m$  de  $F$  ou 1 si  $m$  est une arête externe

Pour chaque arête externe  $i$ 
  Trouver dans  $S$  la colonne  $j$  où  $i$  apparaît
  /* on cherche le prédécesseur de  $i$  */
  Soit  $k$  tel que  $\begin{cases} k = S[0][j] \text{ si } i = S[1][j] \\ k = S[1][j] \text{ sinon} \end{cases}$ 
  Appliquer l'équation 3.4 telle que :
     $N_j = N$  - nombre de "1" dans la colonne  $j$  de  $F$ 
     $N_k =$  nombre de "1" dans la colonne  $k$  de  $F$  ou 1 si  $k$  est une arête externe
return  $P$ 

```

FIG. 3.16 – Version modifiée de l'algorithme **BINARYEDGES** de Bryant et de Waddell.

Chapitre 4

Résultats

4.1 Données de tests

Nous avons testé notre implémentation sur un ensemble d'une centaine d'instances différentes. Cette dernière est divisée en 4 groupes de 25 instances selon le nombre de séquences ADN(représentant chacune un taxon) que ce groupe contient. Le premier groupe contient des instances de 20 séquences, le second de 50 séquences, le troisième de 100 séquences et finalement le dernier groupe contient 25 instances de 500 séquences. Toutes ces séquences ADN sont codées entre 100 et 500 caractères et ont été générées aléatoirement.

Les matrices de distance de ces instances ont été calculées par le logiciel PAUP décrit au Chapitre 4.2. PAUP servira aussi d'indicateur pour nos tests, ce logiciel était considéré jusqu'il y a peu comme le meilleur outil pour l'analyse phylogénétique.

La machine que nous allons utiliser pour nos tests est un PowerPC G4 cadencé à 1.67 GHz, avec 1 Go de SDRAM DDR2 sous Mac OS X version 10.4.6 et le compilateur gcc version 4.0.1.

4.2 PAUP

L'objectif du développement de PAUP est de fournir un programme pour l'analyse phylogénétique qui peut inclure le plus de fonctions possibles dans un seul et même logiciel. PAUP contient un des programmes de parcimonie les plus sophistiqués qui soient. Les fonctions de reconstruction d'arbre dans la version actuelle de PAUP incluent le Maximum de Parcimonie et le Maximum de Vrai-

semblance. Chaque programme de reconstruction d'arbres permet une grande variété d'options. L'option de Maximum de Parcimonie inclut la spécification de n'importe quel schéma de pondération de caractères. Les options de distance permettent de choisir parmi les procédures suivantes : Neighbor-Joining, Evolution Minimum, Fitch-Margoliash et UPGMA. Selon les notes qui accompagnent PAUP test version 4, PAUP trouve généralement des arbres avec une vraisemblance plus grande que PHYLIP : le logiciel phylogénétique considéré comme étant le plus répandu. Ceci est dû d'une part aux réarrangements d'arbre de PAUP qui permettent une recherche plus vaste et d'autre part à ses critères de convergence pour la longueur des branches qui sont plus strictes. Avec n'importe quelle méthode de construction d'arbre, PAUP permet une panoplie variée d'options de recherche. Celles-ci incluent entre autre la spécification d'algorithmes pour générer un arbre initial (ou arbre de départ) : Neighbor-Joining, Addition par pallier ou un arbre donné en input.

PAUP est considéré à ce jour comme étant le logiciel phylogénétique le plus complet.

Ce logiciel utilise trois algorithmes de réarrangement d'arbre ([27]). Il s'agit du :

1. Nearest Neighbor Interchange (NNI) que nous avons déjà décrit.
2. Subtree Pruning-Regrafting (SPR).
3. Tree Bisection-Reconnection (TBR).

Le mouvement NNI ayant déjà été décrit, nous ne nous y attarderons pas davantage.

Dans le SPR, un sous-arbre est détaché de l'arbre (par exemple le sous-arbre contenant les feuilles A et B à la Figure 4.1-(i)). Le sous-arbre est alors rattaché à une location différente sur l'arbre.

Dans le TBR, l'arbre est coupé en deux le long d'une branche, produisant deux sous-arbres disjoints. Les sous-arbres sont ensuite reconnectés en joignant une paire de branches ; une de chaque sous-arbre (Figure 4.1-(ii)).

