UNIVERSITÉ DU QUÉBEC À MONTRÉAL

UN NOUVEL ALGORITHME POUR L'INFÉRENCE DE RÉSEAUX D'HYBRIDATION

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN INFORMATIQUE

PAR
MATTHIEU WILLEMS

MAI 2012

REMERCIEMENTS

Je tiens tout d'abord à remercier chaleureusement mon directeur de recherche, Vladimir Makarenkov, qui a su motiver mes recherches tout en me laissant beaucoup de liberté. Je le remercie aussi pour sa relecture très précise de ce mémoire.

Je remercie également Hydro-Québec pour la bourse que j'ai reçue durant ma première année de maîtrise, ainsi que la Fondation de l'UQÀM qui m'a sélectionné pour recevoir cette bourse. J'adresse également ma gratitude au CRSNG pour m'avoir octroyé une bourse pour ma deuxième année de maîtrise dans le cadre de son programme de bourses d'études supérieures.

J'exprime enfin mes plus vifs remerciements à tous les professeurs du département d'informatique de l'UQÀM qui m'ont accompagné durant ma maîtrise, ainsi qu'à mes collègues du laboratoire de bioinformatique.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Depuis une quarantaine d'années, de nombreux algorithmes et logiciels ont été développés pour inférer des arbres phylogénétiques. Cependant, certains phénomènes biologiques comme l'hybridation ou le transfert latéral de gènes ne peuvent pas être représentés sous la forme d'un arbre. On utilise ainsi de plus en plus des réseaux phylogénétiques. Les recherches sur ce sujet ont débuté il y a une dizaine d'années et les outils disponibles actuellement pour déterminer des réseaux phylogénétiques sont beaucoup moins performants que dans le cas des arbres. L'objectif principal de mes recherches consiste ainsi à développer une nouvelle méthode pour inférer des réseaux phylogénétiques en se limitant au cas de l'hybridation. J'ai ainsi développé un nouvel algorithme qui permet de retrouver tous les arbres phylogénétiques et de détecter tous les hybrides entre des branches voisines. Quand les parents des hybrides ne sont pas voisins, il trouve les bons hybrides avec des taux de détection proches de 100%, mais il trouve trop d'hybrides et n'identifie pas toujours les bons parents de ces hybrides. Ce nouvel algorithme est itératif et est basé sur le critère des moindres carrés qui permet de déterminer la configuration optimale à chaque itération. Il a été implémenté dans le langage C++ et plusieurs centaines de simulations ont été effectuées pour tester ses fonctionnalités.

Mots clés : arbre phylogénétique, inférence phylogénétique, réseau réticulé, hybridation, critère des moindres carrés.

INTRODUCTION

Les travaux de Darwin sur l'évolution des espèces publiés en 1859 (Darwin, 1859) et la découverte de l'acide désoxyribonucléique (ADN) par Watson et Crick (1953) ont permis le développement de la phylogénie moléculaire, dont le but est de reconstituer « l'Arbre de la Vie » à partir de données moléculaires. Cet arbre est censé représenter le processus dynamique de la diversification des espèces : les feuilles correspondent aux espèces étudiées, les nœuds représentent les ancêtres virtuels, alors que les branches identifient les liens de filiation. Cependant, certains processus très importants ne peuvent pas être représentés correctement par le modèle classique de l'arbre phylogénétique. On doit alors utiliser des réseaux qui sont des structures plus complexes que les arbres et on parle ainsi d'évolution réticulée (voir (Makarenkov, Kevorkov et Legendre, 2006) pour une vue d'ensemble sur le sujet). Par exemple, le transfert horizontal de gènes est un phénomène permettant aux espèces de bactéries de s'échanger des gènes. Des réticulations apparaissent également chez les plantes comme résultat de l'hybridation. Un des enjeux majeurs de la bioinformatique est ainsi de développer des outils efficaces pour reconstituer de tels réseaux à partir de différentes données sur un ensemble d'espèces. En me basant sur l'algorithme *Neighbor Joining* (Saitou et Nei, 1987) servant à reconstruire les arbres phylogénétiques classiques, j'ai ainsi conçu un nouvel outil informatique pour inférer des reseaux phylogénétiques qui prennent en compte des phénomènes d'hybridation.

Dans un premier temps, je donnerai les principales définitions biologiques et bioinformatiques nécessaires pour comprendre l'enjeu de mes recherches, je rappellerai les principales méthodes d'inférence d'arbres phylogénétiques, et je ferai une revue de littérature sur le cas plus complexe des réseaux phylogénétiques.

Dans un deuxième temps, j'expliquerai en détails l'algorithme *Neigbor Joining* sur lequel je me suis basé, et je décrirai l'algorithme principal que j'ai développé en explicitant les calculs qui m'ont permis de le concevoir. Je donnerai également deux exemples qui permettent de comprendre comment mon nouvel algorithme retrouve certains réseaux phylogénétiques.

La dernière partie de mon mémoire sera consacrée aux différents résultats obtenus à l'aide de la nouvelle méthode que j'ai développée. Je vérifierai tout d'abord certaines fonctionnalités de mon algorithme, notamment sa capacité à retrouver tous les arbres phylogénétiques additifs, ainsi que les hybrides entre voisins. Je donnerai également des données statistiques sur les résultats obtenus pour des réseaux d'hybridation construits à partir d'arbres aléatoires de différentes tailles auxquels on rajoute un ou plusieurs phénomènes d'hybridation.

CHAPITRE I

PRINCIPALES DÉFINITIONS ET REVUE DE LITTÉRATURE

La phylogénèse étudie la reconstruction de l'histoire évolutive des êtres vivants. Le terme phylogenèse (du grec phulon, signifiant « race, tribu ») a été introduit par Haeckel en 1860, qui l'a défini comme « l'histoire du développement paléontologique des organismes par analogie avec l'ontogénie ou histoire du développement individuel ». Un arbre est dit arbre phylogénétique, phylogénie ou X-arbre (Barthélemy et Guénoche, 1991) si, dans l'analyse des caractères sur laquelle il repose, le concept de « descendance des espèces avec modification de leurs caractères » a été utilisé. Ce dernier concept signifie que les caractères sont transmis d'une génération à l'autre à travers les mécanismes de l'hérédité impliquant leurs éventuelles modifications (par exemple les mutations). Un arbre phylogénétique est une représentation graphique de la phylogenèse d'un groupe d'espèces (ou de taxons).

Un arbre phylogénétique est composé de quatre principaux éléments. Les feuilles ou nœuds externes représentent les espèces pour lesquelles on dispose de distances d'évolution. Les branches (ou arêtes) définissent les relations entre les taxons en termes de descendance. Les nœuds internes sont associés à des ancêtres virtuels. Et enfin, la racine représente l'ancêtre commun de toutes les espèces considérées.

Le degré d'un nœud est le nombre d'arêtes adjacentes à ce nœud. Si ce degré est supérieur à trois, ce nœud est dit non résolu (signifiant la divergence simultanée ou l'incertitude). On distingue deux types d'arbres : les arbres enracinés et les arbres non enracinés. Un

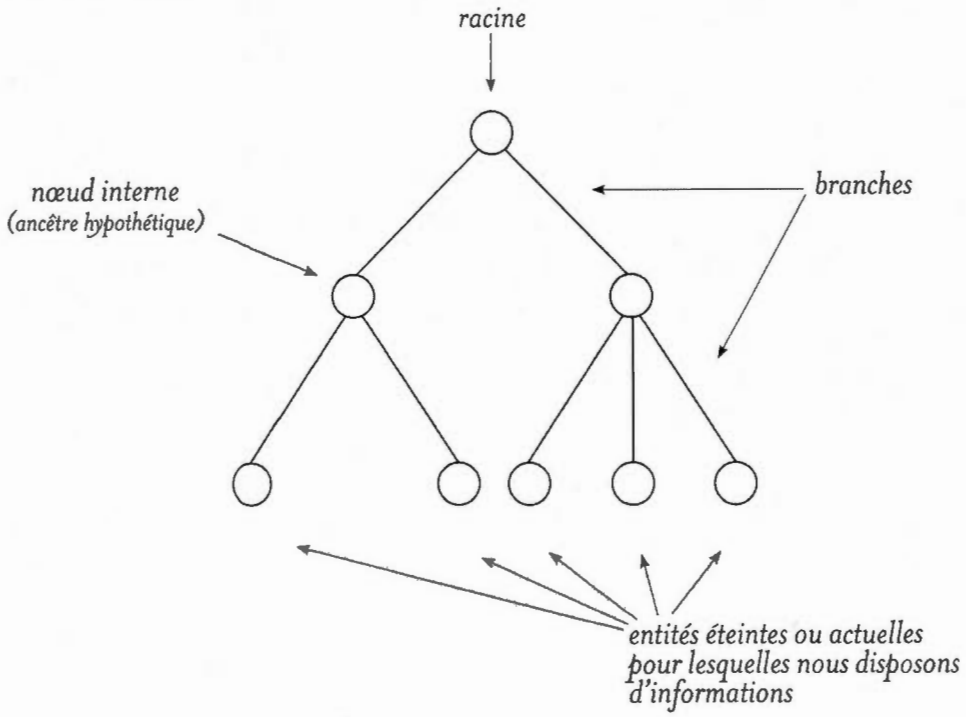


Figure 1.1 Modèle de base introduisant un arbre phylogénétique.

arbre enraciné est orienté et cette orientation correspond au sens d'évolution. Il permet donc de définir une relation *ancêtre - descendant* entre deux nœuds successifs. Dans un arbre non-enraciné, la notion de temps n'existe pas et on ne peut plus définir la relation ancêtre - descendant au niveau des nœuds internes. Ce type d'arbres peut être utilisé lorsque l'on s'intéresse à la classification d'un groupe d'espèces sans considérer le sens d'évolution. La figure 1.1 présente un exemple d'arbre enraciné.

La fin de ce chapitre contient un glossaire des principaux termes biologiques utilisés dans ce mémoire.

1.1 La reconstruction d'arbres phylogénétiques

La reconstruction d'un arbre phylogénétique commence par l'analyse des séquences nucléotidiques ou d'acides aminés associées aux espèces étudiées. Une séquence nucléotidique (assemblage linéaire de nucléotides) représente l'ADN (acide désoxyribonucléique) et est composée de quatre types de base. Les cytosines (C) et thymines (T), qui font partie de la famille des pyrimidines, et les adénines (A) et guanines (G), qui font partie de la famille des purines. Une séquence d'ADN peut représenter un gène qui sera exprimé en une protéine (séquence d'acides aminés). Trois approches principales ont été développées pour construire des arbres phylogénétiques : la phénétique, la cladistique et la probabiliste.

L'approche phénétique ne tient pas compte du procesus de l'évolution. Elle se contente de mesurer les distances entre les espèces et de reconstruire le meilleur arbre possible à l'aide d'une stratégie de regroupement hiérarchique.

L'approche cladistique cherche à établir des relations de parenté en s'intéressant aux caractères (bases ou acices aminés) dérivés, partagés par les taxons. On considère ainsi tous les scénarios d'évolution en inférant les caractères des ancêtres potentiels à chaque noeud, et on choisit l'arbre qui correspond au meilleur scénario d'évolution selon un critère préalablement choisi. Les méthodes utilisées sont essentiellement basées sur le maximum de parcimonie.

La première approche étudie la parenté entre les taxons en s'intéressant à leur degré de similarité alors que la deuxième est basée sur la généalogie.

L'approche probabiliste (ou maximum de vraisemblance), quant à elle, évalue en termes de probabilités l'ordre des branchements et la longueur des arêtes d'un arbre sous un modèle évolutif donné. Les méthodes bayésiennes font aussi partie de cette approche.

Avant de donner plus de détails sur toutes ces méthodes, nous donnons ici quelques définitions de base concernant les arbres phylogénétiques et les métriques d'arbres, en

suivant la terminologie de Barthélemy et Guénoche (1991). La distance $\delta(x,y)$ entre deux sommets x et y dans un arbre phylogénétique (i.e., arbre additif) T est définie comme la somme de toutes les longueurs des arêtes du chemin unique liant x et y dans T. Un tel chemin est noté (x,y). Une feuille est un sommet de degré un. La figure 1.2 est un exemple du calcul d'une telle distance.

Définition 1.1. *Soit X un ensemble fini de n taxons. Une dissimilarité d sur X est une fonction non-négative sur $X \times X$ telle que pour tout x,y appartenant à X :*

(1) $d(x,y) = d(y,x)$, *et*

(2) $d(x,y) = d(y,x) \geq d(x,x) = 0$.

Définition 1.2. *Une dissimilarité d sur X satisfait la condition des quatre points si pour tout x,y,z, et w de X : $d(x,y)+d(z,w) \leq Max\{d(x,z)+d(y,w); d(x,w)+d(y,z)\}$.*

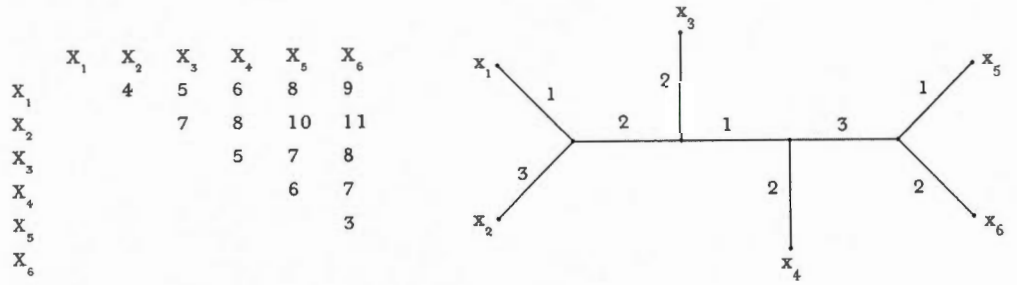
Définition 1.3. *Pour un ensemble fini X, un arbre phylogénétique (i.e., un arbre additif ou un X-arbre) est une paire ordonnée (T,ϕ) consistant en un arbre T, avec un ensemble de sommets V et une relation $\phi : X \rightarrow V$, ayant la propriété que, pour tout $x \in X$ avec un degré d'au moins deux, $x \in \phi(X)$. Un arbre phylogénétique est binaire si ϕ est une bijection de X dans l'ensemble des feuilles de T et que chàque sommet interne a un degré égal à trois.*

Le théorème principal reliant la condition des quatre points et la représentabilité d'une dissimilarité par un arbre phylogénétique (i.e., une phylogénie) est comme suit :

Théorème 1.1. *(Zarestskii, Buneman, Patrinos et Hakimi, Dobson) Toute dissimilarité satisfaisant la condition des quatre points peut être représentée par un arbre phylogénétique tel que pour tout x,y appartenant à X, $d(x,y)$ est égale à la longueur du chemin liant les feuilles x et y dans T.*

Cette dissimilarité est appelée une distance d'arbre. Cet arbre est unique.

Depuis une quarantaine d'années, de nombreux algorithmes ont été développés pour inférer des arbres phylogénétiques.



	X_1	X_2	X_3	X_4	X_5	X_6
X_1		4	5	6	8	9
X_2			7	8	10	11
X_3				5	7	8
X_4					6	7
X_5						3
X_6						

Figure 1.2 Exemple d'une distance d'arbre sur un ensemble X de 6 taxons et l'arbre phylogénétique associé.

Les méthodes de distances sont les plus rapides. Elles prennent en entrée une matrice de distances entre des espèces et déterminent un arbre dont les feuilles sont en correspondance avec les espèces initiales. Ces distances sont calculées en fonction du nombre de molécules différentes dans un ensemble de gènes dont les séquences d'ADN ont été préalablement alignées. La somme des longueurs des branches du chemin le plus court entre deux feuilles est censée être la plus proche possible de la distance réelle entre les deux espèces représentées par ces feuilles. Ce n'est pas le cas si le taux d'évolution n'est pas constant dans tout l'arbre ou si l'hypothèse de l'horloge moléculaire n'est pas vérifiée (voir le glossaire pour la définition de cette hypothèse). On peut alors corriger les distances par différentes transformations (Jukes et Cantor, 1969; Kimura, 1980). Les deux principales méthodes de distances sont *Neighbor Joining* (Saitou et Nei, 1987) qui sera étudiée en détails dans le chapitre suivant, et *UPGMA* (Sneath et Sokal, 1973) qui est l'acronyme de « Unweighted Pair Group Method with Arithmetic mean ». Elles sont toutes les deux basées sur le principe du clustering. Comme on le verra dans le cas de *Neighbor Joining*, la complexité de ces algorithmes est polynomiale en fonction du nombre d'espèces considérées.

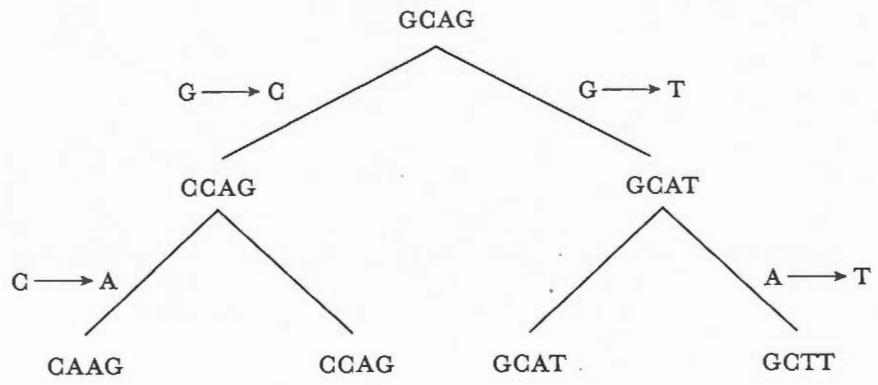


Figure 1.3 Scénario d'évolution le plus parcimonieux.

Quand on étudie un petit nombre d'espèces, on peut également utiliser le principe de parcimonie (Fitch, 1971). On prend alors en entrée une séquence d'ADN ou de protéines pour chaque espèce, et on essaie de reconstituer l'arbre qui porte le scénario d'évolution le plus parcimonieux, i.e., qui nécessite le moins de modifications des séquences au cours de l'évolution. La figure 1.3 montre le scénario d'évolution le plus parcimonieux pour les séquences CAAG, CCAG, GCAT et GCTT. Pour une topologie d'arbre donnée, l'algorithme de Fitch (Fitch, 1971) permet de retrouver un des scénarios les plus parcimonieux en $O(n*c*k)$, où n est le nombre d'espèces, c la longueur des séquences d'ADN ou de protéines et k le nombre d'états possibles ($k = 4$ pour l'ADN et $k = 20$ pour les protéines). Cet algorithme est basé sur les principes de la programmation dynamique. Pour chaque position des séquences considérées, on part des états des feuilles de l'arbre pour remonter l'arbre progressivement : pour chaque nœud, on construit l'ensemble de ses états possibles en fonction des ensembles des états de ses descendants. Le problème de retrouver la topologie la plus parcimonieuse parmi toutes les topologies d'arbres est cependant NP-difficile, ce qui rend les méthodes de distances beaucoup plus rapides en général que ces méthodes de maximum de parcimonie. Notons que les longueurs de branches ne sont pas prises en compte dans ce contexte.

Le principe du maximum de vraisemblance (Felsenstein, 1981) est également utilisé dans le cas d'un petit nombre d'espèces. Ce principe est basé sur un critère probabiliste. On doit ainsi disposer d'un modèle d'évolution, c'est-à-dire qu'on doit définir la probabilité d'une mutation d'un nucléotide (ou d'une protéine) en un(e) autre le long d'une branche d'un arbre en fonction de la longueur de cette branche et des deux nucléotides (ou protéines). Pour un arbre phylogénétique donné, on peut alors calculer la vraisemblance de cet arbre, i.e., la probabilité du scénario d'évolution le plus probable le long de cet arbre. Une des difficultés est d'optimiser les longueurs de branches pour une topologie d'arbre fixée. Comme dans le cas de la parcimonie, le scénario le plus vraisemblable est déterminé position par position en utilisant un principe de programmation dynamique. Le problème de retrouver l'arbre le plus vraisemblable parmi tous les arbres possibles est NP-difficile.

Toutes les méthodes précédemment citées sont implémentées dans les logiciels PAUP (Swafford, 2002), Phylip (Felsenstein, 2005) et T-Rex (Makarenkov, 2001).

Notons que deux méthodes de maximum de vraisemblance sont particulièrement efficaces : la méthode PhyML (Guindon et Gascuel, 2003), implémentée dans le logiciel PhyML 3 (Guindon et al., 2010), et la méthode RAxML (Stamatakis, Hoover et Rougemont, 2008) implémentée dans le logiciel RAxML-Light (Stamatakis et al., 2012).

Plus récemment, des approches bayésiennes (Rannala et Yang, 1996) ont permis d'utiliser le maximum de vraisemblance pour des données plus importantes. Dans ce contexte, l'hypothèse optimale est celle qui maximise la probabilité *a posteriori*. Cette probabilité *a posteriori* est proportionelle à la vraisemblance multipliée par la probabilité *a priori* de l'hypothèse. On peut ainsi dévolopper des algorithmes plus rapides qui peuvent incorporer des modèles d'évolution plus complexes. Par exemple, le logiciel MrBayes (Huelsenbeck et Ronquist, 2001) utilise ainsi les MCMC (chaînes de Markov Monte-Carlo) pour parcourir l'espace de tous les arbres possibles en vue d'obtenir l'arbre le plus vraisemblable.

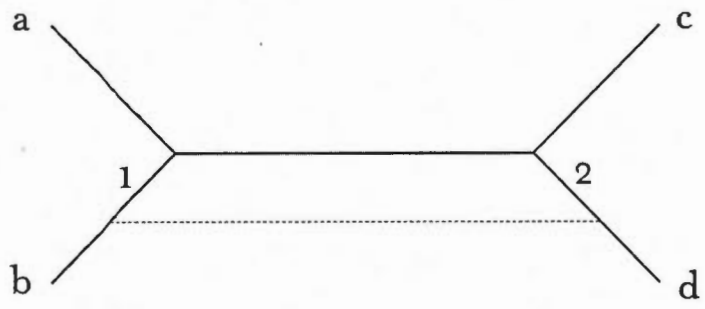


Figure 1.4 Un réseau réticulé.

1.2 L'évolution réticulée

Plusieurs importants mécanismes phylogénétiques s'expliquent par le phénomène de l'évolution réticulée qui suppose des liens supplémentaires entre les espèces par rapport au modèle arborescent classique (Doolittle, 1999; Legendre, 2000b). L'évolution réticulée reflète la part de l'évolution des espèces qui ne peut pas être représentée correctement par le modèle de bifurcation utilisé classiquement en analyse phylogénétique. La figure 1.4 montre un réseau réticulé (ici un réticulogramme). Le trait ajouté entre les arêtes 1 et 2 représente une arête de réticulation ajoutée à l'arbre original.

Dans son célèbre article, Doolittle (1999) a mis l'accent sur le rôle de l'évolution réticulée, et plus précisément du transfert horizontal des gènes, dans l'évolution des bactéries, de même que des espèces plus complexes. La figure 1.5 présentée par Doolittle montre que l'évolution des espèces se produit selon un modèle en réseau plutôt qu'un modèle en arbre. Pour Doolittle, les biologistes moléculaires auraient échoué à trouver le vrai arbre de la vie non pas à cause de méthodes inadéquates ou d'un mauvais choix de gènes, mais parce que l'histoire de la vie ne peut être representée correctement par un arbre. D'autre part, les spécialistes en biologie évolutive ont remarqué que des phénomènes très importants, tels que l'hybridation et l'allopolyploïdie, ne correspondent pas au modèle d'évolution arborescente traditionnel (Legendre, 2000b; Lapointe, 2000).

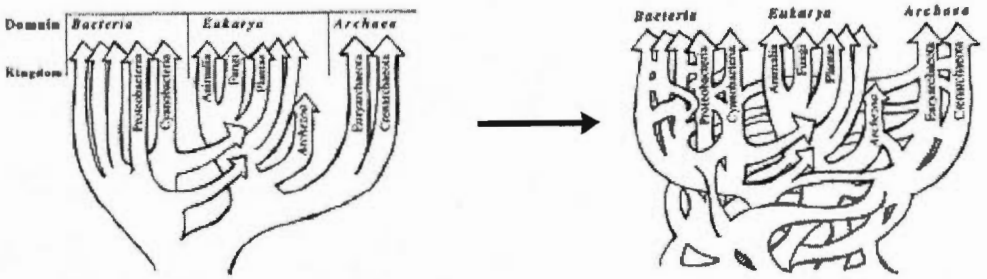


Figure 1.5 Le réseau réticulé représenterait mieux l'histoire de la vie qu'un arbre phylogénétique classique (Doolittle, 1999).

Nous allons détailler les différents phénomènes de réticulation en insistant tout particulièrement sur l'hybridation qui nous intéressera par la suite.

1.2.1 Transfert horizontal de gènes

La prise de conscience du rôle majeur joué par les transferts horizontaux de gènes (THG) dans l'évolution des espèces est un des changements fondamentaux dans la perception des aspects génétiques de la biologie moléculaire (Koonin, 2003; Doolittle et al., 2003).

Le transfert horizontal de gènes est un transfert direct de matériel génétique d'une lignée à une autre. La figure 1.6 représente un scénario de THG entre les espèces 3 et 4. Généralement, seulement une partie du génome est transféré (parfois un seul gène ou même seulement une partie d'un gène). La figure 1.7 montre le scénario d'évolution correspondant aux gènes transférés (à gauche) et celui approprié pour tous les autres gènes hérités directement des ancêtres (à droite).

Les transferts horizontaux de gènes sont fréquents chez les bactéries. Les Bacteria et les Archaea ont développé la capacité de s'adapter à des nouveaux environnements en acquérant des nouveaux gènes par THG plutôt qu'en modifiant son patrimoine génétique

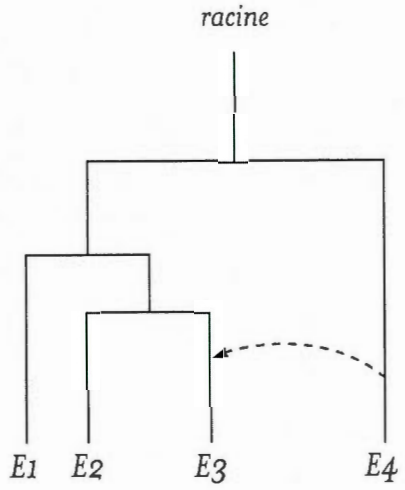


Figure 1.6 Un transfert horizontal de gène de l'ancêtre de l'espèce E_4 vers l'ancêtre de l'espèce E_3.

