Phylogeny Reconstructions Come of Age

Inaugural – Dissertation

zur
Erlangung des Doktorgrades der
Mathematisch–Naturwissenschaftlichen Fakultät
der Heinrich–Heine–Universität Düsseldorf

vorgelegt von

Le Sy Vinh

Düsseldorf

2005

Gedruckt mit der Genehmigung der Mathematisch-Naturwissenschaftlichen
Fakultät der Heinrich-Heine-Universität Düsseldorf

Referent:	Prof. Dr. Arndt von Haeseler
Korreferent:	Prof. Dr. Gerhard Steger
Tag der mündlichen Prüfung:	04.11.2005

This thesis is dedicated to my beloved family!

Acknowledgments

First and foremost, I wish to thank Prof. Arndt von Haeseler for his fruitful discussions, excellent advises and friendly behavior.

I would like to thank my colleagues at Bioinformatics Institute in Düsseldorf, especially Thomas Schlegel, Heiko A. Schmidt, Ingo Ebersberger, Jutta Buschbom, Roland Fleissner, Steffen Klaere, Bui Quang Minh, and Lutz Voigt for their continuous support.

The help on official documents from Helga Frank and Claudia Kiometzis is highly appreciated.

Financial support and the use of supercomputing resources of the ZAM/NIC at the Forschungszentrum Jülich are gratefully acknowledged.

I want to thank Prof. Ho Tu Bao, Dr. Luong Chi Mai, and Dr. Vu Dinh Hoa and Dr. Ha Quang Thuy for introducing me to Bioinformatics and their continuous support.

Parts of this thesis have been published in the following articles:

1. Le Sy Vinh, Heiko A. Schmidt, and Arndt von Haeseler (2005), PhyNav: A Novel Approach to Reconstruct Large Phylogenies, In C. Weihs and W. Gaul (eds.) Classification, the Ubiquitous Challenge Series Studies in Classification, Data Analysis, and Knowledge Organization, 386-393, Springer, Heidelberg/New York.

2. Le Sy Vinh and Arndt von Haeseler (2004), IQPNNI: Moving Fast Through Tree Space and Stopping in Time, *Mol. Biol. Evol.*, 21(8):1565-1571.

3. Le Sy Vinh and Arndt von Haeseler (2005), Shortest Triplet Clustering: Reconstructing Large Phylogenies using Representative Sets, *BMC Bioinformatics*, 6:92.

IQPNNI, PhyNav, STC packages as well as example data are freely available from `http://www.bi.uni-duesseldorf.de/software/`.

Contents

1 Overview

1.1 Motivation

Understanding evolutionary relationships among species is one of the central objectives in biology. According to the Charles Darwin's theory of evolution, species have evolved from ancestors. The evolutionary relationship can be illustrated in an evolutionary tree, a so-called *phylogeny*. The leaves represent contemporary species, whereas the internal nodes can be thought of as speciation events, and the root is considered as the common ancestor of all species in the tree. It is commonly accepted that phylogenies are rooted and bifurcating (Harding, 1971). The total number of possible bifurcating rooted trees for n species is given by

$$B_r(n) = \prod_{i=3}^{n} (2i - 3) \text{ for } n \geq 3. \tag{1.1}$$

$B_r(n)$ increases exponentially with n (Felsenstein, 1978). For $n = 56$ species, the number of trees exceeds the estimated number of 10^{81} atoms in the known universe.

Up to date, a tremendous amount of genetic data (nucleotides/amino acids) has been collected thanks to the development of efficient sequencing technologies (Sanger *et al.*, 1977) and many genome projects. The content of public databases like the GenBank database increases quickly (see Figure 1.1) (Benson *et al.*, 2005). By April 2005, the GenBank database has gained more than 48 billions base pairs. This gives us an unprecedented opportunity to investigate the evolutionary relationships among a large set of species.

The reconstruction of phylogenetic trees for large data sets is a challenging problem in phylogenetic analysis due to the exponentially increasing number of possible trees and available genetic data. Searching the best phylogenies based on the maximum parsimony criterion or the minimum evolution criterion is known to be NP-complete (Graham and

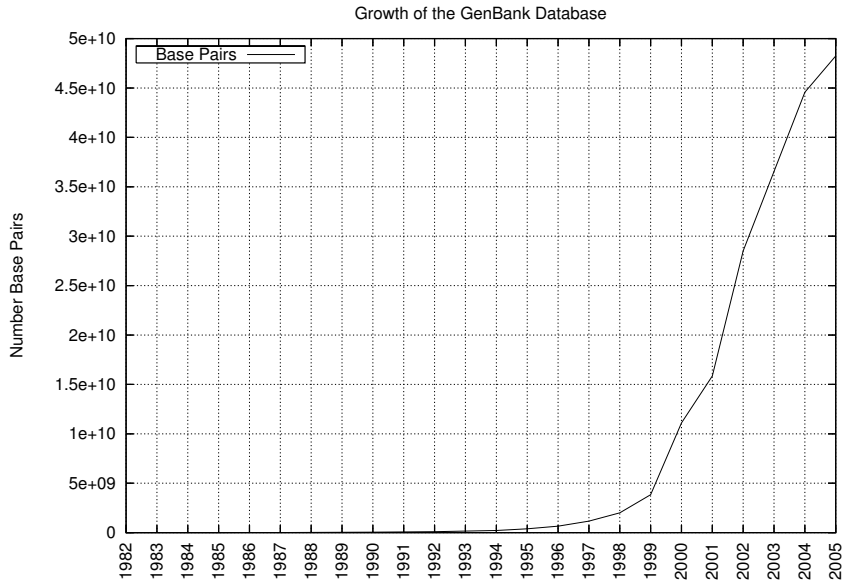


Figure 1.1: The public database GenBank has been growing exponentially. In April 2005, the GenBank database contains more than 48 billions base pairs.

Foulds, 1982; Day and David, 1986). Recently, Chor and Tuller (2005) proved that finding the maximum likelihood phylogeny is NP-hard. Remarkably, NP-complete and NP-hard problems are believed to be unsolvable in polynomial time (Cormen *et al.*, 2001). That is to say, there is an essential need of heuristic methods to efficiently construct phylogenies for large datasets in terms of both accuracy and runtime.

1.2 Organization of this thesis

Chapter 2: An introduction to phylogenetic tree inference is presented. First, biological data used to infer the evolutionary relationships among contemporary species is introduced. Second, the evolutionary process of sequences is modeled under the statistical framework. Then, phylogenetic trees which illustrate the historical relationships among species are described thoroughly. Finally, the state-of-the-art phylogenetic tree reconstruction methods are summarized.

Chapter 3: A novel search strategy, namely *phylogenetic navigator* (PhyNav), is proposed to efficiently elucidate the tree space. The search gives encouraging results compared to other methods.

Chapter 4: The definition of so-called *important quartets* is presented. Important quartets are used as building blocks for our *important quartet puzzling* algorithm (IQP) to construct a tree for n sequences in $O(n^2)$ time. Then, the IQP is implemented in a combined method called *important quartet puzzling and nearest neighbor interchange* (IQPNNI) to search for maximum likelihood trees with up to a thousand of sequences. IQPNNI shows better accuracy than other tested methods on both simulated and real data.

Chapter 5: *The Shortest triplet clustering* algorithm (STC) for construction of very large phylogenies based on distance matrices is presented. STC can build trees with 5000 taxa within one minute. The STC as well as other distance-based methods are examined extensively on a large range of simulated data.

Chapter 6: The content of the thesis is summarized.

Finally, the IQPNNI, PhyNav, and STC packages are described in the **Appendix**.

2 Introduction to phylogenetic tree reconstruction

This chapter gives an introduction to phylogenetic tree reconstruction. To this end, we first present different kinds of biological data used to study evolutionary relationships among contemporary species. Second, generalities about graphs, especially phylogenetic trees, are introduced. Third, the state-of-the-art phylogenetic inferring approaches e.g. maximum parsimony methods, maximum likelihood methods and distance-based algorithms are summarized.

2.1 Biological data

2.1.1 Phylogenetic signals

Species have evolved from a common ancestor (Darwin, 1872). More precisely, *homologous characters* of species have derived from the same ancestral character which can be *morphological characters, gene order data, nucleotide sequences*, or *amino acid sequences*. Two homologous characters are formally defined (Fitch, 2000):

Definition 1 *Two characters that have descended, usually with divergence, from a common ancestral character are called* **homologous**.

The evolutionary relationships among species can be investigated by analyzing the difference among their homologous characters.

Morphological characters were the first class of data used in phylogenetic analysis. They refer to characters which represent the visible features of a species. The biggest advantage of morphological data is that they can be obtained easily and cheaply from a large range of taxa. However, measuring the difference between morphological characters poses a difficult task. Moreover, the number of common morphological characters

among species, specially among distantly related species, is limited. Therefore, they may reveal not enough phylogenetic information to reconstruct the relationships (for more discussions see Hillis and Wiens, 2000).

Besides morphological characters, gene-order data have been employed in construction of phylogenies (Moret and Warnow, 2005, and references therein). The difference among species is manifested by the difference among orders of genes in their genomes caused by genome rearrangement events. The gene-order data can be applied to genomes which are completely sequenced. Unfortunately up to now, only a small number of complete genomes is available (typically, small genomes). This limitation prevents the application of this approach to most of species.

Nowadays, genetic sequences (nucleotides and amino acids) are prevalent in phylogenetic inferences (Swofford *et al.*, 1996; Felsenstein, 2004, and references therein). Generally speaking, nucleotides in DNA sequences exist at four different states: Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). They can be classified into either purine (A and G) or pyrimidine (C and T) bases. When considering RNA sequences, Uracil (U) is substituted for Thymine.

Also, amino acid sequences play an important role in phylogenetic analysis. They are biologically produced from nucleotide sequences through the following (synthesis) process

$$\text{DNA} \xrightarrow{Transcription} \text{mRNA} \xrightarrow{Translation} \text{Protein.}$$

Each triplet of three successive nucleotides in protein-coding DNA sequences is called a codon, which either encodes a single amino acid or signals the end of the process. In the universal genetic code, of the 64 possible codons, 61 encode for amino acids while the remaining three are stop-codons. Twenty different amino acids are available and listed in Table 2.1 (Brown, 2002). The genetic code is degenerated, that is, multiple codons encode for the same amino acid.

Huge amounts of genetic sequences (nucleotides and amino acids) have been collected and stored in public databases like GenBank (Benson *et al.*, 2005). These allow us to study relationships among various species based on large numbers of characters compared to morphological characters. Therefore, they tend to increase the reliability of inferences (Hillis and Wiens, 2000). Moreover, genetic sequences give us a chance to study relationships among distantly related species for whom common morphological characters are unavailable (Hillis and Wiens, 2000).

Table 2.1: Twenty different amino acids

Name	Three-letter code	One letter code
Alanine	Ala	A
Cysteine	Cys	C
Aspartic Acid	Asp	D
Glutamic Acid	Glu	E
Phenylalanine	Phe	F
Glycine	Gly	G
Histidine	His	H
Isoleucine	Ile	I
Lysine	Lys	K
Leucine	Leu	L
Methionine	Met	M
Asparagine	Asn	N
Proline	Pro	P
Glutamine	Gln	Q
Arginine	Arg	R
Serine	Ser	S
Threonine	Thr	T
Valine	Val	V
Tryptophan	Trp	W
Tyrosine	Tyr	Y

2.1.2 Sequence alignment

Understanding the differences among sequences is the first and fundamental task to investigate their historical relationships. The difference between homologous nucleotide sequences from different species are caused by accumulative point mutations e.g. due to errors during DNA replication or damaging effects of mutagens such as chemicals and radiation (Brown, 2002). Point mutations can be divided into three classes: substitutions, deletions, and insertions (Brown, 2002):

- **Substitutions**: Replacing one nucleotide by another in the sequence.
- **Deletions**: Deleting one or several nucleotides from the sequence.

- **Insertions**: Inserting one or several nucleotides into the sequence.

Moreover, two different types of substitutions can be distinguished:

- **Transitions**: Changing a purine into the other purine (A⇆G) or a pyrimidine into the other pyrimidine (C⇆T).
- **Transversions**: Changing a purine into a pyrimidine and vice versa: A⇆C, A⇆T, G⇆C, G⇆T.

2.1.2.1 Pairwise sequence alignment

Consider two following homologous sequences from Human and Chimpanzee:

	1	2	3	4	5	6	7	8	9	10	11	12
Human	A	A	C	C	T	T	T	C	C	C	T	T
Chimpanzee	A	C	C	T	T	T	C	C	C	T	T	

The lengths of these two sequences might be unequal due to insertions and deletions. In other words, two characters in the same column might not be homologous.

The evolutionary relationship between the two is studied by examining the differences between homologous characters. To this end, the two sequences have to be aligned into a *pairwise sequence alignment* such that two characters at the same column (site) are homologous (Waterman, 2000):

	1	2	3	4	5	6	7	8	9	10	11	12
Human	A	A	C	C	T	T	T	C	C	C	T	T
Chimpanzee	A	C	C	-	T	T	T	C	C	C	T	T

The pairwise sequence alignment shows a point mutation a so-called *mismatch* at position 2 at which either a nucleotide substitution 'A' in the Human sequence or a nucleotide substitution 'C' in the Chimpanzee sequence was occurred. The mismatch carries the phylogenetic information and can be used to reconstruct evolutionary events. In addition, we observe another point mutation at position 4 which is either a 'C' was inserted into the Human sequence or deleted from the Chimpanzee sequence. Since ancestral character states are usually not available, one cannot distinguish between an insertion and a deletion. Therefore, they are referred to as *indels*.

The pairwise sequence alignment of two sequences can be constructed in $O(m^2)$ time using dynamic programming (Waterman, 2000) where m is the length of the pairwise sequence alignment, i.e. the number of columns.

2.1.2.2 Multiple sequence alignment

Generally, homologous sequences which are the subject of phylogenetic analysis are aligned into a data matrix **D**, called *multiple sequence alignment* (MSA), such that all homologous characters are assigned into the same column (site) (Waterman, 2000; Higgins, 2003). For instance, consider the following hypothetical multiple sequence alignment **D** of eight sequences:

	1	2	3	4	5	6	7	8	9	10	11	12
Human	A	A	C	C	T	T	T	C	C	C	T	T
Chimpanzee	G	A	C	-	T	T	T	C	C	C	T	T
Gorilla	C	A	C	C	T	T	T	C	C	C	T	T
Rhesus	T	A	C	-	T	T	T	C	C	C	T	T
Cow	T	C	C	-	T	T	T	C	C	C	T	T
Dog	T	C	C	-	T	T	T	C	C	C	T	T
Mouse	T	G	C	-	T	T	T	C	C	C	T	T
Bird	T	G	T	-	T	T	T	C	C	C	T	T

The alignment **D** consists of 12 columns from D_1 to D_{12}. The columns D_1, D_2, D_3 contain substitutions. Indels are introduced at the column D_4. The eight remaining sites are constant, that is, all nucleotides in the respective alignment columns are identical.

The multiple sequence alignment **D** can be constructed in $O(m^n 2^n)$ runtime and $O(m^n)$ memory space using dynamic programming (Waterman, 2000) where n is the number of sequences and m is the length of **D**, i.e. the number of columns. This computational expense limits this approach to a few sequences. For larger number of sequences, approximate methods have been proposed such as CLUSTALW (Thompson *et al.*, 1994), T-COFFEE (Notredame *et al.*, 2000), or MUSCLE (Edgar, 2004).

2.1.3 Models of sequence evolution

Once homologous sequences have been aligned, the relationships can be analyzed based on their homologous characters. The estimate of *pairwise genetic distances* (*evolutionary distances*) between sequences is a fundamental and essential task in sequence analysis such as searching closely related sequences in databases and reconstructing distance-based phylogenetic trees (Strimmer and von Haeseler, 2003).

Mathematically, we denote $\mathbb{A} = \{\text{A, C, G, T}\}$ the alphabet of 4 possible nucleotide states. Similarly, the alphabet of twenty amino acid states is abbreviated by $\mathbb{A} =$

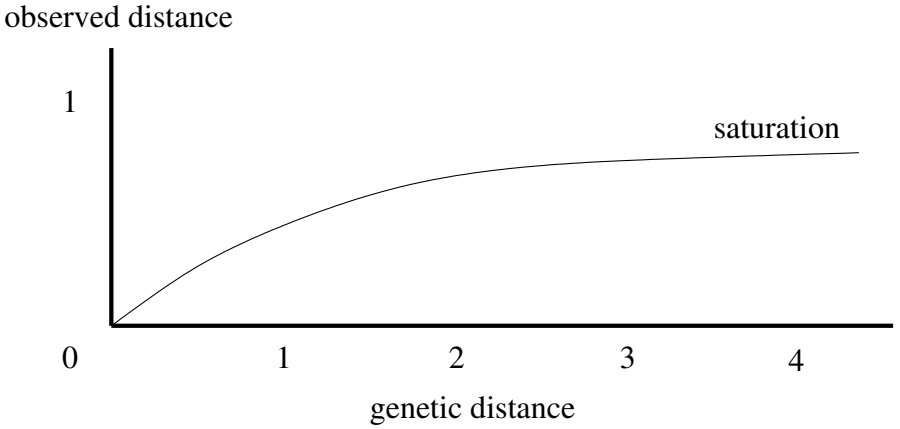


Figure 2.1: The relationship between the observed distance and the genetic distance between two sequences. If the genetic distance is small, it is estimated properly by the observed distance. However, as the genetic distance increases, the observed distance is saturated and limited by one. Consequently, the observed distance underestimates the genetic distance.

{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. Consider two aligned sequences (nucleotides or amino acids) $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ where $x_i, y_i \in \mathbb{A}$ for $i = 1 \ldots m$.

Definition 2 *The* **genetic distance** $d_g(\mathbf{x}, \mathbf{y})$ *between two homologous sequences* $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ *and* $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ *with* $x_i, y_i \in \mathbb{A}$ *for* $i = 1 \ldots m$ *is the actual number of substitutions which have occurred between* $\mathbf{x}$ *and* $\mathbf{y}$ *per site.*

Estimating the genetic distances between sequences typically requires a statistical description of the substitution process between nucleotides/amino acids, called *model of substitution.*

Before describing these models, let us make a short excursion into the *observed distance* between two sequences which is the most simple and intuitive estimate of their genetic distance (Strimmer and von Haeseler, 2003).

Definition 3 *The* **observed distance** $d_o(\mathbf{x}, \mathbf{y})$ *between two homologous sequences* $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ *and* $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ *with* $x_i, y_i \in \mathbb{A}$ *for* $i = 1 \ldots m$ *is the proportion of mismatch sites in their respective pairwise sequence alignment. Mathematically,*

$$d_o(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{m} \delta(x_i, y_i)}{m} \tag{2.1}$$

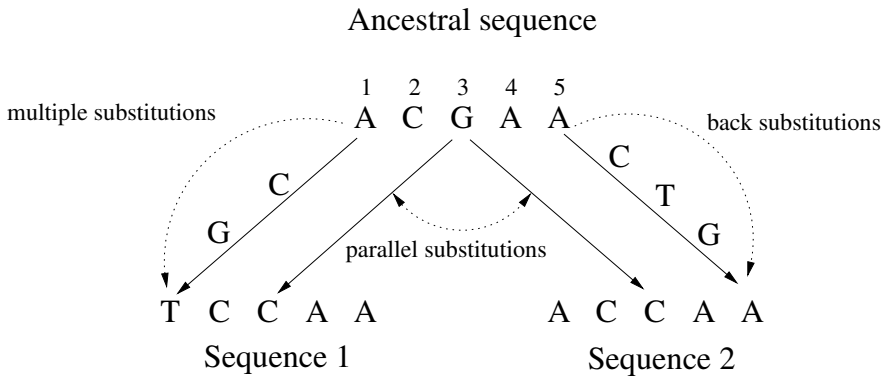


Figure 2.2: The evolution of two sequences from the same ancestral sequence.

where

$$\delta(x_i, y_i) = \begin{cases} 0 & \textit{if } x_i = y_i \\ 1 & \textit{otherwise} \end{cases} \tag{2.2}$$

If the genetic distance $d_g(\mathbf{x}, \mathbf{y})$ is small, it is estimated properly by the observed distance $d_o(\mathbf{x}, \mathbf{y})$ as illustrated by Figure 2.1. However, a high substitution rate or a long evolutionary time between sequences might cause a severe underestimate of the genetic distance from the observed distance. More explicitly, Figure 2.2 shows three instances, namely *multiple substitutions, parallel substitutions* and *back substitutions*, in which the observed distance between two sequences is much smaller than the number of actual substitutions between them:

- **Multiple substitutions**: Two or more substitutions have happened at the same site. However, at most one substitution is observed at the site in the pairwise sequence alignment (see site 1 in Figure 2.2).
- **Parallel substitutions**: The same substitutions have occurred at the same site in both sequences. Consequently, we observe no substitution between two characters at the site in the pairwise sequence alignment (see site 3 in Figure 2.2).
- **Back substitutions**: Two or several substitutions have occurred at the same site in one sequence. However, the last character state is identical to the first one. As a result, no substitution is observed at the site in the pairwise sequence alignment (see site 5 in Figure 2.2).

To overcome these problems, let us now model the substitution process between nucleotides and amino acids.

2.1.3.1 Models of nucleotide substitution

The substitution process between nucleotides is modeled as a *time-homogeneous time-continuous stationary Markov process* (Tavaré, 1986; Strimmer and von Haeseler, 2003, and references therein). The central component of the process is the so-called *instantaneous substitution rate matrix*

$$\mathbf{Q} = \begin{pmatrix} -\sum_{Y \neq A} Q_{AY} & a\pi_C & b\pi_G & c\pi_T \\ a'\pi_A & -\sum_{Y \neq C} Q_{CY} & d\pi_G & e\pi_T \\ b'\pi_A & d'\pi_C & -\sum_{Y \neq G} Q_{GY} & f\pi_T \\ c'\pi_A & e'\pi_C & f'\pi_G & -\sum_{Y \neq T} Q_{TY} \end{pmatrix} \tag{2.3}$$

where Q_{ij} is the number of substitutions from nucleotide i to nucleotide j per time unit. Parameters $a,\ a',\ b,\ b',\ c,\ c',\ d,\ d',\ e,\ e',\ f,\ f'$ correspond to the relative substitution rates from one nucleotide to another. Finally, parameters $\pi_A,\ \pi_C,\ \pi_G,\ \pi_T$ describe the frequencies of nucleotides A, C, G, T, respectively. Note that the diagonal elements Q_{ii} are assigned such that the sum of each row equals zero.

The time-reversibility assumption is usually imposed to the phylogenetic inference, that is, the relative substitution rates between nucleotide i and nucleotide j are the same in both directions. Specifically, the relative substitution rates $a' = a,\ b' = b,\ c' = c,\ d' = d,\ e' = e$ and $f' = f$. Consequently, the *most general time-reversible model* (GTR) (Tavaré, 1986) is

$$\mathbf{Q} = \begin{pmatrix} -\sum_{Y \neq A} Q_{AY} & a\pi_C & b\pi_G & c\pi_T \\ a\pi_A & -\sum_{Y \neq C} Q_{CY} & d\pi_G & e\pi_T \\ b\pi_A & d\pi_C & -\sum_{Y \neq G} Q_{GY} & f\pi_T \\ c\pi_A & e\pi_C & f\pi_G & -\sum_{Y \neq T} Q_{TY} \end{pmatrix} \tag{2.4}$$

The model imposes four conditions:

- The rate of change from nucleotide i to nucleotide j is independent of the history of nucleotide i (*Markov property*).
- The substitution rates are constant over time (*time-homogeneous*).
- The substitution between nucleotides can occur at any time in the process (*time-continuous*).
- The frequencies $\boldsymbol{\pi} = (\pi_A, \pi_C, \pi_G, \pi_T)$ of the nucleotides A, C, G, T are at equilibrium (*stationarity*).

The instantaneous substitution rate matrix $\mathbf{Q}$ can be decomposed into relative substitution rate matrix $\mathbf{R} = \{R_{ij}\}$ and nucleotide frequencies $\boldsymbol{\pi}$ as

$$Q_{ij} = \begin{cases} \pi_j R_{ij} & \text{if } i \neq j \\ -\sum_{x \neq i} Q_{ix} & \text{if } i = j \end{cases} \tag{2.5}$$

where the relative substitution rate matrix is

$$\mathbf{R} = \begin{array}{c} \begin{array}{cccc} \text{A} & \text{C} & \text{G} & \text{T} \end{array} \\ \begin{pmatrix} - & a & b & c \\ a & - & d & e \\ b & d & - & f \\ c & e & f & - \end{pmatrix} \end{array} \begin{array}{c} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} \tag{2.6}$$

Once the instantaneous substitution rate matrix $\mathbf{Q}$ is specified, the so-called *transition probability matrix* $\mathbf{P}(t) = \{P_{ij}(t)\}$ in which $P_{ij}(t)$ is the probability to change from nucleotide i to nucleotide j during the evolutionary time t can be computed by

$$\mathbf{P}(t) = e^{\mathbf{Q}t} = \sum_{\nu=0}^{\infty} \frac{\mathbf{Q}^\nu t^\nu}{\nu!} \tag{2.7}$$

We must note, that the instantaneous substitution rate matrix $\mathbf{Q}$ is typically scaled such that the expected number of substitutions per time unit, called *substitution rate*, is one:

$$-\sum_{Y \in \mathbb{A}} \pi_Y Q_{YY} = 1. \tag{2.8}$$

Consequently, $P_{ij}(t)$ is the probability to change from nucleotide i to nucleotide j after t substitutions (t can be a fractional value).

Since the general-time reversible model $\mathbf{Q}$ is diagonalizable (Keilson, 1979; Gu and Li, 1996), $\mathbf{P}(t)$ can be calculated efficiently using the decomposition of $\mathbf{Q}$ (e.g. von Haeseler, 1999). Specifically,

$$\mathbf{P}(t) = \mathbf{U} \times e^{\mathbf{\Lambda} t} \times \mathbf{U}^{-1} \tag{2.9}$$

or more precisely,

$$P_{ij}(t) = \sum_{\nu=1}^{|\mathbb{A}|} U_{\nu i} \times e^{\lambda_\nu t} \times U^{-1}_{j\nu} \tag{2.10}$$

where

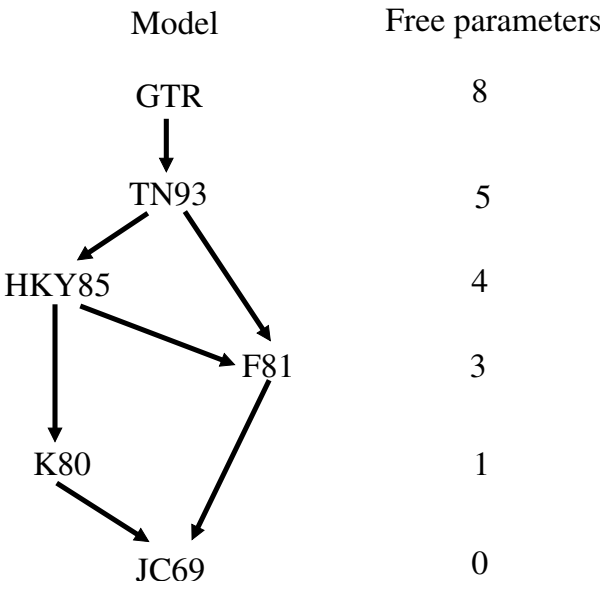


Figure 2.3: Different models of nucleotide substitutions and their number of free parameters.

- $|\mathbb{A}| = 4$ is the number of possible nucleotide states.
- $\mathbf{\Lambda} = \text{diag}\{\lambda_1, \lambda_2, \ldots, \lambda_{|\mathbb{A}|}\}$ is the $|\mathbb{A}| \times |\mathbb{A}|$ diagonal matrix corresponding to the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{|\mathbb{A}|}$ of $\mathbf{Q}$.
- $\mathbf{U} = \{u_1, u_2, \ldots, u_{|\mathbb{A}|}\}$ is the matrix of corresponding eigenvectors of $\mathbf{Q}$ and $\mathbf{U}^{-1}$ is its inverse.