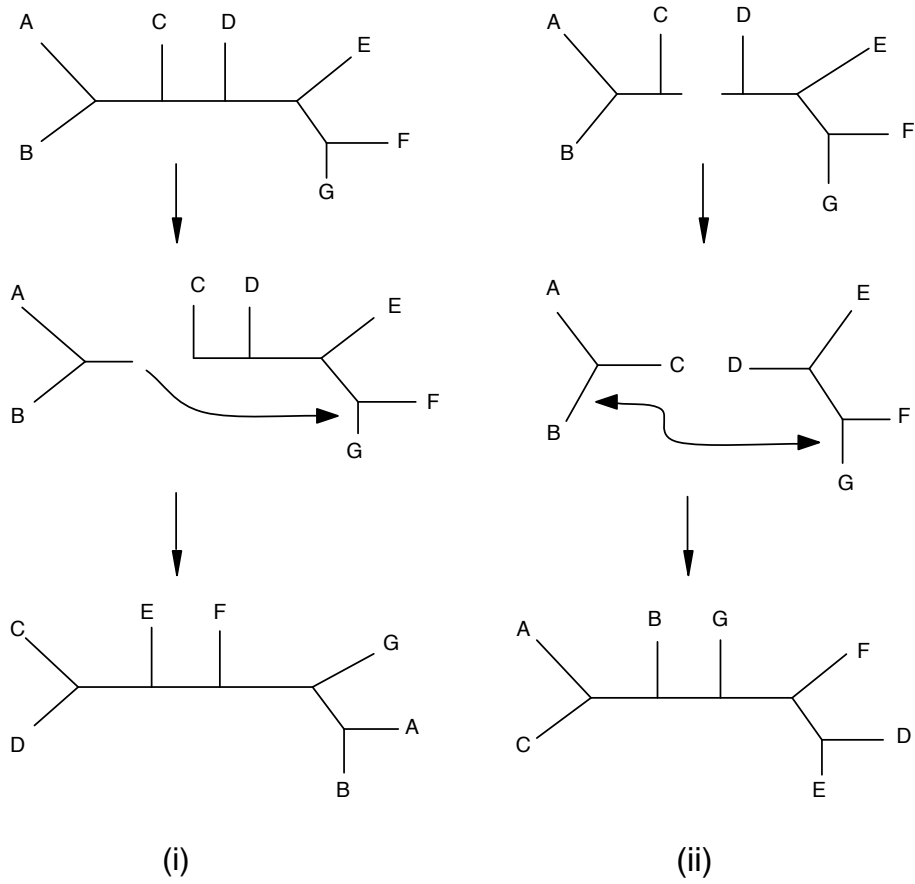


FIG. 4.1 – Un exemple de réarrangement avec (i) SPR et (ii) TBR.

4.3 Analyse

Pour l'analyse de nos résultats, nous allons utiliser un outil statistique particulier : la boîte à dispersion ou boîte à moustaches. La boîte à moustaches, traduction de *Box & Whiskers Plot*, est une invention de Tukey (1977) et est généralement utilisée pour représenter différents jeux de données par une approche statistique. Une boîte représente l'écart de données entre le premier et le troisième quartile (respectivement 25% et 75% des données). La médiane est représentée par une ligne horizontale dans cette boîte. Les "moustaches" s'étendent jusqu'au point de donnée le plus extrême qui ne soit pas un *outlier* ou valeur atypique, c'est-à-dire une valeur qui ne dépasse pas 1,5 fois l'écart interquartile de la boîte. Les *outliers* sont indiqués par des cercles.

Comme nous désirons effectuer une comparaison entre les différents algorithmes (ou mouvements) que nous avons décrits, nous devons avant tout déterminer une valeur de départ, disons zéro, et un intervalle de mesure. Instance après instance, nous pouvons assigner la valeur zéro à l'algorithme qui a trouvé la meilleure valeur, et nous pouvons assigner la valeur un à l'algorithme qui a trouvé la plus mauvaise valeur. Chaque valeur générée par chaque algorithme pour chaque instance peut être représentée dans le diagramme en utilisant la formule suivante :

$$\frac{x_j - \min_i}{\max_i - \min_i} \forall i \in \text{Instances}, \forall j \in \text{Algorithmes} \quad (4.1)$$

où x_j est la meilleure valeur trouvée par l'algorithme j ; \min_i et \max_i sont respectivement la meilleure et la plus mauvaise valeur trouvée par tous les algorithmes sur l'instance i . De cette façon, nous pouvons avoir une analyse claire des résultats trouvés par les différents algorithmes.

Le critère d'évaluation des différents mouvements est le temps. Pour les instances avec 20 taxons, nous avons fait tourner les différentes configurations de notre algorithme (*i.e.* les différents mouvements) pendant 1 seconde chacune, pour celles avec 50 taxons pendant 5 secondes, pour les instances avec 100 taxons les tests ont duré 60 secondes par instance, enfin les tests pour les instances avec 500 taxons ont duré 600 secondes chacun.

Nous allons commencer par une analyse de la configuration dite "statique" de notre implémentation. Ensuite nous verrons la variante "dynamique" de celle-ci, suivie d'une comparaison statique/dynamique. Enfin, nous terminerons par une comparaison de nos résultats avec le logiciel PAUP.

4.3.1 Recherche Taboue statique

Cette Recherche Taboue est appelée statique car nous allons utiliser une taille de liste taboue identique pour toutes les itérations égale à \sqrt{n} où n est le nombre de taxons considérés, plutôt que d'utiliser une taille de liste taboue qui change à chaque itération comme nous l'avons suggéré à la Section 3.2.4.

A la Figure 4.2 nous pouvons voir les résultats obtenus pour les différents mouvements pour une Recherche Taboue statique pour une reconstruction à 20 taxons. Nous constatons que le mouvement LE, sans surprise, obtient les moins bons résultats. Les mouvements NNI, IELE et IE obtiennent des résultats plus ou moins similaires tandis que le mouvement IEE est un peu en retrait de ces derniers. Le mouvement MIX, qui est simplement une combinaison des mouvements NNI et IE obtient des résultats très légèrement supérieurs en moyenne.

