universität wien

DISSERTATION / DOCTORAL THESIS

Titel der Dissertation /Title of the Doctoral Thesis

„Computational methods for fast and accurate phylogenetic inference“

verfasst von / submitted by

Dipl.-Ing. Lam-Tung Nguyen BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Doctor of Philosophy (PhD)

Wien, 2016 / Vienna 2016

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 794 685 490
Dissertationsgebiet lt. Studienblatt / field of study as it appears on the student record sheet:	Molekulare Biologie
Betreut von / Supervisor:	Uni.-Prof. Dr. Arndt von Haeseler

Abstract

The theory of evolution, first popularized by Charles Darwin (Darwin, 1859), laid the foundation of modern biological research. Reconstructing a phylogeny (evolutionary tree) from molecular data is one approach for understanding evolutionary relationships. The advent of high throughput and cheap sequencing technologies has led to the explosion of genetic sequences. To keep up with the speed of data generation, tree reconstruction methods need to be constantly improved. This thesis presents fast and accurate methods for inferring maximum-likelihood phylogenies.

First, we describe a fast and effective stochastic search algorithm to find maximum-likelihood phylogenies. Our algorithm, implemented in the phylogenetic software IQ-TREE, employs hill-climbing search, stochastic perturbation and evolution strategy to sample local optima in the tree space. IQ-TREE performs favorably compared to state of the art methods such as RAxML and PhyML. If we allow the same CPU time as RAxML and PhyML, then IQ-TREE found higher likelihoods between 62.2% and 87.1% of the studied alignments, thus efficiently exploring the tree space. If we use the IQ-TREE stopping rule, RAxML and PhyML are faster in 75.7% and 47.1% of the DNA alignments and 42.2% and 100% of the protein alignments, respectively. However, the range of obtaining higher likelihoods with IQ-TREE improves to 73.3–97.1%.

Second, we show that popular phylogenetic inference software cannot reliably estimate parameters of the widely used model of sequence evolution for rate heterogeneity +I+Γ. The inability to infer the true parameters is caused by inaccurate numerical optimization routines implemented in these programs. Hence, we propose an alternative optimization strategy to improve the accuracy of estimates for the +I+Γ model. As more and more complex models of sequence evolution are being developed, our finding emphasizes the equal importance of developing suitable estimation methods.

Third, we present the data-driven heuristic IQ-TREE-SP to shorten the tree search time. IQ-TREE-SP infers stable tree structures from the generated locally optimal trees to constrain the search space. Our computational results show that IQ-TREE-SP is up to 3.9 times faster than IQ-TREE, while at the same time produces better results.

Finally, we present an MPI parallelization of the IQ-TREE search algorithm which exhibits very good scaling performance.

The described methods are implemented in the software IQ-TREE, available at: http://www.cibiv.at/software/iqtree.

Parts of this thesis have been published:

1. L.-T. Nguyen, H.A. Schmidt, A. von Haeseler, and B.Q. Minh (2014) *IQ-TREE: A fast and effective stochastic algorithm for estimating maximum likelihood phylogenies. Molecular Biology and Evolution.*

In preparation:

2 L.-T. Nguyen, A. von Haeseler, and B.Q. Minh *Complex models of sequence evolution require accurate estimator as exemplified with the invariable site plus Gamma model.*

Zusammenfassung

Die von Charles Darwin aufgestellte Evolutionstheorie (Darwin, 1859) hat den Grundstein zu moderner biologischen Forschung gelegt. Die Rekonstruktion der Phylogenie (Stammbaum) aus molekularen Daten spielt die Hauptrolle bei der Aufklärung evolutionären Beziehungen. Mithilfe von günstigen Sequenzierungstechnologien werden jedes Jahr Unmengen von biologischen Daten produziert. Um mit der Geschwindigkeit der Datengenerierung Schritt zu halten, müssen Baum-Rekonstruktionsmethoden regelmäßig angepasst und verbessert werden. Diese Arbeit befasst sich mit der Entwicklung von effizienten und präzisen Algorithmen für die Rekonstruierung von Phylogenie mittels der Maximum-Likelihood-Methode.

Als Erstes stellen wir einen schnellen und effektiven stochastischen Algorithmus zur Schätzung der Maximum-Likelihood-Stammbaum vor. Wir zeigen, dass eine Kombination von Bergsteigeransatz, Randomisierung und umfassende Probe der Suchraum zu bemerkenswerter Verbesserung der Baumsuche führt. Für die meisten getesteten Datensätzen fand unsere Software IQ-TREE bessere Bäume als die zwei weit bekannte Software RAxML und PhyML.

Zweitens zeigen wir, dass die häufig verwendeten phylogenetische Software Parameter von der populären evolutionären Modell +I+Γ nicht verlässlich schätzen können. Wir haben festgestellt, dass die unzureichende Genauigkeit der implementierten Optimierungsroutinen die Ursache des Problems ist. Dabei schlagen wir eine alternative Optimierungsstrategie vor, welche die Genauigkeit der Schätzungen erheblich verbessert. Unsere Ermittlung unterstreicht die Wichtigkeit der Entwicklung geeigneter Schätzverfahren, insbesondere wenn immer mehr komplexe evolutionäre Modelle zum Einsatz kommen.

Drittens erweitern wir den IQ-TREE Suchalgorithmus um eine effiziente Beschleunigungsheuristik. Wir verwenden hier einen datengetriebenen Ansatz um die stabilen Baumstrukturen zu erkennen. Dadurch können wir den Suchraum einschränken und die Suchzeit bis zu 3,9 Male reduzieren.

Zum Schluss präsentieren wir eine effiziente MPI-Parallelisierung des IQ-TREE Suchalgorithmus.

Alle Methoden sind in der Software IQ-TREE implementiert. IQ-TREE ist auf der folgenden Website erhältlich:
http://www.cibiv.at/software/iqtree.

Teile dieser Arbeit wurden in dem folgenden Artikel publiziert:

1. L.-T. Nguyen, H.A. Schmidt, A. von Haeseler, and B.Q. Minh (2014) *IQ-TREE: A fast and effective stochastic algorithm for estimating maximum likelihood phylogenies. Mol. Biol. Evol.*

In Vorbereitung:

2 L.-T. Nguyen, A. von Haeseler, and B.Q. Minh *Complex models of sequence evolution require accurate estimator as exemplified with the invariable site plus Gamma model.*

Acknowledgements

First and foremost, I want to express my sincere gratitude to my supervisor Arndt von Haeseler for providing me the best opportunities to learn and grow in a fruitful scientific environment.

My sincere thanks goes to my mentor Bui Quang Minh, who has provided me countless assistance throughout the years.

I would like to thank Heiko Schmidt for his sincere help and insightful scientific discussions during the course of my PhD.

Big thanks to my colleagues Olga Chernomor, Milica Krunic, Celine Prakash and Konstantina Kyriakouli for all the joy and cheerful moments we shared. Thanks to all other lab members for contributing to my great experience at CIBIV.

Last but not least, I want to thank my wife Thuy-Duong Nguyen, my parents and parents in law for supporting and encouraging me in the pursuit of my PhD.

Contents

Chapter 1

Introduction

1.1 Motivation

In his seminal work "On the Origin of Species" (Darwin, 1859), Charles Darwin proposed that life on Earth has evolved from a common ancestor. This lays the foundation for the studies of evolution. Understanding evolution has since then become an essential part of biology. Studying evolution does not only improve the classification of species but also answers important questions in medical research such as elucidating pathogen transmission (Grenfell *et al.*, 2004) and unraveling cancer development (Salipante and Horwitz, 2006). To emphasize the importance of evolution, the prominent evolutionary biologist Theodosius Dobzhansky once wrote "Nothing in biology makes sense except in the light of evolution" (Dobzhansky, 1973).

The inheritance of traits and accumulation of mutations in an organism leads to diverging evolutionary paths. A phylogeny or phylogenetic tree represents the evolutionary relationships among different organisms. Figure 1.1 shows an example of a partial phylogenetic tree of mammals. The phylogenetic tree of a group of species is often unknown or controversial in many cases. Reconstructing phylogeny requires collecting data about the characteristics of each species, based on which computational methods are often used to resolve the tree.

Advances in sequencing technologies have led to the emergence of molecular evolution research. Phylogenetic analysis are now mostly performed on molecular data. However, genetic data are being produced at an unprecedented speed. To handle

such massive influx of data, computational methods for inferring phylogeny need to be constantly improved.