The general time-reversible model $\mathbf{Q}$ has 8 free parameters. However, one can impose restrictions to obtain nested models such as JC69 (Jukes and Cantor, 1969), F81 (Felsenstein, 1981b), K2P (Kimura, 1980), HKY85 (Hasegawa *et al.*, 1985), or TN93 (Tamura and Nei, 1993).

Figure 2.3 shows different models of nucleotide substitutions as well as their number of free parameters. The free parameters of the models are usually estimated from data using computer programs such as PAUP* (Swofford, 2002), TREE-PUZZLE (Schmidt *et al.*, 2002), MRBAYES (Ronquist and Huelsenbeck, 2003), PHYML (Guindon and Gascuel, 2003) or IQPNNI (**Chapter** 4).

2.1.3.2 Models of amino acid substitution

Amino acid sequences were among the first kind of molecular data used to study relationships among species in 1960s by Eck and Dayhoff. Similarly to nucleotides, the

substitution process between amino acids is assumed to be a time-homogeneous time-continuous time-reversible stationary Markov process. However, twenty possible amino acid states require too many $\binom{20}{2}$ substitution model parameters to be estimated. Therefore, the parameters are typically derived from empirical studies based on a large amount of data (Dayhoff *et al.*, 1978; Jones *et al.*, 1992; Adachi and Hasegawa, 1996; Müller and Vingron, 2000; Whelan and Goldman, 2001). Hence, models of amino acid substitution are called *empirical substitution models*.

Dayhoff *et al.* (1978) were the firsts to model the amino acid substitutions. They employed 71 sets of closely related proteins and observed 1572 substitutions between amino acids. They compiled these substitutions into the popular *probability of accepted mutation* (PAM) matrices or *Dayhoff models*.

Of these, PAM-001 is the most important PAM matrix which presents the probability of substitution from one amino acid to another if one percent of amino acids have substituted between them. More generally, PAM-t is the probability of substitution from one amino acid to another if the amount of substitutions between them is t percent. PAM-t can be computed easily by raising the PAM-001 matrix to the t power (for more discussions see Felsenstein, 2004).

Jones *et al.* (1992) applied the same methodology as Dayhoff *et al.* (1978) but to larger available protein data sets to tabulate another probability of accepted mutation matrix, namely the *JTT* matrix.

A shortcoming of PAM matrices is that they are only compiled on closely related protein sequences. Müller and Vingron (2000) introduced an improved estimator, called the *resolvent method*, to overcome this limitation. Subsequently, they computed the so-called *VT* matrices based on protein sequences of varying degree of divergence from the SYSTERS database (Krause *et al.*, 1999).

Adachi and Hasegawa (1996) studied the amino acid substitution process in the context of mtDNA-encoded proteins. They constructed a transition probability matrix, called the *mtREV* matrix, using the maximum likelihood method based on 20 complete vertebrate mtDNA-encoded protein sequences. The authors showed that mtREV outperformed other models when analyzing the phylogenetic relationships among species based on their mtDNA-encoded protein sequences.

More thoroughly, Whelan and Goldman (2001) used an approximate maximum likelihood method to estimate a new model of amino acid substitution, namely the *WAG*, based on 3,905 globular protein sequences from 182 protein families. They showed that

WAG was better than Dayhoff models with respect to maximum likelihood values for a large number of globular protein families.

2.1.3.3 Estimating pairwise genetic distances

Having modeled the sequence substitution process $\mathbf{Q}$, let us now estimate the pairwise genetic distance $d_g(\mathbf{x}, \mathbf{y})$ between two aligned sequences (nucleotides or amino acids) $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ with $x_i, y_i \in \mathbb{A}$ for $i = 1, \ldots, m$. Typically, the pairwise genetic distance $d_g(\mathbf{x}, \mathbf{y})$ is estimated using the maximum likelihood principle (Strimmer and von Haeseler, 2003, and references therein).

The prerequisite of maximum likelihood estimate is the definition of the *likelihood function* $L(d)$. Loosely speaking, the likelihood function $L(d)$ measures the probability to observe two sequences $\mathbf{x}$ and $\mathbf{y}$ if d substitutions have occurred between them per site. Mathematically,

$$L(d) = \prod_{i=1}^{m} \pi_{x_i} P_{x_i y_i}(d). \tag{2.11}$$

The distance d^* which maximizes the likelihood function $L(d)$ is called the maximum likelihood estimate of the genetic distance $d_g(\mathbf{x}, \mathbf{y})$ (Strimmer and von Haeseler, 2003). Precisely,

$$d^* = \text{argmax}_{d \geq 0}\{L(d)\}. \tag{2.12}$$

The maximum likelihood estimate d^* can be determined using numerical optimization approaches such as the Brent's method or Newton-Raphson's method (e.g. Press *et al.*, 2002).

2.1.4 Models of rate heterogeneity

It has been shown that substitution rates can vary among sites of sequences. This observation is called *heterogeneous substitution rates* (Felsenstein, 2004, and references therein). For example, substitution rates at third positions on protein-coding nucleotide sequences are typically much faster than at first and second positions (e.g. Nei and Kumar, 2000).

In previous sections, the substitution process between nucleotides/amino acids was modeled with the so-called *homogeneous substitution rates* assumption (substitution

rates are the same among sites of sequences). This unrealistic assumption might cause inaccurate sequence analyzes such as estimating incorrectly genetic distances between sequences, or constructing wrong phylogenies. To relax this assumption, different models of heterogeneous substitution rates have been proposed (Fitch and Margoliash, 1967b; Uzzel and Corbin, 1971; Hasegawa *et al.*, 1985; Churchill *et al.*, 1992; Wakeley, 1993; Meyer and von Haeseler, 2003).

Rate heterogeneity was first modeled by Fitch and Margoliash (1967b) who classified sequence sites as either invariable or variable. Therefore, it is called the *two-state model*. Particularly, the substitution rate scaling factor r_i at site i is

$$r_i = \begin{cases} 0 & \text{if site } s \text{ is invariable} \\ 1 & \text{otherwise} \end{cases} \tag{2.13}$$

Variable and invariable sites are not distinguishable because of possible back substitutions or by chance some variable sites are unvaried (Churchill *et al.*, 1992). To overcome the problem, the two-state model imposes a parameter θ which indicates the percentage of invariable sites on the sequence. In real applications, the parameter θ is usually estimated from data.

Nowadays, the Γ-distribution is widely used to model rate heterogeneity (Uzzel and Corbin, 1971; Wakeley, 1993). Thus, substitution rate scaling factors across sites are typically drawn from a Γ-distribution with expectation 1.0 and variance $1/\alpha$, $\alpha > 0$

$$f(r) = \frac{\alpha^{\alpha} r^{\alpha-1}}{\exp(\alpha r)\Gamma(\alpha)} \tag{2.14}$$

where

$$\Gamma(\alpha) = \int_0^{\infty} e^{-t} t^{\alpha-1} dt. \tag{2.15}$$

The model is called *Gamma rate heterogeneity model*.

The degree of rate heterogeneity across sites is adjusted by varying the shape parameter α as shown in Figure 2.4. A smaller shape parameter α describes a stronger heterogeneity of rates across sites. For example, a strong heterogeneous rate is modeled by setting shape parameter $\alpha = 0.5$. That means substitution rates are very slow at most of sites, but much faster at a few sites. In contrast, if $\alpha = 10$, we observe a weak heterogeneity of substitution rates. In other words, substitution rate scaling factors are close to 1.0 over all sites.

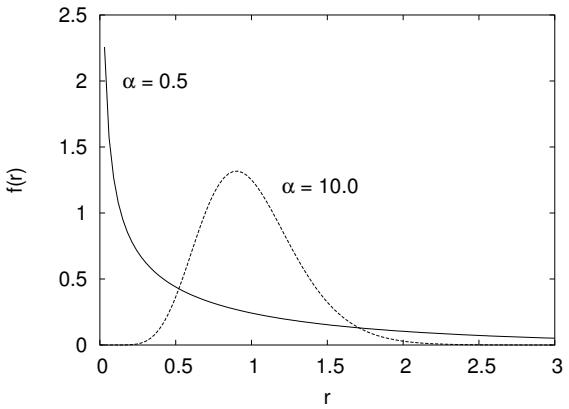


Figure 2.4: Different shapes of Γ-distribution with respect to shape parameter α.

Remarkably, the $\Gamma(\alpha)$ function is approximated efficiently by a discrete Γ-function with a finite number c of equally probable substitution rate scaling factor categories $r_1, r_2, \ldots, r_c$ (Yang, 1994). The shape parameter α is typically estimated from data using phylogenetic packages such as PAUP* (Swofford, 2002), TREE-PUZZLE (Schmidt *et al.*, 2002), MRBAYES 3 (Ronquist and Huelsenbeck, 2003), PHYML (Guindon and Gascuel, 2003) or IQPNNI (**Chapter** 4).

The combination of the two-state model and Γ-distribution model is also possible (Gu *et al.*, 1995). The hybrid model assumes a fraction θ of sequence sites to be invariable, other sites are variable with substitution rate scaling factors drawn from the Γ-distribution.

More recently, Meyer and von Haeseler (2003) have proposed a method to identify *site-specific substitution rates*. The method estimates a substitution rate scaling factor for each site based on the maximum likelihood principle. The site-specific substitution rate model is implemented in the Parat program (Meyer and von Haeseler, 2003) as well as the IQPNNI package.

2.2 Graphs and phylogenetic trees

Graph theory plays an important role in phylogenetic analysis. It provides a means to present relationships among objects (typically species/sequences) concisely and precisely. This section introduces generalities about graphs. More importantly, phylogenetic trees, a standard graphical representation of historical relationships among species, are thoroughly described.

2.2.1 Generalities about graphs

We start with the definition of a graph G (Gondran *et al.*, 1984; Semple and Steel, 2003):

Definition 4 *A* **graph** G *is composed of the pair* (V, E) *where* V *is the set of vertices (nodes) and* $E \subseteq \{(u, v) \,|\, u, v \in V\}$ *is the set of edges.*

Explicitly, we can use $V(G)$ and $E(G)$ as notations of vertex set V and edge set E of graph G, respectively.

In the phylogenetic analysis context, we consider only *simple graphs* $G = (V, E)$ which satisfy two conditions (Semple and Steel, 2003):

- Each edge $e = (u, v) \in E$ connects two distinct vertices $u, v \in V$ (*no loops*),
- each pair of vertices $u, v \in V$ is connected by at most one edge $e = (u, v) \in E$ (*no parallel edges*).

Figure 2.5(a) illustrates a simple graph G with vertex set $V = \{1, 2, 3, 4, 5, 6\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$.

$$\text{A graph } G = (V, E) \text{ is } \begin{cases} \textit{undirected} & \text{If all edges of } G \text{ are undirected. Precisely,} \\ & \text{each edge } (u, v) \in E \text{ is a pair of two} \\ & \text{unordered vertices } u, v \in V. \\ \textit{directed} & \text{Otherwise.} \end{cases}$$

Unless otherwise stated, graph G always indicates an undirected graph.

For an edge $e = (u, v) \in E$, u and v are two endpoints of e. In addition, edge e is said to be *incident* with u and v. We also say that u and v are *adjacent* or *neighbors*. We denote $d(v)$ for each vertex $v \in V$ the *degree* of v which is the number of edges incident with v. For example, the degrees of vertices in Figure 2.5(a) are $d(1) = d(2) = 1,\ d(3) = 4,\ d(4) = d(5) = d(6) = 2$.

The concept *subgraph* provides a means to describe the relationship between two graphs G and G'. A graph G' is called a subgraph of a graph G if and only if

- $V(G') \subseteq V(G)$ and

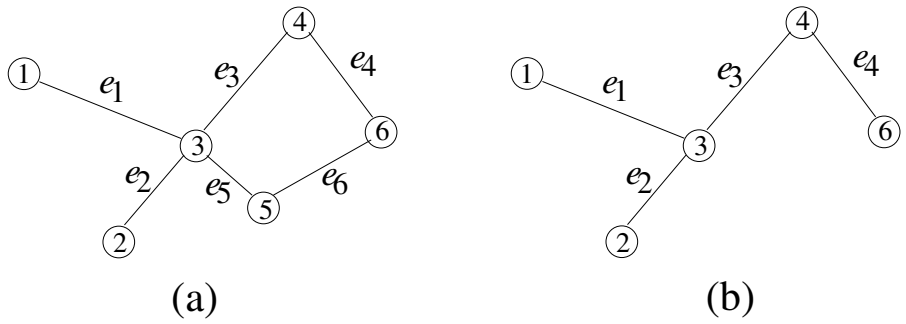


Figure 2.5: (a) Graph G with vertex set $V = \{1, 2, 3, 4, 5, 6\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$. (b) A subgraph G' with vertex subset $V(G') = \{1, 2, 3, 4, 6\}$ and edge subset $E(G') = \{e_1, e_2, e_3, e_4\}$ of graph G.

- $E(G') \subseteq E(G)$.

Figure 2.5(b) shows a subgraph G' with vertex subset $V(G') = \{1, 2, 3, 4, 6\}$ and edge subset $E(G') = \{e_1, e_2, e_3, e_4\}$ of graph G in Figure 2.5(a).

A *path* $p(u, v)$ connecting two vertices $u, v \in V$ in a graph G is a sequence of distinct vertices $v_1, \ldots, v_k$ such that

- $v_1 = u$ and $v_k = v$
- $(v_i, v_{i+1}) \in E \qquad$ for $i = 1 \ldots k - 1$.

In addition, if $(v_k, v_1) \in E$, the subgraph G' with vertex subset $V' = \{v_1, \ldots, v_k\}$ and edge subset $E' = \{(v_i, v_{i+1}) | i = 1 \ldots k - 1\} \cup \{(v_k, v_1)\}$ is called a *k-cycle*. For example, subgraph G' in Figure 2.5(a) with vertex subset $V' = \{3, 4, 6, 5\}$ and edge subset $E' = \{e_3, e_4, e_6, e_5\}$ is a 4-cycle.

$$\text{A graph } G = (V, E) \text{ is } \begin{cases} \textit{connected} & \text{If each pair of vertices } u, v \in V \text{ is connected by at least one path } p(u, v). \\ \textit{unconnected} & \text{Otherwise.} \end{cases} \tag{2.16}$$

We define an *edge length function* $\ell : E \mapsto R_+$ which maps each edge $e \in E$ into a real positive number $l(e) \in R_+$, called the *edge length*. The length $\ell(p(u, v))$ of path

$p(u, v) = \{v_1, v_2, \ldots, v_k\}$ joining two vertices u and v is computed by

$$\ell(p(u, v)) = \sum_{i=1}^{k-1} \ell(v_i, v_{i+1}). \tag{2.17}$$

2.2.2 Generalities about trees

Having described notations and concepts of the graph theory, we now focus on a special graph structure the so-called *trees*. They are widely used to store and present information, especially in phylogenetic analysis (Semple and Steel, 2003).

2.2.2.1 Tree structures

Definition 5 *A* **tree** $T = (V, E)$ *is a connected graph without cycles.*

A vertex $v \in V$ of degree 1 is called a *leaf*, all other vertices are called *interior nodes*. We denote with L the set of all leaves in tree T. Edges whose endpoints are both interior nodes are called *interior edges*, the others are called *external edges*. More precisely, an external edge is incident with a leaf and an interior node. Note that the terms branches and edges can be used interchangeably in trees.

Figure 2.6(a) illustrates a tree T with vertex set $V = \{1, 2, 3, 4, i_1, i_2\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5\}$. More precisely, $L = \{1, 2, 3, 4\}$ is the leaf set; i_1 and i_2 are interior nodes. Besides, e_1, e_3, e_4 and e_5 are external edges whereas e_2 is an interior one.

A tree T has the following important properties:

- $|V| = |E| + 1$ and
- for any two vertices $u, v \in V$, there exists a unique path $p(u, v)$ joining u and v.

The *length of the tree* T, denoted ℓ_T, is simply the sum of lengths over all edges and computed by

$$\ell_T = \sum_{e \in E} \ell(e). \tag{2.18}$$

Definition 6 *A tree T with a distinguished vertex r considered as the* **root** *is called a* **rooted tree** *and denoted by T_r. Otherwise, it is called an* **unrooted tree.**

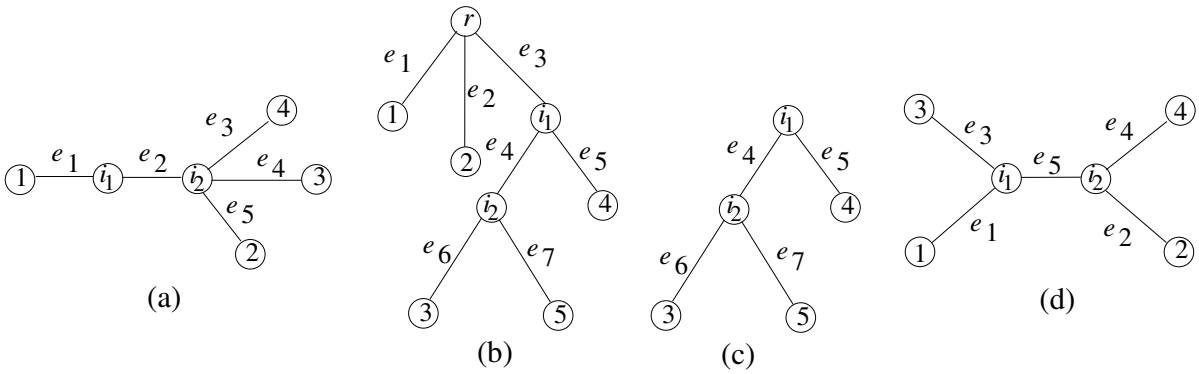


Figure 2.6: (a) A tree T with vertex set $V = \{1, 2, 3, 4, i_1, i_2\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5\}$. (b) A rooted tree T_r with a distinguished root r. (c) A rooted subtree T_{i_1} of rooted tree T_r. (d) A binary unrooted tree with vertex set $V = \{1, 2, 3, 4, i_1, i_2\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5\}$.

We now consider a rooted tree T_r with the root r. A node u in T_r is called an *ancestor* of another node v if u is on path $p(r, v)$ from the root r to vertex v. In other words, v is a *descendant* of u. Obviously, the root r is an ancestor of all other nodes. More precisely, node v is called a *child* of node u if and only if

- v is a descendant of u and
- edge $(u, v) \in E$.

Figure 2.6(b) shows a rooted tree T_r in which the root r has three children 1, 2 and i_1.

A tree $T_{r'} = (V', E')$ is said to be a *rooted subtree* of $T_r = (V, E)$ if and only if

- $V' = \{r'\} \cup \{\text{all descendants of } r'\}$ and
- $E' = \{e' = (u', v') \mid e' \in E, \text{ and } u', v' \in V'\}$ includes all edges of E whose both endpoints belong to vertex subset V'.

Figure 2.6(c) shows a rooted subtree T_{i_1} with vertex subset $V(T_{i_1}) = \{i_1, i_2, 3, 4, 5\}$ and edge subset $E(T_{i_1}) = \{e_4, e_5, e_6, e_7\}$ of rooted tree T_r in Figure 2.6(b).

A special tree structure which is typically used to present the historical relationships among species is *binary or bifurcating trees.*

Definition 7 *A rooted tree T_r is* **bifurcating** *if the root r and interior nodes each has exactly two children.*

Figure 2.6(c) illustrates a bifurcating rooted tree T_{i_1} with vertex set $V = \{3, 4, 5, i_1, i_2\}$ and edge set $E = \{e_4, e_5, e_6, e_7\}$.

Definition 8 *An unrooted tree T is* **bifurcating** *if each interior node has degree of 3.*

Figure 2.6(d) illustrates a binary unrooted tree with vertex set $V = \{1, 2, 3, 4, i_1, i_2\}$ and edge set $E = \{e_1, e_2, e_3, e_4, e_5\}$.

2.2.2.2 Traversals on trees

Traversals on trees to search, collect, set, or update information at their nodes and edges are a fundamental routine in tree-based studies, especially the phylogenetic tree reconstruction. We detail here two widely used traversal strategies to visit all nodes of a rooted tree T_r, namely *preorder* and *postorder* traversals (Aho *et al.*, 1974).

Assume that the root r has k children $r_1, r_2, \ldots, r_k$. A preorder traversal visits nodes of T_r recursively in the following orders:

Algorithm 2.1: Preorder traversal

begin
  (i): visit the root r, then
  (ii): visit rooted subtrees $T_{r_1}, T_{r_2}, \ldots, T_{r_k}$ using the preorder traversal.
end

For example, the order of visited nodes of T_r in Figure 2.6(b) using the preorder reversal is r, 1, 2, i_1, i_2, 3, 5, 4.

A postorder traversal on T_r is defined recursively as follows:

Algorithm 2.2: Postorder traversal

begin
  (i): visit rooted subtrees $T_{r_1}, T_{r_2}, \ldots, T_{r_k}$ using the postorder traversal, then
  (ii): visit the root r.
end

Nodes of T_r in Figure 2.6(b) are visited in orders 1, 2, 3, 5, i_2, 4, i_1, r using the postorder reversal.

2.2.3 Phylogenetic trees

We are now ready to introduce the so-called *phylogenetic trees* which are used to represent the evolutionary relationships among species. Considering a set of n species (typically sequences) $S = \{s_1, s_2, \ldots, s_n\}$, a phylogenetic tree is formally defined (Semple and Steel, 2003):

Definition 9 *A* **phylogenetic tree** $\mathcal{T}(S)$ *is a pair* (T, φ) *consisting of an underlying tree* $T = (V, E)$ *and an injective map* $\varphi : S \mapsto V$. $\mathcal{T}(S)$ *is called a phylogenetic tree on* S.

To avoid unnecessary complications, we consider only bijective maps from the species set S on the leaf set L of T. Moreover, each species $s \in S$ is considered as the *label* of a leaf $\varphi(s) \in L$. This simplification results in an equivalence between the labeled tree T with leaf set L and the phylogenetic tree $\mathcal{T}(S)$ on the species set S. In other words, the species of S are related by labeled tree T. Typically, binary trees are used in phylogenetic analysis (Harding, 1971).

2.2.3.1 The number of trees

Species of S can be related by different binary trees. The numbers of possible rooted and unrooted binary trees can be computed easily (Felsenstein, 1978). The number of binary unrooted trees $B(n)$ with n leaves is

$$B(n) = \prod_{i=3}^{n} (2i - 5). \tag{2.19}$$

The number of binary rooted trees $B_r(n)$ is

$$B_r(n) = \prod_{i=3}^{n} (2i - 3). \tag{2.20}$$

These numbers increase exponentially with n. Table 2.2 presents the number of binary rooted and unrooted trees with $n = 3 \ldots 10$ leaves.

2.2.3.2 Comparison of phylogenetic trees

Since different trees are possible for the same set of species/sequences, the next fundamental task is to measure the difference between trees. This is employed as a proper

Table 2.2: The numbers of binary rooted and unrooted trees with n leaves.

n	rooted	unrooted
3	3	1
4	15	3
5	105	15
6	945	105
7	10,395	945
8	135,135	10,395
9	2,027,025	135,135
10	34,459,425	2,027,025

measure to assess the quality of different phylogenetic tree reconstruction methods. Typically, the difference between two trees is measured by the Robinson and Foulds distance (Robinson and Foulds, 1981). The Robinson and Foulds distance between two trees is easily formulated, but it requires a short excursion into the *bipartition concept* on trees.

Definition 10 *Two disjoint leaf subsets L_A and L_B splitted by an interior edge e in a tree T with leaf set L, that is $L_A \cap L_B = \emptyset$ and $L_A \cup L_B = L$, are called a* **bipartition** *of the tree T and denoted by $L_A|L_B$.*

If T is a bifurcating unrooted tree with n leaves, then $n-3$ bipartitions corresponding to $n-3$ interior edges are possible. For example, edges e_3 and e_4 of tree T_1 in Figure 2.7(a) result in respective bipartitions $\{1,2\} \mid \{3,4,5\}$ and $\{1,2,3\} \mid \{4,5\}$. Similarly, tree T_2 in Figure 2.7(b) has two bipartitions $\{1,3\} \mid \{2,4,5\}$ and $\{1,2,3\} \mid \{4,5\}$.

Definition 11 *The* **Robinson and Foulds** *(RF) distance between two trees is the number of bipartitions present in one of the two trees but not the other.*

Unless otherwise stated, the Robinson and Foulds distance between two trees is standardized by dividing by the total number of possible bipartitions. Two important properties of RF distance are

- the RF distance between two trees ranges from 0.0 to 1.0. It is zero when two trees are identical, or one if the trees do not share any bipartitions,
- the smaller the RF distance between two trees the closer are their topologies.

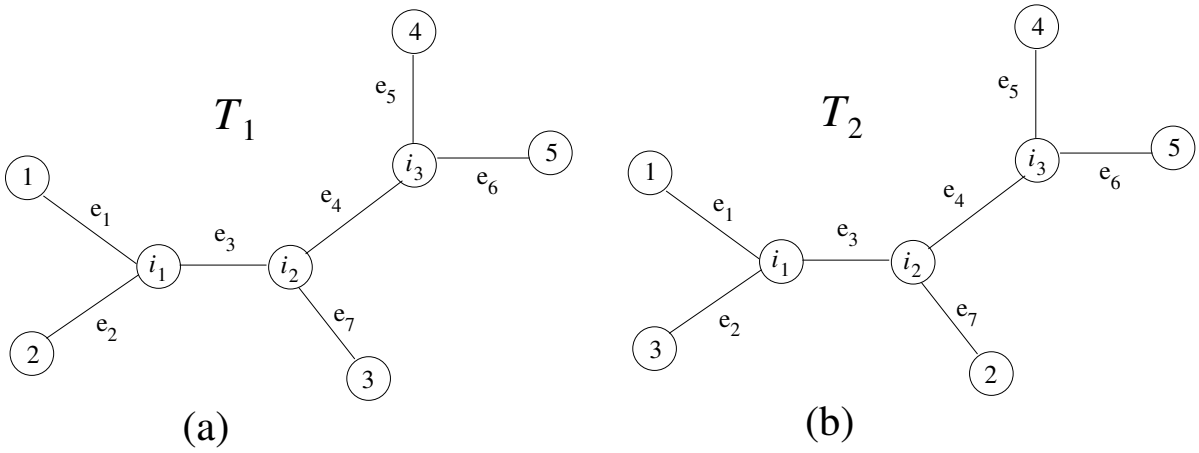


Figure 2.7: Two trees T_1 and T_2 with the same leaf sets $L_1 = L_2 = \{1, 2, 3, 4, 5\}$ but different topologies. (a) Tree T_1 has two bipartitions $\{1, 2\} \mid \{3, 4, 5\}$ and $\{1, 2, 3\} \mid \{4, 5\}$. (b) Similarly, tree T_2 has two bipartitions $\{1, 3\} \mid \{2, 4, 5\}$ and $\{1, 2, 3\} \mid \{4, 5\}$. Bipartition $\{1, 2\} \mid \{3, 4, 5\}$ is present only in T_1 whereas bipartition $\{1, 3\} \mid \{2, 4, 5\}$ occurs only in T_2. Bipartition $\{1, 2, 3\} \mid \{4, 5\}$ exists in both T_1 and T_2. Thus, the standardized Robinson and Foulds distance between T_1 and T_2 is 0.5 (2/4).

We examine two trees T_1 and T_2 with the same leaf sets $L_1 = L_2 = \{1, 2, 3, 4, 5\}$ but different topologies in Figure 2.7. Specifically, bipartition $\{1, 2\} \mid \{3, 4, 5\}$ is present only in T_1 whereas bipartition $\{1, 3\} \mid \{2, 4, 5\}$ occurs only in T_2. Bipartition $\{1, 2, 3\} \mid \{4, 5\}$ exists in both T_1 and T_2. Thus, the RF distance between T_1 and T_2 is 0.5 (2/4).

2.2.3.3 The accuracy of phylogenetic reconstruction methods

The Robinson and Foulds distance provides a means to measure the accuracy of phylogenetic tree reconstruction methods based on simulated data (see later sections), i.e. the ability to infer an underlying tree from data. In other words, the accuracy of an algorithm $\mathcal{A}$ can be considered as the average Robinson and Foulds distance between the reconstructed trees T^{rec} and the model trees T^{mod} used to generate the data sets. The smaller the average Robinson and Foulds distance is between the reconstructed trees T^{rec} the model trees T^{mod}, the higher is the topological accuracy of the tree reconstruction method $\mathcal{A}$.