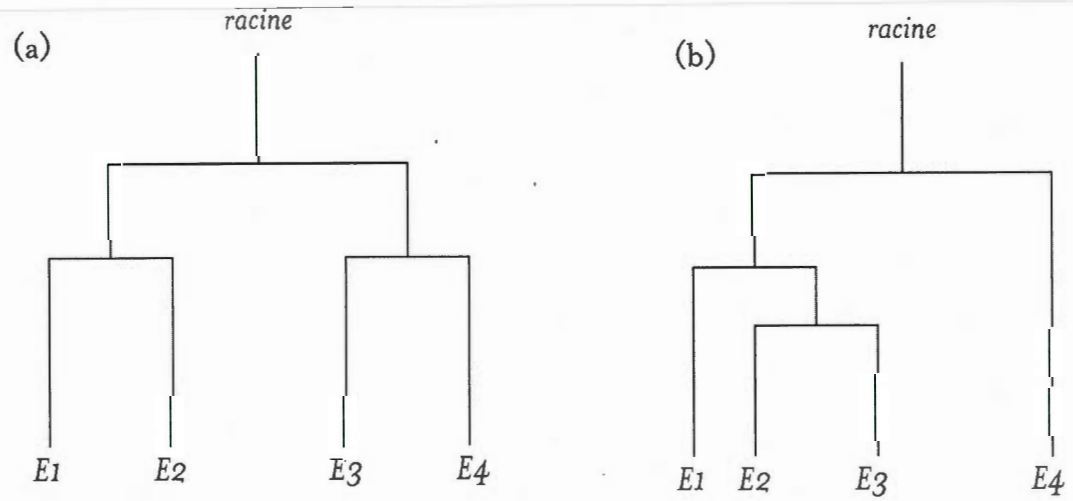


Figure 1.7 Les deux arbres correspondant au transfert horizontal de la figure 1.6.

par une série de mutations (Gogarten, Doolittle et Lawrence, 2002; Zhaxybayeva, Lapierre et Gogarten, 2004). Comme elles n'ont pas de reproduction sexuée, les bactéries ont adopté différents mécanismes de THG (Bauman, 2007) :

1. Transformation : Ce procédé est plus fréquent chez les bactéries qui se transforment naturellement. Ces bactéries prennent des fragments d'ADN dans leur environnement. Cela permet l'échange de n'importe quelle partie d'un chromosome mais seulement de petis fragments.
2. Conjugaison : Ce type de transfert d'ADN nécessite un contact cellule à cellule. Il peut avoir lieu entre des bactéries et des eukaryotes et permet de transférer de plus grand morceaux d'ADN.
3. Transduction : C'est le transfert d'ADN par phage (virus affectant les bactéries). Le donneur et le récepteur doivent avoir des récepteurs cellulaires compatibles. Ces transferts n'ont donc lieu qu'entre deux espèces de bactéries très proches l'une de l'autre. La taille de l'ADN transféré est relativement limitée.

Ces mécanismes de THG peuvent introduire des séquences d'ADN qui ont peu d'homologie avec l'ADN de la cellule réceptrice. Si l'ADN du donneur et le chromosome récepteur ont des séquences homologues, l'ADN du donneur peut être incorporé au chromosome récepteur de manière stable par recombinaison homologue. Si les séquences homologues sont situées près de séquences absentes chez le récepteur, celui-ci peut alors acquérir une insertion de taille quelconque (Jain, Rivera et Lake, 1999).

1.2.2 Hybridation

L'hybridation est un autre exemple d'évolution réticulée (Arnold, 1997). Dans la figure 1.8, deux lignées (Racine-Espèce 2 et Racine-Espèce 3) se recombinent pour créer une nouvelle espèce (Espèce 4). Si la nouvelle espèce possède le même nombre de chromosomes que les espèces parents, on parle d'hybridation *diploïde*. Si elle possède la somme du nombre de chromosomes de ses parents, on parle d'hybridation *polyploïde*. Il existe trois principaux mécanismes d'hybridation :

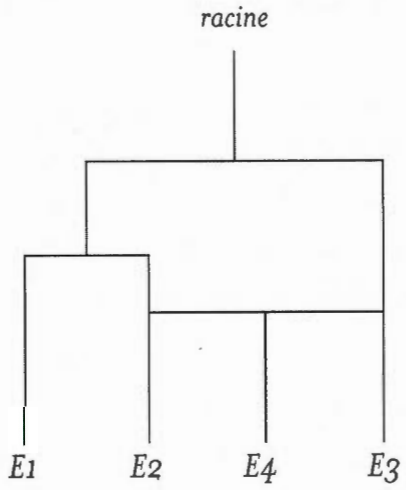


Figure 1.8 Un exemple d'hybridation.

1. L'autopolyploïdisation est un événement de spéciation impliquant le doublement des chromosomes à l'intérieur d'une même espèce. Cela produit une bifurcation dans l'arbre phylogénétique.

2. L'allopolyploïdisation est une hybridation entre deux espèces où la nouvelle espèce acquiert l'ensemble des compléments des chromosomes *diploïdes* des deux parents. Dans ce cas, les parents n'ont pas forcément le même nombre de chromosomes. L'allopolyploïdization provoque une spéciation instantanée car tout croisement avec les parents *diploïdes* risque de produire une espèce *triploïde* stérile.

3. La spéciation par l'hybridation *diploïde* est un événement sexuel normal entre deux parents d'espèces distinctes mais assez proches. Dans la plupart des cas, les deux parents doivent avoir le même nombre de chromosomes. Le croisement avec les parents est alors possible et les hybrides doivent donc être isolés des parents pour devenir une nouvelle espèce.

Dans ce mémoire, nous allons étudier la modélisation et l'inférence d'un réseau réticulé impliquant des phénomèmes de spéciation par l'hybridation *diploïde*. Dans les organismes *diploïdes* normaux, les chromosomes sont regroupés en paires de chromosomes homologues. Dans le processus d'hybridation *diploïde*, l'hybride hérite d'un des deux chromosomes homologues de chaque paire de chromosomes de chacun de ses deux parents. Comme les gènes des deux parents contribuent au patrimoine génétique de l'hybride, l'évolution des gènes hérités de chaque parent peut être représentée dans des arbres séparés à l'intérieur d'un modèle en réseau. L'analyse phylogénétique classique des quatre espèces impliquées dans l'hybridation de la figure 1.8 donnera un des deux arbres de la figure 1.9.

L'hybridation est très fréquente chez les plantes, les poissons, les amphibiens et les reptiles, et est pratiquement absente dans les autres groupes, particulièrement chez les oiseaux, les mammifères, et la plupart des arthropodes, ces derniers étant occasionnellement soumis à ces phénomènes. Ils produisent alors en général des triploïdes qui peuvent se reproduire seulement par reproduction asexuée.

Chez les plantes par exemple, il y aurait plus de 70000 hybrides naturels (Stace, 1991). Des plantes hybrides sont également créées par l'homme pour introduire des caractéristiques souhaitables chez des espèces cultivées (Judd, 2008). Chez les poissons, les amphibiens et les reptiles, la gynogenèse est un mode de reproduction qui permet à des hybrides femelles unisexuées de se reproduire en utilisant le sperme d'une espèce bisexuée proche pour stimuler le développement des oeufs (Dawley, 1989).

1.2.3 Autre phénomènes

Deux autres phénomènes sont à l'origine d'évolution réticulée.

Tout d'abord, l'homoplasie est le développement à l'intérieur de différentes espèces, qui n'ont pas d'ancêtres communs, d'organes qui se ressemblent et qui ont les mêmes fonctions. Ces organes se développent dans le cadre d'évolutions convergentes et sont donc analogues sans être homologues. Par exemple, les ailes des insectes, des oiseaux et

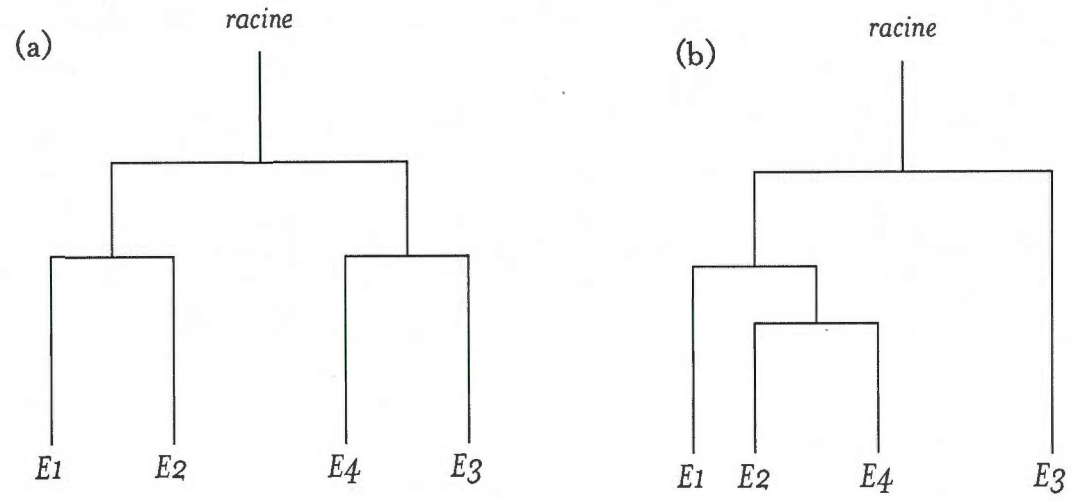


Figure 1.9 Deux arbres pour le phénomène d'hybridation présenté sur la figure 1.8.

des chauve-souris sont homoplastiques, i.e., similaires en forme et en structure mais pas dans leurs origines. Ces phénomènes peuvent compliquer les inférences phylogénétiques (Smouse, 2000). Dans le cas de l'homoplasie par exemple, le but n'est pas de modéliser des événements de réticulations mais de produire un diagramme qui décrit, plus précisément qu'un arbre, les motifs communs trouvés dans les données (Legendre, 2000a). Si les distances entre espèces semblent trop petites à cause d'événements d'homoplasie, on peut ainsi ajouter une branche réticulée et produire ce qu'on appelle un réticulogramme (Legendre et Makarenkov, 2002).

La recombinaison génétique, quant à elle, regroupe tous les processus qui modifient le matériel génétique comme le réassortiment des gènes parentaux lors de la formation des gamètes (crossing-over), ou l'échange de matériel génétique entre chromosomes homologues. Ces recombinaisons créent des réticulations à l'intérieur d'une lignée. Plusieurs méthodes statistiques pour la détection de ces recombinaisons ont été développées. Ces méthodes ont été testées par Posada et Crandall (2001) qui ont estimé la performance

de 14 différents algorithmes.

1.3 Revue de littérature sur l'inférence de réseaux phylogénétiques

Dans cette section, nous présentons une revue de la littérature sur les méthodes et logiciels développés pour la détection et la visualisation de l'évolution réticulée.

L'évolution réticulée a longtemps été négligée dans les analyses phylogénétiques. Les premières méthodes qui étudiaient les mécanismes de l'évolution réticulée sont apparues au milieu des années 1970 (Sneath, Sackin et Ambler, 1975; Sonea et Panisset, 1983). Plusieurs méthodes ont été proposées pour identifier l'évolution réticulée dans les séquences de nucléotides. Celles-ci incluent : l'affichage de compatibilité (Sneath, Sackin et Ambler, 1975), des tests de regroupement (Stephens, 1985), une approche de randomisation (Sawyer, 1989) et une extension de la méthode de reconstruction d'arbres par parcimonie qui permet la recombinaison (Hein, 1993). Rieseberg et Morefield (1995) ont développé un programme, RETICLAD, qui permet d'identifier les hybrides, fondé sur l'idée qu'ils combinent les caractères de leurs parents. Cependant, ce programme permet de trouver des réticulations seulement entre les arêtes terminales d'un arbre. Rieseberg et Ellstrand (1993) ont montré des exemples où le programme semble bien fonctionner. La populaire méthode de décomposition en partitions (split-decomposition) rend possible la représentation des données sous la forme d'un split-graphe révélant les conflits contenus dans les données (Bandelt et Dress, 1992a; Bandelt et Dress, 1992b). Dans un split-graphe, une paire de nœuds peut être reliée par un ensemble d'arêtes parallèles décrivant les hypothèses d'évolution alternatives. Une autre interprétation des split-graphes est qu'ils représentent en deux dimensions des similarités entre les espèces étudiées. Hallett et Lagergren (2001) ont montré comment le transfert horizontal de gènes pouvait être détecté en mesurant la différence topologique entre un arbre d'espèces et un arbre de gène. Bryant, Huson et Multon (Bryant et Moulton, 2004; Huson et Bryant, 2006) ont construit des algorithmes basés sur des décompositions en coupe qui prennent en entrée différentes données sur un ensemble d'espèces et qui déterminent différentes décompositions de cet ensemble. Ils n'obtiennent donc pas un réseau explicite

et les résultats sont souvent difficiles à interpréter même si Gambette et Huson (2008) ont amélioré la visualisation de telles décompositions. Le logiciel SplitsTree (Huson et Bryant, 2006), qu'ils ont développé, est cependant l'outil le plus utilisé actuellement. Chacune de ces méthodes a des propriétés qui la rend utile pour l'analyse de données particulières et elles ont toutes un rôle à jouer dans la détection et la caractérisation de l'évolution réticulée. Legendre et Makarenkov (Legendre et Makarenkov, 2002; Makarenkov et Legendre, 2004) ont proposé d'utiliser les réticulogrammes pour détecter les réticulations dans des données évolutionnaires. Ils ont développé une méthode basée sur les distances qui infère des phylogénies réticulées. Cette méthode utilise la topologie d'un arbre phylogénétique comme une structure de base sur laquelle on ajoute, au fur et à mesure et suivant un critère d'optimisation, des arêtes de réticulation pour construire un réticulogramme. Un enjeu majeur et délicat est alors de déterminer à quel moment l'algorithme doit cesser d'ajouter des branches. Makarenkov et al. (2006), Boc et al. (2010), et Boc et Makarenkov (2012) ont proposé des algorithmes pour identifier des transferts horizontaux à partir d'arbres différents obtenus pour représenter l'histoire évolutive d'un même ensemble d'espèces. Ces méthodes sont implémentées dans le logiciel T-REX (Makarenkov, 2001). Dans le cas d'un ensemble d'individus d'une même population, Bandelt et al. (1999) utilisent un critère de parcimonie pour ajouter des intermédiaires manquants à un réseau initial obtenu en combinant plusieurs arbres de couverture minimale. Doyon et al. (2010) proposent quant à eux une méthode de parcimonie pour réconcilier un arbre d'espèces et des arbres de gènes, en prenant en compte les transferts latéraux, les pertes de gènes et les duplications. Huson et Rupp (2008) ainsi que van Iersel et al. (2010) utilisent la notion de réseaux de clusters pour réconcilier différents arbres phylogénétiques contradictoires. Dans le cas de deux arbres contradictoires, Albrecht et al. (2012) proposent un algorithme parallèle pour trouver un réseau d'hybridation minimum. Parmi d'autres techniques d'inférence de réseaux phylogénétiques, mentionnons : la parcimonie statistique (Templeton, Crandall et Sing, 1992), le Netting (Fitch, 1997), les réseaux medians-joints (Foulds, Hendy et Penny, 1979; Bandelt et Dress, 1992a), la parcimonie à variance moléculaire (Excoffer et Smouse, 1994), les pyramides (Diday et Bertrand, 1984) et les hiérarchies faibles (Ban-

delt et Dress, 1989). Pour une vue d'ensemble sur la question, on se référera à (Huson, Rupp et Scornavacca, 2010).

La plupart de ces techniques ont été testées par Woolley et al. (2008). Elles ne s'avèrent efficaces que dans des configurations particulières. De plus, il n'existe aucun critère statistique permettant de valider les réseaux ainsi obtenus et de choisir le meilleur réseau entre différents résultats obtenus avec différents logiciels.

1.4 Glossaire

ADN (acide désoxyribo nucléique) : Macromolécule constituée de deux chaînes enroulées en double hélice. Ses deux brins sont assemblés à partir de nucléotides. Chaque nucléotide comprend un sucre, le désoxyribose, un phosphate et une des quatre bases azotées (adénine, guanine, cytosine et thymine). L'ADN est le support de l'information génétique des organismes vivants.

Alignement : Opération qui consiste à disposer les unes en dessous des autres des portions de séquences similaires en minimisant leurs différences (on peut aligner entre eux des gènes d'une même famille multigénique ou des gènes d'espèces différentes). Si ces gènes sont homologues, les différences d'aminoacides ou d'acides nucléiques entre les séquences actuelles sont le témoignage de mutations qui ont eu lieu dans le passé.

Aminoacide (acide aminé) : Unité constitutive des protéines. Il existe vingt acides aminés communs : alanine, arginine, asparagine, aspartate, cystéine, glutamine, glycine, histidine, isoleucine, leucine, lysine, méthionine, phénylalanine, proline, glutamate, sérine, thréonine, tryptophane, tyrosine et valine.

Archaea : Les Archées ou Archaea (anciennement appelés archéobactéries, du grec archaios, « ancien » et backterion, « bâton ») sont un groupe majeur de microorganismes. Elles constituent un taxon du vivant caractérisé par des cellules sans noyau et se distinguant des Eubactéries (vraies bactéries) par certains caractères biochimiques, comme la constitution de la membrane cellulaire ou le mécanisme de réplication de l'ADN.

ARN (acide ribonucléique) : Polymère linéaire dont la sous-unité de base, un ribonucléotide, contient le sucre ribose.

Bacteria : Les bactéries appartiennent au vaste ensemble des microbes qui comprennent également les virus, les champignons et les parasites. Microorganismes invisibles à l'œil nu, les bactéries sont constituées d'une seule cellule dépourvue d'un vrai noyau. Elles contiennent un seul chromosome formé d'un long filament d'ADN.

Clade : Vient du grec clados qui signifie « arête ». Taxon strictement monophylétique, c'est-à-dire contenant un ancêtre et tous ses descendants.

Eucarya : Les Eucaryotes (du grec eu, vrai et karuon, noyau) comprennent quatre grands règnes du monde vivant : les animaux, les champignons, les plantes et les protistes. Ils constituent donc un très large groupe d'organismes, unis et pluricellulaires, définis par leur structure cellulaire (noyau, ADN, cytosquelette, etc).

Extragroupe (outgroup) : On dit aussi groupe extérieur ou encore « outgroup » tiré de l'anglais. Groupe que l'on sait a priori placé en dehors d'un ensemble de taxons dont on cherche les relations de parenté.

Horloge moléculaire (hypothèse) : Hypothèse selon laquelle les molécules d'une même classe fonctionnelle évoluent régulièrement dans le temps à un rythme égal dans différentes lignées. Ainsi la quantité des différences moléculaires constatées de nos jours dans des séquences homologues d'espèces distinctes peut être utilisée pour estimer le temps écoulé depuis le dernier ancêtre commun à ces espèces (ou le temps de divergence).

Racine : Segment de l'arête en amont du nœud du rang le plus important, définissant le groupe extérieur (voir Extragroupe). En d'autres termes, c'est la position dans l'arbre du groupe extérieur. La racine peut être considérée comme un point de référence pour l'interprétation des caractères : les états de caractères de l'extragroupe (outgroup) sont des états plésiomorphes, les états qui en diffèrent sont apomorphes. Remarque : pour pouvoir comparer aisément deux arbres, il faut les enraciner chacun avec la même espèce ou avec le même taxon.

Taxon : Ensemble des organismes reconnus et définis dans chacune des catégories de la classification biologique hiérarchisée. En d'autres termes : contenu concret d'une catégorie. Exemple : *Canis lupus,* le loup, est un taxon de rang spécifique (catégorie : espèce) ; les canidés (Chien, Loup, Renard) constituent un taxon de rang familial (catégorie : famille).

CHAPITRE II

DESCRIPTION DU NOUVEL ALGORITHME

L'algorithme Neighbor Joining (Saitou et Nei, 1987) est la plus utilisée des méthodes de distances pour inférer un arbre phylogénétique. Atteson (1999) a prouvé que cet algorithme est capable de retrouver la vraie phylogénie si les distances utilisées sont suffisamment proches des vraies distances d'évolution. Notons que *Bio Neighbor Joining* (Gascuel, 1997) est une version améliorée de NJ que nous ne présenterons pas ici.

Notre but est de généraliser la méthode NJ traditionnelle en introduisant des phénomènes d'hybridation dans l'arbre reconstruit et d'obtenir ainsi un réseau d'hybridation. Nous avons choisi cette méthode car elle permet de retrouver des phylogénies en un temps polynomial en fonction du nombre d'espèces considérées. Elle est également relativement facile à implémenter. De plus, dans le cadre des méthodes de distances, on peut facilement faire des tests avec des arbres additifs (auxquels on peut rajouter des phénomènes d'hybridation) pour vérifier l'efficacité de l'algorithme qu'on élaborera.

Nous allons tout d'abord expliquer très en détails l'algorithme NJ, puis nous présenterons notre algorithme principal.

Dans toute la suite on considérera qu'on dispose en entrée d'une matrice de distances $D = \{D[i][j]\}_{1 \leq i \leq n; 1 \leq j \leq n}$ sur un ensemble de n espèces, et qu'on veut obtenir en sortie un arbre ou un réseau expliquant la phylogénie de ces espèces.

Notons que $D[i][i] = 0$ pour tout $1 \leq i \leq n$ et que $D[i][j] = D[j][i]$ pour $1 \leq i \leq n$ et $1 \leq j \leq n$.

2.1 L'algorithme *Neighbor Joining*

L'algorithme NJ est un algorithme itératif qui part d'un buisson composé de n feuilles et n branches, où n est le nombre d'espèces considérées. Ce buisson est représenté sur la figure 2.1 pour le cas $n = 6$. Cet arbre est graduellement transformé en un arbre phylogénétique binaire non enraciné avec les mêmes n feuilles et avec $2n - 3$ branches. Ainsi, à la i-ème itération, l'algorithme doit choisir deux nœuds voisins parmi $n - i + 1$ candidats. La taille de la matrice de distances est alors réduite de 1, les deux nœuds regroupés étant remplacés par leur ancêtre commun. À chaque itération, pour choisir les deux nœuds à joindre, on considère toutes les $\frac{(n-i+1)(n-i)}{2}$ configurations semblables à celle représentée sur la figure 2.2. Pour chacune de ces configurations, on calcule les longueurs de branches qui minimisent le critère des moindres carrés où on compare les distances entre les nœuds avec les distances qu'on obtiendrait avec les longueurs de branches si on avait affaire à un arbre additif. Par exemple, pour la figure 2.2, on cherche à minimiser :

$$(D[1][2] - L_1 - L_2)^2 + \sum_{3\le j<k\le 6} (D[j][k] - L_j - L_k)^2$$

$$+ \sum_{3\le k\le 6} \left[(D[1][k] - L_1 - L_{XY} - L_k)^2 + (D[2][k] - L_2 - L_{XY} - L_k)^2\right].$$

Dans le cas général, si on joint les nœuds 1 et 2 parmi un buisson de n nœuds, Saitou et Nei (1987) montrent que les longueurs de branches sont alors données par les formules suivantes :

$$L_1 = \frac{1}{2}D[1][2] + \frac{1}{2(n-2)}(P - Q), \qquad L_2 = \frac{1}{2}D[1][2] - \frac{1}{2(n-2)}(P - Q), \tag{2.1}$$

$$L_i = \frac{1}{2(n-2)} \sum_{1\le j\le n; j\ne i} D[i][j] - \frac{1}{(n-2)^2}(P + Q) - \frac{n-4}{(n-2)^2(n-3)}V, \ i \ge 3, \tag{2.2}$$

$$L_{XY} = \frac{1}{2(n-2)}(P + Q) - \frac{1}{2}D[1][2] - \frac{1}{(n-2)(n-3)}V, \tag{2.3}$$

où

$$P = \sum_{3\le j\le n} D[1][j], \ Q = \sum_{3\le j\le n} D[2][j], \ V = \sum_{3\le i<j\le n} D[i][j].$$

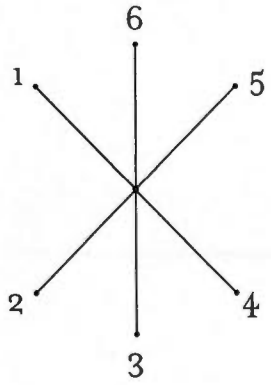


Figure 2.1 Buisson de taille 6.

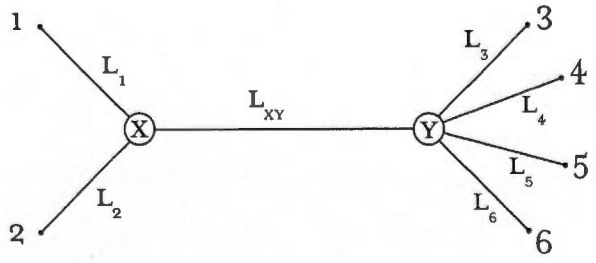


Figure 2.2 Configuration où les nœuds 1 et 2 sont choisis comme voisins.