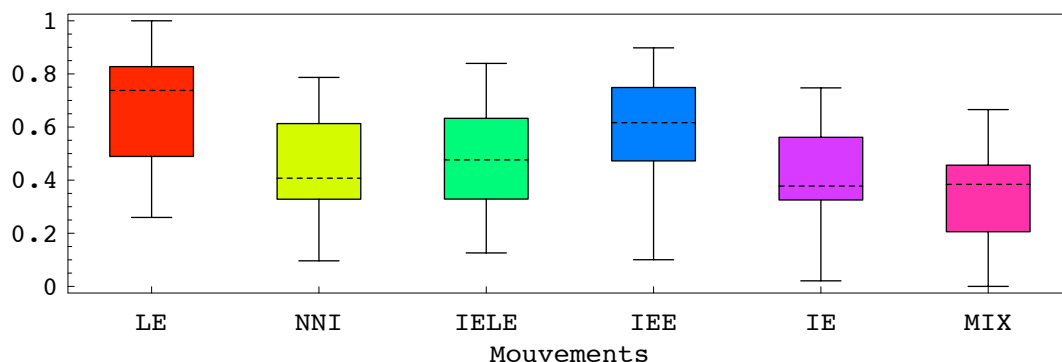


FIG. 4.2 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue statique pour 20 taxons.

Pour ce qui est de la Recherche Taboue statique avec 50 taxons (Figure 4.3), les résultats sont un peu similaires aux précédents. Le mouvement LE obtient toujours les moins bons résultats, tandis que les résultats des mouvements NNI et IELE restent très proches. Le mouvement IEE est toujours faiblement en retrait par rapport aux deux mouvements précédents tandis que le mouvement IE se rapproche des performances du mouvement MIX. Nous pouvons même affirmer que ces deux derniers obtiennent des résultats équivalents en moyenne.

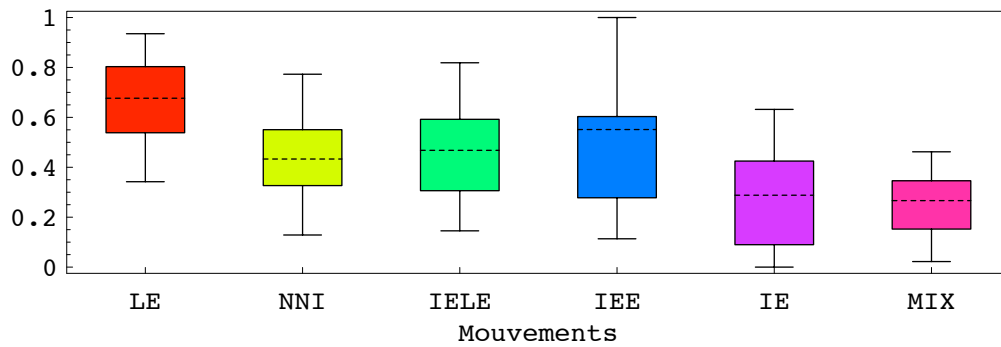


FIG. 4.3 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue statique pour 50 taxons.

A la Figure 4.4, nous avons les résultats pour la Recherche Taboue statique pour 100 taxons. Ici le mouvement LE est loin derrière. Les mouvements IELE et IEE obtiennent des résultats assez proches, tandis que les résultats du mouvement NNI sont un peu supérieurs à ces derniers. Le mouvement IE est sensiblement meilleur que le mouvement NNI, tandis que le mouvement MIX confirme sa supériorité.

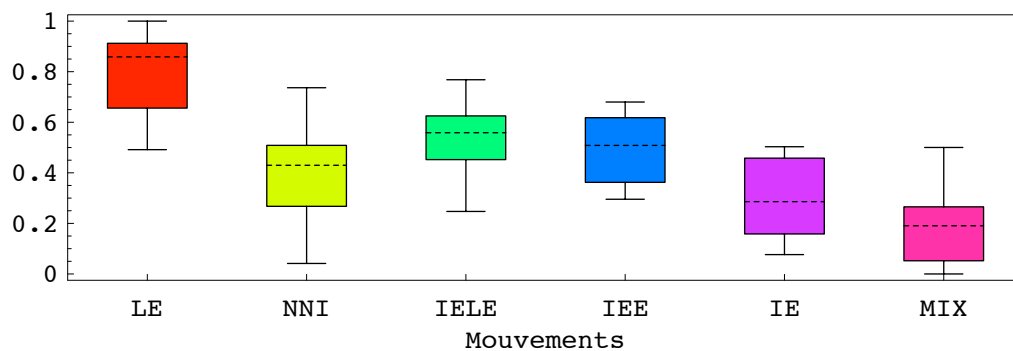


FIG. 4.4 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue statique pour 100 taxons.

A la Figure 4.5, nous constatons les résultats pour 500 taxons. Ici les résultats sont sensiblement différents de ce qui a été observé précédemment. En effet, les mouvements LE, IELE et IEE obtiennent des résultats plus ou moins équivalents, tandis que les mouvements IE et MIX obtiennent eux des résultats faiblement supérieurs. La grosse surprise vient du mouvement NNI qui obtient des résultats sensiblement supérieurs à tous les autres mouvements. Après décortiqués des résultats, il a été observé que le mouvement NNI permettait beaucoup plus de réarrangements d'arbre que les autres (hormis LE) car beaucoup moins complexe, ce qui explique ces résultats.

