Supertree methods for phylogenomics

Celine Scornavacca

UNIVERSITY OF MONTPELLIER II
DOCTORAL SCHOOL I2S
INFORMATION STRUCTURES SYSTEMES

P H D T H E S I S

to obtain the title of

PhD of Science

of the University of Montpellier II
Specialty : Computer Science

Defended by

Celine SCORNAVACCA

Supertree methods for phylogenomics

Méthodes de superarbres pour la phylogénomique

defended on December 8th, 2009

Jury :

Olivier Gascuel	Directeur de recherche, CNRS	*Advisor*
Vincent Berry	Professeur, UM2	*Co-advisor*
Vincent Ranwez	Maître de conférences, UM2	*Co-advisor*
Marie-France Sagot	Directeur de recherche, INRIA	*Referee*
Daniel Huson	Professeur, Tübingen University	*Referee*
Simonetta Gribaldo	Chargé de recherche, Institut Pasteur	*Examinator*

Acknowledgments

All people say that acknowledgments are the last thing to write. So do I, but since my body is falling into pieces after having endured all these months of hard work, I hope to not forget someone.

First of all, I would like to thank Marie-France Sagot, Daniel Huson and Simonetta Gribaldo for agreeing to be members of my thesis committee.

I would like to thank Olivier Gascuel for finding the time to read my thesis, although his time table is full until 2020, and to have improved it with his helpful comments.

What to say about VB & VR, *i.e.*, Vincent Berry and Vincent Ranwez? It is difficult for me to imagine two better co-advisors. They accompanied me in the world of research, always encouraging me, inspiring me and leaving me free to make my choices. They are two excellent scientists and two nice and funny guys. A huge thank-you to my two-headed ogre (a citation for Warcraft fans only).

I wish to thank all the MAB team at the LIRMM for tolerating the high level of my voice in the hallways and for the interesting discussions.

A sincere thank-you to the members of the PPP team at the ISEM for showing me that evolution does not consist only in reconstructing supertrees, especially Emmanuel Douzery and Frédéric Delsuc.

I would like to thank the members of the LBBE team at the University of Lyon I, for welcoming me in France and for making the effort to understand the very bad French that I spoke at that time. A particular thank-you to Marie-France Sagot and Eric Tannier who have believed in me and have given me the possibility to start working in bioinformatics, to Vincent Daubin for his precious suggestions and to Sophie Abby and Simon Penel for their friendship.

Thanks also to all those people who have made my life in Montpellier pleasant. It is impossible to list them all, but a particular thank-you go to Rasta-Amine, for bringing fun and love in my life, Georgia Tsagkogeorga, for being a fantastic flatmate and a really good friend, my moral support during this thesis. And Andrew Rodrigues, a true friend and a faithful climbing parter. I think that all people around me have to thank him to let me take it out on climbing walls rather than on people in the last part of the thesis.

Thank you to my friends that are far away, in Italy and all over the world. The Italian saying "Lontano dagli occhi, lontano dal cuore" does not work for me are they are still in my heart, especially Dada, Giacomone, Circy and Claudia.

A particular thank-you to Fabio Pardi for his help with the Shakespeare language (yes, some Italians can speak a good English!), to Juan Escobar and Samuel Blanquart for answering to all my silly questions about evolution and to Julien Dutheil, who helped me a lot with the Bio++ libraries.

Thank also to Professor Gianpaolo Oriolo for introducing me to Eric Tannier and for giving me the possibility to do my Master stage at the University of Lyon I.

Finally I want to thank my family for their support. They taught me to love sciences and that hard work always pays. I really love you.

This thesis is dedicated to the memory of Roberta Dal Passo, a inspiring mathematician and a true and frank person.

Contents

Introduction

It was three years and a half ago that I arrived in France. At the time, my work was focused on algorithms for Wi-Fi LAN. Yes, it was interesting but I needed to work on something more *warm* than computers. This is why I started working in the field of bioinformatics. Here I am, now, writing this manuscript after having spent several years in this marvelous world where mathematicians and computer scientists mix together with biologists and paleontologists to try to answer to one of the most fascinating questions ever asked: how all organisms on Earth descended from a common ancestor?

This thesis is about combining phylogenies. A *phylogeny* or *phylogenetic tree* consists in *nodes* connected by *branches*. *Leaves*, or *terminal nodes*, represent today organisms for which we can collect data. Internal nodes represent hypothetical ancestors since they cannot be directly observed. The aim of this thesis is to provide algorithms for the reconstruction of phylogenies and, ultimately, to estimate parts of the Tree of Life *i.e.*, the phylogeny describing the relationships of all life on Earth in an evolutionary context.

In **Chapter 1** we introduce the basic objects considered in this thesis, *i.e.*, phylogenetic trees. Moreover, we briefly describe how phylogenies are inferred from biological data, to avoid the reader from thinking that they came "out of the blue" as a *deus ex machina*.

In **Chapter 2** we review the biological phenomena that lead to produce different phylogenies from different data sets *e.g.*, lateral gene transfers, gene duplications and losses. We also present two main approaches to combine different data sets to infer reliable phylogenies, with their pros and cons. The most straightforward approach to combine primary data issued from multiple sources is simply to concatenate them into a single data set called the supermatrix [Sanderson et al., 1998]. On the other hand, the supertree approach first involves inferring partially overlapping, source phylogenetic trees, that were inferred from primary data, and then assembling them into a larger, more comprehensive *supertree* [Bininda-Emonds, 2004b]. In this thesis we focus on the latter approach. The supertree problem is a generalization of a simpler one, called the *consensus* problem, which consists in summarizing a set of trees that classify the same objects into one tree.

In **Chapter 3** we thus present several consensus methods and we provide a review of most supertree methods currently available. We will see that some supertree methods are directly inspired by consensus methods, while others are based on new principles.

When using supertree construction in a divide-and-conquer approach in the attempt to reconstruct large portions of the Tree of Life, conservative supertree methods have to be preferred in order to obtain reliable supertrees. In our opinion a

reliable supertree should display only information that is present in one or several input trees, or induced by their interaction. At the same time, it is desirable that the inferred tree contains as few contradictions as possible with the source trees.

In **Chapter 4** we introduce two combinatorial properties we proposed that implement these ideas. Since no existing supertree method satisfies both these properties, we designed two supertree methods, *PhySIC* and *PhySIC_IST* [Ranwez et al., 2007a; Scornavacca et al., 2008], which infer supertrees satisfying them. A major difference between these two methods is that *PhySIC_IST* can propose non-plenary supertrees while *PhySIC* necessarily proposes a supertree that contains all taxa present in a least one source tree. Further we also designed a statistical preprocessing of the source trees to detect and correct artifactual positions of species. In this chapter we also present an example of application of *PhySIC_IST* to the complex problem of disentangling the phylogeny of Triticeae [Escobar et al., 2009].

Gene trees are usually *multi-labeled*, *i.e.*, a single species can label more than one leaf, since duplication events almost always resulted in the presence of several copies of the genes in the species genomes. Since no supertree method exists to combine multi-labeled trees, until now these trees are simply discarded in a supertree approach. In a phylogenomic framework, where the more data the better, this is not desirable.

In **Chapter 5** we present a way to solve this problem, proposing several algorithms to extract the largest amount of speciation signal for orthologous sequences from multi-labeled trees, and put it under the form of single-labeled trees which can be handled by supertree methods [Scornavacca et al., 2009b]. An application to the HOGENOM database, a database of homologous genes from fully sequenced genomes, is presented.

In this work, the emphasis is on theoretical results, but real biological applications are always kept in mind. The final product of my research tends to be algorithms for which user friendly implementations are freely available. Moreover, for each problem we encounter, biological case studies are presented to demonstrate the relevance of our approaches.

List of publications:

- Escobar, J., A. Cenci, C. Scornavacca, C. Guilhaumon, S. Santoni, E. Douzery, V. Ranwez, S. Glémin, and J. David. 2009. **Combining supermatrix and supertree in Triticeea**. Submitted to Systematic Biology.

- Ranwez, V., V. Berry, A. Criscuolo, P. Fabre, S. Guillemot, C. Scornavacca, and E. Douzery. 2007. **PhySIC: a veto supertree method with desirable properties**. Systematic Biology 56:798–817.

- Scornavacca, C., V. Berry, V. Lefort, E. J. P. Douzery, and V. Ranwez. 2008.

PhySIC_IST: cleaning source trees to infer more informative supertrees. BMC Bioinformatics

- Scornavacca, C., V. Berry, and V. Ranwez. 2009b. **From gene trees to species trees through a supertree approach**. Pages 702–714 in LATA '09: Proceedings of the 3rd International Conference on Language and Automata Theory and Applications, volume 5457 of Lecture Notes in Computer Science, Springer- Verlag, Berlin, Heidelberg. 9:413.

- Scornavacca, C., V. Berry, and V. Ranwez. 2009a. **Building species trees from larger parts of phylogenomic databases**. Submitted to Information and Computation.

Chapter 1

Inferring phylogenies

Contents

Men are curious. Scientists even more. A question that fascinates an increasing number of scientists, especially since the last decades, is to understand how all organisms on Earth descended from a common ancestor. *Phylogenetics* is the subfield of evolutionary biology that studies evolutionary relationships among species through molecular and morphological data.

In this chapter we present how phylogenetics arose as a science and a review of the field.

1.1 From Aristotle to Darwin: an introduction

Since Aristotle, naturalists have always tried to classify the abundance of creatures that populate the Earth. Aristotle believed that creatures were arranged in a graded scale of perfection rising from plants up to man that he called the *scala naturae*. Aristotle's classification of living, even if now completely outdated, contains some truth. For example, he was the first to divide beings in vertebrates and invertebrates (called animals with and without blood in his work). During the Middle-Age and Renaissance almost no progress was done.

The quest of this natural order was the major goal of naturalists of the eighteenth century. Linneaus, maybe the most famous of the systematists, believed in an underlying order in nature that needs to be discovered and expressed as a hierarchy. At his time, classifications of living things were built using as discriminant the phenotype, *i.e.*, any observable characteristics of organisms. Linneaus thought that a reliable discriminant was a character good for ordering as many beings as possible. This method sometimes led Linneaus to classifications that now we consider erroneous. A significant improvement to Linneaus' method was the proposal of the natural classification by Antoine Laurent de Jussieu, based on the use of multiple characters to define groups. No matter the way the groups were formed, in those days all classifications were proposed in the framework of fixism, a theory stating that life on Earth has always been composed of the species we have today and that species never change.

The first naturalist to evoke the possibility that species can evolve was Leclerc de Buffon. He pointed out an evident continuity among individuals of the same species and a less evident, but present, continuity among species. For Buffon the classification was nothing more than an artifact that had to be replaced by the concept of descent.

Jean-Baptiste Lamarck was the first to propose an evolutionary theory. In his oeuvre *Philosophie zoologique* (1809) he introduced the concept of the *general distribution*, *i.e.*, an order produced by the *walk of nature* in living creatures that are seen as being in perpetual evolution. Lamarck was also a fervent opposer of the concept of classification that, for him, «has nothing natural». For Lamarck, the aim of understanding the general distribution was not to be able to classify living creatures but to understand the order that nature followed to produce them. The concept of spontaneous generation of life from inanimate matter prevents Lamarck from proposing a genealogy of living. This is, together with the notion of inheritance of acquired characters, one of the weakest points of his theory.

In *The Origins of Species* (1859), Charles Darwin introduced his theory according to which populations evolve over the course of generations through a process of natural selection and the variability of life arose through a branching pattern of evolution and common descent. He illustrated his theory using a tree where actual species are linked two by two up to a common ancestor species. For Darwin, species could undergo several mutations but the history of life was unique. Others before Darwin used trees to illustrate species classifications in light of fixism (*e.g.*, Augustin Augier) or descent of some species from others (*e.g.*, Charles-Hélion de Barbançois). The originality of Darwin's tree is the coexistence, in the same figure, of the concepts of time and descent: the bifurcations in the tree follow one another over time. It is interesting to note that, unlike Lamarck, Darwin was not a detractor of the concept of classification. For him, once that genealogy of species was found, it would lead to the "natural" classification of living creatures.

It was Ernst Haeckel in 1866 that used for the first time the term *phylogeny* to designate the history of organismal lineages as they change through time. At his

time, phylogenies were built using morphological traits, ontogeny[1] and fossils. With the discovery of DNA by Watson and Crick in 1953 and the design of sequencing techniques, a new kind of information became available: molecular data. Thanks to the huge amount of information available since 10-20 years, phylogenetics entered in its golden age. At that time, some of the problems that are treated in this thesis arose.

Phylogenetics aims at clarifying the evolutionary relationships that exist among different species, represented through phylogenetic trees or *phylogenies*. A phylogeny[2] consists in *nodes* connected by *branches* (see Figure 1.1 for an example). Terminal nodes are called *leaves* or *taxa* and represent today organisms for which we can collect data. Internal nodes represent hypothetical ancestors since they cannot be directly observed. In rooted phylogenetic trees (see Figure 1.1(i)), each internal node represents the most recent common ancestor of its descendants and the only node with no ancestor is called the *root* of the tree. Nodes and branches can have several kinds of information associated with them, such as time or amount of evolution estimates.

1.2 Different types of biological data

Phylogeny reconstruction methods are used to analyze either morphological (structural aspects of organisms such as bone structure, organs, etc.) or molecular (genetic information such as nucleotides, amino acids, codons, SINE or LINE etc.) data. We can consider these data as sequences of characters that can take several states ({0,1} for the presence/absence of a morphological trait, {A,C,G,T} for nucleotidic sites etc.).

To properly reconstruct phylogenies, it is important to be able to determine which characteristics are similar because they were inherited from a common ancestor (*homology*) and which are similar as a result of separate convergent evolution (*homoplasy*). For morphological data, we might consider similar looking features to be homologous when they are not and the similarity is a result of convergent evolution (*e.g.*, the wings of bats and birds). Because the homology among proteins and DNA is often concluded on the basis of sequence similarity, such problems can also arise with molecular data (for example because of gene duplication events[3]).

Moreover, if we want to use molecular data to reconstruct phylogenies, we have to face another problem. For morphological traits, we can only have that the state for a character of a species changes or not in its descendants. In molecular sequences, we can have substitutions (modifications of the site state) as well, but insertions and deletions of some sites are also possible. The result is that the same

[1] Ontogeny is the branch of biology that deals with the development of an individual organism from the fertilized egg to its mature form.

[2] For a formal definition of phylogeny see Chapter 3.

[3] We will introduce the notions of orthology and paralogy in the next chapter.

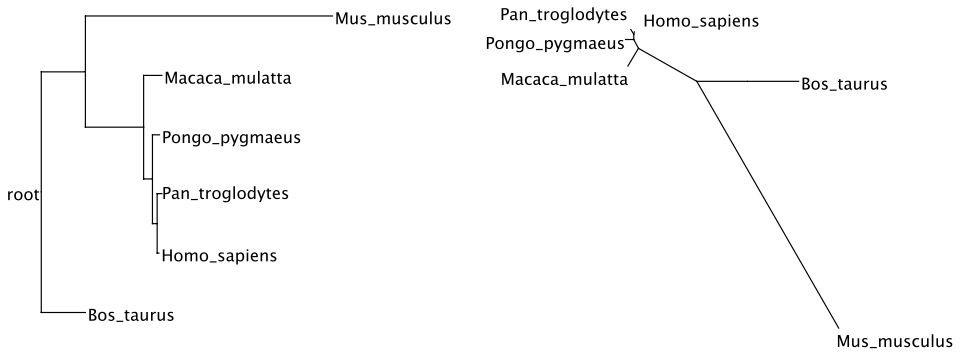


Figure 1.1: **Phylogenetic trees for the Glioma tumor suppressor candidate region gene 1 protein marker (ENSG00000063169), obtained with a maximum likelihood (ML) analysis [Ranwez et al., 2007b]** - where branch lengths represent amounts of evolution between species. Note that in the phylogenetic tree in (i) nodes are connected to other nodes by a horizontal and then a vertical line but only vertical branch lengths represent amounts of evolution. The only difference between these two trees is that the one in (i) is rooted, whereas the one in (ii) is not. In a rooted tree, the root corresponds to the most recent common ancestor of the leaves. This information and therefore the direction of evolution (from the root to the leaves) are lost in an unrooted tree.

molecular marker in different species has different lengths. When this happens, we need to *align* the sequences correctly to be sure that we are really comparing the same characters in all species. A variety of algorithms have been designed to solve the sequence alignment problem, including dynamic programming methods, heuristic algorithms and probabilistic methods. That is why in the following sections we will consider only aligned sets of sequences of same length.

To reconstruct phylogenies two kinds of methods are available:

- character-based methods, which retrieve similarities comparing the states taken by species at different characters; character-based methods can be further divided into:
  - parsimony methods
  - likelihood methods
  - bayesian methods
- distance-based methods, which use pairwise distances to quantify the amount of evolution separating species.

1.3 Parsimony methods

The main hypothesis of these methods is that evolution is parsimonious and the most plausible phylogenies are that requiring the fewest evolutionary changes to explain data. Parsimony methods are based on discrete characters. Input data consist in a set S of n character sequences (one per studied species) $s_1, ..., s_n$ of length m.

The two most widespread variants of parsimony are the Fitch parsimony, where the cost of substituting a state with another is equal to 1 for all states [Fitch, 1971], and Sankoff parsimony, where a substitution cost $C_{x\to y}$ is associated to each pair of states x, y, with $x \neq y$ [Sankoff and Rousseau, 1975]. Fitch parsimony is a special case of the Sankoff parsimony but the algorithm that Fitch proposed for it is not a special case of Sankoff algorithm [Felsenstein, 2004].

Other types of parsimony have been proposed *e.g.*, Dollo parsimony [Farris, 1977; Le Quesne, 1974] and Camin-Sokal parsimony [Camin and Sokal, 1974].

In a parsimony approach each character can be analyzed independently from the others. It follows that, given a phylogeny T, once the parsimony score $\mathrm{P}(c_j|T)$ is calculated for each character c_j, the parsimony score of the set S of all sequences is given by the (weighted) sum of the parsimony score of each character:

$$P(S|T) = \sum_{j=1}^{m} w_j P(c_j|T) \tag{1.1}$$

where w_j is the weight of character c_j. Assuming that internal sequences are known (see figure 1.2) one can easily determine the number of substitutions necessary to explain different states for c_j at the two extremities of a branch e. Denoting this value by $P(c_j|e)$, $P(c_j|T)$ is simply the sum of $P(c_j|e)$ over all branches e of T, weighted by the substitution costs.

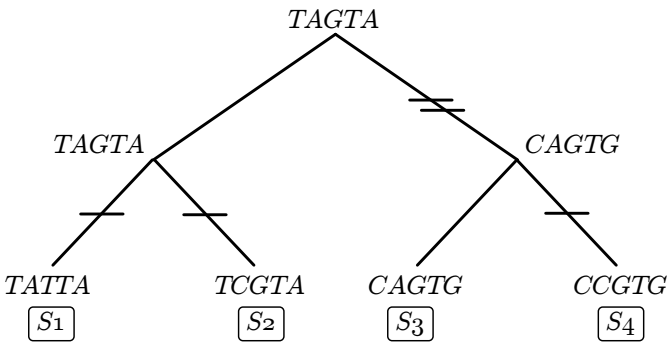


Figure 1.2: **One of the most parsimonious phylogenies for the set of sequences $S = \{S1, S2, S3, S4\}$.** - The five required substitutions are indicated by small horizontal lines.

Since only terminal sequences are known, we need to find the combination of internal sequences that minimizes $P(c_j|T)$ (see Figure 1.3). This problem is not as hard as one may imagine since:

- the number of possible states for a character is limited;
- each character can be analyze independently;
- the choice of the root does not change the parsimony value of a tree, in the usual case where $C_{x\to y} = C_{y\to x}$ holds for each pair of states x, y.

An $O(nm)$ algorithm to calculate $P(S|T)$ was proposed by Fitch [1971]. On the contrary, finding the tree T that gives the minimum value of $P(S|T)$ is an NP-hard problem [Day et al., 1986] for which several heuristic methods were proposed [Felsenstein, 2005; Goloboff et al., 2008; Swofford, 2003].

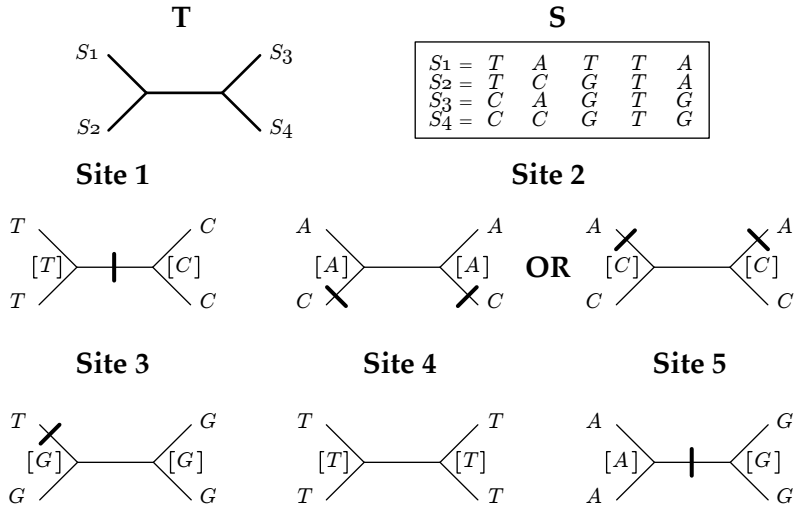


Figure 1.3: **Most parsimonious reconstructions per sites for the set of sequences S given the phylogeny T** - Two equally parsimonious reconstructions are possible for site 2. Deduced internal characters are shown between square brackets.

The main drawback of parsimony methods is that they are not consistent [Cavender, 1978; Felsenstein, 1978]. A method is said to be consistent if the probability to obtain the correct tree converges to one as more and more data are analyzed. For example, parsimony methods are not robust to long branch attractions *i.e.*, when rapidly evolving species that had a separated evolution are inferred to be closely related, regardless of their true evolutionary relationships [*e.g.*, Felsenstein, 1978, see Section 2.1.3]. Indeed, when molecular sequences from two species evolve rapidly, the probability that the same nucleotide appears in both two sequences at the same site increases. When this happens, the most parsimonious scenario is a wrong one, where the two species evolved from a common ancestor. As a matter of fact, rapid evolving species accumulate numerous mutations on a single character and contradict the very foundations of the parsimony approach. For a review of other objections to parsimony methods see Sober [1998].

1.4 Models of sequence evolution

The main limitation of parsimony methods is to try to reconstruct phylogenies without making assumptions on the underlying evolutionary process that species undergo.
At the end of the sixties, the first model of DNA evolution was proposed [Jukes and Cantor, 1969]. The aim of models describing the evolution of sequences is to provide a formal framework to estimate the real number of mutations that a sequence has undergone rather than simply assuming that this number is minimal. This framework has allowed to develop statistically consistent reconstruction methods such that, if the underlying evolutionary model is correct, the method asymptotically converges to the real phylogeny. This section presents a short review of the best known models of nucleotide sequence evolution and evokes protein sequences and codon models. This will be useful in Sections from 1.5 to 1.7.

1.4.1 Nucleotide models

Most nucleotide substitution models share some common hypotheses:

- sequences evolve exclusively through nucleotide substitutions. Nucleotide insertions and deletions are not taken into account;
- substitution processes are independent and identical among sites: substitutions affecting one site do not depend either on substitutions affecting other sites or on the position of the site in the sequence. This implies that knowing the substitution process of sites means knowing that of the sequences;
- substitution process is a first-order Markov model. Having a memory of size 1, the evolution of sequences depends only on the actual state of sequences and not on its previous states;
- substitution process is homogeneous, *i.e.*, it is the same for all branches of the phylogeny and independent among branches;
- substitution process is stationary, *i.e.*, the probability to observe a state x (denoted by π_x) does not depend on the position of the observation date;
- the substitution probability during an infinitesimal time interval dt is proportional to dt.
- there is at most one substitution per infinitesimal time interval dt.

Nucleotides are modeled as discrete characters that can vary in the set of bases $\{A, C, G, T\}$. Nucleotide models are characterized by a 4×4 rate matrix Q where Q_{xy} is the rate at which base x goes to base y. The general expression of Q is the following:

$$Q = \begin{pmatrix} \lambda_A & Q_{AC} & Q_{AG} & Q_{AT} \\ Q_{CA} & \lambda_C & Q_{CG} & Q_{CT} \\ Q_{GA} & Q_{GC} & \lambda_G & Q_{GT} \\ Q_{TA} & Q_{TC} & Q_{TG} & \lambda_T \end{pmatrix} \tag{1.2}$$

where $\lambda_x = -\sum_{x \neq y} Q_{xy}$. The probability matrix is obtained from the rate matrix by computing the system $P(t) = e^{Qt}$. The general expression of $P(t)$ is the following:

$$P(t) = \begin{pmatrix} \bar{\lambda}_A(t) & P_{AC}(t) & P_{AG}(t) & P_{AT}(t) \\ P_{CA}(t) & \bar{\lambda}_C(t) & P_{CG}(t) & P_{CT}(t) \\ P_{GA}(t) & P_{GC}(t) & \bar{\lambda}_G(t) & P_{GT}(t) \\ P_{TA}(t) & P_{TC}(t) & P_{TG}(t) & \bar{\lambda}_T(t) \end{pmatrix} \tag{1.3}$$

where $P_{xy}(t)$ is the probability that a base x changes into a base y in a time interval t and $\bar{\lambda}_x(t) = 1 - \sum_{x \neq y} P_{xy}(t)$.

Time Reversibility

A stationary Markov process is time reversible if (in the steady state) the amount of change from state x to state y is equal to the amount of change from y to x. Almost all DNA evolution models assume time reversibility *i.e.*, that $\forall x, y \in \{A, C, G, T\}$ we have $M_{xy}\pi_x = M_{yx}\pi_y$.

General Time Reversibility (GTR) model

Under this assumption, the general expression of Q in 1.2 becomes:

$$Q_{GTR} = \begin{pmatrix} \lambda_A & \pi_C R_{AC} & \pi_G R_{AG} & \pi_T R_{AT} \\ \pi_A R_{AC} & \lambda_C & \pi_G R_{CG} & \pi_T R_{CT} \\ \pi_A R_{AG} & \pi_C R_{CG} & \lambda_G & \pi_T R_{GT} \\ \pi_A R_{AT} & \pi_C R_{CT} & \pi_G R_{GT} & \lambda_T \end{pmatrix} \tag{1.4}$$

where the term R_{xy} is equal to M_{xy}/π_y.

The GTR substitution model [Tavaré, 1986; Yang, 1994] requires 6 substitution rate parameters, as well as 4 base frequency parameters. Since $\sum \pi_x = 1$ there are only 3 free frequency parameters. Moreover, if rate parameters are considered as relative rate parameters, one rate can be fixed to 1 (*e.g.*, R_{GT}). It follows that the number of free parameters of the GTR model is equal to 8. All models in table 1.4.1 are particular cases of the GTR model. Some models assume Equal Base Frequencies (EBF=y) *i.e.*, $\pi_x = 0.25\ \forall x \in \{A, C, G, T\}$. All other models assume that $\pi_C \neq \pi_G \neq \pi_A \neq \pi_T$, except the T92 model that hypothesizes $\pi_C = \pi_G = \pi/2$ and $\pi_A = \pi_T = (1-\pi)/2$. Models with a Number of Different Types of Substitutions (NDTS) equal to 1 suppose that $R_{xy} = \alpha$, $\forall x, y \in \{A, C, G, T\}$, $x \neq y$. Models with NDTS = 2 distinguish between transitions (A <-> G, *i.e.*, changes from purine to purine, or C <-> T, *i.e.*, changes from pyrimidine to pyrimidine) and transversions (from purine to pyrimidine or vice versa). Models with NDTS = 3 distinguish

between the two different types of transition, *i.e.*, $R_{AG} \neq R_{CT}$ while transversions are all assumed to occur at the same rate. Several other special cases of the GTR

Model	NDTS	EBF	TNP
JC69 [Jukes and Cantor, 1969]	1	y	0
F81 [Felsenstein, 1981]	1	n	3
K80 or K2P [Kimura, 1980]	2	y	1
HKY85 [Hasegawa et al., 1985b]	2	n	4
F84 [Kishino and Hasegawa, 1989] [Felsenstein and Churchill, 1996]	2	n	4
T92 [Tamura, 1992]	2	n	2
K3ST [Kimura, 1981]	3	y	2
TN93 [Tamura and Nei, 1993]	3	n	5
SYM [Zharkikh and Li, 1995]	6	y	5

Table 1.1: **Nucleotide models that are special cases of the GTR model-** NDTS is the Number of Different Types of Substitutions distinguished by the model, EBF specifies whether the model assumes Equal Base Frequencies and TNP is its Total Number of free Parameters.

model have been described and named.

More complex models

The models described above all assume that each position is evolving independently and identically. Site to site rate variation has also been modeled, mostly by a gamma distribution among sites [Yang, 1993, 1996a] and the presence of a proportion of invariable sites in the data set [Hasegawa et al., 1987]. The gamma distribution, introduced in molecular evolution by Uzzell and Corbin [1971] and developed by Jin and Nei [1990] and Yang [1993], has several advantages: it is analytically tractable, varies from 0 to ∞ and has a single parameter to control both the distribution shape and its mean and variance.

Galtier and Gouy [1998] proposed models for which the substitution process is non-homogeneous, *i.e.*, model parameters are not the same for all branches of the phylogeny and can vary at the nodes of the tree. Galtier [2001] proposed heterotachous models of sequence evolution for which rates of evolution can vary among sites. Both the proportion of sites undergoing rate changes and the rate of rate change are free variables. Note that Galtier's 2001 model provides an alternative to the gamma distribution of rates across sites.

The CAT mixture model [Lartillot and Philippe, 2004] accounts for across-site heterogeneities of the substitutional processes. The total number of classes of the underlying mixture is not specified a priori, but is a free variable of the model.

The BP model [Blanquart and Lartillot, 2006] permits model parameters to vary along the phylogeny, changing not only at every node as in Galtier and Gouy [1998],

but also along branches.

The CAT+BP model [Blanquart and Lartillot, 2008] combines the CAT and BP models.

The latter three models are very complex and computationally demanding and have only been implemented into Bayesian frameworks (see section 1.7).

Several other methods have been recently proposed (for a review see Galtier et al. [2005]).

Adding parameters will almost always improve fit to data, but also leads to a larger estimation error. To discourage overfitting, statistical tests that attempt to find the model that best explains the data with a minimum of free parameters have been proposed (*e.g.*, the AIC [Akaike, 1974] and the BIC [Schwarz, 1978]).

1.4.2 Protein models

The first amino acid models have been proposed at the end of the 1970s. The main advantage in favor of using amino acid information is the fact that DNA undergoes much more back substitutions, making it harder to accurately recover tree evolutionary histories, especially those with long evolutionary distances. Since in nature there exist 20 amino acids, a GTR-like model for proteins would require 208 parameters and would be overparameterized for most data sets. That is why models of protein evolution are often based on empirical matrices that are obtained averaging the observed changes and amino acid frequencies between numerous proteins. The resulting matrices state the relative rates of replacement from one amino acid to another. The most commonly used protein models are PAM [Dayhoff et al., 1978], JTT [Jones et al., 1992], Blosum62 [Henikoff and Henikoff, 1992], WAG [Whelan and Goldman, 2001] and LG [Le and Gascuel, 2008] matrices.

Note that the CAT and the BP models afore-described can also be used to model protein evolution.

1.4.3 Codon models

Lately, models of codon evolution have been proposed [Goldman and Yang, 1994]. They are used mainly to infer the selection forces acting on a protein that can be hidden by the fact that most amino acids are encoded by more than one codon[4]. This degeneracy of the genetic code allows substitutions to occur in the DNA sequence that do not result in a change in the corresponding amino acid sequence. For a review of existing codon models see Delport et al. [2009].

1.5 Distance-based methods

Distance-based methods use pairwise evolutionary distances to reconstruct phylogenies. But how do we calculate those distances?

[4] In nature, there exist 61 coding codons and only 20 amino acids.

1.5.1 Estimation of evolutionary distances

The evolutionary distance D_{sz} between two sequences s and z is defined as the average number of substitution events per site that have occurred since s and z have diverged. A rough estimate for D_{sz} is given by f_{sz}, defined as the proportion of sites that have different states in s and z. The observed value f_{sz} is an underestimate of D_{sz} since it cannot take into account such events as multiple, parallel, convergent, coincidental and back substitutions. Better estimations of D_{sz} can be found if we use a substitution model such as those described in the previous section since this would allow to take into account multiple substitutions for a single site. Let suppose we choose JC69 as model and denote by α the unique substitution rate. In this case, computing the system $P(t) = e^{Qt}$ we obtain:

$$P_{xy}(t) = \begin{cases} \frac{1}{4}(1 - e^{-4\alpha t}) & \text{if } x \neq y \\ \\ \frac{1}{4}(1 + 3e^{-4\alpha t}) & \text{otherwise} \end{cases} \tag{1.5}$$

Suppose that a time t elapsed since the divergence of the two sequences. Then the two sequences are separated by a time $2t$ and we can easily calculate the probability for a site to have different states in s and z, denoted by $p_{sz}(t)$:

$$p_{sz}(t) = \frac{3}{4}(1 - e^{-8\alpha t}) \tag{1.6}$$

It follows that:

$$\alpha t = -\frac{1}{8}\ln(1 - \frac{4}{3}p_{sz}(t)) \tag{1.7}$$

From the definition of α, the average number of substitution events per site that occurred since s and z diverged, *i.e.*, D_{sz}, can be estimated by $2t \times 3\alpha$, because each site changes its state with a probability 3α per time unit. This implies that:

$$D_{sz} = -\frac{3}{4}\ln(1 - \frac{4}{3}p_{sz}(t)) \tag{1.8}$$

Since $p_{sz}(t) = E(f_{sz})$, we can use f_{sz} to estimate $p_{sz}(t)$. Then we obtain:

$$\hat{D}_{sz} = -\frac{3}{4}\ln(1 - \frac{4}{3}f_{sz}) \tag{1.9}$$

which is a better estimation of D_{sz} than f_{sz}.

For other simple models, analytical formulae for the estimation of D_{sz} are available. If the model is too complex a likelihood optimization (see Section 1.6) is used to estimate $\hat{D}$. Other kinds of distances have been proposed *e.g.*, the LogDet distance [Barry and Hartigan, 1987; Steel, 1994]. For a review of evolutionary distances see Zharkikh [1994].

Phylogeny-reconstruction distance methods are applied to dissimilarity matrices $\hat{D}$ obtained from sequence matrices (see above). Ideally, the aim of these methods is to find the phylogeny T such that the length of the path between species s and z

in T, also called patristic distance of s and z, is equal to $\hat{D}_{sz}$. If there exists a tree whose patristic distances are $\hat{D}$, $\hat{D}$ is said to be a tree distance. Since we do not have the real distances but only an estimation of them, usually no such tree exists. A result obtained independently by several authors [among others Buneman, 1971] states the properties that a distance matrix $\hat{D}$ has to satisfy to be a tree distance:

Proposition 1.5.1 *$\hat{D}_{sz}$ is a tree distance (also called* additive*) if and only if it verifies the following three conditions:*

- $\hat{D}_{sz} \geqslant 0$ *between two different species, and is zero if and only if* $s = z$,
- *it is symmetric, i.e.,* $\hat{D}_{sz} = \hat{D}_{zs}$,
- *for all quadruplets of species (s,z,t,u),* $\hat{D}_{sz}+\hat{D}_{tu} \leqslant max\{\hat{D}_{st}+\hat{D}_{zu}, \hat{D}_{su}+\hat{D}_{zt}\}$.

The third condition is often called the *four point condition*. Since dissimilarity matrices obtained as explained above hardly ever verify the four point condition, the goal of distance methods is to find the phylogeny T whose patristic distances are as close as possible to $\hat{D}$. The way of defining what a "as close as possible" means varies to one distance method to another. In the next sections we present the most used distance methods.

1.5.2 Least-squares methods

Least-squares methods (LS) aim at adjusting the given distance matrix $\hat{D}$ to obtain a tree distance $\check{D}$ that minimizes a measure of discrepancy, defined as follows:

$$Q = \sum_{s<z} w_{sz}(\hat{D}_{sz} - \check{D}_{sz})^2 \tag{1.10}$$

where w_{sz} are weights that differ among least-squares methods and are used to account for the uncertainty on the value of $\hat{D}_{sz}$. If $w_{sz} = 1$, formula 1.10 corresponds to the *ordinary least-squares* criterion [Cavalli-Sforza and Edwards, 1967]. Otherwise we have a *weighted least-squares* criterion. Commonly used weights are $1/\hat{D}_{sz}$ [Beyer et al., 1974] and $1/(\hat{D}_{sz})^2$ [Fitch and Margoliash, 1967].

For a given phylogeny T, the tree distance minimizing any least-squares criterion can be found in polynomial time. This approach was first presented by Cavalli-Sforza and Edwards [1967] and improved by Gascuel [1997b] and Bryant and Waddell [1998]. On the contrary, finding the best phylogeny minimizing Q is an NP-hard problem [Day, 1987, 1996] for which several heuristic methods have been proposed. Some variations of the least-squares criterion, called the *generalized least-squares* criterion, have been proposed to take into account the natural correlations between distances [Bulmer, 1991; Susko, 2003].

1.5.3 Minimum-evolution methods

The *minimum evolution* method (ME) aims at minimizing the total length of the reconstructed tree T, *i.e.*,

$$Q = \sum_{e \in T} l(e) \tag{1.11}$$

where $l(e)$ is the length of the branch e and branch lengths, which represent quantities of evolution, are computed using a least-squares method. In a minimum evolution approach, the most plausible phylogeny is that demanding the minimum quantity of evolution. This approach has been first proposed by Kidd and Sgaramella-Zonta [1971] and developed by Rzhetsky and Nei [1992]. It has been proved [Denis and Gascuel, 2003; Rzhetsky and Nei, 1993] that if the estimation of $\hat{D}_{sz}$ tends to D_{sz} and branch lengths are estimated with an ordinary least-squares criterion, then the method converges to the correct phylogeny *i.e.*, it is consistent. On the contrary, Gascuel et al. [2001] have proved that some weighted and generalized least-squares methods, if used to estimate branch lengths, lead to inconsistent versions of the minimum evolution method. Since in the minimum evolution approach branch lengths are computed using a least-squares method, methods that improve the complexity and running time of the latter methods [*e.g.*, Bryant and Waddell, 1998], also speed up the former. Improved search methods have also been proposed [Desper and Gascuel, 2002; Kumar, 1996].

Clustering methods for the minimum evolution approach have been proposed. They first construct a star tree connecting one central node to leaf nodes representing all species for which we have distances (Figure 1.4(i)). At each step a pair of nodes x, y to cluster is chosen using the information contained in the distance matrix $\hat{D}$. The two nodes are then connected to a new node v that is in turn connected to the central node (Figure 1.4(ii)). The two rows and columns corresponding to x and y are removed from the matrix $\hat{D}$ while an extra row and an extra column are added to $\hat{D}$ for the new node v. On the whole, the dimension of $\hat{D}$ is decreased by 1. Then the distances $\hat{D}_{iv}$ between all nodes i in the matrix and v are computed. After $n-2$ steps a completely resolved phylogeny is obtained (Figure 1.4(iii)), where n is the number of species. Clustering methods vary in the way they choose nodes to cluster and compute the new distances $\hat{D}_{iv}$. The most widely used are NJ [Saitou and Nei, 1987], UNJ [Gascuel, 1997b], BIONJ [Gascuel, 1997a] and WEIGHBOR [Bruno et al., 2000].

A heuristic method for the ME that is not clustering-based is FASTME [Desper and Gascuel, 2002] that aims at minimizing the balanced minimum evolution criterion introduced by Pauplin [2000]. Also NJ is a heuristic for the same criterion as proved by Gascuel and Steel [2006], while UNJ is a heuristic to minimize the ordinary least-squares version of ME.

.

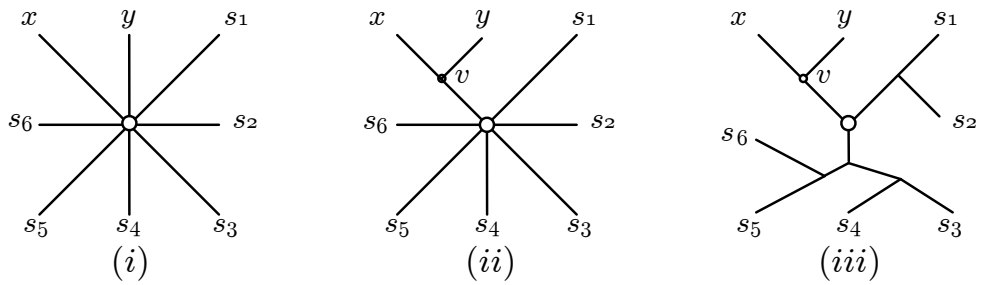


Figure 1.4: **The clustering process to build a phylogenetic tree** - (i) the initial situation. (ii) the first clustering groups x and y. (c) the final situation.

1.6 Likelihood methods

Likelihood methods were first introduced in phylogeny by Edwards and Cavalli-Sforza [1964] to deal with gene frequency data. The first applications to molecular sequences was proposed by Neyman [1971] and improved by Kashyap and Subas [1974] and Felsenstein [1981].

In this section and in the following one, we denote as θ the vector of all the parameters of an evolutionary model, where here an evolutionary model is the combination of a substitution model M (see Section 1.4), a topology and its branch lengths.

Given a sequence alignment S of n character sequences (one per studied species) of length m and a vector of parameters of an evolutionary model θ, the likelihood of θ, denoted by $\mathbb{P}(S|\theta)$, is defined as the probability to observe the data set S, given θ. The likelihood can be viewed as a function of θ.

The hypothesis of the independence of the evolution of each site, already evoked in Section 1.4.1, implies that

$$\mathbb{P}(S|\theta) = \prod_{j=1}^{m} \mathbb{P}(c_j|\theta) \tag{1.12}$$

This simplifies a lot the calculation of the likelihood. When the vector θ is given, the topology of T is known. In such a case, to compute the likelihood of a site c_j, we associate at each node $u \in T$ a likelihood vector $L_{c_j,u} = (L_{c_j,u,A}, L_{c_j,u,T}, L_{c_j,u,G}, L_{c_j,u,C})$, where $L_{c_j,u,x}$ is the probability of observing the state x at the node u, with $x \in \{A, T, G, C\}$. The reversibility hypothesis, assumed by most models of sequence evolution, implies that the likelihood of T does not depend on the position of the root [Felsenstein, 1981, the "pulley principle"]. We can then compute the likelihood of an unrooted phylogeny rooting it on whatever branch or node. Once the tree is rooted, the algorithm starts initializing the likelihood vectors associated to each leaf of T in the following way: $L_{c_j,u,x} = 1$ if the leaf u has state x at the site c_j, otherwise $L_{c_j,u,x} = 0$. If the state of site c_j is unknown, then $L_{c_j,u,x} = 1$ $\forall x \in \{A, T, G, C\}$ [Felsenstein, 2004, page 255]. Internal nodes are considered in

a bottom-up tree traversal *i.e.*, a node cannot be treated before all its sons have been. For each internal node u, its likelihood vector is computed from the likelihood vectors of its sons $l(u)$ and $r(u)$ as follows:

$$L_{c_j,u,x} = \left(\sum_{y \in \{A,T,G,C\}} P_{xy}(b_{u,l_u u}) \cdot L_{c_j,l(u),y} \right) \cdot \left(\sum_{y \in \{A,T,G,C\}} P_{xy}(b_{u,r(u)}) \cdot L_{c_j,r(u),y} \right) \quad (1.13)$$

where $b_{u,l(u)}$, resp $b_{u,r(u)}$, is the length of the branch $(u,l(u))$, resp $(u,r(u))$. The likelihood $\mathbb{P}(c_j|\theta)$ for the site c_j is defined as the product:

$$\prod_{x \in \{A,T,G,C\}} \pi_x L_{c_j,r,x} \quad (1.14)$$

where r is the root node of T and π_x is the equilibrium probability of the base x under the model M. Using this dynamic programming technique [Felsenstein, 1981], the likelihood of T can be computed in $O(nm)$, where n is the number of sequences and m the number of characters of the alignment. Unfortunately, when trying to reconstruct a phylogeny from sequences, θ is unknown. This means that, to find the phylogeny with maximum likelihood, we also need to consider all combinations of its parameters. This is the main limit of this approach, but for simple evolutionary models, when the likelihood of a tree can rapidly be computed, efficient heuristics have been developed. These methods are considered as being among those inferring the most reliable phylogenies.

Heuristic methods for the maximum likelihood approach have been implemented in several programs, *e.g.*, PAUP* [Swofford, 2003], PHYML [Guindon and Gascuel, 2003], IQPNNI [Vinh and Von Haeseler, 2004], RAxML [Stamatakis, 2006] and GARLI [Zwickl, 2006]. The latter uses a stochastic, genetic algorithm-like approach instead of deterministic hill climbing.

For complex methods for which analytical solutions cannot be found even when θ is known, an ML approach is not tractable. That is why a bayesian approach to phylogeny reconstruction has been proposed.

1.7 Bayesian methods

Bayesian methods to infer phylogenies are closely related to likelihood methods. Bayesian inference of phylogeny is based on a quantity called the posterior probability of a parameter vector θ of an evolutionary model, given a sequence alignment S, denoted by $\mathbb{P}(\theta|S)$. Bayes' theorem allows to turn a prior distribution of θ, denoted by $\mathbb{P}(\theta)$ into its posterior distribution:

$$\mathbb{P}(\theta|S) = \frac{\mathbb{P}(\theta)\mathbb{P}(S|\theta)}{\mathbb{P}(S)} \quad (1.15)$$

where $\mathbb{P}(S|\theta)$ is the likelihood of the sequence alignment S given the parameter vector θ. The posterior probability of θ can be interpreted as the probability that

the parameter vector θ is the correct one. In order not to influence the result with personal opinions, a flat prior can be assigned. It is also possible to assign vague priors [Huelsenbeck et al., 2002b]. Note that the ML approach is a particular case of the bayesian approach, for which flat prior are chosen [Kuhner et al., 1995]. The denominator in 1.15 involves a summation over all trees and, for each tree an integration over all possible branch lengths and parameters of the substitution model. This computation is often analytically impossible, but numerical methods can be used to efficiently approximate the distribution of posterior probability. The most used are Markov Chain Monte Carlo (MCMC) methods [*e.g.,* Gilks et al., 1995] that permit to wander randomly through the posterior distribution over parameter and tree space. Once this random walk reaches equilibrium, samples of parameter vectors are collected and will be used to approximate their posterior probability distribution. For phylogeny inference, the MCMC algorithm is based on the Metropolis algorithm [Metropolis et al., 1953] and consists of the following steps:

1. start with a random vector of parameters θ_i;
2. select a new vector θ_j by modifying θ_i in some way;
3. compute the *acceptance ratio*
$$R = \frac{\mathbb{P}(\theta_j|S)}{\mathbb{P}(\theta_i|S)} = \frac{\mathbb{P}(\theta_j)\mathbb{P}(S|\theta_j)}{\mathbb{P}(\theta_i)\mathbb{P}(S|\theta_i)};$$
4. accept θ_j with a probability ρ=max$(R, 1)$;
5. every k generations, save the current tree and all parameters;
6. return to step 2.

Note that denominator in 1.15 disappears in the computation of R. This algorithm has no termination. It is up to the user to stop it after a number of generations considered sufficient. This is one of the limits of this approach. Note also that this algorithm is a Markov chain of order one since θ_j depends only on θ_i. We call the vector of parameters θ_i the "state θ_i". To reach the equilibrium distribution, the Markov chain must be aperiodic (no cycles should be present in the Markov chain), irreducible (every state must be accessible from any other state), and the probability of proposing θ_j if the current state is θ_i has to be the same as that of proposing θ_i if we are in θ_j, denoted respectively by $\mathbb{P}(\theta_j|\theta_i)$ and $\mathbb{P}(\theta_i|\theta_j)$. If this is not true, a variant of Metropolis algorithm, the Metropolis-Hasting algorithm [Hastings, 1970] has to be used. Hasting's algorithm differs from the Metropolis' one in the computation of the acceptance ratio, which equals $R \cdot \frac{\mathbb{P}(\theta_j|\theta_i)}{\mathbb{P}(\theta_i|\theta_j)}$. When the Markov chain has the required properties to reach the equilibrium and is run long enough, the time the chain spends in a state θ_i is proportional to its posterior probability [Metropolis et al., 1953].

If the target distribution has multiple local peaks, separated by low valleys, the Markov chain may have difficulty in moving from one peak to another. As a result,

the chain may get stuck on one peak and the resulting samples will not approximate the posterior probability correctly. Metropolis Coupled MCMC (called also MC^3), a variant of MCMC, allows multiple peaks in the landscape of trees to be more readily explored. This technique consists roughly in running k MCMC chains with different stationary distributions. One chain is called the *cold chain* and only its information is recorded. Periodically, states between chains may be swapped.

Bayesian methods vary in the way they set prior distributions for parameters, obtain the state θ_j from θ_i and summarize the information of the obtained samples (step 5). Several bayesian approaches for phylogeny reconstruction have been recently proposed [*e.g.*, Huelsenbeck et al., 2002b; Huelsenbeck and Ronquist, 2001; Larget and Simon, 1999; Li et al., 2000; Ronquist and Huelsenbeck, 2003; Yang and Rannala, 1997].

For a review of a bayesian approach to phylogeny estimation see the review of Holder and Lewis [2003] or consult the books of Felsenstein [2004] and Yang [2006].

1.8 Testing the reliability of inferred phylogenies

Methods to reconstruct phylogenies usually produce binary trees. This is mainly due to the fact that their tree space exploration relies on topological modifications defined on binary trees (*e.g.*, NNI). This implies that, when data sets contain little phylogenetic signal, some branches of inferred trees can be poorly supported by data. To estimate branch reliability, character resampling techniques such as the bootstrap have been proposed.

First described by Efron [1979], the bootstrap technique has been used for the first time in phylogenetics by Felsenstein [1985]. Given a tree T obtained with an inference method (see Sections 1.3 - 1.7) from a sequence matrix M with n rows (one per species) and m columns (one per site), this technique consists of three steps. First, a set of pseudo matrices $\mathcal{M} = \{M_1, \cdots, M_k\}$ called *bootstrap replicates* is derived from M. Each $M_i \in \mathcal{M}$ is obtained by sampling, with replacement, columns of M until obtaining a matrix with m columns. Note that drawing columns with replacement implies that some columns can be present more than once in a bootstrap matrix and others can be absent. Second, from each bootstrap replicate M_i a tree T_i is inferred, employing the same inference method used to infer T. Finally, the so-obtained forest $\mathcal{F} = \{T_1, \cdots T_k\}$ is used to estimate the reliability of each branch e of T, with the percentage of trees in $\mathcal{F}$ containing e. This value is called the bootstrap value of e and denoted by $bp(e)$.

The bootstrap technique allows to simulate the variability of the sampling process that led to obtain M. Though most people agree of its practical usefulness, its statistical meaning is still debated. Some authors [Efron, 1979; Felsenstein, 1985] see the value of $bp(e)$ as an estimation of the probability to find the same branch e in a tree T' obtained by analyzing another data set M' with the same inference method. Other authors [among others Hillis and Bull, 1993; Sanderson, 1989] consider $bp(e)$ as an estimation of the probability that the branch e is in the correct phylogeny

while others [*e.g.*, Efron et al., 1996; Felsenstein and Kishino, 1993] interpret $bp(e)$ as a confidence threshold of a statistical hypothesis test.

Whatever its statistical interpretation, all authors agree on discarding branches with low bootstrap values since in any case they are considered as not reliable (see Figure 1.5). The majority-rule consensus (see Section 3.2) of the forest $\mathcal{F}$ is usually used to discard all branches not supported by more than 50% of the trees in $\mathcal{F}$.

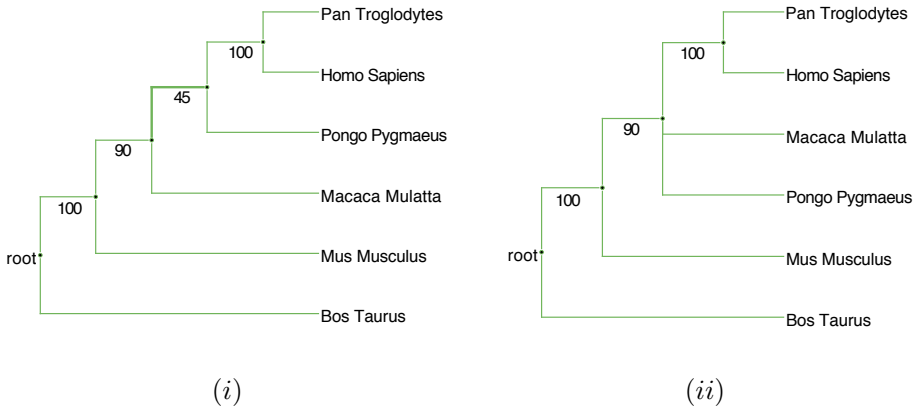


Figure 1.5: **Example of discarding branches with low bootstrap values** - (i) Phylogenenic tree for the Glioma tumor suppressor candidate region gene 1 protein marker (ENSG00000063169), obtained with a maximum likelihood (ML) analysis [Ranwez et al., 2007b]. Support values have been obtained by a bootstrap analysis with 100 replicates. (ii) the tree obtained from that in figure (i), having collapsed branches with $bp \leqslant 50$ (in this case only one branch).

Another well-known character resampling technique is the delete-half jackknife [Felsenstein, 1985; Wu, 1986] that consists in obtaining a set of pseudo matrices randomly by sampling without replacement half of the columns of the matrix M.

To estimate branch reliability for bayesian methods, posterior probabilities are often used, even if tests on simulated data sets have revealed some discrepancies between these values and ML bootstrap estimates [Douady et al., 2003; Erixon et al., 2003]. For a comprehensive review of the ways with which branch reliability can be estimated see Chapter 20 of Felsenstein [2004].

Chapter 2

Deeper insights on multiple data set analysis

Contents

The dawn of molecular techniques for sequencing DNA led to a revolution in phylogenetics. Access to molecular sequences increased the number of homologous characters that could be compared in phylogenetic analyses[1].

A *gene tree* is an evolutionary tree built by analyzing a gene family, *i.e.*, homologous molecular sequences appearing in the genome of different organisms. Gene trees can be used to estimate *species trees*, *i.e.*, trees displaying the evolutionary relationships among studied species. However, as more genes are analysed, topological conflicts between individual gene phylogenies often arise because of methodological or biological reasons. Below we will introduce the most important methodological and biological sources of conflict between gene trees, respectively in Sections 2.1 and 2.2 .

[1]Recall that homologous characters are those that were inherited from a common ancestor.

2.1 Model inadequacy

The first cause of conflicts between individual gene phylogenies is that some gene trees are erroneous because they have been reconstructed using an inadequate model. This happens when the gene sequences evolved according to an evolutionary process that violates the assumptions of the evolutionary model used to infer the gene tree.

There are several causes of model inadequacy. The most important are compositional bias, heterotachy and rapidly evolving lineages [Delsuc et al., 2005].

2.1.1 Compositional bias

One potential pitfall for phylogenetic estimation from biological sequence data is compositional bias. Indeed, convergence in nucleotide composition in unrelated lineages can lead phylogenetic methods to artefactually group together unrelated species with similar nucleotide composition (*e.g.*, G+C or A+T rich) sequences.

It is now well-established that compositional bias in DNA sequences can adversely affect phylogenetic analysis based on those sequences [*e.g.*, Hasegawa et al., 1985a]. The impact of nucleotide bias on protein-based phylogenetic reconstruction is still debated [*e.g.*, Foster and Hickey, 1999; Lockhart et al., 1992; Loomis and Smith, 1990].

2.1.2 Heterotachy

The principle of heterotachy states that the substitution rate of sites in a gene or protein can vary through time [Philippe and Lopez, 2001]. Though often ignored in most used substitution models, heterotachy plays an important role in the process of sequence evolution [Lopez et al., 2002].

There is a growing body of literature on the consistency of likelihood-based methods that ignore heterotachy when the phenomenon is actually present, leading to phylogenetic reconstruction artefacts in cases where the proportions of invariable sites of unrelated taxa have converged [Inagaki et al., 2004; Kolaczkowski and Thornton, 2004; Lockhart et al., 1996; Philippe and Germot, 2000].

Because unlike other types of bias heterotachy does not leave evident trace in sequences [Kolaczkowski and Thornton, 2004], it can lead to artefacts particularly difficult to detect [Inagaki et al., 2004; Philippe and Germot, 2000].

Recently, Kolaczkowski and Thornton [2004] suggested, on the basis of simulations, that MP is substantially less sensitive to heterotachy [Kolaczkowski and Thornton, 2004]. However, Philippe et al. [2005] on the basis of more realistic simulations, showed that MP can also be strongly misled by heterotachy. There is a growing number of models proposed to handle heterotachy, *e.g.*, the covarion model [Tuffley and Steel, 1998], the heterotachous models of Galtier [2001] (see Section 1.4.1 on page 13) and the mixture branch length (MBL) model [Kolaczkowski and Thornton, 2004]. For an evaluation of models handling heterotachy in phylogenetic inference see Zhou et al. [2007].

2.1.3 Rapidly evolving lineages

In phylogenetic analyses, rapidly evolving lineages can be closely related in the inferred tree although they are not. This phenomenon is commonly called Long Branch Attraction (LBA).

Felsenstein [1978] first described the problem on four-taxon trees. He observed that inequalities in the rates of evolutionary change among branches of a four-taxon tree may lead parsimony and compatibility methods to be statistically inconsistent estimators of the phylogeny, grouping together the two rapidly evolving lineages[2]. LBA not only affects parsimony and compatibility methods, but also ML, although less strongly [*e.g.*, Sanderson and Kim, 2000; Sullivan and Swofford, 2001; Swofford et al., 1996].

LBA is a phenomenon of molecular data in particular. Since the number of different states for nucleotides is limited to four (and to 20 for amino acids), when DNA substitution rates are high, the probability that two lineages will evolve the same nucleotide at the same site increases. When this happens, parsimony erroneously interprets this as a *synapomorphy* (*i.e.*, a homologous trait shared by two or more taxa which were derived from a common ancestor) while it is in fact a homoplasy (see Section 1.2 on page 7). This problem can be minimized by using a method less sensitive to LBA, commonly, maximum likelihood [*e.g.*, Huelsenbeck, 1997; Swofford et al., 1996], excluding third codon positions[3] [*e.g.*, Sullivan and Swofford, 1997; Swofford et al., 1996], adding taxa to break up long branches [*e.g.*, Hendy and Penny, 1989; Hillis, 1996; Swofford et al., 1996] etc. For a more extensive review of LBA artifacts and possible solutions to counter it see Bergsten [2005].

2.2 Macro events

Macro events in genome evolution can also lead to topological conflicts between individual gene trees. Here we present these macro events without explaining in detail how they occur, focusing only on how they can lead to conflicts among individual gene phylogenies.

2.2.1 Gene duplications and losses

Gene duplication is considered to play a fundamental role in the evolution of species since the emergence of the last universal common ancestor [*e.g.*, Ohno, 1970; Zhang, 2003], particularly in eukaryotes [*e.g.*, Cotton and Page, 2005; Dujon et al., 2004; Eichler and Sankoff, 2003; Hahn et al., 2007; Lynch and Conery, 2000], and is believed to play a major role in the apparition of novel gene functions [Lynch and Force, 2000].

[2] In reality the slowly evolving lineages are grouped together, leading to group together the two rapidly evolving lineages.

[3] Indeed, the third codon positions in protein-coding sequences, having less selective constraints (because of the degenerescence of the genetic code), evolve faster and are thus often saturated or randomized.

Several processes have been described to account for the origin of gene duplicates, ranging from single gene duplications to whole genome duplications [Durand et al., 2006]. Indeed, major genome duplication events are not uncommon. For instance, it seems that the entire yeast genome underwent a duplication about 100 million years ago [Kellis et al., 2004].

The gene sequences that originate from a gene duplication event are called paralogs (for example, in Figure 2.1(i), the copies α and β for species a). By contrast, orthologous genes are those created from a speciation event (for example, in Figure 2.1(i), the copies α for species b and c).

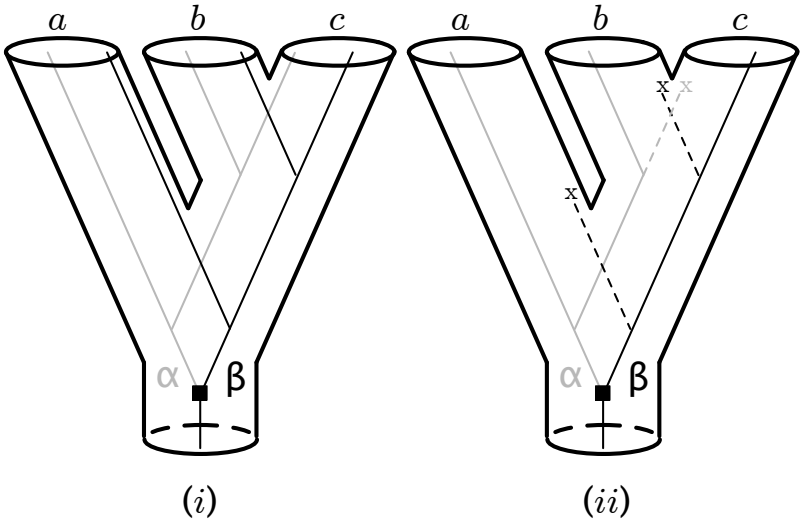


Figure 2.1: **An example of how duplication events can lead to conflict between gene and species trees** - Species trees are depicted as thick pipes or *tubes* and thin lines represent gene trees. Gene losses are represented by an X. (i) the two copies of the gene are available for all species. (ii) the copy α is available for species a and b while the copy β is only available for species c.

Gene duplication can produce conflicts between gene and species trees when some duplicated copies are absent from the analysis, either because they have not been sequenced or because they have been lost at some point during the evolutionary process. For instance, in Figure 2.1(ii), the species tree, depicted as thick pipes, says that b and c are closest relatives with respect to a. Suppose that due to losses during the evolutionary process, the only sequences available are the copy α for species a and b and the copy β for species c. In this case, the gene tree (thin lines inside the pipes) groups species a and b, which are not each other's closest relatives in term of speciation events. This erroneous result comes from the fact that the sequences used to represent species a and b are paralogous with respect to the one used to represent species c. The conflict between gene and species trees due to duplications would disappear if sequences for both copies were available for all species [Doyle, 1992], see Figure 2.1(i) for an example.

2.2.2 Horizontal gene transfers (HGT)

Horizontal gene transfer occurs when an organism transfers its genetic material or part of it to a being other than one of its own offspring. Instead, the two organisms are usually unrelated, and are often of different species. Studies of genes and genomes indicate that considerable horizontal transfer has occurred between prokaryotes [*e.g.*, Jain et al., 1999; Lawrence and Ochman, 1998; Rivera and Lake, 2004]. Indeed, horizontal gene transfer in bacteria is a common phenomenon and is a major factor in accelerating the rate of their evolution [Jain et al., 2003].

There is some evidence that viruses can also transmit genetic information via horizontal gene transfer [Gibbs and Keese, 1995; Pearson, 2008]. The phenomenon appears to have had some significance for unicellular eukaryotes as well. Bapteste et al. [2005] evoked that «*additional evidence suggests that gene transfer might also be an important evolutionary mechanism in protist evolution*». There is some evidence that even higher plants and animals have been affected [*e.g.*, Keeling and Palmer, 2008; Richardson and Palmer, 2007]. However, the prevalence and importance of HGT in the evolution of multicellular eukaryotes remain unclear [Huerta-Cepas et al., 2007; Richardson and Palmer, 2007].

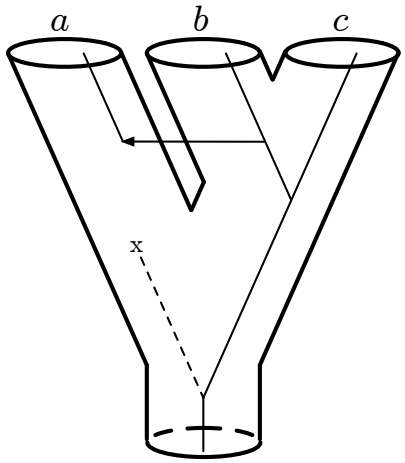


Figure 2.2: **An example of horizontal gene transfer** - Species tree is depicted as thick pipes and thin lines represent the gene tree. The gene lineage with its ancestry in species b is transferred in species a.

Horizontal gene transfer is a potential confounding factor when inferring phylogenetic trees based on the sequence of one gene and can lead to conflicts among individual gene trees. For example, in Figure 2.2, the gene lineage with its ancestry in species b is transferred in species a. Since the sequences of species a and b are more similar with respect to that of species c, the gene tree groups species a and b, which are not each other's closest relatives according to the species tree.

2.2.3 Incomplete lineage sorting

Ancestral polymorphism is the existence of more than one *allele*, (*i.e.*, alternative DNA sequences at the same physical gene locus), at a locus in an ancestral population. The incomplete lineage sorting is the process by which the ancestral polymorphism is retained through speciation events. This can result in misleading similarities of DNA sequences that do not necessarily reflect species relationships. For instance, in Figure 2.3 two alleles α and β are present in an ancestral population and both are present after the speciation events. Since the allele α is retained in species a and b while the allele β is retained in species c, the gene tree reconstructed with this gene family sees a and b as each other's closest relatives while they are not.

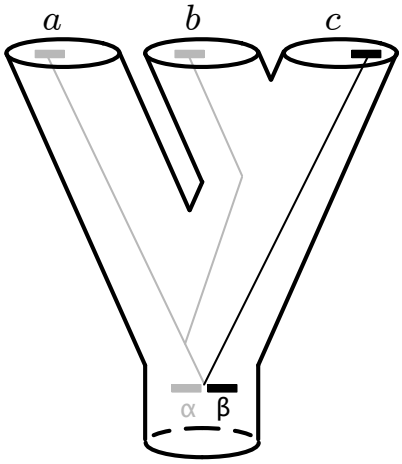


Figure 2.3: **An example of incomplete lineage sorting** - Species tree is depicted as thick pipes and thin lines represent the gene tree. Two alleles α and β are present in an ancestral population and both are retained through speciation events. The allele α is retained in species a and b while the allele β is retained in species c.

2.2.4 Interspecific recombination

Recombination is a molecular process enabling the creation of new combinations of genetic materials through pairing and shuffling of related DNA sequences. Recombination occurs at different levels: individual, population, and species. In prokaryotes and virus, interspecific recombination occurs spontaneously between two organisms.

When interspecific recombination occurs, genetic material is exchanged between different species lineages and this can lead to different histories for neighboring segments within a gene [Posada and Crandall, 2002; Ruths and Nakhleh, 2005]. For instance, in Figure 2.4, species a and c recombined. For the DNA left segment, the gene tree (depicted in thin black lines) is $((b, c), a)$, but for the segment on the right, the gene tree (depicted in thin grey lines) is $((a, b), c)$. We can see interspecies recombination as a reciprocal HGT [Ruths and Nakhleh, 2005].

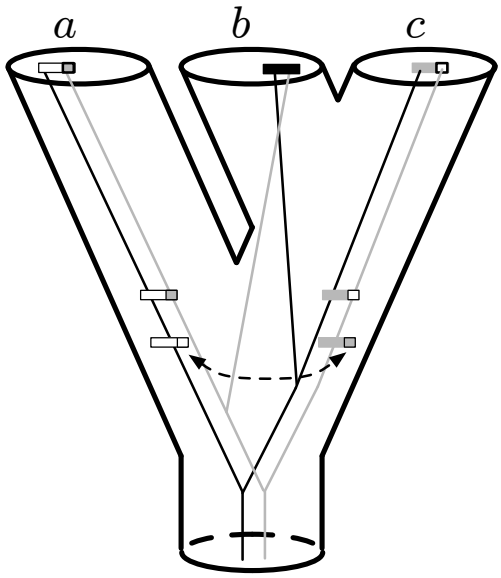


Figure 2.4: **An example of interspecific recombination** - Species tree is depicted as thick pipes and thin lines represent the gene tree. Species a and c recombined. This leads to different histories for neighboring segments within a gene. Indeed, for the left DNA segment, the gene tree is $((b, c), a)$, but for the right DNA segment the gene tree is $((a, b), c)$.

2.2.5 Interspecific hybridization

Interspecific hybriditation is the process by which two individuals of different species come into contact and mate, creating an *hybrid*. The offspring of an interspecific cross are very often sterile, preventing the movement of genes from one species to the other, keeping both species distinct, *e.g.*, mules and hinnies, crosses of horses and donkeys. However, hybridization is a widespread phenomenon in plants [Rieseberg, 1997; Rieseberg et al., 2000] and resulting hybrids are more often fertile than animal hybrids. Interspecific hybridization can lead to conflicts among individual gene trees since the underlying species evolution can no longer be represented by a tree. For example, in Figure 2.5(i), species b is a cross of species a and c. Then, the gene tree reconstructed from a gene that b inherited from a (in grey thin lines) is $((a, b), c)$, but for a gene that b inherited from c, the gene tree (in black thin lines) is $((b, c), a)$.

2.3 Combining data

Since topological conflicts frequently arise among source trees both because of model inadequacy and macro evolutionary events, it is a common practice to include as wide a range of genes for phylogenetic analysis as possible.

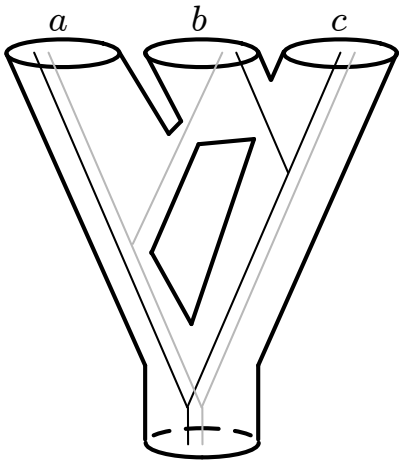


Figure 2.5: **An example of interspecific hybridization** - Species evolution is depicted as thick pipes and thin lines represent the gene tree. Species b is a cross of species a and c. The gene tree for a gene that b inherited from a (in grey thin lines) is $((a, b), c)$, but, for a gene that b inherited from c, the gene tree (in black thin lines) is $((b, c), a)$.

2.3.1 Combining data through a supermatrix approach

The most straightforward approach for combining data issued for multiple sources is simply to concatenate the original sequence alignments into a single larger matrix called the *supermatrix* where unknown character states are coded by question marks. This approach has the advantage that all information of each individual source is retained. This is in accordance with the so-called *total evidence* approach to combining phylogenetic information [Kluge, 1989; Sanderson et al., 1998] *i.e.*, the philosophical principle for which the best hypothesis is the one derived from all the available data.

However, this approach has several limitations.

First, this strategy for assembling ever larger phylogenies is untenable [Sanderson et al., 1998], since, if only a few taxa are common between data sets, most of the newly combined data matrix will be scored as question marks. For instance, one of the biggest supermatrix ever analyzed [Driskell et al., 2004], issued of the concatenation of 1131 protein alignments, containing 469,497 sites for 70 taxa, was composed of 92% of missing data. Analyses of a supermatrix with too many missing data can be in some cases unreliable [Sanderson et al., 1998], notably when the concatenated signal of the supermatrix is not strong enough. Moreover, even without missing taxa, methods such as maximum likelihood tend to become computationally intractable when the data set grows too much.

Additionally, only data of the same type can be concatenated. For instance, no evolutionary model is available for a supermatrix obtained by concatenating nu-

cleotide and amino acids sequences or SINE characters. Moreover, the combined primary data, even encoded in the same way, are analyzed using a single evolutionary model and this can be problematic. For instance, genes can have different stationary frequencies or undergo heterogeneous selective pressures resulting in different evolution rates. These evolutionary rates may also vary from one part of the phylogeny to another but these variations may be specific to each gene and may once again vary from one gene to another. As a result, considering the supermatrix as a single super-gene (the "concatenate model") may be a very rough inadequate approximation. On the other hand the "separate model" [Yang, 1996b], which considers that each gene has its own evolutionary parameters (*i.e.*, its own probability of base mutation, stationary frequencies and branch lengths) requires to design new dedicated optimization heuristics.

Partitioned Bayesian analyses have been recently proposed to cope with this problem. This approach consists in partitioning the supermatrix[4] and then applying appropriate models and their specified parameter estimates to each data partition and subsequently incorporate this into a single tree search. Bayesian methods to conduct such partitioned analyses have recently become available [Ronquist and Huelsenbeck, 2003] and are more and more used [*e.g.*, Brandley et al., 2005; Nylander et al., 2004]. Recent studies [Bevan et al., 2007] have demonstrated that this is the best approach to account for gene rate heterogeneity among those so far designed.

These models have the drawback to introduce a huge number of parameters and this may result in over-parametrized models as unadapted as the under-parametrized "concatenate" one. Furthermore, as evoked at the beginning of the chapter, gene phylogenies can differ among them while in the "separate" model the underlying phylogeny is the same for all partitions.

Another limitation of the supermatrix approach is that some kind of data (e.g. DNA-DNA hybridization, distance data, morphometric data) cannot be analyzed under any of the frameworks developed for more usual kind of data (molecular sequences or morphological traits), *i.e.*, maximum parsimony, maximum likelihood and Bayesian methods [Bininda-Emonds et al., 2003]. For instance, *concatenating* side by side several distance data sets makes no sense. Recently, Criscuolo et al. [2006] proposed a phylogenomic approach to combine distance data (see Section 3.3.2.6).

Note also that the supermatrix approach is sensible to the relative sizes of data sets. For instance, if two data sets conflict and one is substantially smaller than the other, the supermatrix is dominated by the signal of the biggest one. One way to avoid this behavior is to weight data sets with a weight inversely proportional to their number of sites, but the use of weighting in phylogenomics is not established yet.

[4]Note that each partition can contain more than one gene.

2.3.2 Combining data through a supertree approach

Supertree construction is a meta-analysis of phylogenetics: results from the analyses of several smaller data sets are combined together into a larger phylogeny [Sanderson et al., 1998]. This approach, unlike the supermatrix one, combines phylogenies resulted from smaller analyses rather than combining the underlying data. Supertree approach can be used to build very large phylogenies from partially overlapping analyses. It can also be used in some situations where the supermatrix approach cannot. For instance, input trees can be based on different kinds of data, that is, for instance, DNA of different genes, morphology, DNA-DNA hybridization. They can be obtained by different methodologies, for instance, maximum parsimony, maximum likelihood, neighbor-joining, allowing to use the most adapted for each data set.
Supertree methods have been strongly criticized [*e.g.*, Rodrigo, 1993, 1996; Slowinski and Page, 1999; Springer and De Jong, 2001] mainly since the source data are the topologies resulted from the analyses of several smaller data sets form rather than primary character data. The next sections discuss some of the most relevant criticisms against this approach.