On obtient alors que la somme de toutes les longueurs de branches est égale à :

$$S_{1;2} = \frac{1}{2}D[1][2] + \frac{1}{2(n-2)}(P+Q) + \frac{1}{n-2}V$$

$$= \frac{1}{2}D[1][2] + \frac{1}{2(n-2)}\sum_{3\leq j\leq n}[D[1][j] + D[2][j]] + \frac{1}{n-2}\sum_{3\leq i<j\leq n}D[i][j].$$

Pour chaque couple de nœuds i et j, on calcule ainsi la somme des longueurs de branches :

$$S_{i;j} = \frac{1}{2}D[i][j] + \frac{1}{2(n-2)}\sum_{1\leq k\leq n;k\neq i,j}[D[i][k] + D[j][k]] + \frac{1}{n-2}\sum_{1\leq k<l\leq n;k,l\neq i,j}D[k][l]. \tag{2.4}$$

On choisit alors de joindre les nœuds i et j qui minimisent l'évolution totale, i.e., qui minimisent la somme des longueurs de branches $S_{i;j}$. On remplace ces deux nœuds par le nœud X (qui est donc leur ancêtre commun direct), et on obtient une matrice de distances de taille $n-1$. On calcule les nouvelles distances de X aux nœuds restants par la formule suivante :

$$D[X][k] = \frac{1}{2}(D[i][k] + D[j][k]),\ \ k \neq i, j. \tag{2.5}$$

On garde également en mémoire les longueurs des branches de i et j à leur ancêtre X calculées par les formules 2.1. On procède ainsi à $n-3$ itérations successives pour obtenir à la sortie un arbre phylogénétique binaire non enraciné et les longueurs de toutes les branches de cet arbre, comme l'illustre l'exemple suivant.

On prend en entrée la matrice de distances suivante :

$$D = \left(\begin{array}{c|cccccc} 1 & 0 & 4 & 5 & 6 & 8 & 9 \\ 2 & 4 & 0 & 7 & 8 & 10 & 11 \\ 3 & 5 & 7 & 0 & 5 & 7 & 8 \\ 4 & 6 & 8 & 5 & 0 & 6 & 7 \\ 5 & 8 & 10 & 7 & 6 & 0 & 3 \\ 6 & 9 & 11 & 8 & 7 & 3 & 0 \end{array}\right),$$

qui correspond à l'arbre de la figure 1.2. La première colonne indique le nom du nœud correspondant à chaque ligne. On calcule toutes les valeurs des $S_{i;j}$ présentées dans le tableau 2.1.

(i,j)	$S_{i;j}$
$(1,2)$	19
$(1,3)$	20,5
$(1,4)$	21
$(1,5)$	21,75
$(1,6)$	21,75
$(2,3)$	20,5
$(2,4)$	21
$(2,5)$	21,75
$(2,6)$	21,75
$(3,4)$	20,5
$(3,5)$	21,25
$(3,6)$	21,25
$(4,5)$	20,75
$(4,6)$	20,75
$(5,6)$	18,5

Tableau 2.1 Valeurs des $S_{i;j}$ pour la matrice D.

(i,j)	$S_{i;j}$
$(1,2)$	$\frac{95}{6} \simeq 15,83$
$(1,3)$	$\frac{103}{6} \simeq 17,17$
$(1,4)$	$17,5$
$(1,X)$	$17,5$
$(2,3)$	$\frac{103}{6} \simeq 17,17$
$(2,4)$	$17,5$
$(2,X)$	$17,5$
$(3,4)$	$\frac{101}{6} \simeq 16,83$
$(3,X)$	$\frac{101}{6} \simeq 16,83$
$(4,X)$	$\frac{97}{6} \simeq 16,17$

Tableau 2.2 Valeurs des $S_{i;j}$ pour la matrice D_1.

La somme $S_{5;6} = 18,5$ étant la plus petite des 15 valeurs possibles, on choisit de joindre les nœuds 5 et 6. Si on appelle X leur ancêtre commun, on obtient alors l'arbre (b) de la figure 2.3 à partir du buisson (a) de cette même figure. On remplace donc dans la matrice de distances les nœuds 5 et 6 par le nœud X en calculant les distances de X aux nœuds restants par la formule 2.5. On obtient ainsi la matrice suivante :

$$D_1 = \left(\begin{array}{c|ccccc} 1 & 0 & 4 & 5 & 6 & 8,5 \\ 2 & 4 & 0 & 7 & 8 & 10,5 \\ 3 & 5 & 7 & 0 & 5 & 7,5 \\ 4 & 6 & 8 & 5 & 0 & 6,5 \\ X & 8,5 & 10,5 & 7,5 & 6,5 & 0 \end{array}\right).$$

On se retrouve également avec un buisson de taille 5 contenant les nœuds 1, 2, 3, 4 et X. On calcule alors les 10 valeurs des $S_{i;j}$ présentées dans le tableau 2.2.

On s'aperçoit que la plus petite de ces valeurs est $S_{1;2} = \frac{95}{6}$. On choisit donc de joindre les nœuds 1 et 2. Si on appelle Y leur ancêtre commun, on obtient alors l'arbre (c) de la

(i,j)	$S_{i;j}$
$(Y,3)$	13,5
$(Y,4)$	14
(Y,X)	14
$(3,4)$	14
$(3,X)$	14
$(4,X)$	13,5

Tableau 2.3 Valeurs des $S_{i;j}$ pour la matrice D_2.

figure 2.3. On remplace donc dans la matrice de distances les nœuds 1 et 2 par le nœud Y en calculant les distances de Y aux nœuds restants par la formule 2.5. On obtient ainsi la matrice suivante :

$$D_2 = \left(\begin{array}{c|cccc} Y & 0 & 6 & 7 & 9,5 \\ 3 & 6 & 0 & 5 & 7,5 \\ 4 & 7 & 5 & 0 & 6,5 \\ X & 9,5 & 7,5 & 6,5 & 0 \end{array}\right),$$

ainsi qu'un buisson de taille 4. La plus petite valeur des $S_{i;j}$ est alors atteinte pour $S_{X;4}$ et $S_{Y;3}$ comme le montre le tableau 2.3.

On peut donc joindre les nœuds X et 4, ou les nœuds Y et 3, ce qui revient au même. On trouve alors l'arbre (d) de la figure 2.3, où toutes les longueurs de branches sont calculées à chaque itération avec les formules 2.1.

On remarque qu'on retrouve l'arbre additif de la figure 1.2. On peut en fait montrer que l'algorithme NJ retrouve tous les arbres additifs avec les bonnes longueurs de branches (Saitou et Nei, 1987). De manière générale, cette méthode trouve toujours des arbres additifs.

On notera pour finir que la complexité de cet algorithme est en $O(n^3)$ puisqu'on fait

$n-3$ itérations et que la i-ème itération nécessite de considérer tous les couples parmi $n-i+1$ éléments.

2.2 Adaptation de l'algorithme NJ aux cas des arbres contenant des phénomènes d'hybridation

Nous voulons maintenant adapter l'algorithme NJ au cas des phénomènes d'hybridation. Notons que ces hybridations peuvent avoir lieu au niveau de deux branches terminales commes dans la figure 2.4 ou au niveau de deux branches ancestrales comme dans la figure 2.5. Dans ces deux cas, les deux branches en question peuvent provenir directement d'un même nœud comme dans les réseaux (a) des deux figures précédentes, mais elles peuvent également être plus éloignées comme dans les réseaux (b). Dans la figure 2.4, on dira que l'espèce 6 est l'hybride des espèces 1 et 2 (ou de leurs ancêtres) dans le cas (a), et que l'espèce 6 est l'hybride des espèces 2 et 5 dans le cas (b). Dans la figure 2.5, l'espèce ancestrale X est l'hybride des espèces ancestrales Y et Z dans le cas (a) par exemple.

Expliquons les calculs de distances entre espèces dans le cas des phénomènes d'hybridation. On prend comme exemple le réseau (a) de la figure 2.4. La figure 2.6 explicite les informations dont nous avons besoin. Tout d'abord, si on enlève l'hybride, on se retrouve avec un arbre phylogénétique traditionnel (additif) dont les distances entre espèces se calculent à partir des longueurs de branches indiquées sur l'arbre de gauche de cette figure. Pour l'hybride, nous avons besoin d'un nombre réél α entre 0 et 1 qui indique quelle proportion du patrimoine génétique de l'hybride provient de la branche de l'espèce 1 ($1-\alpha$ représente alors la proportion du patrimoine génétique de l'hybride qui provient de la branche de l'espèce 2). Nous devons également connaître les longueurs L_1^0 et L_2^0 entre l'ancêtre X des espèces 1 et 2 et les noeuds X_1 et X_2 entre lesquels a eu lieu l'hybridation, ainsi que la longueur L_6 entre les nœuds 6 et H (voir la partie de droite de la figure 2.6). Notons que les longueurs des branches étiquetées par α et $1-\alpha$ sont nulles. On calcule alors les distances entre l'espèce 6 et les autres espèces comme suit :

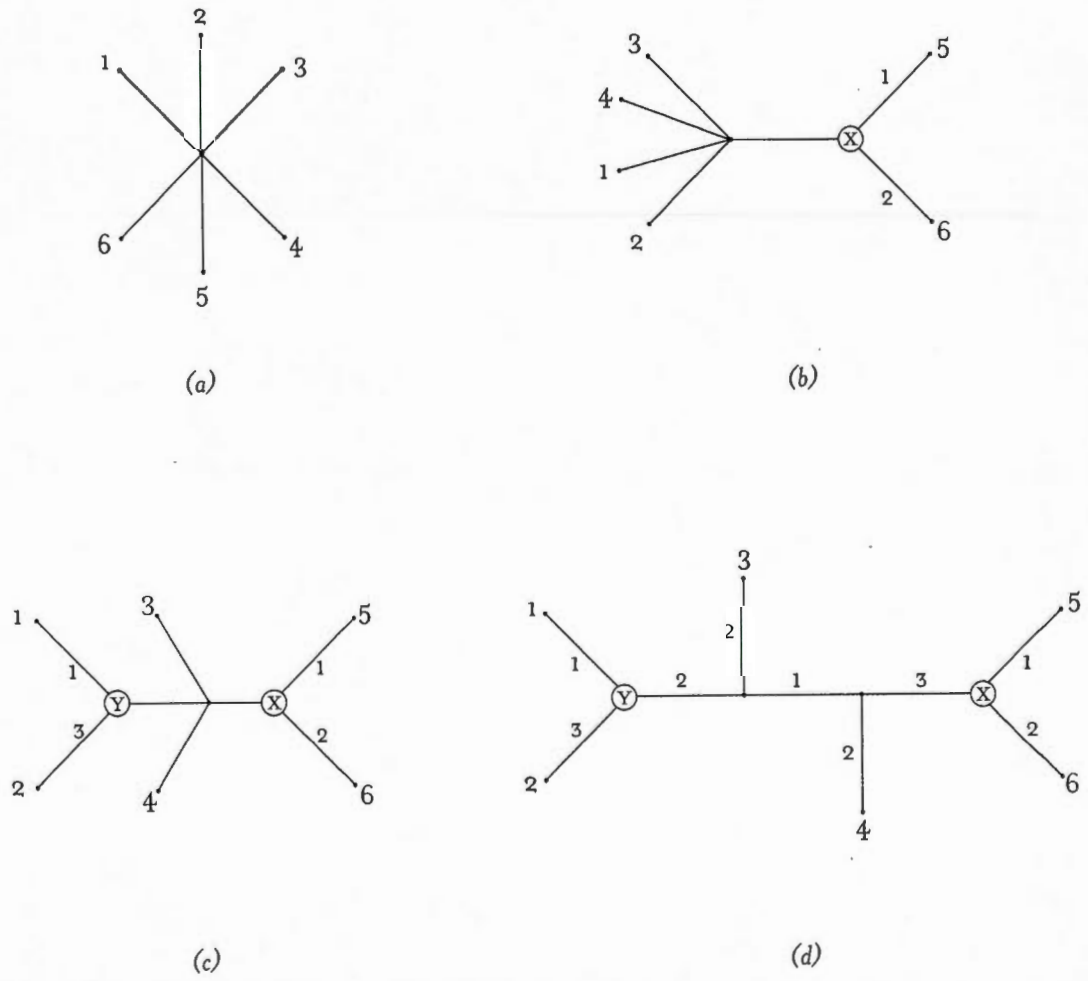


Figure 2.3 Un exemple de la suite des arbres obtenus avec l'algorithme NJ.

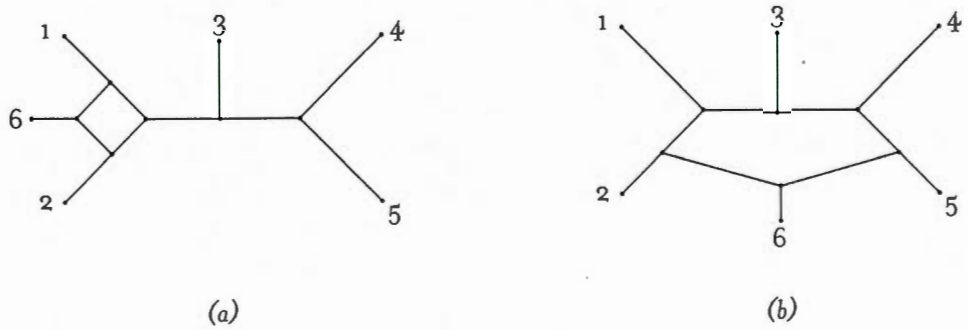


Figure 2.4 Hybrides entre des branches terminales.

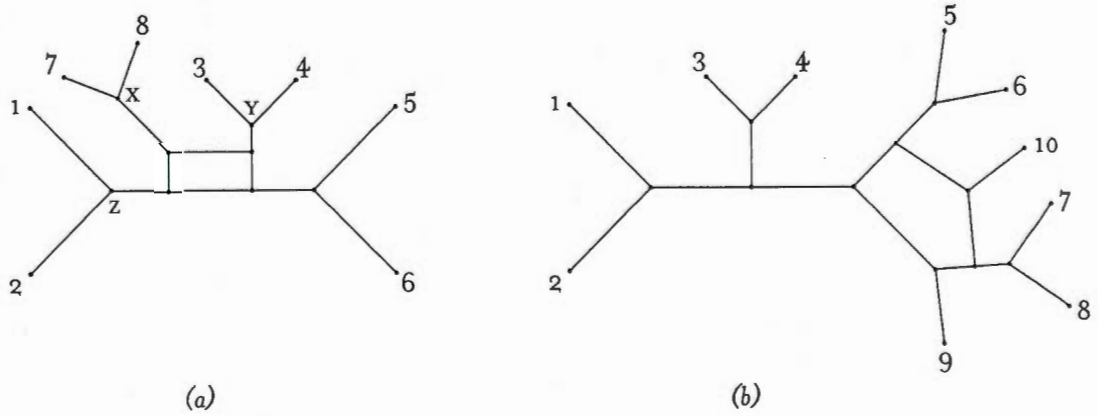


Figure 2.5 Hybrides entre des branches intérieures.

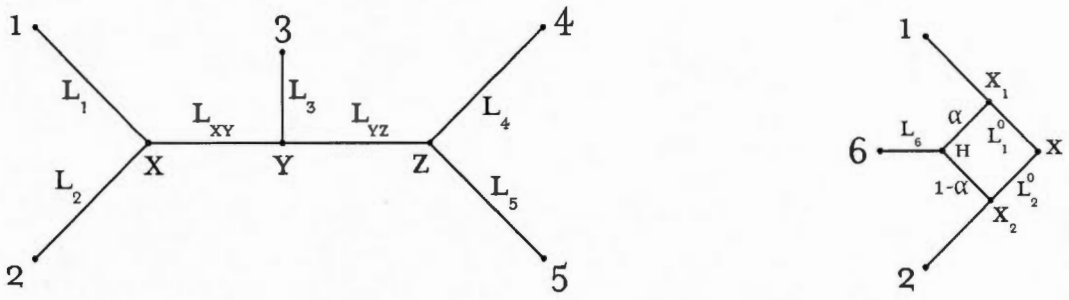


Figure 2.6 Informations nécessaires pour calculer les distances entre un hybride et les autre espèces.

$$D[1][6] = L_6 + \alpha(L_1 - L_1^0) + (1-\alpha)(L_2^0 + L_1), \text{et donc :}$$
$$D[1][6] = L_6 + L_1 - \alpha L_1^0 + (1-\alpha)L_2^0, \tag{2.6}$$

$$D[2][6] = L_6 + (1-\alpha)(L_2 - L_2^0) + \alpha(L_1^0 + L_2), \text{et donc :}$$
$$D[2][6] = L_6 + L_2 + \alpha L_1^0 - (1-\alpha)L_2^0, \tag{2.7}$$

$$D[k][6] = L_6 + \alpha L_1^0 + (1-\alpha)L_2^0 + d(X;k), \;\; 3 \leq k \leq 5, \tag{2.8}$$

où les distances $d(X;k)$ entre les nœuds X et k se calculent comme dans un arbre additif traditionnel.

Nous allons utiliser le même principe que l'algortihme NJ. Notre algorithme est ainsi itératif. À chaque étape, on choisit soit deux nœuds à joindre, soit un nœud qui est un hybride. Pour faire ce choix, on cherche la configuration dont l'évolution est minimale. On a ainsi besoin de définir des configurations correspondant à l'union de deux nœuds (on verra qu'on ne prendra pas toujours les mêmes configurations que dans l'algortihme NJ traditionnel) et des configurations correspondant à un phénomène d'hybridation.

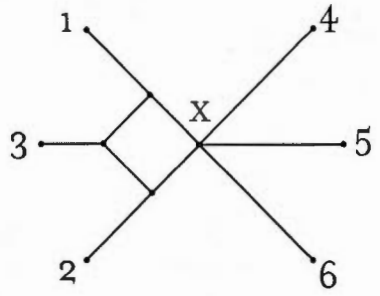


Figure 2.7 Buisson avec un hybride.

2.2.1 Les configurations d'hybridation

Supposons que nous disposons d'une matrice de distances entre n espèces. Si l'espèce 3 est l'hybride des espèces 1 et 2, on utilise alors la configuration représentée sur la figure 2.7 pour $n = 6$. Il s'agit d'un buisson de $n-1$ espèces auquel on ajoute l'hybride 3 entre les nœuds 1 et 2. Nous appellerons L_i, $1 \leq i \leq n$, les distances entre le nœud i et son ancêtre direct, α étant la proportion du patrimoine génétique de l'hybride provenant de la branche de l'espèce 1, L_1^0 et L_2^0 étant les longueurs des branches du noeud X aux bifurcations d'hybridation. On peut retrouver ces notations sur la figure 2.8.

Dans le cas additif, d'après les formules 2.6, 2.7 et 2.8, les distances entre espèces sont alors données par les formules suivantes :

$$D[1][3] = L_3 + L_1 - \alpha L_1^0 + (1-\alpha)L_2^0, \tag{2.9}$$

$$D[2][3] = L_3 + L_2 + \alpha L_1^0 - (1-\alpha)L_2^0, \tag{2.10}$$

$$D[k][3] = L_3 + \alpha L_1^0 + (1-\alpha)L_2^0 + L_k, \ \ 4 \leq k \leq n, \tag{2.11}$$

$$D[i][j] = L_i + L_j, \ \ 1 \leq i < j \leq n, \ \ i,j \neq 3. \tag{2.12}$$

En général, si on dispose d'une matrice de distances arbitraire, on ne peut pas résoudre le système d'équations ci-dessus puisque nous avons $\frac{n(n-1)}{2}$ équations et $n+2$ inconnues

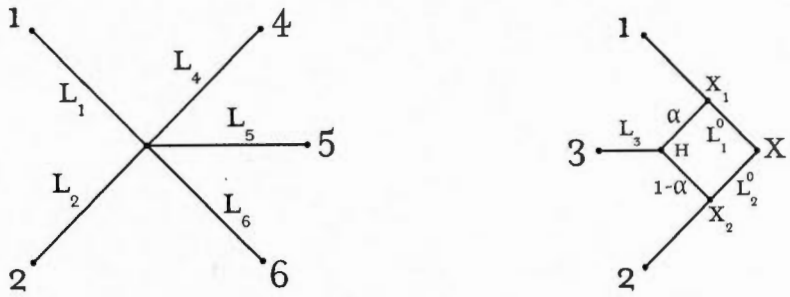


Figure 2.8 Informations nécessaires pour calculer les distances entre espèces dans la configuration de la figure 2.7.

(si on pose $\hat{L}_1^0 = \alpha L_1^0$ et $\hat{L}_2^0 = (1-\alpha)L_2^0$). Donc, pour trouver les longueurs de branches de chacune de ces configurations d'hybridation, nous cherchons la solution des moindres carrés, i.e., nous cherchons à minimiser $(H_n l_n - d_n)(H_n l_n - d_n)^t$, où

$$l_n = (L_1; L_2; \ldots; L_n; \hat{L}_1^0; \hat{L}_2^0)^t,$$

$$d_n = (D[1][2]; D[1][3]; D[2][3]; D[1][4]; \ldots; D[1][n]; D[2][4]; \ldots; D[2][n];$$

$$D[3][4]; \ldots; D[3][n]; D[4][5]; \ldots; D[n-1][n])^t,$$

où les $D[i][j], i < j$ entre $D[4][5]$ et $D[n-1][n]$ sont dans l'ordre lexicographique, et où l'exposant t représente l'opération de transposition des matrices.

De plus, la matrice H_n est la matrice de taille $\frac{n(n-1)}{2} \times n$ correspondant au système d'équations défini par les formules 2.9, 2.10, 2.11 et 2.12.

Par exemple, pour $n = 6$, on obtient :

$$H_6 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & -1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Si la matrice $H_n^t H_n$ est inversible, l'unique solution du problème des moindres carrés est donnée par :

$$l_n^0 = (H_n^t H_n)^{-1} H_n^t d_n.$$

Or, dans notre cas, on obtient pour $n \geq 4$:

$$H_n^t H_n = \begin{pmatrix} & & & & & -1 & 1 \\ & & & & & 1 & -1 \\ & & & & & n-3 & n-3 \\ & & C_n & & & 1 & 1 \\ & & & & & \vdots & \vdots \\ & & & & & 1 & 1 \\ -1 & 1 & n-3 & 1 & \cdots 1 & n-1 & n-5 \\ 1 & -1 & n-3 & 1 & \cdots 1 & n-5 & n-1 \end{pmatrix},$$

où C_n est la matrice de taille $n \times n$ dont tous les coefficients diagonaux sont égaux à $n-1$ et dont tous les coefficients non diagonaux sont égaux à 1. Par exemple, pour $n = 6$, on obtient :

$$H_6^t H_6 = \begin{pmatrix} 5 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ 1 & 5 & 1 & 1 & 1 & 1 & 1 & -1 \\ 1 & 1 & 5 & 1 & 1 & 1 & 3 & 3 \\ 1 & 1 & 1 & 5 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 5 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 5 & 1 & 1 \\ -1 & 1 & 3 & 1 & 1 & 1 & 5 & 1 \\ 1 & -1 & 3 & 1 & 1 & 1 & 1 & 5 \end{pmatrix}.$$

On inverse cette matrice et on obtient pour $(H_n^t H_n)^{-1}$ la matrice suivante :

$$\begin{pmatrix} \frac{2n-5}{2(n-3)(n-2)} & \frac{-1}{2(n-3)(n-2)} & \frac{-1}{2(n-2)} & \frac{-1}{2(n-3)(n-2)} & \cdots & \frac{-1}{2(n-3)(n-2)} & \frac{1}{2(n-3)} & 0 \\ \frac{-1}{2(n-3)(n-2)} & \frac{2n-5}{(n-3)(2n-4)} & \frac{-1}{2n-4} & \frac{-1}{(n-3)(2(n-2))} & \cdots & \frac{-1}{2(n-3)(n-2)} & 0 & \frac{1}{2(n-3)} \\ \frac{-1}{2(n-2)} & \frac{-1}{2(n-2)} & \frac{n-1}{2n-4} & \frac{1}{(n-3)(2n-4)} & \cdots & \frac{1}{2(n-3)(n-2)} & \frac{n-2}{4(n-3)} & \frac{n-2}{4(n-3)} \\ \frac{-1}{(n-3)(2n-4)} & \frac{-1}{2(n-3)(n-2)} & \frac{1}{2(n-3)(n-2)} & & & & \frac{-1}{2(n-3)^2} & \frac{-1}{2(n-3)^2} \\ \vdots & \vdots & \vdots & & W_n & & \vdots & \vdots \\ \frac{-1}{2(n-3)(n-2)} & \frac{-1}{2(n-3)(n-2)} & \frac{1}{2(n-3)(n-2)} & & & & \frac{-1}{2(n-3)^2} & \frac{-1}{2(n-3)^2} \\ \frac{1}{2(n-3)} & 0 & \frac{n-2}{4(n-3)} & \frac{-1}{2(n-3)^2} & \cdots & \frac{-1}{2(n-3)^2} & \frac{(n-2)^2}{4(n-3)^2} & \frac{n-2}{4(n-3)^2} \\ 0 & \frac{1}{2(n-3)} & \frac{n-2}{4(n-3)} & \frac{-1}{2(n-3)^2} & \cdots & \frac{-1}{2(n-3)^2} & \frac{n-2}{4(n-3)^2} & \frac{(n-2)^2}{4(n-3)^2} \end{pmatrix},$$

où W_n est la matrice symétrique de taille $(n-3) \times (n-3)$ dont tous les termes diagonaux sont égaux à $\frac{2(n-3)^2-1}{(n-3)^2(2n-4)}$ et dont tous les termes non diagonaux sont égaux à $-\frac{n-2}{(n-3)^2(2n-4)}$.