1.2 Organization of this thesis

This chapter presents the basic concepts of phylogeny reconstruction. We first introduce the biological and computational aspects of molecular phylogenetics. Subsequently, we explain in detail the fundamentals of maximum-likelihood methods for phylogenetic inference. In chapter 2, we introduce IQ-TREE, an efficient stochastic algorithm for searching the maximum-likelihood tree. Chapter 3 discusses a problem current phylogenetic software encounter when dealing with complex model of sequence evolution, namely the inaccurate estimation of model parameters. Here, we also provide a method to resolve the problem. Chapter 4 describes a data-driven heuristic to constrain the tree space, thus speeding up the tree search. Chapter 5 presents a coarse-grained parallelization of the IQ-TREE algorithm using the Message Passing Interface. In chapter 6, we summarize the main contributions of the thesis.

1.3 Molecular genetic data

Evolution is driven by changes in genetic information which is stored in the *Genome* and encoded in DNA (deoxyribonucleic acid), or for some viruses in RNA (ribonucleic acid). DNA is the building block of genes, which are then transcribed and translated into proteins (Crick *et al.*, 1970).

Molecular genetic data are often represented by sequences of nucleotide or amino acid. There are five different type of nucleotides: Cytosince, Guanine, Adenine, Thymine and Uracil. Thymine is only presented in DNA whereas Uracil is found in RNA. Amino acids are the building block of proteins. There are 20 different amino acids that are encoded in the genome (Table 1.1).

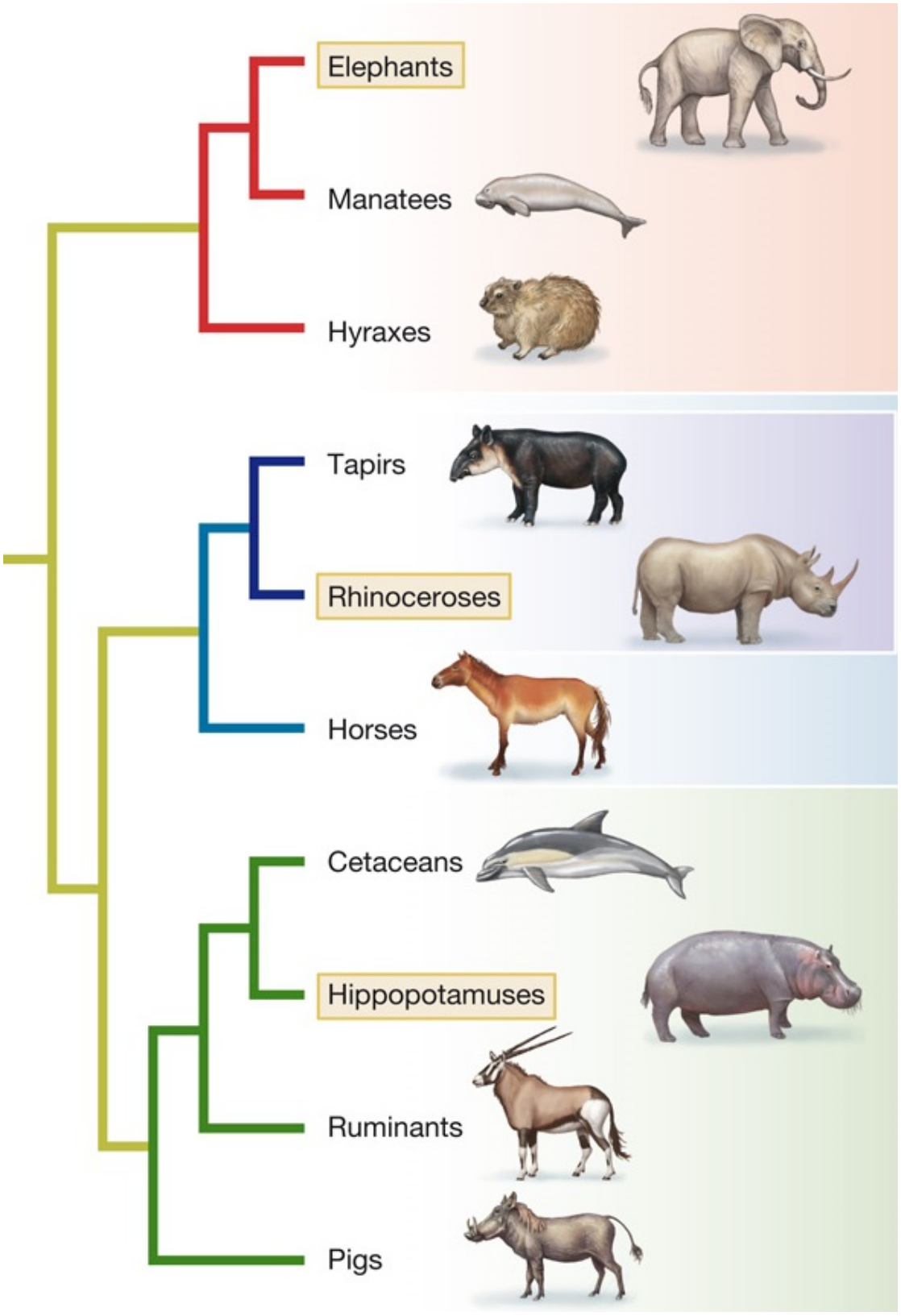

FIGURE 1.1: A partial phylogenetic tree of mammals (source: www.biology-forum.com).

Name	3-letter abbr.	1-letter abbr.	Name	3-letter abbr.	1-letter abbr.
Alanine	Ala	A	Methionine	Met	M
Cysteine	Cys	C	Asparagine	Asn	N
Aspartic acid	Asp	D	Proline	Pro	P
Glutamic acid	Glu	E	Glutamine	Gln	Q
Phenylalanine	Phe	F	Arginine	Arg	R
Glycine	Gly	G	Serine	Ser	S
Histidine	His	H	Threonine	Thr	T
Isoleucine	Ile	I	Valine	Val	V
Lysine	Lys	K	Tryptophan	Trp	W
Leucine	Leu	L	Tyrosine	Tyr	Y

TABLE 1.1: Twenty amino-acids (Vandamme, 2003).

1.4 Multiple sequence alignment

Genetic sequences are called homologous if they have evolved from a common ancestral sequence. Given a set of homologous sequences, one of the most common tasks is to identify regions of similarity by aligning the sequences.

A point mutation represents change in a single character of a sequence. There are three types of point mutation:

- Substitution: A character is replaced by another nucleotide.
- Insertion: A extra nucleotide is added into the sequence.
- Deletion: A nucleotide is removed from the sequence.

In the process of aligning sequences, gaps are inserted in between so that homologous nucleotides are aligned in the same column. A gap represents an *indel* event, meaning that either a nucleotide was deleted from the sequence or there are insertions at the place of the gap on other sequences. Figure 1.2 illustrates a pairwise alignment of two homologous DNA sequences. Based on predefined scoring schemes, the sequences are aligned so that the final alignment produces the highest score. A scoring scheme could be as simple as follows: +1 for a match, -1 for a mismatch and -2 for an indels. The pairwise alignment in Figure 1.2 has 12 matches, 1 mismatch and 2 indels. Thus, this alignment has a total score of 7. There are efficient dynamic programming algorithms to align two sequences

(Needleman and Wunsch, 1970; Smith and Waterman, 1981). These algorithms can find the optimal alignment with time complexity $O(nm)$, where n and m are the lengths of the two sequences.

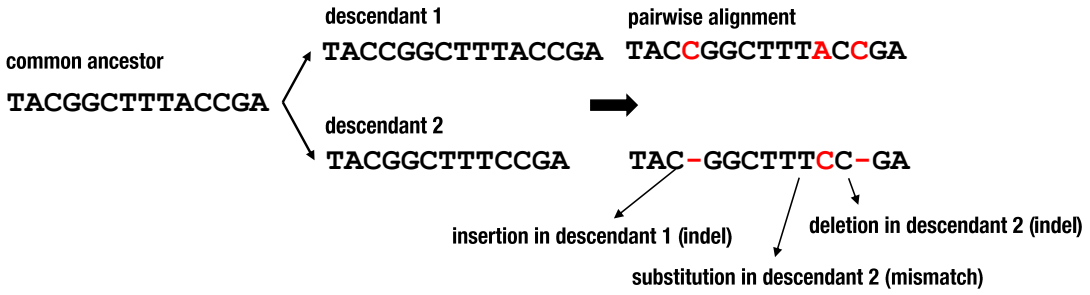

FIGURE 1.2: Pairwise alignment of two homologous DNA sequences. In descendant 1, there is one insertion. In descendant 2, there are one substitution and one deletion. The gap characters ('-') represent either insertion or deletion event (indel).