Inability to account for signal enhancement and the creation of spurious novel clades

It has been demonstrated [Barrett et al., 1991; Chippindale and Wiens, 1994] that a supermatrix analysis of two data sets yielding conflicting phylogenetic trees can produce a phylogeny in which the congruent subsignals in each data set overcome the individual conflicting primary signals. This phenomenon, called *signal enhancement*, cannot occur in supertree construction, which cannot account easily for subsignals in the original data sets since it combines trees and not the underlying data [Pisani and Wilkinson, 2002].

The incapability of supertree methods to account for signal enhancement and the potential for supertree methods to create *novel* clades not supported by any (combination of) intput tree(s) [Bininda-Emonds and Bryant, 1998] have been strongly criticized [among others Gatesy et al., 2002; Gatesy and Springer, 2004; Goloboff and Pol, 2002; Pisani and Wilkinson, 2002; Springer and De Jong, 2001].

Bininda-Emonds [2003] showed that supertree analyses on simulated data sets are often as accurate as supermatrix analyses of the combined primary character data and produce few, if any, novel clades. Bininda-Emonds [2004b] suggested that the inherent loss of information due to the inability to account for signal enhancement is not so harmful in practice. Moreover, the frequency with which clades result from signal enhancement is not yet adequately documented and this phenomenon may be very rare. Furthermore, as evoked at page 31, signal enhancement can be dominated by the signal of the biggest data sets. Additionally, the potential to create spurious novel clades is only a feature of some supertree methods (see Section 3.3.2.1) but is not inherent to the supertree approach in general.

Data Duplication

Gatesy et al. [2002] argued that several supertrees analyses [*e.g.*, Bininda-Emonds et al., 1999; Liu et al., 2001] contained duplicated data that artificially increase their impact on the supertree construction, potentially biasing the results. The continual recycling of phylogenetic data makes difficult to avoid data duplication in a supertree approach where trees are combined instead of the primary character data. Bininda-Emonds et al. [2003] pointed out that duplicated data are also present in supermatrix analysis. For instance, several characters are often described for a single morphological structure. To avoid data duplication in supertree analyses, Bininda-Emonds et al. [2004] proposed a formal data collection protocol for selecting the source phylogenies choosing only those containing what would be considered to be independent data sets for analyses.

Source Data Quality

Gatesy et al. [2002] criticized several supertree analyses for using data of poor quality. Bininda-Emonds et al. [2003] argued that «*the use of poor data may compromise the results in any phylogenetic analysis (i.e., including a supermatrix analysis), and researchers should ensure that all data used are of the highest achievable quality*».

A formal data collection protocol for selecting the source phylogenies, as that proposed by Bininda-Emonds et al. [2004], and the usage of node support estimations (see Section 1.8) can amend this problem.

Data Accountability

Gatesy et al. [2002] also argued that primary data are explicit in supermatrix analyses contrary to the supertree construction that suffers from a lack of both data accountability and transparency.

Though this is true, it is not a discriminating element for the choice of the supermatrix approach. Indeed, supermatrices may also suffer from both limited data transparency [Jenner, 2001] and lack of data accountability, since database information is known to contain some errors due to vector contaminations, transcription errors etc. [Bininda-Emonds et al., 2003].

The validity of supertrees as phylogenetic hypotheses

It has been argued that supertrees, as summaries of summaries, are not valid phylogenetic hypotheses and, therefore, should not be used to propose new phylogenies [*e.g.*, Gatesy et al., 2002; Gatesy and Springer, 2004; Springer and De Jong, 2001]. Bininda-Emonds [2004a] claimed that supertrees propose hypotheses of statements of taxa relationships that have to be evaluated as any other phylogenetic hypothesis. Discrepancies between supertree and supermatrix analyses issued from the same data should be treated in the same manner as conflicts between conventional

phylogenetic analyses.

In 1995, Purvis used the MRP supertree method (see Section 3.3.2.1) to produce a complete phylogeny for all 203 extant species of primates. From then on, supertree methods have been used increasingly to construct phylogenetic trees of clades with several hundred species [*e.g.*, Davies et al., 2004; Pisani et al., 2002; Salamin et al., 2002]. Bininda-Emonds [2005] hypothesized that probably none of the complete supertrees that exist containing hundreds of species could have been constructed using a supermatrix approach.

Bininda-Emonds' consideration comes mainly from the observation that data collection is largely uncoordinated and opportunistic [Sanderson et al., 2003] *i.e.*, some species are overrepresented, whereas others are drastically underrepresented. For instance, in March 2004 the GenBank database (`http://www.ncbi.nlm.nih.gov/Genbank/index.html`) contained nearly two million carnivore sequences, but 99.6% of them were from the domestic dog [Bininda-Emonds, 2005]. It follows that the other species were represented by very few genes and sequences. This engenders a high percentage of missing data in the supermatrix with all associated problems (see page 30).

2.3.3 The eternal dilemma: supermatrix or supertree?

The supermatrix and the supertree approaches are classically seen as competitive ways to analyze large data sets. We are convinced that none of the two approaches is significantly better than the other and that an *ad hoc* choice has to be done for each data set, depending on its size, the kind of data and so on. Additionally, these two approaches can be used simultaneously in order to exploit the strengths and to counterbalance the weaknesses of each method [among others Bininda-Emonds, 2004a; Fulton and Strobeck, 2006; Higdon et al., 2007]. In Section 4.4 we present a simultaneous application of the supermatrix and supertree approaches that led us to disentangle the complex phylogeny of Triticeae (Poaceae).

Moreover, both approaches can be combined in a divide-and-conquer strategy [*e.g.*, Bininda-Emonds, 2004b; Huson et al., 1999]. As suggested by Gordon [1986], «*the analysis of large data sets could proceed by division into overlapping subsets which are classified separately and then recombined to provide a single classification*».

Many authors [*e.g.*, Bininda-Emonds et al., 2004, 2003, 2002; Gatesy and Springer, 2004; Huson et al., 1999; Pennisi, 2003; Soltis and Soltis, 2001] share Gordon's feeling and are convinced that any attempt to reconstruct large portions of the Tree of Life requires the use of supertree construction as part of a divide-and-conquer strategy to phylogenetic reconstruction.

In a divide-and-conquer approach a very large phylogenetic problem is decomposed into smaller subproblems, the solutions to which are combined using a supertree approach to derive the global answer (see Daubin et al. [2002] for a practical example).

Subproblems are both faster to analyze and possibly more accurate than the

larger problem [Roshan et al., 2004] because «*they are both smaller (fewer species) and of reduced breadth, allowing more data to be used* »[Bininda-Emonds, 2005].

We also share Gordon's feeling. In the next chapters we present an exhaustive review of supertree methods. We then investigate properties of supertree methods that are appealing in a divide-and-conquer approach to reconstruct the Tree of Life and we present two new supertree methods that reconstruct supertrees satisfying these appealing properties.

Chapter 3

Methods for combining trees

Contents

Systematists have been constructing informal supertrees for many years. Since the last two decades formal definitions of supertrees have been proposed as well as algorithms to solve the associated computational problems.

The supertree approach uses trees as primary source of information. It first involves inferring partially overlapping phylogenetic trees (commonly called *source* trees) from primary data *e.g.*, amino acids, SINEs or morphological traits. Source trees are successively assembled into a larger, more comprehensive supertree [Bininda-Emonds, 2004b]. Such a supertree includes all, or most of, the taxa from the collection of source trees while preserving the phylogenetic information contained in them [Sanderson et al., 1998]. Ideally, supertrees also state relationships among taxa that cannot be observed from any single source tree alone but that can be deduced by combining the information of several source trees.

Supertree methods are also useful, teamed with supermatrix methods, in a divide-and-conquer approach to reconstruct very large phylogenies: first, the set of data is divided into large but tractable subsets that are analyzed individually, then the resulting phylogenies are combined to reconstruct the global phylogeny [Bininda-Edmonds and Stamatakis, 2006; Bininda-Emonds, 2005].
Supertree methods can be classified into three categories, depending on the way

they deal with topological conflicts, *i.e.*, different arrangements of the same taxa among source trees.

The first suite of methods cannot handle incompatible source trees. The pioneering methods that belong to this category are BUILD (Aho et al. [1981], Section 3.3.1.1) and the strict consensus supertree (Gordon [1986], Section 3.3.1.4). Since, as most systematics know, phylogenies usually conflict with one another [Bininda-Emonds, 2004c, p4], those methods are of limited use.

Liberal methods resolve conflicts [Thorley and Wilkinson, 2003], asking source trees to vote and opting for the topological alternative that maximizes an optimization criterion [Baum and Ragan, 2004; Chen et al., 2006; Page, 2002; Semple and Steel, 2000; Snir and Rao, 2006]. The hope is that each taxon is erroneously placed in only few source trees and this erroneous information will be overcome by the large number of source trees where the taxon is correctly placed. Some examples of vote kind methods are Matrix Representation with Parsimony (MRP, Baum [1992]; Ragan [1992], Section 3.3.2.1), Modified-MinCut (MMC, Page [2002], Section 3.3.1.3) and the Average Consensus Supertree (Lapointe and Cucumel [1997], Section 3.3.2.6). Supertrees proposed by liberal methods are often highly resolved and accurate, though several authors have shown that this approach can lead to propose supertrees containing clades that contradict all source trees [Cotton et al., 2006; Goloboff, 2005; Goloboff and Pol, 2002].

In contrast, *veto* methods do not allow the resulting tree to contain clades that contradict source trees. They adopt a veto philosophy: the phylogenetic information of every source topology is to be respected, and the supertree is not allowed to contain clades that a source tree would vote against. These methods remove conflicts [Thorley and Wilkinson, 2003] either proposing multifurcations in the supertree [*e.g.*, Goloboff and Pol, 2002] or pruning rogue taxa [*e.g.*, Berry and Nicolas, 2004, 2007]. Some examples of veto kind methods are extensions of the strict consensus [*e.g.*, Gordon, 1986; Huson et al., 1999], the semi-strict supertree [Goloboff and Pol, 2002, Section 3.3.2.4], SMAST and SMCT [Berry and Nicolas, 2004, 2007] and *PhySIC* (Ranwez et al. [2007a], Section 4.2). *PhySIC_IST* (Scornavacca et al. [2008], Section 4.3) is the unique veto method that allows to reconstruct supertrees with multifurcations that can also lack some taxa of the forest.

Liberal and veto supertree methods can be further divided in *direct* and *indirect* supertree methods. The former supertree methods (*e.g* Modified-MinCut and *PhySIC*) directly combine the input trees while indirect supertree methods (*e.g* MRP and the Average Consensus Supertree) convert input trees into another kind of data (binary sequences, distances) that is then analyzed using a classical phylogenetic tree reconstruction method.

The supertree problem is a generalization of a simpler one, called the *consensus* problem, that consists in summarizing a set of trees that classify the same objects into one tree. Thus, although supertree methods will work in the consensus setting, the reverse does not hold. The consensus problem is a general computational problem in classification [Barthélemy and Guenoche, 1991]. In phylogenomics, consensus

methods are mainly used to:

1. combine several optimal trees for a single data set;
2. combine several trees issued from a bootstrap analysis of a unique data set;
3. combine several trees issued from the analysis of different data sets;
4. compare several trees to assess how much agreement there is among them.

There are numerous methods to combine trees over the same taxa (consensus) or different taxa (supertree) sets. Section 3.1 gives a formal definition of phylogenetic trees and introduces some notations that will be useful later on in this chapter. Section 3.2 presents several consensus methods, some of which have been extended into supertree methods. Finally, Section 3.3 describes various supertree methods.

3.1 Basic concepts

An *unrooted* phylogenetic tree T (see Figure 3.1) consists of *nodes* connected by *branches* (or *vertices* connected by *edges* in a mathematical vocabulary), in which any two nodes are connected by exactly one path and with no node is of degree two. A *rooted* phylogenetic tree (see Figure 3.2) is defined in the same way, except that it has exactly one node, called the *root* of the tree, that can have degree two. Rooted phylogenetic trees can also be defined as directed trees[1] with a unique node with indegree zero called *root*. Terminal nodes, called also *leaf* nodes, are defined in unrooted trees as nodes with degree one and as nodes with outdegree zero in rooted trees. Leaf labels are also called *taxa*. Leaf nodes are labeled, while other nodes, called *internal* nodes, are usually left unlabelled.

In rooted phylogenetic trees each internal node represents the most recent ancestor common of its descendants[2]. An unrooted phylogenetic tree is *binary* if every internal node has degree three. In a binary rooted phylogenetic tree, all internal nodes have degree three, except the root which has degree two.

The set of leaf nodes, resp. internal nodes, of T is denoted by $\boldsymbol{\mathcal{L}(T)}$, resp. $\boldsymbol{\mathcal{I}(T)}$, while $L(T)$ denotes the label set of T. If v is a leaf node, we denote by $\boldsymbol{l_v}$ its label. These labels represent often taxon names but they can also correspond to gene names or other entities of interest. A leaf-labelling of T is a function $\alpha : \mathcal{L}(T) \longrightarrow L(T)$. In this chapter we suppose that α is a bijection but in Chapter 5 we will treat the case of surjective leaf-labellings. A phylogenetic tree is formally a pair (T, α). Informally, we refer to this pair as the phylogenetic tree T. Phylogenetic trees are also called *semi-labelled* trees, since only leaf nodes are labeled [Semple and Steel, 2003]. A collection of trees is also called a *forest*. Given a forest $\mathcal{F}$, $L(\mathcal{F})$ denotes the set of labels appearing in at least one tree of $\mathcal{F}$, that is $L(\mathcal{F}) = \cup_{T \in \mathcal{F}} L(T)$. We define by

[1] When drawing rooted trees the direction of the branches is not indicated explicitly but can be deduced from the placement of the root since all branches are direct away from it.

[2] This means that the root node represents the most recent common ancestor of all the entities at the leaves of the tree.

$\boldsymbol{T_v}$ the subtree of T with v as root. Given a rooted tree T and a leaf set $L \subseteq L(T)$, we say that $u \in T$ is the *least common ancestor* (lca) for the set L if and only if u is the node that is located farthest from the root of T such that $L \subseteq L(T_u)$.

The Newick format

Note that this manuscript we represent trees in Newick format (see `http://evolution.genetics.washington.edu/phylip/newicktree.html`). For a rooted tree T, its Newick format is computed recursively. Let $\mathcal{N}(u)$ denote the Newick format of a node u. If u is a leaf, then $\mathcal{N}(u) = l_u$. If u is an internal node, then $\mathcal{N}(u) = (\mathcal{N}(u_1), \cdots, \mathcal{N}(u_k))$, where $u_{1,\ldots,u_k}$ are the child nodes of u. For an unrooted tree, we first root it on whatever node.

For instance, the Newick format of the rooted tree in Figure 3.2 is $(((a,b),c),((e,f),d))$ while for the unrooted tree in Figure 3.1 it is $((a,b),c,((e,f),d)$.

Bootstrap values, branch lengths and comments can be also integrated in the Newick format.

3.1.1 Splits and clusters

Splits

Given a phylogenetic tree T and $S \subseteq L(T)$, we denote by $T|_S$ the homeomorphic subtree of T induced by the taxa in S. We say that a tree T *refines* a tree T' if and only if T' can be obtained by collapsing branches in T, *i.e.*, deleting some branches of T and identifying their endpoints. A tree T is said to *display* a tree T' if and only if the tree $T|_{L(T')}$ refines T'. Given a label set L, a split $A|B$ on the label set L is a partition of L into two non-empty sets. A phylogenetic tree T induces a set of splits $\mathcal{S}(T)$ since each branch $x \in T$ leads to a split on L (see figure 3.1). More precisely, the split associated to a branch (u,v) of T is $L(T_u)|L(T_v)$, where T_u and T_v are the two rooted trees obtained from T when removing (u,v). A split $A|B$ is *trivial* if $|A| = 1$ or $|B| = 1$.

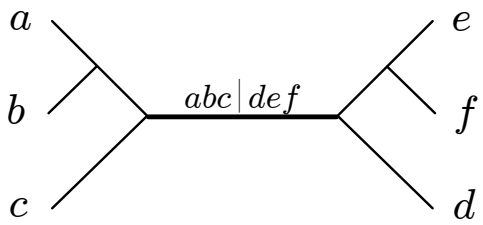


Figure 3.1: **Example of unrooted phylogenetic tree** T - The set of splits $\mathcal{S}(T)$ contains six trivial splits: $a|bcdef$, $b|acdef$, $c|abdef$, $d|abcef$, $e|abcdf$, $f|abcde$ and three non-trivial splits: $ab|cdef$, $abc|def$, $ef|abcd$.

Given a collection of splits $\mathcal{S}$, we say that $\mathcal{S}$ is compatible if there exists an

unrooted tree T such that every split in $\mathcal{S}$ is a split of T, *i.e.*, $\mathcal{S} \subseteq \mathcal{S}(T)$. Two splits $A_1|B_1$ and $A_2|B_2$ are compatible if at least one of the sets $A_1 \cap A_2$, $A_1 \cap B_2$, $B_1 \cap A_2$ or $B_1 \cap B_2$ is the empty set [Buneman, 1971]. The compatibility of splits is an easy problem to solve since Buneman [1971] proved that a collection of splits $\mathcal{S}$ is compatible if and only if all splits are pairwise compatible.

Clusters

Given a label set L, a *group* is a subset of L. Given a rooted phylogenetic tree T, a group G is said *monophyletic* on T if and only if T contains a node v such that $L(T_v) = G$.

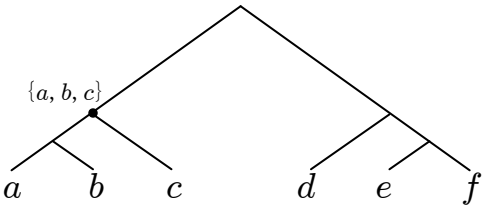


Figure 3.2: **Example of rooted phylogenetic tree** T - The set of clusters $\mathcal{C}(T)$ contains seven trivial clusters: $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$, $\{e\}$, $\{f\}$, $\{a, b, c, d, e, f\}$ and four non-trivial clusters: $\{a, b\}$, $\{a, b, c\}$, $\{e, f\}$, $\{d, e, f\}$.

The monophyletic groups of a tree T are called *clusters* of T (see Figure 3.2). A rooted phylogenetic tree T induces a set of clusters $\mathcal{C}(T)$ since each node $v \in T$ corresponds to a cluster on L. A cluster C is trivial if $|C| = 1$ or $|C| = |L(T)|$. The number of non-trivial clusters of a rooted phylogenetic tree then equals its number of internal nodes. Note that, if T and T' are two rooted phylogenetic trees on the same leaf set such that T refines T', then T contains all clusters of T'. Given a collection of groups $\mathcal{C}$, we say that $\mathcal{C}$ is compatible if there exists a rooted tree T such that every group in $\mathcal{C}$ is a cluster of T, *i.e.*, $\mathcal{C} \subseteq \mathcal{C}(T)$. Two groups C_1 and C_2 such that either C_1 is contained in C_2, or C_2 is contained in C_1, or C_1 and C_2 are disjoint are called *compatible*. A collection of groups $\mathcal{C}$ is compatible if and only if its groups are pairwise compatible.

Several algorithms have been proposed to reconstruct the tree displaying a set of compatible splits or clusters [*e.g.*, Gusfield, 1991; Meacham, 1983].

3.1.2 Quartets and triplets

Quartets

For a set of four leaves $\{a, b, c, d\}$ in $L(T)$ there exist only three unrooted binary trees, called *quartets* and denoted by $ab|cd$, resp. $ac|bd$, resp. $bc|ad$, depending on how the central edge splits the four species. We say that T *induces* or *displays* the quartet $ab|cd$ if $T|_{\{a,b,c,d\}}$=((a,b),(c,d)). For instance, the tree T_1 depicted in Figure 3.3 induces, among others, the quartet $ab|cd$. If a tree does not induce any quartet

for $\{a, b, c, d\}$, we say that $\{a, b, c, d\}$ is unresolved in T. Any unrooted tree T can be equivalently described by the set of quartets that it induces [Bandelt and Dress, 1986]. In other words, this set, denoted by $\mathcal{Q}(T)$ (see Figure 3.3 for an example), suffices to reconstruct T.

Figure 3.3: **Examples of a set of triplets and a set of quartets induced by two trees** - The set of quartets $\mathcal{Q}(T_1)$ contains five quartets *i.e.*, $ab|cd$, $ab|ce$, $ab|de$, $ce|ad$ and $ce|bd$. The set of triplet $\mathcal{R}(T_2)$ contains four triplets *i.e.*, $ab|c$, $ab|d$, $cd|a$ and $cd|b$.

Given a collection $\mathcal{F}$ of unrooted phylogenetic trees, $\mathcal{Q}(\mathcal{F})$ denotes the set of quartets present in these phylogenetic trees, *i.e.*, $\mathcal{Q}(\mathcal{F}) = \bigcup_{T_i \in \mathcal{F}} \mathcal{Q}(T_i)$. A set $\mathcal{Q}$ of quartets is compatible if there is a tree T that displays all quartets in $\mathcal{Q}$.

If a quartet set $\mathcal{Q}$ on a label set L is complete *i.e.*, if $\mathcal{Q}$ contains at least one resolution for every set of four labels of L, then the compatibility of $\mathcal{Q}$ can be easily decided [Bandelt and Dress, 1986, Proposition 2]. If $\mathcal{Q}$ is not complete, quartet compatibility is an NP-complete problem [Steel, 1992, Theorem 1]. Given a compatible forest of unrooted trees $\mathcal{F}$, we say that T is a *parent tree* for $\mathcal{F}$ if and only if $\mathcal{Q}(\mathcal{F}) \subseteq \mathcal{Q}(T)$. To each split $A|B$ of an unrooted tree T we can associate a set of quartets $q(A|B)$ defined as follows:

$$q(A|B) = \{aa'|bb' : a, a' \in A, b, b' \in B\}.$$

Then, we can define the quartet set of a tree T from its set of splits since $\mathcal{Q}(T) = \bigcup_{A|B \in \mathcal{S}(T)} q(A|B)$.

Triplets

In the same way, we can define the set of triplets induced by a rooted phylogenetic tree. Given a rooted tree T, for a set of three labels or equivalently leaves $\{a, b, c\}$ in $L(T)$ we denote by $T|_{\{a,b,c\}}$ the homeomorphic subtree of T induced by the leaves labeled by a, b, and c. If T is binary, $T|_{\{a,b,c\}}$ can be any of the three possible rooted binary trees on $\{a, b, c\}$. These binary trees on $\{a, b, c\}$ are called triplets and are denoted by $ab|c$, resp. $ac|b$, resp. $bc|a$, depending on the unique non-trivial cluster in $T|_{\{a,b,c\}}$ ($\{a, b\}$, resp. $\{a, c\}$, resp. $\{b, c\}$). We say that T *induces* or *displays* the triplet $ab|c$ if $T|_{\{a,b,c\}}$=((a,b),c). For instance, the tree T_2 depicted in Figure 3.3 induces, among others, the triplet $ab|d$. If T is not binary it may happen that $T|_{\{a,b,c\}}$ contains only the trivial cluster $\{a, b, c\}$. In this case we say that $\{a, b, c\}$ is unresolved in T and denote $T|_{\{a,b,c\}}$ by the trichotomy (a, b, c). Given a triplet t, $\bar{t}$ denotes any of the two other triplets on the same set of leaves.

Any rooted tree T can be equivalently described by the set of triplets homeomorphic to its subtrees connecting three leaves (see among others Grunewald et al. [2007]). This triplet set is denoted by $\mathcal{R}(T)$ (see Figure 3.3 for an example). Given a collection $\mathcal{F}$ of rooted phylogenetic trees, $\mathcal{R}(\mathcal{F})$ denotes the set of triplets present in at least one tree of $\mathcal{F}$, *i.e.*, $\mathcal{R}(\mathcal{F}) = \bigcup_{T_i \in \mathcal{F}} \mathcal{R}(T_i)$. A set $\mathcal{R}$ of triplets is compatible if and only if there is a tree T that displays all triplets in $\mathcal{R}$. The compatibility of a set of triplets can be decided in polynomial time [Aho et al., 1981]. Given a compatible forest of rooted trees $\mathcal{F}$, we say that T is a *parent tree* for $\mathcal{F}$ if and only if $\mathcal{R}(\mathcal{F}) \subseteq \mathcal{R}(T)$.

3.1.3 Interpretations of polytomies

In a phylogenetic tree, nodes with more than two children are called *polytomies*. Polytomies can be interpreted in different ways.

First, a polytomy can represent a common ancestral population splitting through speciation into multiple lineages. In this case, the polytomy is usually said to be *hard*.

Second, polytomies can represent an uncertainty for which resolution of the node's child subtrees or lineages is the best hypothesis. In this case polytomies are said to be *soft*. A soft polytomy can have two distinct interpretations, differing in the set of admissible binary phylogenies it encompasses. Consider a soft polytomy with three child nodes forming three clusters S_1, S_2 and S_3.

The most widespread meaning of a soft polytomy accepts any fully-resolved tree on S_1, S_2, S_3 that keeps them separated: $((S_1, S_2), S_3)$, $((S_1, S_3), S_2)$ or $((S_2, S_3), S_1)$. Most of the methods that we present in this chapter interpret polytomies in this way.

A second interpretation of soft polytomies was introduced by the Adams consensus [Adams, 1972, Section 3.2.2.1] and is also intended by MC [Semple and Steel, 2000, Section 3.3.1.2] and MMC [Page, 2002, Section 3.3.1.3]. This interpretation accepts as possible phylogeny any fully-resolved tree that maintains the structure of each subtree respectively, no matter whether or not S_1, S_2 and S_3 are kept separate or are interleaved. In this case, we say that the polytomy is an *Adams* polytomy. Under this interpretation, a soft polytomy represents a much wider range of fully-resolved phylogenies than with the first interpretation, and its meaning is thus harder to grasp.

For instance, for the tree $(((a, b), c), d, e)$, if soft polytomies are interpreted in the common way, we have three admissible binary phylogenies, *i.e.*, $\{((((a, b), c), d), e)$, $((((a, b), c), e), d)$, $(((a, b), c), (d, e))\}$. If the polytomy is interpreted as an Adams one the set of admissible binary phylogenies is comprised of 35 binary trees, *e.g*, $\{((((a, b), d), c), e)$, $\{((((a, e), b), c), d)$, etc., on the 105 possible binary trees on five taxa. This explains why the first interpretation of soft polytomies prevails in phylogenetics.

In this manuscript polytomies are interpreted as soft. Since a tree cannot be interpreted without knowledge of how the method that is used to produce it interprets

polytomies, in this chapter we will mention when methods interpret soft polytomies as Adams polytomies and not in the common way. In the next section we present a review of the most used consensus methods.

3.2 Consensus methods for phylogenetic trees

As stated at the beginning of this chapter, a fundamental problem in classification of biological data is the question of how to combine the information contained in a set of trees that classify the same objects into one tree. Recall that the *consensus tree problem* requires that input trees have identical sets of taxa. The use of consensus methods to summarize several trees issued from a unique data set or to compare trees is widely accepted. More controversial [Barrett et al., 1991] is the use of such methods for the combination of trees issued from different data sets, *i.e.*, as a tool for new phylogenetic inferences, since the construction of most consensus trees is guided by the comparison and the combination of tree topologies, rather than phylogenetic inference criteria. In this section we present the most used consensus methods, with the pros and cons of each of them.

The first methods presented below (sections 3.2.1.1-3.2.1.6), except the asymmetric median tree in Section 3.2.1.4, are all defined for both unrooted and rooted forests. For the sake of simplicity they are only described here in the unrooted setting but all of them can be applied to rooted forests (*e.g.*, replacing, in the definitions, splits with clusters).

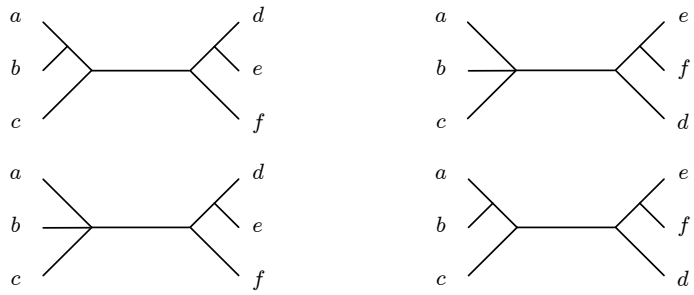


Figure 3.4: **Example of a forest of unrooted phylogenetic trees $\mathcal{F}$** - The trees in this forest are used to illustrate the five consensus methods presented in sections 3.2.1.1-3.2.1.6.

3.2.1 Consensus methods defined for both rooted and unrooted forests

3.2.1.1 Strict consensus tree

The *strict consensus tree* [McMorris et al., 1983; Sokal and Rohlf, 1981] of a collection $\mathcal{F}$ of unrooted trees is the tree that contains exactly the splits shared by all input

trees (see Figure 3.5 for an example) *i.e.*, the tree T such that:

$$\mathcal{S}(T) = \bigcap_{T_i \in \mathcal{F}} \mathcal{S}(T_i).$$

The main advantage of the strict consensus is the simplicity of interpretation: the splits that appear in all the input trees can be considered as reliable. Though strict consensus trees were called *Nelson trees* in Schuh and Farris [1981], Page [1989] demonstrated that these two methods are not equivalent (see Section 3.2.1.6 for a description of Nelson trees).

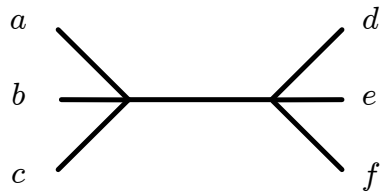


Figure 3.5: **Example of strict consensus tree for the forest depicted in Figure 3.4** - For this forest the strict and the majority-rule consensus trees are identical.

The strict consensus tree tends to display numerous *polytomies* [Funk and Brooks, 1990; Wilkinson, 1996]. This behavior is sometimes due to incongruence among the source trees and sometimes to undesirable properties of this consensus method [Wilkinson, 1995]. Wilkinson and Thorley [2001] proposed a measure of *Consensus Efficiency* (CE) that can help to understand whether the lack of resolution of the strict consensus tree is due to a strong disagreement between input trees or not. The CE measure can be used to evaluate the efficiency of all strict consensus methods sensu Wilkinson [1994] *i.e.*, methods that retain unanimous agreement among the source trees.

The use of the strict consensus method to combine trees issued from different data sets has been criticized by the promoters of the parsimony approach because the returned tree can be less parsimonious than that obtained by an MP analysis on the concatenation of all data sets (see the MRP method in Section 3.3.2.1). Other criticisms come from the advocates of the total evidence approach (Chapter 2) since the strict-consensus tree can be incompatible with the total evidence tree [Barrett et al., 1991]. The latter remark is valid for all consensus methods, since all consensus trees refine the strict consensus tree [see Bryant, 2003].

3.2.1.2 Majority-rule consensus tree

The *majority-rule consensus tree* of a collection $\mathcal{F}$ of unrooted trees is the tree that contains exactly the splits shared by strictly more than 50% of input trees [Barthélemy and McMorris, 1986; Margush and Mcmorris, 1981]. See Figure 3.5 for an example. The 50% rule ensures that all retained splits are compatible since each

pair of splits appears simultaneously in at least one tree. Given two trees T_1 and T_2, the symmetric distance between T_1 and T_2, denoted by $d_S(T_1, T_2)$, is defined as the number of splits appearing in one tree but not the other [Robinson and Foulds, 1981]. Barthélemy and McMorris [1986] proved that the majority-rule tree T for a forest $\mathcal{F}$ minimizes:

$$d_S(T, \mathcal{F}) = \sum_{T_i \in \mathcal{F}} d_S(T, T_i) \tag{3.1}$$

Hence, the majority-rule tree is also a *median* tree with respect to the symmetric distance metric. Several supertree methods are also based on a median tree approach (see Section 3.3.3). Note that the majority-rule consensus tree is not necessarily the unique median tree. More precisely, Dong and Fernandez-Baca [2009] have recently shown that majority-rule consensus is the strict consensus of all median trees.

The majority-rule tree is often used to summarize bootstrap trees. Sharkey and Leathers [2001] criticize the use of this consensus method to combine several optimal trees for a single data set, claiming that majority-rule consensus tends to equate reliability with ambiguity. Indeed, ambiguity in the data set can cause an ambiguous topology, *i.e.*, a topology displaying several polytomies, to be repeated among the input trees and therefore preferred by this method.

3.2.1.3 Semi-strict consensus tree

When some input trees are not binary, splits that are never contradicted may occur in some of the trees but not be retained by the two previously described consensus methods. For example, consider a collection $\mathcal{F} = \{((a, b), (c, d)), (a, b, c, d), (a, b, c, d), (a, b, c, d)\}$. In this case, the split $ab|cd$ would not be retained neither in the strict consensus tree nor in the majority-rule consensus tree, even though this information is present and not contradicted by any tree (for another example see Figures 3.5 and 3.6). However, this split is retained in the semi-strict consensus tree, defined as follows: the *semi-strict consensus tree*, or *combinable component tree* [Bremer, 1990], of an unrooted tree collection $\mathcal{F}$ is the tree that contains exactly those splits of $\mathcal{S}(\mathcal{F})$ compatible with every tree in $\mathcal{F}$. This consensus method has been criticized (among others by De Queiroz [1993]) for

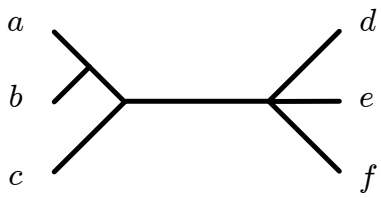


Figure 3.6: **Example of semi-strict consensus tree for the forest depicted in Figure 3.4** - The semi-strict consensus trees contains the split ab|cdef which, although not contradicted by any tree, is not included neither in the strict consensus tree nor in the majority-rule consensus tree. For this forest the semi-strict and the Nelson-Page consensus trees (Section 3.2.1.6) are identical.

the fact that the resulting tree can contain splits that appear in only one of the input trees. Some authors consider the information contained only in one tree as unreliable but, as Bryant [2003] has pointed out, it is not likely for a split to be compatible with a random tree, so we can reasonably rely on this information.

Note that the semi-strict consensus tree refines the strict consensus tree and that they are equal when all source trees are binary.

3.2.1.4 Asymmetric median tree (defined only for unrooted trees)

Given two unrooted trees T_1 and T_2, we define the *asymmetric distance* between T_1 and T_2, denoted by $d_A(T_1, T_2)$, as the number of splits appearing in T_2 but not in T_1. The *asymmetric median tree*, or AMT [Phillips and Warnow, 1996] for a forest of unrooted trees $\mathcal{F}$ is the tree minimizing:

$$d_A(T, \mathcal{F}) = \sum_{T_i \in \mathcal{F}} d_A(T, T_i).$$

Since the AMT problem for k trees is equivalent to the maximum independent set problem on k-colored graphs [Phillips and Warnow, 1996], the former problem is NP-hard for more than two trees. Note that this definition can be easily extended to the supertree context.

The next two consensus methods are related to the notion of AMT.

3.2.1.5 Greedy consensus tree

The strategy for constructing a *greedy consensus tree*, also called *majority-rule extended tree* [Felsenstein, 2005], consists in building up from the empty set a collection of compatible splits $\mathcal{S}$ by considering splits one at time in decreasing order of frequency and adding them to $\mathcal{S}$ if they are pairwise compatible with all splits previously added to this set. The greedy consensus tree of $\mathcal{F}$ is the tree that contains exactly the splits in $\mathcal{S}$. Note that this can be seen as a greedy heuristics to find the AMT [Bryant, 2003]. Greedy consensus trees, as semi-strict consensus trees, can contain splits appearing in only one of the input trees. Since all bipartitions with frequency greater than $|\mathcal{F}|/2$ are compatible, a greedy consensus tree always refines the majority-rule consensus tree. The main problem with this greedy ap-

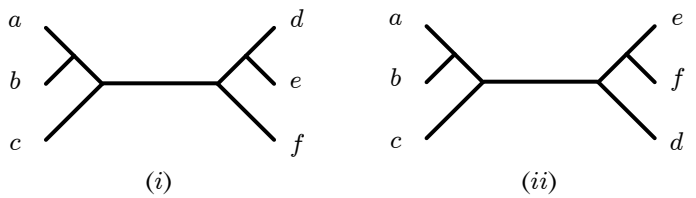


Figure 3.7: **Example of greedy consensus trees for the forest depicted in Figure 3.4** - The tree (i) is obtained if the split de|abcf is preferred to the split ef|abcd, otherwise we obtain the tree (ii).

proach is that when two or more splits appear with the same frequency $< 50\%$, they can be incompatible and one is arbitrarily chosen, eventually preventing the insertion of other splits that are incompatible with it. This arbitrary decision may potentially give rise to different greedy trees (see Figure 3.7 for an example). Berger-Wolf [2004] presented an algorithm returning all possible greedy refinements of the majority-rule consensus tree in $O(m|L(\mathcal{F})|)$ time, where m is the number of possible greedy consensus trees.

3.2.1.6 Nelson and Nelson-Page consensus trees

The *Nelson consensus tree* [Nelson, 1979] of a collection $\mathcal{F}$ of unrooted trees is the tree, if it exists, that contains exactly the splits found in at least two trees (called *replicated components* or *replicated splits*) plus all other *unreplicated* splits compatible with all replicated splits. This definition is founded on the assumption that information appearing in two or more trees is highly likely to be reliable.

When exactly two trees are compared, the Nelson consensus tree is equivalent to the semi-strict consensus tree [Bremer, 1990]. In the literature the strict consensus tree has often been confused with the Nelson consensus tree and most Nelson consensus trees published in the past are in fact strict consensus trees. According to Page [1989], the Nelson consensus tree is equivalent to the strict consensus tree if $\mathcal{F}$ contains only two trees. Swofford [1991] proved that this is true only if the two trees are both binary. Indeed, in such a case any unreplicated split will be incompatible with at least one split appearing in the other tree.

The main problem with Nelson's definition is that if the set of replicated splits is not compatible the method cannot return a tree [among others McMorris et al., 1983; Page, 1990]. Moreover, even for compatible forests, Nelson's definition is ambiguous. Indeed, if there are several distinct groups of unreplicated splits that are compatible with the replicated splits but mutually incompatible, we need to choose arbitrarily one of these groups, since Nelson gives no indication on how to break ties. As in the greedy consensus method, this can potentially lead to propose different consensus trees.

Page [1990] addressed these problems several years later by proposing what is now known as the *Nelson-Page consensus tree*. Page calls *cliques* of compatible splits the sets of splits such that every splits in the set is compatible with every other splits in the set. Each split appearing in the original trees is assigned a weight equal to its frequency in $\mathcal{F}$ minus one and each clique of compatible splits is assigned a score equal to the sum of the weights of its splits. Note that unreplicated splits, having a weight of zero, do not contribute to the clique score[3]. If there is a single clique with the highest score, its splits are used to construct the Nelson-Page consensus tree. In case of several maximum weight compatible cliques then the splits included in the Nelson-Page consensus tree are those common to all maximum weight compatible cliques. Splits found in some but not all of the highest score cliques are classified as *ambiguous*. Nelson consensus and Nelson-Page consensus return the same tree

[3]That is why the weight of a split is set to its frequency in the forest minus one.

if the set of replicated splits is compatible and no unreplicated split is compatible with this set. For the forest depicted in Figure 3.4 on page 44, the Nelson consensus tree is not defined since the set of replicate splits is not compatible (as it contains both $de|abcf$ and $ef|abcd$), while the Nelson-Page consensus tree coincides with the semi-strict consensus tree (see Figure 3.6). No unreplicated split is included in the Nelson-Page consensus tree but the algorithm can be easily adapted to consider also these splits [Swofford, 1991].

One drawback of this approach is that finding a maximum compatible subset of characters (in this case the maximum weight compatible cliques) is an NP-hard problem [Day and Sankoff, 1986].

For unrooted forest, if unreplicated splits are allowed to contribute to the score[4], the Nelson-Page tree is also a median tree with respect to the asymmetric distance metric [Bryant, 2003].

3.2.1.7 The MAST trees

The MAST (*Maximum Agreement SubTree*) problem has been introduced in phylogenetics by Finden and Gordon [1985]. It consists in finding the maximum agreement subset tree for a forest $\mathcal{F}$.

Definition 3.2.1 *Given a forest of trees $\mathcal{F}$, an agreement subtree T is a tree such that $L(T) \subseteq L(\mathcal{F})$ and $T = T_i|_{L(T)}$ $\forall T_i \in \mathcal{F}$.*

The maximum agreement subtree for a forest $\mathcal{F}$ is an agreement subtree with the maximum number of leaves *i.e.*, T_M is a MAST for $\mathcal{F}$ if and only if $|L(T_M)| = max(|L(T_j)|)$ $\forall T_j \in \mathcal{F}_{AG}$, where $\mathcal{F}_{AG}$ is the set of agreement subtrees for $\mathcal{F}$. Note that the number of MAST for a given forest can be exponential although $|L(T_M)|$ is unique. Finden and Gordon [1985] proposed a heuristic approach to the problem.

The first exact polynomial algorithm for forests of only two trees was proposed by Steel and Warnow. Then, numerous algorithms have been proposed. Currently, we dispose of:

- an $O(\sqrt{d}n\log(n)$ algorithm [Przytycka, 1997] and an $O(\sqrt{d}n\log^2(\frac{n}{d})$ algorithm [Kao et al., 2001] for two rooted trees, where $n = |L(\mathcal{F})|$ and d is the maximum degree of the input tree nodes;
- an $O(n^{1.5})$ algorithm for two unrooted trees [Kao et al., 1999], where $n = |L(\mathcal{F})|$.

This problem has been proved NP-hard for more than two trees but can be solved in polynomial time for forests with bounded degree [among others, Amir and Keselman, 1997; Bryant, 1997; Farach et al., 1995; Guillemot and Nicolas, 2006]. Moreover, an FPT algorithm has been proposed for unbounded degree forests [Berry and Nicolas, 2006]. The MAST is particularly useful when only a few taxa are responsible for the incongruence among the input trees, providing a way of identifying rogue

[4]We just need to assign to each split in an unrooted forest a weight equal to its frequency in $\mathcal{F}$.

taxa. For instance, the trees $(((a,b),c),d,e)$ and $((a,c),(b,d),e)$ are homeomorphic if taxon b is pruned from both trees so their MAST is $((a,c),d,e)$. Other methods that are useful to detect rogue taxa are the Adams consensus (see Section 3.2.2.1), in the consensus setting and the SMAST (Section 3.3.4.1) and PhySIC_IST method (Section 4.3), in the supertree setting.

A variant of the MAST consists in finding the maximum compatible subtree (MCT) for a forest $\mathcal{F}$ *i.e.*, a tree T such that T refines all trees $T_i|_{L(T)}$ $\forall T_i \in \mathcal{F}$ and has the maximum number of leaves. This problem has been shown NP-hard on two rooted trees if one of them is of unbounded degree [Hein et al., 1996]. A $O(n^{2^{d+1}}+kn3)$ time algorithm has been recently proposed by Guillemot and Nicolas [2006]. Moreover, an FPT algorithm has been proposed for unbounded degree forests [Berry and Nicolas, 2006]. As the MAST, the MCT tree may not be unique.

Both MAST and MCT problems have been adapted to the supertree setting (see Section 3.3.4.1).

3.2.2 Consensus methods defined only for rooted forests

We now focus on consensus methods defined only for rooted forests.

3.2.2.1 Adams consensus tree

Adams [1972] presented «*a new problem in the science of classification... along with its solution* ». The *Adams consensus* is the first consensus method ever proposed. There are two versions of this method, one for fully-labeled trees and one for semi-labeled ones. As previously mentioned, here we focus on the latter kind of trees. Describing his method, Adams claimed that the consensus tree of two or more trees has to contain only the information shared by all trees and that information not represented in all trees should not be represented in the final consensus tree. Though this sounds as restrictive as the strict consensus definition, we will see that the Adams consensus often preserves more structures than the strict one. The Adams consensus is based on the idea that a tree should be thought of as a «*set of leaf subset nestings*» rather than as a «*set of clusters*». A group of a taxa A *nests* within a larger group B if A is included in B, *i.e.*, if the least common ancestor (lca) of all elements of A is a descendant of the lca of all element of B. Since based on ancestor-descendant relationships, this method can only be used for rooted tree forests.

Before describing the algorithm we need to introduce two more definitions: the product of partitions and the maximal cluster partition for a tree. Given a set of taxa L and k partitions C_1, C_2, ..., C_k of L, the *product* of these partitions is the partition where two taxa a and b are in the same block if and only if they are in the same block for each C_i. This product is denoted by $\prod_{T_i \in \mathcal{F}} C(T_i)$. For example, the product of $abc|de$ and $ad|bce$ is $a|bc|d|e$. Now, the *maximal cluster partition* for a rooted tree T is the partition $C_M(T)$ of $L(T)$ whose blocks correspond to the maximal clusters of T, *i.e.*, the largest non trivial clusters in T. For instance, for

the tree T_1 is Figure 3.8, the maximal clusters of T are (a, b, c, d, e) and (f, g), so its maximal cluster partition is $abcde|fg$.

Algorithm 1: AdamsTree($\mathcal{F}$) (adapted from [Bryant, 2003])

Data: A rooted tree forest $\mathcal{F} = \{T_1, T_2, \cdots, T_k\}$.
Result: The Adams consensus for $\mathcal{F}$.

```
1 if ( T_1 contains only one leaf ) then
2     return T_1; //note that |L(T_1)| = |L(T_2)| = ··· = |L(T_k)|
3 else
4     T^A is a tree composed by a new node r;
      C_M(F) ← ∏_{T_i ∈ F} C_M(T_i);
5
6     foreach block B of C_M(F) do
7         T^A_B ← AdamsTree(T_1|_B, T_2|_B, ··· , T_k|_B);
8         add the root node of T^A_B as son of r in T^A;
9     return T^A ;
```

The Adams consensus tree for a forest $\mathcal{F}$ is calculated recursively computing at each step the maximal cluster partitions $C_M(T_i)$ for all trees T_i in $\mathcal{F}$ and then calculating the product $C_M(\mathcal{F})$ of these partitions, *i.e.*, $C_M(\mathcal{F}) = \prod_{T_i \in \mathcal{F}} C_M(T_i)$. The Adams consensus tree T^A for $\mathcal{F}$ is composed at the beginning of only one node. For each block B in $C_M(\mathcal{F})$, the Adams consensus tree of the restriction of $\mathcal{F}$ to B is calculated (recursively) and the root node of the resulting tree is added as son of T^A. The recursion stops when the forest given as input consists of trees with only one node (see Algorithm 1). An example of Adams consensus tree computation is given in Figure 3.8. An advantage of this method is that it often preserves more structure than the strict consensus method. A drawback is that the Adams consensus tree may contain clusters that do not occur in any of the input trees [Rohlf, 1982; Sokal and Rohlf, 1981], which makes its interpretation difficult. For example, the Adams consensus tree for $(a, (((b, e), c), d))$ and $(a, (((b, d), c), e))$ is $(a, ((b, c), d, e))$. The cluster (b, c) in the Adams consensus tree, not present in any input tree, means only that b and c are more closely related to each other than either is to a, d, or e. In this case the strict consensus tree would be completely unresolved. McMorris et al. [1983] argued that Adams' method lacks a compelling justification and its popularity is primarily a consequence of its historical precedence. In response to criticism, Adams [Adams, 1986] showed that the Adams consensus tree is the unique tree T^A that satisfies the following two conditions:

(i) if a group of taxa X nests a group Y in all input trees, then X nests Y in T^A;

(ii) given a couple of clusters X, Y of T^A such that $X \subseteq Y$, then X nests in Y in every input tree.

A consequence of these nesting properties is that the Adams consensus tree T^A also

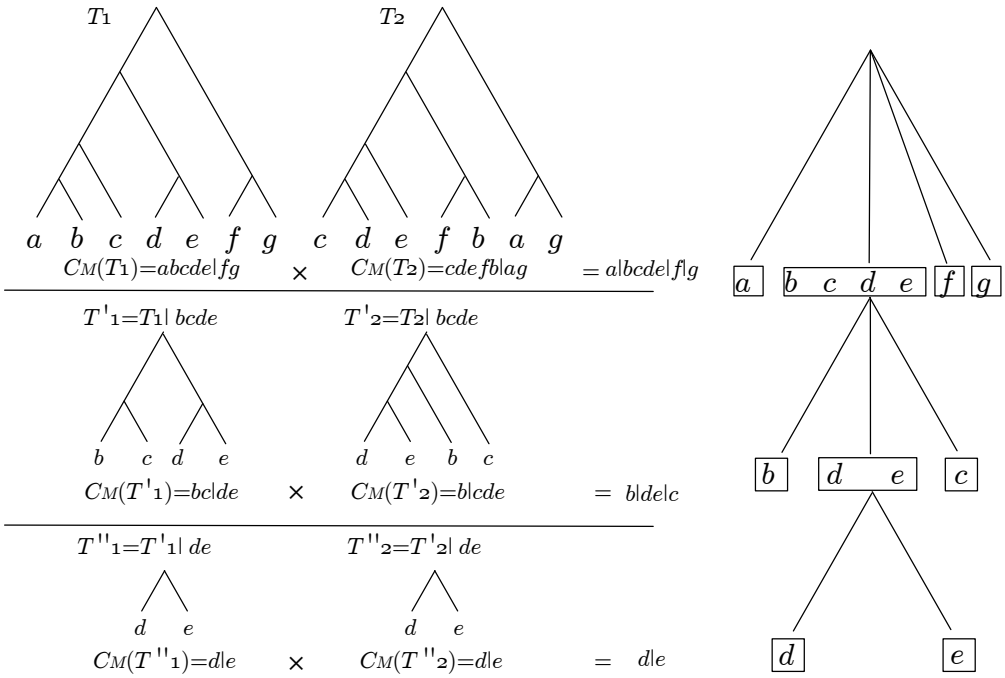


Figure 3.8: **Example of Adams consensus tree (Section 3.2.2.1) for a forest comprised of two trees T_1 and T_2** - The computation of the Adams consensus tree for this forest requires 3 recursive steps.

preserves the triplet information shared by all input trees, without introducing new triplets with respect to the input trees [Bryant, 2003] *i.e.*,

$$\bigcap_{T_i \in \mathcal{F}} \mathcal{R}(T_i) \subseteq \mathcal{R}(T^A) \subseteq \bigcup_{T_i \in \mathcal{F}} \mathcal{R}(T_i).$$

This type of consensus tree is useful for identifying *rogue* taxa, *i.e.*, taxa whose position greatly differs from one input tree to another. For example, the rooted trees $T_1 = ((((((a,b),c),d),e),f),g)$ and $T_2 = ((((((a,g),c),d),e),f),b)$ have the same shape (*i.e.*, they are equivalent if leaf labels are not taken into account) but differ in the positions of taxa b and g. The Adams consensus tree puts these taxa at the most inclusive position that each occupies in any of the input trees. Since each of the taxa was positioned at the basis of T_1 or T_2, both are moved to the basis of the Adams consensus tree *i.e.*, $T^A = (((((a,c),d),e),f),b,g)$. Note that the Adams consensus method interprets polytomies as Adams polytomies (see Section 3.1.3).

Several properties of the MinCut supertree are defined with respect to the Adams consensus (see Section 3.3.1.2 for more details).

Some recent methods are also useful for identifying rogue taxa both in the consensus setting *i.e.*, the afore-descrived MAST method (see Section 3.2.1.7) and in the supertree setting *i.e.*, the SMAST and PhySIC_IST methods, respectively presented in sections 3.3.4.1 and 4.3. For the afore-described forest, those methods

propose the tree $(f,(e,(d,(a,c))))$, containing neither b nor g.

3.2.2.2 Local consensus trees

Kannan et al. [1998] proposed a set of methods that aim to construct consensus trees containing a maximum number of triplets considered as reliable and a minimum number of triplets considered as unreliable.

The construction of local consensus trees [Kannan et al., 1998] is based on the set of rooted triplets $\mathcal{R}(\mathcal{F})$ (see Section 3.1). Recall that, given a triplet t, $\bar{t}$ denotes any of the two other triplets on the same set of leaves. Kannan et al. [1998] distinguish three categories of triplets in $\mathcal{R}(\mathcal{F})$:

1. *constant triplets*: the triplets common to all $T_i \in T$ *i.e.*, $\{t | t \in \bigcap_{T_i \in \mathcal{F}} \mathcal{R}(T_i)\}$;
2. *compatible triplets*: the triplets for which $\mathcal{F}$ contains just one resolution or a trichotomy *i.e.*, $\{t | \bar{t} \notin \mathcal{R}(\mathcal{F})\}$;
3. *incompatible triplets*: triplets for which $\mathcal{F}$ contains several resolutions *i.e.*, $\{t | \bar{t} \in \mathcal{R}(\mathcal{F})\}$.

Kannan et al. [1998] focused on the first two sets[5] and they defined the three consensus methods[6] described in this section. Note that these methods are defined in the Kannan et al.'s paper only for collections of two trees but we can easily extend their definitions to forests of more than two trees.

Definition 3.2.2 *(adapted from Kannan et al. [1998]). A rooted tree T is an RV-I of a forest of rooted trees $\mathcal{F}$ if T leaves unresolved all triples $\{a,b,c\} \in L(\mathcal{F})$ on which the trees in $\mathcal{F}$ disagree or which are unresolved in all trees of $\mathcal{F}$ and preserves a maximum number of constant triplets.*

The authors prove that the RV-I tree always exists, is unique and, for a forest of two trees, coincides with the strict consensus tree.

Definition 3.2.3 *(adapted from Kannan et al. [1998]). A rooted tree T is an RV-II of a forest of rooted trees $\mathcal{F}$ if T preserves the topology of all constant triplets and leaves unresolved a maximal set of the other triplets,* i.e.*, those on which the trees in $\mathcal{F}$ disagree or which are unresolved in all trees of $\mathcal{F}$.*

Let $\mathcal{R}_{CT}(\mathcal{F})$ denote the set of Constant Triplets for a forest $\mathcal{F}$. Kannan et al. [1998] affirm that the RV-II for a tree forest can be computed by the Build algorithm [Aho et al., 1981, see Section 3.3.1.1] inputed with the triplet set $\mathcal{R}_{CT}(\mathcal{F})$. For a set of triplets $\mathcal{R}$, the Build algorithm indicates whether $\mathcal{R}$ is compatible and, in case of a positive answer returns a tree T s.t. (i) $\mathcal{R} \subseteq R(T)$ and (ii) no internal edge

[5] Note that, if $\mathcal{F}$ contains only binary trees these sets coincide.

[6] In the same paper Kannan et al. [1998] described two other consensus methods *i.e.*, the optimistic local consensus (OLC) and the pessimistic local consensus (PLC), that are not defined for all sets of trees.

of T can be contracted so that the resulting tree also displays $\mathcal{R}$. However, several trees having those two properties may exist [Semple, 2003] and display a different number of triplets. As far as we know, it has not been demonstrated whether or not Build returns a tree with a minimum value of $|R(T)|$ so it is doubtful that the tree RV-II tree for a forest can be computed by this algorithm. Since all proofs and algorithms presented in Kannan et al. [1998] are based on the Build algorithm, we redefine the RV-II as the tree T computed from $\mathcal{R}_{CT}(\mathcal{F})$ by the Build algorithm *i.e.*, the local consensus tree under the terminology of Bryant [2003]. Note that this definition, unlike its original formulation, does not require $|R(T)|$ to be minimum.

Bryant [2003] argues that «*the Adams tree is neither equal to the local consensus tree nor is it an RV-II tree. For example, the local consensus tree of* $((a,b,c),d)$ *and* $(a,(b,c,d))$ *is* (a,b,c,d) *while the Adams consensus tree is* $(a,(b,c),d)$». He proves that the local consensus tree for a forest $\mathcal{F}$ equals the Adams consensus tree for the collection made of all trees T such that $\mathcal{R}_{CT}(\mathcal{F}) \subseteq \mathcal{R}(T)$.

We disagree with Bryant on the fact that «*[Kannan et al.] describe an algorithm for constructing an RV-II tree in* $O(n^2)$ *for two trees. The algorithm is identical to that for constructing the Adams consensus tree*». Indeed, for the forest comprised of two trees $T_1 = (((((a,b),c),d),e),f)$ and $T_2 = (((((d,e),f),a),b),c)$, the RV-II tree proposed by the RV-II Construction Algorithm coincides with the tree built by the Build algorithm and not with the Adams consensus tree. The RV-II Construction Algorithm correctly reconstructs the Build tree in $O(n^2)$ by using the fact that the tree necessarily exists, while Build runs in $O(n^4)$ time (for faster implementations of this method in the case of binary trees see Section 3.3.1.1).

Definition 3.2.4 *(adapted from Kannan et al. [1998]). A rooted tree* T *is an RV-III of a forest of rooted trees* $\mathcal{F}$ *if* T *leaves unresolved all triples* $\{a,b,c\} \in L(\mathcal{F})$ *on which the trees in* $\mathcal{F}$ *disagree or which are unresolved in all trees of* $\mathcal{F}$ *and preserves a maximum number of compatible triplets. Moreover* T *cannot display a triplet* t *such that* $\bar{t} \in \mathcal{R}(\mathcal{F})$.

The authors prove that RV-III tree of two trees always exists, is unique and can be computed in $O(n^3)$.

3.2.2.3 The R* consensus trees

The R* consensus method complements the RV-II tree for a forest of rooted trees. This method consists first in computing the set $\mathcal{R}$ of rooted triples t of $\mathcal{R}(\mathcal{F})$ that appear in more trees than their conflicting triples $\bar{t}$. In other words, a triplet $ab|c \in \mathcal{R}(\mathcal{F})$ is kept in $\mathcal{R}$ if and only if $f(ab|c) > f(ac|b)$ and $f(ab|c) > f(bc|a)$, where $f(t)$ is the frequency of the triplet t in $\mathcal{F}$. Note that this set is not always compatible. The *R* consensus tree* is the tree T such that $\mathcal{R}(T) \subseteq \mathcal{R}$ maximizing $|\mathcal{R}(T)|$. Note that this tree is unique and can be obtained using the strong cluster algorithm of Berry and Bryant [1999]. An equivalent method based on quadruplets has also been proposed [the Q* consensus, Berry and Gascuel, 2000; Bryant, 2003].

An example of R* consensus trees is shown in Figure 3.9(iii). When $\mathcal{F}$ contains two rooted trees, the R* consensus tree coincides with the RV-III tree for $\mathcal{F}$ (see Definition 3.2.4). Bryant [2003] proved that, given a forest of rooted trees $\mathcal{F}$, every cluster present in the majority-rule consensus tree or in the semi-strict consensus tree for $\mathcal{F}$ is in the R* consensus tree for $\mathcal{F}$.

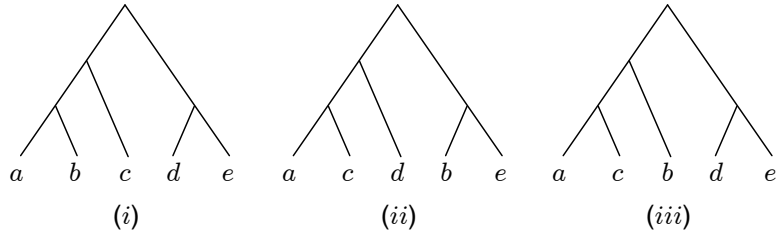


Figure 3.9: **Example of R* consensus trees** - The forest comprises three rooted trees (i)-(iii). The R* consensus tree for this forest coincides with the input tree depicted in (iii).

3.2.2.4 The common pruned-and-regrafted trees

For rooted forests, terminal taxa not included in a MAST tree (see Section 3.2.1.7) can be regrafted to the tree. A way to perform the regrafting is to reconnect removed taxa such that any cluster appearing in all input trees is present in the regrafted tree to obtain the CPRT (*Common Pruned-and-Regrafted Tree*). This method has been introduced by Gordon [Finden and Gordon, 1985; Gordon, 1980] but we describe it as presented by Bryant [2003]. Note that, although the MAST problem is defined for rooted and unrooted forests, a CPRT can be computed only for collections of rooted trees since its computation requires the use of clusters[7].

The CPRT might not be unique. Indeed, since the CPRT consists in regrafting terminal taxa not included in a MAST tree and several MAST trees may exist, it follows that the CPRT may not be unique.

A CPRT for a forest $\mathcal{F}$ is the tree that contains exactly the clusters returned by Algorithm 2 since Finden and Gordon [1985] proved that these clusters are compatible. For example, for the two trees in Figure 3.10(i)-(ii) the set $\mathcal{C}(T_M)$ contains clusters $\{a\}, \{c\}, \{d\}, \{e\}, \{ac\}, \{acd\}, \{acde\}$. The set of clusters C computed by Algorithm 2 contains the clusters of $\mathcal{C}(T_M)$ except the clusters $\{acd\}$ and $\{acde\}$ that are substituted by the clusters $\{abcd\}$ and $\{abcde\}$ respectively. The unique CPRT for this forest is shown in Figure 3.10(iv).

Like other consensus methods, the CPRT has some undesirable properties. One drawback is the difficulty of identifying the MAST that, with the unavailability of an implementation of this method in some widely distributed computer packages as PHYLIP [Felsenstein, 2005], explains the little attention that the CPRT have received from systematists. Moreover, the trees returned by this method do not

[7] Remind that clusters are defined only for rooted trees.

Algorithm 2: CPRT($\mathcal{F}$) (adapted from [Bryant, 2003])

Data: A forest of rooted trees $\mathcal{F} = \{T_1, T_2, \cdots, T_k\}$.
Result: The set of clusters $\mathcal{C}$ induced by the CPRT of $\mathcal{F}$, as defined in Gordon [1980].

1 $C \leftarrow \emptyset$;
2 $T_M \leftarrow MAST(\mathcal{F})$;
3 $L' \leftarrow L(\mathcal{F}) - L(T_M)$;
4 **foreach** (*cluster* $C_i \in \mathcal{C}(T_M)$ **do**
5 $A_i \leftarrow C_i$;
6 **foreach** (*taxon* $l_j \in L'$) **do**
7 $T_S \leftarrow$ strict consensus tree for $\mathcal{F}|_{(L(T_M) \cup l_j)}$;
8 **if** ($\{C_i \cup l_j\} \in \mathcal{C}(T_S)$) **then**
9 $A_i \leftarrow A_i \cup l_j$;
10 $C \leftarrow C \cup A_i$;
11 $T_{SC} \leftarrow$ strict consensus tree for $\mathcal{F}$;
12 **return** $\mathcal{C} \cup \mathcal{C}(T_{SC})$;

have the property to contain all triplets that are common to all trees. For example, for the afore-described forest T_e we have two common triplets *i.e.*, $ab|c$ and $de|f$ but none of the two CPRT contains both triplets. However the CPRTs contain all common clusters of the forest. Indeed, the CPRT refines the strict consensus tree, since it contains all its clusters (see Algorithm 2, line 11). Bryant [2003] proved that for any common pruned-and-regrafted tree T for a forest $\mathcal{F}$, it holds that $\forall t \in \mathcal{R}(T)$, $\exists T_i \in \mathcal{F}$ such that $t \in \mathcal{R}(T_i)$. Note that the Adams consensus tree has the same property.

In the next section we describe the widespread supertree methods.

3.3 Supertree methods

We have seen above *consensus* methods. These methods take as input a forest of trees in which all input trees have the same leaf set. However, there are several situations where input trees leaf sets can overlap yet not exactly coincide *e.g.*, when combining analyses of several data sets, each of which contains information for different groups of taxa or when applying a divide and conquer strategy for constructing large phylogenies. *Supertree* methods have been introduced to deal with such kind of input. In that sense, supertree methods extend consensus methods. In the next sections we provide a review of most supertree methods currently available. We will see that some supertree methods are directly inspired by consensus methods, while others are based on new principles.

In the next sections we often rely on *graph* theory. A graph is a pair (V, E) composed of a set of vertices V and a collection of edges E that connect pairs of

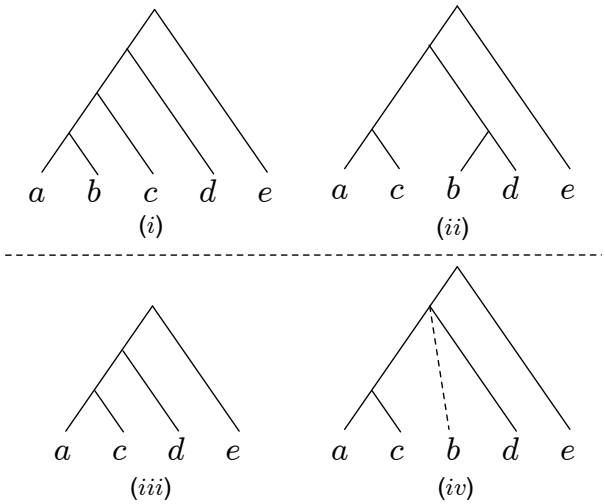


Figure 3.10: **Example of the common pruned-and-regrafted consensus tree** - the input forest is comprised of two rooted trees (i) and (ii). The MAST and the CPRT are depicted respectively in (iii) and (iv).

vertices. A graph may be *undirected*, meaning that there is no distinction between the two nodes associated with each edge, or *directed*, if some edges may be directed from one node to another. Note that, as mentioned in Section 3.1, a tree is a graph in which any two nodes are connected by exactly one path.

3.3.1 The OneTree supertree method and its variants

This set of supertree methods encodes topological relationships contained in the source trees in a graph introduced by Aho et al. [1981], hence is known as the Aho graph. These supertree methods are defined only for rooted trees.

3.3.1.1 The OneTree supertree

The OneTree supertree method, proposed by Ng and Wormald [1996] and then modified by Bryant [1997], is based on the BUILD algorithm [Aho et al., 1981]. The BUILD algorithm is a yes-or-no algorithm that tells whether a collection of triplets $\mathcal{R}$ on a leaf set L is compatible or not. To achieve its goal, BUILD tries to build a tree displaying all triplets in $\mathcal{R}$, *i.e.*, to find a tree such that $L(T) = L$ and $\mathcal{R} \subseteq \mathcal{R}(T)$; if the process is blocked at some step, this means that the input triplets are not compatible. The OneTree supertree method for a rooted forest $\mathcal{F}$ consists in applying the BUILD algorithm to the triplet set $\mathcal{R}(\mathcal{F})$, to obtain a tree T_B such that $L(T_B) = L(\mathcal{F})$ and $\mathcal{R}(\mathcal{F}) \subseteq \mathcal{R}(T_B)$. If such a tree does not exist *i.e.*, if $\mathcal{F}$ is not compatible, this method does not return a tree, so the OneTree supertree method does not handle incompatible source trees. In practice since phylogenies usually conflict with one another (see Chapter 2), this method is of limited use.

However, here we describe it in details since several methods detailed in this section are modifications of the OneTree supertree method.

This method builds a tree recursively, from the root to the leaves. In other words, the largest clusters are first identified, then clusters included in the first ones, and so on. The composition of the clusters is guided by the structure of the *Aho graph*, or *clustering graph*.

The Aho graph for a triplet set $\mathcal{R}$ on a leaf set L, denoted by $\boldsymbol{\mathcal{G}(\mathcal{R}, L)}$ is the undirected graph with vertices L such that there is an edge in $\mathcal{G}(\mathcal{R}, L)$ connecting two vertices a and b if and only if there exists $ab|c \in \mathcal{R}$. Thus, an edge between two taxa means that at least one triplet "sees" these two taxa in the same cluster. The vertices of $\mathcal{G}(\mathcal{R}, L)$ are denoted by $V(\mathcal{G}(\mathcal{R}, L))$. A connected component C_i of a graph is a maximal set of taxa linked to one another, *i.e.*, such that for any pair a, b of taxa in C_i , there is a path from a to b. The connected components of graph $\mathcal{G}(\mathcal{R}, L)$ are denoted by $CC(\mathcal{G}(\mathcal{R}, L))$. The vertices of a component C_i of $\mathcal{G}(\mathcal{R}, L)$ are denoted by $V(C_i)$. When the Aho graph contains several connected components, they correspond to the maximal clusters of the tree that is built to represent the input collection of triplets (if such a tree exists). Then, the sub-clusters contained in each of these primary clusters are found by recursively processing Aho graphs for subsets of triplets that respectively concern the taxa of these clusters: the restriction of $\mathcal{R}$ to taxa of a component C_i is denoted by $\mathcal{R}|_{V(C_i)}$ and defined as $\{ab|c \in \mathcal{R} \mid \{a, b, c\} \subseteq V(C_i)\}$. The algorithm is applied recursively to each couple $(\mathcal{R}|_{V(C_i)}, V(C_i))$. The recursive calls stop when dealing with components containing less than 3 taxa, since there is no triplet (hence incompatibility) on so few taxa. However, if at some point in the recursive process, the Aho graph for several taxa has only one connected component, this means that the input trees are conflicting on the resolution of these taxa. When this happens, the algorithm states that the collection of source trees is incompatible. Otherwise, when all recursive calls succeed, the algorithm concludes that the source trees are compatible and returns a tree T_B containing exactly the deduced clusters and such that $L(T_B) = L$ and $\mathcal{R} \subseteq \mathcal{R}(T_B)$. The outline of the BUILD algorithm is given in Algorithm 3. For instance, let $\mathcal{F}_1$ be the collection comprised of two rooted trees $(((a, c), b), (e, f))$ and $(((a, d), b), c)$. In this case $\mathcal{R}(\mathcal{F}_1) = \{ac|b,\ ac|e,\ ac|f,\ ab|e,\ ab|f,\ bc|e,\ bc|f,\ ef|a,\ ef|b,\ ef|c,\ ad|b,\ ad|c,\ ab|c,\ bd|c\}$ and $L = \{a, b, c, d, e\}$. The Aho graph $\mathcal{G}(\mathcal{R}(\mathcal{F}_1), L)$ is shown in Figure 3.11(i). This graph contains two connected components: $C_1 = \{e, f\}$ and $C_2 = \{a, b, c, d\}$. Since C_2 contains more than three taxa, we call the BUILD algorithm for $\mathcal{G}_2 = \mathcal{G}(\mathcal{R}(\mathcal{F}_1)|_{V(C_2)}, V(C_2))$. More precisely, $\mathcal{R}(\mathcal{F}_1)|_{V(C_2)} = \{ac|b,\ ad|b,\ ad|c,\ ab|c,\ bd|c\}$, so the graph $\mathcal{G}_2$ is connected (see Figure 3.11(ii)), which leads the algorithm to detect the incompatibility of the source trees. Let $\mathcal{F}_2$ be a slightly different collection that comprises the trees $(((a, c), b), (e, f))$ and $((a, d), b, c)$. In this case we obtain the same two connected component C_1 and C_2 of Figure 3.11(i). This time, since $\mathcal{R}(\mathcal{F}_2)|_{V(C_2)} = \{ac|b,\ ad|b,\ ad|c\}$, the graph $\mathcal{G}(\mathcal{R}(\mathcal{F}_2)|_{V(C_2)})$ contains two connected components $C_3 = \{a, b, d\}$ and $C_4 = \{b\}$ (see Figure 3.11(iii)). The component C_3 can be ulteriorly decomposed into two connected components $C_5 = \{a, d\}$ and $C_6 = \{c\}$, shown in Figure 3.11(iv). It follows that the OneTree supertree for this

forest is $((((a,d),c),b),(e,f))$.

Algorithm 3: Build$(\mathcal{R}, L)$

Data: A triplet set $\mathcal{R}$ on a leaf set L.
Result: A tree $T : L(T) = L$ and $\mathcal{R} \subseteq \mathcal{R}(T)$ or a statement that no such a tree exists.

```
1  if (|L| = 1) then return a single node labeled by the label of L;
2  else
3      if (|L| = 2) then
4          return a tree with two leaves respectively labeled by the labels of L;
5      else
6          create a new tree T composed by a single unlabeled node r;
7          construct G(R, L);
8          if (|CC(G(R, L))| = 1) then return "no such a tree exists" ;
9          else
10             foreach (connected component C_i ∈ CC(G(R, L))) do
11                 if (Build(R|_{V(C_i)}, V(C_i)) returns a tree T_{C_i} ) then
12                     add the root node of T_{C_i} as son of r in T;
13                 else
14                     return "no such a tree exists";
15         return T;
```

The OneTree supertree algorithm for a forest $\mathcal{F}$ runs in $O(|\mathcal{R}(\mathcal{F})| \cdot |L(\mathcal{F})|)$ time [Bryant, 1997]. There exists a faster implementation of this method in the case of binary trees that runs in $O(m \cdot |L(\mathcal{F})|^{0.5})$ time, where $m = \sum_{T_i \in \mathcal{F}} \mathcal{I}(T_i)$ [Henzinger et al., 1999], where, recall, $\mathcal{I}(T)$ is the set of interior nodes in T. This algorithm can be improved to $O(m \cdot \log^2(|L(\mathcal{F})|)$ by changing the dynamic connectivity algorithm it resorts to [Berry and Semple, 2006]. Obtaining a tree T such that $\mathcal{R}(\mathcal{F}) \subseteq \mathcal{R}(T)$ for a compatible forest $\mathcal{F}$ can also be done using the AncestralBuild algorithm [Berry and Semple, 2006; Daniel and Semple, 2004]. This method accepts as input trees where some internal nodes can be labeled and is not based on the Aho graph but on a graph called *descendancy graph*. Its running time for input trees of unbounded degree is $O\big(\log^2(|L(\mathcal{F})|) \cdot \sum_{T_i \in \mathcal{F}} \sum_{u \in \mathcal{I}(T_i)} d(u)^2\big)$ where $d(u)$ denotes the degree of the node u.