Algorithm 2.3: Measure the accuracy of phylogeny reconstruction method $\mathcal{A}$

begin
  Set $RF_\Sigma \leftarrow 0$;
  for *s = 1 to #set* (the number of simulated data sets) **do**
    Create a random model n-leaf tree T_s^{mod} together with its edge lengths ;
    Create a random alignment $\mathbf{D}_s$ with length m according to the model tree T_s^{mod} and its edge lengths ;
    Reconstruct a tree T_s^{rec} using algorithm $\mathcal{A}$ for the alignment $\mathbf{D}_s$;
    Compute the Robinson and Foulds distance RF_s between the model tree T_s^{mod} and the reconstructed tree T_s^{rec} ;
    Set $RF_\Sigma \leftarrow RF_\Sigma + RF_s$;
  Return the average Robinson and Foulds distance $\overline{RF} = \frac{RF_\Sigma}{\#set}$;
end

In summary, the phylogenetic tree reconstruction method $\mathcal{A}_1$ is considered to give higher accuracy than the phylogenetic tree reconstruction method $\mathcal{A}_2$ with respect to the Robinson and Foulds distance if the average Robinson and Foulds distance $\overline{RF_1}$ of method $\mathcal{A}_1$ on simulated datasets $(\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_{\#set})$ is smaller than the average Robinson and Foulds distance $\overline{RF_2}$ of method $\mathcal{A}_2$ on the same simulated datasets.

2.2.3.4 Quartet trees

Quartet trees are the smallest informative structures of binary unrooted trees which present relationships among four different species. They are building blocks of quartet-based phylogenetic tree construction methods (Strimmer and von Haeseler, 1996; Willson, 1999; Ranwez and Gascuel, 2001). For a quartet of four different species A, B, C and D, three different bifurcating unrooted quartet trees are possible (see Figure 2.8). Since only three possible bifurcating unrooted quartet trees are available, the best quartet tree for four species can be usually determined easily. However, $\binom{n}{4}$ quartets are possible for n species. This number increases reasonably fast with n.

2.2.3.5 Local tree rearrangement operations

Local tree rearrangement operations play an important role in phylogenetic analysis. In addition to the Robinson and Foulds distance, they can be used to measure the difference

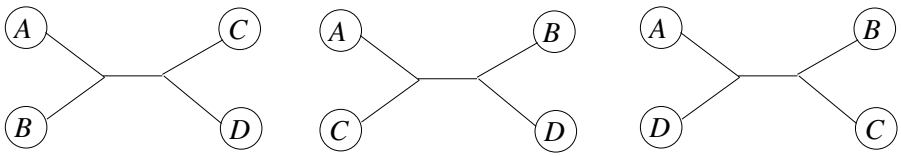


Figure 2.8: Three different bifurcating unrooted quartet trees for four different species A, B, C and D.

between phylogenetic trees (Waterman and Smith, 1978). More importantly, local tree rearrangement operations provide a simple and effective mean to travel through the space of possible phylogenetic trees for searching the best ones based on optimality criteria (Felsenstein, 2004). Furthermore, they are a crucial component of a majority phylogenetic tree reconstruction algorithms (see later sections).

We detail here the most widely used local tree rearrangement operation the so-called *nearest neighbor interchange* (NNI). Other operations such as *subtree pruning and regrafting* (SPR) and *tree bisection and reconnection* (TBR) can be found in Felsenstein (2004).

Regard an interior edge e with four distinct subtrees A, B, C and D in a binary unrooted tree T as depicted in Figure 2.9(a). An NNI operation with respect to the interior edge e simply exchanges two neighboring subtrees crossing interior edge e to obtain a tree in (b) or an alternative one in (c). If tree T' is derived from tree T by applying an NNI operation, tree T' is called a neighbor of T.

For each interior edge e in tree T one can examine two alternative neighbor trees. Since $(n-3)$ interior edges are possible in a binary unrooted tree T with n leaves, we can examine $2(n-3)$ neighbors of T by applying NNI operations.

2.3 Phylogenetic tree reconstructions

This section presents an overview over methods to reconstruct phylogenetic trees for a set of n contemporary species $S = \{s_1, s_2, \ldots, s_n\}$. Recall that these species are related by a binary rooted tree $T_r = (V, E)$ with leaf set L. Each leaf $l \in L$ represents a contemporary species $s \in S$ and interior nodes can be thought of as speciation events. The phylogenetic tree reconstruction methods can be divided into two classes: *character-based* methods and *distance-based* methods.

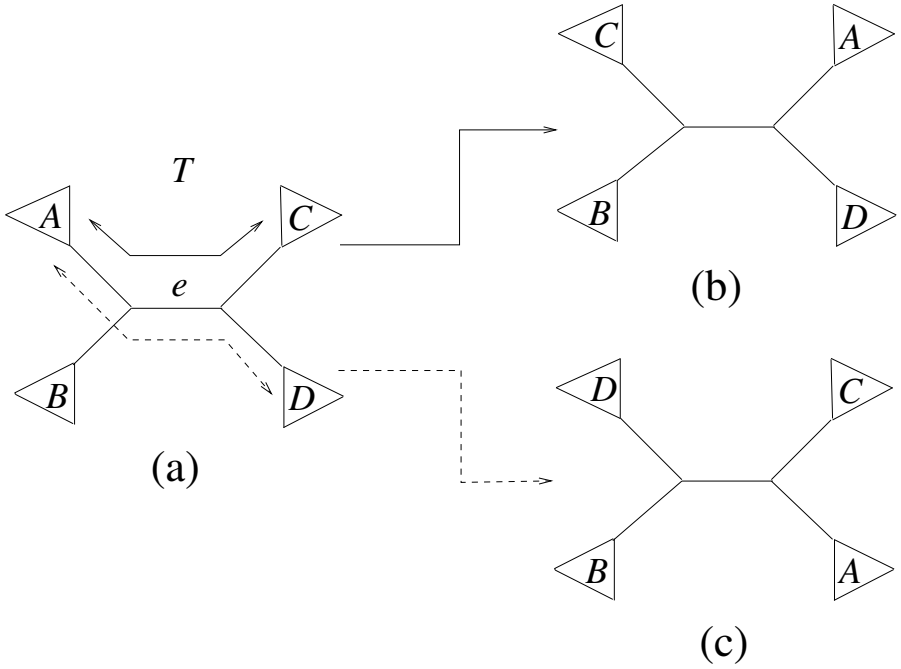


Figure 2.9: An illustration of nearest neighbor interchange (NNI) operations on a binary unrooted tree T. Consider an interior edge e with four distinct subtrees A, B, C and D in (a). An NNI operation exchanges two neighboring subtrees crossing interior edge e to obtain a tree in (b) or an alternative one in (c).

2.3.1 Character-based methods

Species $S = \{s_1, s_2, \ldots, s_n\}$ are represented by a multiple sequence alignment $\mathbf{D} = \{D_1, D_2, \ldots, D_m\}$ of n sequences with m sites. Note that two terms species and sequence can be used interchangeably if they are clear from the context. We denote with D_i^s the state of sequence $s \in S$ at site i. Typically, phylogenetic trees can be reconstructed using *maximum parsimony* methods or *maximum likelihood* approaches.

2.3.1.1 Maximum parsimony methods

Maximum parsimony methods are the simplest character-based methods to infer phylogenetic trees directly from the alignment $\mathbf{D}$. They try to find the phylogenetic tree minimizing the total number of evolutionary events required to explain the diversity of sequences in the alignment $\mathbf{D}$ (Edwards and Cavalli-Sforza, 1963; Fitch, 1971; Swofford *et al.*, 1996; Felsenstein, 2004). For molecular sequences, the evolutionary events are considered as nucleotide or amino acid substitutions.

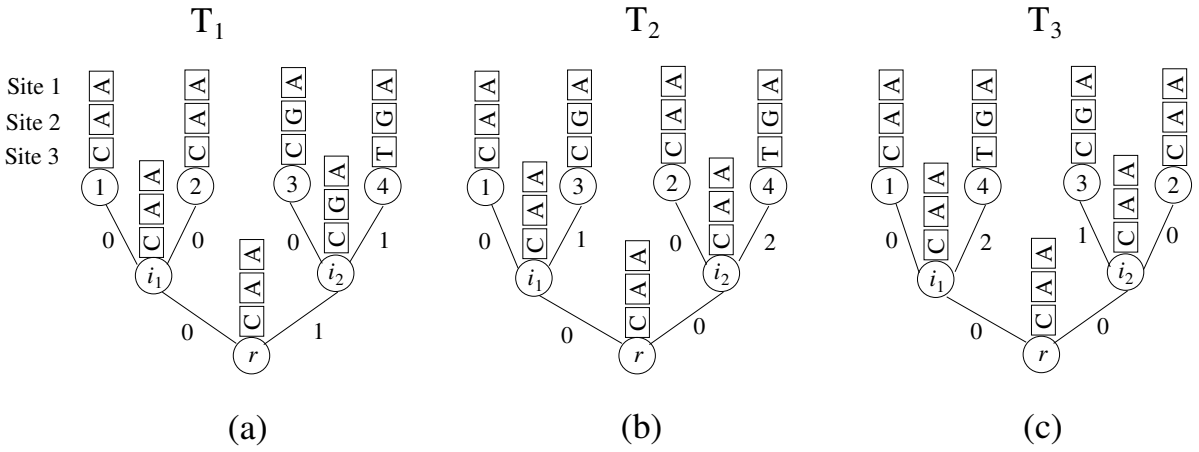


Figure 2.10: The number of evolutionary changes in tree T_1, T_2 and T_3 is 2, 3 and 3, respectively. T_1 is considered as the maximum parsimony tree.

We consider a given rooted T_r whose leaves are labeled by sequences. At this point, we assume that ancestor sequences at the root r as well as interior nodes of T_r are known (see Figure 2.10). The length $\ell(u, v)$ of edge $(u, v) \in E$ is defined as the number of changes between two sequences at u and v. Thus, the length ℓ_{T_r} of tree T_r is the total number of changes in tree T_r.

Figure 2.10(a) shows tree T_1 for four species in which each species is represented by a DNA sequence of length 3. The tree length ℓ_{T_1} is two, that is, tree T_1 needs two changes to explain the data. More costly, each tree T_2 and T_3 requires three changes leading to the data (see Figures 2.10(b) and 2.10(c)). Thus, T_1 is the maximum parsimony tree and considered as the best phylogenetic tree for these four species.

The first task of maximum parsimony methods is to infer ancestor sequences at the root r and interior nodes such that the length of tree T_r is minimized. Different methods to assign ancestor sequences have been proposed (Kluge and Farris, 1969; Farris, 1970; Fitch, 1971; Sankoff, 1975).

Although the minimum number of changes of a given tree T_r can be computed efficiently, searching the maximum parsimony tree for n species is an NP-complete problem (Graham and Foulds, 1982). Notoriously, there might exist many most parsimonious phylogenies for the same set of species (Swofford *et al.*, 1996).

Heuristic searches have been proposed to reduce computational burden including ratchet-based methods (Nixon, 1999), hill-climbing searches based on local tree rearrangement operations (Maddison, 1991; Goloboff, 1999; Quicke *et al.*, 2001), or divide and conquer techniques (Roshan *et al.*, 2004). Nowadays, PAUP* is the most popular

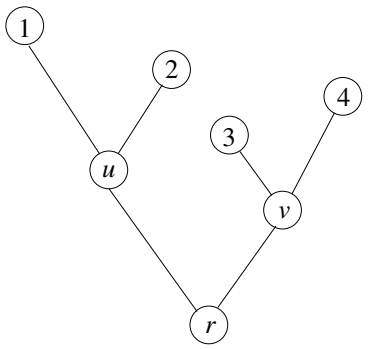


Figure 2.11: A rooted tree T_r with root r, two interior nodes u, v and four leaves $1, 2, 3, 4$. Tree T_r is the subject of the likelihood calculation of hypothesis H given data $\mathbf{D}$.

package to reconstruct phylogenies based on maximum parsimony methods (Swofford, 2002).

2.3.1.2 Maximum likelihood methods

Once the model of sequence substitution is specified, statistical approaches can be employed to make estimates of the phylogeny (Felsenstein, 2004). To date, maximum likelihood methods are widely used to infer the best phylogeny (Felsenstein, 1981a; Swofford *et al.*, 1996; Felsenstein, 2004, and references therein). Studies based on computer simulations show that maximum likelihood methods often give better results than maximum parsimony ones (Tateno *et al.*, 1994; Spencer *et al.*, 2005).

A simple application of maximum likelihood methods was described to estimate the genetic distances between sequences (see section 2.1.3.3). We specify three questions that naturally arise in the maximum likelihood-based phylogenetic tree inference: first, what is the likelihood function of a phylogeny; second, how can it be computed efficiently for a large number of sequences; and third, what is a proper general scheme to search the maximum likelihood phylogeny for a moderately large number of sequences.

Likelihood function

Recall that $S = \{s_1, s_2, \ldots, s_n\}$ is the set of n contemporary species. They are represented by a multiple sequence alignment $\mathbf{D} = \{D_1, D_2, \ldots, D_m\}$ of n sequences with m sites in which D_i^s denotes the state of sequence $s \in S$ at site i.

We let $H = (T_r, \ell, M, \mathbf{r})$ be a hypothesis in which these sequences have evolved from a common ancestor at the root r of tree T_r with length function ℓ according to models

of sequence substitution $M = (\boldsymbol{\pi}, \boldsymbol{R})$ and rate heterogeneity $\mathbf{r}$ (see section 2.1.3). In addition, let $\mathbb{H}$ be the hypothesis space which consists of all possible combinations of tree T_r, function length ℓ, models of sequence evolution M and rate heterogeneity $\mathbf{r}$.

The evolution process of the multiple sequence alignment $\mathbf{D}$ imposes three following assumptions:

- Sites across the multiple sequence alignment $\mathbf{D}$ evolve independently.
- Sites across the multiple sequence alignment $\mathbf{D}$ evolve according to the same hypothesis H.
- Lineages (edges of tree T_r) evolve independently.

The definition as well as computation of likelihood function $L(H)$ of the hypothesis H given the data $\mathbf{D}$ were specified by Felsenstein (1981a):

$$L(H) = \text{Prob}(\mathbf{D} \mid H). \tag{2.21}$$

Since sites are assumed to evolve independently, the likelihood $L(H)$ can be computed from the product of likelihoods at single sites:

$$L(H) = \prod_{i=1}^{m} \text{Prob}(D_i \mid H). \tag{2.22}$$

The computation of the likelihood of H for a site D_i is illustrated by an example in Figure 2.11. We denote by $\mathbf{d} = (d_1, d_2, d_3, d_4, d_u, d_v, d_r)$ a state vector in which $d_1 = D_i^1$, $d_2 = D_i^2$, $d_3 = D_i^3$, $d_4 = D_i^4$, and d_u, d_v, d_r are states at interior nodes u, v and the root r, respectively. The likelihood of H given $\mathbf{d}$ is calculated by

$$\begin{aligned} \text{Prob}(\mathbf{d} \mid T_r, \ell, M, r_i) \;=\; & \pi_{d_r}\, P_{d_r d_u}(\ell(r,u)\; r_i)\; P_{d_r d_v}(\ell(r,v)\; r_i) \times \\ & P_{d_u d_1}(\ell(u,1)\; r_i)\; P_{d_u d_2}(\ell(u,2)\; r_i) \times \\ & P_{d_v d_3}(\ell(v,3)\; r_i)\; P_{d_v d_4}(\ell(v,4)\; r_i). \end{aligned} \tag{2.23}$$

Since states d_u, d_v, d_r at the internal nodes u, v and the root r are unknown, the likelihood of H given single D_i is calculated over all $|\mathbb{A}|^3$ combinations of possible states at u, v and r ($|\mathbb{A}| = 4$ for nucleotide or $|\mathbb{A}| = 20$ for amino acid):

$$\text{Prob}(D_i \mid T_r, \ell, M, r_i) = \sum_{d_u \in \mathbb{A}} \sum_{d_v \in \mathbb{A}} \sum_{d_r \in \mathbb{A}} \text{Prob}(\mathbf{d} \mid T_r, \ell, M, r_i). \tag{2.24}$$

If substitution rate scaling factors across sites **r** are drawn from a Γ-distribution which is approximated by c categories $r_1, r_2, \ldots, r_c$ with the same prior probability $1/c$, then the likelihood L(H) can be computed (Yang, 1994) by

$$L(H) = \sum_{i=1}^{m} \left(\sum_{j=1}^{c} \frac{1}{c} \text{Prob}(D_i \mid T_r, \ell, M, r_j) \right). \tag{2.25}$$

Pruning Algorithm

The computation of Equation 2.24 is tedious because of $|\mathbb{A}|^{n-3}$ possible different state vectors **d** for n species. Felsenstein (1981a) introduced the so-called *pruning algorithm* to reduce the computational burden of the likelihood function using dynamic programing techniques.

We denote L_k for each node $k \in V$ the leaf set of the rooted subtree T_k of T_r. In addition, let $D_i(k) = \{D_i^j \mid j \in L_k\}$ be the state vector of leaf set L_k. We compute the probability $\text{Prob}(D_i(k) \mid k, d_k)$ of observing the state vector $D_i(k)$ at leaves of rooted subtree T_k upon condition that the state at root k of T_k is d_k. Particularly, if k is a leaf of T_r, then

$$\text{Prob}(D_i(k) \mid k, d_k) = \text{Prob}(D_i^k \mid k, d_k) = \begin{cases} 1 & \text{If } D_i^k = d_k \\ 0 & \text{If } D_i^k \neq d_k \end{cases} \tag{2.26}$$

Otherwise, the node k has two children u and v. The $\text{Prob}(D_i(k) \mid k, d_k)$ is computed from probabilities at the children u and v:

$$\begin{aligned} \text{Prob}(D_i(k) \mid k, d_k) \quad = \quad & \left(\sum_{d_u \in \mathbb{A}} P_{d_k d_u}(\ell(k,u)\, r_i)\, \text{Prob}(D_i(u) \mid u, d_u) \right) \times \\ & \left(\sum_{d_v \in \mathbb{A}} P_{d_k d_v}(\ell(k,v)\, r_i)\, \text{Prob}(D_i(v) \mid v, d_v) \right) \end{aligned} \tag{2.27}$$

We now derive $\text{Prob}(D_i(k) \mid k, d_k)$ for all nodes k and possible states $d_k \in \mathbb{A}$ using a postorder traversal starting from the root r of tree T_r:

Algorithm 2.4: Pruning algorithm

Data: The current node k.

Result: Compute probabilities at node k for all states $d_k \in \mathbb{A}$.

begin

  if *node k is a leaf* **then**

    foreach *state* $d_k \in \mathbb{A}$ **do**

      Compute $\text{Prob}(D_i(k) \mid k, d_k)$ using Equation 2.26 ;

  else

    Call the Pruning algorithm at child u ;

    Call the Pruning algorithm at child v ;

    foreach *state* $d_k \in \mathbb{A}$ **do**

      Compute $\text{Prob}(D_i(k) \mid k, d_k)$ using Equation 2.27 ;

end

Once the Pruning algorithm has been performed, the likelihood of hypothesis H given site D_i

$$L(D_i \mid H) = \text{Prob}(D_i \mid T_r, \ell, M, r_i) = \sum_{d_r \in \mathbb{A}} \pi_{d_r} \text{Prob}(D_i(r) \mid r, d_r) \tag{2.28}$$

Since the Pruning algorithm is based on a postorder traversal and the number of character states is constant, its complexity is only $O(n)$.

Maximum likelihood principle

The aim of maximum likelihood estimate is to find the hypothesis H^* with the highest likelihood, that is, H^* makes observed data $\mathbf{D}$ most likely (Felsenstein, 1978). Formally, the maximum likelihood hypothesis

$$H^* = \text{argmax}_{H \in \mathbb{H}}\{L(H)\}. \tag{2.29}$$

Since models of sequence evolution M are typically assumed to be reversible, an unrooted tree T could be rooted at any point in order to compute its likelihood using Equation 2.28 (*pulley principle*, Felsenstein, 1981a). Consequently, unrooted trees T are examined instead of rooted trees to infer the maximum likelihood hypothesis H^*.

Algorithm 2.5: A general scheme to infer the maximum likelihood hypothesis

Data: A multiple sequence alignment $\mathbf{D}$.

Result: The maximum likelihood hypothesis $H^* = (T^*, \ell^*, M^*, \mathbf{r}^*)$.

begin

Reconstruct an initial tree T^* and estimate its length function ℓ^* ;

Estimate parameters of models of sequence evolution M^* and rate heterogeneity $\mathbf{r}^*$ based on data $\mathbf{D}$, tree T^* together with length function ℓ^* ;

Set $maxLikelihood \leftarrow L(H^*)$;

foreach $T \in \mathbb{T}$ (the space of unrooted trees) **do**

Estimate length function ℓ of tree T ;

if $L(T, \ell, M^*, \mathbf{r}^*) > maxLikelihood$ **then**

$maxLikelihood \leftarrow L(T, \ell, M^*, \mathbf{r}^*)$;

$T^* \leftarrow T$;

$\ell^* \leftarrow \ell$;

end

Since estimates of parameters of models M^* and $\mathbf{r}^*$ do not depend significantly on the trees (Sullivan *et al.*, 2005), these parameters are typically estimated once based on an initial tree and fixed during the search to reduce computational expense. The initial tree is typically constructed by a fast and reasonably accurate method such as Neighbor-Joining (Saitou and Nei, 1987). Because parameters of models M^* and $\mathbf{r}^*$ are fixed, searching the maximum likelihood hypothesis H^* now becomes the task of finding the maximum likelihood unrooted tree T^* as well as its edge length function ℓ^*.

The edge lengths of an unrooted tree T are optimized using numerical analyzes such as the Brent's method (Brent, 1973) or Newton-Raphson's method (e.g. Press *et al.*, 2002).

Finding the maximum likelihood hypothesis H^* is an NP-hard problem (Chor and Tuller, 2005). Many heuristic searches have been proposed to obtain a best possible hypothesis H^* in practical time.

Felsenstein (1981a) implemented the *DNAml* program which applied local tree rearrangement operations to search the maximum likelihood tree for nucleotide data. The DNAml program was improved by Olsen *et al.* (1994), namely *fastDNAml*. More recently, Stamatakis *et al.* (2005) further improved fastDNAml by introducing the so-called Randomized Accelerated Maximum Likelihood technique. Stamatakis (2004) also applied a simulated annealing search to infer the maximum likelihood tree.

Besides subtree rearrangement operations-based approaches, genetic algorithms have been proposed (Matsuda, 1996; Lemmon and Milinkovitch, 2002). For example, Lemmon and Milinkovitch (2002) implemented *MetaPIGA* software based on a meta population genetic algorithm. Another attempt to reduce computation time is the construction of quartet trees, which are subsequently used to puzzle an overall tree such as Quartet Puzzling (Strimmer and von Haeseler, 1996).

Moreover, attempts to parallelize tree-construction programs have been introduced to further reduce the running time of the analysis (Olsen *et al.*, 1994; Charleston, 2001; Brauer *et al.*, 2002; Schmidt *et al.*, 2002, 2003; Stamatakis and Ludwig, 2004; Keane *et al.*, 2005; Minh *et al.*, 2005).

By now, PHYML seems to be faster than other methods (Guindon and Gascuel, 2003). The method searches the maximum likelihood tree using hill-climbing techniques based on nearest neighbor interchange operations. It is not surprising that multiple optimal points might exist on the likelihood surface for phylogenetic trees (Steel, 1994; Chor *et al.*, 2000). Consequently, the PHYML search is likely to get stuck at a local optimal point on the likelihood surface. Approaches which are able to visit as many as possible optimal points on the likelihood surface in practical time is our desire (see **Chapters** 3 and 4).

2.3.2 Distance-based methods

Computer simulation studies show that character-based approaches like maximum likelihood methods tend to give high accuracy (Guindon and Gascuel, 2003, and **Chapter** 4). Unfortunately, they typically require huge computation times. To date, distance-based methods introduced by Cavalli-Sforza and Edwards (1967) and Fitch and Margoliash (1967a) appear most appropriate to reconstruct large phylogenies for thousands of sequences. These methods are a compromise between computational speed and accuracy. They run typically in $O(n^3)$ time for n sequences (Saitou and Nei, 1987; Gascuel, 1997; Bruno *et al.*, 2000) or in $O(n^2)$ time for recently suggested approaches (Desper and Gascuel, 2002; Csürös, 2002).

The prerequisite of distance-based methods to construct phylogenies for the species S is the pairwise distance matrix $\mathbb{D} = \{\mathcal{D}(u, v)\}$ where $\mathcal{D}(u, v)$ is the distance between two species $u, v \in S$ (typically, genetic distances as estimated in section 2.1.3.3).

Given a tree $T = (V, E)$ together with its edge length function ℓ, we can derive the pairwise distance matrix $\mathbb{D}_{(T,\ell)} : S \times S \mapsto R_+$ between species according to tree T and

length function ℓ. More specifically,

$$\mathcal{D}_{(T,\ell)}(u,v) = \sum_{e \in p(\varphi(u),\varphi(v))} \ell(e) \quad \text{for all } u,v \in S. \tag{2.30}$$

Definition 12 *A distance matrix* $\mathbb{D}$ *is* **additive** *if and only if it satisfies the* **four-point condition** *(Buneman, 1971): for all quartets* $\{u,v,w,x\}$*,*

$$\mathcal{D}(uv) + \mathcal{D}(wx) \leq max\{\mathcal{D}(uw) + \mathcal{D}(vx), \mathcal{D}(ux) + \mathcal{D}(vw)\}. \tag{2.31}$$

If $\mathbb{D}$ is additive, there exists a length function ℓ such that

$$\mathcal{D}(uv) = \mathcal{D}_{(T,\ell)}(u,v) \quad \text{for all } u,v \in S. \tag{2.32}$$

In other words, the tree T and the length function ℓ for the additive distance matrix $\mathbb{D}$ can be constructed easily in $O(n^2)$ time for n species (Hein, 1989, and references therein).

Unfortunately, distance matrices $\mathbb{D}$ are typically not additive due to stochastic errors in estimating genetic distances between sequences. Thus, in the following arbitrary distance matrices are considered (van de Peer, 2003). Distance-based methods construct tree $\hat{T}$ together with length function $\hat{\ell}$ such that the pairwise distance $\mathcal{D}_{(\hat{T},\hat{\ell})}(u,v)$ according to tree $\hat{T}$ and length function $\hat{\ell}$ is as close as possible to the pairwise distance $\mathcal{D}(u,v)$ for all $u,v \in S$.

Felsenstein (2004) motivated distance-based methods as follows:

> "The general idea of distance-based methods seems as if they would not work very well: calculate a measure of the distance between each pair of species, and then find a tree that predicts the observed set of distances as closely as possible. This leaves out all information from higher-order combinations of character states, reducing the data matrix to a simple table of pairwise distances. One would think that this must leave out so many of subtleties of the data that it could not possibly do a reasonable job of making an estimate of the phylogeny.
>
> Computer simulation studies show that the amount of information about the phylogeny that is lost in doing this is remarkably small. The estimates of the phylogeny are quite accurate. Apparently, it is not common for evolutionary processes (at least not the simple models that we use for them) to leave a trace in high-order combinations of character states without also leaving almost the same information in the pairwise distances between the species."

In principle, distance-based methods are divided into three classes: *least square methods*, *minimum evolution approaches*, and *clustering algorithms*.

2.3.2.1 Least square methods

Least square methods are some of the best statistically justified distance-based approaches (Felsenstein, 2004). Theoretically, they construct a tree $\hat{T}$ together with length function $\hat{\ell}$ such that the total square discrepancy between distances $\mathcal{D}(u,v)$ and $\mathcal{D}_{(\hat{T},\hat{\ell})}(u,v)$ over all pairs of species (u,v) is minimized (e.g. Cavalli-Sforza and Edwards, 1967).