Multiple sequence alignment (MSA) is an alignment of three or more sequences, which is the prerequisite of most phylogenetic analysis. To align the sequences, one can extend the dynamic programming technique. However, this approach is computationally intractable because the search space does not only depend on the sequence length but it also grows exponentially with the number of sequences. In fact, finding the optimal multiple sequence alignment is NP-Complete (Wang and Jiang, 1994). Thus, developing efficient heuristic methods for multiple sequence alignment is also very challenging. Notable examples of MSA method are: CLUSTAL W (Thompson *et al.*, 1994), T-COFFEE (Notredame *et al.*, 2000), MAFFT (Katoh *et al.*, 2002), MUSCLE (Edgar, 2004) and PRANK (Löytynoja and Goldman, 2008). MSAs are typically used as input for phylogenetic inference.

1.5 Phylogenetic tree

A phylogenetic tree or phylogeny is often depicted as a binary tree, in which each internal node has exact two children. Each leaf of a phylogeny represents a sequence (taxon) in the alignment, whereas the internal nodes correspond to extinct ancestors. The branching pattern of a phylogenetic tree defines its topology. The branch lengths represent the expected number of substitutions per site.

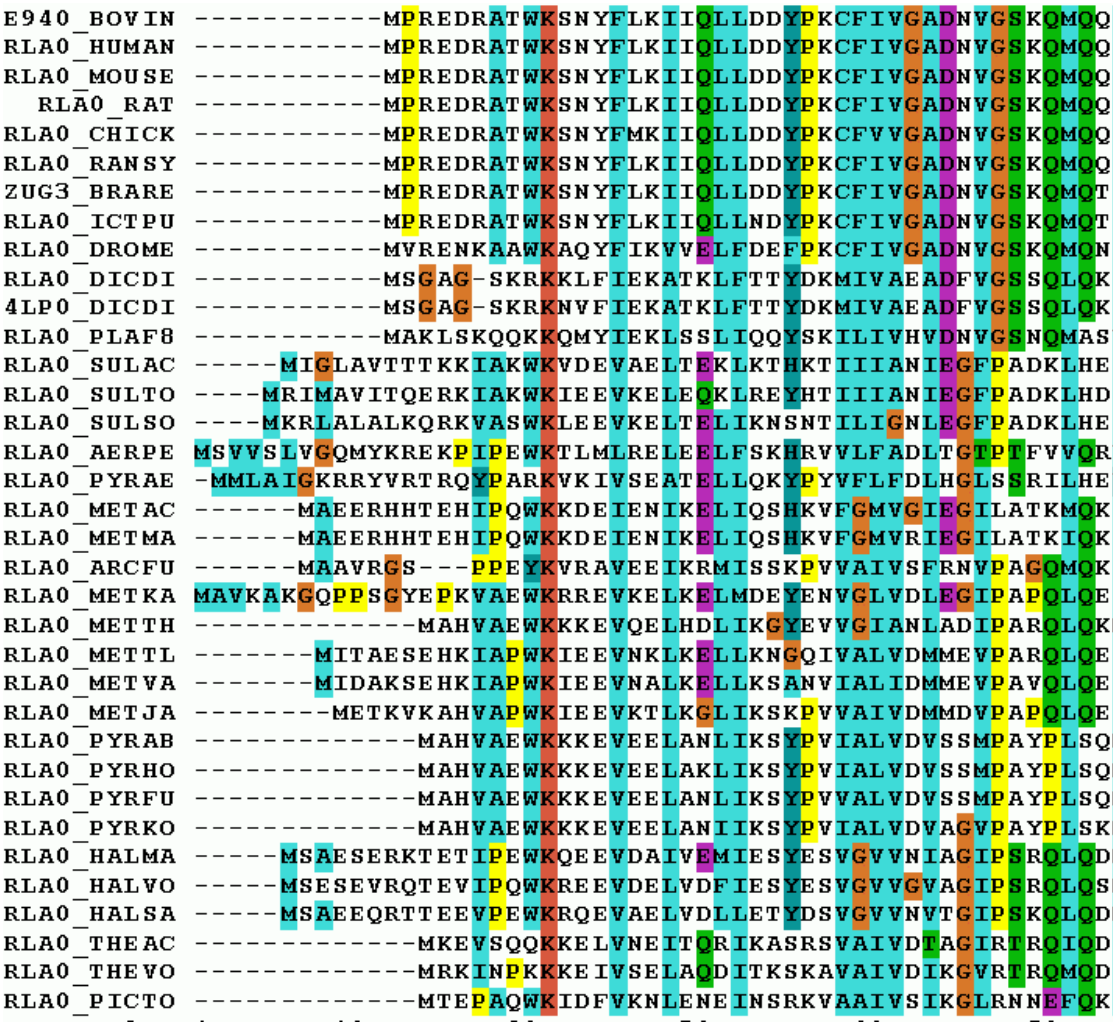

FIGURE 1.3: A multiple sequence alignment of the acidic ribosomal protein from multiple organisms (source: commons.wikimedia.org). The columns are commonly referred to as sites.

A phylogenetic tree can be rooted (Figure 1.4a) or unrooted (Figure 1.4b). In a rooted phylogenetic tree, the direction of evolution is given by the declaration of a root. Whereas, an unrooted phylogenetic tree does not specify the direction of evolution. Most phylogenetic software work on unrooted trees.

Each internal branch divides the taxon set into two non-empty, disjoint subsets. This bipartition is called a *split*. For example, the tree topology in Figure 1.4 is composed of the two splits: $AB|CDE$ and $ABC|DE$.

For n taxa, there are $\prod_{i=3}^{n}(2i-5)$ distinct representations of unrooted binary trees (Felsenstein, 1978b). As the number of trees grow exponentially with the number of taxa, enumerating all possible trees for a large number of taxa is intractable.

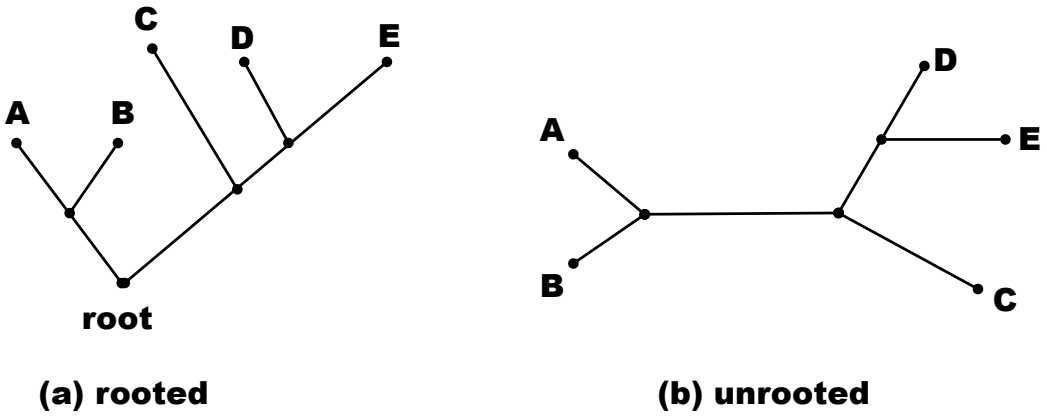

FIGURE 1.4: A rooted (a) and unrooted (b) phylogenetic tree. Both trees have the same topology. In the rooted tree, the evolution starts from the root, whereas the direction of evolution in the unrooted tree is unknown.

1.6 Tree reconstruction methods

There are several methods to reconstruct phylogenetic tree from a multiple sequence alignment. Here, we briefly introduce the primary tree reconstruction methods. For in-depth introduction of each method refer to Felsenstein (2004) and Yang (2006).

- **Maximum parsimony method**: For a given tree topology, one computes the minimum number of character changes required to explain the observed variations in the alignment. This number is referred to as parsimony score of the tree, which can be computed efficiently using the algorithm proposed by Fitch (1971). In maximum parsimony, the goal is to search for trees with the lowest parsimony score.