Note that for a compatible triplet set $\mathcal{R}$ on a leaf set L, there are often more than one tree displaying all triplets in $\mathcal{R}$. Moreover, the number of rooted phylogenetic trees with this property may be exponential in $|\mathcal{R}|$ [Semple, 2003]. Constantinescu and Sankoff [2003] provided an algorithm called SUPERB that takes a compatible set of triplets and returns all binary trees that display $\mathcal{R}$, each of them in polynomial time. Semple [2003] presents the method AllMinTrees that returns all trees $\mathcal{F}_R^{min}$

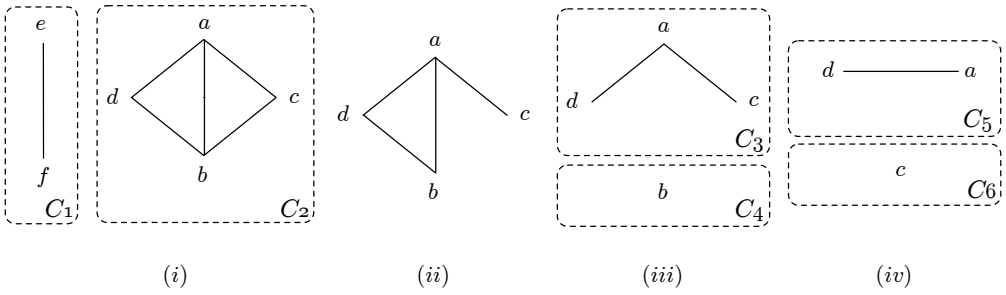


Figure 3.11: **Examples of Aho graphs** - Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two forests comprised respectively of trees $(((a,c),b),(e,f))$ and $(((a,d),b),c)$ and of trees $(((a,c),b),(e,f))$ and $((a,d),b,c)$. (i) the initial Aho graph for both collections. This graph contains two connected components *i.e.*, $C_1 = \{e,f\}$ and $C_2 = \{a,b,c,d\}$. (ii) $\mathcal{G}(\mathcal{R}(\mathcal{F}_1)|_{V(C_2)}, V(C_2))$. This graph is connected, showing that the input trees conflict on the resolution of $\{a,b,c,d\}$, hence are incompatible. (iii) $\mathcal{G}(\mathcal{R}(\mathcal{F}_2)|_{V(C_2)}, V(C_2))$ contains two connected components *i.e.*, $C_3 = \{a,c,d\}$ and $C_2 = \{b\}$. (iv) $\mathcal{G}(\mathcal{R}(\mathcal{F}_1)|_{V(C_3)}, V(C_3))$ contains two connected components *i.e.*, $C_4 = \{a,d\}$ and $C_2 = \{c\}$.

that display $\mathcal{R}$ and are minimal *i.e.*, such that, $\forall T \in \mathcal{F}_R^{min}$ no internal edge of T can be contracted so that the resulting tree also displays $\mathcal{R}$. Both methods are based on the BUILD algorithm.

Ng and Wormald [1996] extend the BUILD algorithm to check the consistency of a set of rooted triples *and* fan trees in polynomial time. A *fan* tree (also called star tree) on a leaf set L is a completely unresolved rooted tree on L, *e.g.*, the fan tree on $\{a,b,c,d\}$ is the tree (a,b,c,d). We say that a tree T is compatible with a fan tree t_F if $T|_{L(t_F)} = t_F$. Ng and Wormald also provided an algorithm called ALLTREES that takes a compatible set of triplets and fan trees and returns all trees T displaying this set. In this manuscript we are not interested in this latter problem, since we interpret polytomies as "soft" (see Section 3.1.3) *i.e.*, when observing an unresolved triplet we do not want to impede its resolution in the consensus tree or supertree.

Also the BUILD-WITH-DISTANCES supertree method of Willson [2004] is an algorithm based on a variation of the BUILD algorithm. This method takes as input rooted weighted trees and makes essential use of input branch length information to construct a supertree when an additive supertree exists. In such a case a supertree that displays the OneTree supertree is returned for which some polytomies may have been resolved using branch length information. When an additive supertree does not exist, the method outputs a tree (the minimal threshold tree) with interesting properties [Willson, 2004].

3.3.1.2 The MinCut (MC) supertree

Semple and Steel's MinCut supertree algorithm [Semple and Steel, 2000] modifies OneTree so that it always returns a tree. Before introducing the MinCut supertree algorithm we recall some notations needed to describe this method.

Let G be a graph with vertices V and edges E and let V' be a set of vertices such that $V' \subseteq V$, we denote by $\boldsymbol{G|_{V'}}$ the graph with vertices V' and edges E', where E' is the subset of E having both endpoints in V'[8]. Given a set of edges E' such that $E' \subseteq E$, we denote by $\boldsymbol{G \backslash E'}$ the graph obtained from G by deleting all edges in E'.

Given an edge $e = (u, v) \in E$, *contracting* e consists in deleting (u, v) and identifying its endpoints, *i.e.*, u and v. We denote by $\boldsymbol{G \odot E'}$ the graph obtained from G by contracting all edges in E', deleting loops, and replacing each parallel class of edges, *i.e.*, edges with identical endpoints, with a single edge. See Figure 3.12 for an example of these graphs.

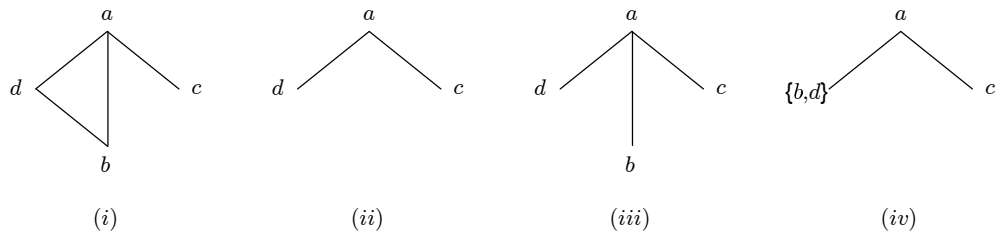


Figure 3.12: **Examples of graphs used by the MC supertree algorithm** - (i) a graph G. (ii) $G|_{V'}$ for $V' = \{a, c, d\}$. (iii) $G \backslash E'$ for $E' = (b, d)$. (iv) $G \odot E'$ for $E' = (b, d)$.

If G is a connected weighed graph, we call a *minimum-weight cut set* of G a set of edges $\bar{E} \subseteq E$ such that $G \backslash \bar{E}$ is disconnected and the sum of the weights of the edges in $\bar{E}$ is minimum over all possible sets E' such that $G \backslash E'$ is disconnected. Let w be a weight function that associates a rational positive weight $w(i)$ to the i^{th} tree of the forest $\mathcal{F}$ *i.e.*, a function $w : \{1, \cdots, |\mathcal{F}|\} \to (\mathbb{Q}^+)^{|\mathcal{F}|}$. [9]

The *weighted Aho graph* for the forest $\mathcal{F}$ and the weighted function w, denoted by $\boldsymbol{\mathcal{G}(\mathcal{F}, w)}$, has the same vertex and edge sets as $\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F}))$ with edges weighted in the following way: the weight of an edge (a, b) , denoted by $w(a, b)$, is the sum of the weights $w(i)$ for all trees T_i such that there exists at least one triplet $ab|c \in \mathcal{R}(T_i)$ (see Figure 3.13 for an example).

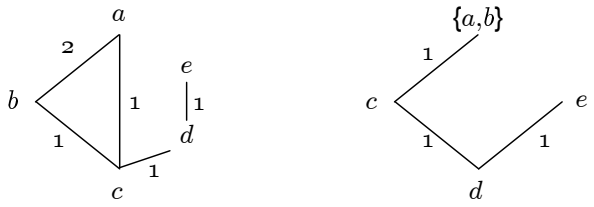
From this graph we can obtain a second graph, called the *weighted collapsed Aho graph*, denoted by $\boldsymbol{\hat{\mathcal{G}}(\mathcal{F}, w)}$. First, we define $\boldsymbol{E_{max}}$ to be the set of edges (a, b) such that $w(a, b) = \sum_{i=1}^{|\mathcal{F}|} w(i)$. Since the weights are strictly positive, E_{max} contains all edges (a, b) supported unanimously by all input trees *i.e.*, a and b are in a non-trivial

[8] We have already implicitly used this notation in Section 3.3.1.1, to define the graph $\mathcal{G}(\mathcal{R}(\mathcal{F})|_{V(C_i)}, V(C_i))$.

[9] $\mathbb{Q}^+$ is chosen instead of $\mathbb{R}^+$ since this limits the computational complexity of the method.

cluster for all $T_i \in \mathcal{F}$. Then, $\hat{\mathcal{G}}(\mathcal{F}, w) = \mathcal{G}(\mathcal{F}, w) \odot E_{max}$. The weigh of the new edges (V_1, V_2), for instance the edge $(\{a, b\}, c)$ in Figure 3.13, is set to the sum of the weights $w(i)$ of those trees $T_i \in \mathcal{F}$ having at least one triplet $xy|z \in \mathcal{R}(T_i)$ such that $x \in V_1$ and $y \in V_2$.

For instance, let $\mathcal{F}$ be a forest comprised of two rooted trees $(((a, b), c), (d, e))$ and $((a, b), (c, d))$ and let w be the constant function $w : \{1, 2\} \to \{1, 1\}$. The graph $\mathcal{G}(\mathcal{F}, w)$ is shown in Figure 3.13(i). The only edge with weight 2 is (a, b), so $E_{max} = \{(a, b)\}$. The graph $\hat{\mathcal{G}}(\mathcal{F}, w)$ is shown in Figure 3.13(ii); there is only a new edge $(\{a, b\}, c)$, which has weight 1, since T_2 contains no triplet grouping together ac or bc.



(i) (ii)

Figure 3.13: **Example of the MC supertree algorithm** - for the forest $\mathcal{F}$ comprised of two trees $T_1 = (((a, b), c), (d, e))$ and $T_2 = ((a, b), (c, d))$; w is the constant function $w : \{1, 2\} \to \{1, 1\}$. (i) the graph $\mathcal{G}(\mathcal{F}, w)$. (ii) the graph $\hat{\mathcal{G}}(\mathcal{F}, w)$. The MC supertree for this forest is the tree $((a, b), c, d, e)$.

Given a forest $\mathcal{F}$ of rooted trees and a weighted function $w : \{1, \cdots, |\mathcal{F}|\} \to (\mathbb{Q}^+)^{|\mathcal{F}|}$, the MC supertree T is built recursively, from the root to the leaves. First, the maximal clusters of T are identified, then clusters included in the first ones, and so on. As for the OneTree algorithm, the composition of the clusters is guided by the structure of the Aho graph. When the Aho graph contains several connected components, the MC algorithm works exactly as the OneTree algorithm (Section 3.3.1.1). If at some point in the recursive process, the Aho graph for a set of at least three taxa has only one connected component, this means that the input trees are conflicting on the resolution of these taxa. In this case the algorithm constructs the $\hat{\mathcal{G}}(\mathcal{F}, w)$ graph as described above. This graph, like the Aho graph, has only one connected component since it is its weighted version. A new disconnected graph $\hat{\mathcal{G}}(\mathcal{F}, w) \backslash \bar{E}$ is obtained from $\hat{\mathcal{G}}(\mathcal{F}, w)$ by deleting all edges $\bar{E}$ comprised in at least one minimum-weight cut set of this graph. The algorithm is then recursively run on each connected component of $\hat{\mathcal{G}}(\mathcal{F}, w) \backslash \bar{E}$. The recursive calls stop when dealing with components containing less than 3 taxa, since there is no triplet on so few taxa. For instance, for the forest $\mathcal{F}$ afore-described, all edges of the graph $\hat{\mathcal{G}}(\mathcal{F}, w)$ (Figure 3.13(ii)) lie in at least one minimum-weight cut set of this graph so the MC supertree for this forest is the tree $((a, b), c, d, e)$. The outline of the MinCut algorithm is given in Algorithm 4.

Note that the tree returned by MC$(\mathcal{F}, w)$ depends on the weighted function

Algorithm 4: MC($\mathcal{F}, w$)

Data: A set of rooted trees $\mathcal{F}$ and a weighed function $w : \{1, \cdots, |\mathcal{F}|\} \rightarrow (\mathbb{Q}^+)^{|\mathcal{F}|}$.
Result: A tree T_{MC} that is the MC supertree for the pair $(\mathcal{F}, w)$.

if $(|L(\mathcal{F})| = 1)$ **then** return a single node labeled by the label of $L(\mathcal{F})$;
else
  if $(|L(\mathcal{F})| = 2)$ **then**
    return a tree with two leaves respectively labeled by the labels of $L(\mathcal{F})$;
  else
    create a new tree T_{MC} composed by an unlabeled node r;
    construct $\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F}))$;
    if $(|CC(\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F})))| = 1)$ **then**
      construct $\hat{\mathcal{G}}(\mathcal{F}, w)$;
      $\mathcal{G} \leftarrow \hat{\mathcal{G}}(\mathcal{F}, w)$;
      construct the set E' of edges of $\mathcal{G}$ that lie in at least one minimum-weight cut set of $\mathcal{G}$;
      $\mathcal{C} \leftarrow CC(\mathcal{G} \backslash E')$;
    else
      $\mathcal{C} \leftarrow CC(\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F})))$;
    foreach (*connected component* $C_i \in \mathcal{C}$) **do**
      $T_{C_i} \leftarrow$ MC($\mathcal{F}|_{V(C_i)}, w$));
      add the root node of T_{C_i} as son of r in T_{MC};
return T_{MC};

w. However, whatever weighted function is used to construct it, the MC supertree method satisfies several desirable properties. First of all, if the forest $\mathcal{F}$ is compatible, $T_{\text{MC}}(\mathcal{F}, w)$ is the OneTree and thus satisfies $\mathcal{R}(\mathcal{F}) \subseteq \mathcal{R}(T_{\text{MC}}(\mathcal{F}, w))$. Moreover, like the Adams consensus (property (i) of section 3.2.2.1 on page 50), this method returns a tree displaying all nestings and triplets shared by all input trees in $\mathcal{F}$. Semple and Steel [2000] also proved that the MC supertree method satisfies several interesting properties.

Lemma 3.3.1 (Semple and Steel, 2000) *Let $\mathcal{F}$ be a (weighted) forest of rooted trees and let T be a rooted tree. Suppose that L is a subset of $L(\mathcal{F})$ such that $\forall T_i \in \mathcal{F}$, $T = T_i|_L$. Then $T_{\text{MC}}(\mathcal{F}, w)$ displays T. Furthermore, if T is binary, then $T = T_{\text{MC}}(\mathcal{F}, w)|_L$.*

This lemma ensures that the MC supertree for a forest $\mathcal{F}$ refines all trees T that have the property to display the contraction of all input trees to a leaf set $L \subseteq L(\mathcal{F})$ *i.e.*, $T = T_i|_L$, $\forall T_i \in \mathcal{F}$.

Recall that, given a rooted tree T, a group of a taxa A nests within a larger group B, denoted by $A <_T B$, if A is included in B.

The following two theorems state the relationships that exist between the Adams consensus tree and the MC tree in the consensus setting.

Theorem 3.3.2 (Semple and Steel, 2000) *Let $\mathcal{F}$ be a set of rooted trees having the same leaf set X. Let A and B be subsets of X and let $\mathcal{A}(\mathcal{F})$ be the Adams consensus tree for $\mathcal{F}$. If $A <_{\mathcal{A}(\mathcal{F})} B$, then $A <_{T_{\text{MC}}(\mathcal{F},w)} B'$ for every cluster B' of $\mathcal{A}(\mathcal{F})$ that contains B.*

However, the Adams consensus tree and the MC supertree do not always coincide, as stated in the following theorem.

Theorem 3.3.3 (Semple and Steel, 2000) *Let $\mathcal{F}$ be a set of rooted trees having the same leaf set X and let $\mathcal{A}(\mathcal{F})$ be the Adams consensus tree for $\mathcal{F}$. Then exactly one of the following holds:*

1. $\mathcal{R}(\mathcal{A}(\mathcal{F})) \subseteq \mathcal{R}(T_{\text{MC}})$
2. $\mathcal{R}(\mathcal{A}(\mathcal{F})) \nsubseteq \mathcal{R}(T_{\text{MC}})$ *and* $\mathcal{R}(\mathcal{A}(\mathcal{F})) \not\supseteq \mathcal{R}(T_{\text{MC}})$

All listed properties are satisfied whatever weighting function w is used. So, the MC supertrees have several attractive properties. However, when applied to some examples this method can give less attractive results than other methods [Page, 2002]. For example, let $\mathcal{F}$ be a forest that comprises two rooted binary trees $T_1 = (((((x_2, x_3), x_1), c), b), a)$ and $T_2 = ((((((y_3, y_4), y_2), y_1), a), b), c)$ and w the constant function $w : \{1, 2\} \to \{1, 1\}$. These two trees share only the three leaves a, b, and c and disagree on the relationships among those leaves since T_1 contains the triplet $bc|a$ and T_2 contains the triplet $ab|c$. The tree produced by MC for these two trees is the tree $((((((y_3, y_4), y_2), y_1), a), b), c, x_1, x_2, x_3)$. This tree does not contain any resolution for the leaves x_1, x_2 and x_3, although there is no information in T_2 that impedes to group these leaves as they are grouped in the first tree. In contrast, relationships among $\{y_1, y_2, y_3, y_4\}$ are fully resolved. Furthermore, the supertree contains the triplet $ab|c$, *i.e.*, is in contradiction with T_1. For the forest comprised of the two rooted binary trees $T_3 = ((((((x_3, x_4), x_2), x_1), c), b), a)$ and $T_4 = (((((y_3, y_2), y_1), a), b), c)$ the MC supertree is the tree $((((((x_3, x_4), x_2), x_1), c), b), a, y_1, y_2, y_3)$. This tree does not contain any resolution for the leaves y_1, y_2 and y_3 and the relationships among $\{x_1, x_2, x_3, x_4\}$ are fully resolved. This time the supertree contains the triplet $bc|a$. This example shows that this method can be sensitive to the size of the input trees, favoring the resolutions contained in the biggest trees.

3.3.1.3 The Modified-MinCut (MMC) supertree

Page [2002] criticized the MC method on several points. It has been proved that there exists no consensus method for rooted trees that ensures to return a tree displaying all the uncontradicted information contained in a set of trees [Steel et al.,

2000, property P7]. Although it is impossible to construct such a supertre method, Page's first criticism to Semple and Steel's method is that the MC supertree method does not even aim to maximize the uncontradicted information contained in the supertree. For example, for the forest comprised of the two afore-described trees T_1 and T_2, the MC supertree does not contain any resolution for the leaves x_1, x_2 and x_3, although T_1 contains the clusters x_1, x_2 and x_1, x_2, x_3 and none is contradicted by T_2. Then we can insert the two clusters in the MC supertree, obtaining a tree that contains more uncontradicted information, without adding any contradicted information. Another criticism that Page formulated against the MC method is that it does not try to minimize the information contained in the supertree that contradicts the source trees (*e.g* the triplets $ab|c, ab|x_1, ab|x_2$ in $\text{MC}(T_1, T_2, w)$ with $w : \{1, 2\} \to \{1, 1\}$).

To try to avoid those drawbacks, Page [2002] proposed a modification of the MC supertree method, called the Modified-MinCut (MMC) supertree method. The MMC supertree method is a heuristic that aims to avoid as much as possible contradicted information, having as consequence to permit to insert more uncontradicted information in the supertree than the MC method, while still returning a tree that has all the properties of the MC supertree.

The only difference between the two methods resides in the graph that is constructed when the graph $\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F})))$ is connected. While the MC supertree method constructs the graph $\hat{\mathcal{G}}(\mathcal{F}, w)$, the MMC supertree method relies on a different graph, called the MMC graph and denoted by $\mathcal{G}_{\text{MMC}}(\mathcal{F}, \hat{\mathcal{G}}(\mathcal{F}, w))$. The MMC algorithm coincides with the MC one but for line 10 of Algorithm 4 that is replaced by $\mathcal{G} \leftarrow \mathcal{G}_{\text{MMC}}(\mathcal{F}, \hat{\mathcal{G}}(\mathcal{F}, w))$ in the MMC algorithm. That is why we do not detail the MMC algorithm but only the construction of the MMC graph.

The graph $\mathcal{G}_{\text{MMC}}(\mathcal{F}, \hat{\mathcal{G}}(\mathcal{F}, w))$ is constructed as follows (see Algorithm 5). First of all, for each edge (a, b) in $\hat{\mathcal{G}}(\mathcal{F}, w)$ we distinguish between unanimous, uncontradicted and contradicted edges. An edge (a, b) is *unanimous* for the forest $\mathcal{F}$ if and only if there exists at least one triplet $ab|c \in \mathcal{R}(T_i)$ $\forall T_i \in \mathcal{F}$. An edge (a, b) is *uncontradicted* if and only if (a, b) is not unanimous and for all trees $T_i \in \mathcal{F}$ one of the following holds:

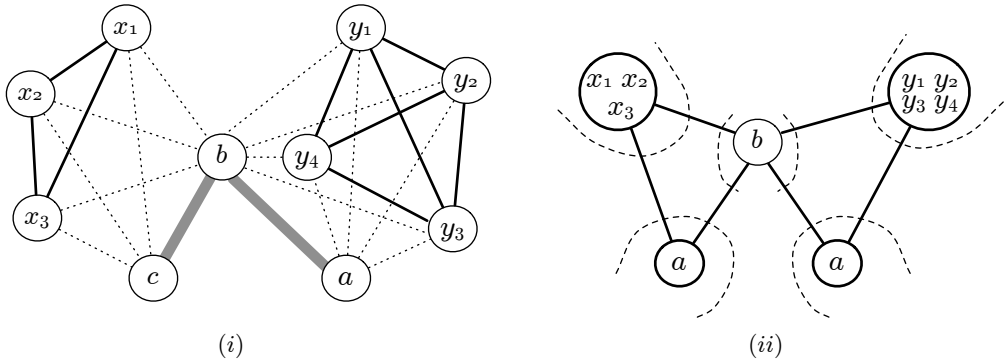
- $lca_{T_i}(a, b) \neq root(T_i)$,
- $lca_{T_i}(a, b) = root(T_i)$ and the root has degree greater than 2,

where $lca_{T_i}(a, b)$ is the lca of a and b in the tree T_i. In other words, an edge (a, b) is uncontradicted if and only if for those trees T_i such that $\nexists ab|c \in \mathcal{R}(T_i)$, the root of T_i is a polytomy. Note that this definition of uncontradicted edges follows from the interpretation of polytomies as soft (see Section 3.1.3). Page interprets polytomies as soft and thus considers a tree containing a triplet $ab|c$ and a tree containing (a, b, c) not in contrast. Edges that are neither unanimous nor uncontradicted are *contradicted* edges. Page's aim was to modify $\hat{\mathcal{G}}(\mathcal{F}, w)$ such as to minimize the number of uncontradicted edges that are cut in the MC method, so he extended

Semple and Steel's approach of merging nodes linked by unanimous edges to include nodes linked by uncontradicted edges.

If we can disconnect $\hat{\mathcal{G}}(\mathcal{F}, w)$ by cutting only contradicted edges (Algorithm 5, line 4), this means that we can preserve all uncontradicted edges at this step. Otherwise, at least one uncontradicted edge must be cut to disconnect the graph. Since the algorithm tries to minimize the contradicted information present in the supertree, we would like the minimum-weight cut sets to include contradicted edges whenever possible. If removing all contradicted edges and all edges adjacent to a contradicted edge disconnects $\hat{\mathcal{G}}(\mathcal{F}, w)$ (Algorithm 5, line 9), then we have identified at least one cut that contains a contradicted edge. In the example in Figure 3.14(ii) we have two minimum-weight cut sets, each containing one contradicted edge.

If the graph remains connected, the two graphs $\mathcal{G}_{\text{MMC}}(\mathcal{F}, \hat{\mathcal{G}}(\mathcal{F}, w))$ and $\hat{\mathcal{G}}(\mathcal{F}, w)$ coincide.



(i) (ii)

Figure 3.14: **Example of the MMC supertree algorithm [Page, 2002]** - (i) The graph $\hat{\mathcal{G}}(\mathcal{F}, w)$ for the forest that comprises two rooted binary trees $(((((x_2, x_3), x_1), c), b), a)$ and $((((((y_3, y_4), y_2), y_1), a), b), c)$. The two bold edges are contradicted, the edges drawn as dashed lines are uncontradicted but adjacent to a contradicted edge. Deleting the contradicted and the adjacent to contradicted edges disconnects the graph. (ii) the graph $\mathcal{G}_{\text{MMC}} \backslash (AC \cap C)$. The six minimum weight cuts of $\mathcal{G}_{\text{MMC}}$ are indicated by dashed lines so the MMC supertree for this forest is the tree $((((y_3, y_4), y_2), y_1), ((x_2, x_3), x_1), a, b, c)$.

The MMC supertree for the pair of trees $(((((x_2, x_3), x_1), c), b), a)$ and $((((((y_3, y_4), y_2), y_1), a), b), c)$ is the tree $((((y_3, y_4), y_2), y_1), ((x_2, x_3), x_1), a, b, c)$ (see Figure 3.14 for more details). Unlike the MC supertree for this forest, this tree contains no contradicted triplets. Moreover, this MMC supertree contains much more uncontradicted triplets than the MC one. Note that both MC and MMC supertree methods can contain clusters not present in any source tree. For instance both MC and MMC supertrees for the trees $(((a, b, c), d, e), f, g)$ and $(((a, b, e), d, c), f)$ contain the cluster $\{a, b\}$. This is related to the fact that both MC and MMC supertree methods share Adams consensus interpretation of soft polytomy (see Sections 3.1.3 and 3.2.2.1).

Algorithm 5: $\mathcal{G}_{\text{MMC}}(\mathcal{F}, \hat{\mathcal{G}}(\mathcal{F}, w))$

Data: A set of rooted trees $\mathcal{F}$ and a weighted graph $\hat{\mathcal{G}}(\mathcal{F}, w)$.
Result: A weighted graph $\mathcal{G}_{\text{MMC}}$.

$\mathcal{G}_{\text{MMC}} \leftarrow \hat{\mathcal{G}}(\mathcal{F}, w)$;
$E' \leftarrow \emptyset$;
compute the set C of contradicted edges in $\mathcal{G}_{\text{MMC}}$;
if ($\mathcal{G}_{\text{MMC}} \backslash C$ *is disconnected*) **then**
    find the connected components of $\mathcal{G}_{\text{MMC}} \backslash C$;
    $E' \leftarrow$ edges connecting two nodes of the same connected component;
else
    build the set AC of all edges in $\mathcal{G}_{\text{MMC}}$ adjacent to a contradicted edge.;
    if ($\mathcal{G}_{\text{MMC}} \backslash (AC \cap C)$ *is disconnected*) **then**
        find the components of $\mathcal{G}_{\text{MMC}} \backslash (AC \cap C)$;
        $E' \leftarrow$ edges connecting two nodes of the same connected component;
return $\hat{\mathcal{G}}(\mathcal{F}, w) \bigodot E'$;

3.3.1.4 The strict consensus supertree

The strict consensus supertree method is often referred to as the first supertree method proposed [Gordon, 1986] although the BUILD algorithm predates it by several years. Like the OneTree supertree method, it deals only with compatible forests. The strict consensus supertree of a compatible forest $\mathcal{F}$ is defined as the strict consensus supertree of all trees T such that T displays each tree in $\mathcal{F}$. Steel [1992] proposed a polynomial time algorithm based on the Aho graph accepting as input any number of rooted trees. This algorithm is based on the following remark: a cluster C is in the strict consensus supertree of $\mathcal{F}$ if and only if, given $x \in C$ and for each pair a, b with $a \in C - \{x\}$ and $b \notin C$, both $\mathcal{G}(\mathcal{R}(\mathcal{F}) \cup \{ab|x\}, L(\mathcal{F}))$ and $\mathcal{G}(\mathcal{R}(\mathcal{F}) \cup \{bx|a\}, L(\mathcal{F}))$ are incompatible. Then, if we construct the OneTree supertree for $\mathcal{F}$ and we eliminate from this tree the clusters that do not pass this test (see Algorithm 6), we obtain the strict consensus supertree for $\mathcal{F}$. This algorithm requires $O(|L(\mathcal{F})|^3 \cdot \lambda)$, where λ is the complexity of computing the graph $\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F}))$ *i.e.*, $O(n^6 \cdot \log(n))$, where $n = |L(\mathcal{F})|$, in the worst case.

Moreover, the MERGETREES algorithm of Berry and Nicolas [2007] can be used to compute in linear time the strict consensus supertree for two rooted trees. A question is whether this algorithm can be extended to obtain a tight complexity for the case of more than two trees.

Bryant [2001] presented a variation of the strict consensus supertree for a bounded number of compatible *unrooted* binary trees. Given a forest of unrooted trees $\mathcal{F}$, Bryant's method returns a supertree T, if it exists, such that $L(T) = L(\mathcal{F})$ and each tree $T_i \in \mathcal{F}$ is an induced subtree of T *i.e.*, $T_i = T|_{L(T_i)}$. When multiple such supertrees exist, Bryant's method returns, in polynomial time, the supertree that is optimal with respect to one of four standard phylogenetic optimization crite-

Algorithm 6: SCS($\mathcal{F}, w$)

Data: A set of rooted trees $\mathcal{F}$.
Result: A tree T_{SCS} that is the strict consensus supertree for $\mathcal{F}$.

```
1 BC ← ∅ // Bad Clusters set;
2 T ← Build(R(F), L(F));
3 foreach (cluster C_i ∈ C(T)) do
4     x ← a leaf of C_i;
5     foreach (a, b ∈ L(F) such that a ∈ C_i − {x}, b ∈ L(F) − L(C_i)) do
6         if !(G(R(F) ∪ {ab|x}, L(F)) and G(R(F) ∪ {bx|a}, L(F))
          incompatible) then
7             BC ← BC ∪ C_i;
8 build the tree T_SCS such that C(T_SCS) = C(T) − BC;
9 return T_SCS;
```

ria: maximum binary compatibility score, maximum quartet score, minimum OLS score and minimum ME score. The time complexity of this approach depends on the chosen optimization criterion [Bryant, 2001, Theorem 3].

3.3.2 Matrix Representation-based methods

In this set of supertree methods the input trees are converted into matrices of another kind of data (binary sequences, distances), and these data are subsequently re-analysed using a standard phylogenetic tree reconstruction method.

3.3.2.1 The Matrix Representation with Parsimony (MRP) supertree

The Matrix Representation with Parsimony (MRP) method is the most commonly used supertree method but also one of the most criticized. It has been independently developed by Baum [1992] and Ragan [1992]. Given a forest $\mathcal{F}$, the MRP method first encodes it into a binary matrix with a row per species of $L(\mathcal{F})$ and a column per cluster of the input trees. Then, a parsimony analysis of the resulting matrix is performed. In more details, this method consists in the following steps:

Rooting: each tree of the input forest is rooted by using a taxon common to all input trees [Baum, 1992]; if a tree is already rooted, re-root it.

Coding: a matrix having an entry for each internal node of each tree is created. Each internal node u of a tree T is encoded as a column in the matrix having state '1' for each taxon in the cluster induced by u and state '0' for all other taxa of T. All taxa that do not belong to the tree are encoded by a '?' (see Figure 3.15 for an example).

Analyzing: the so obtained matrix is analyzed by the parsimony criterion.

Summarizing: return the strict consensus of the most parsimonious trees.

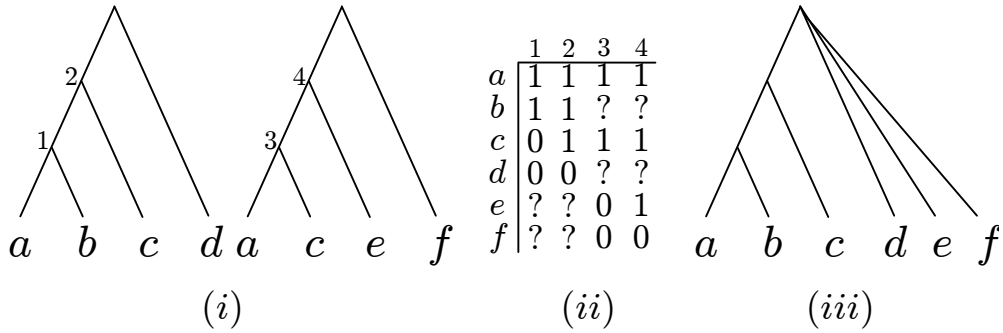


Figure 3.15: **Example of the MRP method** - (i) a forest consisting of two rooted trees. (ii) its MR coding. Internal nodes have their associated clusters encoded in a column of the MR matrix. (iii) the MRP supertree for this forest.

Another way to root the input trees is to root them by an all-zero output [Purvis, 1995a; Ragan, 1992]. The afore-described way to encode the forest is called *matrix representation* or *MR*. Given a forest $\mathcal{F}$ of trees, we denote by MR($\mathcal{F}$) its matrix representation.

Note that there can be an exponential number of most parsimonious trees. Since finding the most parsimonious tree(s) given a character matrix is an NP-complete problem [Graham and Foulds, 1982], heuristics have been proposed, notably the ratchet technique [Goloboff et al., 2008] and MCMC based methods [among others, Ronquist and Huelsenbeck, 2003; Ronquist et al., 2004] that ensure the feasibility of MRP for data sets with large numbers of taxa and/or input trees. Baum and Ragan [2004], in response to Rodrigo's criticism [Rodrigo, 1993, 1996] that the MRP method lacks of an underlying model, argued that their method is based on the idea that input trees can be viewed as state-character trees and, since they often conflict with each other, the characters are combined to infer the species tree. They stated to have chosen parsimony to resolve conflicts for the same reasons that parsimony is used to combine discrete-state characters *i.e.*, efficiency and information content (see Section 1.3).

Several other criticisms have been addressed to this method. First, Purvis [1995a] noticed that the MRP method is biased and some tree topologies can unduly affect the MRP supertree. He attributed this bias to the fact that the information given by the nodes of a tree is not independent. For instance, the matrix column that encodes the node 2 in Figure 3.15 contains some information already present in the matrix column encoding the node 1. To try to avoid this bias, he proposed an alternative coding of the MRP matrix such that each internal node u of a tree T is encoded as an entry in the matrix having:

- '1' for each taxon in the cluster induced by u;
- '0' for each taxon in the clusters induced by the sibling nodes of u ;

- '?' for the remaining taxa.

Ronquist [1996] demonstrated that Purvis's coding leads to less informative matrices and does not eliminate completely the bias. Rodrigo suggested that the MRP bias is due to the different relative sizes of input trees and one would remove the bias by weighting trees. He proposed several weighting schemes, among them one that assigns to each tree a weight in inverse proportion to its number of internal nodes and one based on the bootstrap support for nodes. Wilkinson et al. [2001] suggested that the MRP may also favor source trees that are more unbalanced. Bininda-Emonds and Bryant [1998] argued that the bias does not exactly depend on the different relative sizes of input trees and they presented two examples to prove it. Let $\mathcal{F}_1$ be the forest comprised of trees $T_1 = (((((((a,b),c),d),e),f),g),h)$ and $T_2 = ((b,c),a)$ and let $\mathcal{F}_2$ be the the forest comprised of trees $T_3 = (((a,b),c),d)$ $T_4 = ((a,d),b)$. The MPR supertree for $\mathcal{F}_1$ is the tree that coincides with T_1 but for a polytomy for taxa a,b,c *i.e.*, $((((((a,b,c),d),e),f),g),h)$. This is the expected result, since the two trees only disagree on the resolution of these leaves. In this case the biggest tree does not affect unduly the MRP supertree. On the contrary, if we weight the trees inversely to their number of internal nodes [Ronquist, 1996], we would have as MRP supertree the tree $(((((((b,c),a),d),e),f),g),h)$ that contains the T_2 resolution for taxa a,b,c *i.e.*, Ronquist's weighting has favored the smallest tree. The MRP tree for the $\mathcal{F}_2$ forest coincides with T_3, so in this case the biggest tree is favored. Bininda-Emonds and Bryant [1998] suggested that the bias is due to different relative sizes of the input trees *in the region of conflict*. Indeed, for the forest $\mathcal{F}_1$, the two trees have the same size in the region of conflict (a,b,c) while the forest $\mathcal{F}_2$ the tree T_3 is bigger in the conflict region (a,b,c,d). For this reason, the authors proposed to apply node-based weighting schemes. However, the impact of the size and balance biases is commonly considered to be minimal in practice [Bininda-Emonds et al., 2002] and decreasing with the number of input trees used in the MRP analysis [Bininda-Emonds et al., 1999]. Yet, this is not always true, as demonstrated by other simulation studies [Emonds and Sanderson, 2001].

The MRP method is strongly criticized also for the fact that, when source trees conflict, it can propose clusters not supported by any (combination of) intput tree(s), «*novel clades*» in Bininda-Emonds and Bryant [1998]. Moreover clusters that are contradicted by each and every input tree can be present in the MRP supertree [Cotton et al., 2006; Goloboff, 2005; Goloboff and Pol, 2002]. This even happens in a consensus setting, where combining a set of trees with identical leaf sets. For instance, the MRP supertree of the set of trees comprised of $(((((e,f),d),c),b),a)$ and $(((((b,f),a),e),d),c)$ is the tree $((a,b),c,d,e,f)$ that is in contradiction with both input trees. Note that in this example the size and the shape of both trees are identical.

Moreover, Bininda-Emonds argued that the MRP supertree can also fail to display some triplets common to every input tree «*on sufficiently contrived data, even in the consensus setting*» [Bininda-Emonds et al., 2002].

Personally, we are convinced that, as pointed out by Pisani and Wilkinson [2002] «*MRP may suffer from potentially serious but poorly understood biases and from its potential to produce unjustified new groups. We consider that the properties of MRP [...] should be better understood before MRP can be reasonably adopted as a method of choice for supertree construction*». Some of these numerous criticisms against MRP have motivated our work on supertree methods, presented in Chapter 4.

3.3.2.2 The Matrix Representation with Flipping (MRF) supertree

The Matrix Representation with Flipping (MRF) method ([Chen et al., 2003, 2002], re-formulated in [Eulenstein et al., 2004]) is based on the idea that input trees conflict because of errors *i.e.*, the presence of an incorrect label in a cluster or the absence of one that should be present. In the matrix representation of the input forest, such errors correspond to flips from 0 to 1 or 1 to 0.

Given two matrix representations MR_1 and MR_2 over the same taxa set, we denote by $\mathrm{MR}[i]$ the i^{th} column of a MR. The flip-distance between MR_1 and MR_2 is denoted by $d_f(\mathrm{MR}_1, \mathrm{MR}_2)$ where:

- The flip-distance $d_f(\mathrm{MR}_1[i], \mathrm{MR}_2[j])$ is the minimum number of flips '0' $\leftrightarrow$ '1' needed to convert $\mathrm{MR}_1[i]$ into $\mathrm{MR}_2[j]$. Positions where $\mathrm{MR}_1[i]$ or $\mathrm{MR}_2[j]$ are encoded with '?' are not considered.
- The flip-distance $d_f(\mathrm{MR}_1[i], \mathrm{MR}_2)$ is the minimum flip-distance from $\mathrm{MR}_1[i]$ to any column of MR_2 *i.e.*, $min_j(d_f(\mathrm{MR}_1[i], \mathrm{MR}_2[j]))$.
- The flip-distance $d_f(\mathrm{MR}_1, \mathrm{MR}_2)$ is $\sum_i(d_f(\mathrm{MR}_1[i], \mathrm{MR}_2))$.

For instance, for the matrices in Table 3.1, $d_f(\mathrm{MR}_1, \mathrm{MR}_2)=d_f(\mathrm{MR}_2, \mathrm{MR}_1)=1$. Note that the flip-distance between two matrices is not symmetric. Given a forest of rooted trees, the MRF method consists in finding all binary trees T such that the flip-distance $d(\mathrm{MR}(\mathcal{F}), \mathrm{MR}(T))$ is minimal. If more than one such supertree

a	1	1	1	1
b	1	0	0	0
c	1	1	0	0
d	0	0	**1**	0
e	1	1	0	0

(i)

a	1	1	1	1
b	1	0	0	0
c	1	1	0	0
d	0	0	**0**	0
e	1	1	0	0

(ii)

Table 3.1: Example of the MRF supertree method.

exists, their semi-strict consensus is the MRF supertree [Chen et al., 2003; Eulenstein et al., 2004]. Chen et al. [2003] proved that the MRF supertree displays the strict consensus supertree. Moreover, they proved that, in a consensus setting, the MRF supertree displays the semi-strict consensus tree but does not display either the majority-rule consensus or the Adams consensus trees. Finding the MRF supertree has been shown to be an NP-hard problem [Chen et al., 2002] and several heuristics

have been proposed [Chen et al., 2003, 2006; Eulenstein et al., 2004]. In the latter paper, the authors showed simulation studies for which the MRF supertrees are at least as accurate as supertrees built with MRP. Unfortunately, as MRP, this method can propose new clusters contradicted by each of the input trees [Goloboff, 2005].

3.3.2.3 The Matrix Representation using Compability (MRC) supertree

The Matrix Representation using Compability (MRC) method [Rodrigo, 1996; Ross and Rodrigo, 2004] consists in finding the maximum clique of columns of the MR matrix, where a clique of columns is defined as a set of matrix columns that are pairwise compatible. When computing the pairwise compatibility of two matrix columns, rows involving one or two missing entries are ignored. Note that, Ross and Rodrigo [2004] did not detail how to reconstruct a supertree from the so-obtained maximum clique(s). Note also that, as for the MRP method, an exponential number of solutions is sometimes possible.

This approach has several drawbacks: first of all, finding the maximum clique of a matrix is an NP-hard problem. Second, in a supertree approach, for the presence of missing entries (*i.e.*, states encoded by a '?'), pairs of matrix columns may be all pairwise compatible, but collectivelly non-compatible (see next section for an example). Consequently, this method can also propose new clusters contradicted by each of the input trees [Goloboff, 2005]. Moreover, it has been shown by simulation studies that this method performs worse than MRP [Ross and Rodrigo, 2004].

In the next section we present a better way to use compatibility in a supertree approach.

3.3.2.4 The semi-strict supertree

The first author that proposed a method analogous to the semi-strict consensus (Section 3.2.1.3) for a set of rooted trees with overlapping sets of taxa was Lanyon [1993]. His intent was to propose a method able to return supertrees including clusters supported by a subset of the input trees as long as they are not contradicted by other trees. Lanyon's method represents each tree T_i of the forest $\mathcal{F}$ by two sets of clusters: the set of *observed* clusters, denoted by $\mathcal{C}_o(T_i)$ and the set of *possible* clusters denoted by $\mathcal{C}_p(T_i)$. The set $\mathcal{C}_o(T_i)$ contains all clusters of T_i and coincides with $\mathcal{C}(T_i)$. The set of possible clusters $\mathcal{C}_p(T_i)$ contains clusters obtained by resolving polytomies of T_i in the usual way (see Section 3.1.3) and inserting taxa of $L(\mathcal{F})$ not in $L(T_i)$ in all possible ways. For example, let $\mathcal{F}$ be a forest that comprises two trees $T_1 = (((a,b),c),(d,e))$ and $T_2 = (((e,f),d),(a,c))$, we have that $\mathcal{C}_o(T_1) = \{(a,b), (a,b,c), (d,e)\}$ and $\mathcal{C}_o(T_2) = \{(a,c), (d,e,f), (e,f)\}$ while $\mathcal{C}_p(T_1) = \{(a,f), (b,f), (c,f), (d,f), (e,f), (a,b,f), (a,b,c,f), (d,e,f)\}$, and $\mathcal{C}_p(T_2) = \{(a,b), (b,c), (b,d), (b,e), (b,f), (a,b,c), (b,e,f), (b,d,e,f)\}$.

Lanyon' s supertree is composed by all clusters in $\bigcap_{T_i \in \mathcal{F}}(\mathcal{C}_o(T_i) \cup \mathcal{C}_p(T_i))$ as long as (1) it is not contradicted by any other cluster or (2) it is an observed cluster and is contradicted only by possible clusters. Goloboff and Pol [2002] points out that

Lanyon' approach does not take into account the information induced by combining input trees. For example, the Lanyon supertree for the set of trees $T_1 = ((b, d), c)$, $T_2 = ((a, b), d)$ and $T_3 = ((a, c), b)$ is the tree $((a, c), b, d)$. This tree does not contradict any of the input trees but, since it contains the triplet $ac|b$, it is in contradiction with the combination of T_1 and T_2. Indeed, the first tree contains the triplet $bd|c$ and the second tree contains the triplet $ab|d$. It follows that combining these trees we obtain the information that taxa a, b are more related to each other than either is to c. For the same reason, it can happen that the Lanyon supertree does not contain clusters that are not contained in any of the input tree but are jointly implied by the input trees. For instance, for the set of trees $T_4 = (((c, d), b), a)$ and $T_5 = ((d, e), b)$, the Lanyon supertree is the completely unresolved tree. But from the combination of T_4 and T_5 we can deduce the cluster (c, d, e), since T_4 contains the triplet $cd|b$ and T_5 the triplet $de|b$.

Ultra-cliques These remarks motivated the work of Goloboff and Pol [2002]. They proposed a method called the semi-strict supertree «*displaying* $ab|c$ *if it is found in some input tree or implied by some combination of input trees and no input tree or combination of input trees displays or implies* $ac|b$ *or* $bc|a$». We will return to these properties, called PI' and PC' [Ranwez et al., 2007a], in Section 4.1. The semis-strict supertree method first encodes trees in a matrix representation and then searches for the *ultra-clique* in the MR. An ultra-clique for the MR matrix is defined as a set of columns of MR not contradicted by any other column or sets of columns of MR. Note that no matrix can have more than one maximal ultra-clique [Goloboff and Pol, 2002].

Finding cliques of compatible matrix columns is a well known problem and many solutions have been proposed in the case of matrix columns with no missing entries. On the contrary, the evaluation of compatibility if the matrix columns have missing entries is more complicated. Indeed, when a matrix column has some missing entries, some taxa have undefined positions. This implies that different pairs of matrix columns may be compatible in pairwise comparisons, but collectively non-compatible as shown in the example of the Lanyon supertree of T_1, T_2 and T_3. This remark may strongly invalidate the Lanyon supertree method but also put the MRC method (Section 3.3.2.3) into question.

A heuristic method to find the ultra-cliques Goloboff and Pol [2002] proposed a heuristics to solve the problem of evaluating the compatibility of matrix columns with missing entries. Their method is based on the fact that the interaction with other matrix column(s) may define the state of taxa with missing entries. For instance, by combining matrix columns MR[1] and MR[2] in the matrix of Table 3.2(i), we can deduce that the only way to have the compatibility between these columns is that the taxa e in MR[1] is not in the same cluster as c and d. In this way we obtain a second matrix shown in Table 3.2(ii) that contains the same information as the first one, but with no missing entry. The algorithm combines

a	0	1
b	0	0
c	1	1
d	1	1
e	?	0

(i)

a	0	1
b	0	0
c	1	1
d	1	1
e	0	0

(ii)

Table 3.2: Example of the semi-strict supertree method.

pairwise matrix columns that belong to different trees (see [Goloboff and Pol, 2002] for further details on the induction rules). When no new cluster can be deduced, the algorithm stops and a tree is assembled, using only those matrix columns which have no incompatibilities and no missing entries. Note that the replacement of '?' states can differ depending on the pairs of columns jointly considered, hence can vary depending on the order according to which columns are considered.

Properties of the semi-strict supertree Goloboff and Pol affirmed that the semi-strict supertree contains only triplets found in some input tree or implied by some combination of input trees and are not contradicted by any input tree or combination of input trees. Moreover, they claim that the semi-strict supertree is always compatible with, but possibly less resolved than, the MRP tree. But, in pratice, these properties are not always verified, since the semi-strict supertree method is a heuristics to find the ultra-clique of the MRP matrix. The same authors show an example of a set of three rooted trees $T_1 = ((b, c), a)$, $T_2 = ((c, d), b)$ and $T_3 = ((a, d), b)$ for which the semi-strict supertree method may recover a cluster contradicted by the combination of two input trees. The MR coding for this forest is shown in Table 3.3(i). If we first combine MR[1] and MR[2] and then the modified MR[2] and MR[3] we obtain the matrix in Table 3.3(ii). Since all matrix columns are pairwise incompatible, it follows that the supertree is completely unresolved. On the contrary, if the first matrix columns to be combined are MR[1] and MR[3] followed by the modified MR[3] and MR[2], we obtain the matrix in Table 3.3(iii), containing a column without missing entries and that is not contradicted by any other columns *i.e.*, MR[3] so the returned supertree is $((a, d), b, c)$. Then the supertree contains the cluster (a, d) that is contradicted by the combination of the first and the second column in the forest. Thus, in this case the method may propose a supertree contradicted by a combination of input trees that is more resolved than the MRP tree, that here is completely unresolved.

An implementation of this method was available in a previous version of the phylogeny program TNT [Goloboff et al., 2008]. Currently, no implementation of this method is available.

3.3.2.5 The t-MRP method

Nelson and Ladiges [1994] have been the first to propose a triplet-based encoding of

a	0	?	1
b	1	0	0
c	1	1	?
d	?	1	1

(i)

a	0	1	1
b	1	0	0
c	1	1	1
d	1	1	1

(ii)

a	0	1	1
b	1	0	0
c	1	1	0
d	0	1	1

(iii)

Table 3.3: **Example for which the semi-strict supertree method recovers a contradicted cluster** - (i) the initial MR matrix. (ii) and (iii) are two matrices that can be deduced from (i) depending on column combination order.

trees in a parsimony context. This approach [Nelson and Ladiges, 1994; Wilkinson et al., 2004a, 2001; Williams and Humphries, 2003] uses a matrix representation of the source trees no longer based on bipartitions as in the previous methods but on triplets. This approach has also been called *three-item consensus* by Nelson and Ladiges [1994] and *triplet fit* by Wilkinson et al. [2005a]. Here we refer to it as the *triplet-based Matrix Representation with Parsimony* (t-MRP) from Ranwez et al. [2009].

In practice, determining the t-MRP supertree for a forest $\mathcal{F}$ consists first in computing the set $\mathcal{R}(\mathcal{F})$, then in encoding each triplet $ab|c \in \mathcal{R}(\mathcal{F})$[10] as a matrix column having state '1' for a and b, state '0' for c and the root node and state '?' for all other taxa of $L(\mathcal{F})$. The so-obtained matrix t-MR is then analyzed with the parsimony criterion. The t-MRP supertree is the strict consensus of all most parsimonious trees for M[11]. This approach has several drawbacks, as pointed out by Ranwez et al. [2009].

First of all, the number of matrix columns of t-MR is in the order of $O(|\mathcal{F}| \cdot (L(\mathcal{F})^3)$ while it is in the order of $O(|\mathcal{F}| \cdot (L(\mathcal{F}))$ for the standard MRP[12].

Second, since for each matrix column of t-MR, there are only four informative character states, the proportion of missing characters is very high and grows proportionally with the number of taxa in the forest. It is also known that a high proportion of missing character states slows down parsimony methods.

Third, the supertree returned by t-MRP is the strict consensus of all most parsimonious (fully) resolved trees and usually is not the best according to the parsimony criterion. This depends on the fact that partially resolved candidate supertrees cannot easily be compared with fully resolved supertrees using parsimony. This is why [Ranwez et al., 2009, see Section 3.3.3.3] used the triplet dissimilarity [Wilkinson et al., 2005a, 2001] to compare the trees with different degrees of resolution.

[10] In common pratice, $\mathcal{R}(\mathcal{F})$ is pre-processed in order to keep only one (weighted) representative for the many identical matrix columns.

[11] Note that Thorley and Page [2000] implements q-MRP, which uses quartet trees in a variant of the t-MRP supertree method.

[12] The number of matrix columns of t-MR is of the order of $0(L(\mathcal{F})^3)$ if only one (weighted) representative for the many identical matrix columns is kept.

3.3.2.6 The average consensus supertree or MRD

Lapointe and Cucumel [1997] proposed a consensus method called the average consensus that can be also used in the supertree setting. The average consensus is defined for sets of unrooted weighted trees *i.e.*, unrooted trees with branch lengths representing evolutionary distances through rates of evolution, divergence times, etc. For an unrooted weighted tree T, we denote by $d_T(a, b)$ the patristic distance between a and b in T *i.e.*, the sum of the lengths of the branches of T composing the unique path connecting taxa a and b. The average patristic distance of a and b in the forest $\mathcal{F}$ is

$$\hat{D}_{ab} = \frac{1}{|\mathcal{F}|} \sum_{T_i \in \mathcal{F}} d_{T_i}(a, b). \tag{3.2}$$

The average consensus tree for a forest $\mathcal{F}$ is the tree T minimizing the least squares difference:

$$\sum_{a,b \in L(\mathcal{F})} (d_T(a, b) - \hat{D}_{ab})^2.$$

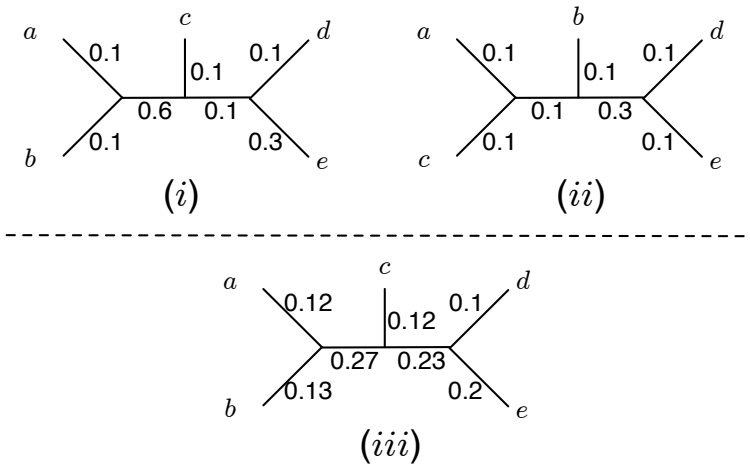


Figure 3.16: **Example of average consensus tree** - The forest $\mathcal{F}$ consists of two unrooted weighted trees (i) and (ii). The average consensus tree is depicted in (iii).

An example of average consensus tree for a forest of two unrooted trees, computed using PAUP* [Swofford, 2003], is shown in Figure 3.16(iii). Note that this forest is the same as the one used in the example 2.17 in [Bryant, 2003] but our average consensus tree differs from that of Bryant. This seems to be due to an error in the computation of $\hat{D}_{ce}$, wrongly set at 0.65 in [Bryant, 2003] (the correct value is 0.55)[13].

The average consensus method can be adapted to trees with overlapping sets of taxa. The average consensus supertree or Matrix Representation with Distances

[13]Moreover, also computing the average consensus tree with the wrong value does not lead to the same tree. The branch length set at 0.6 in Figure 1(iii) in [Bryant, 2003] looks suspicious.

(MRD) is computed as the consensus one but the computation of $\hat{D}_{ab}$ is sightly different:

$$\hat{D}_{ab} = \frac{1}{|\mathcal{F}_{a,b}|} \sum_{T_i \in \mathcal{F}_{a,b}} d_{T_i}(a,b). \tag{3.3}$$

where $\mathcal{F}_{a,b} = \{T \in \mathcal{F} | \{a,b\} \subseteq L(T)\}$. Lapointe et al. [2003] proved that, when all input trees are defined on the same leaf set and all branch lengths are set to 1, MRD and MRP are very tied.

The main problem of the average consensus tree is that there exists no efficient algorithm for constructing it. Another drawback is that no one has demonstrated, even in the consensus setting, that this method returns trees that contain all splits common to all input trees. Variants to this method have been proposed to avoid that rapid genes dominate the computation of the average supertree [Lapointe and Levasseur, 2004; Lapointe and Cucumel, 1997] but they seem to be inaccurate with more than two trees [Lapointe and Levasseur, 2004]. Recently, Criscuolo et al. [2006] proposed SDM, a distance-based method that answers the limitations of the average supertree method. First, SDM deforms the distance matrices obtained from input weighted trees, without modifying their topological message, to bring them as close as possible to each other; the so-obtained matrices are then averaged to obtain a unique distance matrix used to build the supertree.

3.3.3 Median supertrees

This set of supertree methods aims to summarize a collection of phylogenetic trees in a median tree, *i.e.*, the tree minimizing the sum of distances to the source trees.

3.3.3.1 The Most Similar Supertree (MSSA), the Maximum Quartet Fit (QFIT) and Maximum Splits Fit (SFIT) supertrees

Given a forest of trees (rooted or not) the Most Similar Supertree (MSSA), the Maximum Quartet Fit (QFIT) and Maximum Splits Fit (SFIT) supertree methods [Creevey et al., 2004; Creevey and McInerney, 2005] all search for the supertree T minimizing

$$\Delta(T, \mathcal{F}) = \sum_{T_i \in \mathcal{F}} d(T|_{L(T_i)}, T_i), \tag{3.4}$$

i.e., minimizing the sum of the distances between the gene tree T_i and the homeomorphic subtrees of T induced by the leaves of T_i. The three methods only differ on the choice of the distance metric d in equation (3.4).

Let $p_T(a,b)$ denote the number of nodes separating the taxa a and b on the tree T. In the MSSA method, the distance $d(T|_{L(T_i)}, T_i)$ is defined as:

$$d(T|_{L(T_i)}, T_i) = \sum_{a,b \in L(T_i)} |p_{T|_{L(T_i)}}(a,b) - p_{T_i}(a,b)|, \tag{3.5}$$

i.e., as the path length distance under the L^1-norm [Steel and Penny, 1993; Williams and Clifford, 1971] between the trees $T|_{L(T_i)}$ and T_i. In the Clann software [Creevey

and McInerney, 2005] the user can also impose several weighting schemes on this score to avoid an undue influence of large trees. Creevey and McInerney [2005] affirmed that this method is related to the average consensus method (Section 3.3.2.6) when branch lengths are set to unity.

For the SFIT supertree method, $d(T|_{L(T_i)}, T_i)$ is defined as the Robinson and Foulds distance (see Section 3.2.1.2) between $T|_{L(T_i)}$ and T_i.

For the QFIT supertree method, $d(T|_{L(T_i)}, T_i)$ is defined as the quartet distance [Estabrook et al., 1985] between $T|_{L(T_i)}$ and T_i *i.e.*, the number of sets of four species for which the quartet topologies differ in the two trees.

Creevey et al. did not mention whether these problems are NP-hard or not but we suppose that it is the case. In practice, heuristic searches of the tree-space have been proposed for all these methods [Creevey and McInerney, 2005].

3.3.3.2 Majority-rule supertree

Cotton and Wilkinson [2007] tried to extend the majority-rule consensus method to the supertree setting retaining as many of its appealing qualities as possible. They defined two supertree methods: the majority-rule(-) and the majority-rule(+) supertree methods that we note here as MajR$^-$ and MajR$^+$ respectively.

They defined a median$^-$ supertree for a forest $\mathcal{F}$ of trees (rooted or not) as the supertree that minimizes

$$\sum_{T_i \in \mathcal{F}} d_S(T|_{L(T_i)}, T_i) \tag{3.6}$$

over all supertrees for $\mathcal{F}$, where the distance d_S is the Robinson-Foulds or symmetric-difference distance (see Section 3.2.1.2). The MajR$^-$ supertree is the strict consensus of all median$^-$ supertrees. The main drawback of this approach is that finding a median$^-$ supertree is an NP-hard problem [Bryant, 1997].

Define the *binary supertree span* of an input tree T, denoted by $< T >$, to be the set of binary trees on $L(\mathcal{F})$ that display T. A *representative selection* for a forest $\mathcal{F} = \{T_1, \cdots, T_k\}$ is a k-tuple $H = \{T'_1, \cdots, T'_k\}$, where $T'_i \in < T_i >$. The median score of H, denoted by $s(H)$, is defined as:

$$s(H) = min_T\big(\sum_{T'_i \in H} d_S(T, T'_i)\big), \tag{3.7}$$

where T ranges over all trees with leaf set $L(\mathcal{F})$. The candidate supertree associated with H, denoted by T_H, is the majority-rule consensus tree for H. The MajR$^+$ supertree is the strict consensus of all the supertrees $T_{H'}$ associated with H' with $s(H')$ minimum over all possible tuples H. The main drawback of this approach is that it may require the enumeration of an exponential number of representative selections H.

Properties To investigate properties of those two methods, more notations are needed. A split is *full* with respect to a tree T if its leaf set is $L(T)$, otherwise it is partial. A split is said to be *plenary* with respect to a forest of trees $\mathcal{F}$ if its leaf set

is $L(\mathcal{F})$. A split is a *majority* split if it is displayed by a majority of the input trees. A split $A|B$ *extends* another split $C|D$ if $A \supseteq C$ and $B \supseteq D$ or $A \supseteq D$ and $B \supseteq C$.

Cotton and Wilkinson [2007] conjectured that, for each forest $\mathcal{F}$, both the MajR$^-$and the MajR$^+$ supertrees, denoted here by T, had the following desirable properties:

CW1: All majority plenary splits in $\mathcal{F}$ are in T.

CW2: T is compatible with each majority partial split in $\mathcal{F}$.

CW3: All splits in T are compatible with a majority of the trees in $\mathcal{F}$.

CW4: Every plenary split in T extends at least one input tree full split.

Dong and Fernandez-Baca [2009] demonstrated that the MajR$^-$ supertrees satisfy CW1 and CW4 while MajR$^+$ supertrees satisfy CW1, CW2 and CW3. Moreover, they proved that the MajR$^-$ supertree method in the consensus setting is equivalent to the majority-rule consensus while the MajR$^+$ supertree method is not. Additionally, Dong and Fernandez-Baca [2009] proposed two variants of the MajR$^+$ supertree method *i.e.*, the majority-rule $(+)_s$ and the majority-rule $(+)_g$ supertree methods, both satisfying all properties CW1-CW4. Note that only the majority-rule $(+)_g$ supertree method is equivalent to majority-rule consensus method in the consensus setting.

3.3.3.3 The SUPERTRIPLETS method

Ranwez et al. [2009] recently proposed a new method, called SUPERTRIPLETS that aims at finding the asymmetric median supertree according to triplet dissimilarity [Wilkinson et al., 2005a, 2001]. This criterion better allows comparison of trees with different degrees of resolution than the parsimony one. The SUPERTRIPLETS method consists in four steps: i) input trees are encoded as a set of weighted triplets, ii) a starting binary supertree is proposed by an agglomerative procedure, iii) the candidate binary supertree is iterativly improved using small topological changes, and iv) unsupported edges of the binary supertree are collapsed.

Simulations studies showed that SUPERTRIPLETS tends to propose less resolved but more reliable supertrees than those inferred by MRP.

3.3.4 Other approaches to the supertree problem

In the previous sections we reviewed several supertree methods, trying to cover at least the most widespread and the most theoretically appealing.

Since this field has known a substantial development over the past decades, several other approaches to the supertree problem have been proposed. Here we do not detail them since their use in phylogenomics is not established yet.

The supertree methods described in the next two sections may propose non-plenary supertrees. Recall that a non-plenary supertree T for a forest $\mathcal{F}$ is a tree T such that $L(T) \subset L(\mathcal{F})$ *i.e.*, T can lack some taxa of the forest $\mathcal{F}$.

3.3.4.1 The SMAST and the SMCT supertrees

The SMAST and SMCT [Berry and Nicolas, 2004; Jansson et al., 2004] methods are extensions of the MAST and MCT methods respectively (see Section 3.2.1.7) that allow the input trees to have different label sets.

The computational problem behind SMAST coincides with that of MAST except that a maximum agreement *supertree* is sought instead of a maximum agreement *subtree*:

Definition 3.3.4 *Given a forest of trees $\mathcal{F}$, an agreement supertree T is a tree such that $L(T) \subseteq L(\mathcal{F})$ and $T|_{L(T_i)} = T_i|_{L(T)}$ $\forall T_i \in \mathcal{F}$.*

A maximum agreement supertree for a forest $\mathcal{F}$ is an agreement supertree for $\mathcal{F}$ of maximum size.

The SMCT problem consists in finding the maximum compatible supertree for a forest $\mathcal{F}$ *i.e.*, a tree T such that $T|_{L(T_i)}$ refines all trees $T_i|_{L(T)}$ $\forall T_i \in \mathcal{F}$ and has the maximum number of leaves.

These two methods can be used to measure the congruence of a collection of source trees to be combined into a supertree. They can also be used as *seed trees* to improve the accuracy of MRP when the input trees overlap moderately [Emonds and Sanderson, 2001]. Moreover, the SMAST can be used to detect HGTs in the supertree setting.

Extending the MAST and the MCT to the supertree problem increases the complexity of both problems [Berry and Nicolas, 2004, 2007; Guillemot and Berry, 2009; Guillemot and Nicolas, 2006; Hoang and Sung, 2008, 2009]. Complexities for these problems are mainly expressed in terms of the total number n of distinct labels appearing in the source trees, and the number k of input trees. These problems involve several other natural parameters *e.g.* d, the maximum outer degree (number of children) of a node in an input tree (when considering rooted input trees) and p, an upper bound on the number of input labels that are missing in a SMAST (resp. SMCT) solution. The SMAST problem is NP-hard as it generalizes the MAST problem [Amir and Keselman, 1997]. It remains NP-hard when the outer degree d is unrestricted for $k \geq 3$ input trees [Jansson et al., 2005], and for trees with $d \geq 2$ when k is unrestricted [Berry and Nicolas, 2007; Jansson et al., 2005]. The SMCT problem in NP-hard for 2 trees from the result of Hein et al. [1996]. When $k = 2$, SMAST and SMCT can be solved in polynomial time by reduction to MAST and MCT respectively [Berry and Nicolas, 2007; Jansson et al., 2005]. A sufficient condition for SMAST to be solved by resorting to MAST algorithms is also given in Berry and Nicolas [2007]. For such cases, Berry and Nicolas [2007] provided an algorithm for solving SMAST in linear time.

Both SMAST and SMCT problems parameterized in p have been shown to be $W[2]$-hard [Berry and Nicolas, 2007], which rules out the possibility of an FPT algorithm for this parameterization of the problem.

More recently, Guillemot and Berry [2009] considered the SMAST problem for *binary* trees, for which SMAST and SMCT problems coincide. They gave an al-

gorithm that solves SMAST on k rooted binary trees on a label set of size n in $O((2k)^p kn^2)$ time. This algorithm is thus exponential only in p, that roughly represents the extent to which the input trees disagree, *i.e.*, it will be reasonably fast when dealing with trees displaying a low level of conflict. Alternatively, Guillemot and Berry [2009] provided an $O((8n)^k)$ time algorithm, independent of p. This is a significant improvement on the $O(n^{3k^2})$ time algorithm of Jansson et al. [2005] and shows that SMAST is tractable for a small number of trees. Moreover, Guillemot and Berry [2009] showed that SMAST is FPT for complete collections of triplets, i.e., when there is at least one triplet for each set of three taxa.

3.3.4.2 Reduced supertree methods

Wilkinson proposed reduced versions of the strict, majority-rule, semi-strict and Adams consensus trees (see Wilkinson and Thorley [2003] for a review of these methods). These reduced versions may return multiple consensus trees that need not to include all the leaves in the input trees. Wilkinson [1998] suggested to apply the same approach to the supertree problem but, as far as we know, no method to extend reduced consensus in a supertree setting exist.

3.3.4.3 Maximum Likelihood supertree

Steel and Rodrigo [2008] proposed an approach to obtain maximum likelihood (ML) estimates of supertrees. Their method is based on an exponential model of phylogenetic error in which the probability of reconstructing any tree T' on any taxon set Y given a generating tree T (where $Y \subseteq L(T)$), denoted by $\mathbb{P}_{T',Y}(T)$, falls off exponentially with its distance from T *i.e.*,

$$\mathbb{P}_{T',Y}(T) = \alpha \exp(-\beta \cdot d(T', T|_Y)). \tag{3.8}$$

where d is a metric on resolved trees and β is a constant that can vary with the size of Y and other factors *e.g.*, the quality of the data. The constant α ensures that $\sum_{T'} \mathbb{P}_{T',Y}(T) = 1$. For this model, the ML supertree for a forest of trees $\mathcal{F} = \{T_1, \cdots T_k\}$ given a metric d and a vector of weights $\{\beta_1, \cdots \beta_k\}$, is the tree T minimizing the weighted sum:

$$\sum_{T_i \in \mathcal{F}} \beta_i \cdot d(T_i, T|_{L(T_i)}). \tag{3.9}$$

Note that the ML supertree may not be unique.

Steel and Rodrigo [2008] suggested that the choice of the d metric should be guided by the biological context and computational considerations. The authors proved that the ML procedure is statistically consistent as the number of input trees grows. Moreover they proved that, when d is the nearest-neighbor-interchange (NNI) metric, and the β_i values are all the same, the ML supertree coincides with the MajR$^-$ supertree (Section 3.3.3.2) while, in the consensus tree setting, when d is the Robinson-Foulds metric, the consensus of the ML supertrees is the same as the majority-rule consensus tree (Section 3.2.1.2).

3.3.4.4 Bayesian supertree

Ronquist et al. [2004] have developed a Bayesian approach to supertree construction. Because of the huge number of possible trees, it is usually not feasible to estimate the probability of each of them. Therefore, Bayesian supertree methods summarize the distribution typically in terms of split frequencies that are then used to compute tree probabilities.

3.3.4.5 Gene tree parsimony

Maddison [1997]; Page and Charleston [1997a,b]; Slowinski et al. [1997] described a procedure, called *gene tree parsimony* in Slowinski et al. [1997], that aims at finding the supertree that minimizes a weighted sum of deep coalescences, duplications, loss and transfer events necessary to explain the differences between the input trees and the supertree (see Chapter 2 for a recall of these macro events).

Optimization procedures for deep coalescence have been discussed [*e.g.,* Maddison, 1997; Slowinski et al., 1997]. Moreover, several methods that aim at minimizing the number of transfers and/or duplication and loss events have also been proposed [*e.g.,* Chauve et al., 2008; Chauve and El-Mabrouk, 2009; Chen et al., 2000; Hallett and Lagergren, 2000; Ma et al., 2000; Slowinski and Page, 1999; Vernot et al., 2008].

Several of these methods accept as input multi-labeled phylogenetic trees on which we focus in Chapter 5. In that chapter we will propose a new approach to combine such kinds of trees.

3.3.4.6 Quartet supertrees

Piaggio-Talice et al. [2004] have proposed two quartet-based supertree methods: the Quartet Local Inconsistency (QLI) supertree method and the Quartet Inference and Local Inconsistency (QILI) supertree method. The QLI supertree method consists first in applying the local-inconsistency quartet method of Willson [1999] to weighted quartet trees obtained from the input trees $\mathcal{F}$. Willson's method consists in picking a random order of the species in $L(\mathcal{F})$ and adding the species in this order. Each species is inserted in the phylogeny at the placement with the lowest *local inconsistency*, where the local inconsistency that results from placing a species into a particular position in a phylogenetic tree is computed as shown in Willson [1999], using quartet weights.

The QILI supertree method consists in inferring missing quartet trees using the rectifying process for quartet trees proposed by Willson [2001] and then in applying the local-inconsistency quartet method of Willson [1999] to weighted quartet trees.

3.4 Which method to choose?

The choice of which consensus or supertree method to use is partly dependent on the question being asked. For instance, in the consensus setting, the strict and

semi-strict consensus methods present the relationships that are common to or uncontradicted among, respectively, the set of source trees. As such, they provide a conservative summary of the information common to a set of source trees. On the other hand, the Adams consensus can be used to detect finer common statements of relationship among a set of source trees (*e.g.*, a and b are more closely related than either is to c, where a, b, and c need not be each other's closest relatives). Moreover this method, like the MAST can be used to detect rogue taxa.

The same reasoning applies to the supertree context. When using supertree construction in a divide-and-conquer approach in the attempt to reconstruct large portions of the Tree of Life, conservative supertree methods have to be preferred in order to obtain very reliable supertrees. In our opinion a reliable supertree should display only information that is present in one or several input trees, or induced by their interaction and, at the same time, that is not in conflict either directly with a source tree or indirectly with a combination of them. Since no existing supertree method has these characteristics, we designed two new supertree methods that are very useful in a conservative framework like the reconstruction of the Tree of Life. These methods are presented in the next chapter.

Chapter 4

Supertree methods from new principles

Contents

This chapter focuses on the work done during my PhD on the design of supertree methods with good theoretical properties.

As evoked in Chapter 2, supertree methods can be used in a divide-and-conquer approach in the attempt to reconstruct large portions of the Tree of Life. This approach consists in decomposing a very large phylogenetic problem into many subproblems that are analyzed separately. Later on, the solutions of the smaller are combined through a supertree method to derive the global answer of the starting problem. When combining reliable published trees in view of reconstructing large portions of the Tree of Life, conservative supertree methods have to be preferred in order to obtain reliable supertrees.

In the first part of this chapter we present two strict and desirable properties that a conservative supertree method should satisfy. In sections 4.2 and 4.3 we present two supertree methods conceived during my PhD *i.e.*, *PhySIC* and *PhySIC_IST*

[Ranwez et al., 2007a; Scornavacca et al., 2008] that infer supertrees satisfying these desirable properties. Finally, in Section 4.4 we present an application of *PhySIC_IST* to the complex problem of disentangling the phylogeny of Triticeae.

4.1 The PI and PC properties

A conservative supertree method should avoid arbitrary resolutions, *i.e.*, resolutions that are not entailed by the source topologies. Indeed, novel relationships displayed by a supertree «*are worrying if they are not implied by combinations of the input trees*» [Wilkinson et al., 2005b]. This is why we believe that a conservative supertree method should return a supertree such that every piece of phylogenetic information displayed in the supertree is present in one or several source topologies, or induced by their interaction; we call this the *induction property*.

Moreover, we think that a conservative supertree method has to construct supertrees not containing clusters that conflict either directly with a source tree or indirectly with a combination of them. We call this the *non-contradiction* property.

To formally define the induction and the non-contradiction properties, we need to introduce some further notations. In this chapter we will make an extensive use of the notations presented in sections 3.1 and 3.3.1.1.

Given a compatible set $\mathcal{R}$ of triplets, we say that $\mathcal{R}$ *induces* a triplet t, denoted by $R \vdash t$, if and only if $R \cup \{\bar{t}\}$ is not compatible, or equivalently if any tree T that displays $\mathcal{R}$ contains t. For instance, any tree displaying $\{ab|c, bc|d\}$ also displays the triplet $ac|d$ so we have that $\{ab|c, bc|d\} \vdash ac|d$. Bandelt and Dress [1986] and Dekker [1986] were among the first to investigate such induction rules. The set of all triplets induced by a compatible set $\mathcal{R}$ is called the *closure* of $\mathcal{R}$ and is denoted by $cl(\mathcal{R})$. Since a forest of input trees $\mathcal{F}$ is often incompatible, it follows that this is also the case for the set $\mathcal{R}(\mathcal{F})$. In case of an incompatible set of triplets $\mathcal{R}$, we say that a set $\mathcal{R}$ of triplets *induces* a triplet t when there is a compatible subset $\mathcal{R}'$ of $\mathcal{R}$ that induces t.