Par exemple, pour $n = 6$, on obtient :

$$(H_6^t H_6)^{-1} = \begin{pmatrix} \frac{7}{24} & \frac{-1}{24} & \frac{-1}{8} & \frac{-1}{24} & \frac{-1}{24} & \frac{-1}{24} & 0 & \frac{1}{6} \\ \frac{-1}{24} & \frac{7}{24} & \frac{-1}{8} & \frac{-1}{24} & \frac{-1}{24} & \frac{-1}{24} & \frac{1}{6} & 0 \\ \frac{-1}{8} & \frac{-1}{8} & \frac{5}{8} & \frac{1}{24} & \frac{1}{24} & \frac{1}{24} & \frac{-1}{3} & \frac{-1}{3} \\ \frac{-1}{24} & \frac{-1}{24} & \frac{1}{24} & \frac{17}{72} & \frac{1}{18} & \frac{1}{18} & \frac{-1}{18} & \frac{-1}{18} \\ \frac{-1}{24} & \frac{-1}{24} & \frac{1}{24} & \frac{1}{18} & \frac{17}{72} & \frac{1}{18} & \frac{-1}{18} & \frac{-1}{18} \\ \frac{-1}{24} & \frac{-1}{24} & \frac{1}{24} & \frac{1}{18} & \frac{1}{18} & \frac{17}{72} & \frac{-1}{18} & \frac{-1}{18} \\ 0 & \frac{1}{6} & \frac{-1}{3} & \frac{-1}{18} & \frac{-1}{18} & \frac{-1}{18} & \frac{4}{9} & \frac{1}{9} \\ \frac{1}{6} & 0 & \frac{-1}{3} & \frac{-1}{18} & \frac{-1}{18} & \frac{-1}{18} & \frac{1}{9} & \frac{4}{9} \end{pmatrix}.$$

On multiplie alors par H_n^t pour obtenir la matrice qui donne les longueurs de branches (solutions des moindres carrés) en fonction des distances $D[i][j]$. On obtient ainsi :

$$\begin{aligned} L_1 = {} & \tfrac{1}{n-2} D[1][2] + \tfrac{1}{n-2} \textstyle\sum_{4 \le k \le n} D[1][k] \\ & - \tfrac{1}{(n-2)(n-3)} \textstyle\sum_{4 \le k \le n} D[2][k] - \tfrac{1}{(n-2)(n-3)} R, \end{aligned} \tag{2.13}$$

$$\begin{aligned} L_2 = {} & \tfrac{1}{n-2} D[1][2] - \tfrac{1}{(n-2)(n-3)} \textstyle\sum_{4 \le k \le n} D[1][k] \\ & + \tfrac{1}{n-2} \textstyle\sum_{4 \le k \le n} D[2][k] - \tfrac{1}{(n-2)(n-3)} R, \end{aligned} \tag{2.14}$$

$$\begin{aligned} L_3 = {} & -\tfrac{1}{n-2} D[1][2] + \tfrac{1}{2}(D[1][3] + D[2][3]) \\ & - \tfrac{n-4}{(n-3)(2n-4)} \textstyle\sum_{4 \le k \le n} (D[1][k] + D[2][k]) + \tfrac{1}{(n-2)(n-3)} R, \end{aligned} \tag{2.15}$$

$$\begin{aligned} L_k = {} & -\tfrac{1}{(n-2)(n-3)} D[1][2] + \tfrac{(n-3)^2-(n-4)}{(n-3)^2(n-2)} (D[1][k] + D[2][k]) + \tfrac{n-4}{(n-3)(n-2)} D[3][k] \\ & - \tfrac{n-4}{(n-3)^2(n-2)} \textstyle\sum_{4 \le j \le n; j \ne k} (D[1][j] + D[2][j]) - \tfrac{1}{(n-3)(n-2)} \textstyle\sum_{4 \le j \le n; j \ne k} D[3][j] \\ & + \tfrac{(n-3)^2-(n-5)}{(n-2)(n-3)^2} \textstyle\sum_{4 \le j \le n; j \ne k} D[j][k] - \tfrac{n-5}{(n-2)(n-3)^2} \textstyle\sum_{4 \le i < j \le n; i,j \ne k} D[i][j], \ 4 \le k \le n, \end{aligned} \tag{2.16}$$

$$\begin{aligned} \hat{L}_1^0 = {} & \tfrac{1}{2(n-3)} D[1][2] - \tfrac{1}{2} D[1][3] + \tfrac{n-4}{2(n-3)^2} \textstyle\sum_{4 \le k \le n} D[1][k] - \tfrac{1}{2(n-3)^2} \textstyle\sum_{4 \le k \le n} D[2][k] \\ & + \tfrac{1}{2(n-3)} \textstyle\sum_{4 \le k \le n} D[3][k] - \tfrac{1}{(n-3)^2} R, \end{aligned} \tag{2.17}$$

$$\begin{aligned} \hat{L}_2^0 = {} & \tfrac{1}{2(n-3)} D[1][2] - \tfrac{1}{2} D[2][3] - \tfrac{1}{2(n-3)^2} \textstyle\sum_{4 \le k \le n} D[1][k] + \tfrac{n-4}{2(n-3)^2} \textstyle\sum_{4 \le k \le n} D[2][k] \\ & + \tfrac{1}{2(n-3)} \textstyle\sum_{4 \le k \le n} D[3][k] - \tfrac{1}{(n-3)^2} R, \end{aligned} \tag{2.18}$$

où $R = \sum_{4 \le i < j \le n} D[i][j]$.

Par exemple, pour $n = 6$, on obtient :

$$L_1 = \frac{1}{4}D[1][2] + \frac{1}{4}\sum_{4\leq k\leq 6} D[1][k] - \frac{1}{12}\sum_{4\leq k\leq 6} D[2][k] - \frac{1}{12}R,$$

$$L_2 = \frac{1}{4}D[1][2] - \frac{1}{12}\sum_{4\leq k\leq 6} D[1][k] + \frac{1}{4}\sum_{4\leq k\leq 6} D[2][k] - \frac{1}{12}R,$$

$$L_3 = -\frac{1}{4}D[1][2] + \frac{1}{2}(D[1][3] + D[2][3]) - \frac{1}{12}\sum_{4\leq k\leq 6} (D[1][k] + D[2][k]) + \frac{1}{12}R,$$

$$L_4 = -\tfrac{1}{12}D[1][2] + \tfrac{7}{36}(D[1][4] + D[2][4]) + \tfrac{1}{6}D[3][4] - \tfrac{1}{18}(D[1][5] + D[1][6])$$
$$-\tfrac{1}{18}(D[2][5] + D[2][6]) - \tfrac{1}{12}(D[3][5] + D[3][6]) + \tfrac{2}{9}(D[4][5] + D[4][6]) - \tfrac{1}{36}D[5][6],$$

$$L_5 = -\tfrac{1}{12}D[1][2] + \tfrac{7}{36}(D[1][5] + D[2][5]) + \tfrac{1}{6}D[3][5] - \tfrac{1}{18}(D[1][4] + D[1][6])$$
$$-\tfrac{1}{18}(D[2][4] + D[2][6]) - \tfrac{1}{12}(D[3][4] + D[3][6]) + \tfrac{2}{9}(D[4][5] + D[5][6]) - \tfrac{1}{36}D[4][6],$$

$$L_6 = -\tfrac{1}{12}D[1][2] + \tfrac{7}{36}(D[1][6] + D[2][6]) + \tfrac{1}{6}D[3][6] - \tfrac{1}{18}(D[1][4] + D[1][5])$$
$$-\tfrac{1}{18}(D[2][4] + D[2][5]) - \tfrac{1}{12}(D[3][4] + D[3][5]) + \tfrac{2}{9}(D[4][6] + D[5][6]) - \tfrac{1}{36}D[4][5],$$

$$\hat{L}_1^0 = \frac{1}{6}D[1][2] - \frac{1}{2}D[1][3] + \frac{1}{9}\sum_{4\leq k\leq 6} D[1][k] - \frac{1}{18}\sum_{4\leq k\leq 6} D[2][k] + \frac{1}{6}\sum_{4\leq k\leq 6} D[3][k] - \frac{1}{9}R,$$

$$\hat{L}_2^0 = \frac{1}{6}D[1][2] - \frac{1}{2}D[2][3] - \frac{1}{18}\sum_{4\leq k\leq 6} D[1][k] + \frac{1}{9}\sum_{4\leq k\leq 6} D[2][k] + \frac{1}{6}\sum_{4\leq k\leq 6} D[3][k] - \frac{1}{9}R,$$

où $R = D[4][5] + D[4][6] + D[5][6]$.

Comme on recherche la configuration d'évolution minimale on doit calculer :

$$S^H_{1;2;3} = \sum_{1\leq k\leq n} L_k.$$

On ne prend pas en compte les valeurs de $\hat{L}_1^0$ (respectivement $\hat{L}_2^0$) puisque c'est une « sous-branche » de L_1 (respectivement L_2).

Pour calculer $S^H_{1;2;3}$, on utilise les formules 2.13 à 2.18.

Le coefficient de $D[1][2]$ (qui apparait dans L_1, L_2, L_3 et dans les $n-3$ longueurs L_k pour $k \geq 4$) est ainsi égal à :

$$\frac{1}{n-2} + \frac{1}{n-2} - \frac{1}{n-2} - (n-3) \times \frac{1}{(n-2)(n-3)} = 0.$$

Le coefficient de $D[1][3]$ et de $D[2][3]$ est égal à $\frac{1}{2}$ puisqu'il n'apparait que dans L_3.

Pour $k \geq 4$, le coefficient de $D[1][k]$ et de $D[2][k]$ (qui apparaissent dans L_1, L_2, L_3, L_k et dans les $n-4$ longueurs L_j pour $j \geq 4$, $j \neq k$) est égal à :

$$\frac{1}{n-2} - \frac{1}{(n-2)(n-3)} - \frac{n-4}{(n-3)(2n-4)} + \frac{(n-3)^2-(n-4)}{(n-3)^2(n-2)} - (n-4) \times \frac{n-4}{(n-3)^2(n-2)} = \frac{1}{2(n-3)}.$$

Pour $k \geq 4$, le coefficient de $D[3][k]$ (qui apparait dans L_k et dans les $n-4$ longueurs L_j pour $j \geq 4$, $j \neq k$) est égal à :

$$\frac{n-4}{(n-3)(n-2)} - (n-4) \times \frac{1}{(n-3)(n-2)} = 0.$$

Pour $4 \leq i < j \leq n$, le coefficient de $D[i][j]$ (qui apparait dans L_1, L_2, L_3, L_i, L_j et dans les $n-5$ longueurs L_k pour $k \geq 4$, $k \neq i, j$) est égal à :

$$-\frac{1}{(n-2)(n-3)} - \frac{1}{(n-2)(n-3)} + \frac{1}{(n-2)(n-3)} + 2\frac{(n-3)^2-(n-5)}{(n-2)(n-3)^2} - (n-5) \times \frac{n-5}{(n-2)(n-3)^2} = \frac{1}{n-3}.$$

On obtient donc :

$$S^H_{1;2;3} = \frac{1}{2}(D[1][3]+D[2][3]) + \frac{1}{2(n-3)} \sum_{4 \leq k \leq n} (D[1][k]+D[2][k]) + \frac{1}{n-3} \sum_{4 \leq i < j \leq n} D[i][j].$$

En général, si h est l'hybride de i et j, on obtient ainsi :

$$S^H_{i;j;h} = \frac{1}{2}(D[i][h]+D[j][h]) + \frac{1}{2(n-3)} \sum_{1 \leq k \leq n; k \neq i,j,h} (D[i][k]+D[j][k]) + \frac{1}{n-3} \sum_{1 \leq l < k \leq n; l,k \neq i,j,h} D[l][k]. \quad (2.19)$$

Remarque 2.1. *La formule 2.19 est en fait équivalente à :*

$$S^H_{i;j;h} = \frac{1}{2}(D[i][h]+D[j][h]-D[i][j]) + \frac{1}{2}D[i][j] + \frac{1}{2(n-3)} \sum_{1 \leq k \leq n; k \neq i,j,h} (D[i][k]+D[j][k]) + \frac{1}{n-3} \sum_{1 \leq l < k \leq n; l,k \neq i,j,h} D[l][k].$$

Or, la deuxième ligne de cette formule est la valeur $S_{i;j}$ de NJ classique, où on relie les nœuds i et j parmi $n-1$ nœuds (on a ainsi supprimé le nœud h).

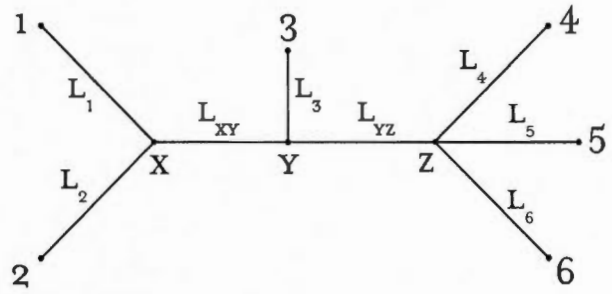


Figure 2.9 Buisson avec trois feuilles regroupées.

De plus, dans le cas où h est l'hybride de i et j, la première ligne de la formule est égale à :

$$L_h = \frac{1}{2}(D[i][h] + D[j][h] - D[i][j]).$$

2.2.2 Les configurations d'arbres

On doit maintenant définir des configurations d'arbres qui peuvent être comparées avec les configurations d'hybrides définies dans la section précédente. On ne peut pas prendre les configurations classiques de l'algorithme NJ puisqu'on aurait alors un degré de liberté de moins ($n+1$ dans le cas de NJ classique au lieu de $n+2$ dans le cas des hybrides). On s'intéresse donc aux configurations présentées dans la figure 2.9. On considère ainsi un triplet d'espèces (comme dans le cas des hybrides), deux nœuds (les nœuds 1 et 2 sur la figure 2.9) étant directement voisins, le troisième (le nœud 3 sur la figure 2.9) étant voisin de l'ancêtre des deux précédents.

Avec les longueurs indiquées sur la figure 2.9, on obtient le système d'équations suivant :

$$D[1][2] = L_1 + L_2, \tag{2.20}$$

$$D[1][3] = L_1 + L_3 + L_{XY}, \tag{2.21}$$

$$D[2][3] = L_2 + L_3 + L_{XY}, \tag{2.22}$$

$$D[k][1] = L_1 + L_{XY} + L_{YZ} + L_k, \ 4 \leq k \leq n, \tag{2.23}$$

$$D[k][2] = L_2 + L_{XY} + L_{YZ} + L_k, \ 4 \leq k \leq n, \tag{2.24}$$

$$D[k][3] = L_3 + L_{YZ} + L_k, \ 4 \leq k \leq n, \tag{2.25}$$

$$D[i][j] = L_i + L_j, \ 4 \leq i < j \leq n. \tag{2.26}$$

En général, si on dispose d'une matrice de distances arbitraire, on ne peut pas résoudre le système d'équations ci-dessus puisque nous avons $\frac{n(n-1)}{2}$ équations et $n+2$ inconnues. Comme dans la section précédente, nous cherchons donc à minimiser :

$$(T_n l_n - d_n)(T_n l_n - d_n)^t,$$

où

$$l_n = (L_1; L_2; \ldots; L_n; L_{XY}; L_{YZ})^t,$$

et où d_n est défini comme précédemment par :

$$d_n = (D[1][2]; D[1][3]; D[2][3]; D[1][4]; \ldots; D[1][n]; D[2][4]; \ldots; D[2][n];$$

$$D[3][4]; \ldots; D[3][n]; D[4][5]; \ldots; D[n-1][n])^t,$$

où les $D[i][j], i < j$ entre $D[4][5]$ et $D[n-1][n]$ sont dans l'ordre lexicographique.

De plus, la matrice T_n est la matrice de taille $\frac{n(n-1)}{2} \times n$ correspondant au système d'équations défini par les formules 2.20 à 2.26.

Par exemple, pour $n = 6$, on obtient :

$$T_6 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Si la matrice $T_n^t T_n$ est inversible, l'unique solution du problème des moindres carrés est donnée par :

$$l_n^0 = (T_n^t T_n)^{-1} T_n^t d_n.$$

Or, dans notre cas, on obtient pour $n \geq 4$:

$$T_n^t T_n = \begin{pmatrix} & & & & n-2 & n-3 \\ & & & & n-2 & n-3 \\ & & & & 2 & n-3 \\ & U_n & & & 2 & 3 \\ & & & & \vdots & \vdots \\ & & & & 2 & 3 \\ n-2 & n-2 & 2 & 2 \cdots 2 & 2(n-2) & 2(n-3) \\ n-3 & n-3 & n-3 & 3 \cdots 3 & 2(n-3) & 3(n-3) \end{pmatrix},$$

où U_n est la matrice de taille $n \times n$ dont tous les coefficients diagonaux sont égaux à $n-1$ et dont tous les coefficients non diagonaux sont égaux à 1. Par exemple, pour $n = 6$, on obtient :

$$T_6^t T_6 = \begin{pmatrix} 5 & 1 & 1 & 1 & 1 & 1 & 4 & 3 \\ 1 & 5 & 1 & 1 & 1 & 1 & 4 & 3 \\ 1 & 1 & 5 & 1 & 1 & 1 & 2 & 3 \\ 1 & 1 & 1 & 5 & 1 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 & 5 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 & 1 & 5 & 2 & 3 \\ 4 & 4 & 2 & 2 & 2 & 2 & 8 & 6 \\ 3 & 3 & 3 & 3 & 3 & 3 & 6 & 9 \end{pmatrix}.$$

On inverse cette matrice et on obtient pour $(T_n^t T_n)^{-1}$ la matrice suivante :

$$\begin{pmatrix} \frac{n}{4(n-2)} & \frac{n-4}{4(n-2)} & 0 & 0 \cdots 0 & \frac{-1}{4} & 0 \\ \frac{n-4}{4(n-2)} & \frac{n}{4(n-2)} & 0 & 0 \cdots 0 & \frac{-1}{4} & 0 \\ 0 & 0 & \frac{n-1}{2(n-2)} & 0 \cdots 0 & \frac{n-6}{8(n-3)} & -\frac{n-4}{8(n-3)} \\ 0 & 0 & 0 & & 0 & \frac{-1}{2(n-3)(n-4)} \\ \vdots & \vdots & \vdots & V_n & \vdots & \vdots \\ 0 & 0 & 0 & & 0 & \frac{-1}{2(n-3)(n-4)} \\ \frac{-1}{4} & \frac{-1}{4} & \frac{n-6}{8(n-3)} & 0 \cdots 0 & \frac{3(n-2)}{8(n-3)} & -\frac{n-2}{8(n-3)} \\ 0 & 0 & -\frac{n-4}{8(n-3)} & \frac{-1}{2(n-3)(n-4)} \cdots \frac{-1}{2(n-3)(n-4)} & -\frac{n-2}{8(n-3)} & \frac{(n-2)^2}{8(n-4)(n-3)} \end{pmatrix},$$

où V_n est la matrice symétrique de taille $(n-3) \times (n-3)$ dont tous les termes diagonaux sont égaux à $\frac{(n-3)^2+1+(n-5)(n-4)}{2(n-2)(n-3)(n-4)}$ et dont tous les termes non diagonaux sont égaux à $-\frac{(n-6)}{2(n-2)(n-3)(n-4)}$.

Par exemple, pour $n = 6$, on obtient :

$$(T_6^t T_6)^{-1} = \begin{pmatrix} \frac{3}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & \frac{-1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & 0 & 0 & 0 & 0 & \frac{-1}{4} & 0 \\ 0 & 0 & \frac{5}{8} & 0 & 0 & 0 & 0 & \frac{-1}{12} \\ 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{-1}{12} \\ 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{-1}{12} \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & \frac{-1}{12} \\ \frac{-1}{4} & \frac{-1}{4} & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{-1}{6} \\ 0 & 0 & \frac{-1}{12} & \frac{-1}{12} & \frac{-1}{12} & \frac{-1}{12} & \frac{-1}{6} & \frac{1}{3} \end{pmatrix}.$$

On multiplie alors par T_n^t pour obtenir la matrice qui donne les longueurs de branches (solutions des moindres carrés) en fonction des distances $D[i][j]$. On obtient ainsi :

$$L_1 = \frac{1}{2}D[1][2] + \frac{1}{2(n-2)} \sum_{3 \le k \le n} (D[1][k] - D[2][k]), \tag{2.27}$$

$$L_2 = \frac{1}{2}D[1][2] - \frac{1}{2(n-2)} \sum_{3 \le k \le n} (D[1][k] - D[2][k]), \tag{2.28}$$

$$L_3 = \frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{4(n-3)} \sum_{4 \le k \le n} (2D[3][k] - D[1][k] - D[2][k]), \tag{2.29}$$

$$\begin{aligned} L_k = {} & \tfrac{n-4}{(n-3)(n-2)}(D[1][k] + D[2][k] + D[3][k]) \\ & - \tfrac{1}{(n-3)(n-2)} \textstyle\sum_{4 \le j \le n; j \ne k} (D[1][j] + D[2][j] + D[3][j]) \\ & + \tfrac{(n-3)(n-5)+3}{(n-4)(n-3)(n-2)} \textstyle\sum_{4 \le j \le n; j \ne k} D[j][k] - \tfrac{n-6}{(n-4)(n-3)(n-2)} \textstyle\sum_{4 \le i < j \le n; i,j \ne k} D[i][j], \quad 4 \le k \le n, \end{aligned} \tag{2.30}$$

$$L_{XY} = -\frac{1}{2}D[1][2] + \frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{4(n-3)} \sum_{4 \le k \le n} (D[1][k] + D[2][k] - 2D[3][k]), \tag{2.31}$$

$$L_{YZ} = -\frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{4(n-3)} \sum_{4 \le k \le n} (D[1][k] + D[2][k] + 2D[3][k]) - \frac{R}{(n-3)(n-4)}, \tag{2.32}$$

où $R = \sum_{4 \le i < j \le n} D[i][j]$.

Par exemple, pour $n = 6$, on obtient :

$$L_1 = \frac{1}{2}D[1][2] + \frac{1}{8} \sum_{3 \le k \le 6} (D[1][k] - D[2][k]),$$

$$L_2 = \frac{1}{2}D[1][2] - \frac{1}{8}\sum_{3\leq k\leq 6}(D[1][k] - D[2][k]),$$

$$L_3 = \frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{12}\sum_{4\leq k\leq 6}(2D[3][k] - D[1][k] - D[2][k]),$$

$$L_4 = \frac{1}{6}(D[1][4]+D[2][4]+D[3][4]) - \frac{1}{12}\sum_{j=5;6}(D[1][j]+D[2][j]+D[3][j]) + \frac{1}{4}\sum_{j=5;6}D[j][4],$$

$$L_5 = \frac{1}{6}(D[1][5]+D[2][5]+D[3][5]) - \frac{1}{12}\sum_{j=4;6}(D[1][j]+D[2][j]+D[3][j]) + \frac{1}{4}\sum_{j=4;6}D[j][5],$$

$$L_6 = \frac{1}{6}(D[1][6]+D[2][6]+D[3][6]) - \frac{1}{12}\sum_{j=4;5}(D[1][j]+D[2][j]+D[3][j]) + \frac{1}{4}\sum_{j=4;5}D[j][6],$$

$$L_{XY} = -\frac{1}{2}D[1][2] + \frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{12}\sum_{4\leq k\leq 6}(D[1][k] + D[2][k] - 2D[3][k]),$$

$$L_{YZ} = -\frac{1}{4}(D[1][3] + D[2][3]) + \frac{1}{12}\sum_{4\leq k\leq 6}(D[1][k] + D[2][k] + 2D[3][k]) - \frac{R}{6},$$

où $R = D[4][5] + D[4][6] + D[5][6]$.

Comme on recherche la configuration d'évolution minimale on doit calculer :

$$S^T_{1;2;3} = \sum_{1\leq k\leq n} L_k + L_{XY} + L_{YZ}.$$

Pour calculer $S^T_{1;2;3}$, on utilise les formules 2.27 à 2.32.

Le coefficient de $D[1][2]$ (qui apparait dans L_1, L_2 etL_{XY}) est ainsi égal à

$$2 \times \frac{1}{2} - \frac{1}{2} = \frac{1}{2}.$$

Le coefficient de $D[1][3]$ et de $D[2][3]$ (qui apparait dans L_3, L_{XY} et L_{YZ}) est égal à

$$\frac{1}{4} + \frac{1}{4} - \frac{1}{4} = \frac{1}{4}.$$

Pour $k \geq 4$, le coefficient de $D[1][k]$ et de $D[2][k]$ (qui apparaissent dans L_1, L_2, L_3, L_k, dans les $n-4$ longueurs L_j pour $j \geq 4$, $j \neq k$, L_{XY} et L_{YZ}) est égal à :

$$\frac{1}{2(n-2)} - \frac{1}{2(n-2)} - \frac{1}{4(n-3)} + \frac{n-4}{(n-3)(n-2)} - (n-4) \times \frac{1}{(n-3)(n-2)} + \frac{1}{4(n-3)} + \frac{1}{4(n-3)}$$
$$= \frac{1}{4(n-3)}.$$

Pour $k \geq 4$, le coefficient de $D[3][k]$ (qui apparait dans L_3, L_k, dans les $n-4$ longueurs L_j pour $j \geq 4$, $j \neq k$, L_{XY} et L_{YZ}) est égal à :

$$\frac{1}{2(n-3)} + \frac{n-4}{(n-3)(n-2)} - (n-4) \times \frac{1}{(n-3)(n-2)} - \frac{1}{2(n-3)} + \frac{1}{2(n-3)}$$
$$= \frac{1}{2(n-3)}.$$

Pour $4 \leq i < j \leq n$, le coefficient de $D[i][j]$ (qui apparait dans L_i, L_j, dans les $n-5$ longueurs L_k pour $k \geq 4$, $k \neq i, j$, et dans L_{YZ}) est égal à :

$$2\frac{(n-3)(n-5)+3}{(n-2)(n-3)(n-4)} - (n-5) \times \frac{n-6}{(n-2)(n-3)(n-4)} - \frac{1}{(n-3)(n-4)}$$
$$= \frac{1}{n-3}.$$

On obtient donc :

$$S^T_{1;2;3} = \frac{1}{4}(D[1][3] + D[2][3] + 2D[1][2]) + \frac{1}{4(n-3)} \sum_{4 \leq k \leq n} (D[1][k] + D[2][k] + 2D[3][k])$$
$$+ \frac{1}{n-3} \sum_{4 \leq i < j \leq n} D[i][j].$$

En général, si h est le voisin de l'ancêtre des voisins i et j, on obtient alors :

$$S^T_{i;j;h} = \frac{1}{4}(D[i][h] + D[j][h] + 2D[i][j]) + \frac{1}{4(n-3)} \sum_{1 \leq k \leq n; k \neq i,j,h} (D[i][k] + D[j][k] + 2D[h][k])$$
$$+ \frac{1}{n-3} \sum_{1 \leq l < k \leq n; l,k \neq i,j,h} D[l][k]. \tag{2.33}$$

Remarque 2.2. *La formule 2.33 est en fait équivalente à :*

$$S^T_{i;j;h} = D[i][j] + \frac{1}{4}(D[i][h] + D[j][h] - D[i][j])$$
$$+ \frac{1}{2(n-3)} \sum_{1 \leq k \leq n; k \neq i,j,h} \left(\frac{1}{2}(D[i][k] + D[j][k] - D[i][j]) + D[h][k] \right)$$
$$+ \frac{1}{n-3} \sum_{1 \leq l < k \leq n; l,k \neq i,j,h} D[l][k].$$

Ainsi, $S^T_{i;j;h}$ est égal à $D[i][j]$ plus la formule de NJ classique si on relie les nœuds m (représentant l'ancêtre commun de i et j si ces derniers sont voisins) et h parmi $n-1$ nœuds, où on a remplacé les nœuds i et j par le nœud m, dont les distances aux nœuds restants sont données par :

$$D[m][k] = \frac{1}{2}(D[i][k] + D[j][k] - D[i][j]).$$

On notera que, si i et j sont voisins, $D[i][j] = L_i + L_j$.