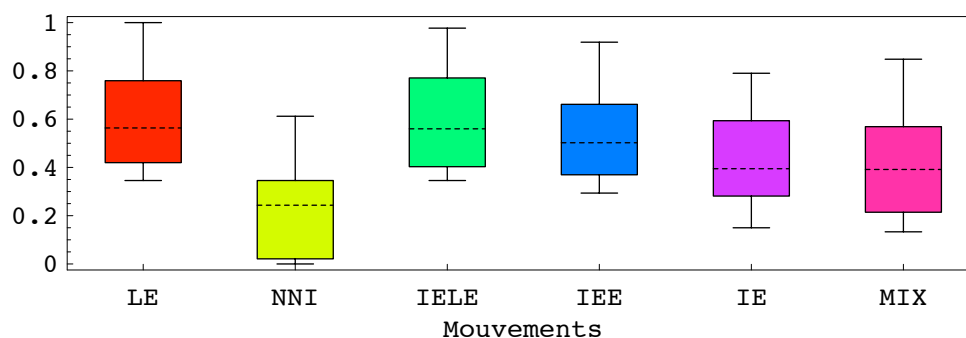


FIG. 4.5 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue statique pour 500 taxons.

4.3.2 Recherche Taboue dynamique

Par dynamique, nous entendons une liste de tabous comme suggérée à la Section 3.2.4.

A la Figure 4.6, nous pouvons observer les résultats obtenus pour des instances de 20 taxons. Ceux-ci sont très similaires à ceux de la Recherche Taboue statique (aussi pour 20 taxons). Le mouvement LE obtient encore une fois les moins bons résultats, les mouvements NNI, IELE et IE ont des résultats similaires tandis que le mouvement IEE est légèrement en retrait. Le mouvement MIX quant à lui est très faiblement supérieur à tous ces mouvements mais sans aucune certitude statistique.

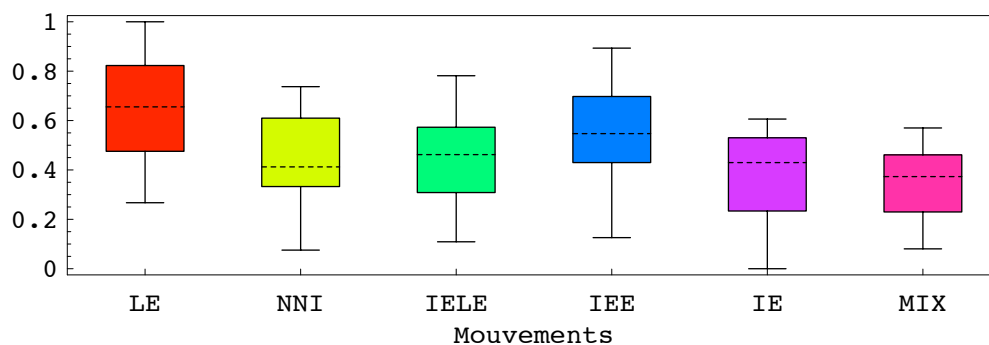


FIG. 4.6 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue dynamique pour 20 taxons.

Idem encore concernant la configuration statique par rapport à la configuration dynamique pour 50 taxons. A la Figure 4.7, nous constatons quasiment la même chose que pour la Recherche Taboue statique. Le mouvement LE reste le moins efficace, vient ensuite le mouvement IEE, légèrement inférieur aux mouvements NNI et IELE dont les résultats sont encore une fois assez similaires. Le mouvement IE et le mouvement MIX obtiennent une fois de plus les meilleurs résultats avec cependant un léger avantage pour ce dernier.

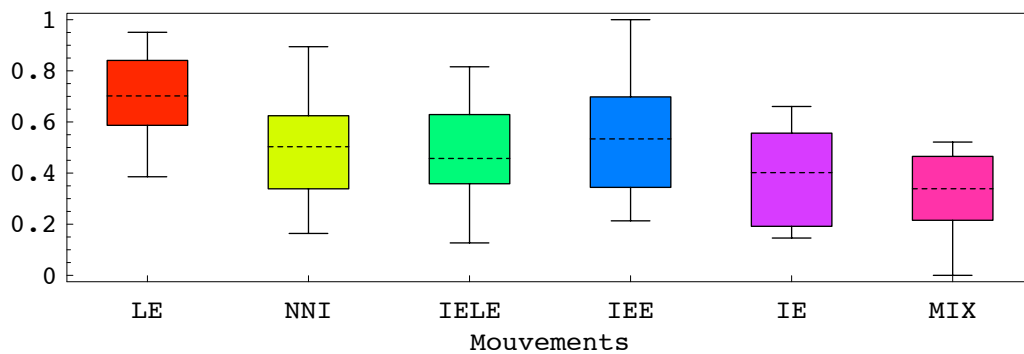


FIG. 4.7 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue dynamique pour 50 taxons.