Given a tree topology T together with length function ℓ, we denote $\Delta_{(T,\ell)}$ the sum of the square differences between $\mathcal{D}(u,v)$ and $\mathcal{D}_{(T,\ell)}(u,v)$ over all pairs of species (u,v):

$$\Delta_{(T,\ell)} = \sum_{u=1}^{n}\sum_{v=1}^{n} w(u,v)(\mathcal{D}(u,v) - \mathcal{D}_{(T,\ell)}(u,v))^2 \tag{2.33}$$

where $w(u,v)$ is the weight of pair (u,v). This weight is assigned differently in different methods. Particularly, Cavalli-Sforza and Edwards (1967) assumed that all species pairs have the same weight, i.e. $w(u,v) = 1$ for all $u, v \in S$. In contrast, Fitch and Margoliash (1967a) proposed that each species pair (u,v) has its own weight $w(u,v) = \frac{1}{\mathcal{D}(uv)}$, which means the deviation in the two matrices of closely related species achieve more weights than the deviation in the two matrices of distantly related species. Similarly, a weight $w(u,v) = \frac{1}{\mathcal{D}(uv)^2}$ for species pair (u,v) is suggested by Beyer *et al.* (1974).

Mathematically, least square methods search the tree $\hat{T}$ together with length function $\hat{\ell}$ such that $\Delta_{(T,\ell)}$ is minimized:

$$(\hat{T},\hat{\ell}) = \text{argmin}_{(T,\ell)}\{\Delta_{(T,\ell)}\}. \tag{2.34}$$

Given a tree T, the first task of least square methods is to estimate the length function ℓ of tree T in oder to minimize $\Delta_{(T,\ell)}$. Although the task is solved by algebraic analysis (Rzhetsky and Nei, 1993), searching the least square tree is an NP-complete problem (Day and David, 1986). Heuristic methods to construct least square phylogenies were implemented in packages such as PHYLIP (Felsenstein, 1993) or PAUP* (Swofford, 2002).

2.3.2.2 Minimum evolution methods

Nowadays, minimum evolution (ME) approaches are the most widely used distance-based methods to infer phylogenetic trees. They are closely related to least square methods in the sense that edge length function ℓ for a given tree T is typically estimated using the least square principle (Rzhetsky and Nei, 1993; Desper and Gascuel, 2002). However, their objective function is to construct the tree $\hat{T}$ as well as its edge lengths such that the length of the tree $\hat{T}$ is the shortest (Kidd and Sgaramella-Zonta, 1971).

Precisely, minimum evolution methods search the tree $\hat{T}$ together with its edge length function $\hat{\ell}$ obeying the following condition:

$$\hat{\ell}_{\hat{T}} = \min\{\ell_T \mid T \in \mathbb{T}\}. \tag{2.35}$$

Note that edge lengths of trees $T \in \mathbb{T}$ cannot simply be set to zero, but must be estimated according to the pairwise distance matrix $\mathbb{D}$.

The theoretical foundation of ME methods was given by Rzhetsky and Nei (1993). They proved that if the pairwise distance matrix $\mathbb{D}$ is estimated unbiasedly, then the length of the true tree is the shortest.

Since minimum evolution methods evaluate the length function ℓ for a given tree T using the least square principle, determining the minimum evolution tree $\hat{T}$ together with its length function $\hat{\ell}$ corresponding to a distance matrix $\mathbb{D}$ is also an NP-complete problem (Felsenstein, 2004). Hence, heuristic searches have been proposed to reduce the computational burden (Saitou and Nei, 1987; Rzhetsky and Nei, 1993; Kumar, 1986; Gascuel, 1997; Bruno *et al.*, 2000; Desper and Gascuel, 2002).

To date, Neighbor-Joining is the most popular method to approximately reconstruct the minimum evolution phylogeny (Saitou and Nei, 1987). More recently, Desper and Gascuel (2002) applied a hill-climbing search strategy based on nearest neighbor interchange operations to find the minimum evolution tree. The method seems to be better than existing approaches in terms of both accuracy and running time.

2.3.2.3 Clustering methods

The third major class of distance-based methods is formed by the clustering algorithms (Hartigan, 1975). In contrast to the least square methods and the minimum evolution approaches, clustering algorithms do not impose explicitly a global objective function that needs to be optimized. They rather group sequences (or taxa) iteratively to reconstruct a distance-based phylogenetic tree.

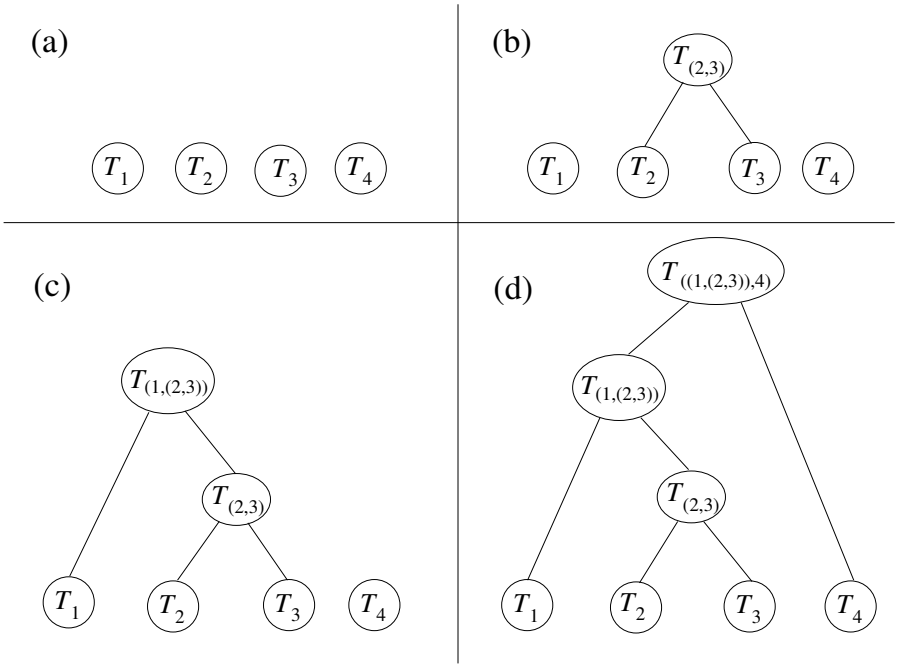


Figure 2.12: An illustration of clustering algorithms to build a tree for four species by agglomerating sequences iteratively.

Among these, UPGMA (unweighted pair-group method with arithmetic averages) is the most simplest method to infer phylogenies with the constraint that a molecular clock is assumed on the evolutionary process (Sneath and Snokal, 1973). Other clustering approaches have been proposed to relax the molecular clock assumption (Farris, 1977; Klotz *et al.*, 1979; Li, 1981; Saitou and Nei, 1987; Gascuel, 1997; Bruno *et al.*, 2000).

Figure 2.12 illustrates a tree construction for four species using a general scheme of clustering algorithms. First, four rooted subtrees T_1, T_2, T_3 and T_4 each corresponds to a species are initialized in 2.12(a). Then, they are iteratively clustered together to build a final 4-species tree. Specifically, rooted subtrees T_2 and T_3 which are determined to be a pair of neighbors are grouped into a new rooted subtree $T_{(2,3)}$ in 2.12(b). The determination of neighbor pair depends on methods (Sneath and Snokal, 1973; Farris, 1977; Li, 1981; Saitou and Nei, 1987). For example, UPGMA considers two rooted subtrees with the smallest distance as a pair of neighbors. After that, rooted subtree T_1 and $T_{(2,3)}$ are agglomerated into a new 3-species rooted subtree $T_{(1,(2,3))}$ in 2.12(c). Finally, rooted tree $T_{((1,(2,3)),4)}$ for 4 species is created by grouping T_4 together with $T_{(1,(2,3))}$ in 2.12(d).

Remarkably, the Neighbor-Joining (NJ) method which builds approximately the min-

imum evolution phylogeny works by clustering (Saitou and Nei, 1987). Variants of NJ algorithm e.g. BIONJ and Weighbor have been proposed to boost the accuracy of NJ (Gascuel, 1997; Bruno *et al.*, 2000). Since the complexity of NJ algorithm is $O(n^3)$, it consumes a reasonable runtime to construct phylogenies for large data sets with thousands of sequences.

We are interested in developing a new clustering method which not only gives a higher accuracy compared to existing clustering algorithms but also reduces their computational expense (see **Chapter** 5).

2.3.3 Finding the best tree by heuristic methods

Searching the best phylogeny exhaustively based on a given optimality criterion, e.g. maximum parsimony, maximum likelihood, minimum evolution is computationally expensive. Mathematically, searching maximum parsimony or minimum evolution phylogenies is NP-complete (Graham and Foulds, 1982; Day and David, 1986). More difficultly, determining the maximum likelihood tree is an NP-hard problem (Chor and Tuller, 2005). To overcome this computational burden, heuristic methods have been proposed to construct phylogenies in practical time (Felsenstein, 2004, and references therein). We must note, that heuristic approaches cannot guarantee to find the best tree(s).

In the following, we introduce two commonly used heuristic methods to construct phylogenies: *hill-climbing* search and *stepwise addition tree reconstruction* method.

2.3.3.1 Hill climbing search

Hill climbing is an intuitive heuristic search strategy to find the best solution with respect to an optimality criterion by improving the current solution in a step-by-step manner. The method is illustrated in Figure 2.13 where one attempts to climb up the highest point from a starting point. One always moves from the current position to a higher position until reaching a locally highest point.

Operations which allow changing from the current position to another are called *traversal operations*. In the context of phylogenetic tree reconstruction, one could employ local subtree rearrangement operations such as nearest neighbor interchange as proper traversal operations.

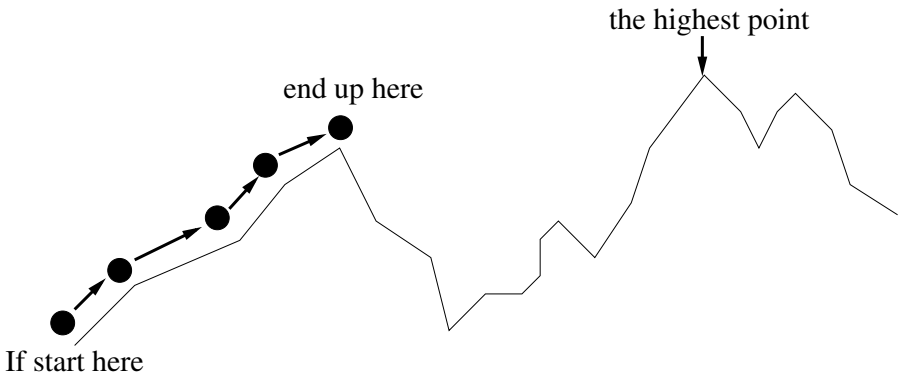


Figure 2.13: An illustration of hill climbing algorithm. One attempts to climb up the highest point by moving from the current position towards a higher position until reaching a locally highest point. The strategy does not guarantee that the highest position is reached.

Algorithm 2.6: A general scheme of hill climbing algorithms

Data: A set of traversal operations $\mathbf{Op} = \{Op_1, \ldots, Op_m\}$, a quality function f.
Result: The best found tree T^*.
begin
  Reconstruct an initial tree T^* ;
  repeat
    Find the best neighbor T' of T^* by applying traversal operations in **Op** ;
    if $f(T) > f(T^*)$ **then**
      $isMoveable \leftarrow true$;
      $T^* \leftarrow T'$;
    else
      $isMoveable \leftarrow false$;
  until *isMoveable*;
end

Obviously, the hill climbing search terminates when no better neighbor of the current best tree T^* is found. Since the hill climbing search accepts only movements which increase the quality of the current tree, it is likely to get stuck at a locally best tree. To overcome this limitation, one could repeat the hill-climbing search several times from different starting trees. Finally, the tree with highest score with respect to an optimality criterion is considered as the best found tree.

Hill climbing methods to search maximum parsimony, maximum likelihood, or minimum evolution phylogenies have been implemented in PAUP* package (Swofford, 2002),

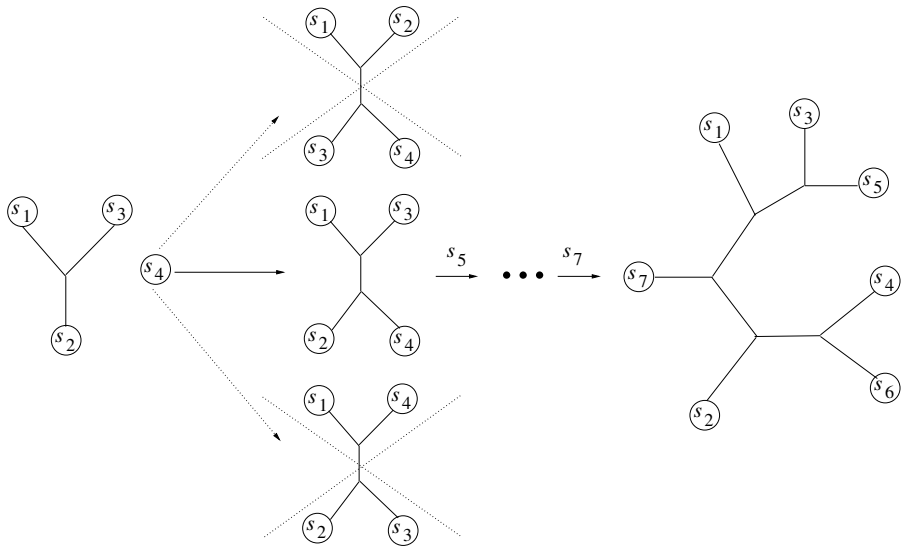


Figure 2.14: The tree construction for seven species $S = \{s_1, s_2, \ldots, s_7\}$ based on stepwise addition. Starting from a unique tree $T(s_1, s_2, s_3)$ for three species s_1, s_2, and s_3, the fourth species s_4 is placed at an edge of the 3-species tree $T(s_1, s_2, s_3)$ that obtains the best 4-species tree $T(s_1, \ldots, s_4)$ according to an optimality criterion. Similarly, species s_5, s_6 and s_7 are sequentially inserted into the current tree to reconstruct a final 7-species tree $T(s_1, \ldots, s_7)$.

fastDNAml (Olsen *et al.*, 1994), PHYML (Guindon and Gascuel, 2003), or RAxML (Stamatakis *et al.*, 2005).

2.3.3.2 Stepwise addition tree construction

The hill climbing search depends significantly on the initial tree in terms of both quality and runtime. The so-called *stepwise addition* strategy is introduced to build a single tree which can serve properly as a starting tree in the hill climbing search (Felsenstein, 2004, and references therein).

In general, the method starts from a unique 3-species tree and sequentially inserts remaining species into the current tree to construct a final n-species tree for n species $S = \{s_1, s_2, \ldots, s_n\}$.

Figure 2.14 illustrates the tree construction for seven species $S = \{s_1, s_2, \ldots, s_7\}$ based on stepwise addition strategy. First, a unique tree $T(s_1, s_2, s_3)$ for three species s_1, s_2 and s_3 is initialized. Then, the fourth species s_4 is placed at an edge of the 3-species tree $T(s_1, s_2, s_3)$ that obtains the best 4-species tree $T(s_1, \ldots, s_4)$ according to an optimality

criterion. Similarly, species s_5, s_6 and s_7 are sequentially inserted into the current tree to reconstruct a final 7-species tree $T(s_1, \ldots, s_7)$.

In principle, to insert new species $s_{(t+1)}$ into the current t-species tree $T(s_1, \ldots, s_t)$, one examines all possible $(t+1)$-species trees $T(s_1, \ldots, s_{(t+1)})$ obtained from inserting new species $s_{(t+1)}$ into the current t-species tree $T(s_1, \ldots, s_t)$. The edge corresponding to the $(t+1)$-species tree with highest score with respect to an optimality criterion is considered as the best place to insert the new species $s_{(t+1)}$. Note that the best $(t+1)$-species tree $T(s_1, \ldots, s_{(t+1)})$ in turn will serve as the current tree for the insertion of the next species.

The optimality criterion can be maximum likelihood, maximum parsimony, or minimum evolution. Other widely used criteria are based on quartet trees. The tree which is supported by maximum number of quartet trees is considered as the best one (Strimmer and von Haeseler, 1996; Willson, 1999; Ranwez and Gascuel, 2001).

Different insertion orders of species may result in different final n-species trees. In other words, we might construct different trees by using the same stepwise addition algorithm, but adding species to the tree in different orders.

3 Phylogenetic navigator

This chapter presents a novel method the so-called *phylogenetic navigator* (PhyNav) to search the best phylogenies with respect to an optimality criterion. First, the definition of the so-called *minimal k-distance subsets* is introduced. These subsets are a cornerstone of the PhyNav algorithm to elucidate the space of phylogenetic trees efficiently. Then, the efficiency of PhyNav and other methods in terms of both accuracy and runtime is examined on simulated as well as real datasets.

3.1 Minimal k-distance subsets

Recall that $S = \{s_1, s_2, \ldots, s_n\}$ is the set of n species which are related by a tree T with leaf set L. Each leaf $l \in L$ represents a species $s \in S$. The terms species and sequence are used interchangeably if they are clear from the context.

The *topological distance* $d_p(s, s')$ in tree T is the number of branches on the path $p(s, s')$ from s to s'. We now introduce the concept of *k-distance representatives*:

Definition 13 *A sequence s is said to be a* **k-distance representative** *for a sequence s' in a tree T if and only if their topological distance $d_p(s, s')$ in T is smaller or equal to $k \geq 0$.*

Intuitively, the smaller the value of k is the better a sequence s represents sequence s', and vice versa. The k-distance representative sequence concept is now used to introduce minimal k-distance subsets:

Definition 14 *A subset S_k of sequences is called a* **minimal k-distance subset** *of an n-sequence set S if and only if the following two conditions hold:*

1. *For each sequence $s \in S$, there exists a sequence $s' \in S_k$ such that the sequence s' is a k-distance representative for the sequence s.*

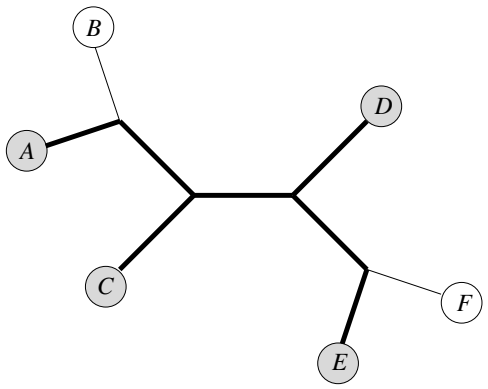


Figure 3.1: An unrooted bifurcating tree of 6 species $\{A, B, C, D, E, F\}$. The bold tree is the scaffold with minimal 2-distance subset $\{A, C, D, E\}$.

2. *If we remove any sequence s' from S_k, S_k will violate the first condition. That means, the subset cannot be reduced any further.*

The idea behind minimal k-distance subsets is that the phylogenetic information in the sequence subset S_k represents phylogenetic information from the whole set. According to our experience $k = 3$ is a good choice because it prevents the deletion of too many sequences as well as the removal of sequences that provide information to bridge long paths between distantly related subtrees.

A sequence $\bar{s} \notin S_k$ is then called a *remaining sequence*. The set $\overline{S_k} := S \backslash S_k$ of all such sequences, which remain to be added to S_k to obtain the full set S, is called *remaining set*.

Since $|S_k| \leq |S|$, the subtree $T(S_k)$ from subset S_k can usually be constructed in less time than the full tree. This subtree is used as a scaffold to build a full tree containing all sequences by adding all sequences $\bar{s} \in \overline{S_k}$. Note that there exist many minimal k-distance subsets and each can be determined in time of $O(n^2)$. For example, sequences A and B in the tree in Figure 3.1 are 2-distance representative of each other, as are E and F. The sequence subsets $\{A, C, D, E\}$, $\{A, C, D, F\}$, $\{B, C, D, E\}$ and $\{B, C, D, F\}$ are minimal 2-distance subsets of the full set $\{A, B, C, D, E, F\}$.

3.2 The PhyNav algorithm

The phylogenetic navigator algorithm is a five-step procedure: (1) the *Initial step*, (2) the *Navigator step*, (3) the *Disembarking step*, (4) the *Comparative step*, and (5) the

Stop step. We could use the algorithm with any objective function, e.g. maximum parsimony, maximum likelihood, to create a list of possible optimal trees. According to the objective function the best tree found is taken as the inferred phylogeny. The PhyNav is detailed in algorithm 3.1.

Algorithm 3.1: Phylogenetic Navigator (PhyNav)

begin

Initial step: We employ some fast tree reconstruction method to create an initial tree. To that end, PhyNav uses the BIONJ (Gascuel, 1997) an improved Neighbor-Joining algorithm (Saitou and Nei, 1987) with the pairwise evolutionary distances and a fast nearest neighbor interchange (NNI) operation as described by Guindon and Gascuel (2003) to create the initial tree. This tree is then called the currently best tree and denoted as T_{best}, with log-likelihood ℓ^{log}_{best}. The currently best tree T_{best} is used to construct the k-distance subsets.

Navigator step: Find a minimal k-distance subset S_k and constructs the corresponding subtree $T(S_k)$. From the minimal k-distance subset S_k the corresponding subtree $T(S_k)$ could be created by several tree reconstruction methods. In PhyNav, $T(S_k)$ is created by optimizing the subtree T_{sub} of T_{best} induced by the leaves in S_k using NNI operations.

Disembarking step: Construct the whole tree T based on the scaffold tree $T(S_k)$ using the k-distance information. To this end, PhyNav inserts the remaining sequences into the scaffold as follows: (1) assign T by $T(S_k)$, (2) insert each remaining sequence $\bar{s} \in \overline{S_k}$ into an external branch e of T such that the corresponding leaf s_e adjacent to e is a k-distance representative for $\bar{s}$. If there are more than one external branches possible one branch is selected randomly, (3) apply NNI operations to T to compensate for incorrect placements. The new resulting whole tree is called intermediate tree, denoted by $T_{\text{intermediate}}$.

Comparative step: If the log-likelihood $\ell^{log}_{\text{intermediate}} > \ell^{log}_{\text{best}}$, then set $T_{\text{best}} \leftarrow T_{\text{intermediate}}$ and update the log-likelihood.

Stop criterion: If the number of optimization steps is less or equal than a pre-defined number of total optimization steps $\#step$, go to the Navigator step, otherwise stop and output T_{best}.

end

It cannot be guaranteed that $T(S_k)$ determined in the *Navigator step* is the optimal tree for S_k due to the use of heuristics. Even if $T(S_k)$ is the best tree it does not guarantee that tree T will be the optimal full tree. Hence, the Navigator, Disembarking, and Comparative steps are repeated several times. Then the program stops and the best tree T_{best} is considered as the final phylogenetic n-tree.

The maximum likelihood criterion is used in this description as an objective function, however the algorithm is not restricted to maximum likelihood, any tree reconstruction method with an objective function to minimize or maximize will work in the presented approach.

All the intermediate trees found from PhyNav during one search run of the tree space are eventually summarized into a majority rule consensus tree with the frequencies of the groupings. The consensus tree reveals complementary information of most frequent groupings found in the collection of similar high likelihood trees.

3.3 The efficiency of PhyNav

To measure the accuracy and the time-efficiency of PhyNav we reconstructed phylogenetic trees from simulated as well as biological datasets. The results are compared to the results of other programs, in particular, Weighbor version 1.2 (Bruno *et al.*, 2000); and PHYML version 2.1 (Guindon and Gascuel, 2003); Computing times were measured on a Linux PC Cluster with 2.0 GHz CPU and 512 MB RAM.

3.3.1 Simulated datasets

To evaluate the accuracy we performed simulations. To simulate realistic datasets we performed the simulations on a tree topology reconstructed from a real dataset. To that end an elongation factor (EF-1α) dataset with 43 sequences was used. The dataset as well as the tree was obtained from TreeBase (`http://www.treebase.org`, accession number S606, matrix accession number M932). The branch lengths of the tree topology were inferred using the TREE-PUZZLE package version 5.1 (Strimmer and von Haeseler, 1996; Schmidt *et al.*, 2002).

Based on that tree topology datasets were simulated using Seq-Gen version 1.2.6 (Rambaut and Grassly, 1997); assuming the Kimura 2-parameter model with an transition:transversion ratio of 2.0 (Kimura, 1980). 1,000 datasets each were simulated with sequence lengths of 700 and 1000 bp.

Table 3.1: Results for the simulated datasets: (a) percentage of correctly reconstructed trees, (b) average Robinson-Foulds distance between the 'true tree' and the reconstructed trees, and (c) average runtime of tree reconstruction (1000 simulations per parameter setting).

	Weighbor	PHYML	PhyNav
700 bp	2.4	12.3	13.1
1000 bp	9.6	33.7	33.9

(a) Percentage of correct trees.

	Weighbor	PHYML	PhyNav
700 bp	.140	.076	.073
1000 bp	.086	.039	.038

(b) Average Robinson-Foulds distance.

	Weighbor	PHYML	PhyNav
700 bp	$3s$	$7s$	$52s$
1000 bp	$4s$	$9s$	$66s$

(c) Average runtime.

The trees for simulated datasets were reconstructed using PhyNav, Weighbor and PHYML. All programs were run with default options. The evolutionary model and its parameters were set to the simulation parameters. The PhyNav options were set to $\#step = 5$ repetitions and $k = 3$.

The results of the tree reconstructions were compared using two different methods. First the percentage of correctly reconstructed tree topologies was derived for each program and sequence length. Moreover, the average Robinson-Foulds distance (Robinson and Foulds, 1981) was employed to measure the variability of the results for each program (see section 2.2.3.2).

Tables 3.1(a) and 3.1(b) display the results for PhyNav, PHYML, and Weighbor. Both tables show that Weighbor is out-performed by both PHYML and PhyNav. PHYML and PhyNav perform similarly well, both in the percentage of correctly reconstructed trees as well as in their average Robinson-Foulds distance to the model tree.

However, PhyNav shows slightly better values for all analyzes. Table 3.1(c) shows that the computing time for the construction of a phylogeny with 43 sequences is not really an issue for the PHYML, Weighbor, and PhyNav methods.

3.3.2 Biological datasets

The PhyNav algorithm was applied to large biological datasets to test its efficiency on real datasets. Three datasets have been obtained from the PANDIT database version 7.6 (`http://www.ebi.ac.uk/goldman-srv/pandit/`; Whelan *et al.*, 2003). The first dataset consists of 76 Glyceraldehyde 3-phosphate dehydrogenase sequences with an alignment length of 633 bp (PF00044), the second of 105 sequences from the ATP synthase alpha/beta family (1821 bp, PF00006), and the last of 193 sequences with Calporin homology with an alignment of 465 bp (PF00307).

Since the true tree is usually not known for real datasets, the Robinson-Foulds distance cannot be used to measure the efficiency of algorithms. Therefore the likelihood value of the reconstructed trees is used to compare the methods.

Since Weighbor does not use likelihoods we only compare PHYML and PhyNav from the methods above. Note that Weighbor already was outperformed in the simulation study. Additionally we wanted to use MetaPIGA (Lemmon and Milinkovitch, 2002), another method for large datasets based on a genetic algorithm. Unfortunately the program crashed on all three datasets. Thus, only PHYML and PhyNav were used for comparison.

As explained above we use the likelihood values of the reconstructed trees to compare the efficiency of the two programs. According to the maximum likelihood framework (cf. for example Felsenstein, 1981a) the tree with the higher likelihood value represents the more likely tree.

The log-likelihood values are given in Table 3.2(a). These results show that PhyNav always find a tree with a higher likelihood. The increase of the log likelihood ranged from 39 up to 343 units.

However, as Table 3.2(b) shows, the price to pay for better likelihood trees is an increase in computing time. Each single repetition in the algorithm has a time consumption comparable to the one run of PHYML. Nevertheless, the substantial increase of the likelihoods might well justify that this effort is worthwhile, since it is still far from the time consumptions demanded by classical ML methods like DNAML (Felsenstein, 1993).

Table 3.2: Results from the biological datasets of 76 Glyceraldehyde 3-phosphate dehydrogenase sequences, of ATP synthase alpha/beta (105 seqs.), and of 193 Calporin homologs: (a) Log-likelihood values of the best reconstructed trees and (b) Runtimes of tree reconstruction consumed by the different methods. The PhyNav column presents the runtime of a single repetition.

sequences	length	PHYML	PhyNav
76	633 bp	-32133	-32094
105	1821 bp	-88975	-88632
193	465 bp	-64919	-64794

(a) Log-likelihood values.

sequences	length	PHYML	PhyNav		
			runtime	repetitions	(single repetition)
76	633 bp	40s	2529s	70	36s
105	1821 bp	117s	14413s	100	144s
193	465 bp	101s	22306s	200	116s

(b) Runtimes.