- **Distance method**: The pairwise distances among the sequences are first computed from the MSA. The distance between two sequences can be as simple as the number of character differences per site (*observed distance*). However, this measure of distance does not account for multiple substitutions per site. Therefore, pairwise distances are typically inferred by maximum-likelihood methods, given an evolutionary model. The pairwise distances are stored in a distance matrix. The MSA is discarded and the distance matrix is now used as input for the tree reconstruction step. The objective

of distance methods is to find a tree, whose branching patterns and lengths best reflect the estimated distances. Popular distance methods are: least squares (L. L. Cavalli-Sforza, 1967), minimum evolution (Rzhetsky and Nei, 1992) and neighbor joining (Saitou and Nei, 1987).

- **Maximum-likelihood (ML) method**: Maximum likelihood is a popular methodology in statistics to estimate parameters of statistical models from data (Harris and Stöcker, 1998). In the context of phylogenetic inference, the parameters are the tree topology, branch lengths, and parameters of the evolutionary model. The multiple sequence alignment is the data. The ML method searches for trees with the highest likelihood given the data. The likelihood of a phylogenetic tree is computed as the probability of the data having evolved according to the tree and an evolutionary model.

- **Bayesian method**: Bayesian inference of phylogeny is closely related to the maximum-likelihood method in the sense that it also relies on the computation of tree likelihood. However, Bayesian methods try to compute the (posterior) probability of the trees and not just their likelihoods. Bayesian methods typically sample the distribution of trees with high probability using Markov chain Monte Carclo methods.

Maximum parsimony, maximum likelihood and distance method use optimality criteria to search for the best trees, whereas Bayesian method perform sampling on the tree posterior distribution. The advantage of distance and maximum parsimony over maximum likelihood and Bayesian method is their computational efficiency in evaluating trees. However, they suffer several shortcomings. Distance method might not work well on highly divergent sequences because estimates of large distances are often subject to high sampling error (Yang and Rannala, 2012). Maximum parsimony method might be affected by the *long-branch attraction* problem (Felsenstein, 1978a) because it does not account for multiple substitutions at the same site. Maximum likelihood and Bayesian methods typically do not have the deficiencies observed in maximum parsimony and distance method. They also work well if the observed sequences evolve under the assumed model (Hall, 2005). Despite their demanding computational requirements, maximum-likelihood and Bayesian method have gained massive popularity in the recent years, thanks to the advances in computational algorithms and computing hardware.

1.7 Phylogenetic inference by maximum likelihood

Inferring phylogeny by maximum likelihood requires the assumption of an evolutionary model, based on which the likelihood of a phylogenetic tree is computed. Hereafter, we discuss different types of evolutionary models, how to efficiently compute the tree likelihood and heuristic search techniques.

1.7.1 Substitution models

A substitution model defines the rates of change of each character in a genetic sequence. Moreover, the following assumptions are often made (Felsenstein, 2004; Strimmer and von Haeseler, 2009):

- *Markov property*: The rate of change of character i to j does not depend on the prior state of i.
- *Time-homogeneous*: The substitution rates stay constant over time.
- *Time-continuous*: Substitutions can happen at any point in time.
- *Time-reversible*: The rate of change does not depend on the direction of evolution.
- *Stationary*: The relative frequency of each character is at equilibrium.

Substitution models are presented in the form of a so-called instantaneous substitution rate matrix Q. Each entry Q_{ij} of the matrix specifies the number of substitutions taking place between sequence character i and j per evolutionary time unit. For example, the instantaneous substitution rates of the four nucleotides A, C, G and T are described by the following Q matrix of the General Time-Reversible Model (GTR) (Tavaré, 1986):

$$Q = \begin{pmatrix} -\sum_{i \neq A} Q_{Ai} & a\pi_C & b\pi_G & c\pi_T \\ a\pi_A & -\sum_{i \neq C} Q_{Ci} & d\pi_G & e\pi_T \\ b\pi_A & d\pi_C & -\sum_{i \neq G} Q_{Gi} & f\pi_T \\ c\pi_A & e\pi_C & f\pi_G & -\sum_{i \neq T} Q_{Ti}) \end{pmatrix}$$

a, b, c, d, e, f are the rate parameters. π_A, π_C, π_G and π_T are the equilibrium frequencies of the nucleotides A, C, G and T, respectively. The entries Q_{ii} on the diagonal are specified so that each row sums up to zero.

The GTR model represents the most general form of DNA substitution model under the aforementioned assumptions. Simpler substitution models are created by enforcing constraints on the GTR model. For instance, the JC69 model (Jukes and Cantor, 1969) assumes that all equilibrium frequencies are equal ($\pi_A = \pi_C = \pi_G = \pi_T = 0.25$) and all substitution type occur at the same rate ($a = b = c = d = e = f = 1$). The K2P model proposed by Kimura (1980) differentiates between two substitution types: transition and transversion (Figure 1.5). Felsenstein (1981) introduced the F81 model with different nucleotide frequencies. Hasegawa *et al.* (1985) combined the characteristics of K2P and the F81 model into the HKY model.

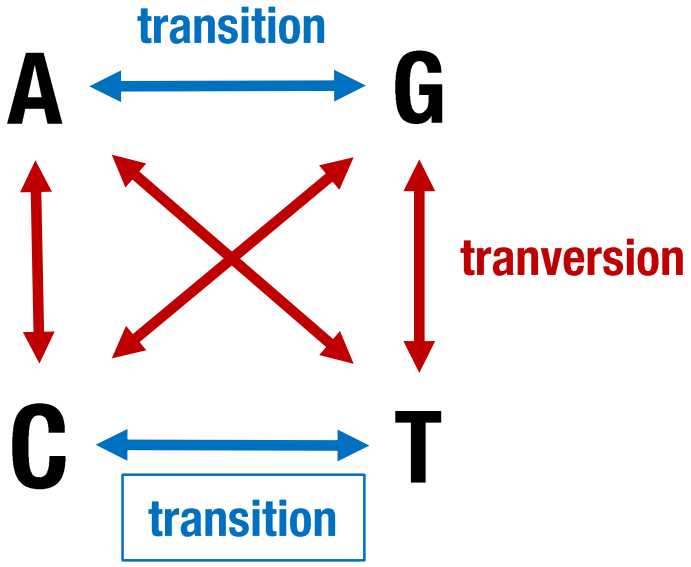

FIGURE 1.5: Different substitution types for nucleotides.

Substitution models for protein sequences are described by 20x20 rate matrices. While in most DNA substitution models the rate parameters are estimated for each individual dataset, using the same approach for protein substitution model is computationally demanding because of the vast number of parameters involved.

For that reason, rate parameters of protein substitution models are often estimated empirically in advance using large number of sequence data.

Based on the instantaneous rate matrix Q, the probability of change from one character to another for a time period t can be calculated. These probabilities are stored in the *transition matrix* $P(t)$ which can be obtained by exponentiating the Q matrix:

$$P(t) = e^{Qt} \tag{1.1}$$

Numerical methods for calculating the transition matrix $P(t)$ are described in von Haeseler (1999) and Yang (2006).

1.7.2 Rate heterogeneity models

It has been observed that substitution rates for sequences evolving under functional or structural constraints might vary among sites (reviewed in Swofford *et al.* (1996) and Yang (1996)). This phenomenon is called *rate heterogeneity*. Substitution models described previously work under the assumption that substitution rates among sites are the same. This simplified assumption may cause inaccurate estimation of phylogenetic distances and phylogenies (Palumbi, 1989; Jin and Nei, 1990; Li *et al.*, 1990; Tateno *et al.*, 1994). Different models were suggested to accommodate rate heterogeneity (Fitch and Margoliash, 1967; Uzzell and Corbin, 1971; Olsen, 1987; Yang, 1993; Meyer and von Haeseler, 2003).

The gamma distribution is most commonly used to model rate heterogeneity (Uzzell and Corbin, 1971; Yang, 1993). The description of the resulting model often has the suffix +Γ, e.g. JC+Γ or GTR+Γ. Here, each site is assigned a rate parameter, whose value is assumed to follow a gamma distribution with mean α/β and variance α/β^2 where α and β are the shape and scale parameter of the gamma distribution, respectively $(\alpha, \beta > 0)$. To reduce the number of parameters, β is set equal to α. Thus, the gamma distribution has mean 1 and variance $1/\alpha$. Figure 1.6 shows the shapes of the gamma distributions for different α's. Depending on the value of α, the gamma distribution can be either bell-shaped $(\alpha > 1)$ or L-shaped $(\alpha < 1)$. This allows great flexibility in modeling rate variation.