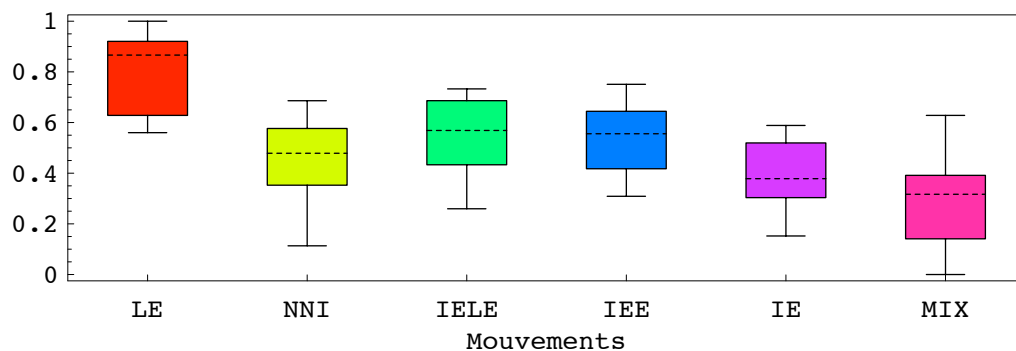


FIG. 4.8 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue dynamique pour 100 taxons.

Pour la recherche avec des instances de 100 taxons (Figure 4.8), la Recherche Taboue dynamique donne des résultats un peu différents de ce qui a précédé. Sans surprise, le mouvement LE obtient les moins bons résultats. Les mouvements IELE et IEE obtiennent des résultats similaires tandis que le mouvement NNI les précède de peu. Le mouvement IE obtient des résultats légèrement supérieurs à ce dernier. Enfin, le mouvement MIX obtient à nouveau les résultats les plus performants.

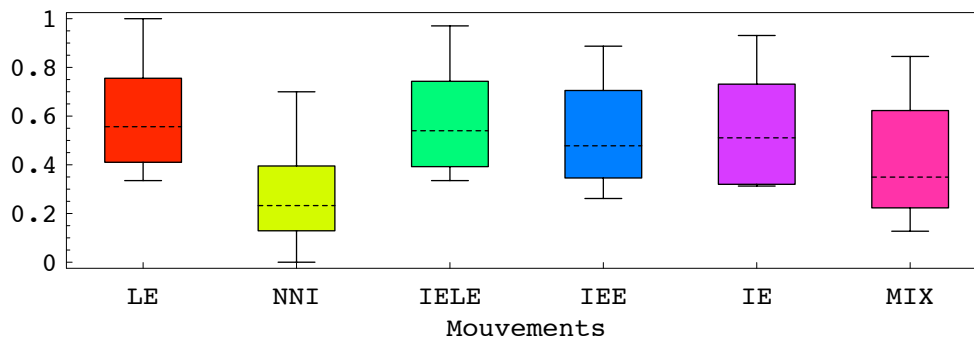


FIG. 4.9 – Diagramme en boîtes à dispersion pour les différents mouvements utilisés par la Recherche Taboue dynamique pour 500 taxons.

Finalement, nous pouvons observer à la Figure 4.9 que les résultats des instances avec 500 taxons sont approximativement les mêmes que ceux de la Recherche Taboue statique. En effet, les mouvements LE, IELE, IEE et IE obtiennent tous des résultats équivalents, tandis que les résultats du mouvement MIX sont légèrement supérieurs. Le mouvement NNI obtient, comme pour la Recherche Taboue statique, les meilleurs résultats pour les instances avec autant de taxons. Encore une fois, il s'agit du nombre de réarrangements d'arbre plus important que NNI effectue par rapport aux autres mouvements (exception faite de LE).

4.3.3 Comparaison recherche statique et recherche dynamique

Pour la comparaison des configurations statiques et dynamiques de la Recherche Taboue, nous avons pris en compte ce qui semblait être le meilleur mouvement en moyenne : le mouvement MIX. Pour effectuer cette comparaison, nous allons reprendre les résultats obtenus aux deux sections précédentes (Sections 4.3.1 et 4.3.2). Cependant vu la trop grande divergence avec les autres résultats, nous ne comparerons pas ceux obtenus pour les instances avec 500 taxons.

A la Figure 4.10, nous pouvons constater que, bien que très proches, les résultats sont très légèrement en faveur de la configuration statique, même si nous n'avons aucune évidence statistique. Nous pouvons en dire autant concernant les instances avec 50 taxons (Figure 4.11). Concernant les instances avec 100 taxons (Figure 4.12) l'avantage est plus marqué pour la configuration statique.

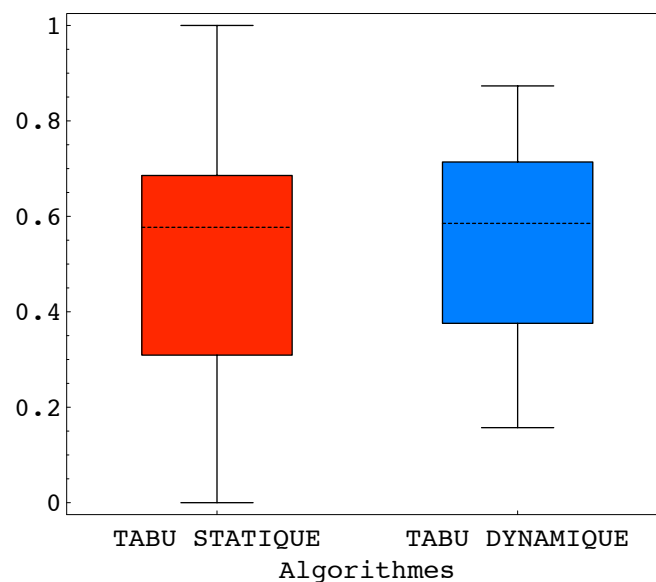


FIG. 4.10 – Diagramme en boîtes à dispersion comparant les configurations statiques et dynamiques de la Recherche Taboue pour 20 taxons.