2.2.3 L'algorithme

Nous pouvons maintenant décrire notre algorithme. Nous prenons en entrée une matrice de distances $D = \{D[i][j]\}_{1 \leq i \leq n; 1 \leq j \leq n}$ sur un ensemble de n espèces.

1. $n_A = n$
2. $D_A = D$

   Tant que $(n_A > 3)$
3. On calcule la plus petite valeur $S^H_{i_0^H;j_0^H;h_0^H}$ des $S^H_{i;j;h}$ (formule 2.19), pour $1 \leq i < j \leq n$, $1 \leq h \leq n$, $h \neq i, j$, pour la matrice D_A.
4. On calcule la plus petite valeur $S^T_{i_0^T;j_0^T;h_0^T}$ des $S^T_{i;j;h}$ (formule 2.33), pour $1 \leq i < j \leq n$, $1 \leq h \leq n$, $h \neq i, j$, pour la matrice D_A.
5. Si $S^H_{i_0^H;j_0^H;h_0^H} < S^T_{i_0^T;j_0^T;h_0^T}$, h_0^H est l'hybride de i_0^H et j_0^H. On élimine la ligne et la colonne h_0^H de D_A. On garde en mémoire la longueur

$$L^H_{h_0} = \frac{1}{2}\left(D[i_0^H][h_0^H] + D[j_0^H][h_0^H] - D[i_0^H][j_0^H]\right), \tag{2.34}$$

   qui peut être déduite des formules 2.9, 2.10 et de la formule $D[i_0^H][j_0^H] = L_{i_0^H} + L_{j_0^H}$. On n'utilise pas la formule 2.15 car elle ne nous donne pas la bonne longueur de branche pour l'hybride.
6. Sinon, i_0^T et j_0^T sont voisins. On remplace donc les lignes et les colonnes i_0^T et j_0^T de D_A par une ligne et une colonne correspondant à l'ancêtre X de i_0^T et j_0^T. Les distances de X aux nœuds restants sont calculées par les formules 2.5. On garde en mémoire les longueurs $L_{i_0^T}$ et $L_{j_0^T}$ données par les formules 2.1.
7. $n_A \leftarrow n_A - 1$

   Fin(Tant que)
8. Il reste alors un triplet de nœuds. On calcule alors les trois distances restant à calculer par les formules 2.1.

À la sortie, nous avons soit un arbre phylogénétique classique avec n feuilles, soit un réseau d'hybridation phylogénétique avec les mêmes n sommets terminaux.

Le code de cet algorithme écrit en langage $C++$ est disponible dans l'annexe A.

2.3 Deux exemples

Nous allons maintenant présenter deux exemples de résolution de la phylogénie de 6 espèces par notre algorithme.

2.3.1 Un exemple d'arbre

Nous reprenons l'exemple de l'arbre additif à 6 feuilles de la figure 1.2. On prend donc en entrée la matrice de distances suivante :

$$D = \left(\begin{array}{c|cccccc} 1 & 0 & 4 & 5 & 6 & 8 & 9 \\ 2 & 4 & 0 & 7 & 8 & 10 & 11 \\ 3 & 5 & 7 & 0 & 5 & 7 & 8 \\ 4 & 6 & 8 & 5 & 0 & 6 & 7 \\ 5 & 8 & 10 & 7 & 6 & 0 & 3 \\ 6 & 9 & 11 & 8 & 7 & 3 & 0 \end{array}\right).$$

La première colonne indique le nom du nœud (i.e., de l'espèce) correspondant à chaque ligne. On calcule toutes les valeurs des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$. Dans le tableau 2.4, on rappelle toutes les valeurs des $S_{i,j}$ déjà présentées dans le tableau 2.1, ainsi que le minimum des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour chaque couple (i,j). On n'indique pas toutes les valeurs des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ car cela prendrait trop de place. Par exemple, pour le couple $(1,2)$, la plus petite valeur de $S^T_{i;j;h}$ est $S^T_{1;2;3} = 18$ (dans le tableau, on indique entre parenthèses la (ou les) valeur(s) de h pour laquelle (lesquelles) le minimum est atteint). De même, pour le couple $(3,4)$, la plus petite valeur de $S^H_{i;j;h}$ est $S^H_{3;4;1} = S^H_{3;4;2} = \frac{62}{3}$. On constate que, pour chaque couple (i,j), le minimum des $S^T_{i;j;h}$ (respectivement des $S^H_{i;j;h}$) est toujours inférieur à $S_{i;j}$. On constate également que, pour chaque couple (i,j), le minimum des $S^T_{i;j;h}$ et le minimum des $S^H_{i;j;h}$ sont atteints pour la (ou les) même(s) valeur(s) de h. On reviendra sur ces points dans le chapitre suivant.

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1\le h\le 6\\ h\neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1\le h\le 6\\ h\neq i,j}}$
$(1,2)$	19	18 $(h=3)$	20 $(h=3)$
$(1,3)$	20,5	19 $(h=2)$	18 $(h=2)$
$(1,4)$	21	20 $(h=2)$	$\frac{56}{3}\simeq 18,67$ $(h=2)$
$(1,5)$	21,75	$\frac{127}{6}\simeq 21,17$ $(h=6)$	19 $(h=6)$
$(1,6)$	21,75	$\frac{127}{6}\simeq 21,17$ $(h=5)$	19 $(h=5)$
$(2,3)$	20,5	19 $(h=1)$	18 $(h=1)$
$(2,4)$	21	20 $(h=1)$	$\frac{56}{3}\simeq 18,67$ $(h=1)$
$(2,5)$	21,75	$\frac{127}{6}\simeq 21,17$ $(h=6)$	19 $(h=6)$
$(2,6)$	21,75	$\frac{127}{6}\simeq 21,17$ $(h=5)$	19 $(h=5)$
$(3,4)$	20,5	$\frac{61}{3}\simeq 20,33$ $(h=1,2)$	$\frac{62}{3}\simeq 20,67$ $(h=1,2)$
$(3,5)$	21,25	$\frac{121}{6}\simeq 20,17$ $(h=6)$	$\frac{55}{3}\simeq 18,33$ $(h=6)$
$(3,6)$	21,25	$\frac{121}{6}\simeq 20,17$ $(h=5)$	$\frac{55}{3}\simeq 18,33$ $(h=5)$
$(4,5)$	20,75	$\frac{115}{6}\simeq 19,17$ $(h=6)$	$\frac{53}{3}\simeq 17,67$ $(h=6)$
$(4,6)$	20,75	$\frac{115}{6}\simeq 19,17$ $(h=5)$	$\frac{53}{3}\simeq 17,67$ $(h=5)$
$(5,6)$	18,5	$\frac{53}{3}\simeq 17,67$ $(h=4)$	$\frac{62}{3}\simeq 20,67$ $(h=4)$

Tableau 2.4 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1\le h\le 6$, $h\neq i,j$, pour la matrice D.

La plus petite valeur des $S^T_{i;j;h}$ est $S^T_{5;6;4} = \frac{53}{3}$, et la plus petite valeur des $S^H_{i;j;h}$ est $S^H_{4;5;6} = S^H_{4;6;5} = \frac{53}{3}$. Comme on a égalité entre le meilleur couple de voisins et le meilleur hybride, on choisit de joindre les nœuds 5 et 6 (comme dans le cas de NJ classique).

Si on appelle X leur ancêtre commun, on obtient alors l'arbre (b) de la figure 2.3 à partir du buisson (a) de cette même figure. On remplace donc dans la matrice de distances les nœuds 5 et 6 par le nœud X en calculant les distances de X aux nœuds restants par la formule 2.5. On obtient ainsi la matrice suivante :

$$D_1 = \left(\begin{array}{c|ccccc} 1 & 0 & 4 & 5 & 6 & 8,5 \\ 2 & 4 & 0 & 7 & 8 & 10,5 \\ 3 & 5 & 7 & 0 & 5 & 7,5 \\ 4 & 6 & 8 & 5 & 0 & 6,5 \\ X & 8,5 & 10,5 & 7,5 & 6,5 & 0 \end{array}\right).$$

On se retrouve également avec un buisson de taille 5 contenant les nœuds 1, 2, 3, 4 et X. On calcule toutes les valeurs des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$ qu'on résume dans le tableau 2.5.

La plus petite valeur des $S^T_{i;j;h}$ est $S^T_{1;2;3} = S^T_{4;X;3} = 15,5$, et la plus petite valeur des $S^H_{i;j;h}$ est $S^H_{3;X;4} = S^H_{1;3;2} = 15,5$. On choisit donc par exemple de joindre les nœuds 1 et 2. Notons que, contrairement à la méthode NJ classique, on aurait pu également joindre les nœuds 4 et X. En fait, comme $n = 5$, les deux configurations sont équivalentes dans le cas des configurations généralisées que nous avons introduites. Si on appelle Y leur ancêtre commun, on obtient alors l'arbre (c) de la figure 2.3. On remplace donc dans la matrice de distances les nœuds 1 et 2 par le nœud Y en calculant les distances de Y aux nœuds restants par la formule 2.5. On obtient ainsi la matrice suivante :

$$D_2 = \left(\begin{array}{c|cccc} Y & 0 & 6 & 7 & 9,5 \\ 3 & 6 & 0 & 5 & 7,5 \\ 4 & 7 & 5 & 0 & 6,5 \\ X & 9,5 & 7,5 & 6,5 & 0 \end{array}\right),$$

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1\leq h\leq 5\\ h\neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1\leq h\leq 5\\ h\neq i,j}}$
$(1,2)$	$\frac{95}{6}\simeq 15,83$	$15,5$ $(h=3)$	$17,5$ $(h=3)$
$(1,3)$	$\frac{103}{6}\simeq 17,17$	$16,5$ $(h=2)$	$15,5$ $(h=2)$
$(1,4)$	$17,5$	$17,25$ $(h=2)$	16 $(h=2)$
$(1,X)$	$17,5$	$17,25$ $(h=2)$	16 $(h=2)$
$(2,3)$	$\frac{103}{6}\simeq 17,17$	$16,5$ $(h=1)$	$15,5$ $(h=1)$
$(2,4)$	$17,5$	$17,25$ $(h=1)$	16 $(h=1)$
$(2,X)$	$17,5$	$17,25$ $(h=1)$	16 $(h=1)$
$(3,4)$	$\frac{101}{6}\simeq 16,83$	16 $(h=X)$	$15,5$ $(h=X)$
$(3,X)$	$\frac{101}{6}\simeq 16,83$	16 $(h=4)$	$15,5$ $(h=4)$
$(4,X)$	$\frac{97}{6}\simeq 16,17$	$15,5$ $(h=3)$	$16,5$ $(h=3)$

Tableau 2.5 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1\leq h\leq 5$, $h\neq i,j$, pour la matrice D_1.

ainsi qu'un buisson de taille 4. On calcule toutes les valeurs des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$ qu'on résume dans le tableau 2.6.

La plus petite valeur des $S^T_{i;j;h}$ (respectivement des $S^H_{i;j;h}$) est donc égale à $13,5$. On remarque que les valeurs des $S_{i,j}$ et des $S^T_{i;j;h}$ sont identiques. En fait, pour $n=4$, les configurations généralisées et les configurations de NJ classique sont également identiques, ce qui explique ces égalités.

Comme dans le cas de NJ classique, on peut ainsi joindre les nœuds X et 4, ou les nœuds Y et 3, ce qui revient au même. On trouve alors l'arbre (d) de la figure 2.3, où toutes les longueurs de branches sont calculées à chaque itération avec les formules 2.27 et 2.28.

On remarque qu'on retrouve l'arbre additif de la figure 1.2. On généralisera ce résultat dans le prochain chapitre.

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1\leq h\leq 4\\ h\neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1\leq h\leq 4\\ h\neq i,j}}$
$(Y,3)$	13,5	13,5 $(h=4,X)$	14,5 $(h=4,X)$
$(Y,4)$	14	14 $(h=3,X)$	13,5 $(h=3,X)$
(Y,X)	14	14 $(h=3,4)$	13,5 $(h=3,4)$
$(3,4)$	14	14 $(h=Y,X)$	13,5 $(h=Y,X)$
$(3,X)$	14	14 $(h=Y,4)$	13,5 $(h=Y,4)$
$(4,X)$	13,5	13,5 $(h=Y,3)$	14,5 $(h=Y,3)$

Tableau 2.6 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1\leq h\leq 4$, $h\neq i,j$, pour la matrice D_2.

2.3.2 Un exemple de réseau

Nous prenons l'exemple du réseau à 6 feuilles de la figure 2.10. Sur cette figure, on représente à droite l'arbre additif à 5 feuilles sous-jacent. À gauche de cette figure, on trouve l'hybride 1 entre les espèces parentes 2 et 3.

On calcule les distances deux à deux entre toutes les espèces (les distances entre l'hybride 1 et les autres espèces sont calculées avec les formules 2.6 à 2.8), et on obtient en entrée la matrice de distances suivante :

$$D^H = \left(\begin{array}{c|cccccc} 1 & 0 & 6 & 5 & 7 & 9 & 10 \\ 2 & 6 & 0 & 7 & 8 & 10 & 11 \\ 3 & 5 & 7 & 0 & 5 & 7 & 8 \\ 4 & 7 & 8 & 5 & 0 & 6 & 7 \\ 5 & 9 & 10 & 7 & 6 & 0 & 3 \\ 6 & 10 & 11 & 8 & 7 & 3 & 0 \end{array}\right).$$

On calcule toutes les valeurs des $S_{i;j}$, des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$ présentées dans le tableau 2.7.

La plus petite valeur des $S^T_{i;j;h}$ est $S^T_{5;6;4} = \frac{56}{3}$, et la plus petite valeur des $S^H_{i;j;h}$ est

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1\le h\le 6\\ h\neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1\le h\le 6\\ h\neq i,j}}$
$(1,2)$	$20,375$	$19,25\ (h=3)$	$20,5\ (h=3)$
$(1,3)$	$21,125$	$19,75\ (h=2)$	$19,5\ (h=2)$
$(1,4)$	22	$\frac{127}{6}\simeq 21,17\ (h=2)$	$20\ (h=2)$
$(1,5)$	$22,75$	$\frac{133}{6}\simeq 22,17\ (h=6)$	$20\ (h=6)$
$(1,6)$	$22,75$	$\frac{133}{6}\simeq 22,17\ (h=5)$	$20\ (h=5)$
$(2,3)$	$21,5$	$20\ (h=1)$	$19\ (h=1)$
$(2,4)$	$21,875$	$\frac{253}{12}\simeq 21,08\ (h=1)$	$\frac{121}{6}\simeq 20,17\ (h=1)$
$(2,5)$	$22,625$	$\frac{253}{12}\simeq 21,92\ (h=6)$	$\frac{119}{6}\simeq 19,83\ (h=6)$
$(2,6)$	$22,625$	$\frac{263}{12}\simeq 21,92\ (h=5)$	$\frac{119}{6}\simeq 19,83\ (h=5)$
$(3,4)$	$21,625$	$21,25\ (h=1)$	$\frac{127}{6}\simeq 21,17\ (h=1)$
$(3,5)$	$22,375$	$\frac{257}{12}\simeq 21,42\ (h=6)$	$19,5\ (h=6)$
$(3,6)$	$22,375$	$\frac{257}{12}\simeq 21,42\ (h=5)$	$19,5\ (h=5)$
$(4,5)$	$21,75$	$\frac{121}{6}\simeq 20,17\ (h=6)$	$\frac{56}{3}\simeq 18,67\ (h=6)$
$(4,6)$	$21,75$	$\frac{121}{6}\simeq 20,17\ (h=5)$	$\frac{56}{3}\simeq 18,67\ (h=5)$
$(5,6)$	$19,5$	$\frac{56}{3}\simeq 18,67\ (h=4)$	$\frac{65}{3}\simeq 21,67\ (h=4)$

Tableau 2.7 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1\le h\le 6$, $h\neq i,j$, pour la matrice D^H.

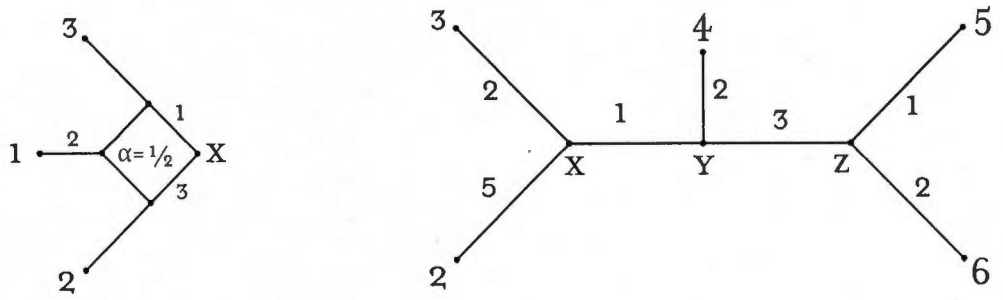


Figure 2.10 Un arbre additif à 5 feuilles auquel on rajoute un hybride.

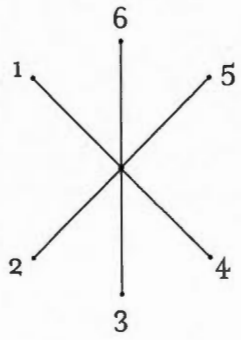
$S^H_{4;5;6} = S^H_{4;6;5} = \frac{56}{3}$. Comme on a égalité entre le meilleur couple de voisins et le meilleur hybride, on choisit de joindre les nœuds 5 et 6.

Si on appelle X leur ancêtre commun, on obtient alors l'arbre (b) de la figure 2.11 à partir du buisson (a) de cette même figure. On remplace donc dans la matrice de distances les nœuds 5 et 6 par le nœud X en calculant les distances de X aux nœuds restants par la formule 2.5. On obtient ainsi la matrice suivante :

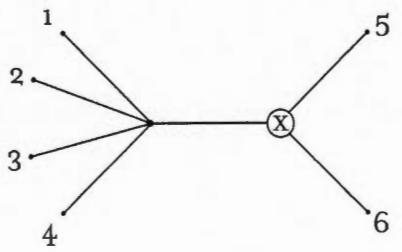
$$D_1^H = \left(\begin{array}{c|ccccc} 1 & 0 & 6 & 5 & 7 & 9,5 \\ 2 & 6 & 0 & 7 & 8 & 10,5 \\ 3 & 5 & 7 & 0 & 5 & 7,5 \\ 4 & 7 & 8 & 5 & 0 & 6,5 \\ X & 9,5 & 10,5 & 7,5 & 6,5 & 0 \end{array}\right).$$

On se retrouve également avec un buisson de taille 5 contenant les nœuds 1, 2, 3, 4 et X. On calcule toutes les valeurs des $S_{i;j}$, des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$, qu'on résume dans le tableau 2.8.

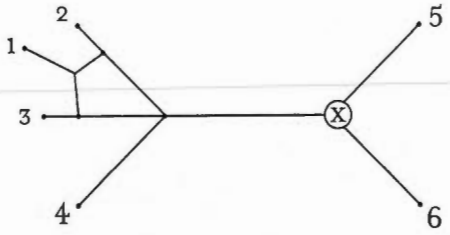
La plus petite valeur des $S^T_{i;j;h}$ est $S^T_{1;2;3} = S^T_{4;X;3} = 16,75$, et la plus petite valeur des $S^H_{i;j;h}$ est $S^H_{2;3;1} = 16,5$. L'espèce 1 est donc l'hybride des espèces 2 et 3. On obtient alors



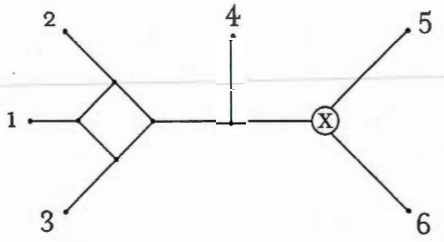
(a, pas 1)



(b, pas 2)



(c, pas 3)



(d, pas 4)

Figure 2.11 La suite des réseaux obtenus par le nouvel algorithme pour le réseau de la figure 2.10.

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1\le h\le 5\\ h\neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1\le h\le 5\\ h\neq i,j}}$
$(1,2)$	$\frac{93}{6}\simeq 17,17$	$16,75\ (h=3)$	$18\ (h=3)$
$(1,3)$	$\frac{53}{3}\simeq 17,83$	$17,25\ (h=2)$	$17\ (h=2)$
$(1,4)$	$18,5$	$18,375\ (h=2)$	$17,25\ (h=2)$
$(1,X)$	$18,5$	$18,375\ (h=2)$	$17,25\ (h=2)$
$(2,3)$	$\frac{99}{6}\simeq 18,17$	$17,5\ (h=1)$	$16,5\ (h=1)$
$(2,4)$	$\frac{55}{3}\simeq 18,33$	$18,125\ (h=X)$	$17,25\ (h=X)$
$(2,X)$	$\frac{55}{3}\simeq 18,33$	$18,125\ (h=4)$	$17,25\ (h=4)$
$(3,4)$	18	$17,375\ (h=X)$	$16,75\ (h=X)$
$(3,X)$	18	$17,375\ (h=4)$	$16,75\ (h=4)$
$(4,X)$	$\frac{93}{6}\simeq 17,17$	$16,75\ (h=3)$	$18\ (h=3)$

Tableau 2.8 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1\le h\le 5$, $h\neq i,j$, pour la matrice D_1^H.

l'arbre (c) de la figure 2.11 avec un hybride dont les espèces parentes sont 2 et 3. On calcule la distance de l'hybride 2 à la bifurcation d'hybridation par la formule 2.34 :

$$L_2=\frac{1}{2}(D[1][2]+D[1][3]-D[2][3])=\frac{1}{2}(6+5-7)=2.$$

On supprime alors dans la matrice de distances le nœud 1 et on obtient ainsi la matrice suivante :

$$D_2^H=\left(\begin{array}{c|cccc} 2 & 0 & 7 & 8 & 10,5\\ 3 & 7 & 0 & 5 & 7,5\\ 4 & 8 & 5 & 0 & 6,5\\ X & 10,5 & 7,5 & 6,5 & 0 \end{array}\right),$$

ainsi qu'un buisson de taille 4. On calcule toutes les valeurs des $S_{i;j}$, des $S^H_{i;j;h}$ et des $S^T_{i;j;h}$, qu'on résume dans le tableau 2.9.

(i,j)	$S_{i;j}$	Minimum des $\{S^T_{i;j;h}\}_{\substack{1 \leq h \leq 4 \\ h \neq i,j}}$	Minimum des $\{S^H_{i;j;h}\}_{\substack{1 \leq h \leq 4 \\ h \neq i,j}}$
$(2,3)$	$14,5$	$14,5$ $(h = 4, X)$	$15,5$ $(h = 4, X)$
$(2,4)$	15	15 $(h = 3, X)$	$14,5$ $(h = 3, X)$
$(2,X)$	15	15 $(h = 3,4)$	$14,5$ $(h = 3,4)$
$(3,4)$	15	15 $(h = 2, X)$	$14,5$ $(h = 2, X)$
$(3,X)$	15	15 $(h = 2,4)$	$14,5$ $(h = 2,4)$
$(4,X)$	$14,5$	$14,5$ $(h = 2,3)$	$15,5$ $(h = 2,3)$

Tableau 2.9 Valeurs des $S_{i;j}$ et des minima des $S^T_{i;j;h}$ et $S^H_{i;j;h}$ pour $1 \leq h \leq 4$, $h \neq i,j$, pour la matrice D_2^H.

La plus petite valeur des $S^T_{i;j;h}$ (respectivement des $S^H_{i;j;h}$) est donc égale à $14,5$. On peut ainsi joindre les nœuds X et 4, ou les nœuds 2 et 3, ce qui revient au même. On trouve alors l'arbre (d) de la figure 2.11, où toutes les longueurs de branches (sauf la longueur de branche de l'hybride) sont calculées à chaque itération avec les formules 2.27 et 2.28.

On remarque qu'on retrouve bien le réseau de départ.

CHAPITRE III

PRINCIPAUX RÉSULTATS SUR LE NOUVEL ALGORITHME

On notera tout d'abord que la complexité de notre nouvel algorithme est en $O(n^4)$ puisqu'on fait $n-3$ itérations, et que la i-ème itération nécessite de considérer tous les triplets parmi $n-i+1$ éléments.

On va maintenant étudier la capacité du nouvel algorithme à retrouver des réseaux d'hybridation.

3.1 Le cas des arbres

On va tout d'abord vérifier que l'algorithme retrouve un arbre phylogénétique quand la matrice de distances donnée à l'entrée de l'algortihme est une matrice de distances d'arbre.

Regardons tout d'abord le cas où on considère seulement 4 espèces. On a alors un arbre semblable à celui de la figure 3.1.