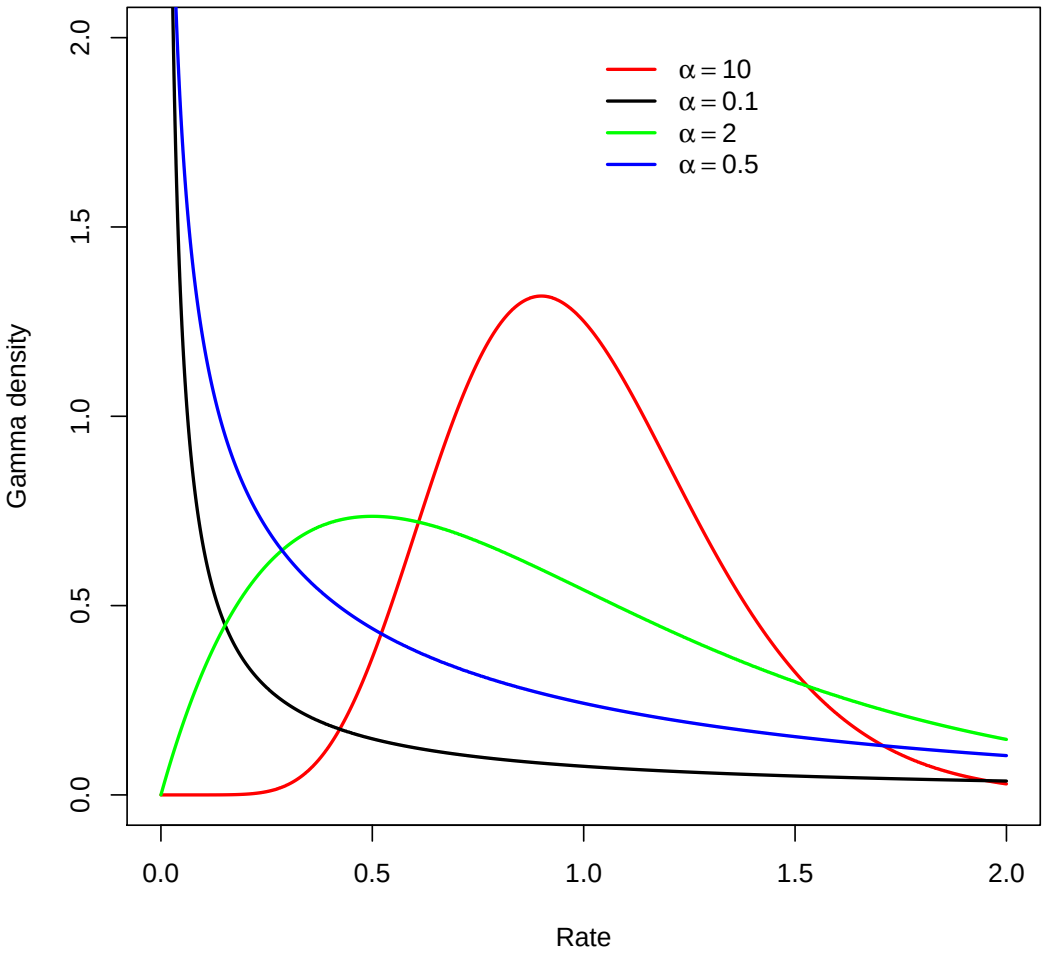

FIGURE 1.6: Probability density functions of gamma distributions having different shape parameters.

The shape parameter α of the gamma distribution is typically estimated from the alignment using maximum likelihood. However, the likelihood calculation under a continuous gamma distribution is impractical for trees with more than 10 taxa (Yang, 2006). As a result, the discrete gamma model is mostly used to approximate the continuous gamma distribution (Yang, 1994). The discrete gamma distribution is comprised of equal-percentile and equal-probability rate categories. The mean or median of each category is used to represent all rates in that category (Yang, 1994). Figure 1.7 shows an example of a discrete gamma distribution with four rate categories and $\alpha = 0.5$. Although discrete gamma models with higher number of rate categories provides better approximation of the

continuous gamma model (Mayrose *et al.*, 2005), four rate categories are typically used to reduce computational requirement.

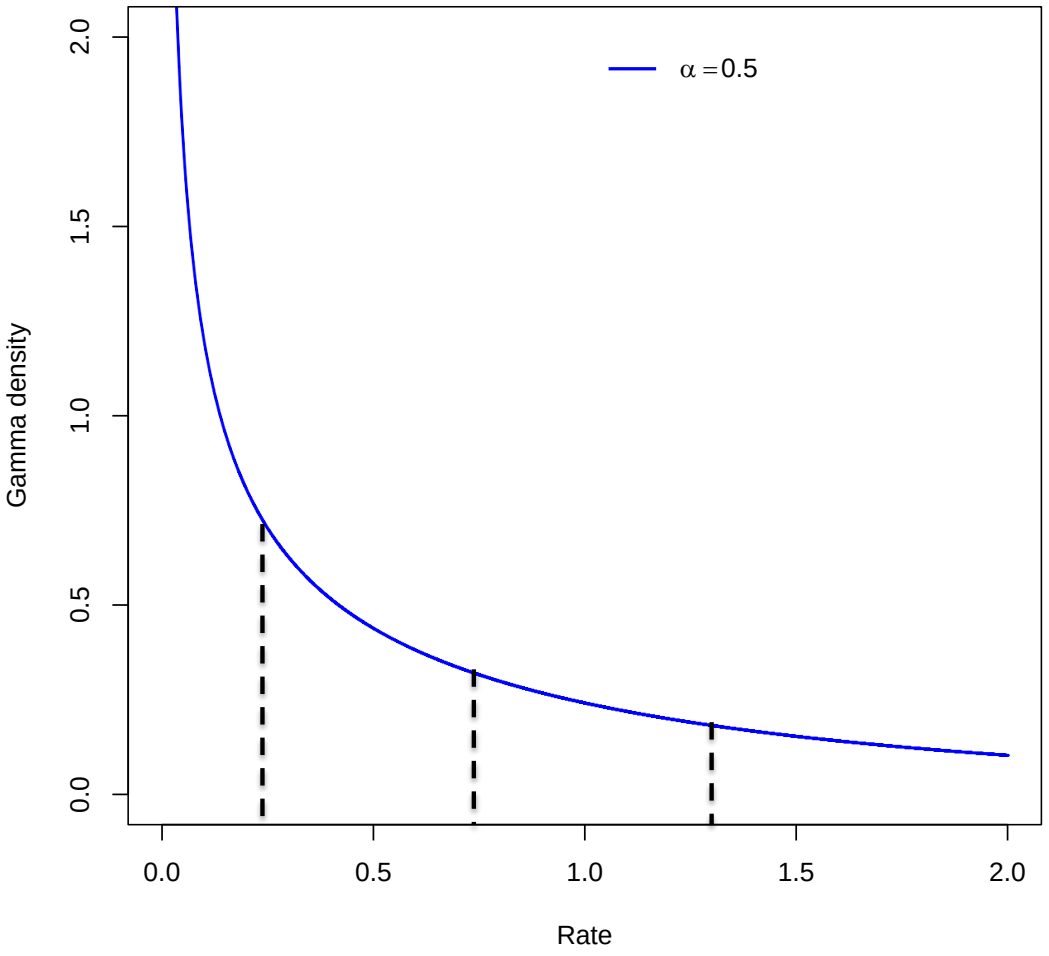

FIGURE 1.7: A discrete gamma model of rate heterogeneity to approximate a continuous gamma distribution with shape parameter $\alpha = 0.5$. The vertical dashed lines are the 25th, 50th and 75th percentiles of the distribution dividing the density into four equal-probability categories. Rates in each category are equals and computed as the mean of the category.

Every biological sequence might have conservative regions, which cannot change due to functional constraints. Thus, having an evolutionary model that includes invariability for some portion of the sequence is sensible and biologically appealing (Fitch and Margoliash, 1967; Hasegawa *et al.*, 1985). This model is called *invariable-site* model, which is often denoted by the suffix +I. Gu *et al.* (1995); Waddell and Steel (1997) suggested combining the gamma model and the

invariable-site model into the mixture +I+Γ model. Most phylogenetic software use the discrete gamma model to implement the +I+ discrete Γ model. Compared to the discrete gamma model, the +I+ discrete Γ model has one additional rate category with zero rate.

1.7.3 Likelihood calculation

Given a multiple sequence alignment D and a hypothesis H consisting of a phylogenetic tree and an evolutionary model, the likelihood of H is:

$$L(H) = P(D|H),$$

where $P(D|H)$ is the probability of D having evolved according to H. The likelihood is calculated under the following assumptions (Felsenstein, 1981):

- Evolution of sites along the alignment is independent.
- The sites are identically distributed.
- All sites evolve according to the same tree.