3.4 Discussions

We propose a new search strategy to optimize the objective function for large phylogenies. Starting from an initial tree the PhyNav method uses heuristics to reduce the number of sequences, to reconstruct scaffold trees, and to add again the remaining sequences. During these steps the constructed trees are optimized using fast NNI operations.

The suggested method produced better results on all dataset compared to Weighbor and PHYML. The trade-off for better accuracy is of course the runtime. While Weighbor outperformed PHYML and PhyNav with respected to the runtime on the simulated datasets, PHYML is 7.5-fold faster than PhyNav. However, spending more time might be well acceptable, because the quality of the results increases.

On the biological datasets PhyNav showed much longer runtimes compared to PHYML. Nevertheless, the substantial increase of the likelihoods might well justify that this effort

is worthwhile, since it is still far from the time consumptions demanded by classical ML methods like DNAML (Felsenstein, 1993).

The mechanism to add the remaining sequences of $\overline{S_k}$ to T_k cannot be expected to give the most accurate results. However, our way is simple and performs efficiently, especially since the NNI operations seem to well remove unfortunate placements during the construction of the full trees T. Additionally, it might be worth trying other algorithms like Important Quartet Puzzling to add the remaining sequences (see **Chapter** 4).

PhyNav can be applied to large dataset. We analyzed an alignment of 1146 Ankyrin amino acid sequences (PF00023) downloaded from the PANDIT database version 12.0 (Whelan *et al.*, 2003). The PhyNav options were set to 1000 repetitions and $k = 3$ and the WAG model (Whelan and Goldman, 2001) was applied. PhyNav found a best tree with -74665 log likelihood and needed about 15 minutes per repetition. Thus, the whole computation took about 10 days.

4 Important quartet puzzling and nearest neighbor interchange

Quartet-based algorithms form a major class of phylogenetic tree reconstruction. The main idea is the reconstruction of quartet trees, which are subsequently used to construct an overall tree (Strimmer and von Haeseler, 1996; Willson, 1999; Ranwez and Gascuel, 2001). However, the complexity of $O(n^4)$ prohibits an application of quartet methods to data with more than approximately 100 sequences because it is necessary to evaluate all quartet trees.

The chapter introduces the so-called *important quartet puzzling Method*, IQP, which uses only $O(n^2)$ quartets for construction a tree to overcome the computational drawback. Then, a combined search strategy called IQPNNI which moves fast through tree space is proposed. The accuracy and computing time of IQPNNI as well as other methods are examined with both simulated and real data. Finally, we suggest a rule, which indicates when to stop the search.

4.1 Important quartet puzzling method

4.1.1 k-representative concept

We consider a binary rooted tree T_r with root r (see Figure 4.1). For each leaf l, we compute the *topological distance* to the root $d_p(l, r)$ as the number of branches on the path from r to l in T_r (in computer science parlance the *depth*). Leaves l_1, l_2 with the same distance $d_p(l_1, r) = d_p(l_2, r) \equiv d_p$ are said to be on *level* d_p. For example, in Figure 4.1 leaves a, b are on level 3, leaves g and h are on level 4 and so on. The distance $d_p(\cdot, r)$ induces a natural ordering of the leaves.

Definition 15 *A set of* k**-representative leaves** $S_k(T_r)$ *of* T_r *is simply a collection of pairs* $(l, d_p(l, r))$ *that includes at most* k *leaves with the smallest distances from the root, where possible ties are resolved randomly.*

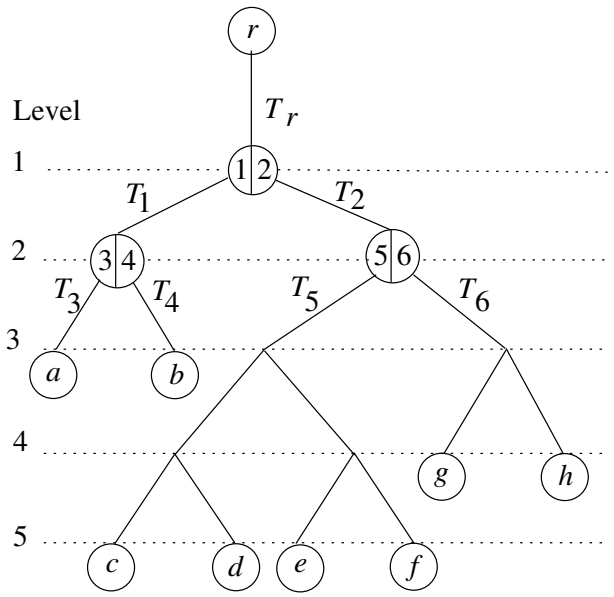


Figure 4.1: A binary rooted tree T_r with root r. The horizontal lines indicate the levels of the leaves, $T_1, T_2, \ldots$ denote the rooted subtrees, where the root of T_i is indicated by the corresponding index i.

We say the tree is represented by the k-representative leaf set. For example, the k-representative leaf set for $k = 4$ is $S_4(T_r) = \{(a,3),(b,3),(g,4),(h,4)\}$ in the current tree (Figure 4.1). For $k = 5$, the set is not unique, either of the leaves $(c,5),(d,5),(e,5)$, or $(f,5)$ can be added to $S_4(T_r)$ to form $S_5(T_r)$.

To motivate this abstract concept, think of $S_k(T_r)$ as a collection of contemporary sequences that are closest to the root (with respect to the defined distance). Thus, this collection resembles the ancestral sequence.

The notation of a k-representative leaf set generalizes to all rooted subtrees of the rooted tree T_r. Moreover, the computation of the corresponding sets can be done in linear time. Figure 4.1 shows that the representative leaf set of T_r can be computed from the representative leaf sets of the rooted subtrees T_1 and T_2. Similarly, the leaf sets of T_1 and T_2 can be obtained from $S_k(T_3)$, $S_k(T_4)$ and $S_k(T_5)$, $S_k(T_6)$, respectively, and so on. The 4-representative leaf sets of subtrees T_5 and T_6 are $S_4(T_5) = \{(c,3),(d,3),(e,3),(f,3)\}$ and $S_4(T_6) = \{(g,2),(h,2)\}$. $S_4(T_2)$ is obtained from $S_4(T_5) \cup S_4(T_6)$ by increasing the corresponding distances by one, choosing the four leaves with the smallest distances and breaking ties randomly. Thus, we obtain $S_4(T_2) = \{(g,3),(h,3),(c,4),(d,4)\}$.

Since the size of the k-representative leaf set of a rooted subtree T_i is $O(k)$, the cost of computing the next representative leaf set from its child leaf sets is $O(k)$. Thus, for

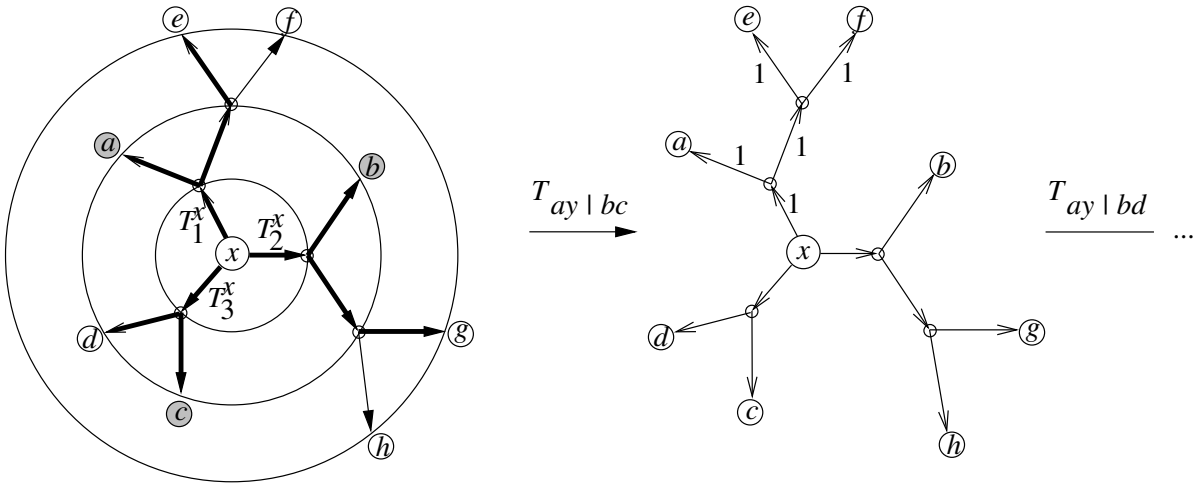


Figure 4.2: One internal node splits the tree into three rooted subtrees T_1^x, T_2^x and T_3^x with $S_2(T_1^x) = \{a, e\}$, $S_2(T_2^x) = \{b, g\}$, $S_2(T_3^x) = \{c, d\}$. A new y is needed to insert into the current tree. There are eight important quartets with respect to the internal node x and the new sequence y, i.e. $(y, a, b, c), (y, a, b, d), (y, a, g, c), (y, a, g, d), (y, e, b, c), (y, e, b, d), (y, e, g, c)$ and (y, e, g, d). The thick lines are the paths from the root x to representative leaves of three rooted subtrees. The gray circles indicate the representative leaves that are needed to compute the quartet tree (y, a, b, c). Because $T_{a,y|b,c}$ is reconstructed, then all branches in the rooted subtree T_1^x receive a weight of 1.

a rooted tree with n leaves the collection of all k-representative leaf sets is computed in $O(nk)$ time.

Now consider an unrooted tree T. Each internal node x splits T into three disjoint subtrees, which we then root with the original internal node x. Let T_1^x, T_2^x, and T_3^x denote the corresponding rooted subtrees with the same root x (see Figure 4.2, left). For these rooted subtrees we compute $S_k(T_1^x)$, $S_k(T_2^x)$, and $S_k(T_3^x)$. As in the case of rooted trees the computation of all representative leaf sets for all rooted subtrees is efficiently possible.

4.1.2 Important Quartets (IQs)

We now are ready to introduce the concept of *important quartets*. In the original PUZZLE algorithm (Strimmer and von Haeseler, 1996) a tree with n leaves was reconstructed sequentially by starting with a randomly chosen quartet tree into which the next sequence

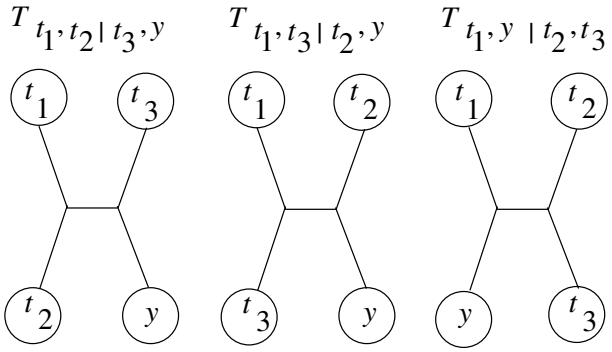


Figure 4.3: The three tree topologies of 4 sequences t_1, t_2, t_3 and y and their abbreviations as used in the text.

is inserted by evaluating all quartet trees that contain three sequences already present in the reconstructed tree and a fourth sequence, which needs to be inserted in this tree. If a tree with n leaves is reconstructed, $\mathrm{O}(n^4)$ quartets need to be evaluated.

Definition 16 *A quartet $q = (t_1, t_2, t_3, y)$ is called an* **important quartet** *of an internal node x of an unrooted tree T if and only if*

- *the sequence y does not belong to the leaves of tree T,*
- *the sequence t_1, t_2, and t_3 are elements of $S_k(T_1^x)$, $S_k(T_2^x)$, and $S_k(T_3^x)$, respectively.*

Thus an important quartet consists of three representatives each from one of the three $k-$representative leaf sets derived from the internal node x and a sequence y, which needs to be inserted in the tree. By construction, t_1, t_2, t_3 are close to x and close to each other, thus the reconstruction of a quartet tree for the quartet (t_1, t_2, t_3, y) is more likely to be accurate, because quartets with closely related sequences are less affected by the accumulation of evolutionary noise due to parallel and back substitutions.

More generally, a quartet q is an important quartet of an unrooted tree T if it is an important quartet of some internal node x of T.

Each internal node splits the tree into three rooted subtrees and each of them is presented by at most k representative leaves. For a new sequence y and an internal node x, there are $O(k^3)$ important quartets. Because T has $O(n)$ internal nodes, $O(nk^3)$ important quartets are possible.

4.1.3 Important quartet puzzling (IQP) algorithm

In order to put a new sequence y into a tree T we first determine the important quartet set of the tree T by specifying important quartets for all internal nodes of the tree T. Then for each important quartet $q = (t_1,\ t_2,\ t_3,\ y)$ the optimal tree topology (with respect to some objective function) is computed among the three unrooted topologies $T_{t_1,t_2|t_3,y}$, $T_{t_1,t_3|t_2,y}$ and $T_{t_1,y|t_2,t_3}$ (see Figure 4.3). In IQP, quartet trees are constructed using Neighbor Joining, these trees are the minimum evolution trees in case of four sequences (Saitou and Nei, 1987). The estimated quartet trees are then used to place sequence y.

Figure 4.2 illustrates the procedure. Consider quartet $q = (a, b, c, y)$ comprising the new sequence y, leaves a, c, and b which are representatives of subtrees T_1^x, T_2^x, and T_3^x, respectively. Assume that based on the sequences a, b, c, y tree $T_{ay|bc}$ is reconstructed, then each branch of T_1^x gets score 1. Then we continue with the next quartet (a, b, d, y).

Repeating the above procedure for all important quartets of T and summing the resulting scores, assigns a total score to each branch. Sequence y is inserted on the branch with the highest score. In case of ties, a branch with highest score is selected at random. Scores are computed in $O(nk^3)$ time by using a simple recursive procedure (Schmidt, 2003, chapter 4).

In the following k was set equal to four. Thus, if one wants to compute a phylogenetic tree following the TREE-PUZZLE procedure but using only important quartets, then one would need $O(n^2)$ computing time. However, we will not pursue this approach here, because simulations showed that the accuracy of this approach is not satisfying (data not shown).

4.2 Combining tree reconstruction methods

We here suggest a combination of tree reconstruction methods to compute an optimal tree. The combination alternates between the nearest neighbor interchanges (NNIs) method and the sequential sequence-by-sequence assembly approach based on IQP as described follows

Algorithm 4.1: Combined-Algorithm (IQPNNI)

begin

Initial step: An initial tree is built applying BIONJ (Gascuel, 1997). Then NNIs are performed until no further improvement of the likelihood function is found (Guindon and Gascuel, 2003). We call the resulting tree T_{best}, with log-likelihood ℓ^{log}_{best}.

Optimization step: Delete each leaf with probability $0 < p_{del} < 1$ from T_{best}. Re-insert the collection of deleted leaves by applying the IQP. Optimize the resulting tree $T_{\text{intermediate}}$ using NNI.

Comparative step: If the log-likelihood $\ell^{log}_{\text{intermediate}} > \ell^{log}_{\text{best}}$, then set $T_{\text{best}} \leftarrow T_{\text{intermediate}}$ and update the log-likelihood.

Stop criterion: If the number of optimization steps is less or equal than a pre-defined number of total optimization steps $\#step$, go to the Optimization step, otherwise stop and output T_{best}.

end

This combined strategy has two main advantages. First, deleting and re-inserting some leaves in the optimization step helps us to escape from a local optimum when applying NNI. Moreover, deleting and re-inserting only a proportion of all leaves conserves parts of the optimized tree and therefore saves computing time.

The algorithm presented here used the maximum likelihood criterion as an objective function, however the algorithm is not restricted to maximum likelihood, any tree reconstruction method with an objective function to minimize or maximize will work in the presented approach.

4.3 Accuracy

We tested the accuracy and computing time of IQPNNI as well as other methods with simulated data and two real data sets. Computing time was measured on a 2.0 GHz PC with 512 MB RAM. The size of the representative leaf set was $k = 4$, the probability to delete a sequence was 0.3 for simulated data and 0.1 for the biological data.

The accuracy of IQPNNI was compared to Weighbor 1.2 (Bruno *et al.*, 2000), fastDNAml version 1.2 (Olsen *et al.*, 1994), and PHYML version 2.0.1 (Guindon and Gascuel, 2003). Weighbor is a distance-based method and is combined with DNADIST version 3.5

(Felsenstein, 1993), the other programs are maximum likelihood methods (see Ranwez and Gascuel, 2001, for a detailed reference about the performance of current quartet-based methods compared to other approaches). MetaPIGA (Lemmon and Milinkovitch, 2002) was not included in the simulation study because it offers no version that runs in batch mode.

All methods were run with default options. However, the parameters of models of sequence evolution were not estimated but set identical to the simulated conditions.

Accuracy was measured as the percentage of cases where the inferred tree topology and the model tree topology are identical. Alternatively, we computed the average Robinson and Foulds distance (Robinson and Foulds, 1981).

4.3.1 Small simulated data

We generated randomly 3,000 trees with 30 taxa. Trees were drawn from the Yule-Harding distribution (Harding, 1971). The branch lengths of trees were drawn from an exponential distribution with mean values equal to 0.03, 0.06, and 0.15 to accommodate for slow, medium, and high rates of evolution, respectively. Seq-Gen (Rambaut and Grassly, 1997) was used to evolve sequences along the trees using the Kimura two-parameter model (Kimura, 1980) with a transition/transversion ratio of 2.0, and a sequence length of 500 bp.

Tables 4.1 shows that IQPNNI outperforms all other methods analyzed. However, the performance is on average only marginally better, i.e., .064 versus .066 for the Robinson and Foulds distance (Table 4.1b). Weighbor, a distance based method, displays the lowest performance in terms of accuracy.

It may be surprising that the probability to reconstruct the true tree is so small, an effect that is due to the short sequence length (500 bp). If we would use sufficiently long sequences, then all methods will perform equally well with respect to accuracy. In this sense, reconstruction of phylogenetic trees is easy for simulated data. However, for biological data one typically has short sequences.

The computing time to estimate a phylogenetic tree with 30 sequences is not really an issue for the four programs tested. Weighbor is the fastest program, it needs on average 0.4 seconds to output a tree, followed by PHYML (2.9 seconds), IQPNNI (16.7 seconds), and fastDNAML (28.9 seconds). Because computing time is not an issue for small data sets one should apply the program with highest accuracy.

Table 4.1: The performance of IQPNNI and other tested methods. Parameter settings: [1] $\#step = 20$, $p_{del} = 0.3$, and $k = 4$.

	Weighbor	fastDNAml	PHYML	**IQPNNI**[1]
slow rate	9.7	14.3	14.3	**14.7**
medium rate	12.5	19.9	19.9	**20.4**
high rate	11.2	16.3	15.9	**16.5**
average	11.1	16.8	16.7	**17.2**

(a) The percentage of cases for that the inferred tree and the model tree are identical in small simulated data sets.

	Weighbor	fastDNAml	PHYML	**IQPNNI**[1]
slow rate	.084	.069	.069	**.067**
medium rate	.076	.062	.060	**.059**
high rate	084	.075	.068	**.067**
average	.081	.069	.066	**.064**

(b) Average Robinson and Foulds distance of small simulated data sets.

4.3.2 Large simulated data

The accuracy of IQPNNI for large data sets was investigated on one random tree topology that was created as described above, but with 1,000 sequences of length 500, 1,000, and 2,000 base pairs. The mean branch length was set to 0.05. We compared Weighbor, PHYML, and IQPNNI. Unfortunately, fastDNAml could not be applied to 1,000 sequence data sets, because the computing time was too long. Table 4.2 displays the Robinson and Foulds (1981) distance for the three tree reconstruction methods designed to deal with large numbers of taxa. The numbers in the table show the results of 10 simulation runs and the average performance for each method as a function of the alignment length. Not surprisingly as the sequence length increases all methods get better, that is they reconstruct trees that are closer to the model tree. However, there is a substantial difference in the performance between Weighbor and PHYML or IQPNNI. Weighbor shows substantially reduced accuracy. The differences in performance between PHYML and IQPNNI are less pronounced.

IQPNNI is in 21 out of 30 simulations closer to the model tree as PHYML, we observe

Table 4.2: Robinson and Foulds distance for 30 simulations of data sets with 1,000 sequences

simulation:	1	2	3	4	5	6	7	8	9	10	**average**
						500 bp					
Weighbor	.129	.119	.098	.121	.130	.117	.122	.132	.101	.113	**.118**
PHYML	.064	.061	.054	.066	.048	.049	.058	.074	.054	.047	**.058**
IQPNNI[1]	.059	.053	.051	.059	.043	.046	.055	.069	.052	.046	**.053**
						1,000 bp					
Weighbor	.078	.073	.082	.076	.073	.080	.076	.079	.086	.085	**.078**
PHYML	.043	.028	.038	.038	.038	.032	.038	.040	.035	.035	**.037**
IQPNNI[1]	.044	.028	.036	.036	.036	.031	.035	.040	.033	.033	**.036**
						2,000 bp					
Weighbor	.047	.054	.060	.062	.053	.051	.056	.046	.050	.052	**.053**
PHYML	.018	.016	.024	.019	.020	.016	.023	.021	.021	.028	**.021**
IQPNNI[1]	.014	.015	.024	.019	.019	.016	.022	.022	.021	.028	**.020**

[1] parameter settings: $\#step = 100$, $p_{del} = 0.3$, and $k = 4$.

7 ties and two cases where PHYML is closer to the model tree. Ties occur for long sequences. Thus, IQPNNI shows a higher accuracy for short sequences. For longer sequences the differences between both approaches disappear. We should note however, that the log-likelihood of the IQPNNI-tree is typically higher than the corresponding PHYML tree. This increase in likelihood is associated with an increase in computing time. Table 4.3 displays the time necessary to evaluate one tree. Because we set the number of intermediate trees to $\#step = 100$ it took on average 4.5 h (500 bp), 7.2 h (1,000 bp), or 10.5 h (2,000 bp) for IQPNNI to run a simulation. On the other hand, since a large number of different trees were analyzed per run, we might put more confidence on the resulting tree. Note, that $\#step = 100$ iterations do not allow for a thorough search in tree-space. For biological data sets the number of iterations must be larger.

4.3.3 Real data

We applied IQPNNI to two large data the ssu-rRNA alignment (218 species, 4182 bp) the rbcl-gene alignment (500 species, 1398 bp) recently analyzed (Guindon and Gascuel, 2003) and compared our results to PHYML (Guindon and Gascuel, 2003) and MetaPIGA

Table 4.3: Computing times (in minutes) for 1,000 sequences and different tree building methods

	Weighbor	PHYML	**IQPNNI (for one intermediate tree)**[1]
500	190.0	6.5	**2.7**
1000	190.0	13.5	**4.3**
2000	172.0	19.0	**6.3**
average	184.0	13.0	**4.4**

[1] parameter settings: $\#step = 100$, $p_{del} = 0.3$, and $k = 4$.

(Lemmon and Milinkovitch, 2002). The parameter settings for IQPNNI are given in Table 4.4. The HKY model (Hasegawa *et al.*, 1985) was used for the DNA data and the transition-transversion parameter was estimated from the data. The results are summarized in Table 4.4.

In both cases IQPNNI found quite a lot of trees with higher likelihood values than the ones obtained with PHYML and MetaPIGA (see Figures 4.4 and 4.5). For the ssu-rRNA data the best IQPNNI tree is 291 and 111 log-likelihood units higher than PHYML and MetaPIGA, respectively. For the rbcl-gene the best IQPNNI tree is 180 log-likelihood units larger than the PHYML tree and 69 log-likelihood units higher than the MetaPIGA tree. Thus, the increase in computation time (see Table 4.4) is rewarded by a better maximum likelihood tree.

4.4 Stopping the search

Figures 4.4 and 4.5 display for the ssu and rbcl DNA data sets the increase in log-likelihood as the number of iterations grows. Already during the first iterations we almost instantaneously observe a drastic increase in the likelihoods. After roughly 100 iterations (220 min or 300 min computing time) we are finding better trees compared to PHYML and MetaPIGA. The + signs above the lines labeled MetaPIGA and PHYML in Figures 4.4 and 4.5 indicate the better trees that we discovered. As the search continues, the rate of discovering better trees decreases. However, as we continue our "guided tour through" tree space we keep on finding better trees.

Thus, like MetaPIGA we need a criterion to stop our quest for the best tree. Here we suggest to apply an estimation method that is based on the time of occurrence (i.e.

Table 4.4: Best log likelihoods and computing times for three tree reconstruction methods for real data

gene	number of taxa	PHYML	MetaPIGA	**IQPNNI**[1]
			loglikelihood	
ssu rRNA	218	-156,895	-156,715	**-156,604**
rbcl	500	-100,191	-100,080	**-100,011**
			runtime (min)	
ssu rRNA	218	5.1	74.5	**379**
rbcl	500	7.5	158.5	**672**

[1]parameter settings: $\#step = 300$ $p_{del} = 0.1$, and $k = 4$.
Note, although we used the same data as Guindon and Gascuel (2003) our likelihood differs slightly from the ones published, which is due to the new version 2.0.1 of PHYML (Guindon and Gascuel, 2003), and results of MetaPIGA (Lemmon and Milinkovitch, 2002) depend on the random process.

number of iterations) of better trees during our search. These time points are indicated by the jumps in the graph in Figures 4.4 and 4.5.

More precisely, let $L_1^{log}, L_2^{log}, \ldots, L_j^{log}$ denote the log-likelihoods for the first j iterations, then the sequence $\tau(k)$ of record times (i.e. iteration number, when a better tree is found) is defined by

$$\tau(1) = 1,\ \tau(k+1) = \min\{j | L_j^{log} > L_{\tau(k)}^{log}\}.$$

This sequence is used to estimate the point in time, τ_{stop}, at which to stop the search, i.e., when it appears unlikely that a further search will lead to a better tree. Using the theory detailed in Cooke (1980) and Roberts and Solow (2003), we estimate during the run of IQPNNI an upper 95% confidence limit $\tau_{95\%}$ of τ_{stop}. More precisely, consider a sequence of record times $\tau_1, \tau_2, \ldots, \tau_k$, then we compute an upper $(1-\alpha)100\%$ stopping time as

$$\tau_{(1-\alpha)100\%} = \tau_1 + \frac{\tau_1 - \tau_k}{\left(\frac{-\log(\alpha)}{k}\right)^{-\hat{\nu}} - 1},$$

where the shape parameter of the joint Weibull distribution ν is estimated as follows

$$\hat{\nu} = \frac{1}{k-1} \sum_{j=1}^{k-2} \log\left(\frac{\tau_1 - \tau_k}{\tau_1 - \tau_{j+1}}\right).$$

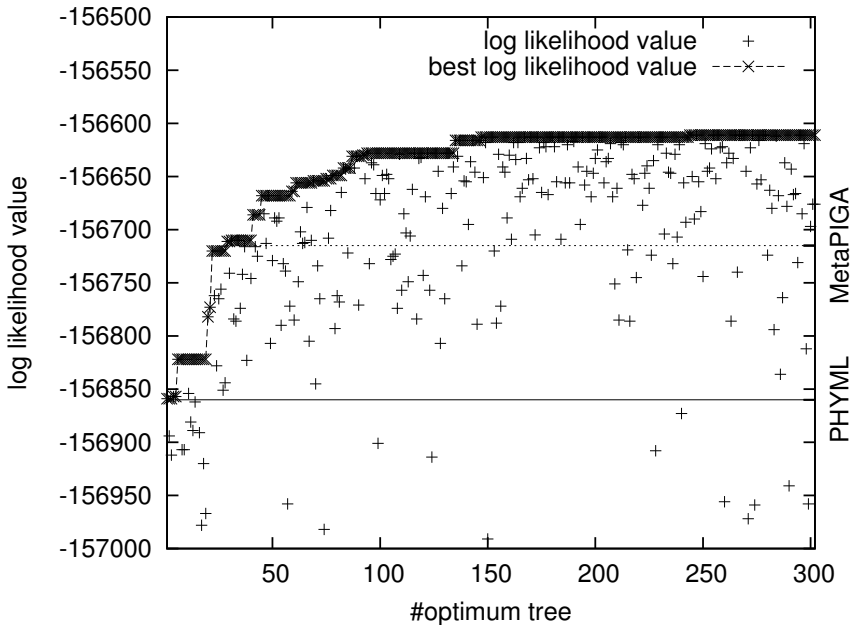


Figure 4.4: Exploring the likelihood surface of the suu-rRNA alignment of 218 species. The thick line shows the improvement in log likelihood during the IQNNI search. The + signs represent trees generated according to the combined algorithm. IQPNNI generated 258 different trees, 219 of them had a higher likelihood than the best PHYML tree (horizontal line) (Guindon and Gascuel, 2003), and 121 were better than MetaPIGA (horizontal dashed line) (Lemmon and Milinkovitch, 2002).