Given a collection $\mathcal{F}$ of input trees and a candidate supertree T, $\mathcal{R}(T, \mathcal{F})$ denotes the set of triplets of $\mathcal{F}$ for which T proposes a resolution. More formally, $\mathcal{R}(T, \mathcal{F}) = \{ab|c \in \mathcal{R}(\mathcal{F})$ such that $\{ab|c, ac|b, bc|a\} \cap \mathcal{R}(T) \neq \emptyset\}$. The set $\mathcal{R}(T, \mathcal{F})$ corresponds to all topological information present in the collection $\mathcal{F}$ that is related to the information present in supertree T. Using this notation, we can express the induction property PI and the non-contradiction property PC as follows:

- T satisfies PI for $\mathcal{F}$ if and only if for all $t \in \mathcal{R}(T)$, it holds that $\mathcal{R}(T, \mathcal{F}) \vdash t$. In other words, PI requires that each and every triplet of T is induced by $\mathcal{R}(T, \mathcal{F})$.

- T satisfies PC for $\mathcal{F}$ if and only if for all $t \in \mathcal{R}(T)$ and all $\bar{t}$, it holds that $\mathcal{R}(T, \mathcal{F}) \nvdash \bar{t}$. This means that, for each and every triplet of T, $\mathcal{R}(T, \mathcal{F})$ induces no alternative resolution.

Links with other advocated properties

Properties similar to PI and PC were described in Goloboff and Pol [2002]. Using our formalism, they can be translated as follows for a supertree T representing a collection $\mathcal{F}$:

- PI': for any $t \in \mathcal{R}(T)$, it holds that $\mathcal{R}(\mathcal{F}) \vdash t$
- PC': for any $t \in \mathcal{R}(T)$ and for all $\bar{t}$, it holds that $\mathcal{R}(\mathcal{F}) \nvdash \bar{t}$.

These properties were also pointed out as being desirable by Grunewald et al. [2007]. The essential difference between PI'-PC' and PI-PC is whether we evaluate supertrees based on triplets in the original set of trees, $\mathcal{R}(\mathcal{F})$, or on the triplets commonly resolved by the supertree and at least one of the source trees, $\mathcal{R}(T, \mathcal{F})$. From the statement of the properties, it is clear that PC' implies PC and PI implies PI'. It is thus natural to wonder which version of the properties is preferable. Below, we show an example where PC' is too restrictive, and an example where PI' is too permissive. In contrast, PI and PC behave correctly in these examples.

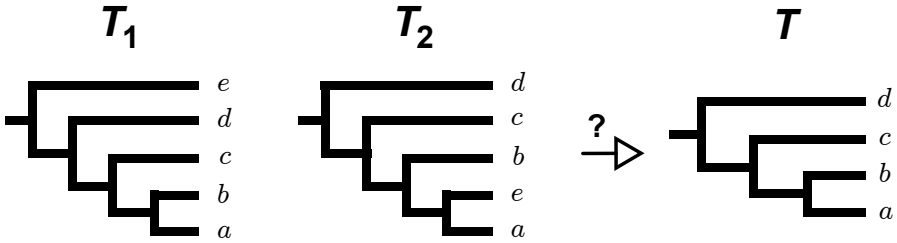


Figure 4.1: **An example of informative non plenary supertree for a forest of two rooted trees** - Excluding rogue taxa from the analysis can lead to more informative supertrees.

Let $\mathcal{F} = \{T_1, T_2\}$ with T_1 and T_2 as shown in Figure 4.1. $\mathcal{R}(\mathcal{F})$ contains $ae|b$ and $ac|e$, therefore $\mathcal{R}(\mathcal{F}) \vdash ac|b$. We also have $\mathcal{R}(\mathcal{F}) \vdash ab|c$ since $ab|c \in \mathcal{R}(T_1)$. Thus any tree providing a triplet on $\{a, b, c\}$ does not satisfy PC'. For analogous reasons PC' does not allow us to propose any triplet in the supertree. Thus PC' rejects the tree T of Figure 4.1. Yet T is a reasonable and informative supertree for $\mathcal{F}$ and satisfies both PI and PC.

We note that T is not a plenary supertree, *i.e.*, it does not contain all input taxa, but this example shows that removing rogue taxa is a way in which more informative supertrees can be obtained. This is in line with the remark of Wilkinson et al. [2004b], who stated that «*non-plenary supertree methods might be most useful for identifying unstable leaves*». For instance, such leaves might be involved in horizontal gene transfers.

The same remark holds for the forest $\mathcal{F} = \{T_1, T_2\}$ and the supertree T shown in Figure 4.2. In this example, though taxa are excluded from the supertree, this latter contains more taxa than any of the individual input trees.

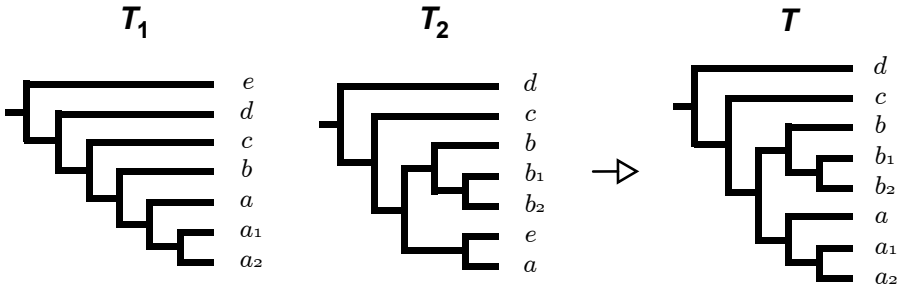


Figure 4.2: **Another example of informative non plenary supertree for a forest of two rooted trees** - Excluding rogue taxa from the analysis can lead to more informative supertrees.

The next example shows a supertree satisfying both PI' and PC', while also displaying irrelevant triplets.

Let $\mathcal{F} = \{T_1, T_2\}$ with T_1 and T_2 as illustrated in Figure 4.3. $\mathcal{R}(\mathcal{F}) = \{ab|c, ab|x, bc|a\}$. The tree T in Figure. 4.3 displays $\{ab|x, bc|x, ac|x\}$. The triplet $ab|x$ is present in (thus induced by) $\mathcal{R}(\mathcal{F})$ but surprisingly the two other triplets can also be induced from $\mathcal{R}(\mathcal{F})$: $\{ab|x, bc|a\} \vdash \{bc|x, ac|x\}$. It follows that T satisfies PI'. Note that this induction is done using the triplet $bc|a$ that is a unreliable since $\mathcal{R}(\mathcal{F})$ contains both $bc|a$ and $ab|c$. Indeed PI' could even have relied on $bc|a$ to justify a triplet of the supertree and on $ab|c$ to justify another triplet of the same supertree. Moreover, it is easily seen that no combination of triplets in $\mathcal{R}(\mathcal{F})$, other than $\{ab|x, bc|a\}$, induces triplets. Thus T also satisfies PC'. However, T is clearly not an ideal supertree for $\mathcal{F}$ as no information in $\mathcal{F}$ induces group a, b, c to nest inside group a, b, c, x. The property PI, not satisfied by T, detects this problem: here $\mathcal{R}(T, \mathcal{F})$ only contains the triplet $ab|x$ and thus it does not induce the triplet $ac|x$ present in T.

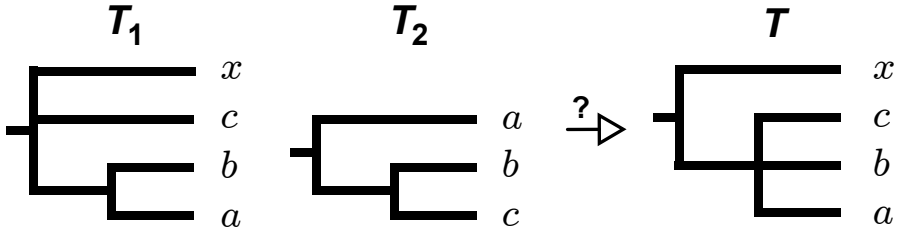


Figure 4.3: **An example showing why properties PC and PI have to be preferred to properties PC'+ PI'** - Contradictions in the source trees can lead to arbitrary resolution. An example where the presence of contradictions in the source trees (namely, $ab|c$ in T_1 versus $bc|a$ in T_2) can lead to the inferrence of arbitrary clades (namely, excluding x from the clade a, b, c in the supertree T).This problem is detected by PI but not by PI' nor PC'.

The PI' property quoted by Goloboff and Pol [2002] is stronger than the *Pareto* property [Neumann, 1983; Wilkinson et al., 2004b] on triplets, which requires that the output tree contains all triplets that occur in all source trees.

The Pareto property is appealing in general and has also been advocated in the supertree context (property P6 of Steel et al., 2000). However imposing the Pareto property on triplets may be problematic, even in the case of compatible source trees [Thorley and Wilkinson, 2003]. This is due to the possibility of having *several* candidate supertrees that are both compatible with source trees and respect the Pareto property. In this case, no *single* supertree exists that satisfies the Pareto property while having no arbitrary resolution. The strict consensus of these supertrees does not necessarily satisfy the Pareto property. A solution is then to return several trees, either all candidate supertrees or their reduced consensus [Wilkinson, 1994]. However, this solution may not well be suited when the aim is to summarize a collection of source trees into a single supertree that is more easily dealt with for further analysis by biologists.

When source trees are incompatible, it may even be impossible to have a supertree satisfying both the Pareto and non-contradiction properties (PC and PC'). The following details such a surprising example. Consider the collection $\mathcal{F} = \{T_1, T_2\}$ where $T_1 = (((a,d),b),((c,f),e))$ and $T_2 = (((a,e),(b,f)),(c,d))$. Triplets $ab|c$ and $ef|d$ are displayed by both trees of $\mathcal{F}$. Thus any supertree T for $\mathcal{F}$ must include all leaves in $\mathcal{F}$ in order to satisfy the Pareto property. Since $\mathcal{R}(\mathcal{F})$ contains $ab|d$ and $ad|b$, any tree T displaying a triplet for the three leaves does not satisfy PC (hence PC'). For similar reasons, no supertree T can display a triplet on the taxa a, c and d. Thus, any supertree satisfying PC (or PC') and including all taxa of $\mathcal{F}$ contains a multifurcating node on taxa a, b, c, d, hence does not display the triplet $ab|c$, *i.e.*, does not satisfy the Pareto property.

In other words, imposing the Pareto property can lead the supertree to explicitly contradict relationships present in some input trees. This shows that the Pareto property on triplets is not compatible with the veto approach, where the proposed supertree must not contradict the source trees. However, the Pareto property can be considered for other topological relationships [Wilkinson et al., 2004b]. For example, there is always a supertree satisfying PI and PC as well as the Pareto property on partial or full splits contained in the source trees.

The Pareto property specifies relations that the supertree *must* contain. The complementary co-Pareto property specifies relations that the supertree *must not* contain. The co-Pareto property in the consensus context requires that the consensus tree contain no relationships that are not present in at least one input. However, Wilkinson et al. [2004b] pointed out that this statement is not reasonable for supertrees, since «*they might contain relashionships that are entailed by the input trees in combination, but are not present in any of them singly*». Then they propose a weaker version that requires that the supertree does not contain relationships that are contradicted by all the input trees whose leaf set makes a contradiction possible. Note that, any supertree satisfying PC also satisfies the latter version of co-Pareto.

Steel et al. [2000] list five other properties that might be requested from supertree methods: changing the order of the trees in the input collection does not change the supertree (P1); renaming the taxa of the source trees gives the same supertree, but with the taxa renamed accordingly (P2); the output tree displays the source trees when they are compatible (P3); each leaf (taxon) that occurs in at least one source tree is in the supertree (P4); the running time of the method grows polynomially with respect to the total number of taxa (P5). First note that any non-plenary supertree method does not satisfy the P4 property. The following example shows that ensuring P3 can force the supertree to contain arbitrary clades. Thus P3 can conflict with PI.

Let $\mathcal{F} = \{T_1, T_2\}$ with $T_1 = ((a, b), w)$ and $T_2 = ((a, b), (x, (y, z)))$. A supertree with taxon set $\{a, b, w, x, y, z\}$ that satisfies $P3$ must display T_2, hence must have a clade including y, z but not x. However, it will contain arbitrary clades, no matter where taxon w is attached. This is because any supertree satisfying PI must include a polytomy on w, x, y, z since source trees include no information on the relative position of w and the group x, y, z. Note that if polytomies of a supertree are interpreted in terms of an Adams consensus (see Section 3.1.3), then this example does not put P3 into question. However, this interpretation of polytomies does not prevail in phylogenetics, as evoked in Section 3.1.3.

Both supertree methods presented in this chapter compute supertrees satisfying PI and PC properties, but with different underlying optimization problems.

4.2 Phylogenetic Signal with Induction and non-Contradiction (PhySIC)

The aim of *PhySIC* is to infer supertrees that satisfy PI and PC and that resolve as many triplets as possible. More formally, given a forest $\mathcal{F}$, the aim of *PhySIC* is to infer a supertree T that satisfies PI and PC and that such that $|\mathcal{R}(T, \mathcal{F})|$ is maximum. This gives rise to the following optimization problem:

Problem	Maximum Induced and non-Contradicting Tree from a Forest (MICTF)
Input	a collection $\mathcal{F}$ of rooted trees.
Output	a tree T such that: (i) T satisfies PI and PC for $\mathcal{F}$ (ii) $\|\mathcal{R}(T, \mathcal{F})\|$ is maximum among the trees satisfying (i).

We conjecture this problem to be hard. A proof of NP-completeness has been proposed in Guillemot and Berry [2007] but, during the redaction of this manuscript we realized that the problem studied by the authors - MIST (Maximum Identifying Subset of rooted Triplets) - is a variant of the problem underlying *PhySIC* not involving the NP-completeness of the latter. The method *PhySIC* is a heuristic for the afore-described problem, but only on the size of $\mathcal{R}(T, \mathcal{F})$ as it always returns a super-trees satisfying PI and PC.

This method consists in two steps. Given a forest of rooted trees $\mathcal{F}$, first a supertree T_{PC} satisfying PC for $\mathcal{F}$ is computed by the $PhySIC_{PC}$ algorithm (detailed in Algorithm 11 of Appendix A.1). Second, some branches of T_{PC} are eventually collapsed by the $PhySIC_{PI}$ algorithm (detailed in Algorithm 14 of Appendix A.1) until the so-modified T_{PC} satisfies also property PI.

4.2.1 The $PhySIC_{PC}$ algorithm

A simple algorithm that infers a supertree from a collection of source trees $\mathcal{F}$ satisfying PC can be obtained modifying the Build algorithm (Section 3.3.1.1). This algorithm, called $Build_{PC}$ (see Algorithm 12 in Appendix A.1), takes as input the triplet set $\mathcal{R} = \mathcal{R}(\mathcal{F})$ of a collection $\mathcal{F}$ of source trees and the list S of taxa contained in these trees *i.e.*, $L(\mathcal{F})$. $Build_{PC}$ mainly differs from Build when the Aho graph contains one connected component on the set S of taxa currently considered. In this case, $Build_{PC}$ returns the star tree on S (*i.e.*, a single polytomy on S, thus contradicting no input triplet), whereas Build simply concludes that the sources trees are incompatible. This star tree is then grafted as a subtree of the tree built by the previous recursive call. Thus, we can now output a supertree even when the source trees are incompatible. The correctness of $Build_{PC}$ is proved in Ranwez et al. [2007a].

$Build_{PC}$ sometimes produces poorly resolved trees due to multifurcations returned in cases where $\mathcal{G}(\mathcal{R}|_S, S)$ contains a single connected component (*i.e.*, when $\mathcal{R}$ contains conflicts covering the considered subset of taxa S). In the most extreme (though unlikely) case, this situation occurs at the first step of the algorithm, which then outputs a star tree.

The $PhySIC_{PC}$ algorithm is a more complex variant of Build that returns supertrees generally much more resolved than those returned by $Build_{PC}$. The $PhySIC_{PC}$ algorithm takes as input a set S of taxa and a set $\mathcal{R}$ of triplets on S as input and returns a tree T_{PC} satisfying PC for $\mathcal{R}$. This algorithm is based on the remark that the most basic conflicts between triplets of $\mathcal{R}$ occurs when two different triplets t and $\bar{t}$ appear in $\mathcal{R}$ for a same set of three taxa. Such a direct contradiction cannot be present in a tree that satisfies PC. Given $\mathcal{R}_{dc}$, the set of triplets s.t. $t, \bar{t} \in \mathcal{R}$ it seems relevant to consider the subset $\mathcal{R}' = \mathcal{R} - \mathcal{R}_{dc}$

For instance, Figure 4.4(ii) shows the graph obtained for $\mathcal{R}|V(C_2)$, where $\mathcal{R}$ are triplets of the collection of rooted trees $\mathcal{F}$ comprised of two rooted trees $(((a,c),b),(e,f))$ and $(((a,d),b),c)$ and C_2 is the connected component shown in Figure 4.4(i). This graph is connected due to the direct conflicts between $ab|c$ (displayed by the first tree) and $bc|a$ (displayed by the second tree). This situation leads $Build_{PC}$ to return a polytomy on a, b, c, d.

In contrast, building the graph on the basis of $\mathcal{R}'$ results in two connected components, C_i and C_j, allowing $PhySIC_{PC}$ to propose a tree with two subtrees for taxa a, b, c, d. This contrasts with the situation for $Build_{PC}$, which can only output a star tree on a, b, c, d since its corresponding graph is connected (see Figure 4.4(i)).

The correctness of $Build_{PC}$ ensures that T' satisfies PC with respect to $\mathcal{R}'$ but

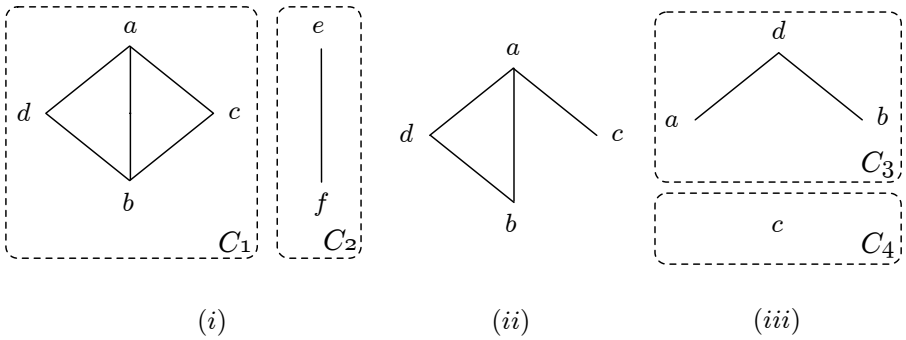


Figure 4.4: **An example of the $PhySIC_{PC}$ algorithm.** - (i) The initial Aho graph created from the triplets $\mathcal{R}(\mathcal{F})$ of the collection of rooted trees $\mathcal{F}$ that comprises $(((a,c),b),(e,f))$ and $(((a,d),b),c)$. The two connected components of this are $C_1 = \{e,f\}$ and $C_2 = \{a,b,c,d\}$. (ii) the Aho graph obtained from $\mathcal{R}|V(C_2)$. This graph is connected, showing that the input trees conflict on the resolution of $\{a,b,c,d\}$, hence are incompatible. (iii) the Aho graph obtained from $\mathcal{R}|V(C_2)$ when removing the triplets $\mathcal{R}_{dc} = \{ab|c, ac|b\}$

without any guarantee that this also holds w.r.t. $\mathcal{R}$. To ensure the latter, and thus the correction of $PhySIC_{PC}$, T' must not resolve any triplet of $\mathcal{R}_{dc}$. A way to ensure this is to collapse any branch of T' that resolves a triplet of $\mathcal{R}_{dc}$ (lines 14-24 of Algorithm 11 of Appendix A.1). The tree thus obtained is still always at least as resolved as the one proposed by $Build_{PC}$ and potentially contains supplementary branches. Indeed, direct contradictions at the root of a clade no longer prevent the proposition of clades on subsets of its taxa. For instance, on the collection of rooted trees $\mathcal{F}$ consisting of trees $(((a,c),b),(e,f))$ and $(((a,d),b),c)$, the tree satisfying PC for $\mathcal{R}'$ obtained by $PhySIC_{PC}$ is $((((a,d),b),c),(e,f))$. But as the branch leading to the clade (a,d,b) contradicts $ac|b \in \mathcal{R}_{dc}$, the branch above this clade is collapsed, and the tree output by $PhySIC_{PC}$ is then the tree $(((a,d),b,c),(e,f))$. This tree contains one clade more than the tree output by $Build_{PC}$ *i.e.*, $((a,d,b,c),(e,f))$. The $PhySIC_{PC}$ is detailed in Algorithm 11 of Appendix A.1 and its correctness is proved in Ranwez et al. [2007a].

The time complexity of $PhySIC_{PC}(L(\mathcal{F}),\mathcal{R}(\mathcal{F}))$ `-- Alg. 11`

The most time consuming operations in $PhySIC_{PC}$ are the computation of $\mathcal{R}'|v(C_i)$ and G'_i (line 20), and that of the connected components of this graph (line 23). Obtaining $\mathcal{R}'|v(C_i)$ and constructing G'_i requires considering each triplet of $\mathcal{R}(\mathcal{F})$ at most once and thus has a time complexity of $O(n^3)$, where $n = |L(\mathcal{F})|$. Determining the $CC(G'_i)$s costs $O(n^2)$ (which is the maximum number of edges for a graph with n vertices). During the whole set of recursive calls to $PhySIC_{PC}$, $Check_{PC}$ is modified at most $O(n)$ times (proportional to the number of clades of a tree with n leaves). Lines 20 and 23 are executed as many times as $Check_{PC}$ is modified, *i.e.*, $O(n)$ times. Thus, for the whole set of recursive calls to $PhySIC_{PC}$, the computation

time required by these critical lines is $O(n^4)$, which is also the complexity of the entire procedure.

4.2.2 The $PhySIC_{PI}$ algorithm

The supertree T_{PC} output by $PhySIC_{PC}$ does not usually satisfy the PI property. The $PhySIC_{PI}$ algorithm transforms T_{PC} so that it also satisfies PI. To that aim $PhySIC_{PI}$ recursively searches the tree and checks that for each branch each triplet is induced by $\mathcal{R}(T_{PC}, \mathcal{F})$. The theorem 3.1.1 of Daniel [2004] provides a useful characterisation to decide when a branch is justified, directly or indirectly, thanks to triplets present in $\mathcal{R}(T_{PC}, \mathcal{F})$. When considering the branch linking u to a subtree S_i, the theorem considers a graph G_{ij} for any sibling subtree S_j of S_i. Any such graph G_{ij} is the Aho graph with vertices $L(S_i)$, and with edges due to triplets of $\mathcal{R}(T_{PC}, \mathcal{F})$ whose three leaves are in $L(S_i) \cup L(S_j)$. The theorem states that the branch from u to the root of S_i is justified if and only if G_{ij} is connected, for any sibling subtree S_j.

Consider for instance the simple example where $\mathcal{F}$ contains the trees ((a,b),x) and ((e,f),x). The Aho graph for $\mathcal{R}(\mathcal{F}) = \{ab|x, ef|x\}$ is made of three connected components: $C_1 = \{a, b\}$, $C_2 = \{e, f\}$ and $C_3 = \{x\}$, therefore applying the $PhySIC_{PC}$ algorithm gives the tree $T_{PC} = ((a, b), (e, f), x)$. T_{PC} displays $ab|e$ even though this information is not induced by $\mathcal{F}$. Indeed, the branch defining the clade (a, b) is detected as not justified since the corresponding connected component, C_1, is not connected in the Aho graph when we consider only edges due to triplets with taxa in $C_1 \cup C_2$.

Daniel's theorem is the basis of a *decision* algorithm called *Identifies*, that states whether a given set of triplets identifies a given tree [Daniel, 2004]. It is possible to design a simple variant of this algorithm that always returns a tree (not just a *yes* or *no* answer): when a branch between a node p and the root of a subtree S_i is not justified, the idea is to replace S_i by a star tree on the taxa of the corresponding clade. This crude variant removes the unjustified branches, but also potentially many other branches, *i.e.*, those inside S_i, those leading to sibling subtrees S_j of S_i, and those inside S_j subtrees. $PhySIC_{PI}$ is a more refined variant that only collapses the unjustified branches. The $PhySIC_{PC}$ is detailed in Algorithm 14 of Appendix A.1 and its correctness is proved in Ranwez et al. [2007a]. In this algorithm, $PhySIC_{PI}$ is given a tree T in which unjustified branches are to be collapsed, and a collection $\mathcal{F}$ of source trees or, equivalently, the corresponding set of triplets. $PhySIC_{PI}$ repeatedly calls the $Check_{PI}$ subroutine to detect unjustified branches that are then removed until none remain.

For instance, from the collection of rooted trees $\mathcal{F}$ comprised of trees $(((a, c), b), (e, f))$ and $(((a, d), b), c)$, $PhySIC_{PC}$ infers the supertree $T_{PC} = (((a, d), b, c), (e, f))$ and none of the three internal branches of T_{PC} are collapsed by $Check_{PI}$. For instance, consider the step where $Check_{PI}$ checks the subtree ((a,d),b,c) of T_{PC}, whose child subtrees are (a, d) plus the two trivial subtrees on b and c. The sole branch that has to be checked in $((a, d), b, c)$ is the one defining

the clade (a, d). Here, $Check_{PI}$ builds two Aho graphs with vertices $\{a, d\}$: one with edges due to triplets on $\{a, d\} \cup \{b\}$ and one with edges due to triplets on $\{a, d\} \cup \{c\}$. Both graphs are connected thanks to triplets of the source tree T_2, therefore, $Check_{PI}$ does not collapse this branch.

The time complexity of $PhySIC_{PI}(T, \mathcal{F})$ -- `Alg.` 14

As for $PhySIC_{PC}$, the most time consuming operations done by $PhySIC_{PI}$ are the construction of the Aho graph G_{ij} and the computation of its connected components in the $Check_{PI}$ subroutine. The G_i graphs that may be used in $Check_{PI}$ can be precomputed in the $PhySIC_{PI}$ part of the pseudo-code (*i.e.*, before calling $Check_{PI}$), knowing $\mathcal{R}(\mathcal{F})$ and the current tree T_{PI} to be examined in $Check_{PI}$. This preprocess clearly requires $O(n^4)$ time, since there are $O(n)$ such graphs (one for each clade of T), each of which is obtained by examining the $O(n^3)$ triplets of $\mathcal{R}(T_{PI}, \mathcal{F})$. Recall that $n = |L(\mathcal{F})|$. Each G_{ij} graph can be obtained from a copy of the corresponding G_i graph, completed by the edges due to triplets $ab|c$ having $a, b \in C_i$ and $c \in C_j$. All the G_{ij} graphs required during the recursive calls to $Check_{PI}$ resulting from an execution of line 6 in $PhySIC_{PI}$ can also be precomputed in the $PhySIC_{PI}$ pseudo-code part. This can be done just before line 6, provided that $Check_{PI}$ is modified to end as soon as an edge is collapsed (line 14) – it is clear that this slight modification does not modify the correctness of the algorithm. Indeed, the only G_{ij}s that are then required by $Check_{PI}$ are those corresponding to two sibling clades C_i and C_j of the current T_{PI} tree. Computing all of these G_{ij}s before line 6 of $PhySIC_{PI}$ is done in $O(n^3)$ since each triplet $ab|c$ of $\mathcal{R}(T_{PI})$ adds an edge between A and B in the one and only graph G_{ij}, such that C_i and C_j are sibling clades in T_{PI} and $A, B \in C_i$ and $C \in C_j$.

Note also that the only information used by $Check_{PI}$ on graph G_i and G_{ij} is the number of their connected components. The total number of edges present in the G_{ij} graphs is in $O(n^3)$: precomputation of the number of connected components for this set of graphs is thus globally $O(n^3)$ time. As this has to be done at each pass of the `Repeat` loop, and as this loop is done at most $O(n)$ times (each pass results in the collapsing of one of the $O(n)$ clades of T), this part of the computation is globally (on the whole for $PhySIC_{PI}$) in $O(n^4)$ time. Determination of the number of connected components of each G_i is done only once just before the `Repeat` loop. For each of these $O(n)$ graphs, this requires examining $O(n^3)$ triplets. Thus, this preprocess also costs $O(n^4)$ time. The preprocesses done for G_i and G_{ij} graphs thus requires $O(n^4)$ time and reduces the running time of $Check_{PI}$. The modification of $Check_{PI}$, consisting of returning to $PhySIC_{PI}$ as soon as an edge is collapsed, also simplifies the algorithm (e.g. the `Repeat` loop is no longer required).

4.2.3 The *PhySIC* algorithm

The *PhySIC* algorithm consists in building a supertree for a collection of k source trees $\mathcal{F}$ by first computing the set $\mathcal{R}(\mathcal{F})$ and then successively calling $PhySIC_{PC}$

and $PhySIC_{PI}$. Since both $PhySIC_{PC}$ and $PhySIC_{PI}$ run in $O(n^4)$ time and $\mathcal{R}(\mathcal{F})$ is computed in $O(kn^3)$, *PhySIC* runs in $O(kn^3 + n^4)$ time.

To illustrate the impact of the PC and PI properties on supertree inference we present a case study centered on primate. The *PhySIC* supertree (see Figure 4.5) was inferred combining 24 input trees issued from 24 data sets (*i.e.*, two mitochondrial DNA (mtDNA), 19 nuclear DNA (nuDNA), and three transposable elements data sets), covering 95% of all primate extant genera. The PhySIC supertree conforms to current ideas on Primate phylogeny, and is close to the informal supertree of Primates at the genus level proposed by Goodman et al. [2005]. Moreover, the supertree polytomies were automatically labeled from the *PhySIC* implementation with a label '**c**' if the polytomy resulted from contradictions among the source trees on phylogenetic relationships of corresponding taxa and/or a label '**i**' if any dichotomous resolution of the clade would be at least partially arbitrary and thus would not respect the PI property. In the same paper, we propose polynomial time procedures to modify supertrees proposed by any existing supertree method in order that they satisfy PC and PI with respect to the tree forest they were built from.

Simulation studies (see Figure 4.11) showed that, in some cases *i.e.*, when the source trees do not sufficiently overlap and/or present a high degree of contradictions (as is the case for gene trees affected by horizontal gene transfers or tree-bulding artifacts, such as long branch attraction), supertrees built by *PhySIC* can be highly unresolved. Since we think that the PI and PC properties are mandatory in view of reconstructing the Tree of Life, we designed another supertree method satisfying these properties but proposing more informative supertrees.

4.3 Phylogenetic Signal with Induction and non-Contradiction Inserting a Subset of Taxa (PhySIC_IST)

When more informative supertrees are expected, a solution is to propose non-plenary supertrees, *i.e.*, supertrees containing a subset of the taxa of the source trees. Figures 4.6 and 4.7 show two cases where proposing supertrees (ST_2) lacking only one taxon provides more information on the phylogenetic relationships among other species.

Both the Maximum Agreement Supertree (*SMAST*) and the Maximum Compatible Supertree (*SMCT*) methods [Berry and Nicolas, 2004, 2007], presented in Section 3.3.4.1, can produce non-plenary supertrees. The former consists in finding one of the largest taxa subsets S such that each input tree T proposes exactly the same resolution as the supertree for the taxa contained in $L(T) \cap S$. In this approach the presence of a multifurcation in an input tree will inhibit resolution according to the information present in other input trees. On the contrary, the *SMCT* method allows these multifurcations to be resolved in the resulting supertree. Unfortunately, both underlying decision problems are NP-hard and no heuristic algorithm currently exists for general instances of these problems.

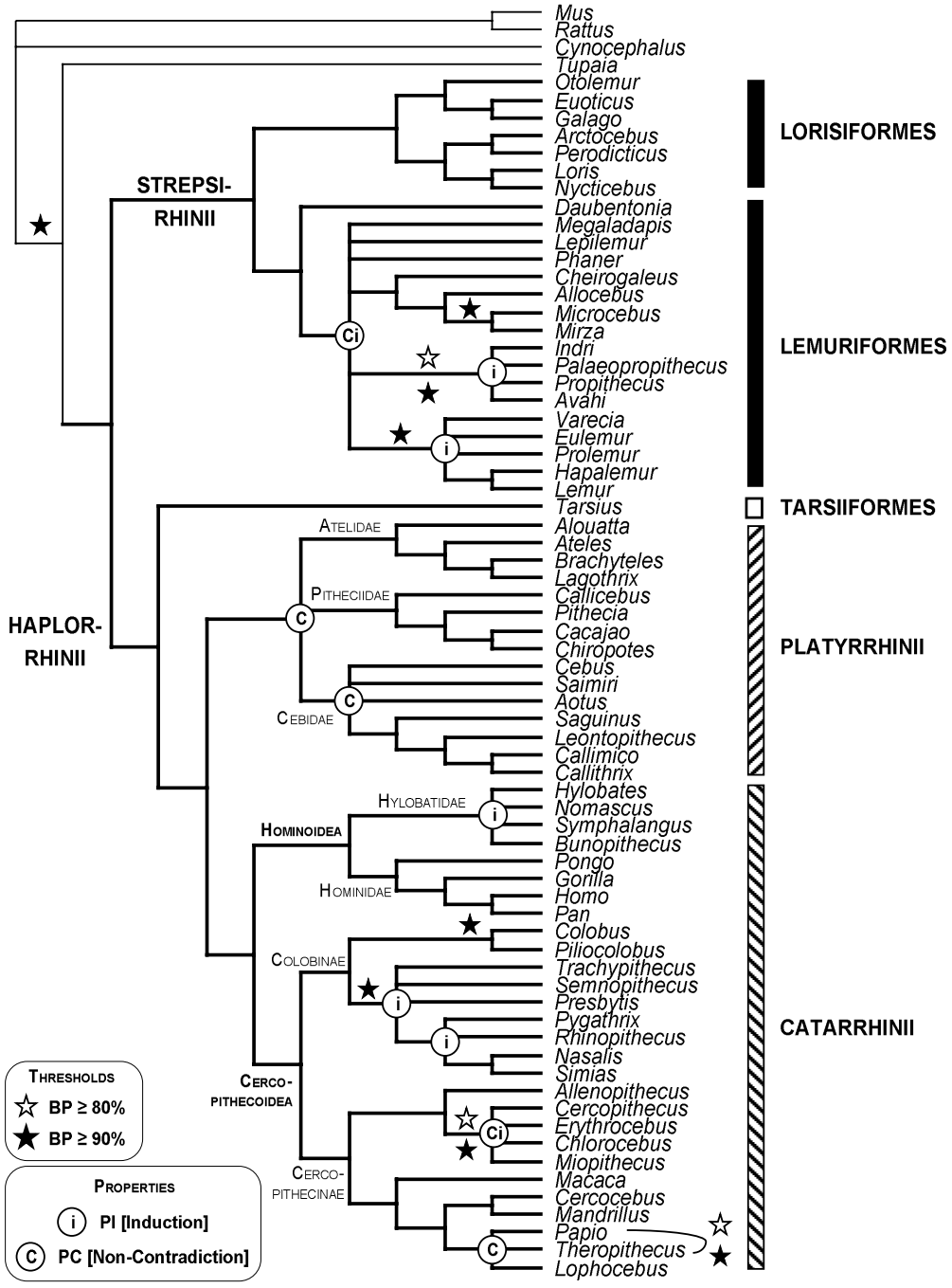


Figure 4.5: **A *PhySIC* supertree covering 95% of all primate extant genera** - the *PhySIC* supertree was inferred from input trees combining 24 input trees issued from 24 data sets, *i.e.*, two mitochondrial DNA (mtDNA), 19 nuclear DNA (nuDNA), and three transposable elements data sets.

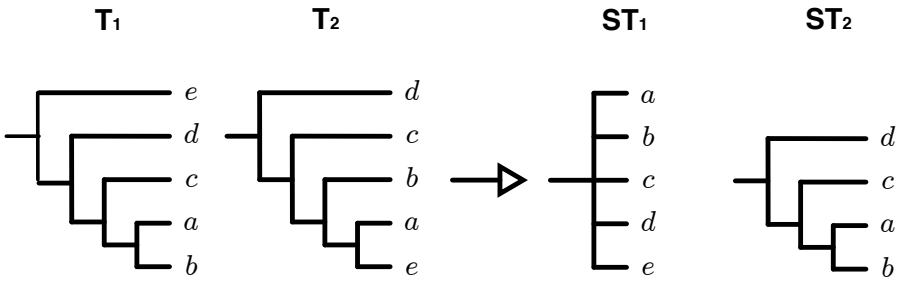


Figure 4.6: **An example of non-plenary supertree for a forest displaying contradictions** - In the case of trees displaying contradictions, such as T_1 and T_2 on the relative position of e, it can be preferable to propose a non-plenary supertree, such as ST_2. In this way, more information on the evolutionary relationships among the remaining species can be obtained. ST_1 is inferred by MRP, ST_2 by *PhySIC_IST*. *PhySIC* produces a star tree on this example.

The algorithm presented in this section, called *PhySIC_IST* (*PHYlogenetic Signal with Induction and non-Contradiction Inserting a Subset of Taxa*), looks for an informative supertree that satisfies PC and PI properties.
PhySIC_IST allows multifurcations in input trees to be resolved thanks to the information present in the other source trees. To deal with topological conflicts *PhySIC_IST* allows, like *SMAST* and *SMCT*, the insertion of only a subset of the species present in the source trees. Moreover, *PhySIC_IST* can also propose new multifurcations to avoid contradicting source trees, while *SMAST* and *SMCT* can only remove taxa.

The aim of *PhySIC_IST* is not only to find *a* supertree T (plenary or not) that satisfies PC and PI but to find the most informative supertree satisfying both properties. Choosing the most informative alternative among several candidate supertrees requires one to be able to compare trees including potentially different subsets of the source taxa (such as ST_1 and ST_2 in Figure 4.7). This is done by using a measure based on a variation of the Cladistic Information Content (*CIC*) criterion [Thorley et al., 1998]. This measure has roots in information theory and is basically proportional to the number of complete binary trees that are compatible with the evaluated supertree.

4.3.1 The *CIC* criterion

Let $\mathcal{F}$ be a collection of source trees on a leaf set of n taxa. The information contained in an incomplete supertree T is a function of both the number $n_R(T, n)$ of its possible biological interpretations (*i.e.*, the number of fully resolved trees on $L(\mathcal{F})$ that encompasses T) and $n_R(n)$, the number of fully resolved trees on n leaves. More precisely, the *CIC* value of T relative to n source taxa is defined as:

$$CIC(T, n) = -\lg \frac{n_R(T, n)}{n_R(n)}$$

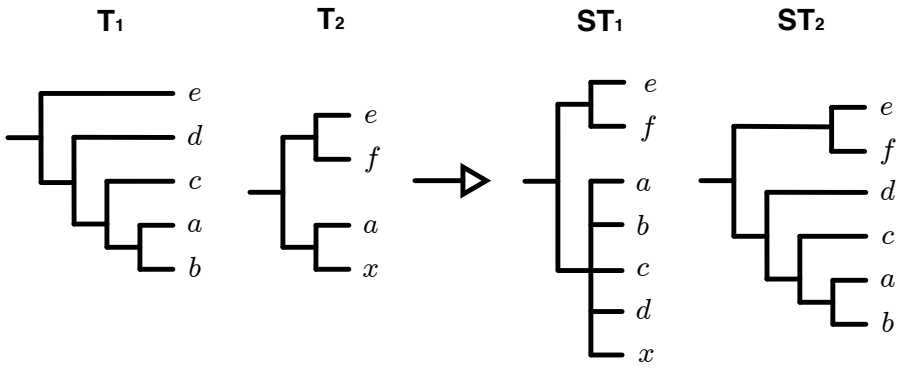


Figure 4.7: **An example of non-plenary supertree for a forest displaying a significant lack of overlap** - In the case of trees displaying a significant lack of overlap, such as T_1 and T_2, it can be preferable to propose a non-plenary supertree, such as ST_2. In this way, more information on the evolutionary relationships among the species included in the supertree can be obtained. ST_1 is inferred by MRP (the same tree is obtained by *PhySIC*), ST_2 by *PhySIC_IST*.

In case of non-plenary supertrees, $n_R(T, n)$ depends on the multifurcations of T (since they reflect an ambiguity) and on the number of source taxa missing in T (since T contains no information for them). More formally, given a collection $\mathcal{F}$ of input trees and a candidate supertree T, the number of permitted binary trees for T referring to $\mathcal{F}$ is the number of binary trees T' such that $L(T') = L(\mathcal{F})$ and $T'|L(T)$ refines T. We observe that, for each internal node u_i with a number c_i of children, we have $(2c_i - 3)!!$ possible resolutions [Semple and Steel, 2003]. Moreover, if $L(T) \subset L(\mathcal{F})$, we have to insert all missing taxa, *i.e.*, those in $L(\mathcal{F}) - L(T)$. A rooted binary tree of i taxa has $2(i-1)$ branches; so, there are $2i-1$ possible positions for the $(i+1)^{th}$ taxon, taking into consideration the possibility of insertions above the root. We detail in Algorithm 17 in Appendix A.2 how the value of $CIC(T, n)$ can be computed. In Figures 4.8 and 4.11 we refer to $CIC_N(T, n)$ as the normalized value of $CIC(T, n)$, *i.e.*,

$$CIC_N(T, n) = CIC(T, n)/(-\lg 1/n_R(n)).$$

Another way to compare the information of different trees is to compare their number of triplets. However, the *CIC* criterion better takes into account missing taxa. For instance, consider the trees T_1 and T_2 in Figure 4.8. The former is completely resolved but lacks taxon h, while the latter contains all taxa but is highly unresolved. Searching for the tree that maximizes the number of triplets, would lead to prefer T_2 (since $|\mathcal{R}(T_1)| = 35$ while $|\mathcal{R}(T_2)| = 48$). However, it seems more reasonable to favor the tree that maximizes the value of the *CIC* criterion (in this case T_1, since $CIC_N(T_1, 8) = 0.78$, while $CIC_N(T_2, 8) = 0.54$).

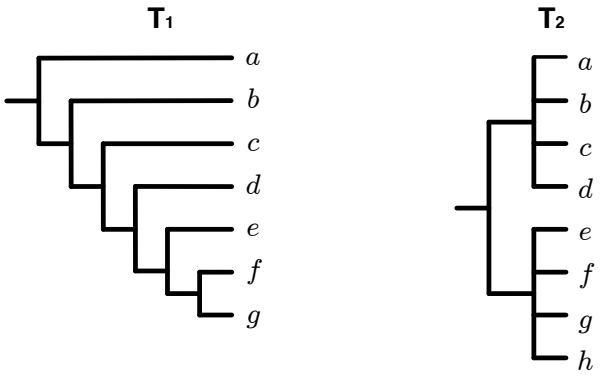


Figure 4.8: **Comparison of two informativeness measures: number of triplets and the *CIC* criterion** - The second tree displays more triplets than the first one ($|\mathcal{R}(T_1)| = 35$ while $|\mathcal{R}(T_2)| = 48$) while the latter has a better *CIC* than the former ($CIC_N(T_1, 8) = 0.78$ while $CIC_N(T_2, 8) = 0.54$).

4.3.2 The *PhySIC_IST* algorithm

The aim of *PhySIC_IST* is to find *a* supertree T (plenary or not) that satisfies PC and PI *and* that have the maximum *CIC*. This gives rise to the following optimization problem:

Problem	Most informative induced and non-contradicting supertree (MIICS)
Input	a collection $\mathcal{F}$ of rooted trees.
Output	a tree T such that: (i) T satisfies PI and PC for $\mathcal{F}$ (ii) $CIC(T, \lvert L(\mathcal{F})\rvert)$ is maximum among the trees satisfying (i).

We conjecture this problem to be hard since it is a variant of the MIST (Maximum Identifying Subset of rooted Triplets) problem and of the ST (Triplet Supertree) problem, both shown to be NP-hard [Bryant, 1997; Guillemot and Berry, 2007; Jansson, 2001; Wu, 2004]. *PhySIC_IST* is a polynomial-time heuristics to solve the MIICS problem. Note that it is heuristics only on point (ii), since it always outputs a supertree satisfying (i).

4.3.2.1 Inferring informative and reliable supertrees: *PhySIC_IST*

In this section we give an outline of the new method *PhySIC_IST*. This algorithm operates successive insertions of taxa on a backbone topology. Given a rooted forest $\mathcal{F}$, the rough outline the *PhySIC_IST* method is the following:

(1) order taxa of $L(\mathcal{F})$ in a priority order;
(2) construct a starting backbone tree T formed of a root connecting two leaf nodes labeled by the first two taxa in the priority list;
(3) for each taxon l in priority order:
  (3a) choose a node or a branch of the backbone tree T where insert l; call T' the tree obtained by inserting l in the chosen placement in T;
  (3b) collapse some branches of T' until it satisfies PI and PC for $\mathcal{F}$;
  (3c) if(CIC($T', L(\mathcal{F})$) $>$ CIC($T, L(\mathcal{F})$)) $T \leftarrow T'$;

We will see that point (3) is oversimplified in this outline and that *PhySIC_IST* acts smarter than that. In the rest of the section we describe *PhySIC_IST* in more details.

Priority order Since *PhySIC_IST* is a greedy algorithm, the order of the insertions has to be chosen carefully. Once a taxon is inserted, its presence in the supertree will never be questioned. It is therefore preferable to first insert the taxa with a strong and unambiguous signal. The first taxa inserted are thus for which we have as much triplet information as possible and involved in as few contradictions as possible. In fact, inserting a taxon that is present in numerous triplets of $\mathcal{F}$ provides information, not only on its position, but also on the position of remaining taxa. On the other hand, delaying the insertion of incongruent taxa lessens the chances to misplace them due to incomplete information and to be unable to proceed with the insertion of remaining taxa. More formally, the priority order is determined as a function of $\mathcal{R}$ and $\mathcal{R}_{dc}$, respectively the set of triplets of $\mathcal{F}$ and the subset of $\mathcal{R}$ that contains direct contradictions. Given a taxon l, we denote by $|\mathcal{R}(l)|$ (resp. $|\mathcal{R}_{dc}(l)|$) the number of triplets containing l present in $\mathcal{R}$ (resp. $\mathcal{R}_{dc}$). For each $l \in L(\mathcal{F})$ we compute the value

$$priority(l) = |\mathcal{R}(l)| - |\mathcal{R}_{dc}(l)|$$

and we order taxa in decreasing priority order.

Then, we build the starting backbone tree, formed of a root node to which are connected two leaves corresponding to the first two taxa in the priority list.

How to choose where to insert a taxon in the backbone tree Given a source tree T_i, the backbone tree T, and a taxon $l \in L(T_i)$ not yet inserted in T, we want to determine within which region of T the taxon l can be inserted without contradicting the information contained in T_i. When the insertion of l on an edge (resp. a node) does not induce contradictions between T and T_i, this edge (resp. node) is said to be *supported*. To delimit the supported region, we map the nodes of T_i with the nodes of T. We define $T_i^{'}$ as $T_i|(L(T) \cup \{l\})$. We denote by $f_i^{'}$ the father of l in $T_i^{'}$ and by $C_i^{'}$ the set of children of $f_i^{'}$ other than l. The position of l in T_i can be seen as delimited by $f_i^{'}$ as an upper bound and by each $c_i \in C_i^{'}$ as lower bounds. The corresponding bounds in T are denoted f and C (see Algorithm 16 in Appendix A.2 for more details and Figure 4.9 for an example).

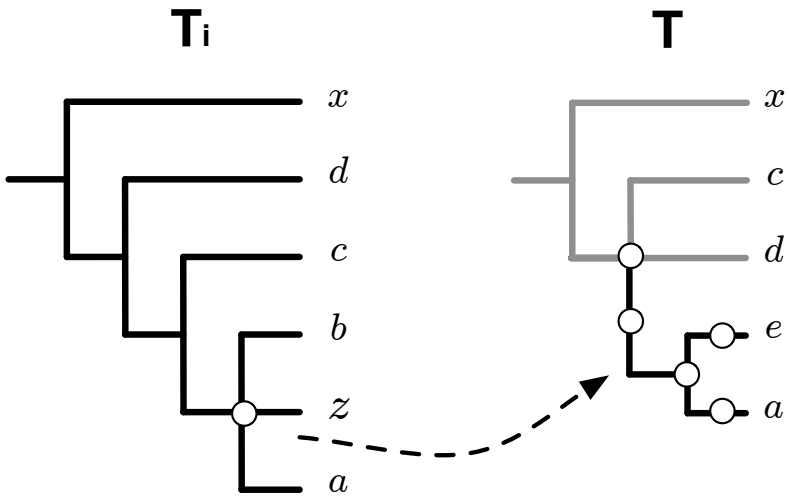


Figure 4.9: **An example showing the supported region of T for the insertion of the taxon z, according to tree T_i** - The taxon z can be inserted in T on all black edges and on all nodes highlighted by a white circle, without contradicting T_i.

Doing this for each T_i containing l give us some information on the best region(s) of T within which the taxon l can be inserted.

The different kinds of insertions Note that, if all source trees support the insertion of a taxon in a region (a node or a branch), inserting it in this region will not create contradictions between the source trees and the supertree. Thus this insertion will not violate PC. Additionally, if the region supported by source trees is not limited to a node or an edge, it means that the information we have is not enough to choose where the taxon has to be inserted. Such an insertion will surely violate PI. These considerations make insertions supported by all trees more appealing than insertions supported by only a part of them, and the insertions on a region well delimited more attractive than insertions on a larger region. This is the reason why in *PhySIC_IST* the insertions of taxa are done in four successive steps, each step being less restrictive than the previous ones in its requirements for inserting taxa. The strictest steps are done first, in order to maximize the chances for future taxa to be inserted and to maximize the *CIC* of the computed supertree. These four steps are differentiated according to two binary parameters, *all* and *cons*. The *all* parameter indicates whether taxa should be inserted only when a *maximum* support is observed for them somewhere in the backbone tree ($all = true$), or whether, in the absence of places with maximum support, places of *maximal* support should be considered ($all = false$). By maximum support at a position we mean that all source trees containing the taxa agree that it could be inserted at the given position. Note though that there might be several places of maximum support for inserting a taxon, due to a lack of overlap between the

source trees and the taxa already in the backbone tree.
The case where $all = false$ leads the backbone tree to temporarily contradict at least one source tree. This means that some of its edges have to be collapsed to ensure that the backbone tree still satisfies PC after the insertions. The collapsing of a minimal number of edges is performed by calling the $Check_{PC}$ procedure (see Algorithm 15 in Appendix A.2); an analogous test to check PI is performed calling the $Check_{PI}$ procedure [Ranwez et al., 2007a]. If this collapsing decreases the value of CIC of the tree compared to its value prior to the insertion, then the insertion is cancelled. Overall, the insertions with $all = true$ promise a more resolved supertree and are hence performed during the first two insertion stages, while the latter two are performed with $all = false$.
The parameter *cons* indicates whether the insertion procedure should insert taxa only when there is a single best supported position for them ($cons = false$) or when *consensus* insertions are allowed ($cons = true$). A consensus insertion means inserting taxa on a node when all best supported places for the taxa are edges incident to the node. In this case, the insertion of the taxon *does not contradict* the source trees. Insertions with $cons = true$ are always on a node, therefore insertions with $cons = false$ are preferable because the possibility to insert taxa on a edge provides a tree with a higher CIC than an insertion on a node. Thus, for each value of all, a step with $cons = false$ is first performed followed by a step with $cons = true$.
During each insertion stage (see Algorithm 20 in Appendix A.2), all taxa not yet inserted in the backbone tree are considered. If the current taxon is inserted (by the `roundIns` procedure detailed in Algorithm 19 in Appendix A.2), then the algorithm retries to insert, always in priority order, all taxa previously considered that could not have been inserted before. These taxa have higher priority than taxa following the current one, and it is possible that the insertion of the current taxon enables the supported position for some of these taxa to be circumvented to a small enough part of the tree for their insertion to be possible. After each insertion the problematic branches are collapsed, to ensure that the backbone tree still satisfies PC. After inserting several taxa, the backbone tree may fail to satisfy PI. However, using the $Check_{PI}$ procedure to collapse problematic edges suffices to ensure that the backbone tree satisfies the property again. Collapsing branches with $Check_{PI}$ is done after each insertion stage and not after every insertion, contrarily to $Check_{PC}$. The reason is that some edges of the backbone tree can fail to satisfy PI only temporarily and satisfy it again after the insertion of other taxa. On the contrary, if the backbone contradicts any source tree, it will keep contradicting it, no matter which taxon we insert afterward; it is thus preferable to detect this immediately to avoid problems that may arise while inserting remaining taxa.

4.3.2.2 Complexity of *PhySIC_IST*

The outlines of *PhySIC_IST* methods are given in Appendix A.2. In this section we compute the running time of *PhySIC_IST*. Denoting by k the number of source trees and by n the number of taxa within the tree collection, the time complexity of *PhySIC_IST* is shown to be $O(n^3(k+n^3))$, *i.e.*, the method runs in polynomial time. To prove this statement, the complexity of each *PhySIC_IST* subroutine is detailed.

The time complexity of `support`(T_i, T, l) is $O(n)$ — Alg. 16

T and T_i have size $O(n)$, hence $L(T)$ and $L(T_i)$ can be obtained in $O(n)$. The lca of all pairs of nodes in T can be computed in $O(n)$ [see Bender and Farach-Colton, 2000; Harel and Tarjan, 1984], then each lca query costs $O(1)$. Other steps involved in this subroutine correspond to a constant number of traversals of parts of the trees T and T_i, each time involving $O(1)$ operations per node and branch. As a result, the complexity of the procedure is $O(n)$.

The time complexity of $Check_{PC}$ $(T, \mathcal{R}, \mathcal{R}_{dc})$ is $O(n^4)$ — Alg. 15

This procedure collapses some branches of the tree T, until T satisfies PC for $\mathcal{F}$ [Ranwez et al., 2007a, Lemma 1]. The set $\mathcal{R}_T$ contains $O(n^3)$ triplets. Checking whether r_T (resp. $\bar{r_T}$) is in $\mathcal{R}_{dc}$ (resp. in $\mathcal{R}$), or not and obtaining the nodes u, v in T corresponding to r_T can be done in constant time, through lca queries (once the tree has been preprocessed in $O(n)$ [Bender and Farach-Colton, 2000; Harel and Tarjan, 1984]). Marking branches of the path $[u, v]$ is done in $O(n)$ time, proportionally to the number of branches in that path. This happens at worst for each triplet, hence costs $O(n^4)$ globally. This is then the complexity of the procedure (removing all marked branches of T only requires a single search of T, *i.e.*, $O(n)$).

The time complexity of `CIC`(T, n) is $O(n)$ — Alg. 17

In the first loop, a number of multiplications equal to the number of branches in the tree is performed (thus requiring $O(n)$ time). The second loop performs a multiplication per missing taxa (requiring $O(n)$ time again). Then computing $n_R(n)$ by the traditional formula in phylogenetics to count the number of rooted trees having n leaves is done in $O(n)$ multiplications.

The time complexity of `betterCIC`$(T, n, \mathcal{R}, \mathcal{R}_{dc}, u, l, above)$ is $O(n^4)$ — Alg. 18

Building T' requires copying T and inserting a taxon l above/on the node u, thus costing $O(n)$ time. The complexity of this subroutine is therefore that of the $Check_{PI}$ and $Check_{PC}$ procedures, *i.e.*, $O(n^4)$, see [Ranwez et al., 2007a, Thm 2] and above, respectively.

The time complexity of `roundIns`$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, l, all, summary)$ is $O(nk + n^4)$ — Alg. 19

The function `support` is called for each tree containing the taxon l. In the worst case, *i.e.*, l is present in all source trees, this step requires $O(nk)$ time. Among the other step of `roundIns`, the most time consuming operations are `betterCIC` and $Check_{PC}$, which both cost $O(n^4)$ time. So, the total cost of `roundIns` is $O(nk+n^4)$.

The time complexity of `insertion`$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, priorityList, all, summary)$ is $O(n^3(k + n^3))$ — Alg. 20

Each of the n taxa is considered at most $O(n)$ times: a first time and if not inserted, each time another taxa is inserted (as long it is not itself inserted). Overall $O(n^2)$ calls to $roundIns$ can be issued, each costing $O(nk+n^4)$ time. Thus, the `insertion` procedure runs in $O(n^3(k + n^3))$ time.

The time complexity of `PhySIC_IST`$(\mathcal{F})$ is $O(n^3(k + n^3))$— Alg. 21

In the procedure `PhySIC_IST` the first step consists in computing $\mathcal{R}$ and $\mathcal{R}_{dc}$. This step requires $O(kn^3)$ because each of the k trees in the collection can host $O(n^3)$ triplets. Then, for each taxon l of the collection, $|\mathcal{R}(l)|$ and $|\mathcal{R}_{dc}(l)|$ are computed (see Section 4.3.2.1) and the taxa are order in decreasing priority order. The total complexity of this task is $O(n^3)$ time. The total cost of `PhySIC_IST` is dominated by the complexity of the `insertion` procedure (that is called a constant number of times) and is therefore $O(n^3(k + n^3))$.

4.3.2.3 The STC preprocessing

The resolution of supertrees computed by veto methods can be poor when considering large numbers of source trees. Indeed, adding more trees provides more information on the relative position of some taxa, but in the same time increases the number of local conflicts. To handle large collections of source trees, one has to resort to the liberal approach that allows to arbitrate between conflicts arising among source trees. The most common way to deal with incongruent source trees is to use a supertree method that takes ad-hoc decisions (according to a chosen objective criterion) in the face of individual conflicts met when building the supertree. The second and much less explored way is to preprocess the data according to a statistical procedure and then to apply a veto method, not contradicting the retained information that was estimated to be reliable. In this section, we follow the latter approach that has the advantage of making the removing of conflicts between source trees explicit. More precisely, we introduce a preprocessing step to detect and correct anomalies in the source trees. This step, called STC (Source Trees Correction), analyzes the contradictions among the source trees; for all contradictions,

it evaluates the possible topological alternatives and it drops the alternative(s) that is (are) statistically less supported (with a threshold chosen by the user). Then STC modifies each source tree (using a schema similar to that of *PhySIC_IST*) so that it does not contain the dropped alternatives and yet remains as informative as possible. In other words the STC aims at correcting the source trees that propose anomalous phylogenetic positions for some taxa (due to horizontal gene transfers, long branch attractions, paralogy ...). For example, if source trees contain two contradicting resolutions, one present in 99% of the trees and the other one present in 1% of the trees, we can reasonably think that the latter resolution is an anomaly and ignore it. If the user approves the proposed modifications, the *PhySIC_IST* veto method is then applied to the modified source trees. The resulting supertree satisfies both PI and PC properties for the collection of modified source trees. If the user is not satisfied with the modified source trees, he can change the threshold and restart the procedure, or choose to skip it. In this way, the liberal component of the supertree inference is not only made explicit but also interactive and parametrized.

The aim of the STC (Source Tree Correction) preprocessing is to analyze the direct contradictions in the source trees, to drop the statistically less supported alternatives and to correct the source trees accordingly.
For a triplet t, we denote by $\dot{t}$ and $\ddot{t}$ the two other possible triplets for the same set of three taxa and by $|t|$, $|\dot{t}|$ and $|\ddot{t}|$ the number of occurrences of t, $\dot{t}$ and $\ddot{t}$ in the source trees. Only resolved triplets (like $ab|c$) are taken into account in the computation of $|t|$, $|\dot{t}|$ and $|\ddot{t}|$, while tricotomies are ignored. Given a set of source trees $\mathcal{F}$, for each $t \in \mathcal{R}(\mathcal{F})$, the vector composed by the three number of occurrences $|t|$, $|\dot{t}|$ and $|\ddot{t}|$ is denoted by $occ(t)$. We indicate with $max(t)$ the maximum value in $occ(t)$. Each time that $occ(t)$ has at least two non-null coordinates, we have a direct contradiction. In this case, we want to drop the statistically less supported alternative(s), if any exists. To do that, the STC preprocessing compares each non-zero value i in $occ(t)$ with $max(t)$ and it uses a Chi-Square test [Fienberg, 1977] with one degree of freedom to check whether the difference between the two values is significant. The null hypothesis $\mathbf{H}_0$ is that $p_i = p_{max(t)} = \frac{1}{2}$, *i.e.*, there is no difference between the observed frequencies of the two triplets (one presents i times and the other $max(t)$ times). For each i, the STC preprocessing uses the basic Chi-square test to assess the plausibility of this hypothesis, computing

$$\chi^2 = \frac{(i - q \cdot p_i)^2}{q \cdot p_i} + \frac{(max(t) - q \cdot p_{max(t)})^2}{q \cdot p_{max(t)}}$$

$$= \frac{(i - \frac{q}{2})^2 + (max(t) - \frac{q}{2})^2}{\frac{q}{2}}$$

where $q = i + max(t)$. This value is compared to the quantile x_0 corresponding to the threshold τ given by the user, *i.e.*, $x_0 : Prob\{X < x_0\} = (1 - \tau)$, where X is the Chi-Square distributed with one degree of freedom. If $\chi^2 > x_0$, the STC preprocessing rejects the $\mathbf{H}_0$ and inserts the triplet associated to i in $\mathcal{W}(\mathcal{F})$, *i.e.*, the set of dropped triplets. Note that the two tests performed on each non-null

coordinate are not independent. The user may use the threshold more as a setting parameter rather than interpret it as the probability that the STC drops a triplet that underlies a real anomaly. After that, the STC preprocessing modifies the source trees applying *PhySIC_IST* to each $T_j \in \mathcal{F}$, with $\mathcal{R} = \mathcal{R}(T_j)$ and $\mathcal{R}_{dc} = \mathcal{W}(\mathcal{F})$. In this way, we force the source trees not to contain the dropped triplets. Essentially, each modified tree may contain either new multifurcations, or lack some of its former taxa (if the phylogenetic position of these taxa changes extremely within the forest). Then *PhySIC_IST* is applied to the modified source trees. If the user does not agree with the source tree modifications, he can change τ and restart the STC procedure or choose to skip it.

4.3.3 Rooting the source trees

When *PhySIC_IST* is provided with unrooted source trees, it first has to root them. There are several approaches to root phylogenetic trees, among which are the outgroup, the molecular clock, and the non-reversible model of character-state changes. It has been shown that the outgroup criterion is consistently able to identify the root [Huelsenbeck et al., 2002a]. In our implementation of *PhySIC_IST*, we provide a rooting tool that automates the procedure. This tool accepts as input different levels θ_i of outgroup, each one being a list of taxa. The rooting procedure considers each unrooted source tree separately. For a given source tree T, it determines the first θ_i such that $\theta_i \cap L(T) \neq \emptyset$. Then the tree is rooted on the branch leading to the smallest subtree hosting all outgroup taxa of θ_i. If the proposed outgroup is not monophyletic, the tree T is discarded from the analysis. This procedure does not alter the resolution inside the ingroup nor in the different outgroup levels that can be present in the tree.
Rooting trees is not trivial, hence outgroup levels have to be chosen carefully.

4.3.4 The *PhySIC_IST* validation

4.3.4.1 Simulation studies

PhySIC_IST and the STC preprocessing were implemented using the *BIO++* libraries [Dutheil et al., 2006], and are available from: `http://www.atgc-montpellier.fr/physic_ist/`.

In this section we present results of large-scale simulations conducted to evaluate both the resolution and the accuracy of *PhySIC_IST* supertrees. These results help to measure both the improvement offered by *PhySIC_IST* on the previous version of the method, and the effectiveness of the *STC* preprocess. We also validate the new methodology by applying STC+*PhySIC_IST* to two biological case studies.

4.3.4.2 Simulations

The simulation protocol, depicted in Figure 4.10, follows the standard guidelines in the field for assessing the effectiveness of supertree methods. Its details are inspired from Criscuolo et al. [2006]. We created 100 different clocklike trees; for each tree, every branch length was multiplied by a random value, chosen in an exponential distribution. Then each branch length was divided by the total branch length (TBL) of the resulting tree, providing 100 normalized (TBL=1) non-clocklike model trees. From each model tree, we derived 50 gene trees with different evolutionary rates, by multiplying every branch by a given value (the same within each gene tree, but different from gene to gene). Then the evolution of DNA sequences along these gene trees was simulated according to the K2P substitution model [Kimura, 1980], obtaining a sequence alignment data set per gene tree. The different taxa overlaps observed in real data sets were simulated by randomly removing some sequences of those 50 data sets.

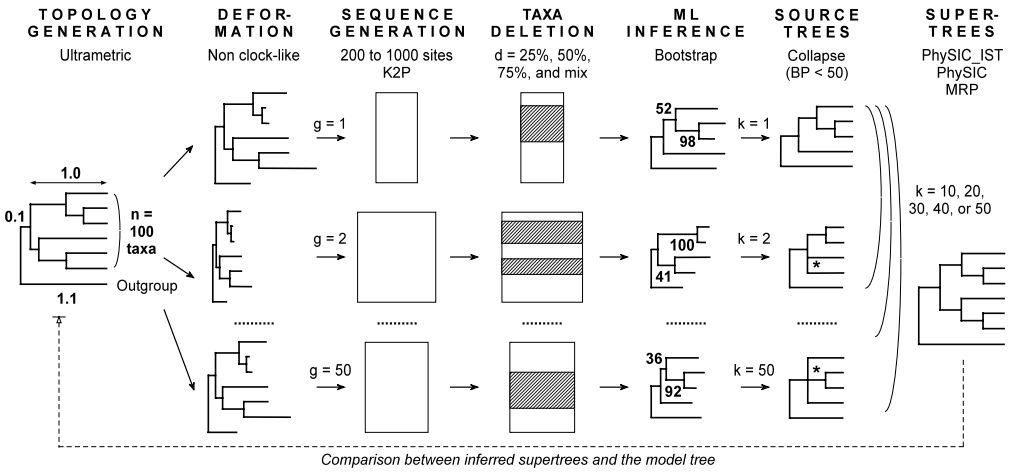


Figure 4.10: **Simulation protocol** -

As in Criscuolo et al. [2006]; Eulenstein et al. [2004], the deletion of sequences was performed according to four different proportions: $d = 25\%$, to model a strong overlap between source trees, $d = 50\%$ and $d = 75\%$, to represent sets with low taxon overlap. Moreover, we added a mixed deletion ratio ($d = mix$), to model a more realistic heterogeneity among source trees sizes. The mixed deletion condition is composed of one tenth of data sets with $d = 25\%$, three tenths with $d = 50\%$ and six tenths with $d = 75\%$. From the resulting data sets, we inferred 50 gene trees for each value of d, using PhyML [Guindon and Gascuel, 2003]. The node supports were estimated using PhyML with a bootstrap process based on 100 replicates. For each inferred tree, we only retained the best supported nodes *i.e.*, those showing a bootstrap proportion at least equal to 50. We built supertrees from all gene trees ($k = 50$) or only a subset of them ($k = 10, 20, 30, 40$). One hundred data sets were obtained for each of the 20 combinations of k and d.

We detail results for three supertree methods applied to the collections of source trees, namely *PhySIC* [Ranwez et al., 2007a], *PhySIC_IST*, and MRP [Baum and Ragan, 2004]. Veto and liberal methods are not really comparable because they are used for different purposes. Veto methods are expected to produce less resolved but more accurate supertrees: showing results for both kinds of methods gives an indication of how much is lost in resolution and of how much is gained in accuracy when using a veto method. For each supertree we evaluate its informativeness by computing its CIC_N (see Section 4.3.1 for more details). Additionally, we compute its type I error, *i.e.*, the number of triplets of the supertree not present in the model tree divided by the number of triplets in the model tree. For each condition, we average these values on the 100 replicates. Figures 4.11 and 4.12 summarize the results of the simulations. The informativeness of supertrees is frequently compared using type II error, *i.e.*, the number of triplets of the model tree that are not present in the supertree divided by the number of triplets in the model tree. It seems to us that the CIC_N is more appropriate when comparing the informativeness of supertrees. Indeed, if a triplet $t \in \mathcal{R}$ is included in the computation of the type II error, this may be a result of it not having been expressed in the supertree or of an alternative resolution having been proposed. To the contrary, the CIC_N strictly measures the information contained in the supertree, whether it is accurate or not. For consistency with the optimization criterion of PhySIC_IST, the average values of CIC_N are provided and commented. Type II error graphics are provided in Figure 4.13 but not commented since they show the same trends of the CIC_N. The accuracy of the supertree is separately measured using the type I error, *i.e.*, the number of triplets of the supertree that are not present in the model tree divided by the number of triplets in the model tree. Graphics showing the sum of Type I and Type II errors are also provided in Figure 4.14.

4.3.4.3 Improvement of *PhySIC_IST* on *PhySIC*

The increase in resolution of *PhySIC_IST* over *PhySIC* is noteworthy no matter the deletion ratio (Figure 4.11). More precisely, the average CIC_N of *PhySIC_IST* supertrees is 1.5 that of *PhySIC* (over all simulation conditions). Since CIC_N is measured on a logarithmic scale, this means a considerable improvement on *PhySIC*. This different behaviour of the two methods is due, most of the time, to the fact that *PhySIC_IST* is allowed to infer non-plenary supertrees.

Indeed, removing just one taxon is sometimes enough to make all source trees agree on a large subset of taxa. As veto methods are not allowed to contradict source trees, keeping the rogue taxa in the supertree means proposing a multifurcation for the surrounding subset of taxa, as done by *PhySIC*. The *PhySIC_IST* version escapes this situation by not including the rogue taxa in the supertree, and is hence able to obtain a relatively important resolution for the remaining taxa.
In the meantime, the type I error of *PhySIC_IST* (Figure 4.12) is always inferior to 1% (except for $d = 75\%$ and $k = 10$) and decreases significantly as the number of source trees increases. From the experimental results, it could appear that there is

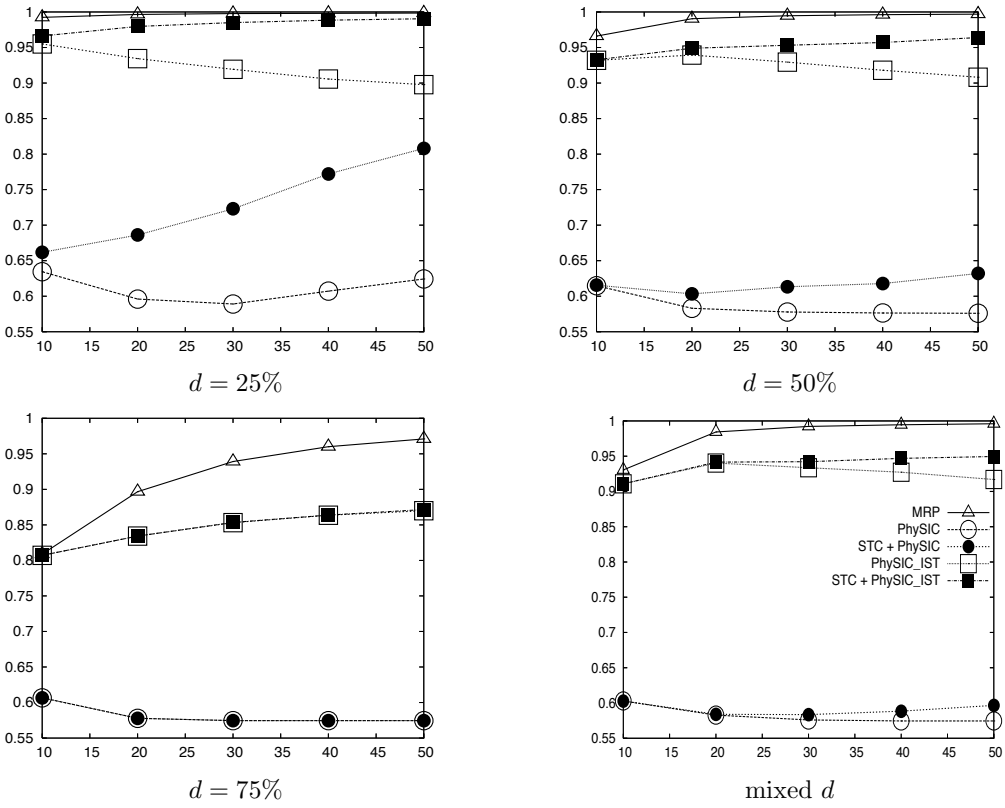


Figure 4.11: **Simulation results: average CIC_N values** - Average CIC_N values (y-axis) of supertrees built with different supertree methods (MRP △, *PhySIC* ○, *PhySIC_IST* □, STC+*PhySIC* ● and STC+*PhySIC_IST* ■), depending on the number of source trees (x-axis). The results are shown for source trees inferred from data sets in which sequences have been deleted with $d = 25\%, 50\%, 75\%$ and mixed proportions

a choice to be made between the two methods since PhySIC displays a significantly lower type I error rate (see Figure 4.12), but this is mainly due to the fact that the trees reconstructed by *PhySIC* can be much less resolved, as expected from a plenary veto method applied to a large number of source trees. Thus, on practical data sets, *PhySIC_IST* is always to be preferred to *PhySIC*.

The table in Figure 4.15(a) shows the average percentage of source taxa not included in the supertrees inferred by *PhySIC_IST*, for each simulation condition. This percentage depends on the number and size of the source trees but remains globally low (*i.e.*, less than 10%, except for $d = 75\%$ where it reaches $\approx 25\%$).

When source trees contain insufficient information (*e.g.* $d = 75\%$ and $k = 10$), *PhySIC_IST* can infer supertrees lacking several taxa. Indeed, in such a case, the insertion of some taxa is impeded by the PI property: very little overlapping information is available and consequently many taxa cannot be placed unambiguously.

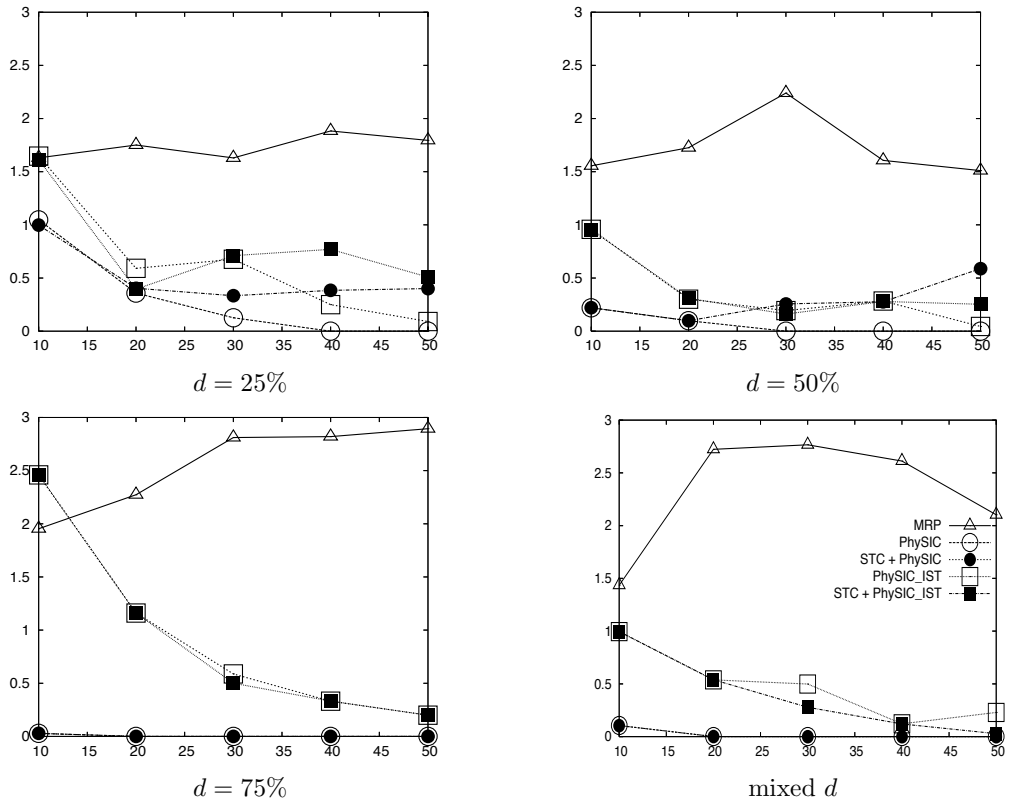


Figure 4.12: **Simulation results: average percentage of type I error** - Average percentage of type I error (y-axis) of supertrees built with different supertree methods (MRP △, *PhySIC* ○, *PhySIC_IST* □, STC+*PhySIC* ● and STC+*PhySIC_IST* ■), depending on the number of source trees (x-axis). The results are shown for source trees inferred from data sets in which sequences have been deleted with $d = 25\%, 50\%, 75\%$ and mixed proportions.