Les distances entre espèces sont alors données par les formules :

$$D[1][2] = L_1 + L_2 \ , \ D[3][4] = L_3 + L_4,$$

$$D[1][3] = L_1 + L_{XY} + L_3 \ , \ D[1][4] = L_1 + L_{XY} + L_4,$$

$$D[2][3] = L_2 + L_{XY} + L_3 \ , \ D[2][4] = L_2 + L_{XY} + L_4.$$

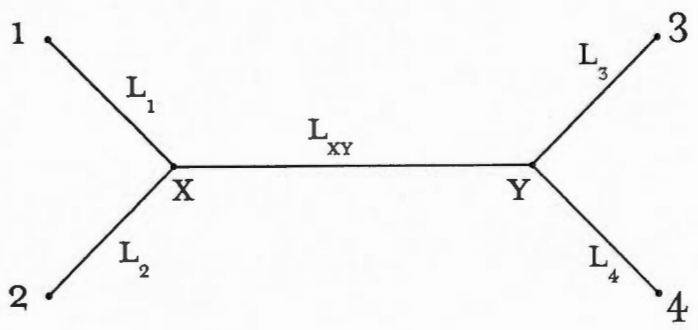


Figure 3.1 Un arbre phylogénétique de taille 4.

Notons que, dans ce cas, $S^T_{i;j;h}$ ne dépend pas de h. On a de plus :

$$S^T_{1;2;3} = S^T_{1;2;4} = S^T_{3;4;1} = S^T_{3;4;2} = S_{1;2} = S_{3;4}$$
$$= \frac{1}{2}\left(D[1][2] + D[3][4]\right) + \frac{1}{4}\left(D[1][3] + D[1][4] + D[2][3] + D[2][4]\right).$$

Si on remplace les distances par les longueurs de branches dans l'arbre, on obtient :

$$S_{1;2} = S_{3;4} = L_1 + L_2 + L_{XY} + L_3 + L_4.$$

On obtient de même :

$$S_{1;3} = S_{2;4} = L_1 + L_2 + \frac{3}{2}L_{XY} + L_3 + L_4.$$
$$S_{1;4} = S_{2;3} = L_1 + L_2 + \frac{3}{2}L_{XY} + L_3 + L_4.$$

On voit ainsi que la plus petite valeur des $S^T_{i;j;h}$ nous donne le bon couple de voisins (1 et 2, ou 3 et 4) et donc le bon arbre. Il faut cependant également démontrer qu'aucune configuration d'hybride ne donne une valeur de $S^H_{i;j;h}$ plus petite que :

$$S_{1;2} = S_{3;4} = L_1 + L_2 + L_{XY} + L_3 + L_4.$$

On calcule donc toutes les valeurs de $S^H_{i;j;h}$. On remarque qu'elles ne dépendent pas non plus de h et on obtient :

$$S^H_{1;2;3} = S^H_{1;2;4} = S^H_{3;4;1} = S^H_{3;4;2} =$$

$$= \frac{1}{2}\left(D[1][3] + D[1][4] + D[2][3] + D[2][4]\right) = L_1 + L_2 + 2L_{XY} + L_3 + L_4.$$

On obtient de même :

$$S^H_{1;3;4} = S^H_{1;3;2} = S^H_{2;4;1} = S^H_{2;4;3} = L_1 + L_2 + L_{XY} + L_3 + L_4,$$

$$S^H_{1;4;2} = S^H_{1;4;3} = S^H_{2;3;1} = S^H_{2;3;4} = L_1 + L_2 + L_{XY} + L_3 + L_4.$$

Aucune de ces trois valeurs n'étant plus petite que $S_{1;2}$, on obtient bien l'arbre initial en joignant les espèces 1 et 2.

Nous avons testé notre algorithme sur plusieurs centaines d'arbres phylogénétiques, et notre algorithme retrouvait le bon arbre sans aucun hybride dans tous les cas. Nous n'avons cependant pas réussi à démontrer cette propriété.

Pour réaliser nos tests, nous avons utilisé un algorithme pour générer des arbres aléatoires que nous ne détaillerons pas ici. Nous entrons ainsi la taille de l'arbre désirée n, nous obtenons un arbre de taille n ayant une topologie et des longueurs de branches aléatoires, puis nous appliquons notre nouvel algorithme à la matrice de distances correspondant à cet arbre. Nous avons ainsi fait des tests sur des centaines d'arbres de tailles variant de 4 à 100. Dans tous les cas, l'arbre aléatoire initialement généré et l'arbre que reconstruisait notre algorithme étaient identiques.

Bien que nous n'avons pas réussi à prouver la capacité de notre algorithme à retrouver tous les arbres phylogénétiques, nous allons quand même démontrer quelques résultats que nous avons déjà constatés sur les exemples du chapitre 2, et qui permettent de mieux comprendre pourquoi notre nouvel algorithme retrouve tous les arbres phylogénétiques quand la matrice au départ est une matrice de distances d'arbre..

Proposition 3.1. *Pour toute matrice de distances de taille n, et pour tout couple $(i,j), 1 \leq i < j \leq n$, le minimum des valeurs des $S^T_{i;j;h}$ est atteint pour la (les) même(s) valeur(s) de h $(1 \leq h \leq n, h \neq i, h \neq j)$ que le minimum des valeurs des $S^H_{i;j;h}$ $(1 \leq h \leq n, h \neq i, h \neq j)$.*

Démonstration. Soit D une matrice de distances de taille n. On va donner une autre formule pour $S^T_{i;j;h}$ et $S^H_{i;j;h}$.

On pose

$$\Sigma_i = \sum_{1\leq j\leq n} D[i][j],$$

et

$$\Sigma = \sum_{1\leq i\leq n} \Sigma_i = \sum_{1\leq i\leq n, 1\leq j\leq n} D[i][j].$$

On a alors :

$$S^H_{i;j;h} = \frac{1}{2}(D[i][h] + D[j][h]) + \frac{1}{2(n-3)} \sum_{1\leq k\leq n; k\neq i,j,h} (D[i][k] + D[j][k])$$

$$+\frac{1}{n-3} \sum_{1\leq l<k\leq n; l,k\neq i,j,h} D[l][k]$$

$$= \frac{1}{2}(D[i][h] + D[j][h]) + \frac{1}{2(n-3)} (\Sigma_i + \Sigma_j - D[i][h] - D[j][h] - 2D[i][j])$$

$$+\frac{1}{n-3} \times \frac{1}{2} (\Sigma - 2\Sigma_i - 2\Sigma_j - 2\Sigma_h + 2D[i][h] + 2D[j][h] + 2D[i][j])$$

$$= \frac{1}{2(n-3)} ((n-3)(D[i][h] + D[j][h]) + \Sigma_i + \Sigma_j - D[i][h] - D[j][h] - 2D[i][j]$$

$$+\Sigma - 2\Sigma_i - 2\Sigma_j - 2\Sigma_h + 2D[i][h] + 2D[j][h] + 2D[i][j]),$$

et on obtient donc la formule :

$$S^H_{i;j;h} = \frac{1}{2(n-3)} (\Sigma + (n-2)(D[i][h] + D[j][h]) - \Sigma_i - \Sigma_j - 2\Sigma_h). \qquad (3.1)$$

On trouve de même :

$$S^T_{i;j;h} = \frac{1}{4}(D[i][h]+D[j][h]+2D[i][j]) + \frac{1}{4(n-3)} \sum_{1\leq k\leq n; k\neq i,j,h} (D[i][k]+D[j][k]+2D[h][k])$$

$$+\frac{1}{n-3} \sum_{1\leq l<k\leq n; l,k\neq i,j,h} D[l][k]$$

$$= \frac{1}{4}(D[i][h]+D[j][h]+2D[i][j]) + \frac{1}{4(n-3)}(\Sigma_i+\Sigma_j+2\Sigma_h-2D[i][j]-3D[i][h]-3D[j][h])$$

$$+\frac{1}{n-3} \times \frac{1}{2} (\Sigma - 2\Sigma_i - 2\Sigma_j - 2\Sigma_h + 2D[i][h] + 2D[j][h] + 2D[i][j])$$

$$= \frac{1}{4(n-3)} ((n-3)(D[i][h] + D[j][h] + 2D[i][j]) + \Sigma_i + \Sigma_j + 2\Sigma_h - 2D[i][j] - 3D[i][h]$$

$$-3D[j][h] + 2\Sigma - 4\Sigma_i - 4\Sigma_j - 4\Sigma_h + 4D[i][h] + 4D[j][h] + 4D[i][j]),$$

et on obtient donc la formule :

$$S^T_{i;j;h} = \frac{1}{4(n-3)} (2\Sigma + (n-2)(D[i][h] + D[j][h] + 2D[i][j]) - 3\Sigma_i - 3\Sigma_j - 2\Sigma_h). \tag{3.2}$$

Si on fixe i et j, on voit alors que pour minimiser $S^H_{i;j;h}$ et $S^T_{i;j;h}$, il faut minimiser

$$(n-2)(D[i][h] + D[j][h]) - 2\Sigma_h,$$

puisque c'est la seule partie des formules 3.1 et 3.2 qui dépend de h.

□

Remarque 3.1. *Ce sont les formules 3.1 et 3.2 qu'on utilise pour programmer notre algorithme.*

Nous pouvons également démontrer le résultat suivant sur les relations entre $S^H_{i;j;h}$ et $S^T_{i;j;h}$.

Proposition 3.2. *Pour toute matrice de distances d'arbre D de taille n, et pour tout triplet $(i,j,h), 1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$, on a les relations suivantes :*

$$S^T_{i;j;h} = S^H_{i;h;j} \quad si \ \ i \ \ et \ \ j \ \ sont \ \ voisins,$$
$$S^T_{i;j;h} > S^H_{i;h;j} \quad si \ \ j \ \ et \ \ h \ \ sont \ \ voisins,$$
$$S^T_{i;j;h} < S^H_{i;h;j} \quad si \ \ i \ \ et \ \ h \ \ sont \ \ voisins.$$

Démonstration. En utilisant les formules 3.1 et 3.2, on obtient :

$$S^T_{i;j;h} - S^H_{i;h;j} = \frac{1}{4(n-3)} (2\Sigma + (n-2)(D[i][h] + D[j][h] + 2D[i][j]) - 3\Sigma_i - 3\Sigma_j - 2\Sigma_h$$

$$-2(\Sigma + (n-2)(D[i][j] + D[j][h]) - \Sigma_i - \Sigma_h - 2\Sigma_j))$$

$$= \frac{1}{4(n-3)} \left((n-2)(D[i][h] - D[j][h]) + \Sigma_j - \Sigma_i \right)$$

$$= \frac{1}{4(n-3)} \sum_{1 \leq k \leq n; k \neq i,j} \left[D[i][h] + D[j][k] - (D[j][h] + D[i][k]) \right].$$

Or, chacun des termes dans la somme ci-dessus est nul si i et j sont des voisins, positif si j et h sont des voisins, et négatif si i et h sont des voisins, ce qui démontre notre proposition.

□

Remarque 3.2. *Dans la proposition précédente, et dans tout ce qui suit, on pose pour tout triplet* $(i,j,h), 1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$,

$$S^T_{j;i;h} = S^T_{i;j;h} \quad et \quad S^H_{j;i;h} = S^H_{i;j;h}.$$

Dans les exemples du chapitre 2, nous avons constaté que les minima des valeurs de $S^H_{i;j;h}$ et de $S^T_{i;j;h}$ étaient toujours plus petites que les minima des valeurs de $S_{i;j}$. Nous avons en fait le résultat plus général suivant :

Proposition 3.3. *Pour toute matrice de distances d'arbre* D *de taille* n, *et pour tout triplet* $(i,j,h), 1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$, *on a les relations suivantes :*

$$S^T_{i;j;h} < S_{i;j} \quad et \quad S^H_{i;j;h} < \ Min\ (S_{i;h}, S_{j;h}).$$

Démonstration. On montre tout d'abord la formule suivante :

$$S_{i;j} = \frac{1}{2} D[i][j] + \frac{1}{2(n-2)} \sum_{1 \leq k \leq n; k \neq i,j} [D[i][k] + D[j][k]] + \frac{1}{n-2} \sum_{1 \leq k < l \leq n; k,l \neq i,j} D[k][l]$$

$$= \frac{1}{2} D[i][j] + \frac{1}{2(n-2)} (\Sigma_i + \Sigma_j - 2D[i][j]) + \frac{1}{n-2} \times \frac{1}{2} \left(\Sigma - 2\Sigma_i - 2\Sigma_j + 2D[i][j] \right),$$

d'où on obtient :

$$S_{i;j} = \frac{1}{2(n-2)} (\Sigma + (n-2)D[i][j] - \Sigma_i - \Sigma_j). \tag{3.3}$$

En utilisant la formule 3.2, on obtient alors :

$$S^T_{i;j;h} = S_{i;j} + \frac{1}{4(n-3)}\left((n-2)(D[i][h]+D[j][h]) - \Sigma_i - \Sigma_j - 2\Sigma_h\right)$$

$$= S_{i;j} + \frac{1}{4(n-3)}\left(\sum_{1\leq k\leq n; k\neq i,h}\left[D[i][h] - (D[i][k]+D[h][k])\right]\right.$$

$$\left. + \sum_{1\leq k\leq n; k\neq j,h}\left[D[j][h] - (D[j][k]+D[h][k])\right]\right).$$

Or, pour tout arbre additif, tous les termes dans les deux sommes ci-dessus sont négatifs, et on obtient donc bien la première inégalité.

Pour montrer la deuxième inégalité, on utilise la formule 3.1 pour obtenir :

$$S^H_{i;j;h} = S_{i;h} + \frac{1}{2(n-3)}\left((n-2)D[j][h] - \Sigma_j - \Sigma_h\right)$$

$$= S_{i;h} + \frac{1}{2(n-3)}\sum_{1\leq k\leq n; k\neq j,h}\left[D[j][h] - (D[j][k]+D[h][k])\right].$$

Pour la même raison que précédemment, on obtient donc $S^H_{i;j;h} < S_{i;h}$, et, par symétrie, $S^H_{i;j;h} < S_{j;h}$.

□

Nous pouvons également démontrer les deux résultats suivants :

Proposition 3.4. *Soit D une matrice de distances d'arbre de taille n, et soit un triplet $(i,j,h), 1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$.*

Si i et h sont voisins, alors :

$$S^H_{i;j;h} < S^H_{k;j;h} \quad \textit{pour tout} \quad 1 \leq k \leq n, k \neq i, k \neq j, k \neq h.$$

Si j et h sont voisins, alors :

$$S^H_{i;j;h} < S^H_{i;k;h} \quad \textit{pour tout} \quad 1 \leq k \leq n, k \neq i, k \neq j, k \neq h.$$

Démonstration. Par symétrie, il suffit de démontrer le premier résultat.

On utilise la formule 3.1 pour trouver :

$$S^H_{i;j;h} - S^H_{k;j;h} = \frac{1}{2(n-3)} \left((n-2)(D[i][h] - D[k][h]) - \Sigma_i + \Sigma_k\right)$$

$$= \frac{1}{2(n-3)} \sum_{1 \leq m \leq n; m \neq i,k} \left[D[i][h] + D[k][m] - (D[i][m] + D[h][k])\right].$$

Or, par le même argument que précédemment, si i et h sont voisins, la somme ci-dessus est négative, ce qui démontre notre résultat.

□

Proposition 3.5. *Soit D une matrice de distances d'arbre de taille n, et soit un triplet $(i,j,h), 1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$.*

Si i et j sont voisins, alors :

$$S^H_{i;j;h} > S^H_{i;h;j} \quad \text{et} \quad S^H_{i;j;h} > S^H_{h;j;i}.$$

Si h et j sont voisins, alors $S^H_{i;j;h} = S^H_{i;h;j}$.

Si h et i sont voisins, alors $S^H_{i;j;h} = S^H_{h;j;i}$.

Démonstration. Par symétrie, pour la première affirmation de la proposition, il suffit de démontrer la première inégalité.

On utilise la formule 3.1 pour trouver :

$$S^H_{i;j;h} - S^H_{i;h;j} = \frac{1}{2(n-3)} \left((n-2)(D[i][h] - D[i][j]) + \Sigma_j - \Sigma_h\right)$$

$$= \frac{1}{2(n-3)} \sum_{1 \leq k \leq n; k \neq j,h} \left[D[i][h] + D[j][k] - (D[i][j] + D[h][k])\right],$$

chacun des termes de cette somme étant positif si i et j sont voisins.

De même, par symétrie, il suffit de prouver la première égalité de ce qu'il reste à démontrer. Or, si h et j sont voisins, chacun des termes de la somme ci-dessus est nul, et donc $S^H_{i;j;h} = S^H_{i;h;j}$.

□

On va maintenant énoncer trois conjectures que nous ne prouverons pas, mais que nous avons vérifiées sur plusieurs centaines d'arbres de tailles variant de 5 à 20. La troisième conjecture est en fait une conséquence des deux premières et des résultats que nous avons démontrés.

Conjecture 3.1. *Soit D une matrice de distances d'arbre de taille n, et soit un triplet (i_0, j_0, h_0) qui minimise $S^H_{i;j;h}$, pour $1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$, alors i_0 et h_0 sont voisins, ou j_0 et h_0 sont voisins.*

Grâce à la proposition 3.5, on voit que i_0 et j_0 ne peuvent pas être voisins, car, sinon, $S^H_{i_0;h_0;j_0} < S^H_{i_0;j_0;h_0}$.

De plus, si ni i_0, ni j_0 n'est voisin de h_0, la proposition 3.4 nous montre que h_0 n'a aucun voisin. En effet, dans ce cas, si h_0 est voisin de k, alors $S^H_{i_0;k;h_0} < S^H_{i_0;j_0;h_0}$.

Cela ne démontre cependant pas notre conjecture.

Conjecture 3.2. *Soit D une matrice de distances d'arbre de taille n, et soit un triplet (i_0, j_0, h_0) qui minimise $S^T_{i;j;h}$, pour $1 \leq i < j \leq n, 1 \leq h \leq n, h \neq i, h \neq j$, alors i_0 et j_0 sont voisins.*

Conjecture 3.3. *Soit D une matrice de distances d'arbre de taille n, alors notre algorithme retrouve l'arbre phylogénétique sous-jacent.*

Nous allons expliquer comment on peut déduire cette conjecture des deux conjectures précédentes.

Tout d'abord, d'après la conjecture 3.2, la plus petite valeur $S^T_{i_T;j_T;h_T}$ correspond bien à un couple de voisins (i_T, j_T).

De plus, soit $S^H_{i_H;j_H;h_H}$ la valeur minimale des $S^H_{i;j;h}$. Alors, d'après la conjecture 3.1, soit i et h sont voisins, soit j et h sont voisins. On suppose, par exemple, que i et h sont voisins. D'après la proposition 3.2, on a $S^H_{i_H;j_H;h_H} = S^T_{h_H;j_H;i_H} \geq S^T_{i_T;j_T;h_T}$.

On va donc bien joindre les nœuds i_T et j_T, puisque le minimum des $S^H_{i;j;h}$ est supérieur ou égal à $S^T_{i_T;j_T;h_T}$.

Notons qu'en fait $S^H_{i_H;j_H;h_H} = S^T_{i_T;j_T;h_T}$. En effet, d'après la proposition 3.2, comme i_T et j_T sont voisins, $S^T_{i_T;j_T;h_T} = S^H_{i_T;h_T;j_T} \geq S^H_{i_H;j_H;h_H}$, et le minimum des $S^H_{i;j;h}$ est donc égal au minimum des $S^T_{i;j;h}$.

Comme on utilise les formules 2.5 pour calculer les distances du nœud X (union de i et de j) aux nœuds restants, on se retrouve avec un arbre additif, où on a remplacé les nœuds i et j par leur ancêtre commun X. Comme on vient de le démontrer, on va continuer à joindre deux voisins, puisque nous avons toujours affaire à un arbre phylogénétique de taille $n-1$. Ainsi de suite, on va donc bien retrouver notre arbre.

3.2 Le cas des hybrides entre voisins

On s'intéresse dans cette section aux résultats obtenus avec des réseaux comportant des phénomènes d'hybridation entre branches voisines comme dans le cas des réseaux (a) des figures 2.4 et 2.5.

3.2.1 Le cas des branches terminales

On regarde tout d'abord des arbres phylogénétiques auxquels on rajoute un phénomène d'hybridation entre deux branches voisines terminales comme dans le réseau (a) de la figure 2.4.

On analyse tout d'abord le cas $n = 4$. On considère donc le réseau de la figure 3.2.

Les distances entre espèces sont alors données par les formules suivantes :

$$D[1][3] = L_1 + L_3\,, \quad D[1][4] = L_1 + L_4\,, \quad D[3][4] = L_3 + L_4,$$

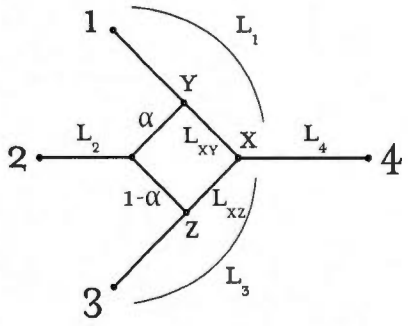


Figure 3.2 Un réseau d'hybridation de taille 4.

$$D[1][2] = L_1 - \alpha L_{XY} + (1-\alpha)L_{XZ} + L_2\,, \quad D[2][3] = L_2 + \alpha L_{XY} - (1-\alpha)L_{XZ} + L_3,$$
$$D[2][4] = L_2 + \alpha L_{XY} + (1-\alpha)L_{XZ} + L_4.$$

On sait que, dans ce cas, $S^T_{i;j;h}$ ne dépend pas de h. On a de plus

$$S^T_{1;2;3} = S^T_{1;2;4} = S^T_{3;4;1} = S^T_{3;4;2} = S_{1;2} = S_{3;4}$$
$$= \frac{1}{2}\left(D[1][2] + D[3][4]\right) + \frac{1}{4}\left(D[1][3] + D[1][4] + D[2][3] + D[2][4]\right).$$

Si on remplace les distances par les longueurs de branches, on obtient :

$$S_{1;2} = S_{3;4} = L_1 + L_2 + L_3 + L_4 + \frac{1}{2}(1-\alpha)L_{XZ}.$$

On obtient de même :

$$S_{1;3} = S_{2;4} = L_1 + L_2 + L_3 + L_4 + \frac{1}{2}\left(\alpha L_{XY} + (1-\alpha)L_{XZ}\right).$$
$$S_{1;4} = S_{2;3} = L_1 + L_2 + L_3 + L_4 + \frac{1}{2}\alpha L_{XY}.$$

On calcule maintenant toutes les valeurs de $S^H_{i;j;h}$. Elles ne dépendent pas non plus de h et on obtient :

$$S^H_{1;2;3} = S^H_{1;2;4} = S^H_{3;4;1} = S^H_{3;4;2}$$

$$= \frac{1}{2}\left(D[1][3] + D[1][4] + D[2][3] + D[2][4]\right) = L_1 + L_2 + L_3 + L_4 + \alpha L_{XY}.$$

On obtient de même :

$$S^H_{1;3;4} = S^H_{1;3;2} = S^H_{2;4;1} = S^H_{2;4;3} = L_1 + L_2 + L_3 + L_4,$$

$$S^H_{1;4;2} = S^H_{1;4;3} = S^H_{2;3;1} = S^H_{2;3;4} = L_1 + L_2 + L_3 + L_4 + (1-\alpha)L_{XZ}.$$

La valeur minimale est donc bien $S^H_{1;3;4} = S^H_{1;3;2} = S^H_{2;4;1} = S^H_{2;4;3}$, ce qui nous donne bien que 2 est l'hybride de 1 et 3. Par symétrie, pour $n = 4$, on obtient également que 4 est l'hybride de 1 et 3, que 3 est l'hybride de 2 et 4, et que 1 est l'hybride de 2 et 4. Ces phénomènes de symétrie n'existent pas pour $n > 4$.

Par exemple, pour $n = 5$, dans le réseau de la figure 3.3, on obtient que la plus petite valeur des $S^H_{i;j;h}$ est unique et est égale à :

$$S^H_{1;3;2} = L_1 + L_2 + L_3 + L_4 + L_5 + L_{XW},$$

cette dernière étant plus petite que toutes les valeurs des $S^T_{i;j;h}$. Par exemple, pour comparaison, on obtient :

$$S^T_{4;5;1} = L_1 + L_2 + L_3 + L_4 + L_5 + L_{XW} + \frac{1}{2}\alpha L_{XY},$$

$$S^T_{4;5;2} = L_1 + L_2 + L_3 + L_4 + L_5 + L_{XW} + \frac{1}{2}\left(\alpha L_{XY} + (1-\alpha)L_{XZ}\right),$$

$$S^T_{4;5;3} = L_1 + L_2 + L_3 + L_4 + L_5 + L_{XW} + \frac{1}{2}(1-\alpha)L_{XZ}.$$

Grâce à plusieurs centaines de tests effectués sur des arbres aléatoires de tailles variant de 6 à 100 auxquels on rajoutait un (ou deux) phénomènes d'hybridation entre des branches voisines terminales sur le modèle de la figure 3.1, on a pu constater que notre algorithme retrouvait systématiquement le bon arbre et le(s) bon(s) hybride(s). Pour réaliser nos tests, nous choisissons la taille de l'arbre n et construisons un arbre (i.e., une matrice de distances) aléatoire de taille n comme précédemment. Nous ajoutons ensuite deux descendants au nœud 1 (avec des longueurs de branches aléatoires) et un hybride entre ces deux descendants (dans le cas d'un seul hybride). Les longueurs de

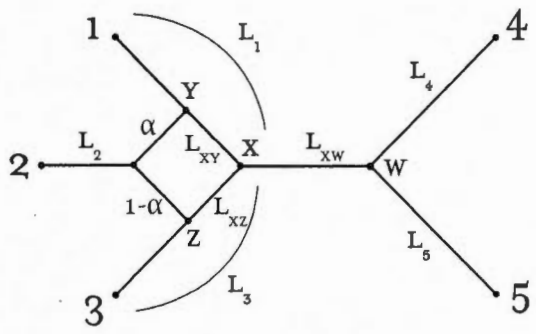


Figure 3.3 Un réseau d'hybridation de taille 5.

branches correspondant à l'hybride sont également choisies aléatoirement. Nous obtenons ainsi une matrice de distances correspondant à un arbre de taille $n+1$ avec un hybride entre deux branches terminales. En appliquant notre algorithme, on retrouve alors systématiquement le bon arbre et le bon hybride.