Once $\tau_{95\%}$ iterations have been carried out and a better tree was not detected the program will stop and output the best tree found. We can conclude that we will not find a better tree with a probability of 95% during this search. On the other hand, if a better tree is found before we hit $\tau_{95\%}$, we re-compute $\tau_{95\%}$ on the basis of the new record time added to the sequence $\tau(k)$.

This additional number of iterations to reach $\tau_{95\%}$ further increases the computation time, i.e for the 218-ssu rRNA about 8 hours were necessary and for the rbcl data we used a total of 15h. But now, we are in the position to know, that with 95% probability we would not have found a better tree when extending the search even longer. If one is willing to spend more computation time, it is of course possible to compute $\tau_{99\%}$ or even higher upper bounds.

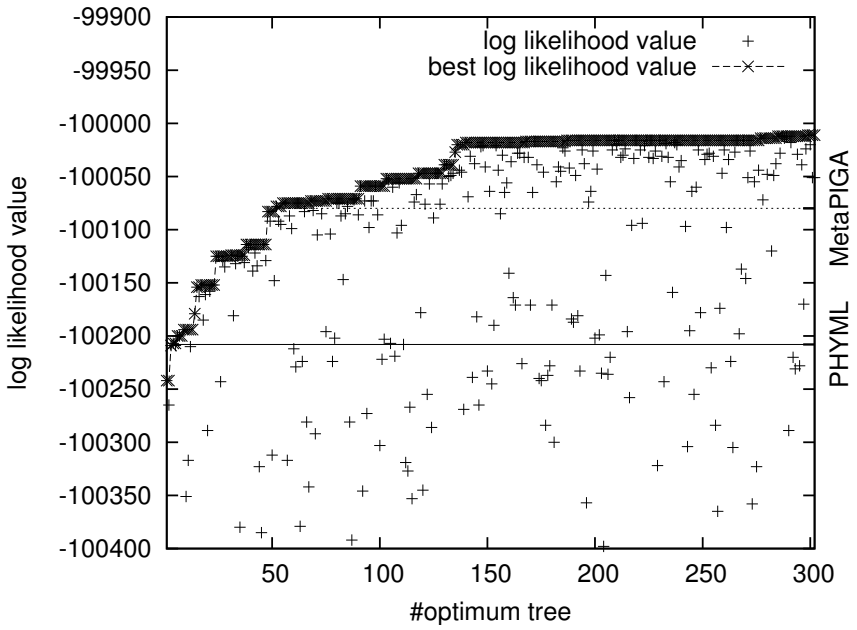


Figure 4.5: Exploring the likelihood surface of the rbcl alignment of 500 species. 283 different trees were generated. 192 had a higher likelihood than the best PHYML tree (horizontal line) (Guindon and Gascuel, 2003), and 125 showed larger log likelihoods as the best MetaPIGA tree (horizontal dashed line) (Lemmon and Milinkovitch, 2002).

4.5 Discussions

We are presenting the IQP method to reconstruct trees from large sequence data. Our simulations and the analysis of biological data show that a combination of nearest neighbor interchange and IQP leads to trees that are either closer to the model tree, as defined by the simulations, or to trees with higher likelihoods than found so far for biological examples. This improved performance, however, is achieved by an increase in computing time as compared to other fast programs. However, the times needed are not unrealistic.

The algorithm presented here turns out to efficiently search the tree space for better trees. We have used the maximum likelihood criterion as an objective function, however the algorithm is not restricted to maximum likelihood, any tree reconstruction method with an objective function to minimize or maximize will work in the presented approach.

Although we have only presented results for $p_{del} = 0.3$ and $k = 4$ for simulated data, further studies with p_{del} ranging from 0.2 to 0.4 and k from 4 to 6 showed that the accuracy based on Robinson and Foulds distance, percentage of correctly reconstructed

trees is not affected (data not shown). Similarly, various combinations of k (4, 5, or 6) and p_{del}, ranging from 0.1 to 0.3 resulted in minor changes in the log likelihood, that varied from -156,715 to -156,604 ($k = 4, p_{del} = 0.1$) for the ssu_rRNA data set and from -100,058 to -100,011 ($k = 4, p_{del} = 0.1$) for the rbcl-gene alignment. We note that this observation does not allow any generalizations. In any real application one should run IQPNNI for different choices of k and p_{del}.

Our simulations indicate that IQPNNI shows a better performance than other tested methods in terms of being closer to the true tree. However, we have only considered a very narrow range of simulations. We have not taken into account the possibility of model violations, uncertainty of estimating the parameters of the model and various other sources of uncertainty.

Since IQPNNI generates trees with similar high likelihoods, we have included an option in the program, that allows to output a majority rule consensus tree of all the intermediate trees found during one search run of the tree space. We also output the frequencies of the groupings found in that tree. If one is interested only in the most frequent groupings found in the collection of trees with a high likelihood, then this option is helpful.

Finally, we are suggesting a statistics that provides a guide to stop the search for a better tree. This very simple and crude estimation procedure proves to be very useful, yet it increases the computation time again. To get additional confidence in the reconstructed tree and its maximum likelihood value, one needs to repeat the search with several independent runs of our program. However, we are sure that we have only scratched the surface of applying statistical inference based on record values to problems of tree reconstruction. Further investigations about the performance of such methods are certainly necessary, but beyond the scope of the thesis.

5 Shortest triplet clustering algorithm

Having introduced efficient PhyNav and IQPNNI methods to search maximum likelihood phylogenies with up to 1000 sequences, we now propose a new distance-based clustering method the so-called *shortest triplet clustering algorithm* (STC) to build a phylogeny in $O(n^2)$ time for n sequences. Therefore, STC can construct phylogenies for extremely large datasets as envisaged in biodiversity studies. To this end, we first describe a simple clustering algorithm to recover a tree from a distance matrix. Second, the natural definition of k-representative sets as well as the construction of shortest triplets are presented. Third, the shortest triplet clustering algorithm is proposed. Finally, the efficiency of STC algorithm in comparison with other methods is examined on a large range of simulated datasets.

5.1 Recovering a tree from a distance matrix

Let us recall some notations. $S = \{s_1, s_2, \ldots, s_n\}$ is a set of n species. $\mathbb{D} = \mathcal{D}(uv)$ is an arbitrary distance matrix where $\mathcal{D}(uv)$ is the distance between two species u and v.

5.1.1 Estimating edge lengths using triplets

Consider a subset X of S, then $\varphi(X) : S \mapsto L$ induces a map on a subtree of T such that the relationships of species in X are displayed by the subtree with leaf set $\varphi(X)$. The complement $S_0(X) = S - X$ will be called the *unclassified species set*, because the relationships of species in $S_0(X)$ to X is not known from the subtree. Note that we will use S_0 instead of $S_0(X)$ if X is clear from the context.

Let $T_r = (V_r, E_r)$ denote a rooted tree with root r and leaf set L_r, and let S_r be a subset of S such that $\varphi(S_r) = L_r$. For convenience, we use S_r and L_r interchangeably.

Now, we consider the most simple edge length estimation problem. That is, we would like to estimate the edge lengths for a triplet tree $\{a, b, c\}$ with distance matrix $\mathbb{D}$ (see Figure 5.1a). Edge lengths are estimated as follows

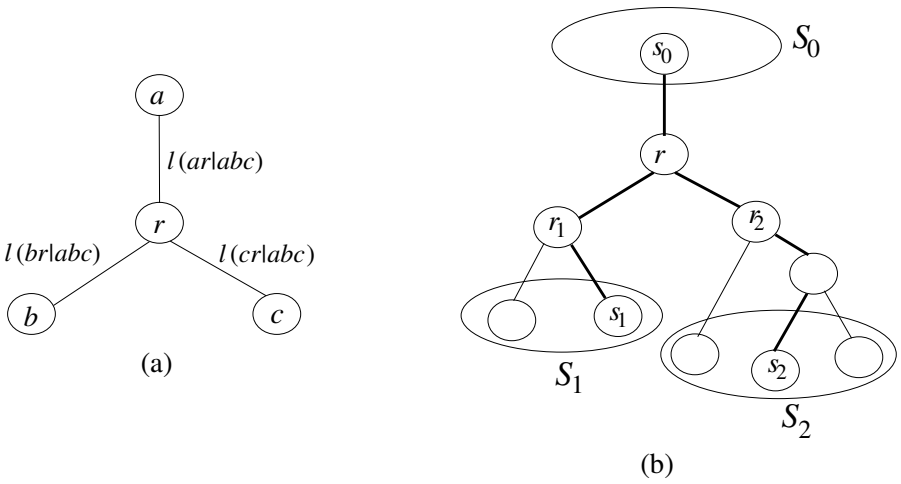


Figure 5.1: On the left, estimation of edge lengths $\ell(ar \,|\, abc), \ell(br \,|\, abc)$ and $\ell(cr \,|\, abc)$ of the triplet tree $\{a, b, c\}$. On the right, estimation of path length $\ell(s_0 r \,|\, s_0 s_1 s_2)$ and edge lengths $\ell(r_1 r \,|\, s_0 s_1 s_2)$, $\ell(r_2 r \,|\, s_0 s_1 s_2)$ based on the triplet tree $\{s_0, s_1, s_2\}$.

$$\ell(ar \,|\, abc) = \frac{1}{2}(\mathcal{D}(ab) + \mathcal{D}(ac) - \mathcal{D}(bc)) \tag{5.1a}$$

$$\ell(br \,|\, abc) = \frac{1}{2}(\mathcal{D}(ab) + \mathcal{D}(bc) - \mathcal{D}(ac)) \tag{5.1b}$$

$$\ell(cr \,|\, abc) = \frac{1}{2}(\mathcal{D}(ac) + \mathcal{D}(bc) - \mathcal{D}(ab)) \tag{5.1c}$$

Now consider a rooted T_r with the inferred tree-like metric $\mathbb{D}_{(T_r,\ell)}$. The rooted tree T_r consists of two rooted subtrees T_{r_1} and T_{r_2} (see Figure 5.1b). For convenience, we will use T_i instead of T_{r_i} if r_i is clear from the context. The leaf set $S_r = \{S_1 \cup S_2\}$ where $S_r \subset S$ and $S_0 = S - S_r$ is not represented in T_r . Then we can compute

$$\ell(s_0 r \,|\, s_0 s_1 s_2) = \frac{1}{2}(\mathcal{D}(s_0 s_1) + \mathcal{D}(s_0 s_2) - \mathcal{D}(s_1 s_2)) \tag{5.2a}$$

$$\ell(s_1 r \,|\, s_0 s_1 s_2) = \frac{1}{2}(\mathcal{D}(s_0 s_1) + \mathcal{D}(s_1 s_2) - \mathcal{D}(s_0 s_2)) \tag{5.2b}$$

$$\ell(s_2 r \,|\, s_0 s_1 s_2) = \frac{1}{2}(\mathcal{D}(s_0 s_2) + \mathcal{D}(s_1 s_2) - \mathcal{D}(s_0 s_1)) \tag{5.2c}$$

for each triplet $(s_0, s_1, s_2) \in (S_0 \times S_1 \times S_2)$.

With $\mathbb{D}_{(T_1,\ell_1)}(s_1r_1)$ and $\mathbb{D}_{(T_2,\ell_2)}(s_2r_2)$ we denote the known distances of s_1 and s_2 to their roots r_1 and r_2. Thus, we can compute for each triplet $\{s_0, s_1, s_2\}$ the lengths $\ell(r_1r)$ and $\ell(r_2r)$ as

$$\ell(r_1r \mid s_0s_1s_2) = \ell(s_1r \mid s_0s_1s_2) - \mathbb{D}_{(T_1,\ell_1)}(s_1r_1) \tag{5.3a}$$
$$\ell(r_2r \mid s_0s_1s_2) = \ell(s_2r \mid s_0s_1s_2) - \mathbb{D}_{(T_2,\ell_2)}(s_2r_2). \tag{5.3b}$$

Note that if $\mathbb{D}$ is additive and T_1, T_2 are isometric subtrees of T, the lengths $\ell(r_1r)$ and $\ell(r_2r)$ do not depend on the choice of the triplet $\{s_0, s_1, s_2\}$.

Regardless of additivity considerations, we may define the average length for a fixed $s_0 \in S_0$ as

$$\ell(s_0r \mid s_0S_1S_2) \equiv \frac{1}{|S_1||S_2|} \sum_{(s_1,s_2)\in S_1\times S_2} \ell(s_0r \mid s_0s_1s_2) \tag{5.4}$$

We can estimate edge lengths $\ell(r_1r)$ and $\ell(r_2r)$ by using all possible triplets as

$$\ell(r_1r \mid S_0S_1S_2) \equiv \frac{1}{|S_0||S_1||S_2|} \sum_{(s_0,s_1,s_2)\in S_0\times S_1\times S_2} \ell(r_1r \mid s_0s_1s_2) \tag{5.5a}$$
$$\ell(r_2r \mid S_0S_1S_2) \equiv \frac{1}{|S_0||S_1||S_2|} \sum_{(s_0,s_1,s_2)\in S_0\times S_1\times S_2} \ell(r_2r \mid s_0s_1s_2) \tag{5.5b}$$

5.1.2 The largest path length criterion

We want to reconstruct a tree $T = (V, E)$ together with length function ℓ with respect to a distance matrix $\mathbb{D}$ such that $\mathbb{D}_{(T,\ell)}$ represents $\mathbb{D}$. To this end, we use triplets and the notation of a rooted tree T_r together with Equations 5.4 and 5.5.

Our algorithm starts with the observation that if we take an arbitrarily rooted tree T_s with $s \in S$ and length function ℓ_{T_s}, then there must be a pair of leaves (*neighboring leaves*) that share an immediate most recent common ancestor *mrca* which is farthest away from the root s with respect to ℓ_{T_s}. In Figure 5.2, the pair $(3, 4)$ satisfies this condition, we say this pair fulfills the *largest path length criterion*. The largest path length criterion easily generalizes to arbitrarily rooted subtrees T_i and T_j of T_s, where all descendants from the roots of T_i and T_j are in the vertex sets V_i or V_j, respectively.

Let $\mathcal{T}_S$ be the set of rooted subtrees of T_s (each leaf $l \in L_s$ is considered as a rooted subtree T_l). Now consider two disjoint rooted subtrees T_i and T_j of T_s where $i, j \in V_s$.

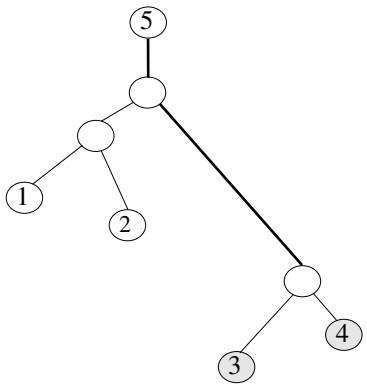


Figure 5.2: The tree is rooted at leaf 5. In the tree, leaves 3 and 4 with the largest path length from their most recent common ancestor to the root 5 are neighbors.

Then the distance $\ell(s,\, mrca \,|\, s\, S_i S_j)$ from the *mrca* of T_i and T_j to s is computed according to Equation 5.4, where S_i and S_j are the leaf sets of T_i and T_j, respectively. Then we pick

$$(T_{i_0}, T_{j_0}) = \text{argmax}\{\ell(s,\, mrca \,|\, s\, S_i S_j) \mid T_i, T_j \in \mathcal{T}_S\} \tag{5.6}$$

as a pair of neighbors (if we detect more than one pair, we randomly select one). By construction, (T_{i_0}, T_{j_0}) fulfills the largest path length criterion.

If $\mathbb{D}$ is additive, $\ell(s,\, mrca \,|\, s\, S_i S_j)$ is exactly the path length from the *mrca* of (T_i, T_j) to s. In other words, the path length from the *mrca* of (T_{i_0}, T_{j_0}) to s is largest and (T_{i_0}, T_{j_0}) is a true neighboring pair. However, in real applications $\mathbb{D}$ is rarely additive, therefore the root s is selected so as to avoid noise from stochastic errors involved with large distance estimates (Fitch and Margoliash, 1967a). To this end, s is selected such that the distance from the farthest species to root s is minimal,

$$med = \text{argmin}_{s' \in S}\{\max\{\mathcal{D}(s'x) \,|\, x = 1, \ldots, n\}\} \tag{5.7}$$

med is called *a median species*.

Moreover, to reduce the computational complexity of finding a pair of neighbors (T_{i_0}, T_{j_0}) using Equation 5.6, we store for each $T_i \in \mathcal{T}_S$ its *potential neighbor* $T_{i'} \in \mathcal{T}_S$ such that

$$T_{i'} = \text{argmax}\{\ell(med, mrca \,|\, med, S_i, S_j) \,|\, T_j \in \mathcal{T}_S\}. \tag{5.8}$$

Now the neighboring pair (T_{i_0}, T_{j_0}) fulfilling the largest path length criterion is determined as follows

$$(T_{i_0}, T_{j_0}) = \text{argmax}\{\ell(med, mrca \,|\, med, S_i, S_{i'}) \mid T_i \in \mathcal{T}_\mathcal{S}\}. \tag{5.9}$$

5.1.3 Clustering algorithm

An intuitive clustering method to reconstruct trees based on distance matrices and the largest path length criterion is described in Algorithm 5.1. This algorithm is similar to approaches described elsewhere (Farris, 1977; Klotz *et al.*, 1979; Li, 1981), however, an essential difference is that we estimate path lengths and edge lengths by using triplets.

Algorithm 5.1: A simple clustering algorithm based on the largest path length criterion.

Data: A pairwise distance matrix $\mathbb{D}$

Result: A tree T together with its edge lengths

begin

Initial step: Find the median species *med* using Equation 5.7. Set $\mathcal{T}_\mathcal{S} = \{T_1, \ldots, T_n\} - \{T_{med}\}$. Find for each $T_i \in \mathcal{T}_\mathcal{S}$ its *potential neighbor* $T_{i'} \in \mathcal{T}_\mathcal{S}$ using Equation 5.8.

Selection step (largest path length criterion): Find the neighboring pair (T_{i_0}, T_{j_0}) using Equation 5.9.

Agglomeration step: Combine T_{i_0} and T_{j_0} into a new rooted tree $T_{\{i_0 j_0\}}$ with root $i_0 j_0$, and estimate new edge lengths of $T_{\{i_0 j_0\}}$ using Equation 5.5. Delete T_{i_0} and T_{j_0} and add $T_{\{i_0 j_0\}}$ to $\mathcal{T}_\mathcal{S}$. Find the potential neighbor for the new rooted tree $T_{\{i_0 j_0\}}$ using Equation 5.8, and replace $T_{i'}$ for each $T_i \in \mathcal{T}_\mathcal{S}$ by $T_{\{i_0 j_0\}}$ if $T_{\{i_0 j_0\}}$ is its potential neighbor.

Stopping step: If $|\mathcal{T}_\mathcal{S}| > 1$ goto the Selection step, otherwise output the tree.

end

5.1.4 Local rearrangement

The heart of the clustering algorithm is the largest path length criterion, at which the path length from the *mrca* of (T_i, T_j) to *med* is estimated by $\ell(med, mrca \,|\, med, S_i, S_j)$ using Equation 5.4. Thus, as path length we take the average of the lengths obtained

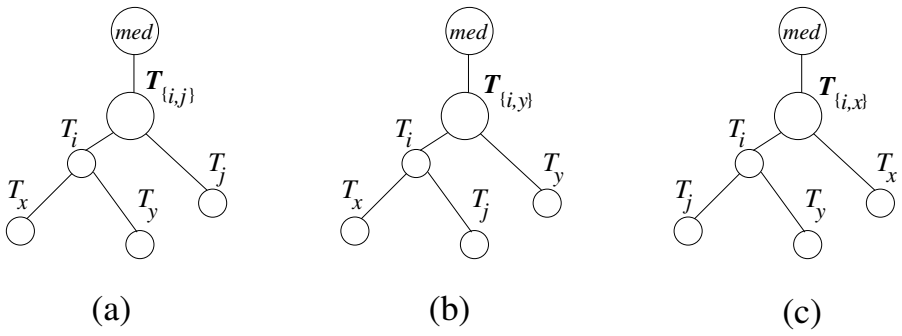


Figure 5.3: Reconstruction of new rooted tree $T_{\{ij\}}$ using the preorder traversal procedure based on the largest average path length criterion. If (T_x, T_y) is the neighboring pair, we stick to the suggested grouping of T_i and T_j (see Figure 5.3a). Otherwise, if (T_x, T_j) or (T_y, T_j) is the neighboring pair, we switch to the trees displayed in Figure 5.3b or 5.3c, respectively.

from at most $O(n^2)$ triplets $\{med, s_i, s_j\} \in med \times S_i \times S_j$. This average may not be the representative estimate of the true path length. Moreover the root *med* may be too far way from the *mrca* and this leads to an inaccurate estimate of the path length.

To take these problems into account, we extend the clustering algorithm. To this end, imagine the algorithm has clustered T_i and T_j with corresponding disjoint leaf sets $S_i,\ S_j \subset S$ (we have finished the agglomeration step). Thus, we have created the newly rooted tree $T_{\{ij\}}$ with leaf set $S_{ij} = \{S_i \cup S_j\}$ and the set of unclassified species $S_0(S_{ij}) = S - S_{ij}$. In the following, we describe the nearest neighbor interchange operation around the root of T_i upon condition that T_i consists of two rooted subtrees T_x, T_y (Figure 5.3a). First, we estimate average path lengths from the unclassified species set $S_0(S_{ij})$ to the *mrca* of $(T_x, T_y),\ (T_x, T_j)$ and (T_y, T_j) as

$$\ell(S_0(S_{ij})S_xS_y|S_xS_y) \equiv \frac{1}{|S_0(S_{ij})||S_x||S_y|} \sum_{(s_0,s_x,s_y)\in S_0(S_{ij})\times S_x\times S_y} \ell(s_0r|s_0s_xs_y) \quad (5.10a)$$

$$\ell(S_0(S_{ij})S_xS_j|S_xS_j) \equiv \frac{1}{|S_0(S_{ij})||S_x||S_j|} \sum_{(s_0,s_x,s_j)\in S_0(S_{ij})\times S_x\times S_j} \ell(s_0r|s_0s_xs_j) \quad (5.10b)$$

$$\ell(S_0(S_{ij})S_yS_j|S_yS_j) \equiv \frac{1}{|S_0(S_{ij})||S_y||S_j|} \sum_{(s_0,s_y,s_j)\in S_0(S_{ij})\times S_y\times S_j} \ell(s_0r|s_0s_ys_j). \quad (5.10c)$$

For convenience, we will use $\ell(S_0(S_{ij}) \,|\, S_xS_y)$ instead of $\ell(S_0(S_{ij})S_xS_y \,|\, S_xS_y)$. We now use the average path lengths from Equation 5.10 to decide which pair of subtrees among $(T_x, T_y),\ (T_x, T_j)$ and (T_y, T_j) is preferred. More specifically, if

$$\ell(S_0(S_{ij}) \,|\, S_xS_y) \geq \max\{\ell(S_0(S_{ij}) \,|\, S_xS_j), \ell(S_0(S_{ij}) \,|\, S_yS_j)\}$$

we stick to the suggested grouping of T_x and T_y (see Figure 5.3a). Otherwise, if $\ell(S_0(S_{ij}) \,|\, S_xS_j)$ or $\ell(S_0(S_{ij}) \,|\, S_yS_j)$ is larger than the remaining average path lengths, we swap T_y and T_j or T_x and T_j as displayed in Figure 5.3b or 5.3c, respectively. Note that this decision can be considered as a correction of the largest path length criterion by taking all possible triplets into account. We call the correction the *largest average path length criterion*.

We now explain the preorder traversal procedure (Aho *et al.*, 1974) to reconstruct the rooted tree T_i using the nearest neighbor interchange operation based on the largest average path length criterion (T_i is a subtree of $T_{\{ij\}} = (T_i, T_j)$):

Algorithm 5.2: Preorder traversal procedure (T_i)

Step 1: If T_i is a single leaf, return.

Step 2: Otherwise, T_i consists of two subtrees T_x and T_y. Do the nearest neighbor interchange operation around the root of T_i based on the largest average path length criterion (Equation 5.10). If T_x and T_j (or T_y and T_j) were exchanged, estimate new edge lengths using Equation 5.5.

Step 3: Apply the preorder traversal procedure to two rooted subtrees of T_i.

5.2 Representative sets and shortest triplets

For a set S of sequences (or taxa), the (genetic) distance matrix $\mathbb{D}$ is typically not additive due to stochastic errors (Fitch and Margoliash, 1967a). Larger distances between two sequences are less accurately estimated. This leads to a low performance of both the clustering algorithm and the preorder traversal procedure for divergent data sets.

In previous chapters, we have presented simple representative concepts to reduce stochastic error involved in large distances. Here, we extend our work by introducing the so-called *k-representative set* concept. We use now genetic distances instead of topological distances (all edges have length 1). Our motivation is to reduce the computational complexity and to exclude species far away from the root under consideration.

In the clustering algorithm, the path length from the *mrca* of (T_i, T_j) to *med* (see Figure 5.4) can be estimated by two approaches. The first method picks randomly one pair $(s_i, s_j) \in S_i \times S_j$ then computes

$$\ell(med, mrca \,|\, med, s_i, s_j) = \frac{1}{2}(\mathcal{D}(med, s_i) + \mathcal{D}(med, s_j) - \mathcal{D}(s_is_j)). \tag{5.11}$$

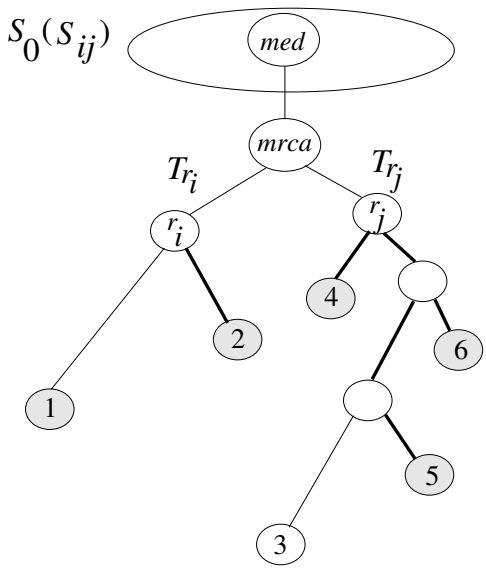


Figure 5.4: We select only $\min(k, |S_i|)$ and $\min(k, |S_j|)$ closest leaves to the root of T_i and T_j with respect to the path length, respectively, i.e. for $k = 3$ we pick $\{1, 2\}$ from T_{r_i} and $\{4, 5, 6\}$ from T_{r_j}. The leaf set $\{1, 2\}$ (or $\{4, 5, 6\}$) is the 3-representative leaf set of the rooted subtree T_{r_i} (or T_{r_j}).

The second approach takes the average distance

$$\ell(med, mrca \,|\, med, S_i, S_j) \equiv \frac{1}{|S_i||S_j|} \sum_{(s_i,s_j)\in S_i\times S_j} \ell(med, mrca \,|\, med, s_i, s_j). \quad (5.12)$$

Both approaches suffer from noise. Estimating the path length using Equation 5.11 may be inaccurate because it randomly picks a pair (s_i, s_j) which may not be really representative. Equation 5.12 may be problematic, especially since it might be susceptible to noise, due to the possibility of including long distances with large stochastic errors.

To overcome these problems, we select only $\min(k, |S_i|)$ and $\min(k, |S_j|)$ closest leaves to the root of T_i and T_j with respect to the path length, respectively. To illustrate, for $k = 3$ we pick $\{1, 2\}$ from T_i and $\{4, 5, 6\}$ from T_j in Figure 5.4.

Definition 17 *The set of min$(k, |S_i|)$ closest leaves to the root of T_i is called the* k**-representative leaf set**.