Because sites evolve independently and identically, the likelihood of the hypothesis H is the product of the likelihoods of each site D^i:

$$L(H) = \prod_i L^i = \prod_i P(D^i|H)$$

We illustrate the calculation of the site likelihood using an example of a rooted four-taxon tree (Figure 1.8). The likelihood of the tree is the sum of the probabilities of all possible nucleotides appearing at the internal nodes. Let $\mathbb{A}$ be the set of all possible characters (for our example in Figure 1.8 we have $\mathbb{A} = \{A, C, G, T\}$). The likelihood of site D^i is:

$$\begin{aligned} P(D^i|H) &= \sum_x \sum_y \sum_z P(A, G, C, C, x, y, z|H) \\ &= \sum_x \sum_y \sum_z P(x) P_{xy}(t_1) P_{xz}(t_2) P_{yA}(t_3) P_{yG}(t_4) P_{zC}(t_5) P_{zC}(t_6) \end{aligned} \tag{1.2}$$

where x, y, z represent the possible nucleotides at node V_0, V_1, V_2, respectively ($x, y, z \in \mathbb{A}$). $P_{ij}(t)$ is the probability of character i mutating to j after time

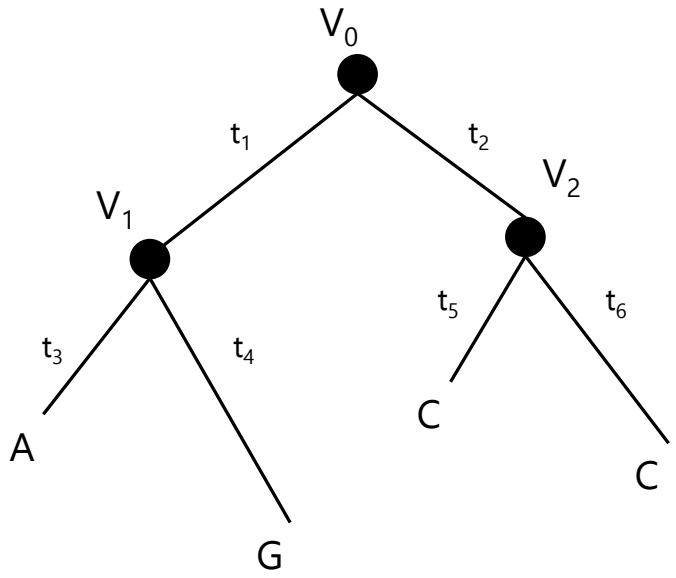

FIGURE 1.8: A rooted phylogenetic tree of four sequences, whose characters at a alignment site i are A, G, C and C.

t. $P_{ij}(t)$ is the entry of the transition matrix computed according to Equation 1.1. $P(x)$ is the probability of each nucleotide, whose value is assumed to be the equilibrium frequency π_x of the nucleotide.

Equation 1.2 requires the summation of a huge number of terms. For a four-taxon tree, one has to sum over $4^3 = 64$ terms. This number grows exponentially as the number of taxa increases, rendering the calculation intractable. Felsenstein (1981) introduced a *dynamic programming* approach coined *pruning algorithm* that makes the likelihood computation feasible. By placing the summation sign in Equation 1.2 directly before the related term we can rewrite the equation as follows:

$$P(D^i|H) = \sum_x P(x) \left(\sum_y P_{xy}(t_1) P_{yA}(t_3) P_{yG}(t_4) \right) \times \left(\sum_z P_{xz}(t_2) P_{zC}(t_5) P_{zC}(t_6) \right) \tag{1.3}$$

This reformulation suggests that we can compute the tree likelihood in a bottom up fashion, starting from the leaves. For each subtree with root V, the pruning algorithm computes a so-called partial likelihood $L_V^i(s)$, which is the probability of the observed characters in the subtree's leaves, conditional on V having character

s. For example, the term $P_{yA}(t_3)P_{yG}(t_4)$ is the partial likelihood of V_1 having character y.

For each subtree we can compute the partial likelihood of every possible character at the subtree's root using the following recursive formula:

$$L^i_{V_0}(x) = \left(\sum_y P_{xy}(t_1)L^i_{V_1}(y)\right)\left(\sum_z P_{xz}(t_2)L^i_{V_2}(z)\right) \tag{1.4}$$

where $L^i_V(x)$ is the partial likelihood of node V having character x; V_0 is the root of the subtree, and V_1 and V_2 are its two immediate descendants.

These partial likelihoods need to be computed only once and can be reused for the calculation of other partial likelihoods. This is the dynamic programming approach used in the pruning algorithm which reduces the time complexity from exponential to linear.

If V is a leaf then:

$$L^i_V(z) = \begin{cases} 1, & \text{if } z = D^i_V \\ 0, & \text{otherwise} \end{cases} \tag{1.5}$$

where D^i_V is the observed character at site D^i in the sequence represented by V.

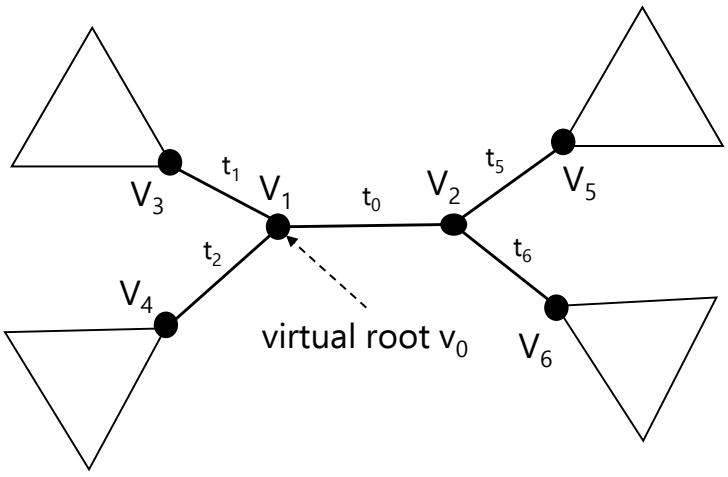

FIGURE 1.9: Example of an unrooted tree.

The site likelihood of the tree in Figure 1.8 is then computed as

$$L^i = \sum_y \sum_z \sum_x P(x) P_{xy}(t_1) P_{xz}(t_2) L^i_{V_1}(y) L^i_{V_2}(z) \tag{1.6}$$

In the previous example, we used a rooted tree to compute the likelihood. Specifying the root requires knowledge about the direction of evolution. Fortunately, the assumption about time-reversibility in the substitution process makes the tree likelihood calculation independent of the placement of the root. This is called the *pulley principle* (see Felsenstein (1981) for more details). Thus, for an unrooted tree we can place a virtual root at any internal node. Figure 1.9 shows an unrooted tree with two internal nodes V_1 and V_2. Here, we can assume a virtual root V_0 at V_1 and compute the site likelihood L^i as follows

$$L^i = \sum_y \sum_x P(x) P_{xy}(t_0) L^i_{V_1}(x) L^i_{V_2}(y) \tag{1.7}$$

If a discrete model of site rate heterogeneity is used (see section 1.7.2), one needs to compute the partial likelihoods for each rate category:

$$L^i_{V_0}(x, j) = \left(\sum_y P_{xy}(t_1 r_j) L^i_{V_1}(y, j) \right) \left(\sum_z P_{xz}(t_2 r_j) L^i_{V_2}(z, j) \right) \tag{1.8}$$

where r_j is the rate of the j-th category. Similar to Equation 1.7, the site likelihood for each category is computed as

$$L^i(j) = \sum_y \sum_x P(x) P_{xy}(t_0 r_j) L^i_{V_1}(x, j) L^i_{V_2}(y, j) \tag{1.9}$$

The final site likelihood is then the weighted sum of the site likelihoods for all rate categories:

$$L^i = p_{inv} L^i(r_0) + (1 - p_{inv}) \frac{1}{n} \sum_{j=1}^{n} L^i(j) \tag{1.10}$$

where p_{inv} is the proportion of invariable sites ($0 \leq p_{inv} < 1$) with rate $r_0 = 0$ and n is the number of rate categories for variable sites.