Providing *PhySIC_IST* with more information (by increasing k or decreasing d) allows to precise the position of some taxa, hence to propose larger supertrees. Yet, as the amount of available information continues to increase, the number of conflicts between source trees augments, leading some taxa no longer to be inserted due to the PC property. This means that increasing the amount of available information after some point can decrease the informativeness and the size of the inferred supertree (this phenomenon can be observed in Figures 4.11 and 4.15 for $d = 50\%$ when increasing k).

The foreseeable but undesirable behavior of veto supertree methods when facing large numbers of source trees can be overcome by an explicit liberal preprocessing of the input trees, such as the STC proposed in Section 4.3.2.3.
It is also interesting to analyze the CIC_N values plotted as a function of the number of removed taxa. For each of the 20 conditions here analyzed, the 100 inferred

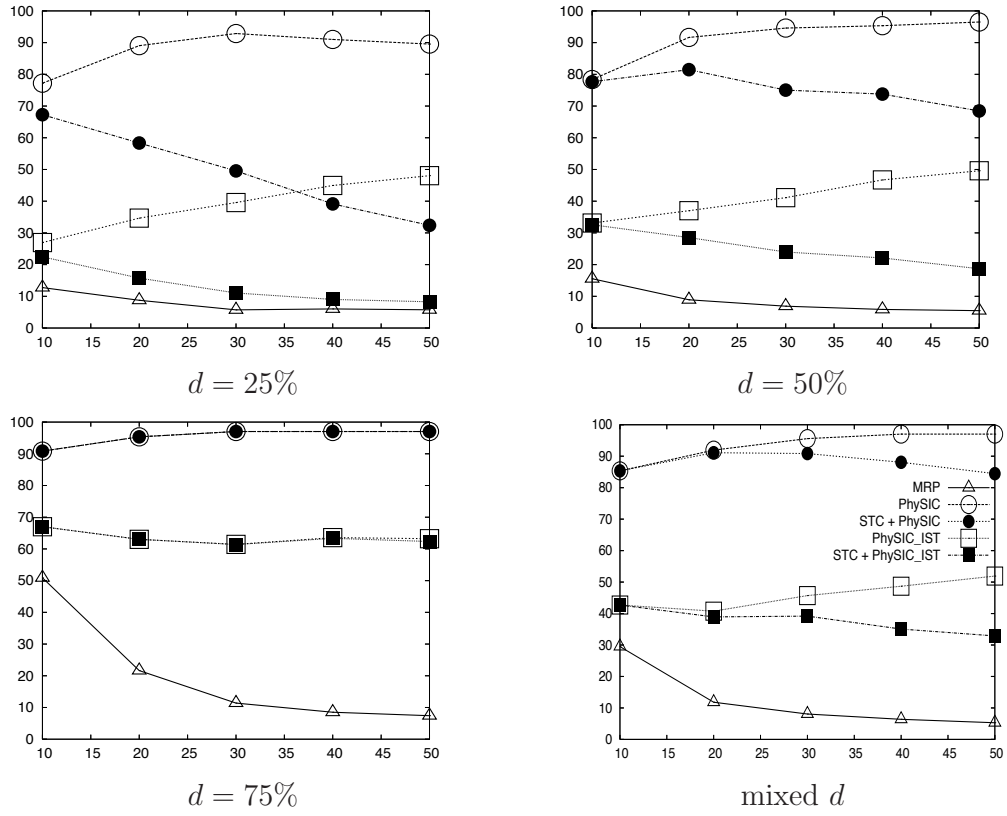


Figure 4.13: **Simulation results: average percentage of type II error** - Average percentage of type II error (y-axis) of supertrees built with different supertree methods (MRP △, *PhySIC* ○, *PhySIC_IST* □, STC+*PhySIC* ● and STC+*PhySIC_IST* ■), depending on the number of source trees (x-axis). The results are shown for source trees inferred from data sets in which sequences have been deleted with $d = 25\%, 50\%, 75\%$ and mixed proportions

supertrees are split into classes, depending on the number of taxa not inserted in the supertrees but present in the source trees. Then, the average CIC_N value is computed for each class (a class usually contains more than one tree) and these values are plotted as a function of the number of input taxa not inserted in the supertrees (see Figure 4.16).

For comparison, we also plotted the CIC_N values of binary trees having the same number of leaves as the supertrees in each class. These values, denoted *max* CIC_N, provide upper bounds for CIC_N values of each class, hence enable to measure by eye the gap between *PhySIC_IST* supertrees and fully resolved supertrees of the same size. The plots obtained for the 20 tested conditions show the same trend with slight variations.

The CIC_N values of the PhySIC_IST supertrees decrease as the number of "not-inserted" taxa increases, *i.e.*, as the size of the supertrees decreases. This is expected

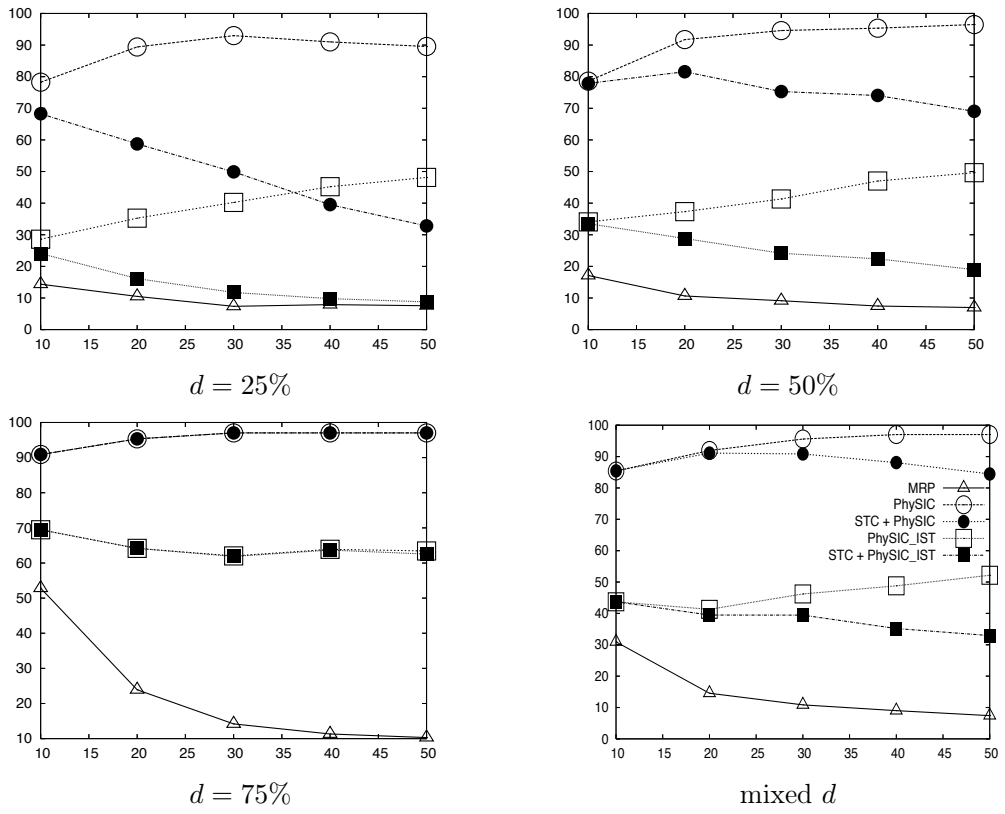


Figure 4.14: **Simulation results: average percentage of type I + type II errors** - Average percentage of type I + type II errors (y-axis) of supertrees built with different supertree methods (MRP △, *PhySIC* ○, *PhySIC_IST* □, STC+*PhySIC* ● and STC+*PhySIC_IST* ■), depending on the number of source trees (x-axis). The results are shown for source trees inferred from data sets in which sequences have been deleted with $d = 25\%, 50\%, 75\%$ and mixed proportions

given the role played by this number in the CIC_N formula (see section *the CIC criterion*). More interestingly, *PhySIC_IST* supertrees overall have CIC values rather close to max CIC values, *i.e.*, *PhySIC_IST* supertree are close to being fully resolved. Moreover, as the size of the supertrees decreases, CIC_N values of *PhySIC_IST* supertrees and max CIC values decrease at a similar pace, the gap between both values narrowing slightly for the smallest supertrees. Thus, overall, the resolution degree of output supertrees appears to be only slightly dependent on the number of taxa inserted in the supertree. The only exception to this rule happens for the conditions $d = 75$ with $k = 10$ and $k = 20$. In these cases, which are the most extreme conditions in terms of overlap between the taxa set of source trees, the two curves decrease with different slopes.
We now detail results obtained when resorting to STC statistical preprocess.

	k=10	k=20	k=30	k=40	k=50
d=25	2.12	3.45	4.87	6.4	7.07
d=50	5.87	3.18	3.51	4.57	5.58
d=75	26.02	21.71	17.89	15.75	14.52
d=mix	10.28	3.8	3.82	4.1	5.25

(a)

	k=10	k=20	k=30	k=40	k=50
d=25	1.21	0.26	0.18	0.06	0.01
d=50	5.73	1.99	1.31	1.08	0.56
d=75	26.02	21.71	17.83	15.73	14.12
d=mix	10.28	3.73	2.7	1.89	1.58

(b)

Figure 4.15: **Average percentage of discarded taxa for supertrees built with *PhySIC_IST* (a) and STC+*PhySIC_IST* (b)** - depending on the deletion ratio and on the number of source trees.

4.3.4.4 Efficiency of the STC preprocessing

Figures 4.11 and 4.12 report simulations results for STC+*PhySIC* and STC+*PhySIC_IST* , when fixing the STC threshold to 95% (see Section 4.3.2.3 for more details). The resolution of both *PhySIC* and *PhySIC_IST* greatly increases thanks to the preprocessing step in most simulation conditions (25%, 50% and mixed deletion ratios d). The STC preprocessing has no effect for $d = 75\%$, where the low overlap between source trees impedes detecting anomalies.

STC+*PhySIC_IST* is on average 1.5 more informative than STC+*PhySIC* according to the CIC_N measure (remember that CIC_N is measured on a logarithmic scale). This replicates the gap observed between the methods without the preprocess, confirming the improvement of *PhySIC_IST* on *PhySIC*. The fact that the STC preprocessing allows the *PhySIC* and *PhySIC_IST* supertrees to be more resolved without significantly changing the type I error, shows that this preprocessing step corrects the source trees in an appropriate way.

When only considering results of STC+*PhySIC_IST* (Figure 4.11), if more information is provided, supertrees are more and more informative, as usually happens with the liberal approach (*e.g.*, see results for MRP and STC+*PhySIC_IST* in Figure 4.11). Indeed, giving more information to STC brings out anomalies more and more clearly, thus tends to modify the source trees more and more accurately.

4.3.4.5 Comparison of liberal and veto methods

As expected, the resolution of supertrees obtained with MRP tends to increase with the number of source trees. In fact, MRP is a liberal method and adding trees supplies more information. Unexpectedly, its type I error does not decrease considerably when adding more trees to the analysis.
As already mentioned, the resolution of supertrees inferred by the two veto methods tends to decrease when including more trees (Figure 4.11, 25%, 75% and mixed deletion rates d). In contrast, their type I error decreases importantly as the num-

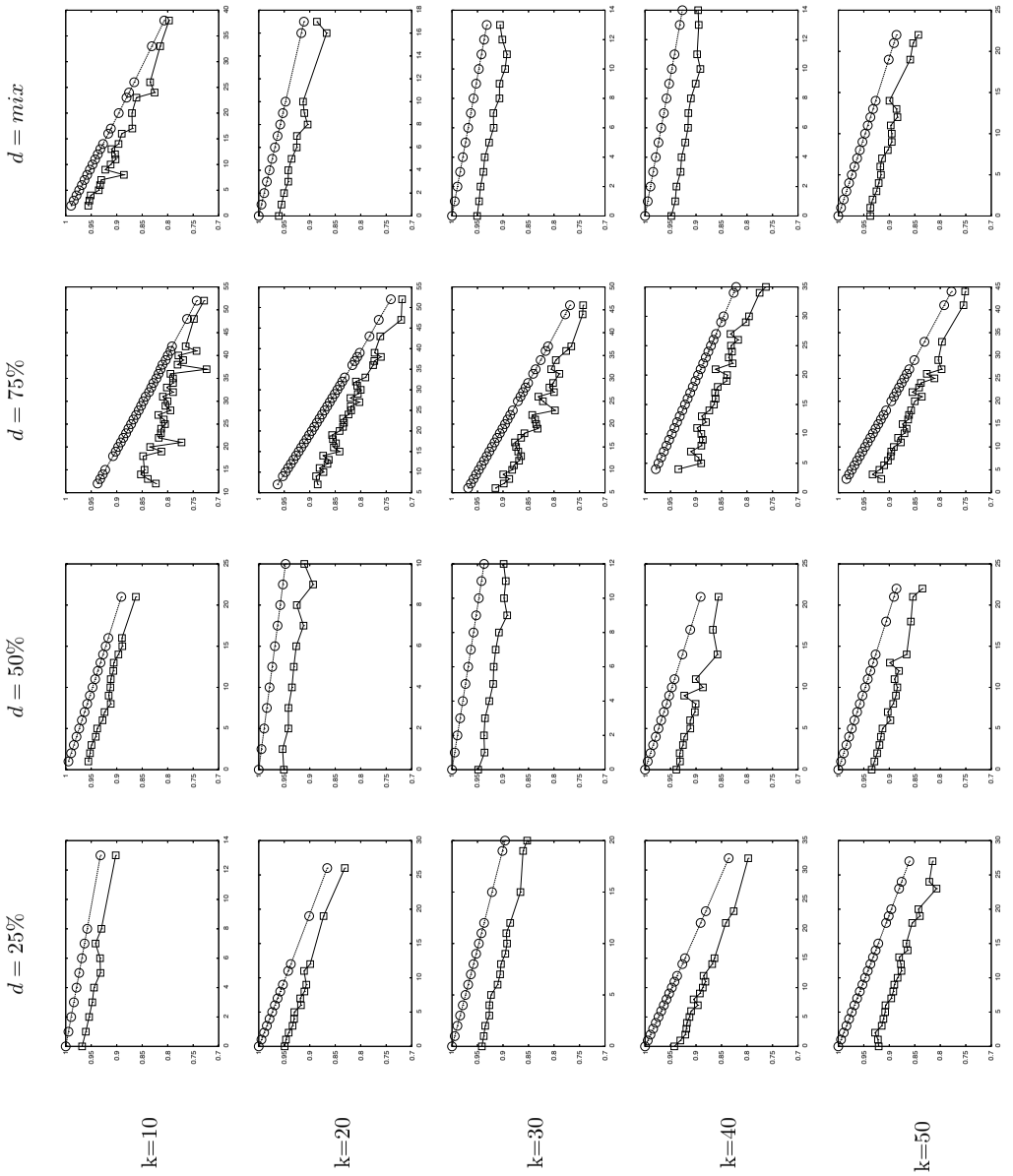


Figure 4.16: Average CIC_N values (denoted by $\square$) plotted as a function of the number of input taxa not inserted in the supertree (x-axis). Max CIC_N values (denoted by $\bigcirc$) indicate the CIC_N value of a fully-resolved tree with the same number of input taxa missing.

ber of source trees increases. By applying the STC preprocessing to *PhySIC* and *PhySIC_IST*, the two methods behave like liberal methods, *i.e.*, the resolution of supertrees increases with the number of trees, as already explained except for $d = 75\%$). This behavior is less apparent for *PhySIC*. Indeed, when faced with an

insufficient number of triplets to satisfy the PI property, *PhySIC* can not benefit from the improvement with respect to PC achieved by the STC preprocess.
Note that in all conditions, MRP provides trees that are, on average, more resolved than other methods. Thus, MRP appears to be the most liberal supertree method among those investigated. This is not a surprise as, when two alternative resolutions conflict with one another, the MRP parsimony criterion favors that supported by the highest number of source trees, while the STC preprocessing favors a resolution only when it is statistically more supported than the other. However, favoring more resolved supertrees also leads to more errors in trees. Indeed, the type I error of *PhySIC* and *PhySIC_IST*, with and without STC preprocessing, is smaller than that of MRP (except for the marginal condition $d = 75\%$ and $k = 10$).

The important question of whether less resolved but more correct supertrees should be preferred to the opposite alternative, can only be answered by knowing the subsequent use of the inferred supertree [see Ranwez et al., 2007a, for a list of cases where the former alternative is to be preferred].

4.3.4.6 Case study focused on placental mammals

To illustrate the effectiveness of *PhySIC_IST* and STC on biological data, we first considered data sets on 12 placental mammals. Primary data was obtained from the OrthoMaM database [Ranwez et al., 2007b] that uses the EnsEMBL (release 41) orthology annotations to identify a set of exonic candidate markers for mammalian phylogenetics. The reliability of the phylogeny inferred from a single marker depends, among other things, on the length of the corresponding sequence alignment. Therefore, we only retained the DNA markers of OrthoMaM associated to the longest sequences, namely those having more than 2000 bp, which provided us with 159 sequence alignments. From the alignments, unrooted phylogenies were then separately inferred with PAUP* [Swofford, 2003] using a maximum likelihood criterion. Using the facilities of our software, we rooted these trees according to one of the two following outgroups: *Monodelphis* or, if it was not present, *Dasypus*, *Echinops* and *Loxodonta* (see Section 4.3.3 for more details). At this step, two of the 159 trees had to be discarded since they did not include monophyletic outgroups. A first supertree data set, called $ortho_{2000}$, was composed of all these source trees. Additionally, we considered a second data set, called $ortho_{3000}$, only composed of the trees obtained from alignment of more than 3000 bp. These two data sets respectively contain 157 and 50 trees, each tree including from 6 to 12 taxa. Figure 4.17 displays the supertrees inferred by *PhySIC_IST* on these data sets, with and without applying the STC preprocessing. The STC threshold has been fixed to 90%.

With exons longer than 3000 bp, the *PhySIC_IST* supertree is extensively multifurcated, with only three obvious clades recovered (Figure 4.17(i)): the two muroid rodents (*Mus* + *Rattus*), the two hominoids (*Homo* + *Pan*), and the catarrhine primates (hominoids + *Macaca*). This reflects the fact that the source trees contain topological conflicts. A closer look at the source trees shows, for instance, that there is likely a long branch attraction phenomenon of the long muroid branch by

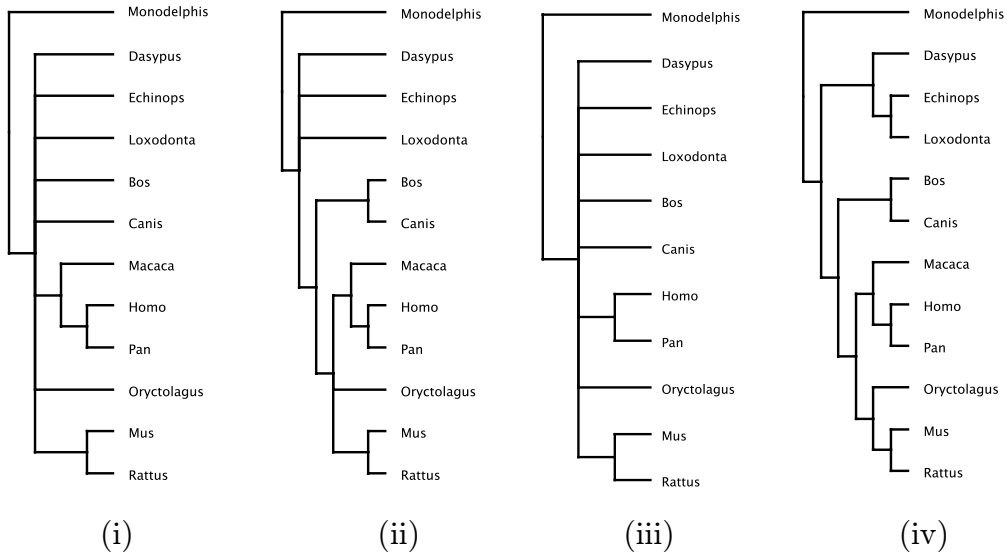


Figure 4.17: **A case study focused on placental mammals** - Supertrees inferred by *PhySIC_IST* from two different collections of source trees. Supertrees in (i-ii) are produced by the *PhySIC_IST* analysis of 50 gene trees obtained from the OrthoMaM database queried for sequences longer than 3000 bp. Tree (i) is inferred without the STC preprocessing while tree (ii) is inferred with this preprocess, setting the threshold to 90%. Supertrees in (iii-iv) are produced from 157 gene trees inferred from sequences longer than 2000 bp. Tree (iii) is inferred without the STC preprocessing while tree (iv) is inferred with STC, setting the statistical threshold to 90%

the marsupial outgroup for the alignment composed of *Pan*, *Macaca*, *Mus*, *Rattus*, *Bos*, *Canis*, and *Monodelphis* exons orthologous to human exon 3 of the CELSR3-SLC26A6 gene (EnsEMBL transcript and exon references ENST00000383733, and ENSE00001498361). In the absence of the rabbit (*Oryctolagus*) ortholog that would break the muroid branch, *Mus* + *Rattus* are artificially attracted towards the basalmost position among placentals. This example illustrates the existence of conflicting resolutions among triplets of different source trees. Thus, without the STC preprocessing, satisfying the PC condition results in a highly multifurcated supertree. In contrast, applying the STC preprocessing leads to a more resolved supertree (Figure 4.17(ii)). The two remaining multifurcations involve (i) the rabbit relative to muroids and primates, and (ii) the armadillo (*Dasypus*), elephant (*Loxodonta*), and tenrec (*Echinops*) relative to the other placentals. This probably reflects the lack of phylogenetic signal for these taxa among the 50 source trees.

With exons longer than 2000 bp, the *PhySIC_IST* supertree is extensively multifurcated, with only two obvious clades recovered (Figure 4.17(iii)): *Mus* + *Rattus* and *Homo* + *Pan*. The greater number of source trees introduces additional conflicts within primates as compared to $ortho_{3000}$. Additionally, the supertree lacks

the taxon *Macaca*. The reason is that, in the source tree reconstructed from the ENSE00001300737 exon (EnsEMBL release 41), *Pan* is unexpectedly more closely related to *Macaca* than to *Homo*. This anomaly appears in only one of the 157 source trees, but this impedes pure veto methods from recovering the correct resolution for the clade. Indeed, inserting *Macaca* while preserving PC, implies losing the clade *Homo* + *Pan*, hence leads to a completely multifurcated tree on the 12 taxa except for the trivial clade *Mus* + *Rattus*. This supertree T' has a CIC_N value inferior to that of the supertree T lacking *Macaca* ($CIC_N(T', 12) = 0.35$ while $CIC_N(T, 12) = 0.435$). For this reason, the taxon *Macaca* is not inserted. In contrast, STC+*PhySIC_IST* infers a plenary supertree (Figure 4.17(iv)), the above-mentioned anomaly being overcome by a significant number of correct resolutions in other source trees. This supertree is also fully-resolved – unlike the supertree obtained from $ortho_{3000}$ – as STC benefits from the signal of 107 source trees additionally present in $ortho_{2000}$. The supertree topology is in agreement with the current view on placental phylogenetics which depicts the monophyly of euarchontoglires (rodents + lagomorphs + primates), laurasiatherians (*Bos* + *Canis*), boreoeutherians (the grouping of the latter two clades), afrotherians (*Loxodonta* + *Echinops*), and xenarthrans (*Dasypus*) + afrotherians [Hallstrom et al., 2007; Murphy et al., 2007; Ranwez et al., 2007b; Wildman et al., 2007].

4.3.4.7 Case study focused on animals

The case study based on OrthoMaM only involved 12 species. To illustrate how PhySIC_IST performs on larger studies, we analyzed an animal phylogenomic data set containing 94 proteins (approximately 20,000 unambiguous amino acid positions) for 79 species, *i.e.*, three poriferans (sponges), 5 cnidarians (sea anemones), and 71 bilaterians (chordates, urchins, mollusks, annelids, flatworms, roundworms, crustaceans, and insects) [Lartillot and Philippe, 2008].

Individual maximum likelihood (ML) protein trees were inferred using Treefinder [Jobb et al., 2004] under the WAG + Γ model of evolution. Among the 94 source trees, 4 (*rpl21*, *rpl37a*, *rpl38*, *rps17*) were discarded because the poriferan outgroup was not monophyletic. The remaining 90 ML topologies were subjected to a PhySIC_IST analysis. To choose the STC threshold, we varied the value of the threshold from 1 to 0.5 and we analyzed the CIC_N values of the resulting supertrees.

Fixing the threshold to a value comprised in the range [0.69, 0.84] leads to the most informative supertree. The topology of the obtained supertree (see Figure 4.18) is in agreement with recent animal phylogenomic studies based on the ML and Bayesian concatenated analyses of conserved proteins under the WAG model of amino acid replacements [Dunn et al., 2008]. For instance, bilaterians are split into protostomians and deuterostomians. Among protostomians, annelids group with molluscs, and crustaceans are paraphyletic due to the grouping of *Artemia* and *Daphnia* with hexapods. Among deuterostomians, Tunicata branches with Vertebrata, and *Xenoturbella* with Ambulacraria. Two taxa are not incorporated, the priapulid *Priapulus* and the nematode *Pratylenchus*. These two taxa are by far

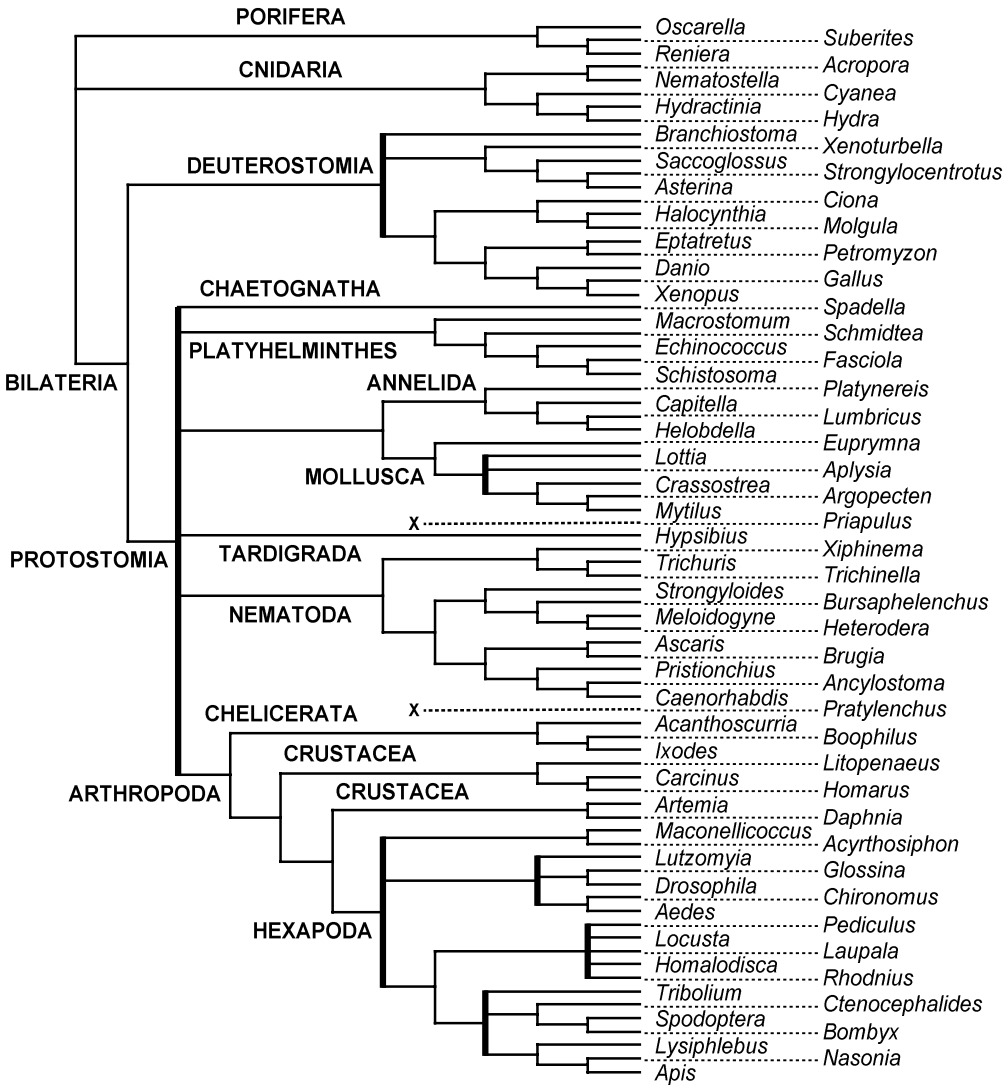


Figure 4.18: **A case study focused on animals** - Supertree reconstructed from the *PhySIC_IST* approach from 90 source trees of a phylogenomic animal data set. The name of the major clades recovered are provided. The two species not incorporated in this non-plenary supertree are indicated by "X". Multifurcations are emphasized by a thicker vertical line.

the less frequent and they are probably not inserted due to a lack of information. Seven multifurcations are displayed by the supertree. This reflects the fact that several source trees were inferred from very short alignments (*e.g.*, *rps28a* possesses 54 sites). The resulting stochastic error yielded a lack of signal and/or contradictions on the position of some taxa, thus diminishing the supertree resolution degree. For instance, the multifurcation involving the 6 major protostomian lineages

reflects the lack of strong signal under the WAG model, whereas the use of a mixture model like CAT provides increased topological resolution with monophylies of Lophotrochozoa (Platyhelminthes, Annelida, Mollusca) and Ecdysozoa (Tardigrada, Nematoda, Arthropoda) [Lartillot and Philippe, 2008].

Note that the *PhySIC_IST* supertree disagrees with the supertree proposed by Lartillot and Philippe [2008] on the relative order of Mollusca, Annelida and Platyhelminthes *i.e.*, Platyhelminthes and Annelida are sister groups in the Lartillot and Philippe supertree while we proposed the more traditional grouping of Annelida and Mollusca. This is due to the fact that we used the WAG model to infer input trees, while recently it has been demonstrated [Lartillot and Philippe, 2008] that for this data set the CAT model has a much better statistical fit than WAG.

4.4 Combining supermatrix and supertree in Triticeea

In this section we present an application of *PhySIC_IST* to the complex problem of disentangling the phylogeny of Triticeae.

This work is issued from a collaboration with Juan S. Escobar, Alberto Cenci, Claire Guilhaumon, Sylvain Santoni, Emmanuel J. P. Douzery, Vincent Ranwez, Sylvain Glémin and Jacques David and it has been submitted at the journal Systematic Biology.

4.4.1 Triticeea: a problematic group

Recent studies have shown that introgressive events (hybridization, gene flows) and incomplete lineage sorting, leading to non-reciprocal monophyly between genes, are more common than previously thought, challenging species concepts, hence historical reconstructions [*e.g.,* Degnan and Rosenberg, 2009; Hudson and Coyne, 2002; Mallet, 2007, Chapter 2].

If gene flows took place in the history of a group and if the genes employed to reconstruct the phylogenetic relationships among genera and species are sampled from introgressed portions of the genome, the trees obtained would likely reflect the history of the introgression rather than the history of the splitting of species lineages. Rapid radiations, especially ancient ones, also challenge phylogenetic reconstructions because of widespread incomplete lineage sorting [Whitfield and Lockhart, 2007].

An appropriate way to handle this problem is through the analysis of the level of congruence among different phylogenies. Some incongruences may only be due to methodological difficulties, such as a reduced number of sampled genes and/or low resolution power of those genes. But others may reflect a true complex, reticulate phylogenetic history involving hybridization and gene flow, and/or rapid diversification and incomplete lineage sorting of ancestral polymorphisms.

A particularly striking example of incongruence among phylogenies when using different genes is provided by Triticeae. Triticeae is a tribe within the Pooideae subfamily of grasses comprising species of major economic importance, including

wheat, barley and rye. Among the world's cultivated species, Triticeae has one of the most complex genetic histories.

In the past years, attempts to reconstruct a reliable phylogeny of the group, based on analyses of single-copy nuclear genes [Helfgott and Mason-Gamer, 2004; Mason-Gamer, 2001, 2005; Petersen and Seberg, 2002], highly repetitive nuclear DNA [Kellogg and Appels, 1995], internal transcribed spacers [Hsiao et al., 1995], and chloroplastic genes [Mason-Gamer et al., 2002; Petersen and Seberg, 1997; Yamane and Kawahara, 2005], have not led to any single definition of groups. Current evidences suggest that different portions of the nuclear genome have different histories, and that the chloroplast genome has yet another one. Because published trees conflict for several taxon positions, it is difficult to obtain a definite picture of the historical relationships among genera and species of this tribe. If the numerous conflicts among published trees are produced by incomplete lineage sorting and/or repeated introgression, it is crucial to know: 1) whether a species phylogeny can be inferred or if reticulate evolution is so complex that this effort would be vain, and 2) to decipher the biological causes of such complex history.

In this chapter, we investigate the methodological and historical problems in the phylogenetic reconstruction of Triticeae by using the most comprehensive molecular data set to date in this group and combining the multigenic supermatrix and supertree approaches. These two approaches, classically seen as competitive ways to analyze large data sets (see Chapter 2), can be used simultaneously in order to exploit the strengths and to counterbalance the weaknesses of each method [Bininda-Emonds, 2004a; Bittner et al., 2008; Comas et al., 2007; Fulton and Strobeck, 2006; Higdon et al., 2007].

The supermatrix approach provides a powerful means of using the evidence from all characters in the final estimation of the phylogeny but it implicitly assumes that all characters have experienced the same branching history, which could not be the case when hybridization, horizontal gene transfer, gene duplication and lineage sorting have played an important role in the history of a group, as could be the case in Triticeae. The supertree approach, on the other hand, does not assume that all genes have experienced the same branching history. We will see in Section 4.4.3.1 that combining a supermatrix analysis of our data set with two supertree analysis we have managed to find a well supported multigenic tree, which clarify the phylogenetic relationships between major clades of Triticeae, compared to the bush of previous single-locus analyses.

In this work, I conducted the *PhySIC_IST* and MRP supertree analyses on different data sets. I also established the procedure to investigate the incongruence between the gene trees and the supermatrix tree to discriminate between gene flow and incomplete lineage sorting as explanation of the complex history of the Triticeae.

4.4.2 Materials and Methods

4.4.2.1 Studied Species and Sampled Loci

Nineteen diploid species, spanning 13 genera of Triticeae were analyzed. One or two accessions per species were analized, making a total of 32 accessions (Table A.1 in appendix A.2). Coding sequences (cDNA) of orthologs of one gene fragment from the chloroplast (*MATK*) and twenty-six nuclear gene fragments, located on three different chromosomes out of the seven chromosomes representative of Triticeae, were sequenced for each accession (Table A.2 in appendix A.2). For more details on how sequences have been obtained and treated see Escobar et al. [2009].

4.4.2.2 Phylogenetic Reconstructions

Raw sequence data were aligned with the Staden Package [Staden et al., 2000], and the resulting alignments were manually corrected. Sequence alignments, for all accessions, were analyzed in two ways: 1) analyses of individual loci; and 2) analyses of concatenated loci (hereafter, supermatrix). The size of the resulting supermatrix was made of 35 sequences and 24,646 aligned sites.

Individual-locus and supermatrix analyses were performed using maximum likelihood (ML) and Bayesian approaches. ML analyses were conducted using the best-fitting model of sequence evolution. Model selection was done based on Akaike's Information Criterion (AIC). PhyML 2.4.4 [Guindon and Gascuel, 2003] was used to obtain the log-likelihood and the phylogenetic trees of the following models: JC69 [Jukes and Cantor, 1969], HKY85 [Hasegawa et al., 1985b], TN93 [Tamura and Nei, 1993] and GTR [Tavaré, 1986; Yang, 1994] (see Section 1.4.1 for a recall). Each model was tested assuming a proportion of invariable sites [Hasegawa et al., 1985b] and a variation among the remaining sites that follows a gamma distribution with shape parameter α [Yang, 1993]. For individual-locus, tree search was performed using the NNI (Nearest-Neighbor Interchange) method, whereas the supermatrix analysis was done using the slower but more extensive tree search based on the SPR (Subtree Pruning and Regrafting) method. Bootstrap analyses (100 replicates for individual loci and 1,000 replicates for the supermatrix) were then performed. Bayesian analyses were performed with MrBayes 3.1.2 [Huelsenbeck and Ronquist, 2001; Ronquist and Huelsenbeck, 2003]. Markov Chain Monte Carlo analyses (MCMC) were run with random starting trees and five simultaneous, sequentially heated independent chains. We used Dirichlet priors (all values were set to 1.0) for base frequencies and for the six substitution rates of the GTR rate matrix; uniform prior distributions for the shape parameter α of the gamma distribution (0.0, 50.0) and for the proportion of invariable sites (0.0, 1.0); an exponential distribution (10.0) for the branch lengths; and a beta prior (1.0, 1.0) for the transition/transversion ratio. All topologies were, *a priori*, equally probable (uniform distribution). In all cases, analyses were run until chains converged (generally several tens of thousands of generations in analyses of individual loci, and up to five million generations for the supermatrix) and a burn-in was established after identifying the

stationary phase, according to the log-likelihood profile.

Majority-rule consensus ML trees on individual loci were used as source trees to construct a supertree according to the MRP method [see Section 3.3.2.1 Baum, 1992; Baum and Ragan, 2004; Ragan, 1992]. Bootstrap trees on individual loci served to construct supertrees according to the MRP and PhySIC_IST methodologies [Scornavacca et al., 2008, see Section 4.3]. In practice, the MRP method permits constructing a supertree close to the supermatrix tree when source trees are relatively congruent and share most species [Criscuolo et al., 2006]. However, unresolved MRP supertrees are often observed when the source trees present several incongruences and/or few overlapping taxa. On the other hand, PhySIC_IST allows the preprocessing of the source trees and the inference of non-plenary supertrees.

MRP supertrees were obtained using the following procedure. The Clann program [Creevey and McInerney, 2005] was used to encode input trees into their binary matrix representations. Maximum parsimony analyses of each matrix were performed using PAUP* [Swofford, 2003] with the following options: 10 random addition sequence replicates, TBR (Tree Bisection-Reconnection) branch swapping and a maximum of 2,000 trees saved per replicate. For all MRP supertrees, 100 nonparametric bootstrap replicates of the initial matrix were generated. For each of these 100 matrices, all most parsimonious trees were saved and weighted by one, divided by the number of equally most parsimonious trees found with this matrix. The final MRP supertree was obtained by performing the weighted majority consensus on the union of those 100 sets of weighted most parsimonious trees. The aim of this weighting scheme is to ensure that each gene tree set has a total weight of 1. On the other hand, PhySIC_IST supertrees were obtained using the PhySIC_IST method with pre-process, with a correction threshold of 99%.

We ran the two supertree methods on three data sets. The first data set comprised the 27 ML trees inferred by PhyML. The second data set included the 2,700 ML bootstrap trees (100 trees per gene). The third data set comprised the 27 majority consensus trees (one per locus) of the 100 ML bootstrap trees of each gene.

4.4.2.3 Quantifying Incongruences

The congruence level of tree topologies obtained from individual and combined loci was assessed by means of SH tests [Shimodaira and Hasegawa, 1999]. Individual alignments were used to compare the topology inferred from individual locus with those obtained from the supermatrix tree and the two supertrees (MRP and PhySIC_IST). Additionally, SH tests using the concatenated sequence of all loci were performed to compare the supermatrix and supertree topologies. Supertrees were tested with and without polytomies. Polytomies were resolved by bipartitions because they are strongly penalized in the log-likelihood score. SH tests were run in the BASEML program implemented in the PAML 4.1 package [Yang, 2007].

In addition, we used the χ^2 test of the PhySIC_IST pre-process (see Section 4.3.2.3) to identify triplets of accessions observed in the supermatrix tree that are

strongly rejected by the 27 bootstrap gene tree collections. A strong rejection is defined as follows. Denoting R_s the set of triplets of the supermatrix, and R_b the set of triplets of the 2,700 bootstrap gene trees, a triplet of R_s is said to be strongly rejected if it contradicts at least one triplet of R_b and fails the χ^2 test, with a threshold of 0.99. This measure of gene incongruence was estimated only with respect to the supermatrix tree since, according to the SH tests, this is the multigenic tree that best describes the evolution of concatenated sequences (see our results below). Using this procedure, we counted the number of strongly rejected triplets that contain a given taxon and obtained the list of strongly rejected triplets per clade. After having replaced each taxon by the clade it belongs to, we also counted the number of strongly rejected triplets that contain a given clade. This provides an overview of conflicts at the taxon and clade levels, respectively.

To quantify the degree of incongruence between the phylogenetic signal of an individual locus and the whole supermatrix, we defined a triplet-based distance between the supermatrix tree (T_s) and the forest (F_j) of the 100 bootstrap trees obtained for the gene j. To put it simply, the triplet distance represents the percentage of triplets that are resolved differently by the supermatrix tree and a gene tree. In order to separate the signal of this gene from stochastic errors, we focused on triplets that appear more than 50 times in F_j. We denoted by $N_{eq(T_s,F_j)}$ the number of retained triplets that have the same resolution as T_s and by $N_{diff(T_s,F_j)}$ the number of those having a different one. We define the distance between the tree T_s and the forest F_j, denoted by $d(T_s,F_j)$, as the triplet fit similarity [Page, 2002] between the triplet set of T_s and the most supported triplets of F_j:

$$d(T_S, F_j) = \frac{N_{diff(T_s,F_j)}}{N_{eq(T_s,F_j)} + N_{diff(T_s,F_j)}} \tag{4.1}$$

Using similar procedures, we computed the triplet distance between all pairs of individual genes. We defined a triplet-based distance between each couple of forests F_i and F_j, where F_i and F_j are, respectively, the forests of the 100 bootstrap trees obtained for the gene i and j. As above, we focused on triplets that appear more than 50 times in each forest in order to eliminate stochastic errors. If $N_{eq(Fi,F_j)}$ is the number of retained triplets that have the same resolution in F_i and F_j, and $N_{diff(Fi,F_j)}$ is the number of those having a different one, then the distance $d(F_i,F_j)$ between F_i and F_j is:

$$d(F_i, F_j) = \frac{N_{diff(F_i,F_j)}}{N_{eq(F_i,F_j)} + N_{diff(F_i,F_j)}} \tag{4.2}$$

In this way, we obtained a symmetric distance matrix M with 27 rows and 27 columns, where each entry M_{ij} contains the triplet distance between genes i and j.

4.4.2.4 Analyses of Patterns of Incongruence

In order to understand the origin of incongruences, we correlated triplet distances between individual genes and the supermatrix tree ($d(T_s,F_j)$ in equation 2) to relevant phylogenetic parameters, including the alignment length, the proportion of

variable sites, the average evolutionary rate [Criscuolo et al., 2006, estimated according to the super-distance matrix methodology,], and the shape parameter α of the gamma distribution, obtained in the analysis of individual loci with PhyML. We additionally tested if incongruences are positively correlated with recombination, using the 21 genes located on chromosome 3. This correlation is expected for two reasons. First, following interspecific hybridization, recombination is necessary for genes of one species to introgress into the other species. Second, because the effective population size is expected to be smaller in low recombining regions than in highly ones [Charlesworth, 2009; Presgraves, 2005], coalescence is expected to be quicker and lineage sorting more complete when recombination is low.

For each locus located on chromosome 3, the genetic distance between the locus and the centromere[1] was computed. We do not discuss here how these genetic distances were computed but this is detailed in section *Location of Loci on the Triticeae Genome* of Escobar et al. [2009]. The values of these distances can be found in Table A.2 of Appendix A.3.

We thus tested if $d(T_s,F_j)$ is lower in centromeric than telomeric regions, by fitting the quadratic regression of $d(T_s,F_j)$ on the genetic distance. We also performed the same analyses on the phylogenetic parameters because recombination could affect incongruences indirectly through these parameters (*e.g.*, higher evolutionary rates in highly recombining regions).

Finally, whatever the underlying mechanism, closely linked genes are more likely to share a common genealogical history than distant ones. To test this, we constructed a matrix of genetic distance between pairs of loci for the 21 genes located on chromosome 3. We correlated this matrix to the matrix of incongruences by pairs (M_{ij} only for genes on chromosome 3) and tested the significance of the correlation by performing 10,000 permutations of gene locations on each chromosome arm, that is, without permuting one gene from one arm to another. All statistical analyses were done with R version 2.6.0 [R-Development-Core-Team, 2006].

4.4.3 Results

4.4.3.1 Phylogenetic Reconstruction of Triticeae: Individual Loci vs. Multigenic Approaches

The best models describing the evolution of the individual loci and the corresponding trees obtained under such models are not presented here but can be found respectively in Supplementary Tables 2 and 3 of Escobar et al. [2009].

Phylogenetic reconstructions using individual loci produce contrasted topologies. Often, relationships among genera and species are not congruent among genes. In some cases, dramatic changes in the position of species are found. For instance, the tree obtained with the locus LOC_Os01g01790 groups the three *Hordeum* species,

[1] The centromere is the chromosomal locus where two identical sister chromatids come in contact, typically found near the middle of a chromosome, and is the most condensed and constricted region of a chromosome. The telomere is a non-coding region of repetitive DNA at the end of a chromosome, involved in the replication and stability of the chromosome.

a genus thought to be one of the deepest among Triticeae [see our results below; Mason-Gamer et al., 2002; Petersen and Seberg, 1997; Petersen et al., 2006], with *Secale* and *Triticum*, two genera that should cluster within one of the most derived clades [see our results below; Kellogg et al., 1996; Petersen et al., 2006]. Likewise, the tree obtained with the locus LOC_Os01g24680 places *Psathyrostachys*, the genus that seems to be the sister group to the rest of the tribe [see our results below Kellogg and Appels, 1995; Mason-Gamer et al., 2002; Petersen et al., 2006], together with *Henrardia* and *Eremopyrum bonaepartis*, two more recently diverging taxa. Several other odd relationships are displayed by individual gene trees. In general, individual gene trees display short internal branches compared with external branches (*i.e.*, low treeness). In addition, support values (bootstrap values and posterior probabilities) of deep nodes are weaker than those of more recent nodes. Similar observations have been previously made when comparing phylogenies obtained with different genes in Triticeae [Kellogg et al., 1996].

Multigenic approaches provide a much more robust picture than individual gene trees. ML analyses reveal that the best model describing the evolution of the supermatrix is GTR with gamma distribution (log likelihood = -92,992.38). Mean base frequencies are 27.89% A, 21.13% C, 24.22% G and 26.76% T, and the shape parameter α of the gamma distribution is 0.294. The Bayesian analysis of the supermatrix produces exactly the same topology as ML and very similar parameters (harmonic mean of marginal log-likelihoods: -93,050.24; base frequencies: 27.82% A, 20.82% C, 24.19% G and 27.17% T; shape parameter α of the gamma distribution: 0.295). As previously noted [Douady et al., 2003; Erixon et al., 2003], some nodes with relatively low bootstrap values are fully supported according to posterior probabilities. The supermatrix tree is presented in Figure 4.19. Within Triticeae, five to seven well supported clades can be distinguished, depending on posterior probability or bootstrap supporting values of the nodes. The first divergent group within Triticeae is *Psathyrostachys* (clade I), followed by *Hordeum* (clade IIA) and *Pseudoroegneria* (clade IIB). Then, internal branches are quite short compared with external branches, suggesting that species split in rapid succession. Two well-distinguishable clades diverge at this point. The first is formed by *Australopyrum* (clade IIIA) and a sub-clade denoted clade IIIB. This latter gathers *Henrardia* and *Eremopyrum bonaepartis*, on the one hand, and *Agropyrum* and *Eremopyrum triticeum*, on the other hand. The second of those two well-distinguishable clades consists of *Dasypyrum* and *Heteranthelium* (clade IV), on the one hand, and *Secale*, *Taeniatherum*, *Triticum* and *Aegilops* (clade V), on the other hand.

The supertrees proposed by MRP and PhySIC_IST on the 27 ML trees (one per locus) are poorly resolved (not shown). This comes from the fact that many branches retrieved in ML analyses are poorly supported by individual locus data and these branches have the same influence on the inferred supertrees as the well supported branches. We overcame this difficulty by running analyses on the bootstrap trees of the ML analyses. Even if these trees are not independent, they consistently improved MRP and PhySIC_IST outputs. Running PhySIC_IST on the 2,700 bootstrap trees (100 trees per each of the 27 loci) leads to a more resolved

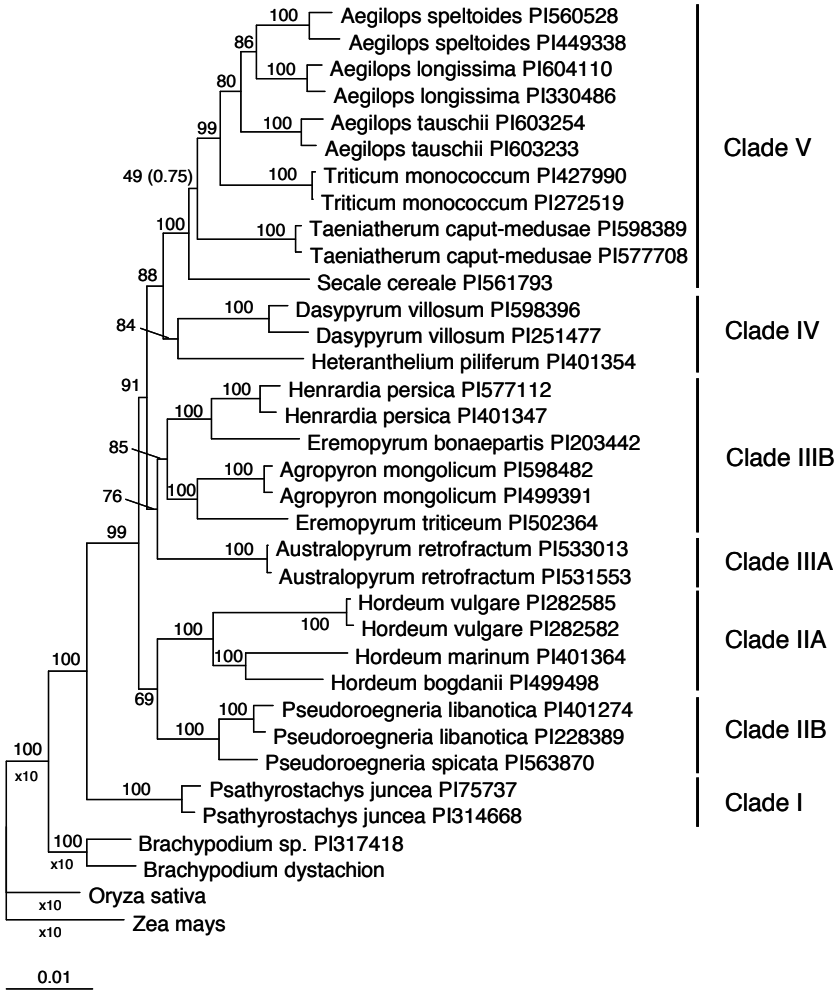


Figure 4.19: **Supermatrix phylogeny of Triticeae** - Bootstrap values are given in percentage. Full posterior supports (100%) are found for all nodes, excepting one that is indicated between brackets. Note that the branch lengths of the outgroups are divided by 10 in order to zoom on Triticeae.

supertree that does not contradict the supermatrix tree (Figure 4.20). Indeed, having more input trees increases the statistical power of the PhySIC_IST pre-process test, allowing discriminating stochastic errors from phylogenetic signal. The MRP supertree obtained on the 2,700 bootstrap trees (figure not shown, see Supplementary Figure 1 of Escobar et al. [2009]) is in contradiction with both supermatrix and PhySIC_IST trees. The PhySIC_IST supertree obtained with the 27 majority consensus trees (one per locus) of the 100 ML bootstrap trees of each gene fragment

(figure not shown, see Supplementary Figure 2 of Escobar et al. [2009]), though less resolved, is in agreement with that obtained by PhySIC_IST on the 2,700 bootstrap trees. The MRP supertree on this data set (27 majority consensus trees of the 100 ML bootstrap trees; Figure 4.21) is in agreement with the supermatrix tree. MRP clade reliability estimations are better on this data set, while the PhySIC_IST pre-process performs better on the previous one (2,700 bootstrap trees).

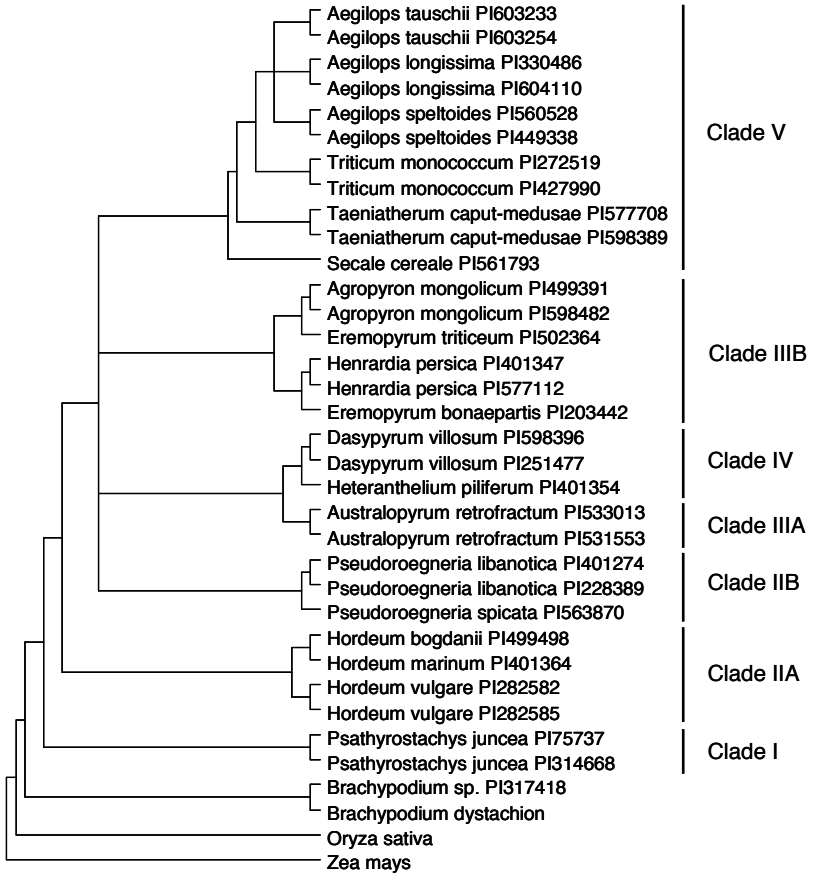


Figure 4.20: **PhySIC_IST supertree obtained with the analysis of the 2,700 ML bootstrap trees (100 trees per gene)** - Clades are named as in Figure 4.19.

According to the PhySIC_IST supertree presented in Figure 4.20, *Psathyrostachys* (clade I) and then *Hordeum* (clade IIA) are the first divergent groups within Triticeae. Like the supermatrix analysis, it retrieves clades IIB (*Pseudoroegneria*), IIIB (*Henrardia*, *Eremopyrum bonaepartis*, *Agropyrum* and *E. triticeum*) and V (*Secale*, *Taeniatherum*, *Triticum* and *Aegilops*), though resolution among *Aegilops* species in clade V is weak. Unlike the supermatrix tree, PhySIC_IST infers a clade formed by *Australopyrum*, *Heteranthelium* and *Dasypyrum*. This clade

forms a polytomy in the middle of the tree together with clades IIB, IIIB, and V.

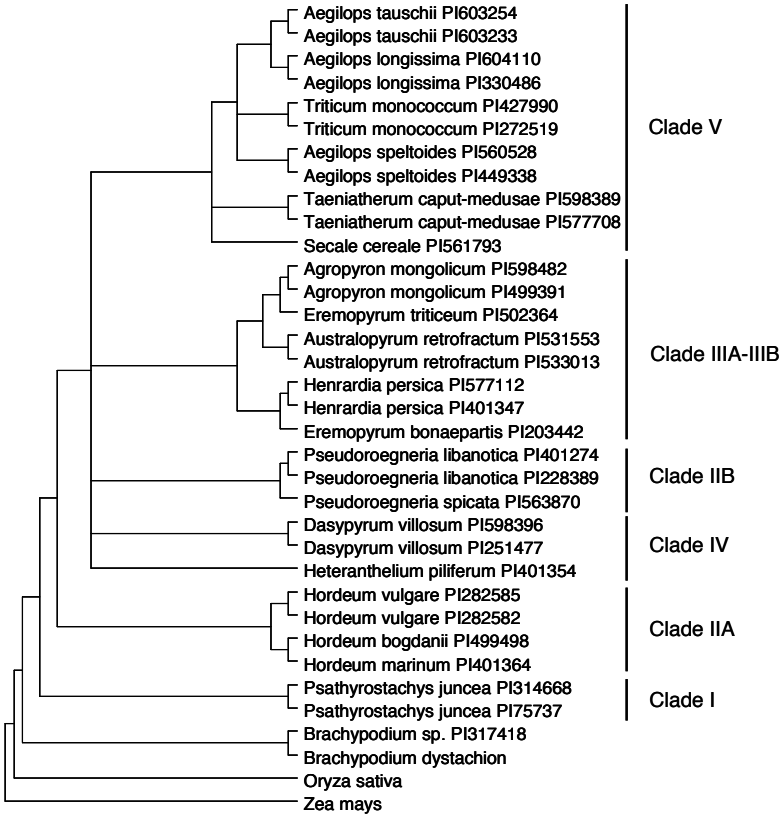


Figure 4.21: **MRP supertree obtained by assembling the 27 majority consensus trees (one per locus) of the 100 ML bootstrap trees of each gene.** - Clades are named as in Figure 4.19.

The MRP supertree presented in Figure 4.21 shows that *Psathyrostachys* (clade I) is the first divergent genus within Triticeae, followed by *Hordeum* (clade IIA); clade V (*Secale*, *Taeniatherum*, *Triticum* and *Aegilops*) is also retrieved, though with poor resolution. This tree exhibits a multifurcation involving clades IIB (*Pseudoroegneria*), IIIA–IIIB (*Australopyrum*, *Henrardia*, *Eremopyrum bonaepartis*, *Agropyrum* and *E. triticeum*) and IV (*Dasypyrum* and *Heteranthelium*). Unlike PhySIC_IST, the MRP supertree does not retrieve *Australopyrum* as the sister group of clade IV.

Despite some differences between supermatrix and supertree phylogenies, the resolution and support gained with multigenic approaches compared with single-locus analyses are striking. Notably, tree topologies are congruent in most cases and differences among trees are mainly due to the lower resolution of supertrees

compared to the supermatrix tree. However, an important difference regarding the position of *Australopyrum* (clade IIIA) is observed among the multigenic phylogenies: it is found either basal to clades IIIB (Figure 4.19) or IV (Figure 4.20), or as the sister clade of *Agropyrum–Eremepyron triticeum* (Figure 4.21). SH tests applied to the concatenated sequence show that the supermatrix tree explains significantly better the phylogenetic relationships among Triticeae than do PhySIC_IST (on 2,700 bootstrap trees) and MRP (on 27 ML consensus trees) supertrees. This was true when analyzing raw supertrees (*i.e.*, supertrees containing polytomies). However, statistical significance disappeared after enforcing binary resolution of supertree polytomies (data not shown, see Supplementary Table 4 of Escobar et al. [2009]). It follows that the position of *Australopyrum* is not significantly better in the supermatrix tree than those proposed by the supertree methods. The exact position of this taxon remains thus an open question.

When more than one accession per species was available, both supermatrix and supertree analyses group them together. External branches are long and there is no ambiguity in the taxonomic status of species. More interestingly, though the sampling of this study was not specifically designed to test monophyly of genera, when several species were available for a given genus, they generally branch as monophyletic groups (*e.g.*, *Aegilops*, *Hordeum* and *Pseudoroegneria*). There are only two exceptions: *Aegilops* for the MRP analysis, and *Eremopyrum*, which splits in two different clades in all multigenic analyses, one including *E. bonaepartis* and *Henrardia*, and the other including *E. triticeum* and *Agropyrum*. In addition, *Aegilops speltoides*, which was thought to be the sister group of the *Aegilops/Triticum* clade [Petersen et al., 2006; Yamane and Kawahara, 2005], is grouped, in all our analyses, with other *Aegilops* species, whereas *T. monococcum* branches at the base of this group.

4.4.3.2 Patterns of Incongruence among Trees

One of the most striking results obtained in this study are the numerous incongruences between individual locus and multigenic trees. In most cases, the conservative SH test confirms that, regarding the locus alignment, the corresponding gene tree has a log-likelihood significantly better than that of trees obtained using the supermatrix and supertree approaches (data not shown, see Supplementary Table 4 of Escobar et al. [2009]). The high level of incongruence between the gene trees and the supermatrix tree clearly confirms that single-gene evolutionary signal significantly contradicts the signal of concatenated sequences. In order to quantify these incongruences, we estimated triplet distances between individual gene trees and the supermatrix tree. The average triplet distance (in absolute value) across all genes is 0.23 ± 0.08 (mean ± SD; range: 0.11–0.48; Table A.2 in appendix A.2). For each accession, we counted the number of triplets strongly rejected by the supermatrix tree (excluding the outgroups). Except the two *Psathyrostachys* accessions, all other taxa are involved in several strongly rejected triplets (38.0 ± 23.6; median = 29; range: 15–113; Table A.3 in appendix A.2). *Pseudoroegne-*

ria, *Hordeum* and *Australopyrum* exhibit the highest number of incongruences. We then pooled accessions according to the clade of origin in the supermatrix tree (Figure 4.19) and counted the number of incongruent triplets. This allows detecting two major incongruent triplets among clades: (1) clade IIA, clade IIB | clade IIIB (*i.e.*, *Hordeum*, *Pseudoroegneria* and *Agropyrum–Eremopyrum–Henrardia*; 68 occurrences); and (2) clade IIA, clade IIB | clade V (*i.e.*, *Hordeum*, *Pseudoroegneria* and *Secale–Taeniatherum–Triticum/Aegilops*; 119 occurrences) (Table A.4 in appendix A.2). Interestingly, both these incongruent triplets involve clades distantly related. Indeed, distantly related clades account for 276 incongruent triplets out of 367 (or 75% of incongruences), while closely related and adjacent clades account for 91 incongruent triplets (or 25% of incongruences).

4.4.3.3 The Effect of Recombination on Incongruences

In order to understand the origin of incongruences, we investigated why some genes are more incongruent than others. First, we tested if variation in incongruence can be simply explained by the nature of the phylogenetic signal. Among the correlations between the triplet distance between individual genes and the supermatrix tree, and the relevant phylogenetic parameters per locus, we only detect a significant positive correlation between triplet distance and the average evolutionary rate (r = 0.28, P= 0.04; data not shown, see Supplementary Table 5 of Escobar et al. [2009]). As expected, rapid evolving genes are more incongruent than slower ones. Then, we investigated the effect of the recombination. Recombination does not significantly affect any phylogenetic parameter (not shown). On the contrary, it affects incongruences in two ways. First, incongruences between single gene trees and the supermatrix tree tend to increase with genetic distance, hence the likelihood of recombination (P = 0.042 on the full data set, P = 0.027 after removing one potential outlier; Figure 4.22(a). Second, genes closely located on the chromosome tend to have more similar genealogical histories than distant ones (Figures 4.22(b) and 4.22(c); Pearson's r= 0.23, P= 0.028; Spearman's ρ = 0.22, P= 0.027).

4.4.4 Discussion

Up to now, morphological and molecular analyses have failed to reconstruct a reliable phylogeny describing the history of the splitting of species lineages in Triticeae. Most previous phylogenetic reconstructions are based on a limited number of genes, in most cases only one (see references above). The numerous conflicts among published trees, combined with a poor resolution of branching among genera and species, impede drawing a clear picture of the basic relationships among members of this tribe. In this chapter, we show that the gene tree-species tree problem is a major obstacle that must be considered in the phylogenetic analysis of Triticeae. More generally, this problem is a major difficulty when reconstructing the phylogeny of any group in which ancestral polymorphism was not unambiguously sorted for every gene or in which important introgressive events have taken place.

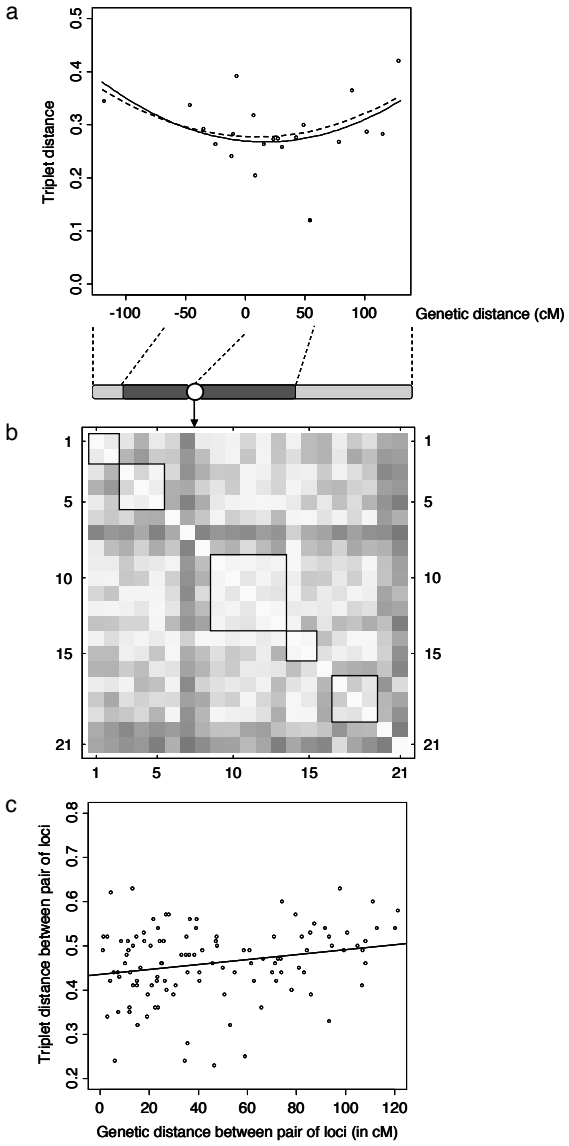


Figure 4.22: **Effect of the likelihood of recombination on incongruences** - a) Relationship between the triplet distance of individual gene trees and the supermatrix tree as function of the genetic distance between genes on the chromosome 3. Solid line: best fit using all points; dashed line: best fit without a purportedly outlier (filled point). The genetic distance is connected to the chromosomal position according to the schematic diagram presented in the lower part (white point: centromere; black: centromeric region; grey: telomeric region). (b) Degree of incongruence among pairs of loci relative to the genetic distance on the chromosome 3. Colors represent the degree of incongruence (white: no incongruence; deep grey: the strongest incongruence). (c) Correlation between the triplet distance and the genetic distance between pairs of loci. Only loci located on chromosome 3 are depicted.

Though it is well known that hybridization is a widespread phenomenon in plants [Rieseberg, 1997; Rieseberg et al., 2000] and is an important source of incongruence in phylogenetic reconstruction, large multigenic phylogenies of plants remain surprisingly rare compared, for instance, with metazoans [but see Zou et al., 2008]. We have shown that multigenic approaches, combining information of several genes located in different chromosomes and cellular compartments (nucleus and chloroplast),

provide much more resolution than the analysis of individual loci and permit resolving the evolution of major clades in Triticeae. Though position of some groups is still uncertain (*e.g.*, *Australopyrum*), the combination of supermatrix and supertree methodologies allowed reconstructing a robust phylogeny of Triticeae, pointing out the evolution of five (if considering posterior probabilities) to seven (for bootstrap values) supported major clades. Most clades we suggest have already been proposed in one study or another, most of the time with weak support. However, it was not possible to choose among the numerous conflicting results previously obtained.

4.4.4.1 Multigenic Phylogeny of Triticeae

According to this phylogeny, *Psathyrostachys* is the sister-group of the remaining Triticeae, followed by the sequential branching of *Hordeum*, with or without *Pseudoroegneria* (compare Figure 4.19 with Figures 4.20 and 4.21). *Psathyrostachys* is involved in no incongruence (Table A.3 in appendix A.2) and the branch leading to the rest of the tribe is among the longest internal branches (Figure 4.19).

This clearly demonstrates that this group diverged before the diversification of other Triticeae. Most previously published phylogenies agree with the early diverging of *Psathyrostachys* and *Hordeum* [*e.g.*, Kellogg and Appels, 1995; Mason-Gamer, 2001; Mason-Gamer et al., 2002; Petersen and Seberg, 1997; Petersen et al., 2006], though other studies contradict this branching [Hsiao et al., 1995; Mason-Gamer, 2005; Petersen and Seberg, 2002; Petersen et al., 2006; Seberg and Frederiksen, 2001]. The position of *Pseudoroegneria* is, in contrast, more debated. No study has raised the possibility that *Pseudoroegneria* branches out with *Hordeum*. Indeed, in all previous phylogenetic studies of Triticeae, *Pseudoroegneria* branched out at variable positions. In some cases, it was proposed as the sister group of *Taeniatherum* and/or *Australopyrum* [Mason-Gamer, 2001; Petersen and Seberg, 2002; Petersen et al., 2006], *Heteranthelium* [Mason-Gamer, 2005] or *Aegilops* [Seberg and Frederiksen, 2001]; in other cases it branched within complex clades [Kellogg et al., 1996; Petersen and Seberg, 1997] and it was even considered paraphyletic in one study [Mason-Gamer et al., 2002]. Consistent with the difficult positioning of this genus, the supermatrix tree groups it with *Hordeum* with a rather weak bootstrap support (0.69), conflicting with supertrees, though full posterior probability (1.00). More strikingly, the three *Pseudoroegneria* accessions are involved in much more incongruent triplets than other species (Table A.3 in appendix A.2). This could be due to a very large ancestral population size or a strong capacity of introgression during the divergence of this group. Whether *Pseudoroegneria* forms a monophyletic group together with *Hordeum* or constitutes a separate clade by itself, we provide evidence supporting a rather basal position of this genus within Triticeae.

The phylogenetic positions of all other species were very variable in previous studies, and almost no consensus emerged. Here, we found strong support for four clades (IIIA, IIIB, IV, and V on Figures 4.19 and 4.20), both using the supermatrix and the PhySIC_IST approaches, while only clade V is fully retrieved in the MRP tree (Figure 4.21). The relationships among these clades are more difficult to de-

termine. In the supermatrix approach, the support is a bit weaker for the internal nodes linking these four clades than for the basal node of each clade. Moreover, the PhySIC_IST tree shows a polytomy between these four clades, including *Pseudoroegneria* (clade IIB). This is congruent with the very short internal branches in this part of the tree and the numerous incongruences involving distantly related clades (Table A.4 in appendix A.2). Overall, this suggests a rapid radiation following or concomitant to the divergence of *Pseudoroegneria*.

Interestingly, the relationships within each clade are more resolved than those among them with both the supermatrix and PhySIC_IST approaches (Figures 4.19 and 4.20). This suggests that subsequent diversifications were more gradual. The *Aegilops*/*Triticum* group could be an exception. Even though our sampling does not reflect the diversity of this clade, the most resolved multigenic phylogenies (*i.e.*, supermatrix and PhySIC_IST trees) do not support the paraphyly of the genus *Aegilops* observed in previous studies based on nuclear and chloroplastic genes [Mason-Gamer, 2001, 2005; Petersen et al., 2006; Yamane and Kawahara, 2005]. However, the relationships among *Aegilops* species are not well resolved, and more work should be done on this genus.

Unlike previous studies in Triticeae, our multigenic phylogenies provide strong support for most nodes and the above described relationships among genera and species. Importantly, we give support to a clade not detected before consisting of *Hordeum* and *Pseudoroegneria*, suggested by the supermatrix tree. Excepting the relationship between these two genera, several other relationships were already present in previous studies. However, the numerous conflicts among previous trees make very difficult distinguishing robust relationships from phylogenetic noise. Our study is the most robust phylogenetic study to date in Triticeae and we hope it will constitute a backbone for future phylogenetic studies in this tribe. Though our sampling was sufficiently informative about diversity among genera, it was not exhaustive of the specific diversity of the tribe. We recommend future studies to position additional species within this phylogenetic framework. Supertree methods (*e.g.*, PhySIC_IST) could be an appropriate way to incorporate new data to the current phylogeny.