Nous n'avons malheureusement pas réussi à démontrer ce résultat. Une des difficultés est que l'algorithme ne trouve pas toujours l'hybride à la première itération. Pour chaque entier n de $n = 6$ à $n = 11$, nous avons mené 50 tests sur des réseaux aléatoires de tailles n comprenant un hybride entre deux branches teminales et construits comme expliqué ci-dessus. On notait alors à quelle itération l'algorithme détectait l'hybride. Les résultats sont présentés dans le tableau 3.1. On constate que pour $n > 6$, il ne semble y a avoir aucune règle pour détecter l'itération en question. Dans tous les cas, pour au moins la moitié (ou tout juste un peu moins de la moitié) des tests, l'hybride est trouvé à la première itération. Cette tendance ne se maintient cependant pas pour des réseaux plus grands. En effet, sur 25 tests menés pour $n = 50$, nous avons trouvé une moyenne de 6 itérations (nécessaires pour détecter l'hybride) avec trois tests seulement où l'hybride est trouvé à la première itération.

Taille du réseau	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$	$n = 11$
Nombre de tests où l'hybride est trouvé à la première itération	23	33	28	27	29	24
Nombre de tests où l'hybride est trouvé à la deuxième itération	27	2	12	4	9	7
Nombre de tests où l'hybride est trouvé à la troisième itération	0	15	0	8	5	4
Nombre de tests où l'hybride est trouvé à la quatrième itération	0	0	10	2	4	8
Nombre de tests où l'hybride est trouvé à la cinquième itération	0	0	0	9	0	3
Nombre de tests où l'hybride est trouvé à la sixième itération	0	0	0	0	3	0
Nombre de tests où l'hybride est trouvé à la septième itération	0	0	0	0	0	4
Nombre total de tests	50	50	50	50	50	50

Tableau 3.1 Résultats sur l'itération à laquelle on détecte l'hybride dans un réseau avec un hybride entre deux branches voisines terminales.

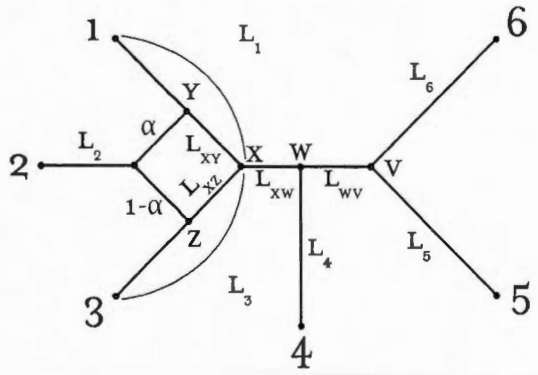


Figure 3.4 Un réseau d'hybridation de taille 6.

Nous allons maintenant expliquer le phénomène qu'on constate pour $n = 6$, à savoir que l'algorithme semble trouver l'hybride à la première itération avec une chance sur deux. Dans ce cas, nous avons un réseau similaire à celui de la figure 3.4. On ne détaillera pas les calculs mais on mentionnera simplement que la plus petite valeur des $S^H_{i;j;h}$ est égale à :

$$S^H_{1;3;2} = L_1 + L_2 + L_3 + L_4 + L_5 + L_6 + L_{XW} + \frac{4}{3}L_{WV},$$

alors que la plus petite valeur des $S^T_{i;j;h}$ est égale à :

$$S^T_{5;6;4} = L_1 + L_2 + L_3 + L_4 + L_5 + L_6 + L_{XW} + L_{WV} + \frac{1}{3}(\alpha L_{XY} + (1-\alpha)L_{XZ}).$$

On choisit donc l'hybride à la première itération si et seulement si :

$$L_{WV} < \alpha L_{XY} + (1-\alpha)L_{XZ}.$$

Or, les longueurs L_{WV}, L_{XY} et L_{XZ} suivent la même loi de probabilité, tandis que α suit une loi uniforme entre 0 et 1. On a donc bien une chance sur deux d'avoir cette inégalité. Sinon, d'après ce qu'on a dit plus haut, on trouve forcément l'hybride à l'itération suivante.

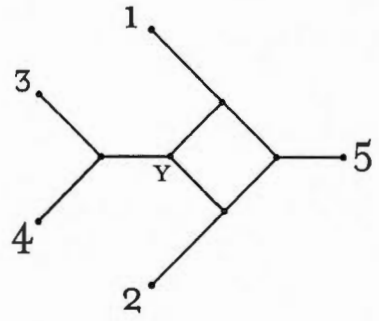


Figure 3.5 Un hybride entre deux branches non terminales.

Notons pour finir que la différence entre le minimum des $S^T_{i;j;h}$ et le minimum des $S^H_{i;j;h}$ au moment où l'hybride est trouvé diminue quand n augmente. Par exemple, pour $n = 10$, on trouve une différence moyenne de $0,29\%$ (sur 10 tests effectués), alors qu'on trouve seulement une différence moyenne de $0,02\%$ pour $n = 100$ (sur 10 tests effectués).

3.2.2 Le cas des branches non terminales

Nous analysons maintenant le cas d'hybrides entre des branches voisines non terminales. Nous prenons un arbre phylogénétique aléatoire auquel nous rajoutons le réseau de la figure 3.5. Le nœud 5 est une feuille de l'arbre aléatoire auquel nous rajoutons les nœuds 1 à 4.

Des tests menés sur plusieurs centaines de réseaux de tailles variant de 6 à 10 nous ont permis de retrouver systématiquement le bon réseau dans ce cas.

Comme dans le cas précédent, le tableau 3.2 indique pour 50 tests effectués sur des réseaux de tailles de $n = 6$ à $n = 10$ avec un hybride entre deux branches voisines non terminales à quelle itération l'algorithme a détecté l'hybride. Comme dans le cas des branches terminales, on ne détecte pas de tendance claire dans ces résultats.

Taille du réseau	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$
Nombre de tests où l'hybride est trouvé à la première itération	0	0	0	0	0
Nombre de tests où l'hybride est trouvé à la deuxième itération	50	31	20	7	20
Nombre de tests où l'hybride est trouvé à la troisième itération	0	19	2	16	2
Nombre de tests où l'hybride est trouvé à la quatrième itération	0	0	28	2	14
Nombre de tests où l'hybride est trouvé à la cinquième itération	0	0	0	25	2
Nombre de tests où l'hybride est trouvé à la sixième itération	0	0	0	0	12
Nombre total de tests	50	50	50	50	50

Tableau 3.2 Résultats sur l'itération à laquelle on détecte l'hybride dans un réseau avec un hybride entre deux branches voisines non terminales.

3.3 Le cas des hybrides entre branches non voisines

On s'intéresse dans cette section aux résultats obtenus avec des réseaux comportant des phénomènes d'hybridation entre branches non voisines comme dans le cas des réseaux (*b*) des figures 2.4 et 2.5.

On regarde uniquement des arbres phylogénétiques auxquels on rajoute un phénomène d'hybridation entre deux branches non voisines mais terminales comme dans le réseau (*b*) de la figure 2.4.

On analyse essentiellement deux cas : le premier cas où on rajoute à une extrémité d'un arbre phylogénétique aléatoire un réseau semblable à celui de la figure 3.6, et le deuxième cas où on rajoute à une extrémité d'un arbre aléatoire un réseau semblable à celui de la figure 3.7. La position et les longueurs de branches de ces deux réseaux sont choisies aléatoirement.

Dans les deux cas, les taux de détection exacte du réseau d'hybridation initial sont très bas, et ce quelle que soit la taille du réseau.

Analysons tout d'abord le premier cas, celui de la figure 3.6. Les nœuds 1 et 2 sont voisins, l'ancêtre de ces deux nœuds est voisin du nœud 4, et le nœud 3 est l'hybride de 2 et 4. L'algorithme détecte alors un ou deux hybrides. Quand il en détecte un seul, il trouve le bon. Cependant, dans pratiquement tous les tests effectués, il en trouve deux. Dans la majorité des cas, il trouve d'abord que 2 est l'hybride de 1 et 3, puis que 3 est l'hybride de 2 et 4. Sinon, il trouve d'abord que 1 est l'hybride de 2 et 3, puis que 3 est l'hybride de 2 et 4.

Dans le deuxième cas, celui de la figure 3.7, nous avons un arbre phylogénétique traditionnel, où les nœuds 1 et 2, ainsi que les nœuds 4 et 5, sont voisins. De plus, le nœud 3 est l'hybride des nœuds 2 et 4. Dans ce cas, l'algorithme trouve en général deux ou trois hybrides, et est donc incapable de trouver la bonne configuration. Le tableau 3.3 montre le nombre d'hybrides trouvés dans ce cas pour des réseaux de taille 15, sur un total de 100 tests effectués.

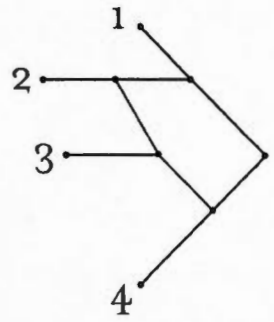


Figure 3.6 Un hybride entre deux branches non voisines.

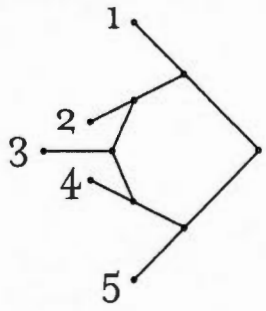


Figure 3.7 Un hybride entre deux branches non voisines.

Nombre d'hybrides	2	3	Total
Nombre de tests	29	71	100

Tableau 3.3 Nombre d'hybrides trouvés pour un hybride semblable à celui de la figure 3.7 dans un réseau de taille 15.

Taille du réseau	10	20	30	40	50	60
Nombre moyen d'hybrides	3,76	5,9	7,58	9,14	10,8	12,48

Tableau 3.4 Nombre moyen d'hybrides trouvés en fonction de la taille du réseau pour un hybride en position aléatoire.

Analysons plus en détails comment l'algortihme fonctionne. S'il trouve trois hybrides, il considère, par exemple, d'abord le nœud 4 comme l'hybride des nœuds 3 et 5, puis le nœud 2 comme l'hybride des nœuds 1 et 3, et enfin le nœud 3 comme l'hybride des nœuds 1 et 5. Quand il n'en trouve que deux, il commence avec le nœud 4 comme l'hybride des nœuds 3 et 5, puis trouve le nœud 3 comme l'hybride des nœuds 2 et 5.

Nous avons enfin analysé la situation où la position de l'hybride est aléatoire, et nous avons compté combien d'hybrides l'algorithme trouvait en fonction de la taille du réseau. Nous avons ainsi calculé le nombre moyen d'hybrides trouvés sur un total de 50 tests pour chaque taille. Les résultats sont présentés dans le tableau 3.4. On semble constater une tendance linéaire du nombre moyen d'hybrides en fonction de la taille du réseau.

On notera que, dans tous les tests effectués, quand l'algortihme détecte un hybride, au moins un des trois nœuds qu'il choisit fait partie du bon triplet d'hybridation. De plus, quand l'algorithme trouve le bon hybride, il n'en détecte plus par la suite.

Pour finir, pour des tailles de réseaux allant de $n = 6$ à $n = 11$, on a effectué 100 tests avec un hybride placé aléatoirement entre deux branches non voisines d'un arbre phylogénétique. On a alors dénombré le nombre de fois où l'algorithme ne trouvait pas l'hybride, trouvait le bon hybride mais aucun bon parent, et trouvait le bon hybride avec au moins un bon parent. Les résultats sont présentés sur la figure 3.8. Tout d'abord, le taux de non détection du bon hybride est très bas, puisqu'il varie de 0% à 4%. On constate que le pourcentage de tests où on détecte le bon hybride avec au moins un bon parent est proche de 100% pour $n = 6$ et $n = 7$, puis qu'il semble se stabiliser autour de 80% pour $n > 7$. Dans le même temps, le taux de détection du bon hybride sans

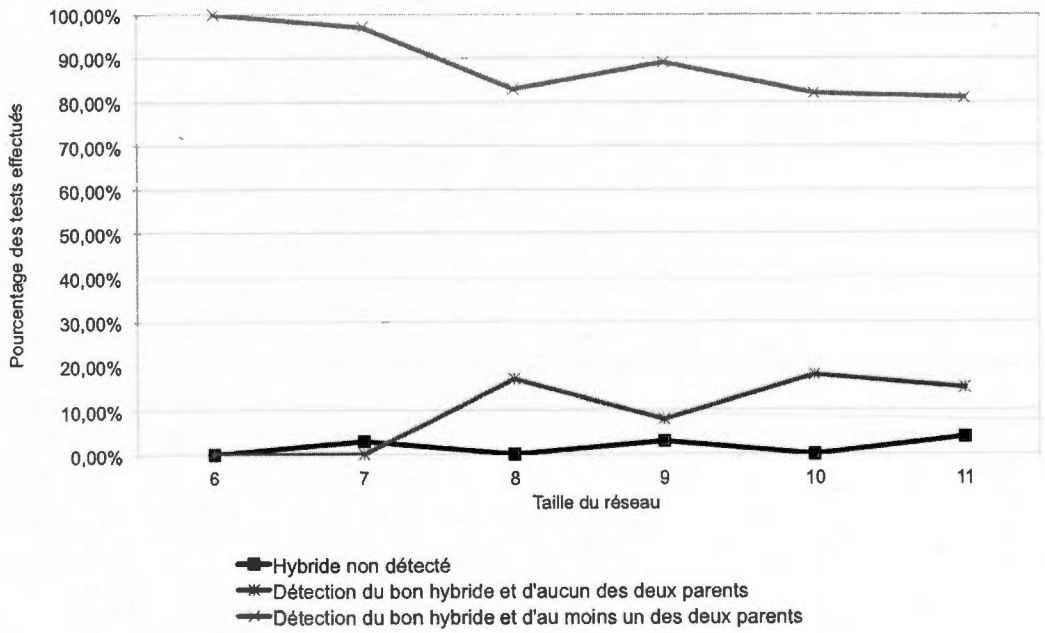


Figure 3.8 Pourcentage des tests effectués selon le résultat obtenu sur l'hybride en fonction de la taille du réseau.

aucun parent correct passe de 0% pour $n = 6$ et $n = 7$ à des valeurs allant de 8% à 18% pour $n > 7$. On rappellera que, pour des hybrides entre voisins, l'algorithme retrouve systématiquement le bon hybride et les deux bons parents.

CONCLUSION

Nous avons développé un nouvel algorithme pour inférer des réseaux phylogénétiques qui comportent des phénomènes d'hybridation en se basant sur le principe de la méthode Neighbor Joining (Saitou et Nei, 1987).

Notre algorithme retrouve tous les arbres phylogénétiques traditionnels (sans hybridation), et est capable de retrouver tous les hybrides dont les espèces parentes sont voisines. Nos tests ont en tout cas été concluants quand on rajoute un ou deux hybrides en position terminale, ou un hybride entre deux branches voisines non terminales.

Dans le cas des hybrides entre branches non voisines, l'algorithme ne retrouve pas toujours le bon hybride. Souvent, il trouve en fait trop d'hybrides, car il ne détecte pas du premier coup le bon hybride, et continue à trouver des hybrides tant qu'il n'a pas détecté le bon. Les taux de détection du bon hybride sont cependant proches de 100%, mais l'algorithme ne trouve pas toujours les bons parents de l'hybride quand il le détecte.

Pour remédier à ce problème, nous avons essayé de considérer d'autres configurations comme celles de la figure 3.9. La première configuration (figure 3.9 (a)) n'apporte rien puisque la somme des longueurs de branches est, dans ce cas, la même que pour la configuration que nous avons utilisée. Nous avons donc utilisé la configuration de la figure 3.9 (b). Notons tout d'abord qu'il faut alors considérer des quadruplets, puisqu'une des branches de l'hybride a un seul voisin, alors que l'autre branche est voisine de tous les nœuds restants. L'ajout de cette configuration n'a cependant pas été concluant puisque l'algorithme ne retrouvait alors plus tous les arbres, ni tous les hybrides entre branches voisines, même si ses performances pour les hybrides entre branches non voisines étaient meilleures. On a alors tenté de rajouter des configurations d'arbres qui considéraient des quadruplets de nœuds. Nous n'avons cependant pas obtenu de résultats concluants dans

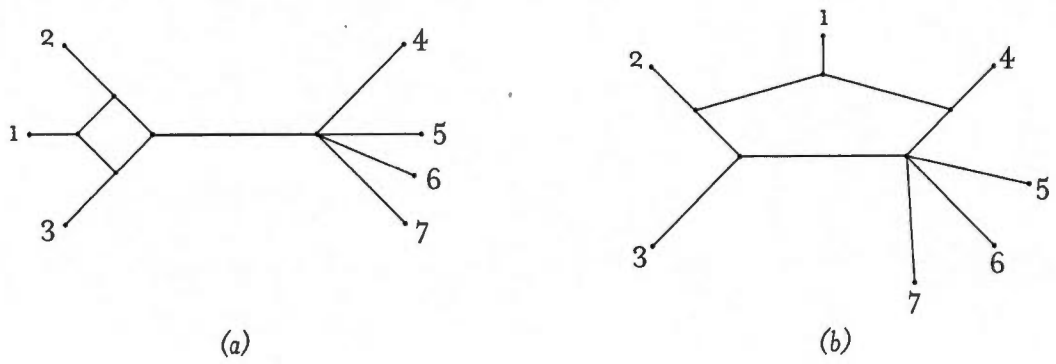


Figure 3.9 Autres configurations étudiées.

ce cas.

Une autre possibilité pour résoudre ce problème serait d'analyser les suites d'hybrides qu'on obtient quand l'algorithme se trompe. On sait déjà qu'il y a toujours un bon nœud dans ces triplets, et que, quand l'algorithme trouve le bon hybride, il n'en détecte plus par la suite. On pourrait ainsi ajouter un module qui détecte les suites de ce type, qui trouve les nœuds qui reviennent systématiquement dans cette suite, puis qui en déduit le bon hybride avec ses bons parents. Cela est cependant plus compliqué à mettre au point sur le plan algorithmique.

Nous avons enfin vérifié la positivité des longueurs de branches des configurations d'hybrides. En effet, la présence de longueurs de branches négatives pourrait introduire un biais et favoriser certains hybrides, ce qui expliquerait le trop grand nombre d'hybrides trouvé en général par notre algorithme. Nous avons ainsi fait une centaine de tests pour des réseaux de taille 5. Dans la grande majorité des cas, les mauvais hybrides ont toutes leurs longueurs de branches positives. Nous avons trouvé seulement quelques cas avec la longueur de branche de l'hybride légèrement négative.