To avoid numerical underflow when taking the product of all site likelihoods, one computes the natural logarithm of each site likelihood $\mathcal{L}^i = \log(L^i)$. The final tree

log-likelihood is then the sum of all site log-likelihoods:

$$\mathcal{L} = \sum_i \mathcal{L}^i \tag{1.11}$$

1.7.4 Parameter optimization

In the previous section, we described the likelihood calculation on a fixed tree topology assuming that all other parameters are known. However, the goal of the maximum-likelihood method is to find a parameter set that maximizes the likelihood. To this end, all parameters have to be optimized. Our parameter set consists of the parameters of the evolutionary model and the branch lengths of the tree. We treat the tree topology as a special parameter, whose optimization will be discussed in section 1.7.5.

Because the tree likelihood can be computed for any given parameter set, one can use many general purpose numerical optimization methods to find the optimal parameter values. Although a multidimensional optimization method can be used to simultaneously optimize all parameters, this approach is slow and hard to implement (Swofford *et al.*, 1996). Therefore, parameters are often estimated separately. Yang (2000) proposed a heuristic for fast estimation of substitution model parameters and branch lengths on a fixed tree topology. This approach consists of two steps. First, substitution model parameters are simultaneously optimized by the BFGS method (Gill *et al.*, 1981), while all branch lengths are fixed. Second, branch lengths are optimized one after another in several tree traversals using the Newton-Raphson method (Press, 2007). The optimization of model parameters and branch lengths is performed alternatively until the likelihood improvement is smaller than a predefined threshold.

The aforementioned optimization heuristic is implemented in IQ-TREE (Nguyen *et al.*, 2015). Nevertheless, there has not been a consensus on the best practice for estimating model parameters and branch lengths. Each phylogenetic software provides its own heuristic for this procedure. While most optimization heuristics seem to work well for the most basic models, it is unclear how well the estimation of parameter-rich models works in modern phylogenetic software. In chapter 3 we will discuss the reliability of parameter estimation for the parameter-rich model +I+G.

1.7.5 Heuristic tree searches

Finding the tree topology with highest likelihood is the most challenging problem in phylogenetic inference by maximum likelihood. Not only one has to estimate the parameters and branch lengths for each possible topology, but the number of topologies to examine is also astronomically large (there are $\prod_{i=3}^{n}(2i-5)$ possible topologies for a set of n taxa). Thus, the computation required for inferring the maximum-likelihood tree is extremely demanding.

Because finding the optimal tree by exhaustive search is intractable for a large number of taxa, heuristic methods are mostly used (Strimmer and von Haeseler, 1996; Guindon and Gascuel, 2003; Vinh and von Haeseler, 2004; Zwickl, 2006; Stamatakis, 2006). Most of them employ tree rearrangement operations to search through tree space. In maximum-likelihood phylogenetic inference, the tree rearrangement operations Nearest Neighbor Interchange (NNI) and Subtree Pruning and Regrafting (SPR) are widely used. In NNI, each internal branch defines a quartet of four subtrees and a subtree is swapped with one of the other two subtrees on the other side of the branch (Figure 1.10a). In SPR, a subtree is pruned and regrafted into a different branch of the tree (Figure 1.10b).

Most phylogenetic software implement hill-climbing searches, using either NNI or SPR moves. The NNI and SPR neighborhood of an unrooted tree with n taxa consist of $2(n-3)$ and $2(n-3)(2n-7)$ trees, respectively (Caceres *et al.*, 2013). An SPR hill-climbing algorithm often has higher chance of finding better trees than its NNI counterpart because it can search in a larger neighborhood. Consequently, SPR hill-climbing algorithms require substantially more computing time. However, hill-climbing algorithms are prone to get stuck in local optima. In chapter 2 we will discuss about different tree search techniques and introduce our efficient stochastic algorithm IQ-TREE which bypasses the typical drawbacks of hill-climbing algorithms.

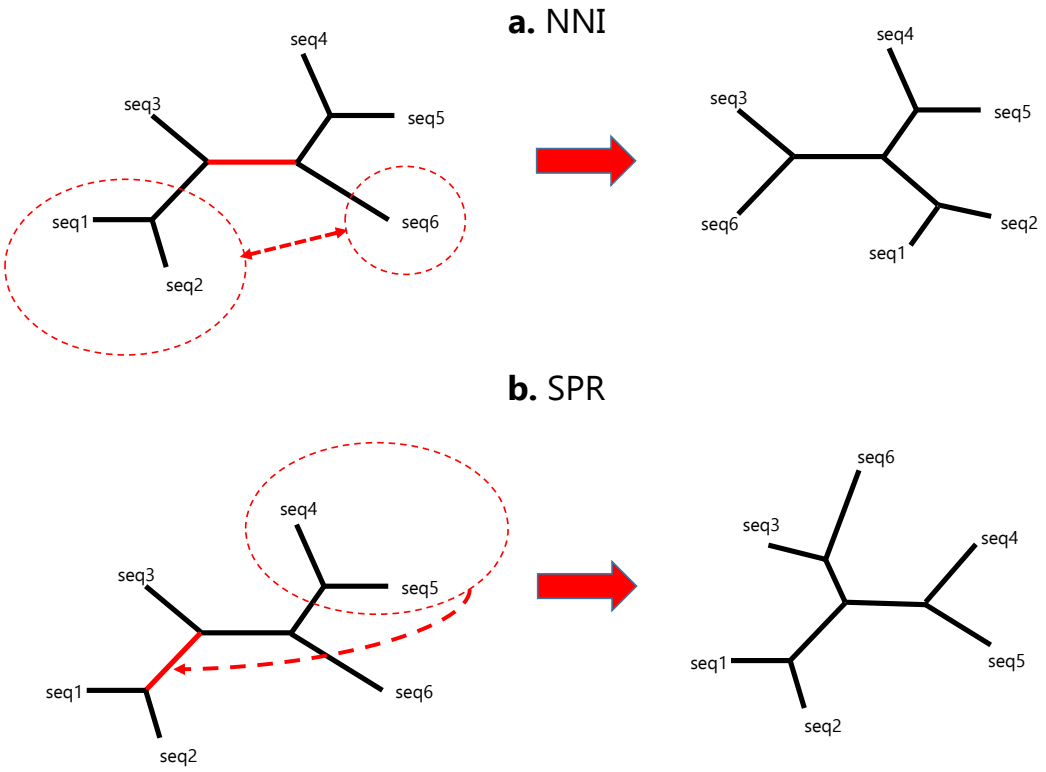

FIGURE 1.10: Tree rearrangement operations: Nearest Neighbor Interchange (NNI) and Subtree Pruning and Regrafting (SPR).

Chapter 2

IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies

Large phylogenetic data sets require fast tree inference methods, especially for maximum-likelihood (ML) phylogenies. Fast programs exist, but due to inherent heuristics to find optimal trees, it is not clear whether the best tree is found. Thus, there is need for additional approaches that employ different search strategies to find ML trees and that are at the same time as fast as currently available ML programs. In this chapter, we show that a combination of hill-climbing approaches and a stochastic perturbation method increase the effectiveness of the tree search.

2.1 Introduction

Phylogenetic inference by maximum likelihood (ML) is widely used in molecular systematics (Felsenstein, 1981, 2004). It involves the estimation of substitution model parameters, branch lengths and tree topology. These parameters are usually estimated one after another with the tree topology being the main parameter of interest. While efficient numerical methods for estimating substitution model parameters and branch lengths on a fixed tree exist (Yang, 2000), finding the

optimal tree topology is an NP-hard combinator optimization problem (Chor and Tuller, 2005). Therefore, one has to rely on search heuristics to find the "best" tree.

ML tree searches apply inter alia local tree rearrangements such as nearest neighbor interchange (NNI), subtree pruning and regrafting (SPR), or tree bisection and reconnection (TBR) to improve the current tree (Guindon *et al.*, 2010; Swofford, 2003; Stamatakis, 2006). Here, only modifications that increase the tree likelihood ("uphill" moves) are allowed. Such approaches are prone to be stuck in local optima (e.g., (Swofford *et al.*, 1996)). The problem becomes more severe if the local tree rearrangement method can only generate a small number of trees in neighborhood of the current tree. As a result, SPR algorithms often find trees with higher likelihoods than those that are based on NNI (Morrison, 2007; Whelan and Goldman, 2001; Money and Whelan, 2012). TBR is not often used due to its high computational demand.