4.4.4.2 Incongruences among Trees and the Complex History of Triticeae

Up to this point, we have shown that methodological problems, due to the use of a reduced number of genes and/or the use of genes with low resolution power, have lead to the numerous conflicts among previous phylogenetic studies in Triticeae. However, it seems that conflicts among source trees are not only due to methodological problems but also to a complex evolutionary history. We provide evidence that the relationships among members of the tribe for a given locus are generally better explained by the tree inferred with that locus than by any of the multigenic trees. This reflects a complex biological reality, where different portions of the genome exhibit different histories (their own phylogenetic histories) and the

supermatrix tree should be a reasonable compromise among all these scenarios to depict the splitting history of species lineages [but see Degnan and Rosenberg, 2006, 2009]. Incongruences can be due to gene properties. For instance, we showed that rapidly evolving genes are more incongruent than slowly evolving ones. Noteworthy, we also pinpointed the role of recombination and gene locations along the chromosome. As expected, closely linked genes are more likely to share a common history than distant ones (Figure 4.22). Genes located in centromeric regions tend to be more congruent with the supermatrix tree than those located in telomeric regions. Such correlation has already been found in *Drosophila* species at the kilobase scale, the scale of linkage disequilibrium in these species [Pollard et al., 2006]. It could be surprising that such a correlation still holds at the scale of a whole large chromosome [˜1 Gb Paux et al., 2008]. On the contrary, in the *Oryza* genus (rice species), the mosaics of conflicting genealogies are distributed randomly over the genome [Zou et al., 2008]. Several non exclusive reasons can explain this pattern. First, the recombination gradient along the chromosomes is very steep in all Triticeae species studied so far, including wheat [Akhunov et al., 2003a,b; Luo et al., 2000], rye [Lukaszewski and Curtis, 1993] and *Aegilops speltoides* [Luo et al., 2005]. For instance, along the 3B chromosome in bread wheat, the cM/Mb ratio spans about two orders of magnitude, from 0.01 to 0.85 [Saintenac et al., 2009]. Despite the impressive chromosome size, linkage disequilibrium (LD) can be high in centromeric regions. Accordingly, in bread and durum wheat (*Triticum aestivum* and *T. durum*, respectively), LD decays slowly over several cM [Somers et al., 2007]. However, the level of LD is low in barley [Morrell et al., 2005]. Second, centromeric genes may have lower local effective size than telomeric ones, because of hitchhiking effects due to the lack of recombination [Charlesworth, 2009; Presgraves, 2005]. In agreement with this prediction, Dvorak et al. [1998] showed that in *Aegilops* species, recombination gradients affect levels of diversity. RFLP polymorphism is 1.5 to 25 times higher in telomeric regions than in centromeric ones. Consequently, ancient polymorphisms would be less completely sorted in genes located in highly recombining regions than in lowly ones. Finally, recombination could play an important role in introgressive events between species (*e.g.*, genes located in highly recombining regions introgress easier than genes located in low recombining regions). Though it is difficult to distinguish gene flow from incomplete lineage sorting, we do not consider gene flows as the most likely scenario explaining the bulk of incongruence among gene trees in Triticeae. On the contrary, we favor incomplete lineage sorting. Two lines of reasoning support this. First, under the incomplete-lineage-sorting hypothesis, we expect internal branches of individual gene trees to be shorter and less supported than external branches. This was basically what we observed in the analysis of individual gene trees and the supermatrix tree. Note that the high support values in the supermatrix tree are due to the combined phylogenetic signal of all loci. This suggests that speciation occurred in rapid succession in a short-time period (divergence of the ancestor of Triticeae is estimated to have occurred ˜12–15 Mya, given that wheat-barley divergence could have occurred ˜10 Mya, Dvorak and Akhunov [2005]). Second, most observed incongruences between individual gene

trees and the supermatrix tree occurred among distantly related clades (75% of incongruences in our data set), separated several million years ago (*e.g.*, *Hordeum*, *Pseudoroegneria* and *Secale–Taeniatherum–Triticum/Aegilops*). Such a pattern is difficult to explain by gene flows alone, all the more that these genera are currently largely intersterile. Indeed, cytogenetic studies have shown that diploid genera of Triticeae are genomically distinct, that is, their chromosomes do not pair well if at all at meiosis [Fernandez-Calvin and Orellana, 1992; Waines and Barnhart, 1992; Wang, 1989, 1992]. Though we presume that gene flow is a mechanism still occurring among closely related taxa (*e.g.*, *Aegilops*/*Triticum*) and explaining incongruences at this level (in our case, 25% of the observed inter-clade incongruences), it does not seem to be the general mechanism explaining the bulk of incongruences observed in Triticeae. In summary, as in *Drosophila* [Pollard et al., 2006] and *Oryza* [Zou et al., 2008], we consider that the majority of incongruences among trees in Triticeae are due to incomplete lineage sorting of ancient polymorphisms.

4.5 Conclusions

In this chapter we introduced PI and PC, two strict and desirable properties that a conservative supertree method should satisfy. Moreover, we presented two supertree methods *PhySIC* and *PhySIC_IST* [Ranwez et al., 2007a; Scornavacca et al., 2008] that infer supertrees satisfying these desirable properties.

PhySIC is a supertree method that enables the user to quickly summarize consensual information of a set of reliable trees. Moreover, since polytomies of the produced supertree are tagged by labels indicating areas of conflict as well as areas with insufficient overlap, *PhySIC* enables the user to localize groups of taxa for which the data requires consolidation.

PhySIC has been proposed mostly to show that it was possible to design a quick supertree method satisfying PI and PC. Indeed, in Ranwez et al. [2007a] the emphasis is given to these properties rather than to the *PhySIC* method. We then relied on this theoretical framework to develop *PhySIC_IST*, an improved supertree method searching for the most informative supertree satisfying PI and PC.

The improvement of *PhySIC_IST* on *PhySIC* shown in Figure 4.11 on page 109 is a consequence of three fundamental differences between *PhySIC* and *PhySIC_IST*. First, the new version operates successive insertions of taxa on a backbone and is not based on a revised version of the Build algorithm [Aho et al., 1981]; ergo, *PhySIC_IST* can frequently find relations between taxa that *PhySIC* cannot detect, being stopped in this analysis by a connected component of the Aho graph. In addition, the two methods do not have the same optimization criterion: indeed, *PhySIC* aims at finding the supertree satisfying PI and PC that proposes a resolution for as many triplets as possible, while *PhySIC_IST* looks for the supertree satisfying PC and PI that maximizes the value of *CIC*. Last, *PhySIC_IST* can propose non-plenary supertrees, *i.e* it will not insert the taxa that would decrease the *CIC* of the supertree, while *PhySIC* necessarily proposes a supertree that contains

all taxa present in a least one source tree.

However, the complexity of *PhySIC* is $O(kn^3 + n^4)$ while *PhySIC_IST* runs in $O(n^3(k + n^3))$ time, where k is the number of input trees of the forest $\mathcal{F}$ and $n = L(\mathcal{F})$. Moreover, *PhySIC* can run on a pre-computed triplet matrix $\mathcal{R}$ on a leaf set of size n in $O(n^4)$ time, while *PhySIC_IST* takes as input a forest so all triplets of $\mathcal{R}$ need to be transformed into trees on three leaves before running *PhySIC_IST*. This means that, in such a case, *PhySIC_IST* runs in $O(n^3(|\mathcal{R}| + n^3))$ time.

In this chapter, we have also introduced a statistical preprocessing of the source trees to detect and correct artifactual positions of taxa. This preprocessing of the source trees can be performed for any collection of source trees and hence benefits any veto supertree method. This approach has the advantage of separating the liberal resolution of conflicts among source trees from the assemblage of the supertree. This makes explicit the choices done to arbitrate between conflicting source trees, and allows the user to choose the extent with which the sources trees can be modified and to identify problematic source tree resolutions. In practice, STC+*PhySIC_IST* closes the gap between veto and liberal methods, as demonstrated in Section 4.4, where we presented an application of STC+*PhySIC_IST* to the biggest multigenic data set ever assembled for the Triticeae group. For this case study the STC+*PhySIC_IST* supertree, depicted in Figure 4.20 on page 127 is more resolved and in accord with the supermatrix tree (see Figure 4.19 on page 126) than the MRP supertree (see Figure 4.21 on page 128). This demonstrates that, in practice and not only in simulation studies, STC+*PhySIC_IST* can infer supertrees that are both resolved and reliable, combining the advantages of veto and vote supertree approaches. The combination of the supermatrix and supertree methodologies allowed us to reconstruct a robust phylogeny of Triticeae and to point out the evolution of several well supported clades. Furthermore, our detailed investigation of the incongruence between the gene trees and the supermatrix tree strongly suggests that the majority of incongruences among trees in Triticeae are due to incomplete lineage sorting of ancient polymorphisms rather than to gene flow.

Chapter 5

Methods to include multi-labeled phylogenies in a supertree framework

Contents

Recall that a *gene tree* is an evolutionary tree built by analyzing a gene family, *i.e.*, homologous molecular sequences appearing in the genome of different organisms. *Species trees*, *i.e.*, trees displaying the evolutionary relationships among studied species, are mainly estimated using gene trees. Unfortunately, as evoked in Chapter 2, most gene trees can significantly differ from the species tree for methodological or biological reasons, such as long branch attraction, lateral gene transfers, incomplete lineage sorting, gene duplications and losses [Cotton and Page, 2005]. For this reason, species trees are usually estimated from a large number of gene trees.

Inferring a species tree from gene trees is mostly done in a two-step approach. First, a micro-evolutionary model that takes into account events affecting individual sites is used to infer the gene trees. The species tree is then inferred on the basis of a macro-evolutionary model, *i.e.*, minimizing the number of transfers and/or duplication and loss events [*e.g.*, Chauve et al., 2008; Chauve and El-Mabrouk,

2009; Chen et al., 2000; Hallett et al., 2004; Hallett and Lagergren, 2000; Ma et al., 2000; Slowinski and Page, 1999; Vernot et al., 2008]. To produce more biologically meaningful trees, unified models have been proposed in which the micro and macro-evolutionary dimensions are entangled [Arvestad et al., 2003; Durand et al., 2006; Goodman et al., 1979]. However, it is difficult to determine how to incorporate in a single model events occurring on different spatial and temporal scales, as well as belonging to neutral and non-neutral processes [Durand et al., 2006]. Lately, a hybrid approach has been proposed, where a first draft of a species tree is inferred with a micro-evolutionary model, the most uncertain parts of which are then corrected according to a macro-evolutionary model [Durand et al., 2006].

In this chapter, we propose instead to take advantage of the very large number of gene trees present in recent phylogenomic projects to avoid entering into the detail of all possible macro-evolutionary scenarios (*e.g.*, is a parsimony approach always justified? Should only the most parsimonious scenario be retained?).

We propose to extract orthologous genes, the relevant part of the topological information contained in the gene trees to build a *species* tree, *i.e.*, the one resulting from *speciation* events as opposed to duplication events, and then apply a traditional supertree method letting the weight of evidence decide in favor of one candidate species tree [Baum and Ragan, 2004; Ranwez et al., 2007a; Scornavacca et al., 2008]. In fact, it is true that the large majority of gene trees include also xenologues and paralogues, but this doesn't mean that we have to discard the whole tree and the orthologues included within.

This approach is only possible when the number of gene trees is very large, and indeed this is now the case in projects such as the HOMOLENS database (`http://pbil.univ-lyon1.fr/databases/homolens.php`) and the HOGENOM database (`http://pbil.univ-lyon1.fr/databases/hogenom.php`) storing several thousands of gene trees. In the release 04 of these databases, respectively 51% and 71% of gene families have paralogous sequences, *i.e.*, sequences where duplications and losses have actually taken place. Currently, these gene families are discarded when inferring a supertree of the concerned species. This echoes, though less severely, the critic of Bapteste et al. [2008] who called "Trees of 1%" the species trees built by the first phylogenomic works that could rely only on single-labeled trees [*e.g.,* Brochier et al., 2005; Ciccarelli et al., 2006]. Moreover, note that as more complete genomes will be available, the percentage of gene families with paralogous sequences will only increase.

Gene trees are usually *multi-labeled*, *i.e.*, a single species can label more than one leaf, since duplication events almost always result in the presence of several copies of the genes in the species genomes (see Section 2.2.1). Since no supertree method exists to combine such trees, the task we therefore have to solve is to extract the largest amount of topological information on speciations from the multi-labeled gene trees. This speciation signal can then be turned into single-labeled trees to feed supertree methods.

This chapter presents a number of results in this direction. A part of this chapter appeared in the paper "From Gene Trees to Species Trees through a Supertree

Approach" [Scornavacca et al., 2009b]. An extended version has been submitted at Information and Computation (Elsevier ed.).

5.1 Basic concepts and preliminary results

In this chapter we focus on rooted **binary** MUlti-Labeled trees, or *MUL trees* for short, such as the one depicted in Figure 5.1(i). Dealing only with binary trees is not so restrictive as one can imagine, since, as evoked in Section 1.8, methods to reconstruct phylogenies usually produce binary trees. For instance in the HOGENOM database [Penel et al., 2009], among 46,535 gene trees containing taxa spanning more than two species, only 116 are not binary. More notations are needed to introduce formally multi-labeled trees.

5.1.1 Basic concepts

Like for single-labeled trees, $\mathcal{L}(M)$ denotes the set of leaf nodes of M and M_v denotes the subtree with node v as root. We denote by $L(v)$ the multiset of labels of M_v and by $L(M)$ the multiset $L(M_{root(M)})$. For a MUL tree M, the leaf-labelling of M $\alpha : \mathcal{L}(M) \longrightarrow L(M)$ is not a bijection, as for single-labeled trees (Section 3.1), but is a surjection *i.e.*, several leaves of M can share the same label.

Let M be a MUL tree and v a node of M. If v is a leaf node, we denote by $\boldsymbol{l_v}$ its label. If v is an internal node, throughout this chapter we denote by $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ the two sons of v and by $\mathbf{sons}(\boldsymbol{v})$ the set $\{v_1, v_2\}$.

Definition 5.1.1 *A node v of M is called an **observed duplication node** (**odn**) if the intersection of $L(v_1)$ and $L(v_2)$ is not empty.*

We use the expression "observed duplication nodes" since Definiton 5.1.1 does not characterize all duplication nodes. For instance, in Figure 5.1(ii) is depicted a tree with the unique duplication node indicated by a grey square. If asymmetric losses of gene copies for species b and c in M_{v_1} and a in M_{v_2} occurred (or these sequences are not available), v is not considered as a duplication node. We denote by $\mathcal{D}(M)$ the set of odns of a MUL tree M. Note that, for an odn v, $L(v)$ will always contain

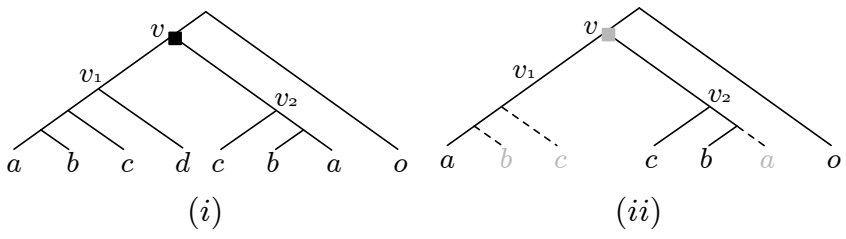


Figure 5.1: **An example of phylogenetic trees involving duplications** - (i) a MUL tree with $L(v) = \{a, c, b, d, c, b, a, o\}$. The unique odn indicated by a black square. (ii) a tree where the duplication is not detected (see text for more details).

some label more than once. A label $l \in L(M)$ is a *repeated label* for M if and only if the label l occurs more than once in $L(M)$. We say that f is a *repeated leaf* for M if and only if l_f is a repeated label.

5.1.2 Identifying observed duplication nodes in linear time

The easiest way to compute $\mathcal{D}(M)$ is by checking for each node v in M if the sets $L(v_1)$ and $L(v_2)$ intersect and adding v to $\mathcal{D}(M)$ in the case of a positive answer. The time complexity of this simple algorithm is $O(n^2)$, since it requires computing $O(n)$ intersections of two sets of $O(n)$ elements. But we can provide a faster algorithm that uses the least common ancestor (lca) to find $\mathcal{D}(M)$ in linear time (see Algorithm 7). This algorithm takes profit from efficient data structures to locate lcas and from the fact that a small number of lcas needs to be examined. To demonstrate the correctness of Algorithm 7, we need to establish some relationships between lcas and odns.

Lemma 5.1.2 *A node is an odn if and only if it is the lca of at least two leaves m and p with the same labels (i.e. $l_m = l_p$).*

Proof From definition 5.1.1, v is an odn if and only if $\mathrm{L}(v_1) \cap \mathrm{L}(v_2) \neq \emptyset$. In this case, there exist two leaf nodes m and p with $m \in M_{v_1}$ and $p \in M_{v_2}$ such that $l_m = l_p$. Thus v is a common ancestor of the two leaves m and p with the same label. Since m and p belong to two different subtrees having v as father ($m \in M_{v_1}$ and $p \in M_{v_2}$), v is indeed their lca in M.

Reciprocally, if v is the lca of two leaves m and p with the same label, this means that $\mathrm{L}(v_1) \cap \mathrm{L}(v_2) \neq \emptyset$, and v is an odn by definition 5.1.1. □

According to Lemma 5.1.2, we can compute $\mathcal{D}(M)$ by searching for the lcas of all pairs of leaves m and p with the same label. To determine the lca between multiple pairs of nodes, one can use an algorithm in Harel and Tarjan [1984] which preprocesses a data structure in $O(n)$ time, where n is the number of nodes and returns the lca of any two specific nodes from the data structure in O(1). We still have $\mathrm{O}(n^2)$ lcas to find, and even achieving constant time for each gives an $\mathrm{O}(n^2)$ total complexity. However, since there are only $O(n)$ internal nodes, many pairs of leaves share the same lca. A smarter approach is used in Algorithm 7: first of all, the subtrees of M are ordered from left to right in an arbitrary way. Then, each leaf, starting from the left of the tree and moving to the right, is tagged with its label followed by its occurrence number (see Figure 5.2). Then, for each repeated label e, the lca of any two successive occurrences e_i and e_{i+1} of e is inserted in $\mathcal{D}(M)$. This leads to a linear time complexity. Indeed, we have $O(n)$ of these couples since each leaf e_i of M is involved in at most the two pairs (e_{i-1}, e_i) and (e_i, e_{i+1}).

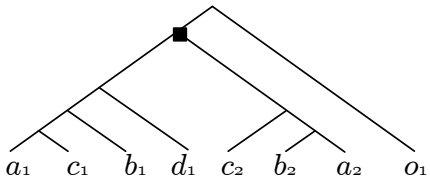


Figure 5.2: **An example of how leaf nodes are tagged in algorithm 7** - each leaf is tagged with its label followed by its occurrence number.

Algorithm 7: CompDuplicationNodes(M)

Data: A MUL tree M.
Result: A set of odns $\mathcal{D}(M)$.
1 Order M in an arbitrary way. In this order, tag each leaf with its label followed by its occurrence number.;
2 Compute the Harel & Tarjan data structure `//see text`;
3 $\mathcal{D}(M) \leftarrow \emptyset$;
4 **foreach** (*repeated label* e) **do**
5 ⌊ **foreach** ($\{e_j, e_{j+1}\}$) **do** $\mathcal{D}(M) \leftarrow \mathcal{D}(M) \cup lca(e_j, e_{j+1})$;
6 **return** $\mathcal{D}(M)$;

The correctness of Algorithm 7 is justified by Lemma 5.1.3 showing that this algorithm retrieves all odns of M.

Lemma 5.1.3 *Let M be a MUL tree. For each odn v, $\exists$ two successive occurrences of a label e denoted by e_i and e_{i+1} s.t. $v = lca(e_i, e_{i+1})$.*

Proof Given an odn v, there exists at least one label e present in both subtrees M_{v_1} and M_{v_2}. We denote by A the set of leaves a_i *s.t.* $a_i \in M_{v_1}$ and $l_{a_i} = e$ and denote by B the set of leaves b_j *s.t.* $b_j \in M_{v_2}$ and $l_{b_j} = e$. We denote by b_1 the rightmost element of B and by $a_{|A|}$ the leftmost element of A. We know that v is the lca of the two nodes $a_{|A|}$ and b_1. Additionally, due to the way we tagged M, we know that there is no other occurrence of the label e between $a_{|A|}$ and b_1. Indeed, if there was another leaf x labeled with e, it would be either in M_{v_1} (and then $x = a_{|A|}$) or in M_{v_2} (and then $x = b_1$). Then $a_{|A|}$ and b_1 are two successive occurrences of the same label and their lca is the node v.

□

5.2 Methods

5.2.1 Isomorphic subtrees

Definition 5.2.1 *Two rooted trees T_1 and T_2 are isomorphic (denoted by T_1=T_2) if and only if there exists an one-to-one mapping from the nodes of T_1 onto the nodes of T_2 preserving leaf labels and descendancy.*

For each odn v, we are interested in testing if the two subtrees M_{v_1} and M_{v_2} are isomorphic or not. In the positive, we can prune one of the two isomorphic subtrees and eliminate the odn v, as in the example of Figure 5.3. Indeed, when successively combining trees by a veto supertree method, all the topological information related to speciation events contained in M is present in M' (see Proposition 5.2.6 in Section 5.2.2). Indeed, the triplet $ab|c$ is still present in the tree M'. This is not the case when combining trees by a vote supertree method, since for the vote strategy, not only the presence of a triplet in the forest is important but also its frequency. In this case, M contains the triplet $ab|c$ twice while M' only once.

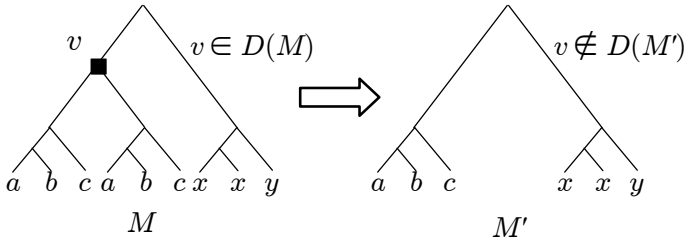


Figure 5.3: **Example of a MUL tree where the two child subtrees of the duplication node are isomorphic** - in this case we can keep only one of them.

For detecting isomorphism of MUL trees, we propose Algorithm 8, an extension to MUL trees of the CHECK-ISOMORPHISM-OR-FIND-CONFLICT algorithm [Berry and Nicolas, 2006]. Alternatively, we could have proposed an appropriate variant of the tree isomorphism algorithm detailed in Aho et al. [1974]. However, such an algorithm would likely have been less space efficient than the one we present here due to numerous string sorting steps using several queues and lists to ensure linear running time.

Algorithm 8 is based on nodes called **cherries** *i.e.*, internal nodes that have only two leaves as children. In the case of single-labeled trees we have the following lemma:

Lemma 5.2.2 *[Gusfield, 1991] Let T_1, T_2 be two isomorphic trees and let c_1 be a cherry in T_1. Then, there is a cherry $c_2 \in T_2$ s.t. $L(c_1) = L(c_2)$.*

In the case of MUL trees, we can have several copies of the same cherry. We call a **multiple cherry** the list of cherries on the same two labels. We note $|mc|$ the number of occurrences of the multiple cherry mc in a tree it belongs to.

Lemma 5.2.3 *Let M_1, M_2 be two isomorphic MUL trees and let mc_1 be a multiple cherry in M_1. Then, there is a multiple cherry $mc_2 \in M_2$ s.t. $L(mc_1) = L(mc_2)$ and $|mc_1| = |mc_2|$.*

The proof is straightforward from that of Lemma 5.2.2 in Gusfield [1991].

Outline of the algorithm

Algorithm 8 first finds all the multiple cherries for the MUL trees M_1 and M_2 that are stored in the list L_{mc} using a simple linked list. Additionally, a hashtable H is used where each $mc \in L_{mc}$ is a key. To each multiple cherry mc, H associates two linked lists, $O_1(mc)$ and $O_2(mc)$, storing pointers to nodes of M_1 and M_2 respectively that correspond to the occurrences of mc. The multiple cherries of a MUL tree are then examined in a bottom-up process. Given a multiple cherry mc in L_{mc} we check if the size of $O_1(mc)$ is the same as that of $O_2(mc)$. If this is not the case, we have found a multiple cherry for which we do not have the same number of occurrences in M_1 and M_2. In this instance, M_1 and M_2 are not isomorphic (Lemma 5.2.3) and the algorithm returns FALSE. Otherwise we turn all the cherries in $O_1(mc)$ and $O_2(mc)$ into leaves to which a same new label, different from all other labels in M_1 and M_2, is assigned. This modification of M_1 and M_2 can turn the fathers of some cherries in $O_1(mc)$ and $O_2(mc)$ into new cherries. Then L_{mc} is updated and the processing of cherries in M_1 is iterated until both MUL trees are reduced to a single leaf with the same label if M_1 and M_2 are isomorphic (*i.e.*, $L_{mc} = \emptyset$), or a FALSE statement is returned.

Algorithm 8: CheckIsomorphismMULTree(M_1,M_2)

Data: Two MUL tree M_1 and M_2.
Result: TRUE if M_1 and M_2 are isomorphic, FALSE otherwise.

```
Initialize the list L_mc of multiple cherries in M_1 and M_2;
Build the hashtable H where each mc ∈ L_mc is a key. To each mc, H
  associates two lists O_1(mc) and O_2(mc), respectively of the occurrences of
  mc in M_1 and M_2;
while (L_mc ≠ ∅) do
    mc ← removeFirst(L_mc);
    if (|O_1(mc)| =|O_2(mc)|) then
        Turn all cherries in O_1(mc) and O_2(mc) into leaves to which a same
        new label is assigned;
        add the new multiple cherries at the end of L_mc and update H;
    else return FALSE;
return TRUE;
```

Theorem 5.2.4 *Let M_1 and M_2 be two rooted MUL trees with $L(M_1) = L(M_2)$ of cardinality n. In time $O(n)$, Algorithm 8 returns TRUE if M_1 and M_2 are isomorphic, FALSE otherwise.*

Proof We show here that we can keep the linear time execution of the CHECK-ISOMORPHISM-OR-FIND-CONFLICT algorithm of Berry and Nicolas [2006], using supplementary data structures. A simple depth-first search of trees M_1 and M_2 initializes L_{mc} and H in $O(n)$ time. At each iteration of the algorithm, obtaining a

multiple cherry mc to process is done in $O(1)$ by removing the first element mc of L_{mc}. H then provides in $O(1)$ the lists $O_1(mc)$ and $O_2(mc)$ of its occurrences in the trees. Checking that these lists have the same number of elements is proportional to the number of nodes they contain, hence costs $O(n)$ amortized time, as each node is only once in such a list, and the list is processed once during the whole algorithm. Replacing all occurrences of mc by a new label is done in $O(n)$ amortized time, since each replacement is a local operation replacing three nodes by one in a tree and at most $O(n)$ such replacements can take place in a tree to reduce it down to a single node (which is the stop condition of the algorithm). Reducing a cherry can create a new occurrence $o_{mc'}$ of a cherry mc'. Checking in $O(1)$ time if mc' is a key in H allows to know whether occurrences of mc' have already been encountered or not in M_1 or M_2. In the positive, we simply add $o_{mc'}$ to the beginning of the list $O_1(mc)$ (if $o_{mc'} \in M_1$) or $O_2(mc)$ (if $o_{mc'} \in M_2$), requiring $O(1)$ time. In the negative, we add mc' to the beginning of L_{mc}, create a new entry in H for mc', and initialize the associated lists $O_1(mc)$ and $O_2(mc)$ so that one contains $o_{mc'}$ and the other is the empty list. Again, this requires only $O(1)$ time. Thus, performing all operations required by the algorithm globally costs $O(n)$ time. □

Applying Algorithm 8 to M_{v_1} and M_{v_2} for each odn v of a MUL tree M in a bottom-up approach requires $O(dn)$ time, where d is the number of odns in M.

5.2.2 Auto-coherency of a MUL tree

Algorithm 8 can be used to lower the number of duplication nodes in gene trees. Let M be a gene tree that still has duplication nodes after having removed isomorphic sibling subtrees in a bottom-up approach as described in Section 5.2.1. Thus, M contains several sequences for some taxa, *i.e.*, multiple copies of some labels. We can then wonder whether these copies display similar relationships with their respective neighboring labels.

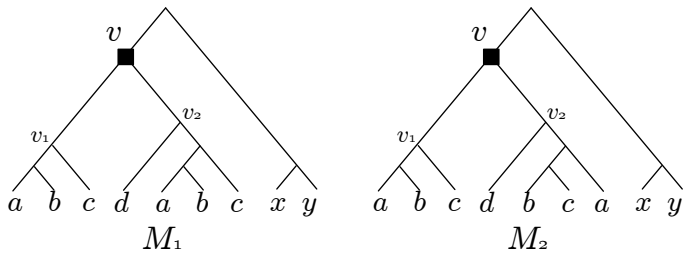


Figure 5.4: **Example of auto-coherent and non auto-coherent MUL trees** - the evolutionary signal of M_1 is coherent with respect to Definition 5.2.7 while the evolutionary signal of M_2 is not.

For instance, the subtrees $M_2(v_1)$ and $M_2(v_2)$ in Figure 5.4 contain respectively the triplet $ab|c$ and the triplet $bc|a$ so M_2 hosts contradictory triplets.

In the case of a MUL tree M with a coherent evolutionary signal (for instance the MUL tree M_1 depicted in Figure 5.4), we can summarize the evolutionary information of M into a single-labeled tree T. We introduce below some notations to formalize this idea.

Definition 5.2.5 *Let M be a MUL tree. We define by $\mathcal{R}_{wd}(M)$ ($\mathcal{R}(M)$ **w**ithout **d**uplications) the set of triplets $ab|c$* s.t. *there exist three leaf nodes $x, y, z \in M$ with $l_x = a$, $l_y = b$, $l_z = c$ such that both*

(i) *$lca(x, y) \neq (lca(x, z) = lca(y, z))$,*

(ii) *$lca(\{x, y, z\}) \notin \mathcal{D}(M)$ and $lca(x, y) \notin \mathcal{D}(M)$*

Part (i) of the condition ensures that $ab|c$ is displayed by M *i.e.*, $M|_{\{x,y,z\}} = ab|c$, while Part (ii) ensures that none of the two internal nodes of $M|_{\{x,y,z\}}$ is an odn of M. For example, for the MUL tree in Figure 5.1(i), $\mathcal{R}_{wd}(M)$={ab|c,ac|d,ab|d,bc|d,ac|o,ab|o,ad|o,bc|o,cd|o,bd|o}. Hence, not all the triplets of $\mathcal{R}(M)$ are kept. We introduce this definition because, once a duplication event occurred in a gene's history, the two copies of the gene evolved independently. The history of each copy is influenced by the species' history but, considering one of them simultaneously with the close relatives of another copy, *i.e.*, with paralogous sequences, may produce information unrelated to the speciation events (see Section 2.2.1). Therefore, to avoid mixing the history of different copies of a gene, it is better to discard the triplets that address paralogous sequences. This is exactly what $\mathcal{R}_{wd}(M)$ achieves.

$\mathcal{R}_{wd}(M)$ has size $O(n^3)$ and can be computed in $O(n^3)$ time, where n is the number of leaf nodes of M. Indeed, once the Harel & Tarjan data structure is computed in $O(n)$ time [see Bender and Farach-Colton, 2000; Harel and Tarjan, 1984], checking if three leaf nodes x, y, z of M satisfy Definition 5.2.5 can be done in $O(1)$ time, thus in $O(n^3)$ for all triplets of leaves in M.

Proposition 5.2.6 *Let M be a MUL tree and M' the MUL tree obtained by applying algorithm 8 to eliminate isomorphic sibling subtrees. Then $\mathcal{R}_{wd}(M) = \mathcal{R}_{wd}(M')$.*

Definition 5.2.7 *A MUL tree M is said to be **auto-coherent** if the triplet set $\mathcal{R}_{wd}(M)$ is compatible,* i.e., *if there exists a single-labeled tree T such that $\mathcal{R}_{wd}(M) \subseteq \mathcal{R}(T)$.*

In the case of an auto-coherent MUL tree, we know that there exists at least one tree T containing all the speciation information contained in $\mathcal{R}_{wd}(M)$, *i.e.*, the information of M that is considered to express speciation information. To check if a MUL tree is auto-coherent, we can resort to the AncestralBuild algorithm of Berry and Semple [2006] (see page 59). For a set of triplets R, this algorithm indicates in $O(|R| \cdot \log^2(|L(R)|))$ time whether R is compatible, where $L(R)$ is the set of leaf labels of the elements of R. Moreover, in case of a positive answer it returns a tree T *s.t.* $R \subseteq \mathcal{R}(T)$.

Steel [1992] proved that any binary single-labeled rooted tree T can be encoded using a triplet set $\mathcal{R}^l(T)$ whose size is the number of inner nodes of T. In this section we show that it is possible to check the auto-coherency of a binary MUL tree M by using as representation of $\mathcal{R}_{wd}(M)$ a triplet set $\mathcal{R}^l_{wd}(M)$ whose size is at most equal to the number of speciation nodes of M. To univocally define the set $\mathcal{R}^l_{wd}(M)$, let $<$ be a total order on the leaf set $L(M)$. For each node v of M, we denote by $\mathbf{sm(v)}$ the smallest element of $L(M_v)$ according to $<$ and by $\mathbf{anc(v)}$ the set of nodes belonging to the path from v to the root of M. Note that the root of M belongs to $anc(v)$ while v does not. Let $\mathbf{lsa(v)}$ be the least speciation ancestor of v, *i.e.*, the speciation node in $anc(v)$ closest to v, and let v' be the son of $lsa(v)$ such that $v \notin M_{v'}$. Note that, if the father of v is not in $\mathcal{D}(M)$, it coincides with $lsa(v)$ while $\boldsymbol{v'}$ is the sibling node of v.

Definition 5.2.8 *Let M be a binary MUL tree and $<$ a total order on $L(M)$. We define by $\boldsymbol{\mathcal{R}^l_{wd}(M)}$ the set of triplets $ab|c$ such that $ab|c \in \mathcal{R}_{wd}(M)$ and there exists a speciation node v in M such that $sm(v_1) = a$, $sm(v_2) = b$ and $sm(v') = c$.*

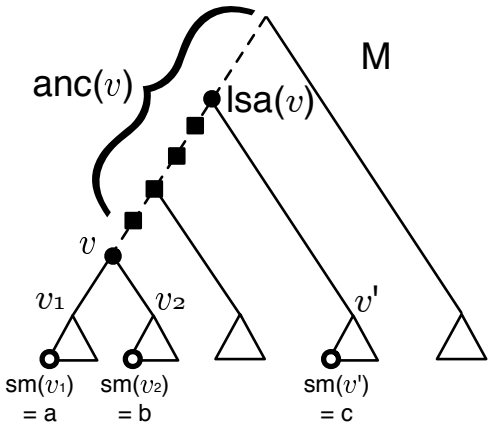


Figure 5.5: **Example of how to compute $\boldsymbol{\mathcal{R}^l_{wd}(M)}$** - the only triplet of $\mathcal{R}^l_{wd}(M)$ associated to the speciation node v is $ab|c$ (see definition 5.2.8), while the triplet set associated to v in $\mathcal{R}_{wd}(M)$ is composed by the triplets $l_x l_y | l_z$ of $\mathcal{R}(M)$, where $x \in \mathcal{L}(M_{v_1})$, $y \in \mathcal{L}(M_{v_2})$ and $z \in \mathcal{L}(M)$ such that $lca(x,y,z) \notin \mathcal{D}(M)$ and $lca(x,y) \neq (lca(x,z) = lca(y,z))$

Note that, for each speciation node v, the set $\mathcal{R}^l_{wd}(M)$ contains at most one triplet $l_x l_y | z$, with $v = lca(x,y)$ while $\mathcal{R}_{wd}(M)$ tipically contains many more such triplets (see Figure 5.5).

Once the set of duplication nodes $\mathcal{D}(M)$ is calculated, Algorithm 9 computes $\mathcal{R}^l_{wd}(M)$ in linear time (see Theorem 5.2.14). We now need to show that checking the auto-coherency of $\mathcal{R}_{wd}(M)$ and $\mathcal{R}^l_{wd}(M)$ is equivalent. To do that, we need to

Algorithm 9: linearRepresentation$(M, \mathcal{D}(M), v)$

Data: A binary MUL tree M, the set of duplication nodes $\mathcal{D}(M)$ of M, a node v in M.

Result: A set of triplets $\mathcal{R}^l$ that is the linear representation of the speciation triplet information of M.

$\mathcal{R}^l \leftarrow \emptyset$;
if (v *is not a leaf* ***and*** v *is not the root node*) **then**
  $f \leftarrow$ the father of v;
  if ($f \notin \mathcal{D}(M)$) **then**
    if ($f_1 = v$) **then** $v' \leftarrow f_2$;
    else $v' \leftarrow f_1$;
  else
    $v' \leftarrow f'$;
$\mathcal{R}^l \leftarrow \mathcal{R}^l \cup$ linearRepresentation$(M, \mathcal{D}(M), v_1)$;
$\mathcal{R}^l \leftarrow \mathcal{R}^l \cup$ linearRepresentation$(M, \mathcal{D}(M), v_2)$;
if ($v \notin \mathcal{D}(M)$) *and* ($v' \neq \emptyset$) **then**
  $\mathcal{R}^l \leftarrow sm(v_1)sm(v_2)|sm(v')$;
return $\mathcal{R}^l$;

introduce more notations. Given a node v of a MUL tree M, we define the height of v, denoted by $\mathbf{h(}\boldsymbol{v}\mathbf{)}$, as the length of the longest path between v and its descendants. More formally, the height of a leaf is fixed to zero and that of an internal node v is max($h(v_1)$,$h(v_2)$)+1. Recall that $\mathcal{G}(\mathcal{R}, L)$ is the Aho graph built from a triplet set $\mathcal{R}$ on a leaf set L (see Section 3.3.1.1). The set of vertices of this graph is L and there is an edge in $\mathcal{G}(\mathcal{R}, L)$ connecting two vertices a and b if and only if there exists at least one triplet $ab|c$ in $\mathcal{R}$. The proof that the auto-coherency of $\mathcal{R}_{wd}(M)$ can be tested by checking that of $\mathcal{R}^l_{wd}(M)$ relies on the following Lemma.

Lemma 5.2.9 *Let M be a binary MUL tree and v a node of M. If $anc(v)$ contains at least one speciation node, then $\mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_v))$ is connected.*

Proof We prove the lemma by induction on the height of the node v. Note that, from the statement of the lemma, we need to consider only those nodes having at least one speciation node as ancestor.

Let us start showing that Lemma 5.2.9 is valid for all nodes with height 0. In this case $L(M_v)$ contains a single label, hence $\mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_v))$ contains only one vertex *i.e.*, is trivially connected.

Now suppose that Lemma 5.2.9 is valid for all nodes v such that $h(v) < \bar{h}$. We want to prove that this implies that the lemma is true for all nodes v: $h(v) \leqslant \bar{h}$. Let v be a node for which $anc(v)$ contains at least one speciation node and $h(v) = \bar{h}$. Since $h(v_1) = h(v) - 1$ and $lsa(v) \in anc(v_1)$ we know that $\mathcal{G}_1 = \mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_{v_1}))$ is made of a single connected component C_1 and the same

holds for $\mathcal{G}_2 = \mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_{v_2}))$, denoting by C_2 this connected component. It remains to prove that there exists an edge connecting the two connected components C_1 and C_2. Either v is a speciation node or a duplication node. If v is a speciation node, then from the definition of $\mathcal{R}^l_{wd}(M)$ there exists a triplet $t \in \mathcal{R}^l_{wd}(M)$ such that $t = sm(v_1)sm(v_2)|sm(v')$ and thus t induces an edge between C_1 and C_2. If v is an observed duplication node, there exists at least a label d such that $d \in L(M_{v_1}) \cap L(M_{v_2})$ and this label is represented by a single vertex present in both C_1 and C_2 in $\mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_v))$ that contains all vertices and edges of $\mathcal{G}_1$ and $\mathcal{G}_2$. Thus, $\mathcal{G}(\mathcal{R}^l_{wd}(M), L(M_v))$ is connected in both cases. □

Lemma 5.2.9 will be useful while proving Lemma 5.2.11. Let us introduce the notion of closure of a compatible triplet set. Given a compatible triplet set $\mathcal{R}$, we say that a triplet $ab|c$ is in the *closure* of $\mathcal{R}$, denoted by $\boldsymbol{cl(\mathcal{R})}$, if and only if $ab|c \in \mathcal{R}(T), \forall T : \mathcal{R} \subseteq \mathcal{R}(T)$. This is equivalent to requiring that both sets $\{\mathcal{R} \cup \{ac|b\}\}$ and $\{\mathcal{R} \cup \{bc|a\}\}$ are incompatible [Grunewald et al., 2007]. We introduce a result on the closure of a triplet set that will be useful later on.

Proposition 5.2.10 *If $\mathcal{R}$ is a compatible triplet set, then $cl(\mathcal{R})$ is compatible.*

Proof From the definition of compatibility, a triplet set $\mathcal{R}$ is compatible if there exists a tree T such that $\mathcal{R} \subseteq \mathcal{R}(T)$. From proposition 4(6)[1] of Bryant and Steel [1995] we know that if such a tree exists, this tree has also the property $cl(\mathcal{R}) \subseteq \mathcal{R}(T)$. It follows that $cl(\mathcal{R})$ is compatible. □

Using this result, we can now prove the following Lemma.

Lemma 5.2.11 *Let M be a binary MUL tree. If the triplet set $\mathcal{R}^l_{wd}(M)$ is compatible, then $\mathcal{R}_{wd}(M) \subseteq cl(\mathcal{R}^l_{wd}(M))$.*

Proof We prove this statement for all subtrees M_v of M by induction on the height of the node v in M. As $M = M_{root(M)}$ this shows the statement.

If $h(v) = 0$ then $\mathcal{R}_{wd}(M_v) = cl(\mathcal{R}^l_{wd}(M_v)) = \emptyset$. Now suppose that $\mathcal{R}_{wd}(M_v) \subseteq cl(\mathcal{R}^l_{wd}(M_v))$ for all nodes v such that $h(v) < \bar{h}$ and let v be a node such that $h(v) = \bar{h} > 0$.

i) If v is a duplication node, then, if $|\mathcal{L}(M_{v_1}) > 1|$, for $x, y \in \mathcal{L}(M_{v_1})$ with $x \neq y$ and $z \in \mathcal{L}(M_{v_2})$ we have that $lca(x, y, z) \in \mathcal{D}(M)$. The same holds for the symmetric case *i.e.*, $|\mathcal{L}(M_{v_2}) > 1|$. This implies that $\mathcal{R}_{wd}(M_v) = \mathcal{R}_{wd}(M_{v_1}) \cup \mathcal{R}_{wd}(M_{v_2})$ and $\mathcal{R}^l_{wd}(M_v) = \mathcal{R}^l_{wd}(M_{v_1}) \cup \mathcal{R}^l_{wd}(M_{v_2})$. It follows that $\mathcal{R}_{wd}(M_v) \subseteq cl(\mathcal{R}^l_{wd}(M_{v_1})) \cup cl(\mathcal{R}^l_{wd}(M_{v_2})) \subseteq cl(\mathcal{R}^l_{wd}(M_{v_1}) \cup \mathcal{R}^l_{wd}(M_{v_2})) = cl(\mathcal{R}^l_{wd}(M_v))$. Note that, if $|\mathcal{L}(M_{v_1})| = 1$ and $|\mathcal{L}(M_{v_2})| = 1$, then $\mathcal{R}_{wd}(M_v) = \mathcal{R}^l_{wd}(M_v) = \emptyset$ and the lemma still holds.

[1] Proposition 4 of Bryant and Steel [1995] is defined for quartets but it remains valid for rooted triplets (see page 441 of this reference).

ii) If v is a speciation node, then by induction all triplets $l_x l_y | l_z \in \mathcal{R}_{wd}(M_v)$ with $x, y, z \in \mathcal{L}(M_{v_1})$ or $x, y, z \in \mathcal{L}(M_{v_2})$ are in $cl(\mathcal{R}^l_{wd}(M_v))$. Let t be a triplet $l_x l_y | l_z$ of $\mathcal{R}_{wd}(M_v)$ with $x, y \in \mathcal{L}(M_{v_1})$ and $z \in \mathcal{L}(M_{v_2})$. We prove that t is in $cl(\mathcal{R}^l_{wd}(M_v))$ by proving that $(\mathcal{R}^l_{wd}(M_v) \cup l_x l_z | l_y)$ and $(\mathcal{R}^l_{wd}(M_v) \cup l_y l_z | l_x)$ are both incompatible. From Lemma 5.2.9 we know that $\mathcal{G}(\mathcal{R}^l_{wd}(M_v), L(M_{v_1}))$ and $\mathcal{G}(\mathcal{R}^l_{wd}(M_v), L(M_{v_2}))$ are two connected components C_1 and C_2, since v is a speciation node above v_1 (resp v_2). As $L(M_v) = L(M_{v_1}) \cup L(M_{v_2})$, the graph $\mathcal{G}(\mathcal{R}^l_{wd}(M_v), L(M_v))$ has at most two connected components. Since $\mathcal{R}^l_{wd}(M)$ is compatible, $\mathcal{R}^l_{wd}(M_v) \subseteq \mathcal{R}^l_{wd}(M)$ is also compatible then $\mathcal{G}(\mathcal{R}^l_{wd}(M_v), L(M_v))$ is composed by exactly two connected components [Bryant and Steel, 1995, Theorem 2] *i.e.*, C_1 and C_2. Since $l_x l_y | l_z \in \mathcal{R}_{wd}(M)$ then $l_x \neq l_y \neq l_z$: this means that $l_x, l_y \in C_1$ and $l_x, l_y \notin C_2$ while $l_z \in C_2$ and $l_z \notin C_1$. Then both triplets $l_x l_z | l_y$ and $l_y l_z | l_x$ would connect the two connected components. This implies that $(\mathcal{R}^l_{wd}(M_v) \cup l_x l_z | l_y)$ and $(\mathcal{R}^l_{wd}(M_v) \cup l_y l_z | l_x)$ are both incompatible and then t is in $cl(\mathcal{R}^l_{wd}(M_v))$. The same result holds for the symmetric case $x, y \in \mathcal{L}(M_{v_2})$ and $z \in \mathcal{L}(M_{v_1})$. Note that this lemma works also if $|\mathcal{L}(M_{v_1})| = 1$ and/or $|\mathcal{L}(M_{v_2})| = 1$.
This concludes the proof that $\mathcal{R}_{wd}(M) \subseteq cl(\mathcal{R}^l_{wd}(M))$. □

Lemma 5.2.12 *Let M be a binary MUL tree. If the triplet set $\mathcal{R}^l_{wd}(M)$ is compatible, then $cl(\mathcal{R}^l_{wd}(M)) = cl(\mathcal{R}_{wd}(M))$.*

Proof If $\mathcal{R}^l_{wd}(M)$ is compatible then it follows from Proposition 5.2.10 that $cl(\mathcal{R}^l_{wd}(M))$ is compatible. Lemma 5.2.11 thus implies that $\mathcal{R}_{wd}(M)$, as subset of the compatible set $cl(\mathcal{R}^l_{wd}(M))$, is also compatible. In such a case the closure of $\mathcal{R}_{wd}(M)$ is well defined. The definition of the closure operation implies that, if $\mathcal{R}_1 \subseteq \mathcal{R}_2$ are two compatible triplet sets then $cl(\mathcal{R}_1) \subseteq cl(\mathcal{R}_2)$ [Grunewald et al., 2007, page 4]. From this observation and Lemma 5.2.11, it follows that $cl(\mathcal{R}_{wd}(M)) \subseteq cl(cl(\mathcal{R}^l_{wd}(M)))$. Since $cl(cl(\mathcal{R}^l_{wd}(M))) = cl(\mathcal{R}^l_{wd}(M))$ [Grunewald et al., 2007, page 4], we obtain that $cl(\mathcal{R}_{wd}(M)) \subseteq cl(\mathcal{R}^l_{wd}(M))$.

By construction $\mathcal{R}_{wd}(M) \supseteq \mathcal{R}^l_{wd}(M)$. This implies that $cl(\mathcal{R}_{wd}(M)) \supseteq cl(\mathcal{R}^l_{wd}(M))$.
This concludes the proof. □

Corollary 5.2.13 *The triplet set $\mathcal{R}^l_{wd}(M)$ is compatible if and only if the triplet set $\mathcal{R}_{wd}(M)$ is compatible.*

Proof The fact that $\mathcal{R}_{wd}(M) \supseteq \mathcal{R}^l_{wd}(M)$ implies that if $\mathcal{R}_{wd}(M)$ is compatible then $\mathcal{R}^l_{wd}(M)$ is also compatible while if $\mathcal{R}^l_{wd}(M)$ is not then $\mathcal{R}_{wd}(M)$ is not compatible.

While proving Lemma 5.2.12 we proved that if $\mathcal{R}^l_{wd}(M)$ is compatible then $\mathcal{R}_{wd}(M)$ is compatible. This implies that if $\mathcal{R}_{wd}(M)$ is not compatible then $\mathcal{R}^l_{wd}(M)$ is also not compatible, otherwise we would have $\mathcal{R}^l_{wd}(M)$ compatible and $\mathcal{R}_{wd}(M)$ incompatible and this would contradict Lemma 5.2.12. This proves the corollary. □

Theorem 5.2.14 *Checking the auto-coherency of a binary MUL tree M can be done in $O(n \cdot \log^2 n)$ time.*

Proof From Lemma 5.2.12 and Corollary 5.2.13 it follows that checking the auto-coherency of a binary MUL tree M can be done using the triplet set $\mathcal{R}^l_{wd}(M)$. This set can be computed in linear time by Algorithm 9. Given the set $\mathcal{D}(M)$ and having previously calculated $sm(v)$ for each node v, Algorithm 9 computes v' for each node v in M in a top-down approach. If $v \notin \mathcal{D}(M)$ and $v' \neq \emptyset$, Algorithm 9 inserts in $\mathcal{R}_{wd}(M)$ the triplet $sm(v_1)sm(v_2)|sm(v')$: this is exactly the definition of $\mathcal{R}^l_{wd}(M)$. This proves that Algorithm 9 computes $\mathcal{R}^l_{wd}(M)$. Note that $\mathcal{R}^l_{wd}(M)$ has an $O(n)$ size, since we can have at most one triplet for each internal node of M.

Let us demonstrate that Algorithm 9 computes $\mathcal{R}^l_{wd}(M)$ in linear time. The value of $sm(v)$ for each node can be computed in a single bottom-up search requiring linear time. The set of duplication nodes $\mathcal{D}(M)$ can be also computed in linear time (see Section 5.1.2). Algorithm LINEARREPRESENTATION$(M, \mathcal{D}(M), root(M), \mathcal{R}^l)$ consists in a postorder search walk on the MUL tree M and takes a linear time. Since AncentralBuild checks the compatibility of a triplet set $\mathcal{R}$ on a label set of size n in $O(|\mathcal{R}| \cdot \log^2 n)$ time, this concludes the proof. □

5.2.3 Computing a largest duplication-free subtree of a MUL tree

If a MUL tree is not auto-coherent, identifying duplication nodes still allows for the discrimination of leaves representing orthologous and paralogous sequences. Since only orthologous sequence history reflects the species history, a natural question is to determine the most informative orthologous sequence set for a given gene. As long as the gene tree contains odns, it will also contain leaves representing paralogous sequences. Yet, if for each node $v \in \mathcal{D}(M)$ of M we choose to keep either M_{v_1} or M_{v_2}, we obtain a pruned single-labeled tree containing only *apparent*[2] orthologous sequences (observed paralogous have been removed by pruning subtrees of odns). Note that the so obtained single-labeled tree is auto-coherent by definition.

Definition 5.2.15 *Let M be a MUL tree. We say that T is obtained by (duplication)* pruning *M if and only if T is obtained from M by choosing for each odn v either M_{v_1} or M_{v_2} and restricting M to the conserved subtrees. We denote this operation by the symbol $\lesssim$.*

One can wonder, for a non auto-coherent MUL tree M, what is the most informative single-labeled tree T *s.t.* $T \lesssim M$. We define this problem as the $MIPT$ (*Most Informative Pruned Tree*) problem.
To evaluate the informativeness of a tree we can use either its number of triplets of T [see Page, 2002; Ranwez et al., 2007a; Semple and Steel, 2000] that, for binary trees, only depends on the number of leaves, or the CIC criterion [see Scornavacca et al., 2008; Thorley et al., 1998, introduced in Section 4.3.1]. Recall that the CIC of a not fully resolved and incomplete[3] tree T with $|L(T)|$ leaves among the n possible

[2] Recall that, as evoked in Section 5.1.1 on page 139, we may fail to detect some duplication nodes.

[3] A tree is called incomplete when it misses some taxa.

Algorithm 10: PRUNING(v,M,$\mathcal{D}(M)$)

Data: A node v, a MUL tree M, and a set of odns $\mathcal{D}(M)$.
Result: The most informative MUL tree M' *s.t.* $M'_v \lesssim M_v$ and M'_v is single-labeled.

```
foreach (m ∈ sons(v)) do PRUNING(m,M,D(M));
if (v ∈ D(M)) then
    if (|L(v1)| > |L(v2)|) then
        prune M_{v2} from M and merge nodes v1 and v2;
    else
        prune M_{v1} from M and merge nodes v1 and v2;
return M;
```

is a function of both the number $n_R(T, n)$ of fully resolved trees T' on $L(T)$ such that $\mathcal{R}(T) \subseteq \mathcal{R}(T')$ and the number $n_R(n)$ of fully resolved trees on n leaves:

$$CIC(T, n) = -\log\Big(n_R(T, n)/n_R(n)\Big)$$

In the case of binary trees, $n_R(T, n)$ depends only on the number of source taxa missing in T since T does not contain multifurcations. Thus, dealing with binary MUL trees T, finding the MIPT (*i.e.*, maximizing the number of triplets or minimizing the CIC value) consists in finding the subtree of T with the largest number of leaves.

A natural approach for the MIPT problem on binary MUL trees is an algorithm that, after having computed $\mathcal{D}(M)$, uses a bottom-up starting from root(M), to keep the most informative subtree between M_{v_1} and M_{v_2}, for each odn v (see Algorithm 10).

Theorem 5.2.16 *Let M be a MUL tree on a set of n leaves. In time $O(n)$, Algorithm* PRUNING*(M,root(M),$\mathcal{D}(M)$) returns the most informative single-labeled tree T* s.t. *$T \lesssim M$.*

Proof First of all, it's obvious that PRUNING(M,root(M),$\mathcal{D}(M)$) returns a tree. Indeed, if for each odn v only one node between v_1 and v_2 is kept, at the end of the bottom-up procedure one copy of each duplicated leaf is present in the modified M. Now, we have to show that the resulting tree is the most informative tree *s.t.* $T \lesssim M$, *i.e.*, the tree with as many leaves as possible. For an odn v that is the ancestor of other duplication nodes, the choices made for v_1 do not influence the choices for v_2 since for each duplication node we can keep only one of the two subtrees, the most crowded one. Thus we can search for the best set of choices left/right for v_1 and v_2 independently and then choose the most crowded pruned subtree between v_1 and v_2. Iterating recursively this reasoning, we demonstrate that the tree obtained by Algorithm 10 is the most informative tree T *s.t.* $T \lesssim M$. The computation of the set $\mathcal{D}(M)$ of odns takes linear time. The subroutine PRUNING($M, root$(M), $\mathcal{D}(M)$) requires a tree search, thus the time complexity of Algorithm 10 is $O(n)$. □

5.2.4 Compatibility of single-labeled subtrees obtained from MUL trees

We can also ask if it is possible, given a collection $\mathcal{M}$ of MUL trees, to discriminate leaves representing orthologous and paralogous sequences in a gene tree using the information contained in the other gene trees to obtain a compatible forest $\mathcal{F}$, *i.e.*, a forest for which there exists a tree T *s.t.* $(\cup_{T_i \in \mathcal{F}} \mathcal{R}(T_i)) \subseteq \mathcal{R}(T)$. We denote this problem by Existence of a Pruned and Compatible Forest (EPCF).
Unfortunately, the EPCF problem is NP-complete.

EPCF	**Instance** : A set of leaves X and a collection $\mathcal{M}=\{M_1, \cdots M_k\}$ of MUL trees on X. **Question** : $\exists$ a set S of choices left/right, $S : \mathcal{M} \to \mathcal{F}$, with $\mathcal{F}=\{T_1, \cdots T_k\}$ *s.t.* $T_i \lesssim M_i$ and $\mathcal{F}$ is compatible?

Theorem 5.2.17 *The EPCF problem is NP-complete.*

Proof We start by proving that EPCF is in NP, *i.e.*, checking if a set S of choices left/right is a solution for the instance $\mathcal{I} = (\mathcal{M}, X)$ can be done in polynomial time. First of all, for each MUL tree $M_j \in \mathcal{M}$, we choose for each node $v \in \mathcal{D}(\mathcal{M})$ to keep either M_{v_1} or M_{v_2} (following the left/right choices of S) obtaining a forest $\mathcal{F}$ of single-labeled trees. Second, we check the compatibility of $\mathcal{F}$ with the Aho graph (see Section 3.3.1.1) and this can be done in polynomial time.

Given that EPCF is in NP, we use a reduction of 3-SAT to EPCF to demonstrate that the latter is NP-complete.

3-SAT	**Instance** : A boolean expression $\mathcal{C}=(C_1 \wedge C_2 \wedge \cdots \wedge C_n)$ on a finite set $L=\{l_1, l_2, \cdots, l_m\}$ of variables with $C_j=(a \vee b \vee c)$ where $\{a, b, c\} \in \{l_1, l_2, \cdots, l_m, \overline{l_1}, \overline{l_2} \cdots, \overline{l_m}\}$ **Question** : $\exists$ a truth assignment for L such that $\mathcal{C}$=TRUE ?

We need to show that every instance of 3-SAT can be transformed into an instance of EPCF; then we will show that given an instance $\mathcal{I} = (\mathcal{C}, L)$ of 3-SAT, $\mathcal{I}$ is a positive instance, *i.e.*, an instance for which a solution exists, if and only if the corresponding instance for EPCF is positive.

Given an instance $\mathcal{I} = (\mathcal{C}, L)$ of 3-SAT, we build an instance $\mathcal{I}' = (\mathcal{M}, X)$ of EPCF associating to each l_i in L the binary tree $T(l_i) = (((x_i, y_i), z_i), d)$ and to $\overline{l_i}$ the binary tree $T(\overline{l_i}) = (((z_i, y_i), x_i), d)$ (see Figure 5.6 for an example).
The set of subtrees $\left\{T(a) \mid a \in \{l_1, l_2, \cdots, l_m, \overline{l_1}, \overline{l_2}, \cdots, \overline{l_m}\}\right\}$ is denoted by $\mathcal{F}_L$. Then, for each clause $C_j = (a \vee b \vee c)$ in C, a binary MUL tree M_j is built, formed by three subtrees $((T(a), T(b)), T(c))$. Note that M_j has exactly two duplication nodes due to the presence of d in $T(a)$, $T(b)$ and $T(c)$, so that any left/right choice of M_j will reduce it to either $T(a)$, $T(b)$ or $T(c)$. Figure 5.7 displays an example of

a MUL tree built from a clause. In this way we obtain a forest of MUL trees $\mathcal{M}$ on the leaf set $X = \left\{ \{ \bigcup_{i=1}^{m} \{x_i, y_i, z_i\} \} \cup \{d\} \right\}$, *i.e.*, an instance of the EPCF problem. Clearly $\mathcal{M}$ can be built in polynomial time.

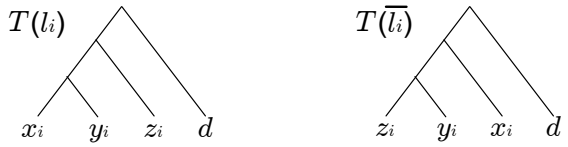


Figure 5.6: **Binary trees on four leaves associated to l_i and to $\overline{l_i}$** - where l_i is a literal of a 3-SAT instance.

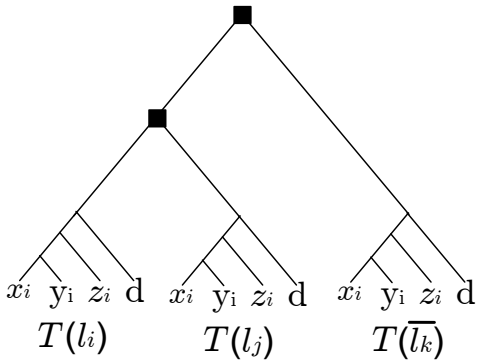


Figure 5.7: **MUL tree built from the clause** $\{l_i \vee l_j \vee \overline{l_k}\}$ - odns are indicated by black squares.

We now need to show that a positive instance of 3-SAT gives a positive instance of EPCF through the previous transformation. Having a positive instance for 3-SAT implies that for each $C_j \in \mathcal{C}$ with $C_j = (a \vee b \vee c)$, at least one of the three literals is TRUE. Without loss of generality, let us suppose that a is TRUE. Then in the MUL tree M_j corresponding to C_j we set the left/right choices so that only the subtree $T(a)$ is kept. Doing this for each $C_j \in \mathcal{C}$, we then obtain a forest $\mathcal{F}$ that is a subset of $\mathcal{F}_L$. We need to prove that $\mathcal{F}$ is compatible. Let $\tilde{T}(a)$ denote the tree $T(a)|(L(T(a)) - \{d\})$ and $\tilde{\mathcal{F}}$ the forest composed by all trees $\{\tilde{T}(a)|T(a) \in \mathcal{F}\}$ where each tree occurs only once, even if the same literal was chosen in different clauses. Then, we can build a tree $T_s = (\tilde{T}_1, \tilde{T}_2, \cdots, \tilde{T}_{|\tilde{\mathcal{F}}|}, d)$ multifurcating at the root[4]. Note that each label is present only once in this tree. Indeed l_i cannot have the value TRUE and FALSE at the same time and either $T(l_i)$ or $T(\overline{l_i})$ are in $\tilde{\mathcal{F}}$. The tree T_s is therefore a single-labeled tree. Moreover, by construction, $T_s|(L(T(a))$ is identical to $T(a)$, for all $T(a)$ in $\mathcal{F}$ ensuring that $\bigcup_{T_i \in \mathcal{F}} \mathcal{R}(T_i) \subseteq \mathcal{R}(T_s)$. Thus $\mathcal{F}$ is compatible.

Now, the only thing left to prove is that if the built instance of EPCF leads to a compatible forest $\mathcal{F}$, then the boolean expression of 3-SAT can be satisfied.

[4]This tree is express in Newick format, see Section 3.1 on page 40.

The repetition of the taxon d in each subtree makes the two nodes connecting the subtrees in each M_j be odn. Thus a left/right choice set S reduces each M_j in $\mathcal{M}$ into a tree $T(a) \in \mathcal{F}_L$, providing the forest $\mathcal{F}$. Setting the value of a to TRUE ensures that the clause C_j corresponding to M_j is TRUE. This can be done simultaneously for all clauses $\in \mathcal{C}$ since the forest compatibility implies that there is no contradiction among the trees in $\mathcal{F}$. Then, either $T(l_i)$ or $T(\overline{l_i})$ is in $\mathcal{F}$. This ensures us that either l_i or $\overline{l_i}$ is assigned to TRUE, but not both. □

Note that the problem to find the most informative forest $\mathcal{F} = \{T_1, \cdots T_k\}$ *s.t.* $T_i \lesssim M_i$ and $\mathcal{F}$ is compatible, denoted by MIPCF (Most Informative Pruned and Compatible Forest) is FPT. Indeed, analyzing all possible scenarios left/right choices gives simple FPT algorithm, the exponential running times of which only depends on the total number of duplication nodes in $\mathcal{M}$.

5.3 Experiments

We now present an application of the algorithms described in this chapter to analyse the HOGENOM database release 4 [Penel et al., 2009]. HOGENOM is a database of homologous genes from 514 fully sequenced genomes[5] for 381 species, containing 147,586 gene families for which alignments and trees are available. We focused on building trees at the species level, thus we only retained the 46,419 families containing taxa spanning more than two species and for which gene trees are binary[6]. Other gene families concern different strains of the same few species, which can be of use when studying macro-evolutionary events, *e.g.*, gene duplications and losses (see Section 2.2), but are of no use when building the species tree.

The 46,419 families span 376 species and 33,041 of these families have several sequences from the same species, their gene tree being hence a MUL-tree. This first observation shows that only 28.9% of the gene families can be used directly by supertree methods. This echoes, though less severely, the critic of Bapteste et al. [2008] who called "Trees of 1%" the species trees built by the first phylogenomic works that could rely only on single-labeled trees [Brochier et al., 2005; Ciccarelli et al., 2006]. We note that as more complete genomes will be available, the percentage of multi-labeled gene trees will only increase.

In this chapter, we proposed fast algorithms that allow to process MUL-trees in order to distinguish and extract the speciation signal from the signal due to macro-evolutionary events such as gene duplications and transfers. The significant increase in the number of gene families whose phylogenetic signal can then be used is expected to allow phylogenomic methods to obtain a more accurate picture of the estimated species trees. Targeted phylogenomic methods are both supermatrix and supertree approaches, though here we will focus on the latter as this manuscript puts the emphasis on the latter approach.

[5] In details, HOGENOM contains the complete genome for 34 eukarya, 437 bacteria and 39 archaea.

[6] Recall that on the 46,535 gene trees containing taxa spanning more than two species, only 116 are not binary.

5.3.1 Enlarging the amount of gene families to be used for species tree building

The afore-described forest Fcontains both single-labeled and MUL trees. The latter can be turned into single-labeled trees by pruning isomorphic parts (Algorithm 8), pruning less informative subtrees of odns (Algorithm 10) and/or summarizing the triplets they contain that carry the speciation signal (Section 5.2.2). To explore the interest in these different approaches, we distinguished several sets of single-labeled gene trees obtained from F:

- F_1, the forest of single-labeled gene trees of F;
- F_2, the forest of trees of F that are multi-labeled and can be turned into single-labeled trees when removing a copy of each pair of isomorphic sibling subtrees (Section 5.2.1);
- F_3, the forest of trees of F that are still multi-labeled after applying the isomorphic simplification, but are auto-coherent (Section 5.2.2). This third set of trees can be turned into single-labeled trees by two alternative ways:
  - F_3^p is the set of trees obtained from F_3 by applying the algorithm of Section 5.2.3 (*i.e.*, by keeping for each duplication node, the largest subtree);
  - F_3^s is the set of trees obtained when summarizing each MUL-tree M of F_3 by another tree containing only its speciation signal. This is done by first computing the linear triplet decomposition $R^l_{wd}(M)$ of the tree, then obtaining a tree T that represents as much as possible this set of triplets while not containing at all additional, hence arbitrary, triplets. For building T we rely on the PhySIC heuristic algorithm [Ranwez et al., 2007a, see Section 4.2] since, if running on pre-computed triplet sets, this method is significantly faster than *PhySIC_IST* (see Section 4.5 for a discussion on the running times of these methods).

Note that F_1, F_2 and F_3 correspond to mutually exclusive sets of HOGENOM families, while F_3^p and F_3^s are composed of alternative single-labeled trees that correspond to the same families. Note also that some families of F do not fall in either of these categories *i.e.*, those corresponding to MUL trees that are not auto-coherent. Then we considered the largest data sets that can be composed by combining these forests, *i.e.*, $F^s_{all} = F_1 \cup F_2 \cup F_3^s$ and $F^p_{all} = F_1 \cup F_2 \cup F_3^p$. These forests, composed of single-labeled trees, can be assembled by supertree methods to produce species trees. For this purpose, the most informative forest is obviously the union of F_1, F_2 with either F_3^p or F_3^s. Note that F_3^s and F_3^p cannot be used at the same time, since this would bias the supertree inference toward the phylogenetic signal contained in families of F_3. Note also that applying Algorithm 10 to F_3, uninformative trees can be obtained ,*i.e.*, trees that contain less than two taxa. These uninformative trees are not included in F_3^p and this explains why $|F_3^s| \neq |F_3^p|$.

	F_1	F_2	F_3^s	F_3^p	F_{all}^s	F_{all}^p
nb trees	13,378	11,891	17,674	16,148	42,943	41,417
total nb triplets	151,287	2×10^6	421×10^6	424×10^6	423×10^6	426×10^6
avg nb triplets/tree	11	169	23,819	26,261	18,472	10,291
nb distinct triplets	68,538	601,429	22.9×10^6	22.2×10^6	22.9×10^5	22.3×10^6
nb of taxa	369	374	374	374	376	376
% of input triplets	0.3%	2.3%	86.8%	84.4%	86.9 %	84.4%

Table 5.1: **Information contained in the six considered forests to build the species tree for the 376 species present in HOGENOM**. The first row reports the number of trees in each forest, while other rows give indications on the amount of triplet information contained in the forests. Considered triplets are *speciation triplets* $R_{wd}(M_i)$ as defined earlier in this chapter. The second row reports the total number of triplets (with repetitions) for each forest (*i.e.*, the sum of $|R_{wd}(M_i)|$ for all MUL-trees M_i in the forest). The third row is the average number of speciation triplets per tree. The fourth row displays the number of distinct triplets, *i.e.*, when not considering the fact that some triplets are found several times. The fifth row reports the total number of taxa in each forest. The sixth row details the percentage of speciation triplets available as input to the methods in proportion of the number of possible triplets for building a supertree of that size *i.e.*, $\binom{\text{nb taxa}}{3}$.

We first report on characteristics of the forests detailed above (see Table 5.1). This allows to measure the phylogenetic signal contained in each part of the initial tree collection and the gain obtained by the possible enlargements of the F_1 forest. This is measured here using both the number of trees in the forests and the number of triplets they contain. To that aim, we report sizes of R_{wd} sets, rather than that of R_{wd}^l sets, because this gives a more precise idea of the information contained in the collections.

From the number of trees in the different collections displayed in Table 5.1, it can be observed that the algorithms proposed in this chapter allow to use up to 43k gene families instead of the 13k trees corresponding to orthologous genes with no detected paralogs. These 43k trees represent more than 90% of HOGENOM gene families, *i.e.*, more than three times the number of gene families that can be used in classical supertree-based phylogenomic studies.

What is even more impressive is the gain in the amount of topological information for building the species tree. Indeed, from the second and third row of the table, it can be seen that trees in F_1 include on average few species. This is due to the fact that most of the large trees contain duplication nodes. Indeed, widening the scope of considered species for a same family increases the probability of observing duplicated sequences. This is particularly true for some species that are known to have undergone ancient duplications of their whole genome. Taking the presence of duplications into account, even in a very simple way as done to obtain F_2, allows a significant increase in the expressed phylogenetic signal. Indeed, though

F_2 contains roughly the same number of trees than F_1, it contains 10 times more speciation triplets. However, as F_2 only allows for identical resolution of duplicated sequences, most trees containing several duplication and/or transfer events can only be represented in the F_3 forests. The table shows that the more refined analyses conducted to compose F_3^p and F_3^s lead to a considerable increase in the number of speciation information extracted (about 2,000 times more speciation triplets than F_1 and 300 times more distinct speciation triplets).

Moreover, the increase of the additionally available information better covers the set of all possible triplets, as the number of distinct triplets for which the input forest contains a resolution goes from 68.5k to almost 23 millions. In terms of percentage of information available to build a species tree, the last row of Table 5.1 shows that the critic of Bapteste et al. [2008] was well founded since less than 1% triplets of all possible triplets are contained in the F_1 forest. In contrast, this increases up to 86.9% in the best case that we can now consider (forest F_{all}^s).

5.3.2 Running times

All algorithms have been implemented in C++ using Bio++ [Dutheil et al., 2006]. In table 5.3.2 we report the running times of the algorithms presented in Sections 5.2.1-5.2.3 on the HOGENOM data base using a Linux-based machine running with 3 GHz processor and 4 GB RAM.

applied algorithms	input	output	runn. time
checking if $\mathcal{D}(M) \neq \emptyset$ (Alg. 7)	46,335 trees of F	F_1	2m20s
Algorithm 8	33,041 trees not in F_1	F_2	5m1s
AncestralBuild algorithm to $\mathcal{R}_{wd}^l(M)$	21,150 trees not in F_2	F_3	14m40s
Algorithm 10	17,674 trees of F_3	F_3^p	0m14s
PhySIC algorithm	17,674 trees of F_3	F_3^s	21m14s

Table 5.2: Running times of the algorithms presented in Sections 5.2.1- 5.2.3 on the HOGENOM gene tree collection.

5.3.3 Improvement in supertree inference

It now remains to be seen whether the increase in the amount of available information benefits the species tree construction step, *i.e.*, whether the extracted information is of good quality. This is the question we now address. To build supertrees, we composed several data sets from the above forests: the four forests F_1, F_2, F_3^s, F_3^p were each considered separately, then we considered the two largest forests that could be composed from these basic ones, namely F_{all}^s and F_{all}^p. Two supertree methods were considered: the well-known MRP method [Baum and Ragan, 2004, see Section 3.3.2.1] and the PhySIC_IST method [Scornavacca et al., 2008, see Section 4.3]. Recall that the two methods differ in the way they deal with contradictory

topological signals found in the source trees. MRP is a voting method, *i.e.*, arbitrating between conflicting signals in favor of the most frequent one being guided by the maximum parsimony criterion. In contrast, PhySIC_IST is a non-plenary method merely built from a veto principle. As a result, PhySIC_IST infers more reliable but less resolved supertrees. This veto behavior can be tempered by removing the less significantly frequent triplets from the input trees. This preprocess is regulated by the STC parameter (see Section 4.3.2.3), for which we used different values in our experiments: 0.9, 0.8 and 0.5, ordered by increasing tolerance to contradictory signal.

	F_1	F_2	F_3^s	F_3^p	F_{all}^s	F_{all}^p
nb of taxa	369	374	374	374	376	376
CIC of PhySIC_IST (0.9)	2%	12%	48%	46%	47%	44%
# species PhySIC_IST (0.9)	22	67	204	198	200	189
CIC of PhySIC_IST (0.8)	3%	16%	59%	54%	57%	51%
# species PhySIC_IST (0.8)	22	81	241	225	234	213
CIC of PhySIC_IST (0.5)	3%	19%	81%	79%	60%	61%
# species PhySIC_IST (0.5)	23	96	323	318	246	248
CIC of MRP supertree	N/A	N/A	98.01%	99.90%	99.73%	99.95%
# of most pars. trees for MRP	N/A	N/A	510	2	4	1

Table 5.3: **Characteristics of the supertrees built by MRP and PhySIC_IST from investigated forests**. The first row reports the total number of taxa of each forest. CIC values [*i.e.*, resolution degree, Scornavacca et al., 2008, see Section 4.3.1] of the inferred supertrees are detailed, as well as the number of species in the supertrees for the non-plenary PhySIC_IST method. For the computation of the CIC values for *PhySIC_IST*, the number of taxa missing in the supertree have been calculated with respect to the total number of input taxa (first row). The latter method was run for three different values of its STC threshold (*i.e.*, contradiction intolerance, see main text and Section 4.3.2.3): 0.5, 0.8 and 0.9. Last row details the number of most parsimonious trees found by MRP in each case. On data sets F_1 and F_2, MRP was interrupted after a week computation (N/A entries).

A first general observation is that, the resolution degree (CIC value) of the supertrees proposed by all methods increases when going from F_1 to F_2 and from F_2 to F_3 forests (see Table 5.3). When going from F_3^x forests to the corresponding F_{all}^x ones, the MRP method follows again the same tendency, while the PhySIC_IST method does not. This is however explained by an increase in the level of contradictory signal present in the information that PhySIC_IST extracts from the forests when going from F_3^s to F_{all}^s and similarly from F_3^p to F_{all}^p by adding the trees of F_1 and F_2 (data not shown). This can be explained by the fact that the latter forests contain trees with few taxa (see table 5.1) that likely do not represent the overall diversity of the studied groups. As such, they might be less accurate. Indeed, several studies [among others Bininda-Edmonds and Stamatakis, 2006; Hillis, 1998] have demonstrated the general benefit of adding taxa to the analysis *e.g.*, to break long

branches (see Section 4.3.4.6 for an example).