APPENDICE A

CODE EN C++

On construit une matrice de distances d'arbre aléatoire, on ajoute un hybride entre voisins, et on applique notre algorithme à la matrice de distances ainsi construite. Notons que le code doit être très légèrement modifié si on veut ajouter un hybride entre des branches non voisines. Nous n'indiquerons pas ces modifications dans cet annexe.

```
/*
 * File:   main.cpp
 * Created on 28 janvier 2011, 13:01
 */
#include <stdlib.h> #include <stdio.h> #include <math.h>
#include <time.h> #include <string.h> #include <vector>
#include<iostream> #include<fstream>
using namespace std;
int floor1(double x);
void tree_generation(double **DA, double **DI, int n, double Sigma);
#define INFINI 999999.99 #define ECRAN "ECRAN"
#define FICHIER "FICHIER" #define MaxRF 0
double MaxLong=0; int n;
double MinLong = INFINI; double seuil; double epsilon = 0.00005;
/*
 */
int main()
{
```

```
time_t t; srand((unsigned) time(&t)); int i;
/* n=Taille de l'arbre */
n=3;
double **DAA,**DI; double *LONGUEUR, Sigma;
DI = (double **) malloc((n+1)*sizeof(double*));
DAA=(double **) malloc((2*n-1)*sizeof(double*));
for (i=0;i<=n;i++)
{
    DI[i]=(double*)malloc((n+1)*sizeof(double));

    if (DI[i]==NULL)
    {
        printf("_Data_matrix_is_too_large(2)\n");
        exit(1);
    }
}
for(i=0;i<=2*n-2;i++)
{
    DAA[i]=(double*)malloc((2*n-1)*sizeof(double));
}
/* Arbre aleatoire de taille n (voir fonction plus bas) */
tree_generation(DAA, DI, n, Sigma = 0.0);
double **D,**DA,**H,**C,*T1,*S,*LP,Som,Smin,SminH,Sijh,Sihj,Sij;
double SminTest,L,Lii,Ljj,L1,L2,Lnm2,Lnm1,Lhy,U,LONH[5];
double Lhh,LhhH,alpha,beta,gamma,l1,l2,l3;
int *T,j,h,k,ii,jj,hh,hhT,iiH,jjH,hhH,iT,jT,n1,nA;
/* Longueurs de branche pour l'hybride */
U = 0.1;
for (i=0;i<=4;i++)
    LONH[i] = -1.0/(2*n-3)*log(U);
i=0;
while (i<=4)
{   U=1.0*rand()/RAND_MAX; LONH[i]=1.0*LONH[i]*(1.0+0.8*(-log(U)));
```

```
    LONH[i] = LONH[i]*2;
    if (LONH[i]>2*epsilon) i++;
}
L1=2.0*LONH[0]; L2=2.0*LONH[1]; Lnm2=LONH[2]; Lnm1=LONH[3];
Lhy=LONH[4]; alpha=1.0*rand()/RAND_MAX;
beta=1.0*rand()/RAND_MAX; gamma=1.0*rand()/RAND_MAX;

n=n+3; nA=n;
D=(double **) malloc((n+1)*sizeof(double*));
DA=(double **) malloc((n+1)*sizeof(double*));
T1=(double *) malloc((n+1)*sizeof(double));
S=(double *) malloc((n+1)*sizeof(double));
LP=(double *) malloc((n+1)*sizeof(double));
T=(int *) malloc((n+1)*sizeof(int));
for (i=0;i<=n;i++)
{
    D[i]=(double*)malloc((n+4)*sizeof(double));
    if (D[i]==NULL)
    {
        printf("probleme_de_memoire"); exit(1);
    }
}
for (i=0;i<=n;i++)
{
    DA[i]=(double*)malloc((n+1)*sizeof(double));
    if (DA[i]==NULL)
    {
        printf("probleme_de_memoire"); exit(1);
    }
}
for (i=1;i<=n-3;i++)
{
    for (j=1;j<=n-3;j++)
```

```
    {
        D[i][j]=DI[i][j];
    }
}
for (i=1;i<=n;i++)
{
    for (j=1;j<=n;j++)
    {
        DA[i][j]=0;
    }
}
/*/* Ajout de l'hybride */*/
/*  Ajout du premier noeud 1-(n-2) */
for (j=2;j<=n-3;j++)
{
    D[1][j]=D[1][j]+L1; D[j][1]=D[1][j];
}
for (j=2;j<=n-3;j++)
{
    D[n-2][j]=D[1][j]+Lnm2-L1; D[j][n-2]=D[n-2][j];
}
D[1][n-2]=L1+Lnm2; D[n-2][1]=D[1][n-2];
/*  Ajout du deuxieme noeud 2-(n-1) */
for (j=1;j<=n-2;j++)
{
    if(j!=2)
    {
        D[2][j]=D[2][j]+L2; D[j][2]=D[2][j];
    }
}
for (j=1;j<=n-2;j++)
{
    if(j!=2)
```

```
    {
        D[n-1][j]=D[2][j]+Lnm1-L2; D[j][n-1]=D[n-1][j];
    }
}
D[2][n-1]=L2+Lnm1; D[n-1][2]=D[2][n-1];
/*Ajout de l'hybride entre 1 et n-2 (entre voisins)*/
for (j=2;j<=n-1;j++)
{
    if(j!=(n-2))
    {
      D[n][j]=Lhy+alpha*(D[1][j]-beta*L1);
      D[n][j]=D[n][j]+(1-alpha)*(D[n-2][j]-gamma*Lnm2);
      D[j][n]=D[n][j];
    }
}
D[n][1]=Lhy+alpha*beta*L1+(1-alpha)*(D[n-2][1]-gamma*Lnm2);
D[n][n-2]=Lhy+alpha*(D[1][n-2]-beta*L1)+(1-alpha)*gamma*Lnm2;
D[1][n]=D[n][1]; D[n-2][n]=D[n][n-2];
/*Impression de la matrice (facultatif)*/
printf("\n"); printf("%d", n); printf("\n");
for(i=1;i<=n;i++)
{
    printf("s%d ", i);
    for(j=1;j<=n;j++)
    {
        printf("%f ", D[i][j]);
    }
printf("\n");
}
L=0; Som=0;
for (i=1;i<=n;i++)
{
    S[i]=0; LP[i]=0;
```

```
    for (j=1;j<=n;j++)
    {
        S[i]=S[i]+D[i][j];
    }
    Som=Som+S[i]/2.0; T[i]=i; T1[i]=0;
}
    /* Procedure principale*/
n1=n;
while (n1>3)
{
    /*Recherche du meilleur triplet (i,h,j) pour hybridation*/
    SminH=2.0*Som;
    for (i=1;i<=(n1-1);i++)
    {
        for (j=i+1;j<=n1;j++)
        {
            for (h=1;h<=n1;h++)
            {
                if ((h!=i)&&(h!=j))
                {
                    /*Formule 2.19 */
                    Sihj=(n1-2)*(D[i][h]+D[j][h])+2.0*Som;
                    Sihj=Sihj-(S[i]+S[j])-2*S[h];
                    Sihj=Sihj/(n1-3.0)/2;
                    Lhh=0.5*(D[i][h]+D[h][j]-D[i][j]);
                    if ((Sihj<SminH)&&(Lhh>0))
                    {
                        SminH=Sihj; LhhH=Lhh;
                        iiH=i; jjH=j; hhH=h;
                    }
                }
            }
        }
```

```
}
/* Recherche du meilleur triplet (i,j,h) pour l'arbre*/
Smin=4.0*Som;
for (i=1;i<=(n1-1);i++)
{
    for (j=i+1;j<=n1;j++)
    {
        for (h=1;h<=n1;h++)
        {
            if ((h!=i)&&(h!=j))
            {
                /*Formule 2.33 */
                Sijh=(n1-2)*(D[i][h]+D[j][h]+2*D[i][j])+4*Som;
                Sihj=Sijh-3*(S[i]+S[j])-2*S[h];
                Sijh=Sijh/4.0/(n1-3.0);
                if (Sijh<Smin)
                {
                    Smin=Sijh; iT=i; jT=j;
                }
            }
        }
    }
}
/* Si on choisit un hybride*/
if (SminH<Smin-0.00001)
{
    /*hhH est l'hybride de iiH et jjH*/
    cerr<<"Hybride"<< endl;
    cerr<<"SminH="<<SminH<<endl<<"SminT="<<Smin<<endl;
    cerr<<"n1="<<n1<< endl<<"h="<<hhH<<endl;
    cerr<<"i="<<iiH<< endl<<"j="<<jjH<<endl;
    /* Mise a jour de D */
    Som=Som-S[hhH];
```

```
        for (i=1;i<=n1;i++)
        {
            if ((i!=hhH))
            {
                S[i]=S[i]-D[i][hhH];
            }
        }
        if (hhH!=n1)
        {
            for (i=1;i<=(n1-1);i++)
            {
                D[i][hhH]=D[i][n1]; D[hhH][i]=D[n1][i];
            }
            D[hhH][hhH]=0; S[hhH]=S[n1];LP[hhH]=LP[n1];
        }
        /* Mise a jour de DA */
        if (hhH!=nA)
        {
            for (i=1;i<=(nA-1);i++)
            {
                DA[i][hhH]=DA[i][nA]; DA[hhH][i]=DA[nA][i];
            }
            DA[hhH][hhH]=0;
        }
        nA--; n1--;
    }
    /* Si on choisit de joindre deux noeuds */
    else
    {
        /*iT et jT sont voisins*/
        cerr<<"Arbre"<<endl<<"n1="<<n1<< endl;
        cerr<<"i="<<iT<<endl<< "j="<<jT<< endl;
        cerr<<"SminH="<<SminH<< endl<<"SminT="<<Smin<<endl;
```

```
/* Reunion de iT et jT */
ii=iT; jj=jT;
Lii=(D[ii][jj]+(S[ii]-S[jj])/(n1-2))/2-LP[ii];
Ljj=(D[ii][jj]+(S[jj]-S[ii])/(n1-2))/2-LP[jj];
/* Mise a jour de D */
if (Lii<0.00001) Lii=0.00005; if (Ljj<0.00001) Ljj=0.00005;
L=L+Lii+Ljj; LP[ii]=0.5*D[ii][jj];
Som=Som-(S[ii]+S[jj])/2;
for (i=1;i<=n1;i++)
{
    if ((i!=ii)&&(i!=jj))
    {
        S[i]=S[i]-0.5*(D[i][ii]+D[i][jj]);
        D[i][ii]=(D[i][ii]+D[i][jj])/2; D[ii][i]=D[i][ii];
    }
}
D[ii][ii]=0; S[ii]=0.5*(S[ii]+S[jj])-D[ii][jj];
if (jj!=n1)
{
    for (i=1;i<=(n1-1);i++)
    {
        D[i][jj]=D[i][n1]; D[jj][i]=D[n1][i];
    }
    D[jj][jj]=0; S[jj]=S[n1]; LP[jj]=LP[n1];
}
/* Mise a jour de DA */
for (i=1;i<=n;i++)
{
    if (T[i]==ii) T1[i]=T1[i]+Lii;
    if (T[i]==jj) T1[i]=T1[i]+Ljj;
}
for (j=1;j<=n;j++)
{
```

```
            if (T[j]==jj)
            {
                for (i=1;i<=n;i++)
                {
                    if (T[i]==ii)
                    {
                        DA[i][j]=T1[i]+T1[j]; DA[j][i]=DA[i][j];
                    }
                }
            }
        }
        for (j=1;j<=n;j++)
            if (T[j]==jj)  T[j]=ii;
        if (jj!=n1)
        {
            for (j=1;j<=n;j++)
                if (T[j]==n1) T[j]=jj;
        }
        n1--;
    }
}
/*On joint les trois noeuds restants */
/*On calcule les longueurs des branches restantes */
l1=(D[1][2]+D[1][3]-D[2][3])/2-LP[1];
l2=(D[1][2]+D[2][3]-D[1][3])/2-LP[2];
l3=(D[1][3]+D[2][3]-D[1][2])/2-LP[3];
if (l1<0.00001) l1=0.00005; if (l2<0.00001) l2=0.00005;
if (l3<0.00001) l3=0.00005; L=L+l1+l2+l3;
for (j=1;j<=n;j++)
{
    for (i=1;i<=n;i++)
    {
        if ((T[j]==1)&&(T[i]==2))
```

```
        {
            DA[i][j]=T1[i]+T1[j]+l1+l2; DA[j][i]=DA[i][j];
        }
        if ((T[j]==1)&&(T[i]==3))
        {
            DA[i][j]=T1[i]+T1[j]+l1+l3; DA[j][i]=DA[i][j];
        }
        if ((T[j]==2)&&(T[i]==3))
        {
            DA[i][j]=T1[i]+T1[j]+l2+l3; DA[j][i]=DA[i][j];
        }
    }
    DA[j][j]=0;
}
free(T); free(T1); free(S); free(LP);
for (i=0;i<=n;i++)
{
    free(D[i]); free(DA[i]);
}
for(i=0;i<=2*n-8;i++)
{
    free(DAA[i]);
}
for (i=0;i<=n-3;i++)
{
    free(DI[i]);
}
free(DA); free(D);free(DI);free(DAA); return 0;
}
/* Fonction qui genere une matrice de distance d'arbre aleatoire */
void tree_generation(double **DAA, double **DI, int n, double Sigma)
{
    struct TABLEAU { int V; } **NUM, **A;
```

```
int i,j,k,p,a,a1,a2,*L,*L1,n1; double *LON,X0,X,U;
n1=n*(n-1)/2;
L=(int *)malloc((2*n-2)*sizeof(int));
L1=(int *)malloc((2*n-2)*sizeof(int));
LON=(double *)malloc((2*n-2)*sizeof(double));
NUM=(TABLEAU **)malloc((2*n-2)*sizeof(TABLEAU*));
A=(TABLEAU **)malloc((n1+1)*sizeof(TABLEAU*));
for (i=0;i<=n1;i++)
{
    A[i]=(TABLEAU*)malloc((2*n-2)*sizeof(TABLEAU));
    if (i<=2*n-3) NUM[i]=(TABLEAU*)malloc((n+1)*sizeof(TABLEAU));

    if ((A[i]==NULL)||((i<=2*n-3)&&(NUM[i]==NULL)))
    {
        printf("\nData matrix is too large\n"); exit(1);
    }
}
    /* Topologie de l'arbre additif aleatoire T */
for (j=1;j<=2*n-3;j++)
{
    for (i=1;i<=n;i++)
    {
        A[i][j].V=0; NUM[j][i].V=0;
    }
    for (i=n+1;i<=n1;i++)
        A[i][j].V=0;
}
A[1][1].V=1; L[1]=1; L1[1]=2; NUM[1][1].V=1; NUM[1][2].V=0;
for (k=2;k<=n-1;k++)
{
    p=(rand() % (2*k-3))+1;
    for (i=1;i<=(n*(k-2)-(k-1)*(k-2)/2+1);i++)
        A[i][2*k-2].V=A[i][p].V;
```

```
for (i=1;i<=k;i++)
{
    a=n*(i-1)-i*(i-1)/2+k+1-i;
    if (NUM[p][i].V==0)
        A[a][2*k-2].V=1;
    else
        A[a][p].V=1;
}
for (i=1;i<=k;i++)
{
    a=n*(i-1)-i*(i-1)/2+k+1-i; A[a][2*k-1].V=1;
}
for (j=1;j<=k;j++)
{
    if (j==L[p])
    {
        for (i=1;i<=2*k-3;i++)
        {
            if (i!=p)
            {
                if (L1[p]>L[p])
                    a=floor1((n-0.5*L[p])*(L[p]-1)+L1[p]-L[p]);
                else
                    a=floor1((n-0.5*L1[p])*(L1[p]-1)+L[p]-L1[p]);
                if (A[a][i].V==1)
                {
                    if (NUM[i][L[p]].V==0)
                        a=floor1((n-0.5*L[p])*(L[p]-1)+k+1-L[p]);
                    else
                        a=floor1((n-0.5*L1[p])*(L1[p]-1)+k+1-L1[p]);
                        A[a][i].V=1;
                }
            }
```

```
            }
        }
        else if (j!=L1[p])
        {
            a=floor1((n-0.5*j)*(j-1)+k+1-j);
            if (j<L[p])
            a1=floor1((n-0.5*j)*(j-1)+L[p]-j);
            else
                a1=floor1((n-0.5*L[p])*(L[p]-1)+j-L[p]);
            if (j<L1[p])
              ba2=floor1((n-0.5*j)*(j-1)+L1[p]-j);
            else
              ba2=floor1((n-0.5*L1[p])*(L1[p]-1)+j-L1[p]);
            for (i=1;i<=2*k-3;i++)
            {
                    if ((i!=p)&&((A[a1][i].V+A[a2][i].V==2)||
                    ((NUM[i][j].V+NUM[i][L[p]].V==0)
                    &&(A[a2][i].V==1))||((NUM[i][j].V+
                    NUM[i][L1[p]].V==0)&&(A[a1][i].V==1))))
                            A[a][i].V=1;
            }
        }
    }
    for (i=1;i<=k;i++)
        NUM[2*k-2][i].V=NUM[p][i].V;
    NUM[2*k-2][k+1].V=1;
    for (i=1;i<=k;i++)
        NUM[2*k-1][i].V=1;
    for (i=1;i<=2*k-3;i++)
    {
        if(((NUM[i][L[p]].V+NUM[i][L1[p]].V)!=0)&&(i!=p))
            NUM[i][k+1].V=1;
    }
```

```
    L[2*k-2]=k+1; L1[2*k-2]=L1[p];
    L[2*k-1]=L1[p]; L1[2*k-1]=k+1; L1[p]=k+1;
}
    /* Calcul des longueurs de branches a partir d'une
    distribution exponentielle d'esperance 1/(2n-3) */
U = 0.1;
for (i=1;i<=2*n-3;i++)
    LON[i] = -1.0/(2*n-3)*log(U);
i=1;
while (i<=2*n-3)
{
    U = 1.0*rand()/RAND_MAX;
    LON[i] = 1.0*LON[i]*(1.0+0.8*(-log(U))); LON[i] = LON[i]*2;
    if (LON[i]>2*epsilon) i++;}
    /* Calcul d'une matrice de distance d'arbre */
for (i=1;i<=n;i++)
{
    DAA[i][i]=0;
    for (j=i+1;j<=n;j++)
    {
        DAA[i][j]=0;
        a=floor1((n-0.5*i)*(i-1)+j-i);
        for (k=1;k<=2*n-3;k++)
            if (A[a][k].V==1)
            {
                DAA[i][j]=DAA[i][j]+LON[k]; DAA[j][i]=DAA[i][j];
            }
    }
}
    /* Ajout d'un bruit a la matrice d'arbre */
for (i=1;i<=n;i++)
{
    DI[i][i]=0.0;
```

```
        for (j=i+1;j<=n;j++)
        {
            X=0.0;
            for (k=1;k<=5;k++)
            {
                X0 = 1.0*rand()/RAND_MAX; X=X+0.0001*X0;
            }
            X=2*sqrt(0.6)*(X-2.5); U=X-0.01*(3*X-X*X*X);
            DI[i][j]=DAA[i][j]+Sigma*U;
            if (DI[i][j]<0)
            {
                DI[i][j]=0.01; DI[j][i]=DI[i][j];
            }
        }
    }
    free (L); free(L1); free(LON);
    for (i=0;i<=n1;i++)
    {
        free(A[i]);
        if (i<=2*n-3) free(NUM[i]);
    }
    free(NUM); free(A);
}
/*Fonction qui arrondit un reel x a l'entier le plus proche en
prenant la valeur par defaut si x est egal a un entier plus un demi */
int floor1(double x)
{
    int i;
    if (ceil(x)-floor(x)==2) i=(int)x;
    else if (fabs(x-floor(x)) > fabs(x-ceil(x))) i=(int)ceil(x);
    else i=(int)floor(x);
    return i;
}
```

BIBLIOGRAPHIE

Albrecht, B., C. Scornavacca, A. Cenci, et D. H. Huson. 2012. « Fast computation of minimum hybridization networks », *Bioinformatics*, vol. 28, no. 2, p. 191–197.

Arnold, M. L. 1997. *Natural Hybridization and Evolution*. New York, NY, États-Unis : Oxford University Press.

Atteson, K. 1999. « The Performance of Neighbor-Joining Methods of Phylogenetic Reconstruction », *Algorithmica*, vol. 25, no. 2-3, p. 251–278.

Bandelt, H.-J. et A. W. Dress. 1989. « Weak hierarchies associated with similarity measures–an additive clustering technique. », *Bulletin of Mathematical Biology*, vol. 51, no. 1, p. 133–166.

Bandelt, H.-J. et A. W. M. Dress. 1992a. « A canonical decomposition theory for metrics on a finite set », *Advances Math*, vol. 92, p. 47–105.

Bandelt, H.-J. et A. W. M. Dress. 1992b. « Split decomposition : A new and useful approach to phylogenetic analysis of distance data », *Molecular Phylogenetics and Evolution*, vol. 1, no. 3, p. 242–252.

Bandelt, H.-J., P. Forster, et A. Röhl. 1999. « Median-joining networks for inferring intraspecific phylogenies », *Molecular Biology and Evolution*, vol. 16, no. 1, p. 37–48.

Barthélemy, J.-P. et A. Guénoche. 1991. *Trees and proximity representations*. Coll. « Wiley-Interscience series in discrete mathematics and optimization ». Hoboken, NJ, États-Unis : John Wiley & Sons.

Bauman, R. W. 2007. *Microbiology : With Diseases by Taxonomy*. San Francisco, CA, États-Unis : Pearson/Benjamin Cummings.

Boc, A. et V. Makarenkov. 2012. « Towards an accurate identification of mosaic genes and partial horizontal gene transfers », *Nucleic Acids Research*. À paraître.

Boc, A., H. Philippe, et V. Makarenkov. 2010. « Inferring and Validating Horizontal Gene Transfer Events Using Bipartition Dissimilarity », *Systematic Biology*, vol. 59, no. 2, p. 195–211.

Bryant, D. et V. Moulton. 2004. « Neighbor-net : an agglomerative method for the construction of phylogenetic networks », *Molecular Biology and Evolution*, vol. 21, no. 2, p. 255–265.

Darwin, C. 1859. *On the Origin of Species by Means of Natural Selection.* Londres, Grande-Bretagne : John Murray.

Dawley, R. M. 1989. *An introduction to unisexual vertebrates.* T. Bulletin 4, p. 1–18. Albany, NY, États-Unis : New York State Museum.

Diday, E. et P. Bertrand. 1984. « An extension of hierarchical clustering : the pyramidal representation ». In *Pattern Recognition in Practice*, p. 411–424, Amsterdam, Pays-Bas. Elsevier.

Doolittle, W. F. 1999. « Phylogenetic classification and the universal tree », *Science*, vol. 284, no. 5423, p. 2124–2128.

Doolittle, W. F., Y. Boucher, C. L. Nesbø, C. J. Douady, J. O. Andersson, et A. J. Roger. 2003. « How big is the iceberg of which organellar genes in nuclear genomes are but the tip ? », *Philosophical Transactions of the Royal Society of London - Series B : Biological Sciences*, vol. 358, no. 1429, p. 39–58.

Doyon, J.-P., C. Scornavacca, K. Y. Gorbunov, G. J. Szöllosi, V. Ranwez, et V. Berry. 2010. « An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers ». In Tannier, E., éditeur, *Comparative Genomic.* T. 6398, série *Lecture Notes in Computer Science*, p. 93–108, Berlin Heidelberg, Allemagne. Springer.

Excoffer, L. et P. Smouse. 1994. « Using allele frequencies and geographic subdivision to reconstruct gene trees within a species : molecular variance parsimony », *Genetics Society of America*, vol. 136, p. 343–359.

Felsenstein, J. 1981. « Evolutionary trees from DNA sequences : a maximum likelihood approach », *Journal of Molecular Evolution*, vol. 17, no. 6, p. 368–376.

Felsenstein, J. 2005. PHYLIP (Phylogeny Inference Package). Version 3.6. Distributed by the author.

Fitch, W. M. 1971. « Toward defining the course of evolution : minimum change for a specific tree topology », *Systematic Zoology*, vol. 20, no. 4, p. 406–416.

Fitch, W. M. 1997. « Networks and viral evolution », *Journal of Molecular Evolution*, vol. 44, no. Suppl. 1, p. S65–S75.

Foulds, L. R., M. D. Hendy, et D. Penny. 1979. « A graph theoretic approach to the development of minimal phylogenetic trees », *Journal of Molecular Evolution*, vol. 13, no. 2, p. 127–49.

Gambette, P. et D. H. Huson. 2008. « Improved layout of phylogenetic networks », *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 5, no. 3, p. 472–479.

Gascuel, O. 1997. « BIONJ : an improved version of the NJ algorithm based on a simple model of sequence data », *Molecular Biology and Evolution*, vol. 14, p. 685–695.

Gogarten, J. P., W. F. Doolittle, et J. G. Lawrence. 2002. « Prokaryotic evolution in light of gene transfer. », *Molecular Biology and Evolution*, vol. 19, no. 12, p. 2226–2238.

Guindon, S., J.-F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, et O. Gascuel. 2010. « New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies : Assessing the Performance of PhyML 3.0 », *Systematic Biology*, vol. 59, no. 3, p. 307–321.

Guindon, S. et O. Gascuel. 2003. « A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood », *Systematic Biology*, vol. 52, no. 5, p. 696–704.

Hallett, M. T. et J. Lagergren. 2001. « Efficient algorithms for lateral gene transfer problems ». In *RECOMB '01 : Proceedings of the fifth annual international conference on Computational biology*, p. 149–156, New York, NY, États-Unis. ACM.

Hein, J. 1993. « A heuristic method to reconstruct the history of sequences subject to recombination », *Journal of Molecular Evolution*, vol. 36, p. 396–406.

Huelsenbeck, J. P. et F. Ronquist. 2001. « MRBAYES : Bayesian inference of phylogenetic trees. », *Bioinformatics*, vol. 17, no. 8, p. 754–755.

Huson, D. H. et D. Bryant. 2006. « Application of phylogenetic networks in evolutionary studies », *Molecular Biology and Evolution*, vol. 23, no. 2, p. 254–267.

Huson, D. H. et R. Rupp. 2008. « Summarizing multiple gene trees using cluster networks ». In Crandall, K. A. et J. Lagergren, éditeurs, *Algorithms in Bioinformatics*. T. 5251, série *Lecture Notes in Computer Science*, p. 296–305, Berlin Heidelberg, Allemagne. Springer.

Huson, D. H., R. Rupp, et C. Scornavacca. 2010. *Phylogenetic Networks : Concepts, Algorithms and Applications*. Coll. « Phylogenetic Networks : Concepts, Algorithms and Applications ». Cambridge, Grande Bretagne : Cambridge University Press.

Jain, R., M. C. Rivera, et J. A. Lake. 1999. « Horizontal gene transfer among genomes : the complexity hypothesis. », *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, no. 7, p. 3801–3806.

Judd, W. S. 2008. *Plant systematics : a phylogenetic approach*. Sunderland, MA, États-Unis : Sinauer Associates.

Jukes, T. H. et C. R. Cantor. 1969. *Evolution of Protein Molecules*. New York, NY, États-Unis : Academy Press.

Kimura, M. 1980. « A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. », *Journal of Molecular Evolution*, vol. 16, no. 2, p. 111–120.

Koonin, E. V. 2003. « Horizontal gene transfer : the path to maturity », *Molecular Microbiology*, vol. 50, no. 3, p. 725–727.

Lapointe, F.-J. 2000. « How to account for reticulation events in phylogenetic analysis : A comparison of distance-based methods », *Journal of Classification*, vol. 17, no. 2, p. 175–184.

Legendre, P. 2000a. « Biological applications of reticulate analysis », *Journal of Clinical Laboratory Analysis*, vol. 17, p. 191–195.

Legendre, P. 2000b. « Special section on reticulate evolution », *Journal of Classification*, no. 17, p. 153–195.

Legendre, P. et V. Makarenkov. 2002. « Reconstruction of Biogeographic and Evolutionary Networks Using Reticulograms », *Systematic Biology*, vol. 51, no. 2, p. 199–216.

Makarenkov, V. 2001. « T-REX : reconstructing and visualizing phylogenetic trees and reticulation networks », *Bioinformatics*, vol. 17, no. 7, p. 664–668.

Makarenkov, V., A. Boc, C. Delwiche, A. B. Diallo, et H. Philippe. 2006. « New efficient algorithm for modeling partial and complete gene transfer scenarios ». In *Data Science and Classification*. Coll. « Studies in Classification, Data Analysis, and Knowledge Organization », p. 341–349, Berlin Heidelberg, Allemagne. Springer.

Makarenkov, V., D. Kevorkov, et P. Legendre. 2006. *Phylogenetic network construction approaches*. Coll. Arora, D. K., R. M. Berka, et G. B. Singh, éditeurs, Coll. « *Bioinformatics* ». T. 6, série *Applied Mycology and Biotechnology*, p. 61 – 97. Amsterdam, Pays-Bas : Elsevier.

Makarenkov, V. et P. Legendre. 2004. « From a phylogenetic tree to a reticulated network », *Journal of Computational Biology*, vol. 11, no. 1, p. 195–212.

Posada, D. et K. A. Crandall. 2001. « Evaluation of methods for detecting recombination from DNA sequences : Computer simulations », *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 24, p. 13757–13762.

Rannala, B. et Z. Yang. 1996. « Probability distribution of molecular evolutionary trees : A new method of phylogenetic inference », *Journal of Molecular Evolution*, vol. 43, no. 3, p. 304–311.

Rieseberg, L. H. et N. C. Ellstrand. 1993. « What can molecular and morphological markers tell us about plant hybridization ? », *Critical Reviews in Plant Sciences*, vol. 12, no. 3, p. 213–241.

Rieseberg, L. H. et J. D. Morefield. 1995. *Character expression, phylogenetic reconstruction, and the detection of reticulate evolution*, p. 333–353. Saint-Louis, MO, États-Unis : Monographs in Systematic Botany at the Missouri Botanical Garden.

Saitou, N. et M. Nei. 1987. « The neighbor-joining method : a new method for reconstructing phylogenetic trees », *Molecular Biology and Evolution*, vol. 4, no. 4, p. 406–425.

Sawyer, S. 1989. « Statistical tests for detecting gene conversion. », *Molecular Biology and Evolution*, vol. 6, no. 5, p. 526–538.

Smouse, P. E. 2000. « Reticulation inside the species boundary », *Journal of Clinical Laboratory Analysis*, vol. 17, p. 165–173.

Sneath, P. H. A., M. J. Sackin, et R. P. Ambler. 1975. « Detecting Evolutionary Incompatibilities From Protein Sequences », *Systematic Biology*, vol. 24, no. 3, p. 311–332.

Sneath, P. H. A. et R. R. Sokal. 1973. *Numerical Taxonomy. The Principles and Practice of Numerical Classification.* San Francisco, CA, États-Unis : W. H. Freeman.

Sonea, S. et M. Panisset. 1983. *A new bacteriology.* Burlington, MA, États-Unis : Jones and Bartlett.

Stace, C. A. 1991. *Plant Taxonomy and Biosystematics.* Cambridge, Grande-Bretagne : Cambridge University Press.

Stamatakis, A., A. J. Aberer, C. Goll, S. A. Smith, S. A. Berger, et F. Izquierdo-Carrasco. 2012. « RAxML-Light : A Tool for computing TeraByte Phylogenies », *http ://www.exelixis-lab.org/.*

Stamatakis, A., P. Hoover, et J. Rougemont. 2008. « A Rapid Bootstrap Algorithm for the RAxML Web Servers », *Systematic Biology*, vol. 57, no. 5, p. 758–771.

Stephens, J. C. 1985. « Statistical methods of DNA sequence analysis : detection of intragenic recombination or gene conversion », *Molecular Biology and Evolution*, vol. 2, no. 6, p. 539–556.

Swafford, D. 2002. PAUP*. Phylogenetic Analysis Using Parsimony (*and other methods). Version 4.

Templeton, A. R., K. A. Crandall, et C. F. Sing. 1992. « A cladistic analysis of phenotypic associations with haplotypes inferred from restriction endonuclease mapping and DNA sequence data. III. Cladogram estimation. », *Genetics*, vol. 132, no. 2, p. 619–633.

van Iersel, L., S. Kelk, R. Rupp, et D. Huson. 2010. « Phylogenetic networks do not

need to be complex : using fewer reticulations to represent conflicting clusters », *Bioinformatics*, vol. 26, no. 12, p. i124–i131.

Watson, J. D. et F. H. Crick. 1953. « Molecular structure of nucleic acids ; a structure for deoxyribose nucleic acid », *Nature*, vol. 171, p. 737–738.

Woolley, S. M., D. Posada, et K. A. Crandall. 2008. « A comparison of phylogenetic network methods using computer simulation », *PLoS ONE*, vol. 3, no. 4, p. e1913.

Zhaxybayeva, O., P. Lapierre, et J. P. Gogarten. 2004. « Genome mosaicism and organismal lineages. », *Trends in genetics : TIG*, vol. 20, no. 5, p. 254–260.