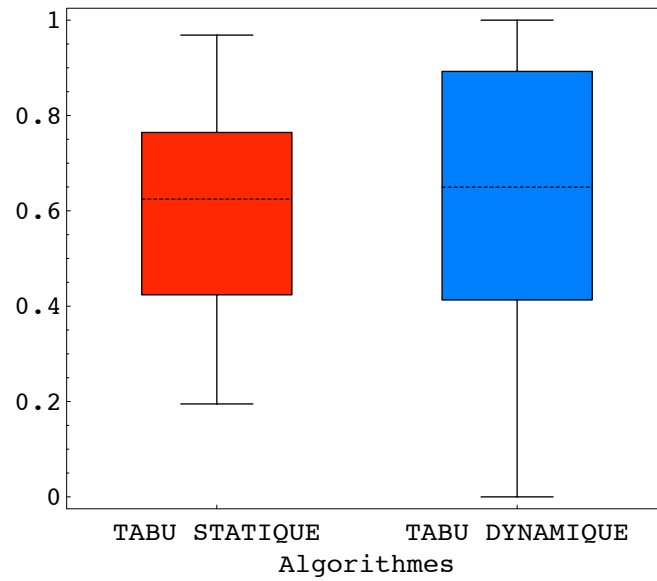


FIG. 4.11 – Diagramme en boîtes à dispersion comparant les configurations statiques et dynamiques de la Recherche Taboue pour 50 taxes.

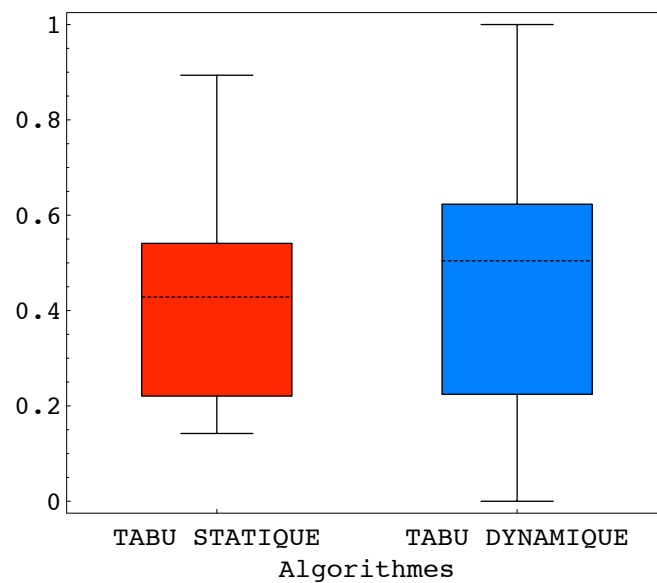


FIG. 4.12 – Diagramme en boîtes à dispersion comparant les configurations statiques et dynamiques de la Recherche Taboue pour 100 taxes.

4.3.4 Comparaison avec PAUP

Pour notre comparaison avec le logiciel PAUP, nous avons repris le même jeu d'exemples que précédemment. Nous avons par ailleurs utilisé ce qui semblait être la meilleure configuration de notre implémentation, à savoir une liste taboue statique et la combinaison des algorithmes NNI et IE pour les mouvements : MIX. La durée des tests pour chaque instance est, quant à elle, similaire à l'analyse interne de notre algorithme : pour les instances avec 20 taxons, nous avons fait tourner les deux programmes pendant 1 seconde chacun, pour celles avec 50 taxons pendant 5 secondes et pour les instances avec 100 taxons les tests ont duré 60 secondes par instance. Enfin, PAUP construit son arbre de départ également avec la méthode du Neighbor-Joining.

Pour des instances avec peu de taxons (Figure 4.13), Tabu Search et PAUP sont relativement similaires. La plupart du temps les résultats sont équivalents, même si, à de rares occasions, PAUP obtient des résultats très faiblement supérieurs, ce qui explique la médiane de PAUP très légèrement inférieure à celle de Tabu Search.

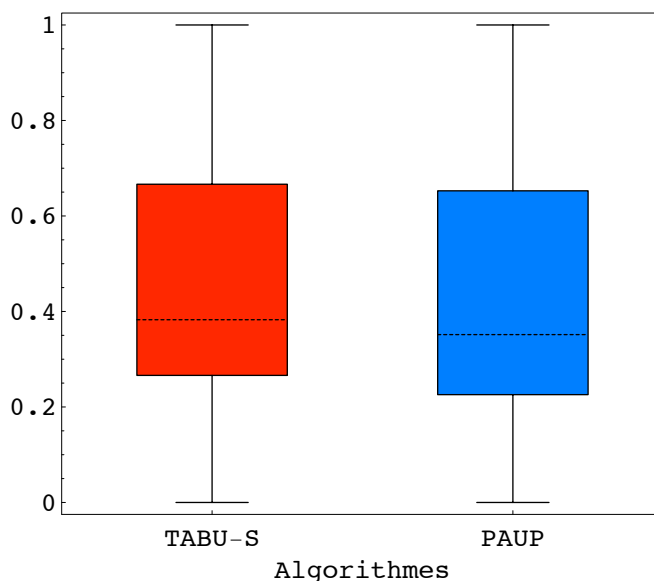


FIG. 4.13 – Diagramme en boîtes à dispersion relatif à PAUP et la meilleure configuration de la Recherche Taboue pour 20 taxons.