Hereafter, we estimate similar to Equation 5.4 the path length from the *mrca* of (T_i, T_j) to *med* as

$$\ell(med, mrca \,|\, med, S_i^k, S_j^k) = \frac{1}{|S_i^k||S_j^k|} \sum_{(s_i^k,s_j^k)\in S_i^k\times S_j^k} \ell(med, mrca \,|\, med, s_i^k, s_j^k) \quad (5.13)$$

which is only based on the k-representative leaf sets. Now we can perform the clustering algorithm with reduced complexity. However, we also want to improve the preorder traversal procedure. The average path length from the unclassified species set $S_0(S_{ij})$ to the $mrca$ of (T_i, T_j) is estimated by Equation 5.10 which also suffers from noise. To overcome this problem, we select only $\min(k, |S_0(S_{ij})|)$ unclassified species closest to the root of tree $T_{\{ij\}}$ with respect to distances $\ell(s_0 r \mid s_0 S_i^k S_j^k)$ where $s_0 \in S_0(S_{ij})$. We call the subset, denoted $S_0^k(S_{ij})$, *k-representative unclassified species set.*

Definition 18 *A triplet $\{s_0^k, s_i^k, s_j^k\} \in S_0^k(S_{ij}) \times S_i^k \times S_j^k$ which contains three representatives of the three $k-$representative sets is called a* **shortest triplet**.

By construction, s_0^k, s_i^k, s_j^k are close to the root of $T_{\{ij\}}$ and close to each other. Therefore, the edge length estimates based on shortest triplet $\{s_0^k, s_i^k, s_j^k\}$ are less susceptible to estimation errors.

We now rewrite Equation 5.10 to estimate the average path length from the representative unclassified species set $S_0^k(S_{ij})$ to the $mrca$ of (T_i, T_j) using only shortest triplets as

$$\ell(S_0^k(S_{ij}) \mid S_x^k S_y^k) \equiv \frac{1}{|S_0^k(S_{ij})||S_x^k||S_y^k|} \sum_{(s_0^k, s_x^k, s_y^k) \in S_0^k(S_{ij}) \times S_x^k \times S_y^k} \ell(s_0^k r \mid s_0^k s_x^k s_y^k) \tag{5.14a}$$

$$\ell(S_0^k(S_{ij}) \mid S_x^k S_j^k) \equiv \frac{1}{|S_0^k(S_{ij})||S_x^k||S_j^k|} \sum_{(s_0^k, s_x^k, s_j^k) \in S_0^k(S_{ij}) \times S_x^k \times S_j^k} \ell(s_0^k r \mid s_0^k s_x^k s_j^k) \tag{5.14b}$$

$$\ell(S_0^k(S_{ij}) \mid S_y^k S_j^k) \equiv \frac{1}{|S_0^k(S_{ij})||S_y^k||S_j^k|} \sum_{(s_0^k, s_y^k, s_j^k) \in S_0^k(S_{ij}) \times S_y^k \times S_j^k} \ell(s_0^k r \mid s_0^k s_y^k s_j^k) \tag{5.14c}$$

In short, the preorder traversal procedure uses only shortest triplets to estimate path lengths as well as edge lengths.

5.3 Shortest triplet clustering algorithm (STC)

We introduce now the shortest triplet clustering algorithm by combining the clustering algorithm, the local rearrangement, the k-representative sets, and the shortest triplets approach.

Algorithm 5.3: Shortest triplet clustering algorithm (STC)

Data: A pairwise distance matrix $\mathbb{D}$.

Result: A tree T together with its edge lengths.

begin

Initial step:

(i): Find the median species *med* using Equation 5.7.

(ii): Set $\mathcal{T}_{\mathcal{S}} = \{T_1, \ldots, T_n\} - \{T_{med}\}$ and for each $T_i \in \mathcal{T}_{\mathcal{S}}$ its representative leaf set $S_i^k = \{i\}$.

(iii): Find for each $T_i \in \mathcal{T}_{\mathcal{S}}$ its *potential neighbor* $T_{i'} \in \mathcal{T}_{\mathcal{S}}$ using Equation 5.8.

Selection step (largest path length criterion): Find the neighboring pair (T_{i_0}, T_{j_0}) using Equation 5.9.

Agglomeration step:

(i): Combine T_{i_0} and T_{j_0} into a new rooted tree $T_{\{i_0 j_0\}}$ with root $i_0 j_0$, and estimate new edge lengths of $T_{\{i_0 j_0\}}$ using Equation 5.5 based on shortest triplets.

(ii): Compute the $k-$representative leaf set $S_{i_0 j_0}^k$ of $T_{\{i_0 j_0\}}$ based on k-representative leaf sets $S_{i_0}^k$ and $S_{j_0}^k$ of T_{i_0} and T_{j_0}, respectively.

(iii): Compute the k-representative unclassified species set $S_0^k(S_{i_0 j_0})$ of $T_{\{i_0 j_0\}}$.

(iv): Delete T_{i_0} and T_{j_0} and add $T_{\{i_0 j_0\}}$ to $\mathcal{T}_{\mathcal{S}}$.

(v): Find the potential neighbor for the new rooted tree $T_{\{i_0 j_0\}}$ using Equation 5.8 based on representative sets, and replace $T_{i'}$ for each $T_i \in \mathcal{T}_{\mathcal{S}}$ by $T_{\{i_0 j_0\}}$ if $T_{\{i_0 j_0\}}$ is its potential neighbor.

Local rearrangement step: Apply the preorder traversal procedure to the rooted subtrees T_{i_0} and T_{j_0} of the new rooted tree $T_{\{i_0 j_0\}}$ based on only shortest triplets.

Stopping step: If $|\mathcal{T}_{\mathcal{S}}| > 1$, goto Selection step, otherwise output the tree.

end

Now we briefly describe the complexity of the STC. At the initial step, (i), (ii), and (iii) are done in $O(n^2)$, $O(n)$ and $O(n^2)$ time, respectively. Thus, the complexity of the initial step is $O(n^2)$. The selection step is done in $O(n)$. At the agglomeration step, (i), (ii), (iii), (iv), and (v) are done in $O(k^3)$, $O(k)$, $O(nk^2)$, $O(1)$, and $O(nk^2)$ time, respectively. Thus, the complexity of the agglomeration step is $O(nk^2 + k^3)$.

Finally, we are estimating the complexity of the preorder traversal procedure based on only shortest triplets. Step 1 is done in constant time. Step 2, the nearest neighbor interchange operation around the root of T_i costs $O(k^3)$. Estimating new edge lengths is done in $O(k^3)$ time. Re-computing the k-representative leaf set S_i^k of T_i based on k-representative leaf sets of its rooted subtrees T_x and T_y costs $O(k)$ time. Finally, re-computing the k-representative unclassified species set $S_0^k(S_i)$ of T_i based on the k-representative leaf set S_j^k of T_j and the k-representative unclassified species set $S_0^k(S_{ij})$ of $T_{\{ij\}}$ is done in $O(k)$ time. Thus, the complexity of step 2 is $O(k^3)$. Step 3 is done in constant time. Step 1, step 2, and step 3 are repeated $O(n)$ times so the complexity of the preorder traversal procedure is $O(nk^3)$.

In the STC algorithm, the selection step, the agglomeration step and the local rearrangement step are repeated $(n-2)$ times so the overall complexity of the STC algorithm is $O(n^2k^3)$. Practically, we chose $k = 5$ as a good compromise between the accuracy and computational complexity for all data sets. That is, the practical complexity of the STC algorithm is only $O(n^2)$.

5.4 Results

Simulations were run on a PC cluster with 16 nodes. Each node has two 1.8 GHz processors and 2GB RAM. Seq-Gen (Rambaut and Grassly, 1997) was used to evolve sequences along trees using the Kimura two-parameter model (Kimura, 1980) with a transition/transversion ratio of 2.0. We generated 100 simulated data sets of 500 sequences each with sequence lengths 500, 1000 and 2000 nucleotides (nt), respectively. As one model tree, we used the rbcl gene tree with diameter 0.36 substitutions per site as inferred from an alignment of 500 rbcl-genes in **Chapter** 4. We call this *the rbcl-simulation*.

In a second experiment, the so-called *large simulation*, tree topologies were drawn from the Yule-Harding distribution (Harding, 1971), and edge lengths were drawn from an exponential distribution and subsequently rescaled such that the mean diameter of the tree was either 0.1, 0.5, 1.0, or 1.5. For each value of the diameter we generated 100 trees with 1000 sequences and 100 trees with 5000 sequences. Thus, a total of 800 trees were used.

Finally, we tested the accuracy and runtime of the STC and compared it with six other commonly used distance-based methods. More specifically, we investigate the performance of the Neighbor-Joining method (NJ) (Saitou and Nei, 1987) implemented in

Table 5.1: The average Robinson and Foulds distance of 100 simulated data sets of 500 sequences each with sequence lengths 500, 1000 and 2000 nt (rbcl simulation).

sequence length	NJ	BIONJ	Weighbor	HGT/FP	GME	BME	**STC**$^{k=5}$
500	.190	.188	.194	.512	.240	.184	**.177**
1000	.100	.098	.099	.409	.144	.096	**.088**
2000	.049	.048	.050	.313	.082	.046	**.040**

(a) Methods are used without BNNI

sequence length	NJ	BIONJ	Weighbor	HGT/FP	GME	BME	**STC**$^{k=5}$
500	.162	.162	.162	.166	.163	.163	**.162**
1000	.079	.079	.079	.079	.080	.079	**.079**
2000	.035	.035	.035	.036	.036	.035	**.035**

(b) Methods are used with BNNI

PAUP* 4.0 (Swofford, 2002), BIONJ (Gascuel, 1997), Weighbor 1.2 (Bruno *et al.*, 2000), Harmony Greedy Triplet and Four Point Condition (HGT/FP) (Csürös, 2002) as well as Greedy Minimum Evolution (GME) and Balanced Minimum Evolution (BME) (Desper and Gascuel, 2002). Unfortunately, no distance-based program is available for the disk-covering method (Huson *et al.*, 1999). All methods were combined with DNADIST version 3.5 (Felsenstein, 1993) and pairwise distances were corrected for multiple hits according to the model used in the simulation. Moreover, we examined the performance of all methods when the balanced nearest neighbor interchange (BNNI) (Desper and Gascuel, 2002) is used as a post-processing step.

Further, to illustrate the performance of STC we re-analyzed the 96-taxon alignments of sequence length 500 nt, that were analyzed in (Desper and Gascuel, 2002) and available at (`http://www.lirmm.fr/~guindon/simul/`). The 6000 trees were split into three groups called "slow" (0.2 substitutions per site), "moderate" (0.4 substitutions per site) and "fast" (1.0 substitutions per site). We call this *the re-analyzed simulation.*

The accuracy of a tree reconstruction method for a simulated data set is measured by the average Robinson and Foulds distance (Robinson and Foulds, 1981) between the inferred tree and the model tree used to generate the data set. Recall that the smaller the RF distance is between the inferred tree and the model tree the higher is the topological

Table 5.2: The average Robinson and Foulds distance of 100 simulated data sets of 1000 taxa for each tree diameter 0.1, 0.5, 1.0 and 1.5 and with sequence length 1000 nt (large simulation).

number sequences	NJ	BIONJ	HGT/FP	GME	BME	**STC**$^{k=5}$
1000 (0.1)	.146	.146	.378	.168	.143	**.139**
1000 (0.5)	.093	.089	.193	.126	.075	**.066**
1000 (1.0)	.094	.090	.188	.132	.074	**.062**
1000 (1.5)	.097	.091	.182	.138	.073	**.061**

(a) Methods are used without BNNI

number sequences	NJ	BIONJ	HGT/FP	GME	BME	**STC**$^{k=5}$
1000 (0.1)	.137	.137	.137	.137	.137	**.138**
1000 (0.5)	.061	.061	.061	.061	.061	**.061**
1000 (1.0)	.057	.057	.057	.057	.057	**.056**
1000 (1.5)	.055	.055	.055	.055	.055	**.055**

(b) Methods are used with BNNI

accuracy of the tree reconstruction method.

In the following we discuss the results of *the rbcl-simulation*, and *the large simulation* and *the re-analyzed simulation*.

5.4.1 rbcl-simulation

Table 5.1(a) shows that the STC outperforms all other methods analyzed in terms of topological accuracy. For instance, the RF distance between the STC-tree and the model tree is on average 0.177 (with respect to the sequence length of 500 nt) and better than NJ (0.190), slightly better than the second best method BME (0.184) and much better than HGT/FP (0.512). Table 5.1a also demonstrates that all tested methods including STC give higher topological accuracy when the sequence length is increased. However, Table 5.1b shows that other methods in combination with BNNI outperform STC without BNNI. The combination of STC and BNNI shows similar performance as the combinations of NJ (BIONJ, Weighbor) and BNNI and, slightly better results than

Table 5.3: The average Robinson and Foulds distance of 100 simulated data sets of 5000 taxa for each tree diameter 0.1, 0.5, 1.0 and 1.5 and with sequence length 1000 nt (large simulation)

number sequences	NJ	BIONJ	HGT/FP	GME	BME	**STC**$^{k=5}$
5000 (0.1)	.178	.179	.442	.207	.173	**.170**
5000 (0.5)	.109	.105	.210	.156	.084	**.072**
5000 (1.0)	.107	.102	.192	.155	.073	**.064**
5000 (1.5)	.112	.106	.188	.164	.072	**.063**

(a) Methods are used without BNNI

number sequences	NJ	BIONJ	HGT/FP	GME	BME	**STC**$^{k=5}$
5000 (0.1)	.168	.168	.168	.168	.168	**.168**
5000 (0.5)	.066	.066	.066	.066	.066	**.066**
5000 (1.0)	.057	.057	.057	.057	.057	**.057**
5000 (1.5)	.055	.055	.055	.055	.055	**.055**

(b) Methods are used with BNNI

the combination of GME (HGT/FP) and BNNI.

5.4.2 Large simulation

Due to the increase in runtime, Weighbor could not be tested. Table 5.2a and 5.3a show that STC gives better results than the other methods independent of the diameter. All methods display a decrease in accuracy when the number of sequences changes from 1000 to 5000. As shown in Table 5.2b and 5.3b, BNNI boosts the accuracy of all methods including STC. All methods give similar results when being used together with BNNI.

5.4.3 Re-analyzed simulation

Except for STC, the accuracies for the other methods displayed in Table 5.4a and 5.4b were taken from Desper and Gascuel (2002). Table 5.4a shows that STC outperforms the other methods in terms of topological accuracy with the exception that Weighbor is

Table 5.4: The average RF distance of the 96-taxon alignments of sequence length 500 nt, that were analyzed in (Desper and Gascuel, 2002). The 6000 trees were split into three groups called "slow" (0.2 substitutions per site), "moderate" (0.4 substitutions per site) and "fast" (1.0 substitutions per site). Except for STC, the accuracies for the other methods were taken from (Desper and Gascuel, 2002)

number sequences	NJ	BIONJ	Weighbor	HGT/FP	GME	BME	**STC**$^{k=5}$
96 (slow)	.183	.180	.178	.512	.199	.186	**.179**
96 (moderate)	.136	.134	.129	.480	.158	.137	**.125**
96 (fast)	.115	.112	.103	.465	.144	.117	**.102**

(a) Methods are used without BNNI

number sequences	NJ	BIONJ	Weighbor	HGT/FP	GME	BME	**STC**$^{k=5}$
96 (slow)	.173	.173	.173	.175	.173	.173	**.173**
96 (moderate)	.119	.118	.118	.123	.118	.118	**.116**
96 (fast)	.090	.090	.091	.098	.091	.090	**.090**

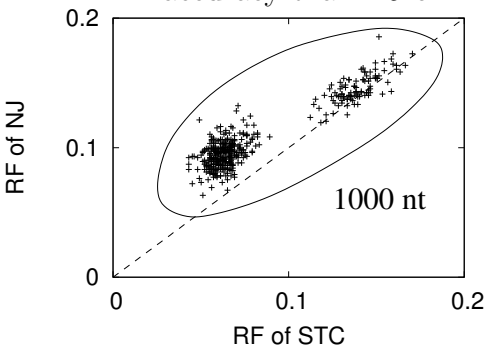
(b) Methods are used wit BNNI

slightly better than STC with respect to the slow simulation group. If BNNI is applied, all methods exhibit an almost identical performance (see Table 5.4b).

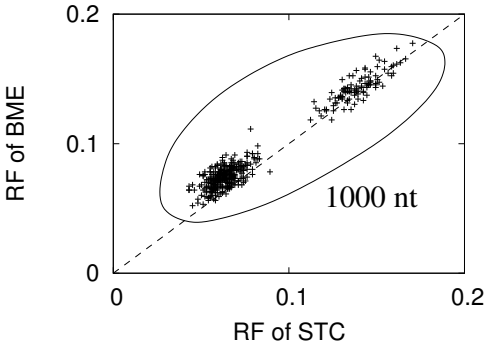
5.4.4 Another look at the performance

Instead of looking at the average RF distance, we suggest to take a closer look at the simulated data. For each simulated data set, that is subjected to the STC and six other tree reconstruction methods mentioned above, we compute the RF distance between the reconstructed tree and the model tree for all methods. Figure 5.5 illustrates the results for the large simulation when comparing STC with NJ (left column) and STC with the second best method BME (right column). In each diagram specified by the number of taxa and reconstruction methods, 400 points are displayed, that resulted from 100 simulations for each of the tree-diameters (0.1, 0.5, 1.0 and 1.5). Although four tree-diameters were studied only two clouds of points are discernible, where the cloud in

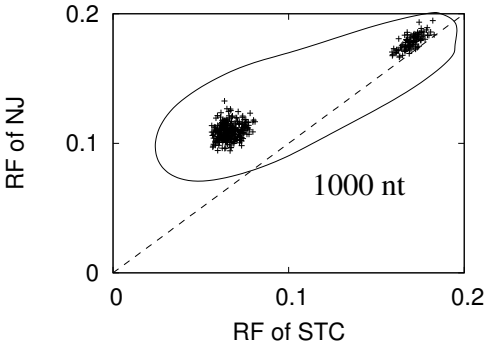
Figure 5.5: The comparisons of topological accuracy between STC, NJ and BME for the *large simulation*. Each point in the graph presents the Robinson and Foulds (RF) distance for a simulated data set. Points above the dotted line are examples where the RF distance of the STC-tree is less than the RF distance of the NJ-tree or BME-tree. Thus, the STC gives higher topological accuracy than NJ or BME with respect to the simulated data set.



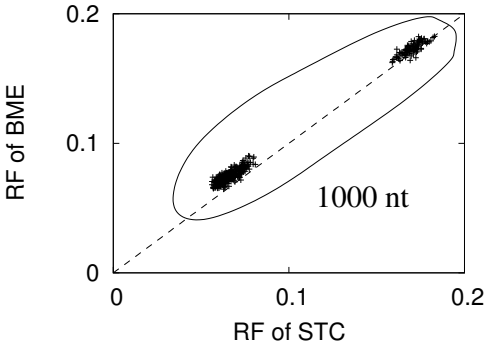
(a) STC versus NJ (1000 taxa)



(b) STC versus BME (1000 taxa)



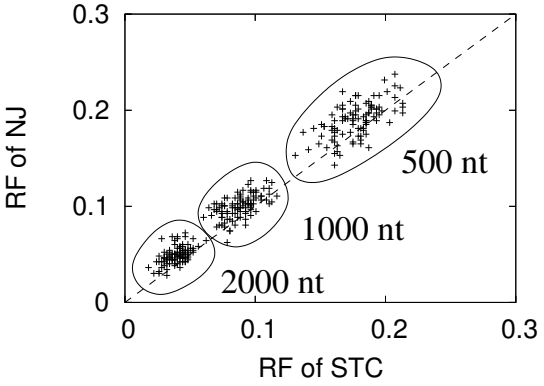
(c) STC versus NJ (5000 taxa)



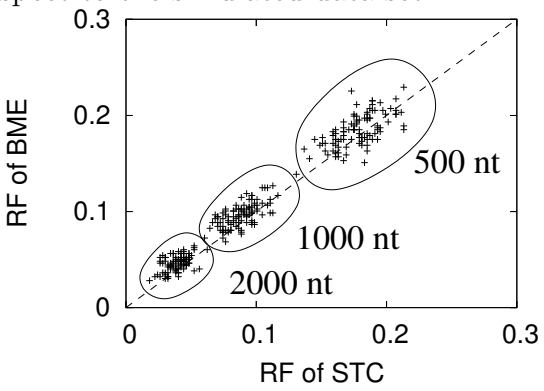
(d) STC versus BME (5000 taxa)

the north-east corner of each diagram represents the simulations with the tree-diameter 0.1. The remaining 300 points gather in the south-west cloud because the RF-distances from trees with diameter 0.5, 1.0, 1.5 are not substantially different from each other (see Table 5.2a and 5.3a). More precisely, the horizontal and vertical axes indicate the RF distances of STC and NJ (or BME), respectively. Each point in the graph presents the RF distance for a simulated data set. Points above the dotted line are examples where the RF distance of the STC-tree is less than the RF distance of the NJ-tree or BME-tree. Thus, the STC gives higher topological accuracy than NJ or BME with respect to the simulated data set. For example, Figure 5.5a illustrates the comparison between STC and NJ with respect to 1000 taxa data sets. 379 out of 400 points are above

Figure 5.6: The comparisons of topological accuracy between STC, NJ and BME for the *rbcl simulation*. Each point in the graph presents the Robinson and Foulds (RF) distance for a simulated data set. Points above the dotted line are examples where the RF distance of the STC-tree is less than the RF distance of the NJ-tree or BME-tree. Thus, the STC gives higher topological accuracy than NJ or BME with respect to the simulated data set.



(a) STC versus NJ (500 sequences)



(a) STC versus BME (500 sequences)

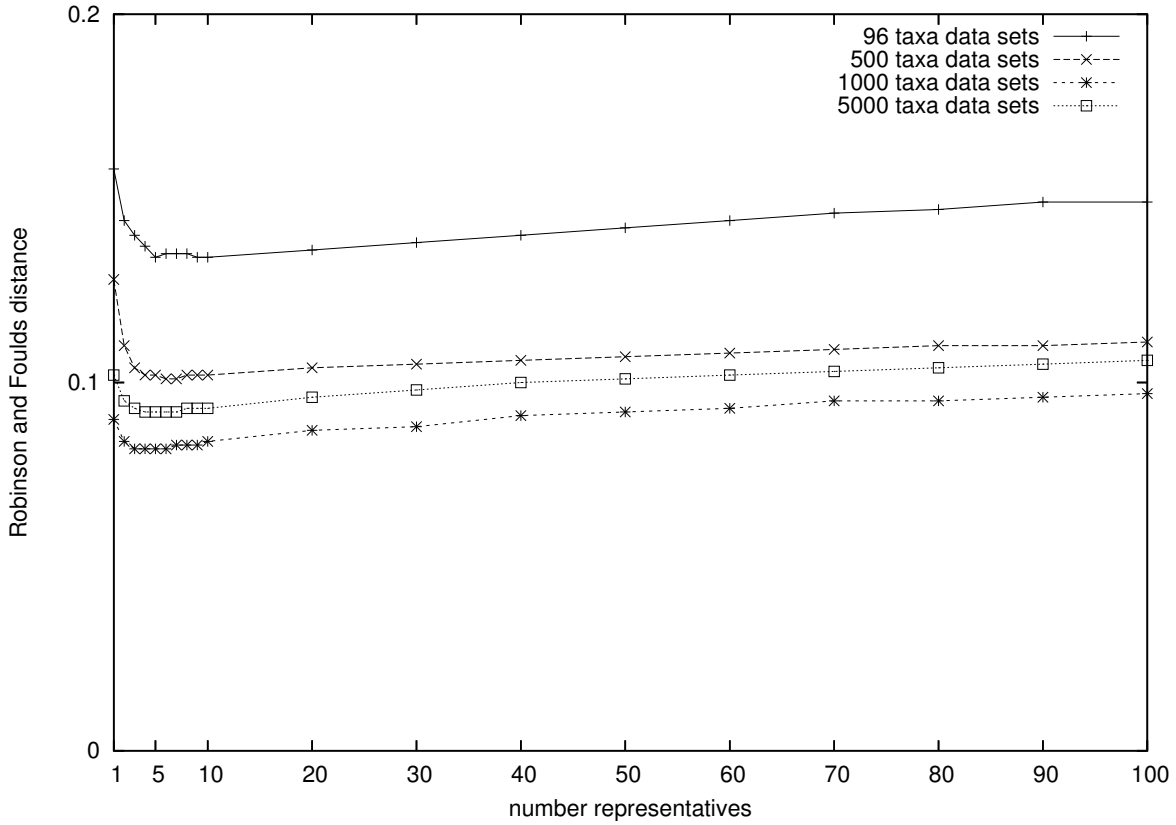
the diagonal, thus, STC gives better results than NJ in about 95% of the simulations. For the remaining 21 alignments (points), two methods showed the same RF distance. Finally, we found 19 points below the diagonal in which case NJ outperforms STC. For the *large simulation* (5000 taxa), NJ is worse than STC in all cases. However, the second best method BME is better than STC in 11% and 5% of the cases with respect to 1000 and 5000 sequence data sets.

Figure 5.6 shows the same analysis for the rbcl simulation. It shows that with increasing sequence length the cloud of points moves towards zero. From Figure 5.6 we learn that in some instances NJ (or BME) performs better (with regard to the RF distance) than STC, i.e. 20%, 12%, 8% (or 34%, 17%, 14%) of the simulations for sequence lengths 500, 1000 and 2000 nt, respectively.

Similar results hold for the other methods. These results are summarized in Table 5.5a where we show the percentage of simulations in which STC is at least as good as the other methods.

Again, if BNNI is applied we observe that no substantial difference among the various approaches. The accuracy of the methods is mostly determined by BNNI (see Table 5.5b).

Figure 5.7: The impact of the number of representatives k. The RF distance of STC decreases when k grows from 1 to 5. When k ranges from 5 to 10, the RF distance remains more or less unchanged. For $k \geq 10$, the RF distance increases steadily indicating a loss of accuracy.



5.5 Discussions

We are presenting k-representative sets which allow us to design a fast and accurate method to reconstruct phylogenies from large data sets with 1000 or more taxa. Simulations show that STC gives better results than other tested methods in terms of topological accuracy. However, if BNNI is introduced as a subsequent optimization step, the differences in the performance disappear. All methods show more or less the same accuracy. Thus, one should apply BNNI to improve the topological accuracy.

The time to reconstruct a tree of up to 1000 sequences is not really an issue for all tested distance-based methods, with the exception of Weighbor. Weighbor needed about 19 minutes to reconstruct a tree with 500 sequences, thus it is only applicable to data sets with up to some hundred sequences. For data sets with up to 1000 sequences, the

Table 5.5: The performance of STC compared to other methods

number sequences	NJ	BIONJ	Weighbor	HGT/FP	GME	BME
96 (500 nt)	68 (16)	65 (15)	57 (16)	100 (0)	73 (10)	70 (14)
500 (500 nt)	80 (4)	76 (4)	88 (3)	100 (0)	100 (0)	66 (1)
500 (1000 nt)	88 (3)	79 (4)	84 (4)	100 (0)	100 (0)	83 (6)
500 (2000 nt)	92 (6)	90 (4)	92 (3)	100 (0)	100 (0)	86 (9)
1000 (1000 nt)	95 (2)	95 (1)	n.d.	100 (0)	100 (0)	89 (15)
5000 (1000 nt)	100 (0)	99 (0)	n.d.	100 (0)	100 (0)	95 (1)

(a) The percentage of cases where STC is at least as good as other tested methods in terms of RF distance. The number in parentheses is the percentage of cases where STC is equally good as other tested methods. Methods are used without BNNI.

number sequences	NJ	BIONJ	Weighbor	HGT/FP	GME	BME
96 (500 nt)	9 (8)	8 (8)	10 (10)	12 (10)	10 (8)	10 (9)
500 (500 nt)	34 (37)	35 (39)	35 (36)	59 (29)	46 (33)	41 (39)
500 (1000 nt)	22 (19)	17 (23)	18 (22)	23 (28)	30 (20)	24 (20)
500 (2000 nt)	10 (13)	8 (7)	10 (8)	9 (8)	12 (10)	7 (10)
1000 (1000 nt)	30 (28)	27 (29)	n.d.	28 (22)	30 (24)	28 (27)
5000 (1000 nt)	48 (40)	42 (44)	n.d.	45 (45)	52 (37)	43 (43)

(b) The percentage of cases where STC is better than other tested methods in terms of RF distance. The number in parentheses is the percentage of cases where STC is worse than other tested methods. Methods are used with BNNI.

remaining methods needed less than one minute to output a tree, thus the difference between methods in terms of runtime is not significant. For data sets with 5000 sequences, STC (GME, HGT/FP or BME) with BNNI took about 2.0 (2.5, 3.0 or 3.5) minutes to reconstruct a tree. NJ (BIONJ) with BNNI were slower and consumed approximately six minutes to output a tree. In short, the combination of STC and BNNI efficiently reconstruct trees for large data sets in both terms of topological accuracy and runtime.