We first analyze the results of the MRP method. On data sets F_1 and F_2, the method was interrupted after a week computation. Most probably, the method couldn't give any supertree in these cases[7] due to the too poor phylogenetic signal contained in the forests (as can be checked in Table 5.1). As a result, the parsimony criterion could not distinguish between candidate supertrees due to a huge number of most parsimonious trees. Other data sets did not suffer from this problem as they contained several thousand times more signal. However, even for the relatively large data sets F_3^p and F_3^s, the parsimony analysis found several most parsimonious trees. The number of most parsimonious trees was always reduced when completing these forests with the relatively small F_1 and F_2 forests (*i.e.*, data sets F_{all}^s and F_{all}^p). This shows how important it is to use every possible bit of information that can be extracted from the data when dealing with such large phylogenies spanning the origins of life.

When observing the structure of the inferred supertrees, for all data sets it can be observed that domains are respected up to 5 taxa over the 376 considered: Archaea and Eukaryotes are monophyletic, while Bacteria are splitted into several paraphyletic groups. Moreover, the number of badly placed species always decreases when going from F_3^p, F_3^s to F_{all}^p, F_{all}^s forests, again showing the interest in using all possibly available information.

The five problematic species are:

- the *Candidatus Carsonella ruddii* is a gamma Proteobacterium that lives within the cells of an insect. Its genome is so reduced that Carsonella may be in the process of becoming an organelle such as the mitochondrion. This bacteria groups with Archaea for the F_3^s data set and within Eukaryotes with F_{all}^s. It is however placed just outside Eukaryotes in other data sets;

- the *Encephalitozoon cuniculi* (a.k.a. *microsporidians*) is a highly derived Fungus that parasites the cells of animals. Its sequences are so fast evolving that it is basically always at the base of the eukaryotes tree due to long branch attraction, but it was shown to go with Fungi in the late 90s by the groups of Manolo Gouy and Martin Embley who used specific non-stationary models. It groups with bacteria when building supertrees from F_3^s and F_3^p, however it goes to the root of eukaryotes when analyzing to F_{all}^s and F_{all}^p.

- the *Guillardia theta* is an extremely reduced red algae that lives within another alga. It has retained a minuscule genome and its sequences are very fast evolving. This eukaryote behaves like *Encephalitozoon cuniculi* except that it is correctly placed only when using F_{all}^s. This species is well known to be problematic from a phylogenetic point of view, as it results from a long branch.

- the two bacteria *Aquiflex aeolicus* and *Thermotoga* are hyperthermophilic bacteria that usually place at the base of the bacterial tree. However, many people

[7] Even when asked to restrict to a small number of most parsimonious trees.

[*e.g.,* Brochier and Philippe, 2002] think that they are misplaced due to amino acid composition biases. In RNA trees, they may be attracted towards the base of the tree due to high G+C content, similar to that of hyperthermophilic archaea. It is believed that these taxa are indeed the closest bacteria from archaea [*e.g.*, Henz et al., 2005] since they have picked up many genes via HGT from hyperthermophilic archaea. In this sense, they are typically close to archaea in many large scale automated analysis that do not correctly identify these transfers. These bacteria branch from a polytomous node at the root of archaea when analyzing F^s_{all} but are within bacteria for other data sets.

The fact that bacteria are paraphyletic could be due to several effects. Firstly, perturbations introduced by an incorrect rooting of gene trees in general: the midpoint rooting procedure was used in HOGENOM without manual curation. Second, it has been established that some genes in eukaryotes have an endosymbiotic origin: mitochondria from alpha proteobacteria and plastids from cyanobacteria [Gray, 1992; Margulis, 1993]. Thus, it is likely that such eukaryotic genes vote for an incorrect placement of eukaryotes inside bacteria, making the latter paraphyletic.

Nonetheless, species from the three domains are overall well separated in inferred supertrees. This shows the general good quality of the speciation information that we extracted from HOGENOM multigene families thanks to algorithms presented here. That is, not only one can now extract more phylogenetic signal from phylogenomic databases, but this signal seems to be useful to build species trees. The next step is looking into details of the changes induced in the species tree inferred when going from F^p_3, resp. F^s_3 to F^p_{all}, resp. F^s_{all}, but this deeper analysis is beyond the scope of this manuscript. A collaboration with the group that maintains the HOGENOM database [Penel et al., 2009] is needed to conduct further studies. The results obtained on the HOGENOM data by the PhySIC_IST supertree method are complementary to those obtained by MRP. Overall, the supertrees output by PhySIC_IST are less resolved (as can be observed by CIC values of Table 5.1), but more correct phylogenies seem to be inferred in return as far as our analysis goes, *i.e.*, mostly looking at the separation between eukaryotes, bacteria and archaea. In all inferences from F_1, $F^s_3, F^p_3, F^s_{all}, F^p_{all}$, eukaryotes were always monophyletic, as well as archaea. Bacteria were monophyletic in 13 of these trees, while one group of bacteria went to the root of the tree for the data set F^s_{all} analysed with threshold 0.8 and one group of bacteria went to the root of the archaea in the supertree inferred from F^s_3 with threshold 0.8. Supertrees proposed from forest F_2 form a less idyllic picture, since we observe the same problems as for MRP supertrees, *i.e.*, several bacteria branching into the eukaryotic group.

We note that the smaller CIC values obtained by PhySIC_IST in comparison to MRP are almost exclusively explained by the fact that some species are not inserted, *i.e.*, the PhySIC_IST supertree contains very few polytomies (unresolved nodes), most trees being binary. This goes to an extreme for the smallest forest, where PhySIC_IST supertrees contain less than 10% species, and only eukaryotes. This indicates that the method finds the positioning of bacteria and archaea too difficult

given the small amount of information available in F_1. Recall also that MRP could not terminate for this forest. The supertrees proposed by PhySIC_IST in this case conform mostly to what is known on eukaryotes, *e.g.*, as encoded in the NCBI taxonomy. The two differences are *Encephalitozoon cuniculi* going to the root of the eukaryotes, and the group composed of *Leishmania major* and *Trypanosoma brucei* that goes into the *Coelomata* group instead of being at the root of eukaryotes. Recall that the eukaryote *Encephalitozoon cuniculi* is a problematic species for MRP. As an improvement, PhySIC_IST places it most often at the basis of the eukaryotic group, and not among bacteria. Though, the acknowledged position for this taxa is deeper in the eukaryotes. All in all, this confirms the hypothesis of a problematic positioning of this taxa in the HOGENOM gene trees.

In contrast to what happens for F_1, supertrees inferred by PhySIC_IST from other forests contain species from the three super kindgoms, most usually well-separated as indicated above. Lastly, we note that the resolution proposed by PhySIC_IST supertrees for these groups oscillates between the two possible topologies, *i.e.*, the two grouped ones being different depending on the forests, and sometimes also depending on the STC thresholds used. This confirms that contradictory signal exist in HOGENOM data for deciding how to root the Tree of Life, likely due to a too crude rooting procedure of the gene trees, as recognized by the authors.

5.4 Conclusions

In this chapter we have presented several algorithms to transform multi-labeled evolutionary trees into single-labeled ones so that they can be used by all existent supertree methods. We studied the impact of these algorithms on a phylogenomic database. Results showed that not only these algorithms allow to extract more information with respect to traditional approaches, but that supertrees inferred from this extra information are much more resolved and, at a first rough level of analysis, globally in accordance with phylogenetic knowledge. Moreover, the effort to obtain efficient algorithms results in very reasonable running times.

Future work includes a more thorough analysis of the inferred supertrees, *i.e.*, to look at the proposed phylogeny for major bacterial groups. However, this could only be done after refining the rooting procedure applied to HOGENOM gene trees.

We also intend to extend the usage of the algorithms presented in this chapter to sequence phylogenomic databases to extract sets of orthologous sequences in data sets containing both paralogous and orthologous sequences. Indeed, once that a gene tree M is reconstruct for a gene family S and the set $\mathcal{D}(M)$ is computed, we can prune isomorphic parts of M (Algorithm 8) and use Algorithm 10 to prune the less informative subtrees of the remaining odns of M. If we prune from S the sequences corresponding to the leaf nodes pruned in M, we obtain the largest set of sequences S' containing only apparent orthologous sequences that can be then assembled into a supermatrix.

Chapter 6

Conclusions and further research

This thesis presents a number of novel results on supertree methods and their applications to the field of phylogenomics.

First, we have presented a review of most supertree methods currently available, with the pros and cons of each of them. This can be useful for those who aim to use a supertree approach but cannot decide among the several available supertree methods. We are currently preparing a theoretical study on supertree methods that explores their links [as done for consensus methods by Bryant, 2003] and determines for each method which properties it possesses among the ones that a *good* supertree method should satisfy [*e.g.*, Goloboff and Pol, 2002; Ranwez et al., 2007b; Steel et al., 2000; Wilkinson et al., 2004b].

Second, we have introduced PI and PC, two strict and desirable properties that a conservative supertree method should satisfy and we designed two supertree methods *i.e.*, *PhySIC* [Ranwez et al., 2007a] and *PhySIC_IST* [Scornavacca et al., 2008] that infer reliable supertrees satisfying these properties, the latter proposing more resolved supertrees that can be non-plenary. *PhySIC* can help the users to evaluate the quality of the input forest. Indeed, the polytomies of the *PhySIC* supertree are labeled to indicate whether a further resolution of the clade has been impeded since not respecting PI and/or PC. Thanks to this tagging, *PhySIC* points out whether the unresolved parts of the supertree are due to a lack of information (PI), which can be overcome by adding more trees to the input forest, and/or to contradictions between source trees (PC). In the latter case, a deeper look to the input trees is needed to determine the origins of contradictions. This can be done using our statistical preprocessing of source trees, *i.e.* the STC, that allows the correction of source trees using the triplet information contained in other source trees. Indeed, trees that contain information massively contradicted by other source trees will be modified by the STC. This preprocessing thus allows to identify rogue source tree resolutions. The STC is also very useful in supertree inference since it allows the correction of source trees before applying a veto supertree method. This approach has the advantage of separating the liberal resolution of conflicts among source trees from the assemblage of the supertree, allowing the user to control the extent to which the source trees can be modified. In practice, the STC is the "missing link" between veto and liberal methods.

Third, we have proposed several algorithms to extract the largest amount of speciation signal from *multi-labeled* trees [Scornavacca et al., 2009a,b], and put it in the form of single-labeled trees. Those trees can then be exploited by supertree

methods or be used to identify the largest set of orthologous sequences for each gene family and assemble them into a supermatrix. We put the emphasis on the fact that this is the first approach that allows to include multi-labeled trees in a supertree analysis. The application of our approach to the HOGENOM database shows, as already pointed out by Bapteste et al. [2008], that multi-labed trees can no more be ignored in supertree inference if we want to have a reliable picture of species evolution.

PhySIC and *PhySIC_IST* supertree methods and the STC preprocessing are freely available on the ATGC bioinformatics platform (`http://www.atgc-montpellier.fr/`). A program implementing the algorithms turning multi-labeled trees into single-labeled ones (presented in Chapter 5) will be soon available on the same platform. All these softwares have been implemented in C++ using the Bio++ libraries [Dutheil et al., 2006]. Moreover, the routine for automatically rooting trees (presented in Section 4.3.3) is now part of the Bio++ Suite (`https://gna.org/projects/bppsuite`).

The work presented here can be extended in several directions.

The results we presented on multi-labeled trees focused on the speciation signal, but other *Gene Evolution Events* (GEE), or *macro-events* such as gene duplications, gene losses, and/or lateral gene transfers can occur in gene history. Taking explicitely into account such events enables to explain the observed incongruency between a gene tree and a corresponding species tree. The approach taking these events into account is called *reconciliation* [*e.g.*, Chauve et al., 2008; Chauve and El-Mabrouk, 2009; Chen et al., 2000; Hallett et al., 2004; Hallett and Lagergren, 2000; Ma et al., 2000; Slowinski and Page, 1999; Vernot et al., 2008]. We are working on an algorithm to simultaneously identify duplications, losses and lateral gene transfers. Our approach extends the mathematically rigorous model of Hallett et al. [2004] by associating to duplications, transfers and losses different costs that can vary across genes and branches of the species tree. Indeed, different genes often evolve at different rates, and even a single gene may evolve at different rates in different organisms (*i.e.*, areas of the species tree). Not accounting for this heterogeneity may lead to inaccurate reconciliations. In a second time, we aim at taking into account the possible inaccuracies in gene trees since a major problem with the today reconciliation methods is that they assume that both the gene and the species trees are error free. To demonstrate the relevance of our approach to the reconciliation problem, we plan to conduct large-scale simulations, adapting the model of horizontal gene transfer of Galtier [2007] to model also gene duplications and losses.

Future work includes also more thorough analyses of the HOGENOM database, in collaboration with the group that maintains this database [Penel et al., 2009]. Our first aim is to refine the rooting procedure applied to HOGENOM gene trees and we conjecture that this can be done using the speciation signal contained in a MUL tree, *i.e.*, $\mathcal{R}_{wd}(M)$. Indeed, some MUL trees contained in HOGENOM are not auto-coherent with the current rooting but they turn out to be auto-coherent when choosing a different root. Moreover, some auto-coherent MUL trees contain more coherent speciation signal when rooted differently. We thus think that $\mathcal{R}_{wd}(M)$ can

be used to develop a better rooting procedure for multi-labeled gene trees. The reconciliation algorithm on which we are currently working can be also used for this purpose, since multi-labeled trees are often rooted using ancient duplication events [*e.g.* Brown and Doolittle, 1995; Gogarten et al., 1989; Gribaldo and Cammarano, 1998; Iwabe et al., 1989; Lawson et al., 1996]. Another purpose of this collaboration is to bring new insights on the phylogeny of the major bacterial groups, which is still debated. The preliminary results we obtained on HOGENOM data (presented in Section 5.3) convinced us that the new information, resulting from our extraction of the speciation signal of multi-labeled trees will give us a clearer picture of bacterial evolution.

Chapter 7

Résumé en français

Une question qui passionne un nombre croissant de scientifiques, en particulier depuis les dernières décennies, est de comprendre comment tous les organismes sur terre descendent d'un ancêtre commun.

Depuis Aristote, les naturalistes ont toujours essayé de trouver un ordre dans l'abondance de créatures qui peuplent la Terre. Leclerc de Buffon fut le premier naturaliste à évoquer la possibilité que les espèces puissent évoluer. Avant ce dernier, toutes les classifications étaient proposées dans le cadre du fixisme, une théorie affirmant que la vie sur Terre a toujours été composée des espèces que nous observons aujourd'hui et que ces espèces ne changent pas. Charles Darwin, le très célèbre naturaliste anglais, introduisit la première théorie évolutive, selon laquelle les populations évoluent au fil des générations par le biais d'un processus de sélection naturelle. La découverte de l'ADN par Watson et Crick en 1953 et la mise au point des techniques de séquençage, ont permis l'utilisation d'un nouveau type d'information, les données moléculaires (*e.g.* séquences d'ADN ou de protéines, codons, etc.) qui se sont ajoutées aux données morphologiques (*e.g.* aspects structurels des organismes tels que la présence de certains os du crâne, organes, etc.) utilisées jusque là pour étudier les relations évolutives entre les espèces.

Le champ de recherche de la biologie qui étudie les relations évolutives entre les espèces grâce à des données moléculaires et morphologiques est appelé *phylogénétique*. Ces relations peuvent être résumées dans un arbre communément appelé *arbre (ou phylogénie) des espèces*. Les données moléculaires et morphologiques sont exprimées sous forme de séquences de caractères qui peuvent prendre plusieurs états, tels que, $\{0, 1\}$ pour la présence/absence d'un trait morphologique, {A, C, G, T} pour les sites nucléotidiques, etc. Pour reconstruire des phylogénies, deux types de méthodes sont disponibles :

- les méthodes basées sur les caractères, qui évaluent les similitudes entre espèces en comparant les états observés pour chacun des sites (positions) des séquences; les méthodes basées sur les caractères peuvent être subdivisées en :
  - méthodes de parcimonie
  - méthodes de vraisemblance
  - méthodes bayésiennes
- les méthodes basées sur les distances, s'appuient sur une quantification de l'évolution séparant chaque couple d'espèces (ou distance évolutive) pour reconstruire une phylogénie.

Dans un premier temps les biologistes espéraient que les phylogénies reconstruites à partir des différents jeux de données seraient toutes équivalentes et qu'elles coïncideraient avec la phylogénies des espèces. Malheureusement ce n'est pas le cas : pour des raisons à la fois méthodologiques et biologiques, les phylogénies inférées à partir dès différents jeux de donnés peuvent différer entre elles et différer de l'arbre des espèces.

En e?et, le fait que le processus évolutif suivi par des séquences soit mal estimé peut aboutir à la construction d'un arbre de gène erroné.

De plus, les macro-événements dans l'évolution des génomes, comme par exemple la duplication des gènes dans un génome, peuvent aussi conduire à des conflits topologiques entre les phylogénies. Ces conflits apparaissent notamment lorsque certaines copies dupliquées d'un gène sont absentes de l'analyse, soit parce qu'elles n'ont pas été séquencées, soit parce qu'elles ont été perdues à un moment donné au cours du processus d'évolution. Par exemple, dans la figure 7.1, selon l'arbre des

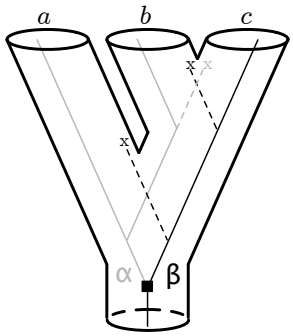


Figure 7.1: La duplication des gènes peut produire des conflits entre l'arbre de gène et l'arbre des espèce

espèces, représenté comme des tuyaux épais, b et c sont *évolutivement* plus proches l'un de l'autre qu'ils ne le sont de a. Supposons que, en raison de pertes pendant le processus d'évolution, les séquences disponibles sont la copie α pour les espèces a et b et la copie β pour l'espèce c. Dans ce cas, l'arbre de gènes (représenté sous forme de lignes fines à l'intérieur des tuyaux) groupe a et b, mais ces espèces ne sont pas les plus proches en termes d'événements de spéciation.

Pour estimer l'arbre des espèces, les biologistes analysent donc simultanément plusieurs jeux de données (ou "matrices") correspondant à différentes familles de gènes, pour faire émerger le signal de spéciation.

L'approche la plus immédiate pour combiner des données provenant de plusieurs sources est simplement de concaténer les séquences d'origine dans une seule grande matrice appelée *supermatrice*. Une deuxième façon de combiner plusieurs jeux de données consiste à reconstruire dans un premier temps des arbres (appelés communément arbres *sources*) à partir de chaque jeu de données, puis à les assembler en un arbre plus grand, appelé *super-arbre* [Bininda-Emonds, 2004b].

Ces deux approches, traditionnellement considérées comme des concurrentes,

présentent toutes deux des avantages et des inconvénients lors de l'analyse de grandes quantités de données. Nous sommes convaincus qu'aucune de ces deux approches n'est nettement meilleure que l'autre et qu'un choix ad hoc doit être fait pour chaque ensemble de données, en fonction de sa taille, du type de données etc. En outre, ces deux approches peuvent être utilisées parallèlement sur un même jeu de données afin d'exploiter les points forts et de contrebalancer les faiblesses de chaque méthode. Elles peuvent également être combinées en une stratégie diviser-pour-régner [Bininda-Edmonds and Stamatakis, 2006; Bininda-Emonds, 2005].

Ce travail de thèse s'est focalisé sur l'approche super-arbre pour combiner les jeux de données. Dans les dernières décennies une grande quantité de méthodes de super-arbre ont été proposées. Les méthodes de super-arbre peuvent être classées en trois catégories, selon leur façon de traiter les conflits topologiques *i.e.*, des dispositions différentes des mêmes espèces parmi les arbres sources.

La première série de méthodes ne peut pas gérer les arbres sources incompatibles, c'est-à-dire en désaccord sur la position phylogénétique de certaines espèces ou groupes d'espèces, appelés respectivement *taxons* et *clades*. Les méthodes pionnières qui appartiennent à cette catégorie sont Build [Aho et al., 1981] et le strict consensus supertree [Gordon, 1986]. Puisque les phylogénies sont souvent en conflit les unes avec les autres [Bininda-Emonds, 2004c, p4], ces méthodes sont d'un usage limité.

Les méthodes *libérales* ou de *vote* résolvent les conflits [Thorley and Wilkinson, 2003], en "faisant voter" les arbres sources et en optant pour l'alternative topologique qui maximise un critère d'optimisation (celui-ci variant d'une méthode à l'autre). L'espoir est que chaque taxon soit placé de façon erronée dans seulement quelques arbres et que cette information erronée soit surmontée par le grand nombre d'arbres sources où le taxon est correctement placé. Quelques exemples de méthodes de vote type sont la Représentation Matricielle avec Parcimonie (MRP, Baum [1992]; Ragan [1992]), Modified-MinCut (mmc, Page [2002]) et l'Average Consensus Supertree Lapointe and Cucumel [1997]. Même si les super-arbres proposés par ces méthodes sont souvent de très bonne qualités, plusieurs auteurs ont montré que dans certains cas, cette approche peut conduire à proposer des super-arbres contenant des clades qui contredisent tous les arbres sources [Cotton et al., 2006; Goloboff, 2005; Goloboff and Pol, 2002].

La troisième série de méthodes adoptent une philosophie de *veto* : le message phylogénétique de chaque arbre source est respecté. Ainsi, un clade est retenu dans le super-arbre si, et seulement si, les topologies sources sont unanimement en accord avec sa présence. Ces méthodes *éliminent* les conflits [Thorley and Wilkinson, 2003], soit en proposant des multifurcations dans le super-arbre [*e.g.*, Goloboff and Pol, 2002] siot en élaguant les taxons problématiques [*e.g.*, Berry and Nicolas, 2004, 2007]. Quelques exemples de méthodes de type veto sont des extensions du consensus strict [*e.g.*, Gordon, 1986; Huson et al., 1999], le semi-strict supertree [Goloboff and Pol, 2002], le SMAST et le SMCT [Berry and Nicolas, 2004, 2007].

Les méthodes de super-arbre de type vote et veto peuvent être divisées en méthodes *directes* et *indirectes*. Alors que les premières (*e.g.*, Modified-MinCut et

PhySIC) combinent directement les arbres sources , les secondes (e.g. MRP et l'Average Consensus Supertree) procèdent en deux étapes. Dans un premier temps, elles convertissent les arbres d'entrée en un autre type de données (e.g. séquences binaires, distances), elles utilisent ensuite une méthode de reconstruction phylogénétique classique pour analyser ces données intermédiaires.

Dans cette thèse nous présentons une revue des principales méthodes de super-arbre actuellement disponibles, et nous détaillons les avantages et les inconvénients de chacune d'elles. Cette synthèse devrait permettre de choisir la méthode de super-arbre la plus adaptée, en fonction du problème traité.

Pour reconstituer des grandes parties de l'arbre de la vie, il est préférable d'utiliser une méthode de super-arbres conservatrice afin d'obtenir des arbres très fiables. Dans ce contexte, une méthode de super-arbre doit afficher seulement des informations qui sont présentes dans les arbres sources ou induites par ces arbres (propriété d'induction – PI). De plus, le super-arbre proposé ne doit pas favoriser une résolution plutôt qu'une autre lorsque plusieurs possibilités contradictoires existent, autrement dit, il ne doit pas contenir des informations qui entrent en conflit avec les arbres sources individuellement ou collectivement (propriété de non contradiction – PC). Avant de pouvoir définir formellement ces deux propriétés, nous devons au préalable introduire plusieurs concepts et notations.

Il n'existe que trois arbres binaires enracinés ayant pour uniques feuilles a, b, c. Ces arbres binaires sont appelés *triplets* et sont notés $ab|c$, resp. $ac|b$, resp. $bc|a$, selon l'unique clade non triviale qu'ils contiennent ($\{a, b\}$, resp. $\{a, c\}$, resp. $\{b, c\}$). On dit qu'un arbre T *induit* ou *contient* un triplet t si l'arbre obtenu en restreignant T aux feuilles a, b, c (noté $T|_{(a,b,c)}$) coïncide avec t. Par exemple, l'arbre représenté en figure 7.2 induit, entre autres, le triplet $ab|d$. Si T n'est pas binaire, il

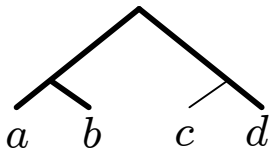


Figure 7.2: L'arbre représenté en figure induit, entre autres, le triplet $ab|d$. L'ensemble $\mathcal{R}(T)$ pour l'arbre en figure contient quatre triplets *i.e.*, $ab|c$, $abd|$, $cd|a$ et $cd|b$.

peut arriver que $T|_{(a,b,c)}$ ne contienne que le clade trivial $\{a, b, c\}$, *i.e.*, $T|_{(a,b,c)}$ est constitué d'un seul nœud interne directement relié aux trois feuilles. Dans ce cas, on dit que $\{a, b, c\}$ est non résolu dans T et on note $T|_{(a,b,c)}$ par la trichotomie (a, b, c). Etant donné un triplet t, $\bar{t}$ représente n'importe lequel des deux autres triplets ayant les mêmes feuilles que t. Tout arbre enraciné T peut être décrit de façon équivalente par l'ensemble des triplets homéomorphes à ses sous-arbres reliant trois feuilles [voir entre autres Grunewald et al., 2007]. Cet ensemble de triplets est notée $\mathcal{R}(T)$ (voir la figure 7.2 pour un exemple). Pour toute collection $\mathcal{F}$ d'arbres phylogénétiques enracinés, $\mathcal{R}(\mathcal{F})$ désigne l'ensemble des triplets présents dans au moins un arbre de

$\mathcal{F}$, *i.e.*, $\mathcal{R}(\mathcal{F}) = \bigcup_{T_i \in \mathcal{F}} \mathcal{R}(T_i)$.

Un ensemble $\mathcal{R}$ de triplets est dit *compatible* si, et seulement si, il existe un arbre T qui contient tous les triplets de $\mathcal{R}$. La compatibilité d'un ensemble de triplets peut être décidée en temps polynomial [Aho et al., 1981].

Étant donné un ensemble de triplets compatible $\mathcal{R}$, on dit que $\mathcal{R}$ induit un triplet t (noté $\mathcal{R} \vdash t$) si, et seulement si, $\mathcal{R} \cup \{\bar{t}\}$ n'est pas compatible, ou encore si tout arbre T qui contient $\mathcal{R}$ contient t [Grunewald et al., 2007]. Par exemple, nous avons que $\{ab|c, bc|d\} \vdash ac|d$ car tous les arbres contenant $\{ab|c, bc|d\}$ contiennent aussi le triplet $ac|d$. En pratique, la forêt d'arbre sources $\mathcal{F}$ et donc l'ensemble $\mathcal{R}(\mathcal{F})$, sont souvent incompatibles. Pour un ensemble incompatible de triplets $\mathcal{R}$, on dit que $\mathcal{R}$ induit un triplet t s'il existe un sous-ensemble compatible $\mathcal{R}'$ de $\mathcal{R}$ qui induit t.

Étant donnée une collection d'arbres sources $\mathcal{F}$ et un super-arbre candidat T pour $\mathcal{F}$, $\mathcal{R}(T, \mathcal{F})$ désigne l'ensemble des triplets de $\mathcal{F}$ pour lesquels T propose une résolution. Autrement dit, l'ensemble $\mathcal{R}(T, \mathcal{F})$ correspond à toute information topologique présente dans la collection $\mathcal{F}$ qui est liée à l'information présente dans le super-arbre T. Plus formellement, $\mathcal{R}(T, \mathcal{F}) = \big\{ab|c \in \mathcal{R}(\mathcal{F})$ tel que $\{ab|c, ac|b, bc|a\} \cap \mathcal{R}(T) \neq \emptyset\big\}$. Notons qu'il est possible que $\mathcal{R}(T, \mathcal{F})$ soit incompatible. C'est notamment le cas dès que T contient un triplet t pour le quel $\mathcal{R}(\mathcal{F})$ propose deux ou trois résolutions différentes.

Avec ces notations, les propriétés d'induction et non-contradiction PI et PC pour une collection d'arbres $\mathcal{F}$ et un super-arbre T peuvent être exprimés comme suit :

- T satisfait PI pour $\mathcal{F}$ si, et seulement si, pour tout $t \in \mathcal{R}(T)$, on a $\mathcal{R}(T, \mathcal{F}) \vdash t$. En d'autres termes, PI exige que chaque triplet de T soit induit par $\mathcal{R}(T, \mathcal{F})$.

- T satisfait PC pour $\mathcal{F}$ si, et seulement si, pour tout $t \in \mathcal{R}(T)$, et tout $\bar{t}$, $\mathcal{R}(T, \mathcal{F}) \nvdash \bar{t}$. Cela signifie que, pour chaque triplet de T, $\mathcal{R}(T, \mathcal{F})$ n'induit aucune solution alternative.

Toute méthode de veto devrait proposer un super-arbre qui vérifie ces propriétés, mais ce n'est pas le cas des méthodes existantes avant ce travail de thèse.

En premier lieu, nous avons donc développé un algorithme polynomial qui permet de modifier un super-arbre T produit par une méthode de super-arbre quelconque afin qu'il satisfasse PI et PC pour une forêt $\mathcal{F}$ donnée. Cet algorithme consiste à identifier les triplets de $\mathcal{R}(T)$ qui ne satisfont pas PI ou PC et à écraser certaines des arêtes de T afin que ces triplets non-justifiés se soient plus dans $\mathcal{R}(T)$.

Nous avons également conçu deux méthodes, *PhySIC* et PhySIC_IST, qui, pour une collection d'arbres donnée $\mathcal{F}$, renvoient d'emblée des super-arbres satisfaisant PI et PC pour $\mathcal{F}$.

La première méthode est appelée *PhySIC* – Phylogenetic Signal with Induction and non-Contradiction [Ranwez et al., 2007b]. L'objectif de cette méthode est de reconstruire des super-arbres qui satisfont PI et PC et qui résolvent le plus grand nombre possible de triplets de $\mathcal{R}(\mathcal{F})$. Plus formellement, étant donné une collection d'arbres $\mathcal{F}$, *PhySIC* vise à proposer un super-arbre T tel que T satisfait PI et PC pour $\mathcal{F}$ et que $\mathcal{R}(T, \mathcal{F})$ a une taille maximale sur tous les sous-ensembles de $\mathcal{R}(\mathcal{F})$.

Nous conjecturons que ce problème est NP-complète. Une preuve de NP-complétude a été proposé dans Guillemot and Berry [2007] mais le problème étudié par les auteurs – MIST (Maximum Identifying Subset of Triplets) – n'est qu'une variante du problème sous-jacent *PhySIC* n'impliquant pas la NP-complétude de ce dernier. La méthode *PhySIC* est donc une heuristique, mais seulement sur la taille de $\mathcal{R}(T, \mathcal{F})$ car elle renvoie toujours des super-arbres qui satisfont PI et PC.

La méthode *PhySIC* consiste en deux étapes. D'abord un super-arbre T_{PC} satisfaisant PC pour une collection d'arbres enracinés $\mathcal{F}$ est calculé par l'algorithme $PhySIC_{PC}$ (voir l'Algorithme 11 de l'Annexe A.1). Deuxièmement, certaines arêtes de T_{PC} sont éventuellement écrasées par l'algorithme $PhySIC_{PI}$ (voir l'Algorithme 14 de l'Annexe A.1) jusqu'à obtenir un arbre T_{PC} satisfaisant aussi la propriété PI. Les deux algorithmes sont basés sur la construction d'un graphe appelé le *graphe de Aho* [Aho et al., 1981].

Le graphe de Aho $\mathcal{G}(\mathcal{R}, L)$ pour un ensemble de triplets $\mathcal{R}$ et un ensemble de taxon L est le graphe ayant L comme sommets et tel que il existe une arrête entre deux sommets a et b si et seulement si il existe un triplet $ab|c \in \mathcal{R}$. On note $v(C_i)$ l'ensemble des sommets d'une composante connexe C_i de $\mathcal{G}(\mathcal{R}, L)$. La restriction de $\mathcal{R}$ aux sommets de C_i est $\mathcal{R}|v(C_i) = \{ab|c \in \mathcal{R} \text{ tel que } \{a, b, c\} \subseteq v(Ci)\}$.

L'algorithme $PhySIC_{PC}$ consiste en trois étapes. Pour première chose $PhySIC_{PC}$ calcule $\mathcal{R}_{dc}(\mathcal{F})$, *i.e,* l'ensemble des triplets tel que $t, \bar{t} \in \mathcal{R}(\mathcal{F})$ et l'ensemble $\mathcal{R}'(\mathcal{F}) = \mathcal{R}(\mathcal{F}) - \mathcal{R}_{dc}(\mathcal{F})$. En effet, les conflits les plus fondamentaux entre les triplets de $\mathcal{R}(\mathcal{F})$ se produisent lorsque deux différents triplets t et $\bar{t}$ apparaissent dans $\mathcal{R}(\mathcal{F})$ pour un même ensemble de trois taxons. Evidemment, ni t ni $\bar{t}$ peuvent être présents dans un arbre qui satisfait PC.

Une fois $\mathcal{R}'(\mathcal{F})$ calculé, $PhySIC_{PC}$ construit le graphe de Aho $\mathcal{G}(\mathcal{R}(\mathcal{F}), L(\mathcal{F}))$. Lorsque le graphe Aho contient plusieurs composantes connexes, elle correspondent à des clades de l'arbre qui est construit pour représenter $\mathcal{R}(\mathcal{F})$. Puis, le sous-clades contenus dans chacun de ces groupes principaux sont trouvés en appliquant l'algorithme de manière récursive pour chaque couple $(\mathcal{R}(\mathcal{F})|v(C_i), v(C_i))$. Les appels récursifs sont arrêtés lorsque les composantes contiennent moins de 3 taxons, puisqu'il n'y a pas de triplets (donc incompatibilité) sur un tel nombre de taxons. Toutefois, si à un certain moment dans le processus récursif le graphe de Aho pour plus de 2 taxons n'a qu'une seule composante connexe C, cela signifie que les arbres sources sont en conflit sur la résolution des taxons dans $v(C)$. Dans ce cas, $PhySIC_{PC}$ renvoie un arbre étoile composé par un noeud connecté à des feuilles ayant comme labels les taxons de $v(C)$ (voir l'Algorithme 12 de l'Annexe A.1).

Dans cette façon $PhySIC_{PC}$ reconstruit un arbre T_{PC} qui satisfait PC par rapport à $\mathcal{R}'(\mathcal{F})$ mais sans garantie que cela vaut aussi à l'égard de $\mathcal{R}(\mathcal{F})$. Pour assurer cela, T_{PC} ne doit résoudre aucun triplet de $\mathcal{R}_{dc}(\mathcal{F})$. Si cela arrive, $PhySIC_{PC}$ écrase certaines des arêtes de T_{PC} afin qu'il ne contienne aucun triplet de $\mathcal{R}_{dc}(\mathcal{F})$. Utiliser $PhySIC_{PC}$ avec $\mathcal{R}'(T)$, en écrasant éventuellement certaines arêtes *a posteriori* et pas directement avec $\mathcal{R}(T)$ permet d'obtenir des super-arbres en moyenne plus résolus.

Le super-arbre T_{PC} renvoyé par $PhySIC_{PC}$ ne satisfait pas généralement la

propriété PI. L'algorithme $PhySIC_{PI}$ transforme T_{PC} pour qu'elle satisfasse aussi PI, en identifiant les triplets de $\mathcal{R}(T_{PC})$ qui ne satisfont pas PI et en écrasant les arêtes de T_{PC} qui induisent ces triplets non-justifiés (voir l'Algorithme 14 de l'Annexe A.1 pour les détails).

Des études de simulation (voir Figure 4.11) ont montré que, dans certains cas, par exemple lorsque les arbres source ne se chevauchent pas suffisamment ou présentent un degré élevé de contradictions, les super-arbres reconstruits par *PhySIC* peuvent être très irrésolus. Puisque nous pensons que PI et PC sont des propriétés très importantes en vue de la reconstruction de l'Arbre de Vie, nous avons conçu une autre méthode de super-abres qui renvoie des super-abres avec ces propriétés, mais en moyenne plus informatifs : *PhySIC_IST* – Phylogenetic Signal with Induction and non-Contradiction Inserting a Subset of Taxa.

Choisir le super-arbre plus informatif parmi plusieurs candidats nécessite de savoir comparer des arbres qui peuvent avoir un nombre de taxons différent (comme ST_1 et ST_2 dans la figure 4.7). Dans ce but nous avons utilisé une mesure basée sur une variation du critère CIC (Cladistic Information Criterion) [Thorley et al., 1998]. Cette mesure a des racines dans la théorie de l'information et est fondamentalement proportionnelle au nombre d'arbres binaires complets qui sont compatibles avec le super-arbre évalué. Plus précisément, le *CIC* d'un super-arbre T relativement à n taxons est défini comme suit :

$$CIC(T, n) = -\lg \frac{n_R(T, n)}{n_R(n)}$$

où $n_R(T, n)$ est le nombre d'arbres binaires à n feuilles compatibles avec T et $n_R(n)$ est le nombre d'arbres binaires ayant n feuilles.

La méthode *PhySIC_IST* fonctionne par insertions successives des taxons sur un arbre squelette. Étant donnée une forêt d'arbres enracinés $\mathcal{F}$, *PhySIC_IST* est principalement constitué des étapes suivantes :

1. ordonner les taxons dans $L(\mathcal{F})$ suivant un ordre de priorité bien déterminé;
2. construire un arbre squelette T formé par un noeud racine relié à deux noeuds qui ont comme labels les deux premiers taxons dans l'ordre de priorité;
3. pour chaque taxon l dans l'ordre de priorité :
   (a) choisir un noeud ou une branche de l'arbre squelette T où insérer l de façon à vérifier PC;
   (b) insérer l dans T à l'emplacement choisi, puis écraser des arêtes afin que l'arbre obtenu, dénoté T', vérifie aussi PI.
   (c) si CIC$(T', L(F))$ > CIC $(T, L(F))$ alors T' est le nouvel arbre squelette.

Les taxons qui ont une priorité élevée sont ceux pour lesquels nous avons le plus d'information en terme de triplets et qui sont impliqués dans moins de contradictions possible. Plus formellement, pour chaque taxon l, on a :

$$\text{priorité(l)} = |\mathcal{R}(l)| - |\mathcal{R}_{dc}(l)|,$$

où on note $|\mathcal{R}(l)|$ (resp. $|\mathcal{R}_{dc}(l)|$) le nombre de triplets qui contiennent l presents dans $\mathcal{R}(\mathcal{F})$ (resp. $\mathcal{R}_{dc}(\mathcal{F})$). En effet, l'insertion d'un taxon qui est présent dans de nombreux triplets de $\mathcal{R}(\mathcal{F})$ fournit de l'information, non seulement sur sa position, mais aussi sur la position des taxons restants. D'autre part, retarder l'insertion des taxons au placement contesté diminue les chances de les placer incorrectement en raison d'informations incomplètes et d'être incapable de procéder à l'insertion des taxons restants.

Afin de choisir l'endroit de l'arbre squelette T où essayer d'insérer le taxon l, on utilise l'information des arbres sources, en déterminant, pour chaque arbre source T_i qui contient l, dans quelle région de T le taxon l peut être inséré sans contredire T_i. Il faut noter que, si tous les arbres sources soutiennent l'insertion d'un taxon dans une région (un noeud ou une branche) de T, l'insérer dans cette région ne créera pas de contradictions entre les arbres sources et le super-arbre. Ainsi, cette insertion ne violera pas PC. En outre, si la région soutenue par les arbres sources est limitée à un noeud ou une arête, cela signifie qu'une telle insertion satisfera aussi PI. Dans les autres cas PI ou PC ne sont pas satisfaites et nous sommes forcés à écraser certaines arêtes de T' avant de comparer le CIC de T et T'.

Cette description de l'algorithme *PhySIC_IST* est fortement simplifiée. Pour plus de détails voir la Section 4.3.2.1.

En moyenne, les super-arbres reconstruits par *PhySIC_IST* sont bien plus résolus que les super-arbres reconstruits par *PhySIC* (voir Figure 4.11) avec un taux d'erreur qui reste très faible (voir Figure 4.12). Ceci est une conséquence de trois différences fondamentales entre *PhySIC* et *PhySIC_IST*. Premièrement, *PhySIC_IST* fonctionne par insertions successives de taxons sur un un arbre squelette et n'est pas basé sur une version révisée de l'algorithme de Aho. En outre, les deux méthodes n'ont pas le même critère d'optimisation : en effet, *PhySIC* vise à trouver le super-arbre satisfaisant PI et PC tel que $\mathcal{R}(T, \mathcal{F})$ a une taille maximal sur tous les sous-ensembles de $\mathcal{R}(\mathcal{F})$ tandis que *PhySIC_IST* cherche un super-arbre satisfaisant PC et PI qui maximise la valeur du CIC. Enfin, *PhySIC_IST* peut proposer des super-arbres non complets, c'est à dire qu'il n'insère pas les taxons qui entraîneraient une baisse du CIC du super-arbre, tandis que *PhySIC* propose nécessairement un super-arbre qui contient tous les taxons présents dans au moins un arbre source.

Cependant, la complexité de *PhySIC* est $O(kn^3 + n^4)$, tandis que *PhySIC_IST* s'exécute en $O(n^3(k + n^3))$, où k est le nombre d'arbres d'entrée de la forêt $\mathcal{F}$ et $n = L(\mathcal{F})$. En plus, *PhySIC* peut donner un retour sur les arbres sources. En effet, le polytomies des super-arbres reconstruits par *PhySIC* sont marqués pour indiquer si une autre résolution du clade n'est pas possible car elle n'aurait pas respecté PC et/ou PI. Grâce à ce marquage, *PhySIC* souligne que les parties du super-arbre non résolues sont dues à des contradictions entre les arbres source (PC) et/ou à une manque d'information (PI), qui peut être surmonté en ajoutant plus d'arbres dans la forêt d'entrée .

Dans ce travail de thèse, nous avons également présenté un pré-traitement statis-

tique des arbres sources, appelée STC (Source Trees Correction), pour détecter et corriger les positions artefactuelles de certains taxons. Ce pré-traitement, pour toute contradiction directe contenue dans $\mathcal{R}(\mathcal{F})$, évalue les alternatives possibles et détecte les triplets qui sont statistiquement moins soutenus en utilisant un test χ^2 [Fienberg, 1977], avec un seuil choisi par l'utilisateur. Dans un deuxième temps le STC modifie chaque arbre source (en utilisant un schéma similaire à celui de *PhySIC_IST*) afin qu'il ne contienne pas les triplets jugés comme non-significatifs et qu'il reste aussi informatif que possible. En d'autres termes le STC vise à corriger les arbres sources qui proposent une position anomale pour certains taxons (en raison de transferts horizontaux de gènes, des attractions longue branche, de la paralogie...). Par exemple, si les arbres sources contiennent deux résolutions contradictoires, l'une présente dans 99 % des arbres et l'autre présente dans 1 % des arbres, on peut raisonnablement penser que cette dernière résolution est une anomalie et décider de l'ignorer.

Si l'utilisateur approuve les modifications proposées, la méthode de veto *PhySIC_IST* est ensuite appliquée aux arbres source modifiés. Le super-arbre résultant satisfait à la fois PI and PC pour la collection d'arbres source modifiés. Si l'utilisateur n'est pas satisfait avec les arbres sources modifiés, il peut modifier le seuil et redémarrer la procédure, ou choisir de l'ignorer. De cette manière, la composante libérale de l'inférence des super-arbres n'est pas seulement rendu explicite, mais également interactive et paramétrée.

Le STC peut être utilisé pour toute collection d'arbres sources et les arbres sources modifiés peuvent être utilisés par une méthode de super-arbres quelconque. Le STC peut donc avantager toute méthode de super-arbres de type veto. En effet, cette approche a l'avantage de séparer la résolution libérale des conflits entre les arbres sources de l'assemblage des super-arbres. Cela rend explicite le choix fait pour arbitrer entre les arbres sources contradictoires et permet à l'utilisateur de choisir le degré avec lequel les arbres sources peuvent être modifiés. Dans la pratique, le STC + *PhySIC_IST* comble l'écart entre les méthodes de veto et les méthodes de vote. Ces recherches ont été appliquées à des problèmes biologiques pour lesquels l'équipe *Phylogénie Moléculaire* (de l'*Institut des Sciences de l'évolution de Montpellier*) dispose de données et d'expertise. Notamment, l'application de PhySIC_IST et du prétraitement des arbres sources au problème complexe de la phylogénie des Triticeae (voir Section 4.4) a permis de mieux comprendre l'histoire évolutive de ce groupe.

Une limite actuelle des méthodes de super-arbres est l'impossibilité de gérer la grande majorité des arbres de gènes qui ont subi des événements de duplication. En effet, ces événements aboutissent presque toujours à la présence de plusieurs copies du même gène dans les génomes, donc les arbres de gènes sont généralement multi-étiquetés, *i.e.*, une seule espèce peut étiqueter plusieurs feuilles. Comme aucune méthode de super-arbres n'existe actuellement pour combiner ce type d'arbres, ils sont complètement ignorés dans les approches phylogénomiques classiques. Pourtant, ils représentent 60% à 80% des arbres de gènes disponibles dans les banques de données phylogénomiques. Dans cette thèse, nous proposons plusieurs algorithmes

pour extraire une quantité maximale de signal de spéciation à partir d'arbres multi-étiquetés. Ce signal est rendu sous la forme d'arbres où chaque espèce n'apparaît qu'une fois, *i.e.,* d'arbres que les méthodes de superarbres savent gérer.

Dans ce travail de thèse, nous nous sommes concentrés sur les arbres multi-étiquetés enracinés binaires, ou arbres MUL pour faire court, comme celui qui est représenté dans la Figure 5.1(i). Ne traiter que les arbres binaires n'est pas si restrictif, puisque, comme évoqué dans la Section 1.8, les méthodes pour reconstruire des phylogénies produisent généralement des arbres binaires. Par exemple, dans la base de données HOGENOM [Penel et al., 2009], parmi les 46.535 arbres de gènes contenant plus de deux espèces, seulement 116 ne sont pas binaires. Pour un arbre multi-étiqueté M, nous avons conçu un algorithme linéaire en $O(L(M))$ pour identifier les noeuds de duplication.

Nous avons aussi adapté l'algorithme d'isomorphisme de Gusfield [1991] aux arbres multi-étiquetés, en préservant le temps d'exécution linéaire. Cet algorithme peut être utilisé pour faire baisser le nombre de noeuds de duplication dans les arbres de gènes, en ne gardant qu'une copie des sous-arbres isomorphes "frères" dans une approche bottom-up.

Pour un arbre de gènes M qui reste multi-étiqueté après avoir gardé une seule copie des sous-arbres isomorphes frères, nous avons défini un sous-ensemble des triplets contenus dans M. Ce sous-ensemble, noté $\mathcal{R}_{wd}(M)$, contient les triplets de M qui donnent de l'information sur le signal de spéciation, utile pour reconstruire l'arbre des espèces. Si le signal de spéciation de M peut être contenu dans un arbre non multi-étiqueté, on dit que M est auto-cohèrent. L'auto-cohérence d'un arbres multi-étiqueté peut être calculée en temps linéaire. Si M est auto-cohèrent, son signal de spéciation peut être résumé dans un arbre non multi-étiqueté par une méthode de super-arbres basée sur les triplets comme *PhySIC* et *PhySIC_IST*.

Pour un arbre de gènes M qui n'est pas auto-cohèrent, nous avons proposé un algorithme linéaire pour extraire une sous-arborescence maximale qui est à la fois auto-cohérent et libre d'événements de duplication.

Une application de ces algorithmes à la base de données HOGENOM est présentée. Les résultats ont montré que ces algorithmes permettent d'extraire plus d'information que les approches traditionnelles; notamment la forêt obtenue en utilisant ces algorithmes contient environ 23 millions de triplets (sans compter les doublons), au lieu des environ 68K de la forêt constituée que d'arbres non multi-étiquetés. En plus, les super-arbres déduits à partir de ces informations supplémentaires sont beaucoup plus résolus et, à première analyse, conformes aux connaissances phylogénétiques d'aujourd'hui. En outre, les temps d'exécution sont très raisonnables (quelques minutes pour tester et convertir les arbres sources).

L'accent de cette thèse est mis sur des résultats théoriques mais les applications à la vraie vie ont toujours été gardé à l'esprit. Chaque partie de ces travaux de recherche présente des algorithmes pour lesquels un programme convivial est disponible en téléchargement ou pour exécution en ligne sur la plate-forme de bioinformatique. En outre, les contributions théoriques de cette thèse sont appliquées à des études de cas biologiques afin de cerner leurs intérêts et leurs limites.

Appendix A

Appendix to Chapter 4

Contents

A.1 Outline of main *PhySIC* subroutines

Algorithm 11: Details of the $PhySIC_{PC}$ subroutine taking a set S of taxa and a set $\mathcal{R}$ of triplets on S as input.

```
1  Algorithm PhySIC_PC (S, R)
2  if S contains less than 3 taxa then return the trivial tree on S;
3  Let G denote the Aho graph for R;
4  if G has several connected components then C_PC ← CC(G) ;
5  else
6      Let R_dc be the set of triplets t s.t. t, t̄ ∈ R ;
7      R' ← R − R_dc;
8      Let G' be the Aho graph for R';
10     if G' is connected then C_PC ← v(G) ;
11     else
12         C_PC ← CC(G');
14         repeat
16             foreach ab|c ∈ R_dc do
18                 if a, b ∈ c_i and c ∈ C_j (with C_i, C_j ∈ C_PC and i ≠ j) then
20                     Build G'_i the Aho graph for R'|v(C_i) ;
21                     if G'_i is connected then C_PC ← (C_PC − {C_i}) ∪ v(C_i) ;
23                     else C_PC ← (C_PC − {C_i}) ∪ CC(G'_i)
24         until C_PC no longer changes;
25 foreach C_i ∈ C_PC do
26     if (R|v(C_i)) = ∅ then T_i ← star tree on v(C_i) ;
27     else T_i ← PhySIC_PC(v(C_i), R|v(C_i));
29 return the tree made of a root node connected to T_1, T_2, ..., T_|C_PC| ;
```

Algorithm 12: Details of the $Build_{PC}$ subroutine taking a set S of taxa and a set $\mathcal{R}$ of triplets on S as input.

```
Algorithm Build_PC (S, R)
if S contains less than 3 taxa then return the trivial tree on S;
Let G denote the Aho graph for R;
if G has only one connected component then
    return the star tree on L(R)
else
    C_PC ← CC(G) ;
    foreach C_i ∈ C_PC do
        if (R|v(C_i)) = ∅ then T_i ← star tree on v(C_i) ;
        else T_i ← Build_PC (v(C_i), R|v(C_i))
    return the tree made of a root node connected to T_1, T_2, ..., T_|C_PC| ;
```

Algorithm 13: Details of the $Check_{PI}$ subroutine taking a tree T and a set $\mathcal{R}$ of triplets on S as input. $S(T)$ denotes (complete) subtrees connected to the root of T, *i.e.*, the subtrees corresponding to the largest clades under the root of T.

```
Algorithm Check_PI (T, R)
if T is made of a single leaf then return T ;
Let G be the Aho graph for R;
if |CC(G)| = 1 then return "error, R is incompatible";
repeat
    foreach T_i ∈ S(T) do
        Let G_i be the Aho graph for R|L(T_i) ;
        foreach T_j ∈ S(T) s.t. T_i ≠ T_j do
            Build G_ij from G_i and R|(L(T_i) ∪ L(T_j)) ;
            if G_ij is not connected then
                Collapse the branch between the root of T and T_i
until no branch of T is collapsed;
foreach T_i ∈ S(T) do
    T'_i ← Check_PI (T_i, R|L(T_i))
return the tree made of a root node connected to T'_1, T'_2, ..., T'_|S(T)|
```

Algorithm 14: Details of the $PhySIC_{PI}$ subroutine taking a tree T and a forest $\mathcal{F}$ as input.

```
Algorithm PhySIC_PI (T, F)
T_PI ← T;
repeat
    R_PI ← R(T_PI, F);
    T_PI ← Check_PI (T_PI, R_PI)
until T_PI no longer changes;
return T_PI
```

A.2 Outline of main *PhySIC_IST* subroutines

Algorithm 15: Procedure ensuring that the tree T does not contain any branch contradicting triplets in the set $\mathcal{R}$.

```
1  Algorithm Check_PC (T, R, R_dc)
2  R_T ← R(T);
3  foreach r_T ∈ R_T do
4      if !(r_T ∉ R_dc and r̄_T ∉ R) then
5          Let [u, v] be the path of T corresponding to the internal branch of r_T;
6          Mark all branches of the path [u, v];
7  Remove from T branches that have been marked above;
8  return T;
```

Algorithm 16: Procedure that increments the supports of edges and nodes of T, within the region where the taxon l can be inserted without contradicting the tree T_i.

```
1  Algorithm support(T_i, T, l)
2  T_i' ← T_i|(L(T) ∪ {l});
3  f_i' ← the father of l in T_i';
4  C_i' ← the sons of f_i' (other than l) in T_i';
5  I ← L(T_i') - L(subTree(f_i'));
6  foreach s ∈ C_i' do
7      C ← C ∪ lca_T(subTree(s)) // i.e. the lca in T of the taxa present in subTree(s);
8  f ← the lowest node in T s.t. ∀s ∈ C, L(subTree(s)) ⊆ L(subTree(f)) and
   L(subTree(f)) ∩ I ≠ ∅;
9  M ← {m ∈ children(f) s.t. L(subTree(m)) ∩ I = ∅ };
10 suppOn(f) ++;
11 foreach m ∈ M do
12     foreach u ∈ subTree(m) do
13         if ∄s ∈ C s.t. L(subTree(u)) ⊂ L(subTree(s)) then
14             suppAbv(u) ++;
15             if u is not a leaf then
16                 suppOn(u) ++;
```

Algorithm 17: Procedure computing the CIC value of a tree T, when source tree taxa contain n leaves.

```
1  Algorithm CIC (T, n)
2  nr_{T,n} ← 1;
3  Let I the set of internal nodes of T;
4  foreach u ∈ I do
5      c ← |children(u)|;
6      for j in [2, c] do
7          nr_{T,n} ← (nr_{T,n} * (2 * j − 3));
8  max ← n − |L(T)|; j ← |L(T)|;
9  for k in [1, max] do
10     nr_{T,n} ← (nr_{T,n} * (2 * j − 1));
11     j ← j + 1;
12 nr_n ← (2n − 1)!!
13 return − log (nr_{T,n}/nr_n)
```

Algorithm 18: Procedure returning true if inserting a taxon l in a tree T leads to a tree T' with a greater CIC value, while satisfying PC and PI (the $Check_{PC}$ and $Check_{PI}$ subroutines ensure it).

```
1  Algorithm betterCIC(T, n, R, R_dc, u, l, above)
2  if above then
3      T' ← T with l inserted above u;
4  else
5      T' ← T with l inserted on u;
6  T' ← Check_PC (T',R,R_dc);
7  T' ← Check_PI (T',R);
8  if CIC(T', n) > CIC(T, n) then
9      return true;
10 else
11     return false;
```

Algorithm 19: Details of the `roundIns` procedure. This function tries to insert a given taxa l in the backbone tree T. The insertion is performed only if the source trees containing l all indicate the same zone to graft l and the insertion does not decrease the CIC of the built supertree.

```
1  Algorithm roundIns(T, F, R, R_dc, l, all, summary)
2  change ← false ; n ← |L(F)| ;
3  foreach u ∈ nodes(T) do
4      suppAbv(u) ← 0; suppOn(u) ← 0;
5  F' ← {T_j ∈ F such that l ∈ L(T_j) and L(T_j) ∩ L(T) > 2};
6  foreach T_j ∈ F' do
7      support(T_j, T, l);
8  nbMaxAbv ← 0; nbMaxOn ← 0 ;
9  suppMax ← max_{u∈nodes(T)}( max(suppAbv(u), suppOn(u)) );
10 if (suppMax = |F'| or all = false) then
11     foreach u ∈ nodes(T) do
12         if (suppAbv(u) = suppMax) then nbMaxAbv ++; u_abv ← u ;
13         if (suppOn(u) = suppMax) then nbMaxOn ++; u_on ← u ;
14     if (nbMaxAbv = 1 and nbMaxOn = 0) then
15         if (all = true) or (betterCIC(T, n, R, R_dc, u_abv, l, true)) then
16             T ← T with l inserted above node u_abv;
17             change ← true;
18     else if (nbMaxAbv = 0 and nbMaxOn = 1) then
19         if (all = true) or (betterCIC(T, n, R, R_dc, u_on, l, false)) then
20             T ← T with l inserted on node u_on;
21             change ← true;
22     else if (nbMaxOn = 1 and nbMaxAbv > 0 and summary = true) then
23         AbvMax ← {u ∈ nodes(T) such that suppAbv(u) = suppMax};
24         if AbvMax ⊆ Children(u_on) ∪ {u_on} then
25             if (all = true) or (betterCIC(T, n, R, R_dc, u_on, l, false)) then
26                 T ← T with l inserted on u_on;
27                 change ← true;
28 if (change and suppMax < |F'|) then
29     T ← Check_PC (T,R, R_dc)
30 return change;
```

Algorithm 20: Details of the `insertion` procedure. Taxa not yet inserted in the backbone tree are considered in decreasing priority order. Each time a taxon can be inserted (which is decided by the *roundIns* procedure), the taxa with higher priority (that are not yet inserted) are reconsidered. $Check_{PC}$ and $Check_{PI}$ ensure that the output tree still satisfies PI and PC properties.

```
1  Algorithm insertion(T, F, R, R_dc, priorityList, all, summary)
2  i ← 1 ;
3  while i ≤ size(priorityList) do
4      Let l be the i^th element in priorityList;
5      if roundIns(T, F, R, l, all, summary ) then
6          remove l from priorityList ; i ← 1
7      i ← i + 1 ;
8  T ← Check_PC (T,R,R_dc) ; T ← Check_PI (T,R);
```

Algorithm 21: Details of the `PhySIC_IST(`$\mathcal{F}$`)` algorithm. After computing $\mathcal{R}$, $\mathcal{R}_{dc}$, the priority list and the starting backbone tree T, the insertions of taxa are done in four successive steps. These four steps differ on whether a maximum or maximal support is required to insert a taxon (first boolean parameter of the *insertion* algorithm) and whether insertions can temporarily contradict some source trees (second boolean parameter of the *insertion* algorithm).

1 `Algorithm PhySIC_IST(`$\mathcal{F}$`)`
2 $\mathcal{R} \leftarrow \mathcal{R}(\mathcal{F})$;
3 Let $\mathcal{R}_{dc}$ be the set of triplets $r : r, \bar{r} \in \mathcal{R}$;
4 $priorityList \leftarrow orderList(L(\mathcal{F}), \mathcal{R})$;
5 Remove the first two leaves, called a and b, from $priorityList$;
6 Let T be the rooted tree composed of a root node connected to two leaves a and b;
7 insertion$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, priorityList, true, false)$;
8 insertion$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, priorityList, true, true)$;
9 insertion$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, priorityList, false, false)$;
10 insertion$(T, \mathcal{F}, \mathcal{R}, \mathcal{R}_{dc}, priorityList, false, true)$;

A.3 Supplementary materials of Section 4.4

Species	Accession No.	Origin
Aegilops longissima	PI 330486	Unknown
Aegilops longissima	PI 604110	Israel
Aegilops speltoides var. speltoides	PI 449338	Israel
Aegilops speltoides var. ligustica	PI 560528	Turkey
Aegilops tauschii	PI 603233	Azerbaijan
Aegilops tauschii	PI 603254	Iran
Agropyron mongolicum	PI 499391	China
Agropyron mongolicum	PI 598482	Unknown
Australopyrum retrofractum	PI 531553	Australia
Australopyrum retrofractum	PI 533013	Australia
Brachypodium sp.*	PI 317418	Afghanistan
Dasypyrum villosum	PI 251477	Turkey
Dasypyrum villosum	PI 598396	Greece
Eremopyrum bonaepartis	PI 203442	Turkey
Eremopyrum triticeum	PI 502364	Russia
Henrardia persica	PI 401347	Iran
Henrardia persica	PI 577112	Turkey
Heteranthelium piliferum	PI 401354	Iran
Hordeum bogdanii	PI 499498	China
Hordeum marinum subsp. marinum	PI 401364	Iran
Hordeum vulgare subsp. spontaneum	PI 282582	Israel
Hordeum vulgare subsp. spontaneum	PI 282585	Israel
Psathyrostachys juncea	PI 314668	Former URSS
Psathyrostachys juncea	PI 75737	Former URSS
Pseudoroegneria libanotica	PI 228389	Iran
Pseudoroegneria libanotica	PI 401274	Iran
Pseudoroegneria spicata	PI 563870	United States
Secale cereale	PI 561793	Turkey
Taeniatherum caput-medusae	PI 577708	Turkey
Taeniatherum caput-medusae	PI 598389	Turkey
Triticum monococcum subsp. aegilopoides	PI 272519	Hungary
Triticum monococcum subsp. aegilopoides	PI 427990	Lebanon

Table A.1: **Species, accession numbers in the USDA database, and geographic origin of Triticeae.** *This species is incorrectly identified in the USDA database as Eremopyrum triticeum.

Locus	Alignment length (bp)	Genomic location	Relative distance to the centromere	Average evolutionary rate	Shape parameter α	Proportion of variable sites	Triplet distance*
LOC_Os01g01790	860	Chr. 3S, Tel.	0.976	1.687	0.379	0.313	0.168
LOC_Os01g09300	861	Chr. 3S, Tel.	0.722	0.883	0.730	0.285	0.230
LOC_Os01g11070	1050	Chr. 3S, Cen.	0.652	1.033	0.386	0.305	0.270
LOC_Os01g13200	897	Chr. 3S, Cen.	0.568	0.659	0.282	0.220	0.209
LOC_Os01g19470	942	Chr. 3S, Cen.	0.352	0.906	0.686	0.321	0.143
LOC_Os01g21160	1017	Chr. 3S, Cen.	0.307	1.596	0.489	0.393	0.207
LOC_Os01g24680	1014	Chr. 3S, Cen.	0.184	0.875	0.266	0.260	0.374
LOC_Os01g37560	1005	Chr. 3L, Cen.	0.160	1.060	0.403	0.310	0.235
LOC_Os01g39310	945	Chr. 3L, Cen.	0.202	0.989	0.335	0.290	0.150
LOC_Os01g48720	939	Chr. 3L, Cen.	0.417	1.252	0.845	0.399	0.196
LOC_Os01g53720	1101	Chr. 3L, Cen.	0.526	0.921	0.534	0.320	0.144
LOC_Os01g55530	1068	Chr. 3L, Cen.	0.567	0.890	0.819	0.309	0.151
LOC_Os01g56630	915	Chr. 3L, Cen.	0.592	0.731	0.504	0.312	0.179
LOC_Os01g60230	999	Chr. 3L, Cen.	0.673	0.929	0.363	0.283	0.216
LOC_Os01g61720	935	Chr. 3L, Tel.	0.705	1.131	0.400	0.328	0.257
LOC_Os01g62900	951	Chr. 3L, Tel.	0.732	0.897	0.240	0.257	0.113
LOC_Os01g67220	1101	Chr. 3L, Tel.	0.827	1.303	0.798	0.322	0.231
LOC_Os01g68770	998	Chr. 3L, Tel.	0.862	1.307	0.633	0.278	0.258
LOC_Os01g70670	883	Chr. 3L, Tel.	0.898	0.899	0.381	0.310	0.156
LOC_Os01g72220	1131	Chr. 3L, Tel.	0.933	0.974	0.261	0.255	0.278
LOC_Os01g73790	966	Chr. 3L, Tel.	0.965	0.850	0.684	0.180	0.242
eIFiso4E	630	Chr. 1L, Cen.	NA	0.952	0.013	0.128	0.224
CRTISO	529	Chr. 4L	NA	1.165	0.136	0.163	0.477
PinA	456	Chr. 5S	NA	1.375	0.267	0.189	0.394
PinB	453	Chr. 5S	NA	2.411	0.258	0.218	0.289
PSY2	461	NA	NA	0.978	0.332	0.150	0.318
MATK	1545	Chloroplast	NA	0.462	0.374	0.177	0.127

Table A.2: **Relevant phylogenetic and genomic parameters for all sequenced loci.**
Chr.: chromosome; S: short arm; L: long arm; Tel.: telomere; Cen.: centromere; NA: not available. Loci on chromosome 3 were considered telomeric when located at a relative distance from the centromere greater than 70% and centromeric otherwise. * The triplet distance of each gene is calculated relative to the supermatrix tree (see Equation 4.1 in the main text)

Species	Accession No.	Incongruent triplets
Pseudoroegneria libanotica	PI 228389	113
Pseudoroegneria libanotica	PI 401274	82
Pseudoroegneria spicata	PI 563870	80
Hordeum bogdanii	PI 499498	63
Hordeum vulgare subsp. spontaneum	PI 282585	63
Hordeum vulgare subsp. spontaneum	PI 282582	62
Hordeum marinum subsp. marinum	PI 401364	51
Australopyrum retrofractum	PI 531553	46
Australopyrum retrofractum	PI 533013	40
Eremopyrum bonaepartis	PI 203442	39
Taeniatherum caput-medusae	PI 577708	33
Agropyron mongolicum	PI 598482	32
Dasypyrum villosum	PI 598396	30
Taeniatherum caput-medusae	PI 598389	30
Aegilops speltoides var. ligustica	PI 560528	29
Henrardia persica	PI 401347	29
Secale cereale	PI 561793	29
Henrardia persica	PI 577112	27
Triticum monococcum subsp. aegilopoides	PI 272519	27
Aegilops tauschii	PI 603233	24
Eremopyrum triticeum	PI 502364	24
Agropyron mongolicum	PI 499391	21
Dasypyrum villosum	PI 251477	21
Aegilops tauschii	PI 603254	20
Aegilops speltoides var. speltoides	PI 449338	19
Triticum monococcum subsp. aegilopoides	PI 427990	19
Aegilops longissima	PI 330486	17
Heteranthelium piliferum	PI 401354	16
Aegilops longissima	PI 604110	15
Psathyrostachys juncea	PI 314668	0
Psathyrostachys juncea	PI 75737	0

Table A.3: **Number of incongruent, strongly rejected triplets per accession. Incongruent triplets were calculated between individual loci and the supermatrix tree**. Rows are sorted in decreasing number of incongruent triplets.

Clade triplet	Incongruent triplets	Relative proximity among clades
IIA, IIB | V	119	Distantly related
IIA, IIB | IIIB	68	Distantly related
IIIA, IIIB | V	30	Closely related
IIA, IIB | IV	29	Distantly related
IIA, IIB | IIIA	22	Distantly related
V, V | V	18	Adjacent
IV, V | IIIB	18	Closely related
V, IIIA | IIB	14	Distantly related
IIIA, IV | IIB	12	Distantly related
IIIB, IV | IIB	11	Distantly related
IIIA, IIIB | IV	8	Closely related
IIIB, IIIB | V	5	Closely related
IIIB, IIIB | IIIA	5	Adjacent
V, IV | IIIA	2	Closely related
IIIB, IIIB | IV	2	Closely related
IIIB, IIIA | V	2	Closely related
V, IV | IIIB	1	Closely related
IIIB, IIIB | IIA	1	Distantly related

Table A.4: **Number of incongruent, strongly rejected triplets, pooled by clades. Clades are named as depicted in Figure 4.19**. Incongruent triplets are calculated between individual loci and the supermatrix tree. Rows are sorted in decreasing number of incongruent triplets.