Pour la comparaison avec 50 et 100 taxons (Figure 4.14 et 4.15), la constatation est la même : les deux programmes sont très similaires avec toujours un léger avantage pour PAUP, même si cet avantage est plus évident dans ce cas-ci qu'avec les résultats obtenus pour les exemples avec peu de taxons.

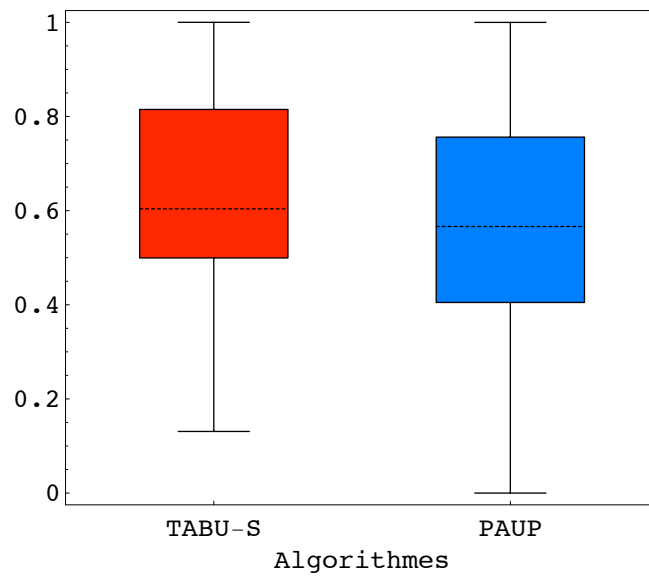


FIG. 4.14 – Diagramme en boîtes à dispersion relatif à PAUP et la meilleure configuration de la Recherche Taboue pour 50 taxons.

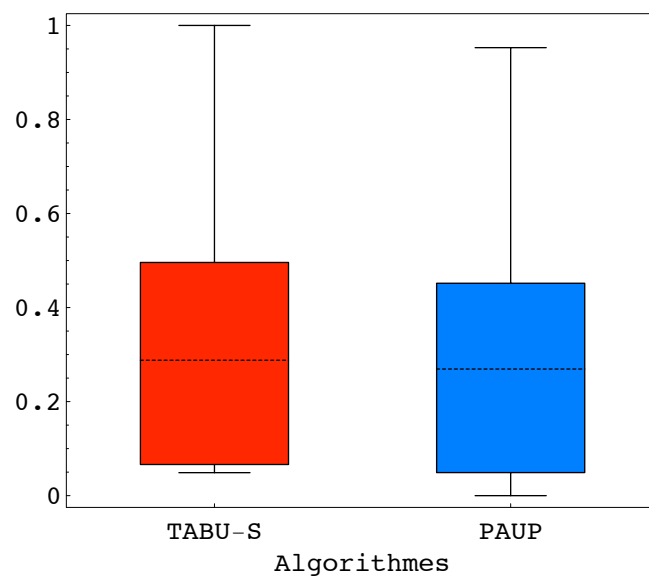


FIG. 4.15 – Diagramme en boîtes à dispersion relatif à PAUP et la meilleure configuration de la Recherche Taboue pour 100 taxons.

4.4 Conclusion

Après analyse des différents résultats obtenus aux sections précédentes, nous pouvons observer que :

- la combinaison des mouvements IE et NNI, MIX, obtient les meilleurs résultats en moyenne pour notre implémentation, que la configuration de la liste taboue soit statique ou dynamique ; sauf, comme nous l’avons vu, pour les instances avec un nombre très élevé de taxons (ici 500). Ceci est dû au fait que les mouvements que nous avons développés, bien qu’efficaces la plupart du temps, sont trop “lourds” (complexité plus élevée que NNI, gestion de listes, . . .etc.). Nous pouvons donc conclure que pour un nombre très élevé de taxons, il est préférable d’avoir un mouvement “léger” plutôt qu’un mouvement plus efficace mais plus exigeant en terme de temps de calcul.
- la configuration statique est légèrement plus efficace que la configuration dynamique. Mais la façon dont le métaheuristique explore l’espace de solution ne semble pas tenir toutes ses promesses. En effet, si on observe le cheminement de plusieurs recherches avec notre algorithme, on a l’impression que celui-ci agit un peu comme une simple recherche aléatoire. Aussi est-il possible que la façon dont nous avons implémenté la liste taboue ne soit pas adaptée au problème, bien que courante pour beaucoup d’autres implémentations.
- l’utilisation du bloc F permet d’obtenir une implémentation avec des résultats relativement corrects. Ceci ouvre une nouvelle approche pour résoudre le problème de l’inférence phylogénétique. Les résultats obtenus par notre implémentation sont satisfaisants dans le sens où, bien que légèrement inférieurs, ceux-ci sont pratiquement équivalents à ceux obtenus par le logiciel PAUP. Cependant nous avons de bonnes raisons de croire que les améliorations restent nombreuses.

4.5 Futurs travaux

L’algorithme Tabu Search que nous avons développé obtient d’assez bons résultats, cependant celui-ci peut encore être amélioré. En effet, ce qui semble faire sa faiblesse est la façon dont il explore l’espace de solution. Pour résoudre ce problème, nous pouvons envisager d’implémenter une méthode de recherche

taboue réputée plus efficace : la Recherche Taboue Réactive ([1]) imaginée par Roberto Battiti et Giampietro Tecchioli. Cependant celle-ci s'avère difficile à implémenter de manière efficace pour notre problème. En effet, cette méthode utilise beaucoup de structures de données annexes qui peuvent alourdir l'algorithme.