Finally, we did not systematically evaluate the impact of the number of representatives k. We present some preliminary results for k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. Figure 5.7 shows that the RF distance of STC decreases when k grows from 1 to 5. This proves our intuition that a too small number of triplets

leads to an inaccurate estimate of path lengths and edge lengths. When k ranges from 5 to 10, the RF distance remains more or less unchanged. For $k \geq 10$, the RF distance increases steadily indicating a loss of accuracy. The decrease in accuracy is explained by the inclusion of triplets with large distances which include noise and disturb the reconstruction. Thus, we chose $k = 5$ as a good compromise between the accuracy and computational complexity for all data sets. That is, the practical complexity of the STC algorithm is only $O(n^2)$.

6 Summary

The reconstruction of evolutionary relationships among contemporary species is a cornerstone of phylogenetic analysis. Nowadays, phylogenetic trees are typically constructed based on nucleotide and amino acid sequences. Since huge amounts of nucleotide and amino acid sequences can be obtained easily from public databases like GenBank, they allow us to study the evolutionary relationships among a large range of species. However, they cause an urgent need of approaches for construction large phylogenetic trees.

This thesis provides four contributions to the phylogenetic tree reconstruction. First, essential concepts and notations to model and present the evolution of species were introduced concisely but thoroughly for both molecular biologists and computer scientists. Also, commonly used phylogenetic tree construction methods such as maximum parsimony, maximum likelihood or minimum evolution approaches were fully presented in the algorithmic style. Second, we proposed a novel search strategy, called *phylogenetic navigator* (PhyNav), in evolutionary tree construction. The search gives encouraging results compared to other methods. Third, a new quartet-based maximum likelihood approach, namely *important quartet puzzling and nearest neighbor interchange* (IQPNNI), was described to construct large phylogenies with up to 1000 sequences in practical time. Fourth, a new distance-based clustering algorithm, called *shortest triplet clustering* (STC), which is able to build extremely large phylogenies was presented. For example, the clustering algorithm constructs a phylogeny tree for 5000 species within a minute. Remarkably, all proposed methods were implemented in computer softwares and examined extensively on both real and simulated datasets.

Introduction to phylogenetic tree reconstruction (Chapter 2): This chapter introduced different kinds of biological data e.g. morphological characters, nucleotide sequences, amino acid sequences and gene-orders which carry phylogenetic signals to investigate the evolutionary relationships of contemporary species. In addition, the evolutionary process of nucleotides and amino acids which are prevalent in phylogenetic analysis was modeled using Markov chain techniques. The statistical description of

sequence evolution allows us to study the historical relationships of species under the statistical framework. Moreover, graph theory and phylogenetic trees were described comprehensively to present compactly the evolution of species.

Furthermore, perhaps more importantly, we thoroughly summarized widely used phylogenetic tree construction methods e.g. maximum parsimony methods, maximum likelihood methods, and distance-based algorithms. Both advantages and disadvantages of methods were discussed in terms of accuracy and runtime. Since reconstructing phylogenies with respect to an optimality criterion is computational expensive, efficient heuristic strategies e.g. hill climbing and stepwise addition were introduced to construct phylogenies in practical time.

Phylogenetic navigator (Chapter 3): We proposed a novel strategy for constructing large phylogenetic trees. The key idea is the definition of the so-called minimal k-distance subsets. Each minimal k-distance subset consists of at most k sequences but retains most of the relevant phylogenetic information from the whole sequence set. For each subset the subtree is created faster and serves as a scaffold to construct the full tree for the whole sequence set. Because many minimal subsets exist the procedure is repeated several times and the best tree with respect to some optimality criterion is considered as the inferred phylogenetic tree.

The search strategy was implemented in PhyNav package using the maximum likelihood principle. PhyNav gave encouraging results compared to other programs on both simulated and real datasets.

Important quartet puzzling and nearest neighbor interchange (Chapter 4): Although quartet-based methods are widely used in phylogenetic analysis, the complexity of $O(n^4)$ prevents these approaches from applying to large datasets with more than approximately 100 sequences.

To overcome the computational burden, we proposed the important quartet puzzling method (IQP) which uses only $O(n^2)$ important quartets to construct a tree. The IQP approach was implemented as a part of a combined algorithm IQPNNI, which efficiently elucidates the landscape of possible optimal trees. Finally, we applied an estimator that is based on the time series of sights of better trees during the tree search to estimate when it becomes unlikely that a further search for a better tree will be successful.

Experiments with both biological and simulated datasets showed that IQPNNI gave better accuracy than other tested methods e.g. fastDNAml (Olsen *et al.*, 1994), Weigh-

bor (Bruno *et al.*, 2000), MetaPIGA (Lemmon and Milinkovitch, 2002), and PHYML (Guindon and Gascuel, 2003). However, the computational cost of IQPNNI was more expensive than other methods except fastDNAml. Nevertheless, the times needed are not unrealistic. For example, the computation of the maximum likelihood tree of 500 rbcl sequences took less than 10 hours. This is only a small amount of time compared to the time it took to obtain the data.

Shortest triplet clustering algorithm (Chapter 5): Maximum likelihood methods PhyNav and IQPNNI are quite efficient to construct phylogenies up to with 1000 sequences. However, they are unlikely proper to build phylogenies for larger datasets of say thousands of sequences due to the computational expense. To overcome the problem, we proposed a new distance-based clustering algorithm, namely shortest triplet clustering, to construct a phylogeny for n sequences in $O(n^2)$ time. Therefore, the STC algorithm can build large phylogenies with up to five thousands of sequences within a minute.

The efficiency of STC and various distance-based methods were examined with a large range of simulated datasets. STC showed better performance than widely used distance-based methods e.g. Neighbor Joining algorithm (Saitou and Nei, 1987), BIONJ (Gascuel, 1997) and Weighbor (Bruno *et al.*, 2000) in terms of both accuracy and runtime. However, the balanced nearest neighbor interchange (Desper and Gascuel, 2002) is recommended as a post-processing step for further topological accuracy improvements.

Appendix

IQPNNI package

IQPNNI is a computer program to reconstruct the evolutionary relationships among contemporary species based on nucleotide sequences, amino acid sequences or protein-coding nucleotide sequences. It is able to construct maximum likelihood phylogenies with up to a thousand of sequences. IQPNNI is menu-driven program which allows users to specify the parameter values or let the program estimate them from input data.

IQPNNI is able to work with different models of sequence evolution described as follows

- **Nucleotide sequences:** JC69 (Jukes and Cantor, 1969), K2P (Kimura, 1980), HKY (Hasegawa *et al.*, 1985), TN93 (Tamura and Nei, 1993), and general time reversible model (GTR) (Tavaré, 1986).
- **Protein sequences:** Dayhoff (Dayhoff *et al.*, 1978), BLOSUM62 (Henikoff and Henikoff, 1992), JTT (Jones *et al.*, 1992), mtREV24 (Adachi and Hasegawa, 1996), WAG (Whelan and Goldman, 2001), VT (Müller and Vingron, 2000).
- **Protein-coding nucleotide sequences:** GY94 (Goldman and Yang, 1994), NY98, YN98 (Nielsen and Yang, 1998) and CpG Depression (Pedersen *et al.*, 1998). The models of protein-coding nucleotide sequences were implemented by Bui Quang Minh and available in IQPNNI package version 3.0.

Moreover, different models of rate heterogeneity e.g. two-state model, Γ-distribution model, or site-specific substitution rate model were implemented in IQPNNI package.

The parallel version of IQPNNI called pIQPNNI was implemented by Minh *et al.* (2005) using message passing interface (MPI). Both sequential and parallel versions

were written in the object-oriented language C++. They are available on all popular platforms e.g. MacOX, Linux, Windows at `http://www.bi.uni-duesseldorf.de/software/iqpnni`.

PhyNav package

Like the IQPNNI package, PhyNav constructs phylogenies based on the maximum likelihood principle for both nucleotide sequences or amino acid sequences. It is able to work with all models of sequence evolution and rate heterogeneity as described in the IQPNNI package.

Sequential version of PhyNav on Linux is available at `http://www.bi.uni-duesseldorf.de/software/phynav`. We are planning to thoroughly investigate both advantages and disadvantages of the search strategy.

STC package

STC is a distance-based phylogenetic tree construction program. The main advantage of STC is the ability of building very large phylogenies. For example, STC constructs a phylogeny with 5000 sequences in a minute. However, the balanced nearest neighbor interchange is recommended as a post-processing step for further topological accuracy improvements (Desper and Gascuel, 2002).

STC takes distance matrices in PHYLIP format and constructs for each distance matrix a phylogenetic tree. The program was written in object-oriented language C++. Therefore, it can run on all popular platforms e.g. MacOX, Linux, Windows. STC is freely downloaded at `http://www.bi.uni-duesseldorf.de/software/stc`.

Bibliography

Adachi, J. and Hasegawa, M. (1996) Model of amino acid substitution in proteins encoded by mitochondrial DNA. *J. Mol. Evol.*, **42**, 459–468.

Aho, A. V., Hopcroft, J. E. and Ullman, J. D. (1974) *The Design and Analysis of Computer Algorithms*. Addison-Wesley Publishing Company.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. and Wheeler, D. L. (2005) GenBank. *Nucl. Acids Res.*, **33**, D34–D38.

Beyer, W., Stein, M., Smith, T. and Ulam, S. (1974) A molecular sequence metric and evolutionary trees. *Mathematical Biosciences*, **19**, 9–25.

Brauer, M. J., Holder, M. T., Dries, L. A., Zwickl, D. J., Lewis, P. l. O. and Hillis, D. M. (2002) Genetic algorithms and parallel processing in maximum-likelihood phylogeny inference. *Mol. Biol. Evol.*, **19**, 1717–1726.

Brent, P. R. (1973) *Algorithms for minimization without derivatives*. Prentice-Hall, Englewood Cliffs, NJ.

Brown, T. (2002) *Genomes*. BIOS Scientific Publishers Ltd, Oxford, UK, Second edn..

Bruno, W. J., Socci., N. D. and Halpern, A. L. (2000) Weighted Neighbor Joining: A likelihood based-approach to distance-based phylogeny reconstruction. *J. Mol. Evol.*, **17**, 189–197.

Buneman, P. (1971) The recovery of trees from measures of dissimilarity. In Hodson., Lendall. and Tautu (eds.), *Mathematics in the archaeological and historical sciences*, Edinburgh University Press, Edinburgh.

Cavalli-Sforza, L. and Edwards, A. W. F. (1967) Phylogenetic analysis: Models and estimation procedures. *American Journal of Human Genetic*, **19**, 233–257.

Charleston, M. A. (2001) Hitch-hiking: A parallel heuristic search strategy, applied to the phylogeny problem. *J. Comput. Biol.*, **8**, 79–91.

Chor, B., Hendy, M. D., Holland, B. R. and Penny, D. (2000) Multiple maxima of likelihood in phylogenetic trees: An analytic approach. *Mol. Biol. Evol.*, **17**, 1529–1541.

Chor, B. and Tuller, T. (2005) Maximum likelihood of evolutionary trees is hard. In *Proceedings of the 9th Annual International Conference on Research in Computational Molecular Biology (RECOMB 2005)*, vol. 3500 of *Lecture Notes in Computer Science*, pages 296–310, New York, USA, ACM Press.

Churchill, G. A., von Haeseler, A. and Navidi, W. C. (1992) Sample size for phylogenetic inference. *Mol. Biol. Evol.*, **9**, 753–769.

Cooke, P. (1980) Optimal linear estimation of bounds of random variables. *Biometrika*, **67**, 257–258.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2001) *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, Second edn..

Csürös, M. (2002) Fast recovery of evolutionary trees with thousands of nodes. *J. Comput. Biol.*, **9**, 277–297.

Darwin, C. (1872) *On the Origin of Species*. John Murray, London, 6th edn..

Day, W. H. E. and David, S. (1986) Computational complexity of inferring phylogenies by compatibility. *Syst. Zool.*, **35**, 224–229.

Dayhoff, M. O., Schwartz, R. M. and Orcutt, B. C. (1978) A model of evolutionary change in proteins. In Dayhoff *et al.* (1978), pages 345–352.

Desper, R. and Gascuel, O. (2002) Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *J. Comput. Biol.*, **9**, 687–706.

Eck, R. and Dayhoff, M. O. (1966) *In Atlas of protein sequence and structure*. National Biomedical Research Foundation, Silver Spring, Maryland, USA.

Edgar, R. C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucl. Acids Res.*, **32**, 1792–1797.

Edwards, A. W. F. and Cavalli-Sforza, L. L. (1963) The reconstruction of evolution. *Annals of Human Genetics*, **27**, 105–106.

Farris, J. (1970) Methods for computing wager trees. *Syst. Zool.*, **19**, 83–92.

Farris, J. (1977) *On the phenetic approach to vertebrate classification*, vol. 17. Plenum, New York.

Felsenstein, J. (1978) The number of evolutionary trees. *Syst. Zool.*, **27**, 27–33.

Felsenstein, J. (1981a) Evolutionary trees from DNA sequences: a maximum likelihood approach. *J. Mol. Evol.*, **17**, 368–376.

Felsenstein, J. (1981b) Evolutionary trees from gene frequencies and quantitative characters: Finding maximum likelihood estimates. *Evolution*, **35**, 1229–1242.

Felsenstein, J. (1993) *PHYLIP (Phylogeny Inference Package) version 3.5c*. Department of Genetics, University of Washington, Seattle, Distributed by the author.

Felsenstein, J. (2004) *Infering Phylogenies*. Sinauer Associates, Sunderland, Massachusetts.

Fitch, W. and Margoliash, E. (1967a) Construction of phylogenetic trees. *Science*, **155**, 279–284.

Fitch, W. and Margoliash, E. (1967b) A method for estimating the number of invariant amino acid position in a gene using cytochrome c as a model case. *Biochem. Gene*, **1**, 65–71.

Fitch, W. M. (1971) Toward defining the course of evolution: Minimum change for a specific tree topology. *Syst. Zool.*, **20**, 406–416.

Fitch, W. M. (2000) Homology – a personal view on some of the problems. *Trends Genetic*, **16**, 227–231.

Gascuel, O. (1997) BIONJ: An improved version of the NJ algorithm based on a simple model of sequence data. *Mol. Biol. Evol.*, **14**, 685–695.

Goldman, N. and Yang, Z. (1994) A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol. Biol. Evol.*, **11**, 725–736.

Goloboff, P. A. (1999) Analyzing large data sets in reasonable times: Solutions for composite optima. *Cladistics*, **15**, 415–428.

Gondran, M., Minoux, M. and Vajda, S. (1984) *Graphs and algorithms*. John Wiley and Sons Ltd.

Graham, R. L. and Foulds, L. R. (1982) Unlikelihood that minimal phylogenies for a realistic biological study can be constructed in reasonable computational time. *Mathematical Biosciences*, **60**, 133–142.

Gu, X., Fu, Y.-X. and Li, W.-H. (1995) Maximum likelihood estimation of the heterogeneity of substitution rate among nucleotide sites. *Mol. Biol. Evol.*, **12**, 546–557.

Gu, X. and Li, W. H. (1996) A general additive distance with time-revesibility and rate variation among nucleotide sites. *Proc. Natl. Acad. Sci. USA*, **93**, 4671–4676.

Guindon, S. and Gascuel, O. (2003) A simple, fast and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst. Biol.*, **52**, 696–704.

von Haeseler, A. (1999) Model based phylogenetic inference. *Bolyai Soc. Math. Studies*, **7**, 307–321.

Harding, E. F. (1971) The probabilities of rooted tree-shapes generated by random bifurcation. *Adv. Appl. Prob*, **3**, 44–77.

Hartigan, A. J. (1975) *Clustering Algorithms*. John Wiley and Sons, Inc.

Hasegawa, M., Kishino, H. and Yano, T.-A. (1985) Dating of the human–ape splitting by a molecular clock of mitochondrial DNA. *J. Mol. Evol.*, **22**, 160–174.

Hein, J. J. (1989) An optimal algorithm to reconstruct trees from additive data. *Bulletin of mathematical biology*, **51**, 597–603.

Henikoff, S. and Henikoff, J. G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, **89**, 10915–10919.

Higgins, D. (2003) Multiple alignment. In Salemi, M. and Vandamme, A.-M. (eds.), *The Phylogenetics Handbook A Practical Approach to DNA and Protein Phylogeny*, pages 45–60, Cambridge University Press, Cambridge.

Hillis, D. M. and Wiens, J. J. (2000) Molecules versus morphology in systematics: Conflicts, artifacts and misconceptions. In Wiens, J. J. (ed.), *Phylogenetic analysis of morphological data*, pages 1–19, Smithsonian Institution press, Washington and London.

Huson, D. H., Nettles, S. M. and Warnow, T. J. (1999) Disk-covering, a fast-converging method for phylogenetic reconstruction. *J. Comput. Biol.*, **6**, 369–386.

Jones, D. T., Taylor, W. R. and Thornton, J. M. (1992) The rapid generation of mutation data matrices from protein sequences. *Comput. Appl. Biosci.*, **8**, 275–282.

Jukes, T. H. and Cantor, C. R. (1969) Evolution of protein molecules. In Munro, H. N. (ed.), *Mammalian Protein Metabolism*, vol. 3, pages 21–123, Academic Press, New York.

Keane, T. M., Naughton, T. J., Travers, S. A. A., McInerney, J. O. and McCormack, G. P. (2005) DPRml: distributed phylogeny reconstruction by maximum likelihood. *Bioinformatics*, **21**, 969–974.

Keilson, J. (1979) *Markov chain models: rarity and exponentially*. Spinger, New York, USA.

Kidd, K. K. and Sgaramella-Zonta, L. A. (1971) Phylogenetic analysis: Concepts and methods. *American Journal of Human Genetics*, **23**, 235–252.

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.

Klotz, L. C., Komar, N., Blanken, R. and Mitchell, R. M. (1979) Calculation of evolutionary trees from sequence data. *Proc. Natl. Acad. Sci. USA*, **76**, 4516–4520.

Kluge, A. and Farris, J. S. (1969) Quantitative pheletics and the evolution of anurans. *Syst. Biol.*, **18**, 1–32.

Krause, A., Nicodème, P., Bornberg-Bauer, E., Rehmsmeier, M. and Vingron, M. (1999) WWW-access to the SYSTERS protein sequence cluster set. *Bioinformatics*, **15**, 262–263.

Kumar, S. (1986) A stepwise algorithm for finding minimum evolutionary trees. *Mol. Biol. Evol.*, **13**, 584–593.

Lemmon, A. R. and Milinkovitch, M. C. (2002) The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation. *Proc. Natl. Acad. Sci. USA*, **99**, 10516–10521.

Li, W.-H. (1981) Simple method for constructing phylogenetic trees from distance matrices. *Proc. Natl. Acad. Sci. USA*, **78**, 1085–1089.

Maddison, D. R. (1991) The discovery amd importance of multiple islands of most parsimonous trees. *Syst. Biol.*, **42**, 200–210.

Matsuda, H. (1996) Protein phylogenetic inference using maximum likelihood with a genetic algorithm. In *Proceedings of the 1st Pacific Symposium on Biocomputing (PSB 1996)*, pages 512–523, Hawaii.

Meyer, S. and von Haeseler, A. (2003) Identifying site-specific substitution rates. *Mol. Biol. Evol.*, **20**, 182–189.

Minh, B. Q., Vinh, L. S., von Haeseler, A. and Schmidt, H. A. (2005) pIQPNNI-parallel reconstruction of large maximum likelihood phylogenies. *Bioinformatics*.

Moret, B. and Warnow, T. (2005) Advances in phylogeny reconstruction from gene order and content data. In Zimmer, E. and Roalson, E. (eds.), *Methods in Enzymology*, vol. 395, pages 673–700, Elsevier.

Müller, T. and Vingron, M. (2000) Modeling amino acid replacement. *J. Comput. Biol.*, **7**, 761–776.

Nei, M. and Kumar, S. (2000) *Molecular Evolution and Phylogenetics*. Oxford Univerity Press.

Nielsen, R. and Yang, Z. (1998) Likelihood models for detecting positively selected amino acid sites and applications to the HIV-1 envelope gene. *Genetics*, **148**, 929–936.

Nixon, K. C. (1999) The parsimony ratchet, a new method for rapid parsimony analysis. *Cladistics*, **15**, 407–414.

Notredame, C., Higgins, D. and Heringa, J. (2000) T-COFFEE: A novel method for multiple sequence alignments. *Journal of Molecular Biology*, **302**, 205–217.

Olsen, G. J., Matsuda, H., Hagstrom, R. and Overbeek, R. (1994) fastDNAml: A tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. *Comput. Appl. Biosci.*, **10**, 41–48.

Pedersen, A.-M. K., Wiuf, C. and Christiansen, F. B. (1998) A codon-based model designed to describe lentiviral evolution. *Mol. Biol. Evol.*, **15**, 1069–1081.

van de Peer, Y. (2003) Phylogeny inference based on distance methods. In Salemi, M. and Vandamme, A.-M. (eds.), *The Phylogenetics Handbook A Practical Approach to DNA and Protein Phylogeny*, pages 101–119, Cambridge University Press, Cambridge.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (2002) *Numerical Repcipes in C: The art of scientific computing*. Cambridge University Press, New York.

Quicke, D. L. J., Taylor, J. and Purvis, A. (2001) Changing the landscape: A new strategy for estimating large phylogenies. *Syst. Biol.*, **50**, 60–66.

Rambaut, A. and Grassly, N. C. (1997) Seq-Gen: An application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. *Comput. Appl. Biosci.*, **13**, 235–238.

Ranwez, V. and Gascuel, O. (2001) Quartet-based phylogenetic inference: Improvements and limits. *Mol. Biol. Evol.*, **18**, 1103–1116.

Roberts, D. L. and Solow, A. R. (2003) When did the dodo become extinct. *Nature*, **426**, 245–245.

Robinson, D. R. and Foulds, L. R. (1981) Comparison of phylogenetic trees. *Mathematical Biosciences*, **53**, 131–147.

Ronquist, F. and Huelsenbeck, J. P. (2003) MRBAYES 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, **19**, 1572–1574.

Roshan, U., Moret, B. M. E., Williams, T. L. and Warnow, T. (2004) Rec-I-DCM3: A fast algorithmic technique for reconstructing large phylogenetic trees. UNM Computer Science Tech-Reports TR-CS-2004-07, University New Mexico, Albuquerque, NM, USA.

Rzhetsky, A. and Nei, M. (1993) Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Mol. Biol. Evol.*, **10**, 1073–1095.

Saitou, N. and Nei, M. (1987) The Neighbor-Joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Sanger, F., Nicklen, S. and Coulsen, A. R. (1977) DNA sequencing with chain–terminating inhibitors. *Proc. Natl. Acad. Sci. USA*, **74**, 5463–5467.

Sankoff, D. (1975) Minimal mutation trees of sequences. *SIAM Journal of Applied Mathematics*, **28**, 35–42.

Schmidt, H. A. (2003) *Phylogenetic Trees from Large Datasets*. Ph.D. thesis, Universität Düsseldorf.

Schmidt, H. A., Petzold, E., Vingron, M. and von Haeseler, A. (2003) Molecular phylogenetics: Parallelized parameter estimation and quartet puzzling. *J. Parallel Distrib. Comput.*, **63**, 719–727.

Schmidt, H. A., Strimmer, K., Vingron, M. and von Haeseler, A. (2002) TREE-PUZZLE: Maximum likelihood phylogenetic analysis using quartets and parallel computing. *Bioinformatics*, **18**, 502–504.

Semple, C. and Steel, M. (2003) *Phylogenetics*. Oxford Univerity Press.

Sneath, P. H. A. and Snokal, R. R. (1973) *Numerical taxonomy*. W. H. Freeman, San Francisco, USA.

Spencer, M., Susko, E. and Roger, A. J. (2005) Likelihood, parsimony, and heterogeneous evolution. *Mol. Biol. Evol.*, **22**, 1161–1164.

Stamatakis, A. and Ludwig, T. (2004) The AxML program family for phylogenetic tree inference. *Concurr. Comput.-Pract. Exp.*, **16**, 975–988.

Stamatakis, A. P. (2004) An efficient program for phylogenetic inference using simulated annealing. In *Online Proceedings of the 4th IEEE International Workshop on High Performance Computational Biology (HICOMB 2005)*, page 8, Denver.

Stamatakis, A. P., Ludwig, T. and Meier, H. (2005) RAxML-III: a fast program for maximum likelihood-based inference of large phylogenetic trees. *Bioinformatics*, **21**, 456–463.

Steel, M. (1994) The maximum likelihood point for a phylogenetic tree is not unique. *Syst. Biol.*, **43**, 560–564.

Strimmer, K. and von Haeseler, A. (1996) Quartet puzzling: A quartet maximum–likelihood method for reconstructing tree topologies. *Mol. Biol. Evol.*, **13**, 964–969.

Strimmer, K. and von Haeseler, A. (2003) Nucleotide substitution models. In Salemi, M. and Vandamme, A.-M. (eds.), *The Phylogenetics Handbook A Practical Approach to DNA and Protein Phylogeny*, pages 72–100, Cambridge University Press, Cambridge.

Sullivan, J., Abdo, Z., Joyce, P. and Swofford, D. L. (2005) Evaluating the performance of a successive-approximations approach to parameter optimization in maximum-likelihood phylogeny estimation. *Mol. Biol. Evol.*, **22**, 1386–1392.

Swofford, D. L. (2002) *PAUP*: Phylogenetic analysis using parsimony (and other methods)*. Sinauer Associates, Sunderland, MA.

Swofford, D. L., Olsen, G. J., Waddell, P. J. and Hillis, D. M. (1996) Phylogeny reconstruction. In Hillis, D. M., Moritz, C. and Mable, B. K. (eds.), *Molecular Systematics*, pages 407–514, Sinauer Associates, Sunderland, Massachusetts, Second edn..

Tamura, K. and Nei, M. (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.

Tateno, Y., Takezaki, N. and Nei, M. (1994) Relative efficiencies of the maximum–likelihood, neighbor–joining, and maximum–parsimony methods when substitution rate varies with site. *Mol. Biol. Evol.*, **11**, 261–277.

Tavaré, S. (1986) Some probabilistic and statistical problems on the analysis of DNA sequences. *Lec. Math. Life Sci.*, **17**, 57–86.

Thompson, J. D., Higgins, D. G. and Gibson, T. J. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*, **22**.

Uzzel, T. and Corbin, K. W. (1971) Fitting discrete probability distributions to evolutionary events. *Science*, **172**, 1089–1096.

Wakeley, J. (1993) Substitution rate variation among sites in hypervariable region 1 of human mitochondrial DNA. *J. Mol. Evol.*, **37**, 613–623.

Waterman, M. and Smith, T. (1978) On the similarity of dendrograms. *Journal of Theoretical Biology*, **73**, 789–800.

Waterman, M. S. (2000) *Introduction to Computational Biology*. Chapman and Hall, London, UK, first crc press edn..

Whelan, S., de Bakker, P. I. W. and Goldman, N. (2003) Pandit: a database of protein and associated nucleotide domains with inferred trees. *Bioinformatics*, **19**, 1556–1563.

Whelan, S. and Goldman, N. (2001) A general empirical model of protein evolution derived from multiple protein families using a maximum likelihood approach. *Mol. Biol. Evol.*, **18**, 691–699.

Willson, S. J. (1999) Building phylogenetic trees from quartets by using local inconsistency measures. *Mol. Biol. Evol.*, **16**, 685–693.

Yang, Z. (1994) Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximative methods. *J. Mol. Evol.*, **39**, 306–314.