List of Figures

List of Tables

Bibliography

Adams, E. 1972. **Consensus techniques and the comparison of taxonomic trees**. Syst Zool 21:390–397. 43, 50

Adams, E. 1986. **N-trees as nestings: complexity, similarity, and consensus**. J Classif 3:299–317. 51

Aho, A., J. Hopcroft, and J. Ullman. 1974. **The Design and Analysis of Computer Algorithms**. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 142

Aho, A. V., Y. Sagiv, T. G. Szymanski, and J. D. Ullman. 1981. **Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions**. SIAM J. Comp. 10:405–421. 38, 43, 53, 57, 135, 169, 171, 172

Akaike, H. 1974. **A new look at the statistical model identification**. IEEE Transactions on Automatic Control 19:716–723. 14

Akhunov, E. D., A. R. Akhunova, A. M. Linkiewicz, J. Dubcovsky, D. Hummel, G. Lazo, S. M. Chao, O. D. Anderson, J. David, L. L. Qi, B. Echalier, B. S. Gill, M. J. P. Gustafson, M. La Rota, M. E. Sorrells, D. S. Zhang, H. T. Nguyen, V. Kalavacharla, K. Hossain, S. F. Kianian, J. H. Peng, N. L. V. Lapitan, E. J. Wennerlind, V. Nduati, J. A. Anderson, D. Sidhu, K. S. Gill, P. E. McGuire, C. O. Qualset, and J. Dvorak. 2003a. **Synteny perturbations between wheat homoeologous chromosomes caused by locus duplications and deletions correlate with recombination rates**. Proceedings of the National Academy of Sciences of the United States of America 100:10836–10841. 134

Akhunov, E. D., A. W. Goodyear, S. Geng, L. L. Qi, B. Echalier, B. S. Gill, Miftahudin, J. P. Gustafson, G. Lazo, S. M. Chao, O. D. Anderson, A. M. Linkiewicz, J. Dubcovsky, M. La Rota, M. E. Sorrells, D. S. Zhang, H. T. Nguyen, V. Kalavacharla, K. Hossain, S. F. Kianian, J. H. Peng, N. L. V. Lapitan, J. L. Gonzalez-Hernandeiz, J. A. Anderson, D. W. Choi, T. J. Close, M. Dilbirligi, K. S. Gill, M. K. Walker-Simmons, C. Steber, P. E. McGuire, C. O. Qualset, and J. Dvorak. 2003b. **The organization and rate of evolution of wheat genomes are correlated with recombination rates along chromosome arms**. Genome Research 13:753–763. 134

Amir, A. and D. Keselman. 1997. **Maximum agreement subtree in a set of evolutionary trees : metrics and efficient algorithm**. SIAM J on Comp 26:1656–1669. 49, 80

Arvestad, L., A. Berglund, J. Lagergren, and B. Sennblad. 2003. **Bayesian gene/species tree reconciliation and orthology analysis using MCMC**. Bioinformatics 19 Suppl 1:7—15. 138

Bandelt, H.-J. and A. Dress. 1986. **Reconstructing the shape of a tree from observed dissimilarity data**. Adv in appl math 7:309–343. 42, 86

Bapteste, E., E. Susko, J. Leigh, D. MacLeod, R. L. Charlebois, and W. F. Doolittle. 2005. **Do orthologous gene phylogenies really support tree-thinking?** BMC Evol Biol 5. 27

Bapteste, E., E. Susko, J. Leigh, I. Ruiz-Trillo, J. Bucknam, and W. F. Doolittle. 2008. **Alternative methods for concatenation of core genes indicate a lack of resolution in deep nodes of the prokaryotic phylogeny**. Mol Biol Evol 25:83–91. 138, 154, 157, 164

Barrett, M., D. M. J., and E. Sober. 1991. **Against consensus.** Syst Biol 40:486–493. 32, 44, 45

Barry, D. and J. A. Hartigan. 1987. **Statistical analysis of hominoid molecular evolution.** Statist Sci 2:191–207. 15

Barthélemy, J. and A. Guenoche. 1991. **Trees and proximity representations**. Wiley (Chichester, New York). 38

Barthélemy, J. P. and F. R. McMorris. 1986. **The median procedure for n-trees.** J Classif 3:329–334. 45, 46

Baum, B. R. 1992. **Combining trees as a way of combining data sets for phylogenetic inference, and the desirability of combining gene trees**. Taxon 41:3–10. 38, 68, 122, 169

Baum, B. R. and M. A. Ragan. 2004. **The MRP method**. Pages 17–34 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 4. Kluwer. 38, 69, 108, 122, 138, 157

Bender, M. A. and M. Farach-Colton. 2000. **The lca problem revisited**. Pages 88–94 *in* LATIN '00: Proceedings of the 4th Latin American Symposium on Theoretical Informatics (Springer-Verlag, ed.). 103, 145

Berger-Wolf, T. 2004. **Properties of compatibility and consensus sets of phylogenetic trees**. Tech. Rep. TR-CS-2004-24 University of New Mexico. 48

Bergsten, J. 2005. **A review of long-branch attraction?** Cladistics 21:163–193. 25

Berry, V. and D. Bryant. 1999. **Faster reliable phylogenetic analysis**. Pages 59–68 *in* RECOMB '99: Proceedings of the third annual international conference on Computational molecular biology ACM, New York, NY, USA. 54

Berry, V. and O. Gascuel. 2000. **Inferring evolutionary trees with strong combinatorial confidence**. Theoretical Computer Science 240:271—298. 54

Berry, V. and F. Nicolas. 2004. **Maximum agreement and compatible supertrees**. Pages 205–219 *in* Proceedings of CPM (S. C. Sahinalp, S. Muthukrishnan, and U. Dogrusoz, eds.) vol. 3109 of *LNCS*. 38, 80, 95, 169

Berry, V. and F. Nicolas. 2006. **Improved parameterized complexity of the maximum agreement subtree and maximum compatible tree problems**. IEEE/ACM Trans. Comput. Biol. Bioinformatics 3:289–302. 49, 50, 142, 143

Berry, V. and F. Nicolas. 2007. **Maximum agreement and compatible supertrees**. JDA 5:564–591. 38, 67, 80, 95, 169

Berry, V. and C. Semple. 2006. **Fast computation of supertrees for compatible phylogenies with nested taxa**. Syst. Biol. 55:270–288. 59, 145

Bevan, R. B., D. Bryant, and B. F. Lang. 2007. **Accounting for gene rate heterogeneity in phylogenetic inference.** Syst Biol 56:194–205. 31

Beyer, W., M. Stein, T. Smith, and S. Uiam. 1974. **A molecular sequence metric and evolutionary trees.** Math Biosci 19:9–25. 16

Bininda-Edmonds, O. R. P. and A. Stamatakis. 2006. **Taxon sampling versus computational complexity and their impact on obtaining the tree of life**. Pages 77–95 *in* Reconstructing the Tree of Life: Taxonomy and Systematics of Species Rich Taxa (T. Hodkinson and J. Parnell, eds.). Systematics Association Special Series, volume 72, CRC Press., New York. 37, 158, 169

Bininda-Emonds, O. 2004a. **Trees versus characters and the supertree/supermatrix 'paradox'.** Syst. Biol. 53:342–355. 33, 34, 120

Bininda-Emonds, O., J. L. Gittleman, and A. Purvis. 1999. **Building large trees by combining phylogenetic information: a complete phylogeny of the extant carnivora (mammalia)**. Biological Reviews 74:143–175. 33, 70

Bininda-Emonds, O., K. Jones, S. Price, M. Cardillo, R. Grenyer, and A. Purvis. 2004. **Garbage in, garbage out: data issues in supertree construction**. chap. 12, Pages 267–280 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 33, 34

Bininda-Emonds, O., K. Jones, S. Price, R. Grenyer, M. Cardillo, M. Habib, A. Purvis, and J. Gittleman. 2003. **Supertrees are a necessary not-so-evil: A comment on gatesy et al.** Systematic Biology 52:724–729. 31, 33, 34

Bininda-Emonds, O. R. P. 2003. **Novel versus unsupported clades: assessing the qualitative support for clades in mrp supertrees**. Syst. Biol. 52:839–848. 32

Bininda-Emonds, O. R. P. 2004b. **The evolution of supertrees**. Trends Ecol. Evol. 19:315–322. 1, 32, 34, 37, 168

Bininda-Emonds, O. R. P. 2004c. **Phylogenetic supertrees (combining information to reveal the tree of life)** vol. 4 of *computational biology series*. Kluwer academic publishers. 38, 169

Bininda-Emonds, O. R. P. 2005. **Supertree construction in the genomic age.** Methods Enzymol 395:745–57. 34, 35, 37, 169

Bininda-Emonds, O. R. P. and H. N. Bryant. 1998. **Properties of matrix representation with parsimony analyses**. Syst. Biol. 47:497–508. 32, 70

Bininda-Emonds, O. R. P., J. L. Gittleman, and M. A. Steel. 2002. **The (super)tree of life: Procedures, problems, and prospects**. Annual Review of Ecology and Systematics 33:265–289. 34, 70

Bittner, L., C. E. Payri, A. Couloux, C. Cruaud, B. de Reviers, and F. Rousseau. 2008. **Molecular phylogeny of the dictyotales and their position within the phaeophyceae, based on nuclear, plastid and mitochondrial dna sequence data**. Molecular Phylogenetics and Evolution 49:211–226. 120

Blanquart, S. and N. Lartillot. 2006. **A bayesian compound stochastic process for modeling nonstationary and nonhomogeneous sequence evolution**. Mol. Biol. Evol. 23:2058–2071. 13

Blanquart, S. and N. Lartillot. 2008. **A site- and time-heterogeneous model of amino-acid replacement**. Mol. Biol. Evol. 25:842–858. 14

Brandley, M., A. Schmitz, and T. Reeder. 2005. **Partitioned bayesian analyses, partition choice, and the phylogenetic relationships of scincid lizards**. Systematic Biology 54:373–390. 31

Bremer, K. 1990. **Combinable component consensus.** Cladistics 6:369–372. 46, 48

Brochier, C., P. Forterre, and S. Gribaldo. 2005. **An emerging phylogenetic core of archaea: phylogenies of transcription and translation machineries converge following addition of new genome sequences**. BMC Evol Biol 5:36–36. 138, 154

Brochier, C. and H. Philippe. 2002. **Phylogeny: a non-hyperthermophilic ancestor for bacteria.** Nature 417. 160

Brown, J. and W. Doolittle. 1995. **Root of the universal tree of life based on ancient aminoacyl-tRNA synthetase gene duplications**. Proc. Natl. Acad. Sci. USA 92:2441–2445. 165

Bruno, W., N. Socci, and A. Halpern. 2000. **Weighted neighbor-joining:a likelihood-based approach to distance-based phylogeny reconstruction.** Mol Biol Evol 17:189–197. 17

Bryant, D. 1997. **Building trees, hunting for trees and comparing trees.** Ph.D. thesis University of Canterbury, Department of Math. 49, 57, 59, 78, 99

Bryant, D. 2001. **Optimal agreement supertrees**. Pages 24–31 *in* Proc. of the 1st International Conference on Biology, Informatics, and Mathematics (JOBIM 2000), LNCS vol. 2066 Springer. 67, 68

Bryant, D. 2003. **A classification of consensus methods for phylogenetics**. Pages 163–184 *in* Bioconsensus (M. Janowitz, F. Lapointe, F. McMorris, and F. Roberts, eds.). DIMACS-AMS Series, Providence, RI, USA. 45, 47, 49, 51, 52, 54, 55, 56, 76, 163

Bryant, D. and M. Steel. 1995. **Extension operations on sets of leaf-labelled trees**. Adv. Appl. Math. 16:425–453. 148, 149

Bryant, D. and P. Waddell. 1998. **Rapid evaluation of least-squares and minimum-evolution criteria on phylogenetic trees.** Mol Biol Evol 15:1346–1359. 16, 17

Bulmer, M. 1991. **Use of the method of generalized least squares in reconstructing phylogenies from sequence data.** Mol Biol Evol 8:868–883. 16

Buneman, P. 1971. **The recovery of trees from measures of dissimilarity.** Pages 387–395 *in* Mathematics in Archeological and Historical Sciences (D. Kendall and T. P., eds.). Edinburgh University Press. 16, 41

Camin, J. H. and R. R. Sokal. 1974. **A method for deducing branching sequences in phylogeny**. Evolution 19:311–326. 9

Cavalli-Sforza, L. and A. Edwards. 1967. **Phylogenetic analysis. models and estimation procedures.** Am J Hum Genet 19:233–57. 16

Cavender, J. 1978. **Taxonomy with confidence.** Math Biosci 40:271–280. 10

Charlesworth, B. 2009. **Effective population size and patterns of molecular evolution and variation**. Nature Reviews Genetics 10:195–205. 124, 134

Chauve, C., J. P. Doyon, and N. El-Mabrouk. 2008. **Gene family evolution by duplication, speciation, and loss**. J Comput Biol 15:1043—1062. 82, 137, 164

Chauve, C. and N. El-Mabrouk. 2009. **New perspectives on gene family evolution: Losses in reconciliation and a link with supertrees**. Pages 46–58 *in* RECOMB (S. Batzoglou, ed.) vol. 5541 of *Lecture Notes in Computer Science* Springer. 82, 137, 164

Chen, D., L. Diao, O. Eulenstein, D. Fernandez-Baca, and M. Sanderson. 2003. **Flipping: a supertree construction method**. Pages 135–160 *in* Bioconsensus (M. Janowitz, F. Lapointe, F. McMorris, and F. Roberts, eds.) vol. 61. DIMACS: Series in Discrete Mathematics and Theoretical Computer Science, Providence, RI, USA. 71, 72

Chen, D., O. Eulenstein, D. Fernández-Baca, and M. Sanderson. 2002. **Supertrees by flipping**. Pages 391–400 *in* Computing and Combinatorics, 8th Annual International Conference, COCOON 2002, Singapore, August 15-17, 2002, Proceedings Lecture Notes in Computer Science Springer. 71

Chen, D., O. Eulenstein, D. Fernandez-Baca, and M. Sanderson. 2006. **Minimum-flip supertrees: Complexity and algorithms**. IEEE/ACM Trans. Comput. Biol. Bioinformatics 3:165–173. 38, 72

Chen, K., D. Durand, and M. Farach-Colton. 2000. **Notung: a program for dating gene duplications and optimizing gene family trees**. J Comput Biol 7:429—444. 82, 138, 164

Chippindale, P. and J. Wiens. 1994. **Weighting, partitioning, and combining characters in phylogenetic analysis**. Syst. Biol. 43:278–287. 32

Ciccarelli, F. D., T. Doerks, C. von Mering, C. J. Creevey, B. Snel, and P. Bork. 2006. **Toward automatic reconstruction of a highly resolved tree of life**. Science 311:1283–1287. 138, 154

Comas, I., A. Moya, and F. Gonzalez-Candelas. 2007. **From phylogenetics to phylogenomics: the evolutionary relationships of insect endosymbiotic gamma-proteobacteria as a test case**. Systematic Biology 56:1–16. 120

Constantinescu, M. and D. Sankoff. 2003. **An efficient algorithm for supertrees.** J. Classif. 12:101–112. 59

Cotton, J. A. and R. D. M. Page. 2005. **Rates and patterns of gene duplication and loss in the human genome**. Proceedings of the Royal Society B: Biological Sciences 272:277–283. 25, 137

Cotton, J. A., C. S. C. Slater, and M. Wilkinson. 2006. **Discriminating supported and unsupported relationships in supertrees using triplets.** Syst. Biol. 55:345–350. 38, 70, 169

Cotton, J. A. and M. Wilkinson. 2007. **Majority-rule supertrees.** Syst Biol 56:445–452. 78, 79

Creevey, C. J., D. A. Fitzpatrick, G. K. Philip, R. J. Kinsella, M. J. O'Connell, M. M. Pentony, S. A. Travers, M. Wilkinson, and J. O. Mcinerney. 2004. **Does a tree-like phylogeny only exist at the tips in the prokaryotes?** Proceedings: Biological Sciences 271:2551–2558. 77

Creevey, C. J. and J. O. McInerney. 2005. **Clann: investigating phylogenetic information through supertree analyses**. Bioinformatics 21:390–392. 77, 78, 122

Criscuolo, A., V. Berry, E. J. P. Douzery, and O. Gascuel. 2006. **SDM: a fast distance-based approach for (super)tree building in phylogenomics**. Syst. Biol. 55:740–755. 31, 77, 107, 122, 124

Daniel, P. 2004. **Supertree methods: some new approaches**. Master's thesis University of Canterbury. 93

Daniel, P. and C. Semple. 2004. **Supertree algorithms for nested taxa**. chap. 7, Pages 151–171 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 59

Daubin, V., M. Gouy, , and G. Perrière. 2002. **A phylogenomic approach to bacterial phylogeny: Evidence of a core of genes sharing a common history**. Genome Res 12:1080–1090. 34

Davies, T. J., T. G. Barraclough, M. W. Chase, P. S. Soltis, D. E. Soltis, and V. Savolainen. 2004. **Darwin's abominable mystery: Insights from a supertree of angiosperms**. Proc. Natl. Acad. Sci. USA 101:1904–1909. 34

Day, W. 1987. **Computational complexity of inferring phylogenies from dissimilarity matrices.** Bull Math Biol 49:461–467. 16

Day, W. 1996. **Complexity theory: An introduction for practitioners of classification**. Pages 199–233 *in* Clustering and Classification, (P. Arabie and L. Hubert, eds.). World Scientific Publishing Co. Inc. 16

Day, W., S. Johnson, and D. Sanko. 1986. **The computational complexity of inferring rooted phylogenies by parsimony.** Math Biosci 81:33–42. 10

Day, W. and D. Sankoff. 1986. **Computational complexity of inferring phylogenies by compatibility**. Syst. Zool. 35:224–229. 49

Dayhoff, M. O., R. M. Schwartz, and B. C. Orcutt. 1978. **A model of evolutionary change in proteins**. Atlas of protein sequence and structure 5:345–351. 14

De Queiroz, A. 1993. **For consensus (sometimes)**. Syst Biol 42:13429–13434. 46

Degnan, J. H. and N. A. Rosenberg. 2006. **Discordance of species trees with their most likely gene trees**. PLoS Genetics 2:e68. 134

Degnan, J. H. and N. A. Rosenberg. 2009. **Gene tree discordance, phylogenetic inference and the multispecies coalescent**. Trends in Ecology & Evolution 24:332–340. 119, 134

Dekker, M. C. 1986. **Reconstruction methods for derivation trees**. Master's thesis University of Amsterdam. 86

Delport, W., K. Scheffler, and C. Seoighe. 2009. **Models of coding sequence evolution**. Brief Bioinform 10:97–109. 14

Delsuc, F., H. Brinkmann, and H. Philippe. 2005. **Phylogenomics and the reconstruction of the tree of life**. Nature Reviews Genetics 6:361–375. 24

Denis, F. and O. Gascuel. 2003. **On the consistency of the minimum evolution principle of phylogenetic inference.** Discrete appl math 127:63–77. 17

Desper, R. and O. Gascuel. 2002. **Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle.** J Comput Biol 9:687–705. 17

Dong, J. and D. Fernandez-Baca. 2009. **Properties of Majority-Rule Supertrees**. Syst Biol 58:360–367. 46, 79

Douady, C., F. Delsuc, Y. Boucher, W. F. Doolittle, and E. Douzery. 2003. **Comparison of bayesian and maximum likelihood bootstrap measures of phylogenetic reliability.** Mol. Biol. Evol. 20:248–254. 22, 125

Doyle, J. 1992. **Gene trees and species trees: molecular systematics as one-character taxonomy.** Syst. Bot. 17:144–163. 26

Driskell, A. C., C. Ané, J. G. Burleigh, M. M. McMahon, B. C. O'meara, and M. J. Sanderson. 2004. **Prospects for building the tree of life from large sequence databases.** Science 306:1172–1174. 30

Dujon, B., D. Sherman, G. Fischer, P. Durrens, S. Casaregola, I. Lafontaine, J. De Montigny, C. Marck, C. Neuvéglise, E. Talla, N. Goffard, L. Frangeul, M. Aigle, V. Anthouard, A. Babour, V. Barbe, S. Barnay, S. Blanchin, J. M. Beckerich, E. Beyne, C. Bleykasten, A. Boisramé, J. Boyer, L. Cattolico, F. Confanioleri, A. De Daruvar, L. Despons, E. Fabre, C. Fairhead, H. Ferry-Dumazet, A. Groppi, F. Hantraye, C. Hennequin, N. Jauniaux, P. Joyet, R. Kachouri, A. Kerrest, R. Koszul, M. Lemaire, I. Lesur, L. Ma, H. Muller, J. M. Nicaud, M. Nikolski, S. Oztas, O. Ozier-Kalogeropoulos, S. Pellenz, S. Potier, G. F. Richard, M. L. Straub, A. Suleau, D. Swennen, F. Tekaia, M. Wésolowski-Louvel, E. Westhof, B. Wirth, M. Zeniou-Meyer, I. Zivanovic, M. Bolotin-Fukuhara, A. Thierry, C. Bouchier, B. Caudron, C. Scarpelli, C. Gaillardin, J. Weissenbach, P. Wincker, and J. L. Souciet. 2004. **Genome evolution in yeasts.** Nature 430:35–44. 25

Dunn, C. W., A. Hejnol, D. Q. Matus, K. Pang, W. E. Browne, S. A. Smith, E. Seaver, G. W. Rouse, M. Obst, G. D. Edgecombe, M. V. Sorensen, S. H. D. Haddock, A. Schmidt-Rhaesa, A. Okusu, R. M. Kristensen, W. C. Wheeler, M. Q. Martindale, and G. Giribet. 2008. **Broad phylogenomic sampling improves resolution of the animal tree of life**. Nature 452:745–U5. 117

Durand, D., B. Halldórsson, and B. Vernot. 2006. **A hybrid micro-macroevolutionary approach to gene tree reconstruction**. J. Comput. Biol. 13:320—335. 26, 138

Dutheil, J., S. Gaillard, E. Bazin, S. Glemin, V. Ranwez, N. Galtier, and K. Belkhir. 2006. **Bio++: a set of c++ libraries for sequence analysis, phylogenetics, molecular evolution and population genetics.** BMC Bioinformatics. 7:188. 106, 157, 164

Dvorak, J. and E. D. Akhunov. 2005. **Tempos of gene locus deletions and duplications and their relationship to recombination rate during diploid and polyploid evolution in the aegilops-triticum alliance**. Genetics 171:323–332. 134

Dvorak, J., M. C. Luo, and Z. L. Yang. 1998. **Restriction fragment length polymorphism and divergence in the genomic regions of high and low recombination in self-fertilizing and cross-fertilizing aegilops species**. Genetics 148:423–434. 134

Edwards, A. W. F. and L. L. Cavalli-Sforza. 1964. **Reconstruction of evolutionary trees**. Pages 64–76 *in* Phenetic and Phylogenetic Classification (V. H. Heywood and J. Mcneill, eds.). Systematics Association Publ., No. 6, London. 18

Efron, ., E. Halloran, and S. Holmes. 1996. **Bootstrap confidence levels for phylogenetic trees.** Proc Natl Acad Sci USA 93:13429–13434. 22

Efron, B. 1979. **Bootstrap methods: Another look at the jackknife**. Ann Stat 7:1–26. 21

Eichler, E. E. and D. Sankoff. 2003. **Structural Dynamics of Eukaryotic Chromosome Evolution**. Science 301:793–797. 25

Emonds, B. O. R. and M. J. Sanderson. 2001. **Assessment of the accuracy of matrix representation with parsimony analysis supertree construction.** Syst Biol 50:565–79. 70, 80

Erixon, P., B. Svennblad, T. Britton, and B. Oxelman. 2003. **Reliability of bayesian posterior probabilities and bootstrap frequencies in phylogenetics**. Syst. Biol. 52:665–673. 22, 125

Escobar, J., A. Cenci, C. Scornavacca, C. Guilhaumon, S. Santoni, E. Douzery, V. Ranwez, S. Glémin, and J. David. 2009. **Combining supermatrix and supertree in triticeea**. Submitted to Syst. Biol. . 2, 121, 124, 126, 127, 129, 130

Estabrook, G., F. McMorris, and A. Meachan. 1985. **Comparison of undirected phylogenetic trees based on subtree of four evolutionary units.** Syst Zool 34:193–200. 78

Eulenstein, O., D. Chen, G. Burleigh, D. Fernandez-Baca, and M. J. Sanderson. 2004. **Performance of flip supertree construction with a heuristic algorithm**. Syst. Biol. 53:299–308. 71, 72, 107

Farach, M., T. Przytycka, and M. Thorup. 1995. **Agreement of many bounded degree evolutionary trees**. Inf Proc Letters 55:297–301. 49

Farris, J. S. 1977. **Phylogenetic analysis under dollo's law**. Syst. Zool. 26:77–88. 9

Felsenstein, J. 1978. **Cases in which parsimony and compatibility methods will be positively misleading.** Syst Zool 27:401–410. 10, 25

Felsenstein, J. 1981. **Evolutionary trees from dna sequences: a maximum likelihood approach.** J Mol Evol 17:368–376. 13, 18, 19

Felsenstein, J. 1985. **Confidence limits on phylogenies: An approach using the bootstrap.** Evolution 39:783–791. 21, 22

Felsenstein, J. 2004. **Inferring Phylogenies**. Sinauer Associates. 9, 18, 21, 22

Felsenstein, J. 2005. **PHYLIP (Phylogeny Inference Package) version 3.6.** Distributed by the author. Department of Genome Sciences, University of Washington, Seattle. 10, 47, 55

Felsenstein, J. and G. A. Churchill. 1996. **A hidden markov model approach to variation among sites in rate of evolution.** Mol Biol Evol 13:93–104. 13

Felsenstein, J. and H. Kishino. 1993. **Is there something wrong with the bootstrap on phylogenies? a reply to hillis and bull**. Syst Biol 42:193–200. 22

Fernandez-Calvin, B. and J. Orellana. 1992. **Relationship between pairing frequencies and genome affinity estimations in aegilops ovata x triticum aestivum hybrid plants**. Heredity 68:165–172. 135

Fienberg, S. E. 1977. **The analysis of cross-classified categorical data.** Cambridge, MA: MIT Press. 105, 175

Finden, R. and A. Gordon. 1985. **Obtaining common pruned trees**. J of Classif 2:255–276. 49, 55

Fitch, W. M. 1971. **Toward defining the course of evolution: Minimum change for a specific tree topology**. Systematic Zoology 20:406–416. 9, 10

Fitch, W. M. and E. Margoliash. 1967. **Construction of phylogenetic trees.** Science 155:279–284. 16

Foster, P. and D. Hickey. 1999. **Compositional bias may affect both dna-based and protein-based phylogenetic reconstructions.** J. Mol. Evol. 48:284–290. 24

Fulton, T. and C. Strobeck. 2006. **Molecular phylogeny of the arctoidea (carnivora): effect of missing data on supertree and supermatrix analyses of multiple gene data sets**. Mol Phyl and Evol 41:165—181. 34, 120

Funk, V. and D. Brooks. 1990. **Phylogenetic systematics as the basis of comparative biology**. Smithsonian Institution Press, Washington. 45

Galtier, N. 2001. **Maximum-likelihood phylogenetic analysis under a covarion-like model**. Mol. Biol. Evol. 18:866–873. 13, 24

Galtier, N. 2007. **A Model of Horizontal Gene Transfer and the Bacterial Phylogeny Problem**. Syst Biol 56:633–642. 164

Galtier, N., O. Gascuel, and A. Jean-Marie. 2005. **An introduction to markov models in molecular evolution**. Pages 3–24 *in* Statistical Methods in Molecular Evolution (R. Nielsen, ed.). Spinger. 14

Galtier, N. and M. Gouy. 1998. **Inferring pattern and process: maximum-likelihood implementation of a nonhomogeneous model of dna sequence evolution for phylogenetic analysis**. Mol. Biol. Evol. 15:871–879. 13

Gascuel, O. 1997a. **Bionj: an improved version of the nj algorithm based on a simple model of sequence data.** Mol Biol Evol 14:685–695. 17

Gascuel, O. 1997b. **Concerning the nj algorithm and its unweighted version, unj.** Pages 149–170 *in* Mathematical Hierarchies and Biology (B. Mirkin, F. McMorris, S. Roberts, and R. A., eds.). American Mathematical Society, Providence, RI. 16, 17

Gascuel, O., D. Bryant, and F. Denis. 2001. **Strengths and limitations of the minimum evolution principle.** Syst Biol 50:621–627. 17

Gascuel, O. and M. Steel. 2006. **Neighbor-joining revealed.** Mol Biol Evol 23:1997–2000. 17

Gatesy, J., M. C., D. R., and H. C. 2002. **Resolution of a supertree/supermatrix paradox**. Syst. Biol. 51:652–664. 32, 33

Gatesy, J. and M. Springer. 2004. **A critique of matrix representation with parsimony supertrees**. chap. 8, Pages 369–388 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 3. Kluwer academic publishers. 32, 33, 34

Gibbs, A. and P. Keese. 1995. **In search of origins of viral genes**. Pages 76–91 *in* Molecular Basis of Virus Evolution (A. J. Gibbs, C. H. Callisher, and F. Garcia-Arena, eds.). Cambridge Univ. Press, Cambridge, England. 27

Gilks, W., S. Richardson, and D. Spiegelhalter. 1995. **Markov Chain Monte Carlo in Practice**. Chapman & Hall/CRC. 20

Gogarten, J. P., H. Kibak, P. Dittrich, L. Taiz, E. J. Bowman, B. J. Bowman, M. F. Manolson, R. J. Poole, T. Date, and T. Oshima. 1989. **Evolution of the vacuolar H+-ATPase: implications for the origin of eukaryotes**. Proceedings of the National Academy of Sciences of the United States of America 86:6661–6665. 165

Goldman, N. and Z. Yang. 1994. **A codon-based model of nucleotide substitution for protein-coding dna sequences.** Mol Biol Evol 11:725–736. 14

Goloboff, P. A. 2005. **Minority-rule supertrees? MRP, compatibility, and MinFlip may display the least frequent groups.** Cladistics 21:282–294. 38, 70, 72, 169

Goloboff, P. A., J. S. Farris, and K. C. Nixon. 2008. **Tnt, a free program for phylogenetic analysis**. Cladistics 24:774–786. 10, 69, 74

Goloboff, P. A. and D. Pol. 2002. **Semi-strict supertrees**. Cladistics 18:514–525. 32, 38, 70, 72, 73, 74, 87, 89, 163, 169

Goodman, M., J. Czelusniak, G. Moore, R.-H. A.E., and G. Matsuda. 1979. **Fitting the Gene Lineage into its Species Lineage, a Parsimony Strategy Illustrated by Cladograms Constructed from Globin Sequences**. Systematic Zoology 28:132—163. 138

Goodman, M., L. I. Grossman, and D. E. Wildman. 2005. **Moving primate genomics beyond the chimpanzee genome**. Trends Genet. 21:511–517. 95

Gordon, A. 1980. **On the assessment and comparison of classifications**. Pages 149–160 *in* Analyse de Données et Informatique (R. Tomassine, ed.). NRIA, Le Chesnay. 55, 56

Gordon, A. G. 1986. **Consensus supertrees: the synthesis of rooted trees containing overlapping sets of labelled leaves**. J. Classif. 3:335–348. 34, 38, 67, 169

Graham, R. L. and L. R. Foulds. 1982. **The unlikelihood that minimal phylogenies for a realistic biological study can be constructed in reasonable computational time.** Math Biosci 60:133–142. 69

Gray, M. W. 1992. **The endosymbiont hypothesis revisited**. Int.Rev.Cytol. 141. 160

Gribaldo, S. and P. Cammarano. 1998. **The root of the universal tree of life inferred from anciently duplicated genes encoding components of the protein-targeting machinery**. J. Mol. Evol. 47:508–516. 165

Grunewald, S., M. A. Steel, and M. S. Swenson. 2007. **Closure operations in phylogenetics**. Math Biosci 208:521–537. 43, 87, 148, 149, 170, 171

Guillemot, S. and V. Berry. 2007. **Finding a largest subset of rooted triples identifying a tree is an NP-hard task**. Tech. rep. LIRMM, Univ. Montpellier 2 (`www.lirmm.fr/~vberry/Publis/RR07010.pdf`). 90, 99, 172

Guillemot, S. and V. Berry. 2009. **Fixed-parameter tractability of the maximum agreement supertree problem**. IEEE/ACM Transactions on Computational Biology and Bioinformatics 99:274–285. 80, 81

Guillemot, S. and F. Nicolas. 2006. **Solving the maximum agreement subtree and the maximum compatible tree problems on many bounded degree trees**. Pages 165–176 *in* Combinatorial Pattern Matching, 17th Annual Symposium, CPM 2006, Barcelona, Spain, July 5-7, 2006, Proceedings (M. Lewenstein and G. Valiente, eds.) vol. 4009 of *Lecture Notes in Computer Science*. Springer. 49, 50, 80

Guindon, S. and O. Gascuel. 2003. **A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood.** Syst Biol 52:696–704. 19, 107, 121

Gusfield, D. 1991. **Efficient algorithms for inferring evolutionary trees**. Networks 21:19–28. 41, 142, 176

Hahn, M. W., M. V. Han, and S.-G. Han. 2007. **Gene family evolution across 12 drosophila genomes**. PLoS Genetics 3:e197+. 25

Hallett, M., J. Lagergren, and A. Tofigh. 2004. **Simultaneous identification of duplications and lateral transfers**. Pages 347–356 *in* RECOMB '04: Proceedings of the eighth annual international conference on Resaerch in computational molecular biology ACM, New York, NY, USA. 138, 164

Hallett, M. T. and J. Lagergren. 2000. **New algorithms for the duplication-loss model**. Pages 138–146 *in* RECOMB2000, Fourth Annual International Conference on Computational Molecular Biology. 82, 138, 164

Hallstrom, B. M., M. Kullberg, M. A. Nilsson, and A. Janke. 2007. **Phylogenomic data analyses provide evidence that xenarthra and afrotheria are sister groups**. Mol. Biol. Evol. 24:2059–2068. 117

Harel, D. and R. E. Tarjan. 1984. **Fast algorithms for finding nearest common ancestors**. SIAM J. Comput. 13:338–355. 103, 140, 145

Hasegawa, M., H. Kishino, and T. Yano. 1985a. **Dating of the human-ape splitting by a molecular clock of mitochondrial dna**. J. Mol. Evol. 22:160–174. 24

Hasegawa, M., H. Kishino, and T. Yano. 1987. **Man's place in hominoidea as inferred from molecular clocks of dna.** J Mol Evol 26:132–147. 13

Hasegawa, M., H. Kishino, and T.-A. Yano. 1985b. **Dating of the human-ape splitting by a molecular clock of mitochondrial dna**. J Mol Evol 22:160–174. 13, 121

Hastings, W. K. 1970. **Monte carlo sampling methods using markov chains and their applications**. Biometrika 57:97–109. 20

Hein, J., T. Jiang, L. Wang, and K. Zhang. 1996. **On the complexity of comparing evolutionary trees**. Discrete Appl. Math 71:153–169. 50, 80

Helfgott, D. and R. J. Mason-Gamer. 2004. **The evolution of north american elymus (triticeae, poaceae) allotetraploids: evidence from phosphoenolpyruvate carboxylase gene sequences**. Systematic Botany 29:850–861. 120

Hendy, M. and D. Penny. 1989. **A framework for the quantitative study of evolutionary trees**. Syst. Zool 38:297–309. 25

Henikoff, S. and J. G. Henikoff. 1992. **Amino acid substitution matrices from protein blocks.** Proc Natl Acad Sci USA 89:10915–10919. 14

Henz, S. R., D. H. Huson, A. F. Auch, K. Nieselt-Struwe, and S. C. Schuster. 2005. **Whole-genome prokaryotic phylogeny**. Bioinformatics 21:2329–2335. 160

Henzinger, M., V. King, and T. Warnow. 1999. **Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology**. Algorithmica 24:1–13. 59

Higdon, J. W., O. Bininda-Emonds, R. M. D. Beck, and S. H. Ferguson. 2007. **Phylogeny and divergence of the pinnipeds (carnivora: Mammalia) assessed using a multigene dataset**. BMC Evol Biol 7. 34, 120

Hillis, D. 1996. **Inferring complex phylogenies**. Nature (London) 383:130–131. 25

Hillis, D. 1998. **Taxonomic sampling, phylogenetic accuracy, and investigator bias.** Syst Biol 47:3–8. 158

Hillis, D. and J. Bull. 1993. **An empirical test of bootstrapping as a method for assessing confidence in phylogenetic analysis.** Syst Biol 42:182–192. 21

Hoang, V. T. and W.-K. Sung. 2008. **Fixed parameter polynomial time algorithms for maximum agreement and compatible supertrees**. CoRR abs/0802.2867. 80

Hoang, V. T. and W.-K. Sung. 2009. **Improved algorithms for maximum agreement and compatible supertrees**. Algorithmica . 80

Holder, M. and P. O. Lewis. 2003. **Phylogeny estimation: Traditional and bayesian approaches**. Nat Rev Genet 4:275–284. 21

Hsiao, C., N. J. Chatterton, K. H. Asay, and K. B. Jensen. 1995. **Phylogenetic relationships of the monogenomic species of the wheat tribe triticeae (poaceae), inferred from nuclear rdna (internal transcribed spacer) sequences**. Genome 38:211–223. 120, 132

Hudson, R. R. and J. A. Coyne. 2002. **Mathematical consequences of the genealogical species concept**. Evolution 56:1557–1565. 119

Huelsenbeck, J. 1997. **Is the felsenstein zone a fly trap?** Syst. Biol. 46:69–74. 25

Huelsenbeck, J., J. Bollback, and A. Levine. 2002a. **Inferring the root of a phylogenetic tree**. Syst. Biol. 51:32–43. 106

Huelsenbeck, J. P., B. Larget, R. E. Miller, and F. Ronquist. 2002b. **Potential applications and pitfalls of bayesian inference of phylogeny.** Syst Biol 51:673–688. 20, 21

Huelsenbeck, J. P. and F. Ronquist. 2001. **Mrbayes: Bayesian inference of phylogenetic trees.** Bioinformatics 17:754–755. 21, 121

Huerta-Cepas, J., H. Dopazo, J. Dopazo, and T. Gabaldón. 2007. **The human phylome**. Genome Biol. 8:R109. 27

Huson, D. H., S. M. Nettles, and T. J. Warnow. 1999. **Disk-covering, a fast-converging method for phylogenetic tree reconstruction.** J. Comput. Biol. 6:369–386. 34, 38, 169

Inagaki, Y., E. Susko, N. Fast, and A. J. Roger. 2004. **Covarion shifts cause a long-branch attraction artifact that unites microsporidia and archaebacteria in ef-1α phylogenies**. Mol. Biol. Evol. 21:1340–1349. 24

Iwabe, N., K. Kuma, M. Hasegawa, S. Osawa, and T. Miyata. 1989. **Evolutionary relationship of archaebacteria, eubacteria, and eukaryotes inferred from phylogenetic trees of duplicated genes**. Proceedings of the National Academy of Sciences of the United States of America 86:9355–9359. 165

Jain, R., M. C. Rivera, and J. A. Lake. 1999. **Horizontal gene transfer among genomes: The complexity hypothesis**. PNAS 96:3801–3806. 27

Jain, R., M. C. Rivera, J. E. Moore, and J. A. Lake. 2003. **Horizontal gene transfer accelerates genome innovation and evolution**. Mol Biol Evol 20:1598–1602. 27

Jansson, J. 2001. **On the complexity of inferring rooted evolutionary trees.** Proceedings of GRACO 2001. Electron. Notes in Disc. Math. 7:50–53. 99

Jansson, J., J. H.-K. Ng, K. Sadakane, and W.-K. Sung. 2004. **Rooted maximum agreement supertrees**. Pages 499–508 *in* LATIN (M. Farach-Colton, ed.) vol. 2976 of *Lecture Notes in Computer Science* Springer. 80

Jansson, J., J. H. K. Ng, K. Sadakane, and W.-K. Sung. 2005. **Rooted maximum agreement supertrees**. Algorithmica 43:293–307. 80, 81

Jenner, R. 2001. **Bilaterian phylogeny and uncritical recycling of morphological data sets**. Syst. Biol. 50:730–742. 33

Jin, L. and M. Nei. 1990. **Limitations of the evolutionary parsimony method of phylogenetic analysis.** Mol Biol Evol 7:82–102. 13

Jobb, G., A. von Haeseler, and K. Strimmer. 2004. **Treefinder: a powerful graphical analysis environment for molecular phylogenetics**. BMC Evol Biol 4:18 1471-2148 (Electronic) Journal Article Research Support, Non-U.S. Gov't. 117

Jones, D. T., W. R. Taylor, and J. M. Thornton. 1992. **The rapid generation of mutation data matrices from protein sequences.** Comput Appl Biosci 8:275–282. 14

Jukes, T. H. and C. R. Cantor. 1969. **Evolution of protein molecules**. Pages 21–132 *in* Mammalian protein metabolism (H. N. Munro, ed.). Academy Press, New York. 11, 13, 121

Kannan, S., T. Warnow, and S. Yooseph. 1998. **Computing the local consensus of trees**. SIAM J. Comput. 27:1695–1724. 53, 54

Kao, M., T. Lam, W. Sung, and H. Ting. 1999. **A decomposition theorem for maximum weight bipartite matchings with applications to evolutionary trees.** Pages 438–449 *in* Proc. of the 8th Ann. Europ. Symp. Alg. (ESA) Springer-Verlag, New York, NY. 49

Kao, M., T. Lam, W. Sung, and H. Ting. 2001. **An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings**. J. of Algo 40:212–233. 49

Kashyap, R. and S. Subas. 1974. **Statistical estimation of parameters in a phylogenetic tree using a dynamic model of the substitutional process.** J Theor Biol 47:75–101. 18

Keeling, P. J. and J. D. Palmer. 2008. **Horizontal gene transfer in eukaryotic evolution**. Nat Rev Genet 9:605–618. 27

Kellis, M., B. W. Birren, and E. S. Lander. 2004. **Proof and evolutionary analysis of ancient genome duplication in the yeast saccharomyces cerevisiae**. Nature 428:617–624. 26

Kellogg, E. and R. Appels. 1995. **Intraspecific and interspecific variation in 5s rna genes are decoupled in diploid wheat relatives**. Genetics 140:325–343. 120, 125, 132

Kellogg, E., R. Appels, and R. Mason-Gamer. 1996. **When genes tell different stories: the diploid genera of triticeae (gramineae)**. Systematic Botany 21:321–347. 125, 132

Kidd, K. and L. Sgaramella-Zonta. 1971. **Phylogenetic analysis: concepts and methods.** Am. J. Hum. Genet. 23:235–252. 17

Kimura, M. 1980. **A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences.** J Mol Evol 16:111–120. 13, 107

Kimura, M. 1981. **Estimation of evolutionary distances between homologous nucleotide sequences**. Proc Natl Acad Sci USA 78:454–458. 13

Kishino, H. and M. Hasegawa. 1989. **Evaluation of the maximum likelihood estimate of the evolutionary tree topologies from dna sequence data, and the branching order in hominoidea.** J Mol Evol 2:170–179. 13

Kluge, A. G. 1989. **A concern for evidence and a phylogenetic hypothesis of relationships among epicrates (boidae, serpentes)**. Syst. Zool. 38:7–25. 30

Kolaczkowski, B. and J. Thornton. 2004. **Performance of maximum parsimony and likelihood phylogenetics when evolution is heterogeneous**. Nature 431:980–984. 24

Kuhner, M. K., J. Yamato, and J. Felsenstein. 1995. **Estimating effective population size and mutation rate from sequence data using metropolis-hastings sampling**. Genetics 140:1421–1430. 20

Kumar, S. 1996. **A stepwise algorithm for finding minimum evolution trees**. Mol Biol Evol 13:584–593. 17

Lanyon, S. 1993. **Phylogenetic frameworks: Towards a firmer foundation for the comparative approach**. Biol. J. Linn. Soc. 49:45–61. 72

Lapointe, F. and C. Levasseur. 2004. **Everything you always wanted to know about the average consensus, and more**. Pages 87–105 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.). Kluwer. 77

Lapointe, F., M. Wilkinson, and D. Bryant. 2003. **Matrix representations with parsimony or with distances: two sides of the same coin?** Syst Biol 52:865–868. 77

Lapointe, F.-J. and G. Cucumel. 1997. **The Average Consensus Procedure: Combination of Weighted Trees Containing Identical or Overlapping Sets of Taxa**. Syst Biol 46:306–312. 38, 76, 77, 169

Larget, B. and D. Simon. 1999. **Markov chain monte carlo algorithms for the bayesian analysis of phylogenetic trees.** Mol Biol Evol 16:750–759. 21

Lartillot, N. and H. Philippe. 2004. **A bayesian mixture model for across-site heterogeneities in the amino-acid replacement process**. Mol. Biol. Evol. 21:1095–1109. 13

Lartillot, N. and H. Philippe. 2008. **Improvement of molecular phylogenetic inference and the phylogeny of bilateria**. Philos Trans R Soc Lond B Biol Sci 0962-8436 (Print) Journal article. 117, 119

Lawrence, J. G. and H. Ochman. 1998. **Molecular archaeology of the Escherichia coli genome**. Proceedings of the National Academy of Sciences of the United States of America 95:9413–9417. 27

Lawson, F., R. Charlebois, and J. Dillon. 1996. **Phylogenetic analysis of carbamoylphosphate synthetase genes: complex evolutionary history includes an internal duplication within a gene which can root the tree of life**. Mol Biol Evol 13:970–977. 165

Le, S. Q. and O. Gascuel. 2008. **An improved general amino acid replacement matrix**. Mol Biol Evol 25:1307–1320. 14

Le Quesne, W. J. 1974. **The uniquely evolved character concept and its cladistic application**. Syst. Zool. 23:513–517. 9

Li, S., D. Pearl, and H. Doss. 2000. **Phylogenetic tree construction using markov chain monte carlo.** 21

Liu, F., M. M.M., N. Freire, P. Ong, M. Tennant, T. Young, and K. Gugel. 2001. **Molecular and Morphological Supertrees for Eutherian (Placental) Mammals**. Science 291:1786–1789. 33

Lockhart, P., C. Howe, D. Bryant, T. Beanland, and A. Larkum. 1992. **Substitutional bias confounds inference of cyanelle origins from sequence data.** J. Mol. Evol. 34:153–162. 24

Lockhart, P., A. Larkum, M. Steel, P. Waddell, and D. Penny. 1996. **Evolution of chlorophyll and bacteriochlorophyll: the problem of invariant sites in sequence analysis**. Proc. Natl. Acad. Sci. USA 93:1930–1934. 24

Loomis, W. and D. Smith. 1990. **Molecular phylogeny of dictyostelium discoideum by protein sequence comparison**. Proc. Natl. Acad. Sci. USA 87:9093–9097. 24

Lopez, P., D. Casane, and H. Philippe. 2002. **Heterotachy, an important process of protein evolution**. Mol. Biol. Evol. 19:1–7. 24

Lukaszewski, A. J. and C. A. Curtis. 1993. **Physical distribution of recombination in b-genome chromosomes of tetraploid wheat**. Theoretical and Applied Genetics 84:121–127. 134

Luo, M. C., K. R. Deal, Z. L. Yang, and J. Dvorak. 2005. **Comparative genetic maps reveal extreme crossover localization in the aegilops speltoides chromosomes**. Theoretical and Applied Genetics 111:1098–1106. 134

Luo, M. C., Z. L. Yang, R. S. Kota, and J. Dvorak. 2000. **Recombination of chromosomes 3a(m) and 5a(m) of triticum monococcum with homeologous chromosomes 3a and 5a of wheat: the distribution of recombination across chromosomes**. Genetics 154:1301–1308. 134

Lynch, M. and J. S. Conery. 2000. **The Evolutionary Fate and Consequences of Duplicate Genes**. Science 290:1151–1155. 25

Lynch, M. and A. Force. 2000. **The Probability of Duplicate Gene Preservation by Subfunctionalization**. Genetics 154:459–473. 25

Ma, B., M. Li, and L. Zhang. 2000. **From gene trees to species trees**. SIAM J. Comput 30:729—752. 82, 138, 164

Maddison, W. 1997. **Gene trees in species trees**. Systematic Biology 46:523–536. 82

Mallet, J. 2007. **Hybrid speciation**. Nature 446:279–283. 119

Margulis, L. 1993. **Symbiosis in Cell Evolution**. W.H. Freeman and Company, New-York. 160

Margush, T. and F. R. Mcmorris. 1981. **Consensus n-trees**. Bulletin of Mathematical Biology 43:239–244. 45

Mason-Gamer, R. J. 2001. **Origin of north american elymus (poaceae: Triticeae) allotetraploids based on granule-bound starch synthase gene sequences**. Systematic Botany 26:757–768. 120, 132, 133

Mason-Gamer, R. J. 2005. **The β-amylase genes of grasses and a phylogenetic analysis of the triticeae (poaceae)**. American Journal of Botany 92:1045–1058. 120, 132, 133

Mason-Gamer, R. J., N. L. Orme, and C. M. Anderson. 2002. **Phylogenetic analysis of north american elymus and the monogenomic triticeae (poaceae) using three chloroplast dna data sets**. Genome 45:991–1002. 120, 125, 132

McMorris, F. R., D. B. Meronk, and D. A. Neumann. 1983. **A view of some consensus methods for trees**. Pages 122–125 *in* Numerical Taxonomy (J. Felsenstein, ed.). Springer-Verlag. 44, 48, 51

Meacham, C. 1983. **Theoretical and computational considerations of the compatibility of qualitative taxonomic**. Pages 304–314 *in* Numerical Taxonomy (J. Felsenstein, ed.). NATO ASI vol. G1. Springer-Verlag. 41

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. 1953. **Equation of state calculations by fast computing machines**. J. Chem. Phys. 21:1087–1092. 20

Morrell, P. L., D. M. Toleno, K. E. Lundy, and M. T. Clegg. 2005. **Low levels of linkage disequilibrium in wild barley (hordeum vulgare ssp. spontaneum) despite high rates of self-fertilization**. Proceedings of the National Academy of Sciences of the United States of America 102:2442–2447. 134

Murphy, W. J., T. H. Pringle, T. A. Crider, M. S. Springer, and W. Miller. 2007. **Using genomic data to unravel the root of the placental mammal phylogeny.** Genome Res. 17:413–421. 117

Nelson, G. 1979. **Cladistic analysis and synthesis: principles and definitions, with a historical note on adanson's familles des plantes (1763-1764)**. Syst Zool 28:1–21. 48

Nelson, G. and P. Ladiges. 1994. **Three-item consensus: empirical test of fractional weighting**. Pages 193–209 *in* Models in Phylogeny Reconstruction (R. Scotland, D. Seibert, and D. Wiliams, eds.). Systematics Association Special Volume No. 52. Oxford Univ. Press, Oxford. 74, 75

Neumann, D. A. 1983. **Faithful consensus methods for n-trees**. Matematical Biosciences 63:271–287. 89

Neyman, J. 1971. **Molecular studies of evolution: a source of novel statistical problems.** Pages 1–27 *in* Statistical decision theory and related topics. (S. Gupta and J. Yackel, eds.). Academy Press, New York. 18

Ng, M. and N. Wormald. 1996. **Reconstruction of rooted trees from subtrees**. Discrete Appl. Math. 69:19–31. 57, 60

Nylander, J. A., F. Ronquist, J. P. Huelsenbeck, and J. L. Nieves-Aldrey. 2004. **Bayesian phylogenetic analysis of combined data.** Systematic biology 53:47–67. 31

Ohno, S. 1970. **Evolution by gene duplication**. Springer-Verlag. 25

Page, D. M. 1990. **Tracks and trees in the antipodes: A reply to humphries and seberg**. Syst Zool 39:288–299. 48

Page, R. and A. Charleston. 1997a. **From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem.** Mol. Phylogenet. Evol. 7:231–240. 82

Page, R. and M. Charleston. 1997b. **Reconciled trees and incongruent gene and species trees**. Mathematical Heirarchies and Biology, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 37:57–70. 82

Page, R. D. M. 1989. **Comments on component-compatibility in historical biogeography**. Cladistics 5:167–182. 45, 48

Page, R. D. M. 2002. **Modified mincut supertrees**. Pages 537–552 *in* Proceedings of the 2nd International Workshop on Algorithms in Bioinformatics (WABI'02) (R. Guigó and D. Gusfield, eds.). 38, 43, 64, 65, 66, 123, 150, 169, 189

Pauplin, Y. 2000. **Direct calculation of tree length using a distance matrix.** J Mol Evol 51:41–47. 17

Paux, E., P. Sourdille, J. Salse, C. Saintenac, F. Choulet, P. Leroy, A. Korol, M. Michalak, S. Kianian, W. Spielmeyer, E. Lagudah, D. Somers, A. Kilian, M. Alaux, S. Vautrin, H. Berges, K. Eversole, R. Appels, J. Safar, H. Simkova, J. Dolezel, M. Bernard, and C. Feuillet. 2008. **A physical map of the 1-gigabase bread wheat chromosome 3b**. Science 322:101–104. 134

Pearson, H. 2008. **'virophage' suggests viruses are alive.** Nature 454. 27

Penel, S., A. M. Arigon, J. F. Dufayard, A. S. Sertier, V. Daubin, L. Duret, M. Gouy, and G. Perrière. 2009. **Databases of homologous gene families for comparative genomics**. BMC Bioinformatics 10 Suppl 6:S3. 139, 154, 160, 164, 176

Pennisi, E. 2003. **Modernizing the tree of life**. Science 300:1692–1697. 34

Petersen, G. and O. Seberg. 1997. **Phylogenetic analysis of the triticeae (poaceae) based on rpoa sequence data**. Molecular Phylogenetics and Evolution 7:217–230. 120, 125, 132

Petersen, G. and O. Seberg. 2002. **Molecular evolution and phylogenetic application of dmc1**. Molecular Phylogenetics and Evolution 22:43–50. 120, 132

Petersen, G., O. Seberg, M. Yde, and K. Berthelsen. 2006. **Phylogenetic relationships of triticum and aegilops and evidence for the origin of the a, b, and d genomes of common wheat (triticum aestivum)**. Molecular Phylogenetics and Evolution 39:70–82. 125, 129, 132, 133

Philippe, H. and A. Germot. 2000. **Phylogeny of eukaryotes based on ribosomal rna: long-branch attraction and models of sequence evolution**. Mol. Biol. Evol. 17:830–834. 24

Philippe, H. and P. Lopez. 2001. **On the conservation of protein sequences in evolution**. Trends Biochem. Sci. 26:414–416. 24

Philippe, H., Y. Zhou, H. Brinkmann, N. Rodrigue, and F. Delsuc. 2005. **Heterotachy and long-branch attraction in phylogenetics**. BMC Evolutionary Biology 5:50. 24

Phillips, C. and T. Warnow. 1996. **The assymetric median tree - a new model for building consensus trees**. Discrete Appl Math 71:311–335. 47

Piaggio-Talice, R., J. Burleigh, and O. Eulenstein. 2004. **Quartet supertrees**. chap. 8, Pages 173–190 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 82

Pisani, D. and M. Wilkinson. 2002. **Matrix representation with parsimony, taxonomic congruence, and total evidence**. Syst. Biol. 51:151–155. 32, 71

Pisani, D., A. M. Yates, M. C. Langer, and M. J. Benton. 2002. **A concern for evidence and a phylogenetic hypothesis of relationships among epicrates (boidae, serpentes)**. Philos. Trans. R. Soc. Lond. B. Biol Sci. 269:915–921. 34

Pollard, D. A., V. N. Iyer, A. M. Moses, and M. B. Eisen. 2006. **Widespread discordance of gene trees with species tree in drosophila: evidence for incomplete lineage sorting**. PLoS Genetics 2:1634–1647. 134, 135

Posada, D. and K. Crandall. 2002. **The effect of recombination on the accuracy of phylogeny estimation.** J. Mol. Evol. 54:396–402. 28

Presgraves, D. C. 2005. **Recombination enhances protein adaptation in drosophila melanogaster**. Current Biology 15:1651–1656. 124, 134

Przytycka, T. 1997. **Sparse dynamic programming for maximum agreement subtree problem**. Pages 249–264 *in* Mathematical Hierarchies in Biology DIMACS series in Discrete Mathematics and Theoretical Computer Science (B. Mirkin, F. McMorris, F. Roberts, and A. Rzhetsky, eds.). Amer. Math. Society, Providence. 49

Purvis, A. 1995a. **A modification to baum and ragan's method for combining phylogenetic trees**. Syst. Biol. 44:251–255. 69

Purvis, A. 1995b. **Philos. trans. r. soc. lond. b. biol sci.** Syst. Zool. 1326:405–421. 34

R-Development-Core-Team. 2006. **R: A language and environment for statistical computing**. 124

Ragan, M. A. 1992. **Matrix representation in reconstructing phylogenetic relationships among the eukaryots**. Biosystems 28:47–55. 38, 68, 69, 122, 169

Ranwez, V., V. Berry, A. Criscuolo, P. Fabre, S. Guillemot, C. Scornavacca, and E. Douzery. 2007a. **PhySIC: a veto supertree method with desirable properties**. Syst. Biol. 56:798–817. 2, 38, 73, 86, 91, 92, 93, 102, 103, 108, 115, 135, 138, 150, 155, 163

Ranwez, V., A. Criscuolo, and E. Douzery. 2009. **Supertriplets: A triplet-based supertree approach to molecular systematics and phylogenomics**. Submitted to Mol Biol Evol . 75, 79

Ranwez, V., F. Delsuc, S. Ranwez, K. Belkir, M.-K. Tilak, and E. J. P. Douzery. 2007b. **Orthomam: A database of orthologous genomic markers for placental mammal phylogenetics**. BMC Evol Biol 7:241+. 8, 22, 115, 117, 163, 171, 189

Richardson, A. O. and J. D. Palmer. 2007. **Horizontal gene transfer in plants.** Journal of experimental botany 58:1–9. 27

Rieseberg, L. H. 1997. **Hybrid origins of plant species**. Annual Review of Ecology Evolution and Systematics 28:359–389. 29, 131

Rieseberg, L. H., S. J. E. Baird, and K. A. Gardner. 2000. **Hybridization, introgression, and linkage evolution**. Plant Molecular Biology 42:205–224. 29, 131

Rivera, M. C. and J. A. Lake. 2004. **The ring of life provides evidence for a genome fusion origin of eukaryotes**. Nature 431:152–155. 27

Robinson, D. F. and L. R. Foulds. 1981. **Comparison of phylogenetic trees**. Mathematical Biosciences 53:131–147. 46

Rodrigo, A. G. 1993. **A comment on baum's method for combining phylogenetic trees.** Taxon 42:631–636. 32, 69

Rodrigo, A. G. 1996. **On combining cladograms.** Taxon 45:267–274. 32, 69, 72

Rohlf, F. J. 1982. **Consensus indices for comparing classifications**. Math Biosci 59:131–144. 51

Ronquist, F. 1996. **Matrix representation of trees, redundancy, and weighting**. Syst. Biol. 45:247–253. 70

Ronquist, F. and J. P. Huelsenbeck. 2003. **Mrbayes 3: Bayesian phylogenetic inference under mixed models.** Bioinformatics 19:1572–1574. 21, 31, 69, 121

Ronquist, F., J. P. Huelsenbeck, and T. Britton. 2004. **Bayesian supertrees**. chap. 9, Pages 193–224 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 69, 82

Roshan, U., B. Moret, T. Warnow, and T. Williams. 2004. **Performance of supertree methods on various data set decompositions**. chap. 8, Pages 301–328 *in* Phylogenetic supertrees: combining information to reveal the Tree of Life (O. Bininda-Emonds, ed.) vol. 3. Kluwer academic publishers. 35

Ross, H. A. and A. G. Rodrigo. 2004. **An assessment of matrix representation with compatibility in supertree construction.** *in* Phylogenetic supertrees (combining information to reveal the tree of life) (O. R. P. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 72

Ruths, D. and L. Nakhleh. 2005. **Recombination and phylogeny: effects and detection**. Int. J. Bioinformatics Res. Appl. 1:202–212. 28

Rzhetsky, A. and M. Nei. 1992. **Statistical properties of the ordinary least-squares, generalized least-squares, and minimum-evolution methods of phylogenetic inference.** J Mol Evol 35:367–375. 17

Rzhetsky, A. and M. Nei. 1993. **Theoretical foundation of the minimum-evolution method of phylogenetic inference.** Mol Biol Evol 1095:1073–1095. 17

Saintenac, C., M. Falque, O. C. Martin, E. Paux, C. Feuillet, and P. Sourdille. 2009. **Detailed recombination studies along chromosome 3b provide new insights on crossover distribution in wheat (triticum aestivum l.)**. Genetics 181:393–403. 134

Saitou, N. and M. Nei. 1987. **The neighbor-joining method: a new method for reconstructing phylogenetic trees.** Mol Biol Evol 4:406–425. 17

Salamin, N., T. R. Hodkinson, and V. Savolainen. 2002. **Building supertrees: an empirical assessment using the grass family (poaceae)**. Syst. Biol. 51:136–150. 34

Sanderson, M. J. 1989. **Confidence limits on phylogenies: The bootstrap revisited.** Cladistics 5:113–129. 21

Sanderson, M. J., A. C. Driskell, R. H. Ree, O. Eulenstein, and S. Langley. 2003. **Obtaining maximal concatenated phylogenetic data sets from large sequence databases**. Mol. Biol. Evol. 20:1036–1042. 34

Sanderson, M. J. and J. Kim. 2000. **Parametric Phylogenetics?** Syst Biol 49:817–829. 25

Sanderson, M. J., A. Purvis, and C. Henze. 1998. **Phylogenetic supertrees: assembling the trees of life**. Trends Ecol. Evol. 13:105–109. 1, 30, 32, 37

Sankoff, D. and P. Rousseau. 1975. **Locating the vertices of a steiner tree in an arbitrary metric space.** Math Progr 9:240–246. 9

Schuh, R. T. and J. S. Farris. 1981. **Methods for investigating taxonomic congruence and their application to the leptopodomorpha.** Syst Zool 30:331–351. 45

Schwarz, G. 1978. **Estimating the dimension of a model**. Annals of Statistics 6:461–464. 14

Scornavacca, C., V. Berry, V. Lefort, E. J. P. Douzery, and V. Ranwez. 2008. **Physic_ist: cleaning source trees to infer more informative supertrees**. BMC Bioinformatics 9:413. 2, 38, 86, 122, 135, 138, 150, 157, 158, 163

Scornavacca, C., V. Berry, and V. Ranwez. 2009a. **Building species trees from larger parts of phylogenomic databases**. Submitted to Information and Computation (Elsevier ed.) . 163

Scornavacca, C., V. Berry, and V. Ranwez. 2009b. **From gene trees to species trees through a supertree approach**. Pages 702–714 *in* LATA '09: Proceedings of the 3rd International Conference on Language and Automata Theory and Applications Springer-Verlag, Berlin, Heidelberg. 2, 139, 163

Seberg, O. and S. Frederiksen. 2001. **A phylogenetic analysis of the monogenomic triticeae (poaceae) based on morphology**. Botanical Journal of the Linnean Society 136:75–97. 132

Semple, C. 2003. **Reconstructing minimal rooted trees.** Discrete appl math 127:489–503. 54, 59

Semple, C. and M. Steel. 2000. **A supertree method for rooted trees**. Discrete Appl. Math. 105:147–158. 38, 43, 61, 63, 64, 150

Semple, C. and M. Steel. 2003. **Phylogenetics (Oxford Lecture Series in Mathematics and Its Applications, 24)**. Oxford University Press, USA. 39, 98

Sharkey, M. and J. Leathers. 2001. **Majority does not rule: The trouble with majority-rule consensus trees**. Cladistics 17:282–284. 46

Shimodaira, H. and M. Hasegawa. 1999. **Multiple comparisons of log-likelihoods with applications to phylogenetic inference**. Molecular Biology and Evolution 16:1114–1116. 122

Slowinski, J., A. Knight, and R. P. 1997. **Inferring species trees from gene trees: A phylogenetic analysis of the elapidae (serpentes) based on the amino acid sequences of venom proteins.** Mol. Phylogenet. Evol. 8:349–362. 82

Slowinski, J. and R. Page. 1999. **How should phylogenies be inferred from sequence data?** Syst Biol 48:814–825. 32, 82, 138, 164

Snir, S. and S. Rao. 2006. **Using max cut to enhance rooted trees consistency.** IEEE/ACM Trans. Comput. Biol. Bioinformatics 3(4):323–333. 38

Sober, E. 1998. **Reconstructing the past : parsimony, evolution and inference.** MIT Press, Cambridge, Mass. 10

Sokal, R. and J. Rohlf. 1981. **Taxonomic congruence in the leptopodomorpha re-examined.** Syst Zool 30:309–325. 44, 51

Soltis, P. and D. Soltis. 2001. **Molecular systematics: assembling and using the tree of life**. Taxon 50:663–677. 34

Somers, D. J., T. Banks, R. Depauw, S. Fox, J. Clarke, C. Pozniak, and C. McCartney. 2007. **Genome-wide linkage disequilibrium analysis in bread wheat and durum wheat**. Genome 50:557–567. 134

Springer, M. and W. De Jong. 2001. **Phylogenetics. which mammalian supertree to bark up?** Science 291:1709–1711. 32, 33

Staden, R., D. P. Judge, and J. K. Bonfield. 2000. **The staden package, 1998**. Methods in Molecular Biology 132:115–130. 121

Stamatakis, A. 2006. **Raxml-vi-hpc: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models**. Bioinformatics 22:2688–2690. 19

Steel, M. 1994. **Recovering a tree from the markov leaf colouration it generates under a markov model..** Appl Math Lett 7:19–23. 15

Steel, M., A. Dress, and S. Böcker. 2000. **Simple but fundamental limitations on supertree and consensus tree methods**. Syst Biol 49:363–368. 64, 89, 163

Steel, M. and D. Penny. 1993. **Distributions of tree comparison metrics - some new results.** Syst Biol 42:126–141. 77

Steel, M. and A. Rodrigo. 2008. **Maximum Likelihood Supertrees**. Syst Biol 57:243–250. 81

Steel, M. and T. Warnow. 1993. **Kaikoura tree theorems : Computing the maximum agreement subtree**. Information Processing Letters 48:77–82. 49

Steel, M. A. 1992. **The complexity of reconstructing trees from qualitative characters and subtree**. J. Classif. 9:91–116. 42, 67, 145

Sullivan, J. and D. Swofford. 1997. **Are guinea pigs rodents? the importance of adequate models in molecular phylogenetics.** J. Mamm. Evol. 4:77–86. 25

Sullivan, J. and D. L. Swofford. 2001. **Should We Use Model-Based Methods for Phylogenetic Inference When We Know That Assumptions About Among-Site Rate Variation and Nucleotide Substitution Pattern Are Violated?** Syst Biol 50:723–729. 25

Susko, E. 2003. **Confidence regions and hypothesis tests for topologies using generalized least squares .** Mol Biol Evol 20:862–868. 16

Swofford, D., G. Olsen, P. Waddell, and D. Hillis. 1996. **Phylogenetic inference**. Pages 407–514 *in* Molecular Systematics (D. Hillis, C. Moritz, and B. Mable, eds.). Sunderland MA: Sinauer Associates. 25

Swofford, D. L. 1991. **When are phylogeny estimates from molecular and morphological data incongruent?** Pages 295–333 *in* Phylogenetic analysis of DNA sequences (M. M. Miyamoto and J. Cracraft, eds.). Oxford University Press, New York. 48, 49

Swofford, D. L. 2003. **PAUP*: Phylogenetic Analysis Using Parsimony (*and Other Methods). Version 4.** Sinauer Associates, Sunderland, Massachusetts. 10, 19, 76, 115, 122

Tamura, K. 1992. **Estimation of the number of nucleotide substitutions when there are strong transition-transversion and G+C-content biases**. Mol Biol Evol 9:678–687. 13

Tamura, K. and M. Nei. 1993. **Estimation of the number of nucleotide substitutions in the control region of mitochondrial dna in humans and chimpanzees.** Mol Biol Evol 10:512–526. 13, 121

Tavaré, S. 1986. **Some probabilistic and statistical problems on the analysis of dna sequences**. Lectures in Mathematics and Life Sciences 17:57–86. 12, 121

Thorley, J., M. Wilkinson, and M. Charleston. 1998. **The information content of consensus trees**. Pages 91–98 *in* Advances in Data Science and Classification. Studies in Classification, Data Analysis, and Knowledge Organization (A. Rizzi, M. Vichi, and H.-H. Bock, eds.). 97, 150, 173

Thorley, J. L. and R. D. M. Page. 2000. **Radcon: phylogenetic tree comparison and consensus.** Bioinformatics Pages 486–487. 75

Thorley, J. L. and M. Wilkinson. 2003. **A view of supertrees methods**. Pages 185–194 *in* Bioconsensus (M. F. Janowitz, F.-J. Lapointe, F. R. McMorris, and F. S. Roberts, eds.) vol. 61 of *Discrete Mathematics and Theoretical Computer Science* DIMACS. 38, 89, 169

Tuffley, C. and M. Steel. 1998. **Modeling the covarion hypothesis of nucleotide substitution**. Math. Biosci. 147:63–91. 24

Uzzell, T. and K. W. Corbin. 1971. **Fitting discrete probability distributions to evolutionary events**. Science 172:1089–1096. 13

Vernot, B., M. Stolzer, A. Goldman, and D. Durand. 2008. **Reconciliation with non-binary species trees**. J Comput Biol 15:981—1006. 82, 138, 164

Vinh, L. S. and A. Von Haeseler. 2004. **Iqpnni: Moving fast through tree space and stopping in time**. Mol Biol Evol 21:1565–1571. 19

Waines, J. G. and D. Barnhart. 1992. **Biosystematic research in aegilops and triticum**. Hereditas 116:207–212. 135

Wang, R. R. C. 1989. **An assessment of genome analysis based on chromosome pairing in hybrids of perennial triticeae**. Genome 32:179–189. 135

Wang, R. R. C. 1992. **Genome relationships in the perennial triticeae based on diploid hybrids and beyond**. Hereditas 116:133–136. 135

Whelan, S. and N. Goldman. 2001. **A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach**. Mol Biol Evol 18:691–699. 14

Whitfield, J. B. and P. J. Lockhart. 2007. **Deciphering ancient rapid radiations**. Trends in Ecology and Evolution 22:258–265. 119

Wildman, D. E., M. Uddin, J. C. Opazo, G. Liu, V. Lefort, S. Guindon, O. Gascuel, L. I. Grossman, R. Romero, and M. Goodman. 2007. **Genomics, biogeography, and the diversification of placental mammals**. Proc. Nat. Acad. Sci. 104:14395–14400. 117

Wilkinson, M. 1994. **Common cladistic information and its consensus representation: Reduced adams and reduced cladistic consensus trees and profiles.** Syst Biol 43:343–368. 45, 89

Wilkinson, M. 1995. **Coping with missing entries in phylogenetic inference using parsimony.** Syst Biol 44:501–514. 45

Wilkinson, M. 1996. **Majority-rule reduced consensus methods and their use in bootstrapping.** Mol Biol Evol 13:437–444. 45

Wilkinson, M. 1998. **Reduced supertrees**. Trends Ecol. Evol. 13:283–283. 81

Wilkinson, M., J. Cotton, C. Creevey, O. Eulenstein, S. Harris, F. Lapointe, C. Levasseur, J. Mcinerney, D. Pisani, and J. Thorley. 2005a. **The shape of supertrees to come: Tree shape related properties of fourteen supertree methods**. Syst. Biol. 54:419–431. 75, 79

Wilkinson, M., J. Cotton, and J. Thorley. 2004a. **The information content of trees and their matrix representations**. Syst Biol 53:989–1001. 75

Wilkinson, M., D. Pisani, J. A. Cotton, and I. Corfe. 2005b. **Measuring support and finding unsupported relationships in supertrees**. Syst. Biol. 54:823–831. 86

Wilkinson, M. and J. Thorley. 2001. **Efficiency of strict consensus trees.** Syst Biol 50:610–613. 45

Wilkinson, M. and J. Thorley. 2003. **Reduced consensus**. Pages 195–204 *in* Bioconsensus (M. F. Janowitz, F.-J. Lapointe, F. R. McMorris, and F. S. Roberts, eds.) vol. 61 of *Discrete Mathematics and Theoretical Computer Science* DIMACS. 81

Wilkinson, M., J. Thorley, D. Pisani, F.-J. Lapointe, and O. McInerney. 2004b. **Some desiderata for liberal supertree**. Pages 227–246 *in* Phylogenetic supertrees (combining information to reveal the tree of life) (O. R. P. Bininda-Emonds, ed.) vol. 4. Kluwer academic publishers. 87, 89, 163

Wilkinson, M., J. L. Thorley, D. T. J. Littlewod, and R. A. Bray. 2001. **Towards a phylogenetic supertree of Platyhelminthes?** Pages 292–301 *in* Interrelationships of the Platyhelminthes (D. T. J. Littlewood and R. A. Bray, eds.). Taylor & Francis, London. 70, 75, 79

Williams, D. and C. Humphries. 2003. **Component coding, three-item coding, and consensus methods**. Syst Biol 52:255–259. 75

Williams, W. and H. Clifford. 1971. **On the comparison of two classifications of the same set of elements**. Taxon 20:519–522. 77

Willson, S. J. 1999. **Building phylogenetic trees from quartets by using local inconsistency measures**. Mol Biol Evol 16:685–693. 82

Willson, S. J. 2001. **An error correcting map for quartets can improve the signals for phylogenetic trees**. Mol Biol Evol 18:344–351. 82

Willson, S. J. 2004. **Constructing rooted supertrees using distance.** Bull. Math. Biol. 66:1755–1783. 60

Wu, B. Y. 2004. **Constructing the maximum consensus tree from rooted triples.** Journal of Combinatorial Optimization 29:29–39. 99

Wu, C. F. J. 1986. **Jackknife, bootstrap and other resampling methods in regression analysis**. The Annals of Statistics 14:1261–1295. 22

Yamane, K. and T. Kawahara. 2005. **Intra- and interspecific phylogenetic relationships among diploid triticum-aegilops species (poaceae) based on base-pair substitutions, indels, and microsatellites in chloroplast noncoding sequences**. American Journal of Botany 92:1887–1898. 120, 129, 133

Yang, Z. 1993. **Maximum-likelihood estimation of phylogeny from dna sequences when substitution rates differ over sites**. Molecular Biology and Evolution 10:1396–1401. 13, 121

Yang, Z. 1994. **Estimating the pattern of nucleotide substitution**. Journal of Molecular Evolution 39:105–111. 12, 121

Yang, Z. 1996a. **Among-site rate variation and its impact on phylogenetic analyses**. Trends Ecol Evolut 11:367–372. 13

Yang, Z. 1996b. **Maximum-likelihood models for combined analyses of multiple sequence data**. Journal Of Molecular Evolution 42:587–596. 31

Yang, Z. 2006. **Computational Molecular Evolution**. Oxford University Press. 21

Yang, Z. 2007. **Paml 4: a program package for phylogenetic analysis by maximum likelihood**. Molecular Biology and Evolution 24:1586–1591. 122

Yang, Z. and B. Rannala. 1997. **Bayesian phylogenetic inference using dna sequences: A markov chain monte carlo method**. Mol Biol Evol 14:717–724. 21

Zhang, J. 2003. **Evolution by gene duplication: an update**. Trends in Ecology & Evolution 18:292–298. 25

Zharkikh, A. 1994. **Estimation of evolutionary distances between nucleotide sequences.** J Mol Evol 39:315–329. 15

Zharkikh, A. and W.-H. Li. 1995. **Estimation of confidence in phylogeny: the complete-and-partial bootstrap technique**. Mol Phylogenet Evol 4:44–63. 13

Zhou, Y., N. Rodrigue, N. Lartillot, and H. Philippe. 2007. **Evaluation of the models handling heterotachy in phylogenetic inference.** BMC Evol. Biol. 7:206. 24

Zou, X. H., F.-M. Zhang, J.-G. Zhang, L.-L. Zang, L. Tang, J. Wang, T. Sang, and S. Ge. 2008. **Analysis of 142 genes resolves the rapid diversification of the rice genus**. Genome Biology 9:R49. 131, 134, 135

Zwickl, D. J. 2006. **Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion.** Ph.D. thesis The University of Texas at Austin. 19

Résumé:

La phylogénétique est un champ de recherche de la biologie qui étudie les relations évolutives entre les espèces grâce à des données moléculaires et morphologiques. Ces relations peuvent être résumées dans un arbre communément appelé "arbre des espèces". Ces arbres sont principalement estimés en analysant des "arbres de gènes", *i.e.*, des arbres évolutifs construits par l'analyse d'une famille de gènes. Toutefois, pour des raisons à la fois méthodologiques et biologiques, un arbre de gènes peut différer par endroits de l'arbre des espèces. Pour estimer ce dernier, les biologistes analysent donc simultanément plusieurs jeux de données correspondant à différentes familles de gènes, laissant le poids de l'évidence décider.

Ce travail de thèse s'est focalisé sur l'approche "super-arbre" pour combiner les jeux de données. Cette approche consiste premièrement à construire des arbres (appelés communément arbres sources) à partir de données primaires, puis à les assembler en un arbre plus grand et plus complet, appelé super-arbre. Si elles sont utilisées au sein d'une approche "diviser pour régner" dans le but de reconstituer des grandes parties de l'arbre de vie, il est préférable d'utiliser une méthode de super-arbres conservative afin d'obtenir des arbres très fiables. Dans ce contexte, une méthode de super-arbre doit afficher seulement des informations fiables qui sont présentes ou induites par les arbres sources (propriété d'induction – PI), et qui n'entrent pas en conflit avec ces derniers ou avec une de leurs combinaisons (propriété de non contradiction – PC). Nous avons défini de manière formelle ces deux propriétés. De plus, comme aucune des méthodes de super-arbres existantes ne garantissait l'obtention d'un super-arbre satisfaisant PI et PC, nous avons développé un algorithme permettant de modifier un super-arbre afin qu'il les satisfasse. Nous avons également conçu deux méthodes, *PhySIC* et *PhySIC_IST*, qui construisent directement des super-arbres satisfaisant ces deux propriétés. L'application de *PhySIC_IST* au problème complexe de la phylogénie des Triticeae a permis de mieux comprendre l'histoire évolutive de ce groupe.

Les événements de duplication aboutissent presque toujours à la présence de plusieurs copies du même gène dans les génomes. Les arbres de gènes sont donc généralement multi-étiquetés, *i.e.*, une seule espèce étiquette plusieurs feuilles. Comme aucune méthode n'existe actuellement pour combiner ce type d'arbres, ils sont le plus souvent complètement ignorés dans les approches phylogénomiques classiques. Pourtant, ils représentent 60% à 80% des arbres de gènes disponibles dans les banques de données moléculaires. Dans cette thèse, nous proposons plusieurs algorithmes permettant d'obtenir, à partir d'un arbre multi-étiqueté, un arbre classique (*i.e.*, où chaque espèce n'apparaît qu'une seule fois) contenant un maximum d'informations de spéciation présentes dans l'arbre initial. Cet arbre peut ensuite être utilisé par n'importe quelle méthode de super-arbres. Une application à la base de données HOGENOM est présentée.

Abstract:

Phylogenetics is the field of evolutionary biology that studies the evolutionary relationships between species through morphological and molecular data. These relationships can be summarized in the so- called "species tree". A gene tree is an evolutionary tree constructed by analyzing a gene family. Species trees are mainly estimated using gene trees. However, for both methodological and biological reasons, a gene tree may differ from the species tree. To estimate species tree, biologists then analyze several data sets at a time, letting the weight of the evidence decide.

This thesis focuses on the "supertree" approach to combine data sets. This approach consists first in constructing trees (commonly called source trees) from primary data, then assembling them into a larger and more comprehensive tree, called supertree. When using supertree construction in a divide-and-conquer approach in the attempt to reconstruct large portions of the Tree of Life, conservative supertree methods have to be preferred in order to obtain reliable supertrees. In this context, a supertree method should display only information that is displayed or induced by source trees (induction property – PI) and that does not conflict with source trees or a combination thereof (non contradiction property – PC). In this thesis we introduce two combinatorial properties that formalize these ideas. We proposed algorithms that modify the output of any supertree methods such that it verifies these properties. Since no existing supertree method satisfies both PI and PC, we have developed two methods, *PhySIC* and *PhySIC_IST*, which directly build supertrees satisfying these properties. An application of *PhySIC_IST* to the complex problem of the history of Triticeae is presented.

Since duplication events often result in the presence of several copies of the same genes in the species genomes, gene trees are usually multi-labeled, *i.e.*, , a single species can label more than one leaf. Since no supertree method exists to combine multi-labeled trees, until now these gene trees were simply discarded in supertree analyses. Yet, they account for 60% to 80 % of the gene trees available in phylogenomic databases. In this thesis, we propose several algorithms to extract a maximum amount of speciation signal from multi-labeled trees and put it under the form of single-labeled trees which can be handled by supertree methods. An application to the HOGENOM database is presented.

Mots-clés: Phylogénie, Phylogénomique, Superarbe, Méthodes de type veto, Arbres multi-étiquetés

Keywords: Phylogenetics, Phylogenomics, Supertree, Veto methods, Multi-labeled trees