Une autre possibilité d'amélioration pourrait se situer au niveau de la rapidité d'exécution. Nous avons vu au chapitre 3.4 la méthode utilisée pour l'évaluation d'une solution. Pour améliorer celle-ci il serait possible de se passer de la fonction `SUCCESEURS` appelée à chaque itération, et de ne l'appeler qu'une fois à l'initialisation. En fonction du mouvement effectué, il est possible de modifier directement la structure de donnée manipulée par la fonction `SUCCESEURS`, structure utilisée pour le calcul des solutions trouvées.

Les améliorations possibles sont multiples et laissées à l'imagination du chercheur.

Bibliographie

- [1] BATTITI, R., AND TECCHIOLLI, G. The reactive tabu search. *ORSA Journal on Computing* 6, 2 (1994), 126–140.
- [2] BAXENIS, A., AND OUELLETE, F. *Bioinformatics : A practical guide to the Analysis of Genes and Proteins*, second ed. Wiley-Interscience, 2001.
- [3] BERRY, V. *Méthodes et algorithmes pour reconstruire les arbres de l'évolution*. PhD thesis, Université des Sciences et Techniques du Languedoc, 1997.
- [4] BRYANT, D., AND WADDELL, P. Rapid evaluation of least-squares and minimum-evolution criteria on phylogenetic trees. *Molecular Biology and Evolution*, 15(10) (1998), 1346–1359.
- [5] CATANZARO, D. Metaheuristic approaches for inferring phylogenies. Master's thesis, IRIDIA - Université Libre de Bruxelles, 2004.
- [6] CATANZARO, D., PESENTI, R., AND MILINKOVITCH, M. A branch-and-bound algorithm for minimum evolution principle. Submitted to *Bioinformatics*.
- [7] DARLU, P., AND TASSY, P. *La reconstruction phylogénétique, concepts et méthodes*. Masson, 1993.
- [8] DAY, W. H. E. Computational complexity of inferring phylogenies by dissimilarity matrices. *Bulletin of Mathematical Biology* 49, 4 (1987), 461–467.
- [9] DENIS, F., AND GASCUEL, O. On the consistency of the minimum evolution principle of phylogenetic inference. *Discrete Applied Mathematics* 127, 1 (2003), 63–77.
- [10] DESPER, R., AND GASCUEL, O. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *Journal of Computational Biology* 9, 5 (2002), 687–706.
- [11] DRÉO, J., PÉROWSKI, A., SIARRY, P., AND TAILLARD, E. *Métaheuristiques pour l'optimisation difficile*. Eyrolles, Sept. 2003.

- [12] FELSENSTEIN, J. Distance methods for inferring phylogenies : a justification. *Evolution* 38, 1 (1984), 16–24.
- [13] FELSENSTEIN, J. *Inferring Phylogenies*. Sinauer Associates, Sept. 2004.
- [14] GASCUEL, O. *Mathematics of evolution and phylogeny*. Oxford University press, 2005.
- [15] GASCUEL, O., BRYANT, D., AND DENIS, F. Strengths and limitations of the minimum-evolution principle. *Systematic Biology* 50, 5 (2001), 621–627.
- [16] GLOVER, F. Tabu search - part I. *ORSA Journal on Computing* 1, 3 (1989), 190–206.
- [17] GLOVER, F. Tabu search - part II. *ORSA Journal on Computing* 2, 1 (1990), 4–32.
- [18] GLOVER, F., AND KOCHENBERGER, G. *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2002.
- [19] LEMMON, A. R., AND MILINKOVITCH, M. C. The metapopulation genetic algorithm : An efficient solution for the problem of large phylogeny estimation. *PNAS* 99, 16 (2002), 10516–10521.
- [20] LI, W.-H. *Molecular Evolution*. Sinauer Associates, July 1997.
- [21] LÖYTYNOJA, A., AND MILINKOVITCH, M. C. A hidden markov model for progressive multiple alignment. *Bioinformatics* 19, 12 (2003), 1506–1513.
- [22] RZHETSKY, A., AND NEI, M. A simple method for estimating and testing minimum-evolution trees. *Mol. Biol. Evol.* 9 (1992), 945–967.
- [23] RZHETSKY, A., AND NEI, M. Statistical properties of the ordinary least-squares, generalized least-squares, and minimum-evolution methods of phylogenetic inference. *Journal of Molecular Evolution* 35 (1992), 367–375.
- [24] RZHETSKY, A., AND NEI, M. Theoretical foundations of the minimum evolution method of phylogenetic inference. *Molecular Biology and Evolution* 10 (1993), 1073–1095.
- [25] SAITOU, N., AND NEI, M. The neighbor-joining method : a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.* 4 (1987), 406–425.
- [26] SEMPLE, C., AND STEEL, M. *Phylogenetics*. Oxford University Press, 2003.
- [27] SWOFFORD, D., AND BEGLE, D. *PAUP 3.1 Manual*.
- [28] TAILLARD, E. D. Robust taboo search for the quadratic assignment problem. *Parallel Computing* 17, 4-5 (1991), 443–455.