Stochastic algorithms were developed to overcome the problem of local optima encountered by hill-climbing algorithms. Current ML implementations of stochastic algorithms allow "downhill" moves (Salter and Pearl, 2001; Vos, 2003; Vinh and von Haeseler, 2004) or maintain a population of candidate trees (Lewis, 1998; Zwickl, 2006; Helaers and Milinkovitch, 2010) to avoid local optima. However, in terms of both likelihood maximization and computation time such implementations have been found not to perform as well as SPR-based hill-climbing algorithms (Stamatakis, 2006; Morrison, 2007). The large variety of techniques makes it difficult to combine them into effective and efficient stochastic algorithms. While the possibilities to enhance a hill-climbing algorithm are limited, the potential to improve the effectiveness and efficiency of stochastic algorithms is not yet fully explored.

Here we present a fast and effective stochastic algorithm for finding ML trees. The core idea is to perform an efficient sampling of local optima in the tree space. Here, the best local optimum found represents the reported ML tree. To this end, we combine elements of hill-climbing algorithms, random perturbation of current best trees, and a broad sampling of initial starting trees. Comparative analyses for many large DNA and amino acid (AA) multiple sequence alignments retrieved from TreeBASE (Sanderson *et al.*, 1994) showed that our new search strategy often achieves higher likelihoods compared with RAxML (Stamatakis, 2006) and PhyML (Guindon *et al.*, 2010).

2.2 Methods

In the following, we describe the ingredients of the fast tree reconstruction method that are implemented in IQ-TREE. We first describe our fast hill-climbing NNI algorithm that is used throughout the tree search. Subsequently, we will explain the initial tree generation and the stochastic NNI process. For likelihood and parsimony computations, we used the phylogenetic likelihood library (Flouri *et al.*, 2015).

2.2.1 Hill-Climbing NNI

For the determination of locally optimal trees, we implemented a fast hill-climbing NNI search. Our approach is based on the work of Guindon and Gascuel (2003) where they applied several NNIs simultaneously. The hill-climbing NNI is a central element of the search strategy (Figure 2.1 box c).

NNI is a local tree rearrangement operation that swaps two subtrees across an internal branch. Each inner branch defines two distinct NNIs. Thus, for an unrooted bifurcating tree with n taxa, there are $2(n-3)$ NNI-trees in the NNI-neighborhood of that tree.

For a given tree (current tree), we first compute the approximate likelihoods of each NNI-tree by optimizing the respective inner branch and the four adjacent branches. We only consider NNIs that increase the tree likelihood compared with the current tree. We then create a list of so-called non-conflicting NNIs. Two NNIs are considered conflicting if they operate on the same inner branch or adjacent branches. The list is initialized with the best NNI. We then add the next best NNI to the list if it does not conflict with any existing NNI in the list, otherwise we discard it. We repeat this procedure until all NNIs have been processed.

Afterwards, we simultaneously apply all NNIs in the list to the current tree and compute the likelihood of the resulting tree by doing one tree traversal of ML branch length optimization. If the likelihood of the resulting tree is worse than the likelihood of the best NNI-tree, we discard all topological modifications except that of the best NNI. Thus, if the list is not empty, a new tree with higher likelihood will always be found. This tree will replace the current tree, completing one round of hill-climbing optimization. If the list is empty, we stop the hill-climbing search

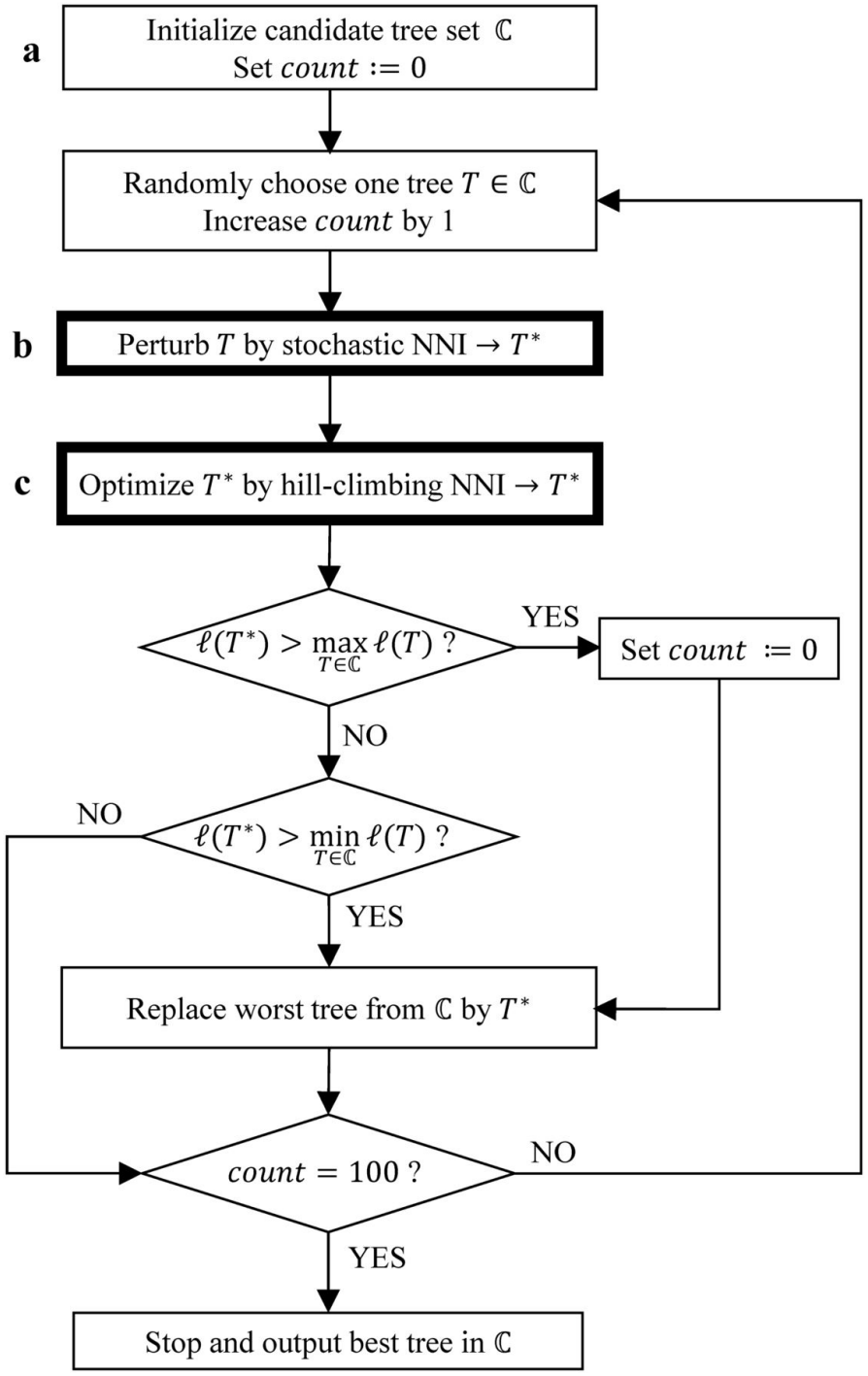

FIGURE 2.1: Flowchart for the stochastic search algorithm. The variable count counts the number of random perturbations (box b and box c) as a new best tree was found

because no further improvement can be made. Otherwise, we continue with the next round of hill-climbing optimization.

To reduce computing time in the next optimization rounds, we apply a heuristic that works as follows. We tag the inner branches on the new current tree on which NNIs were applied in the previous hill-climbing round. Instead of evaluating the full NNI-neighborhood we only evaluate likelihoods of NNI-trees induced by inner branches that are at most two branches away from the tagged branches. The remaining steps, starting with the initialization of the non-conflicting NNIs, are performed as previously described.

2.2.2 Initial Tree Generation

Tree search heuristics typically start with a quickly built initial tree that is subsequently improved. For example, PhyML starts with a BIONJ tree (Gascuel, 1997) whereas RAxML starts with a MP tree constructed by stepwise addition (Farris, 1970; Fitch, 1971) and further improved by SPR tree rearrangements. To get a representative sample of plausible initial trees, we generate 100 parsimony trees using the same strategy as RAxML. From the 100 trees, we collect all unique topologies and compute their approximate likelihoods by doing one tree traversal of ML branch length optimization. From the ranked list of ML values, we select the top 20 parsimony trees and perform hill-climbing NNI on each tree to obtain the locally optimal ML trees. We then retain the top five topologies with highest likelihood in the so-called candidate tree set $\mathbb{C}$ for further optimization.

2.2.3 Optimization by stochastic and hill-climbing NNI

Trees in the candidate set $\mathbb{C}$ are randomly perturbed so that new locally optimal trees can be found. To this end, we introduce a so-called stochastic NNI step (Figure 2.1 box b). Here, we perform $(n-3)/2$ random NNIs on a tree T randomly drawn from $\mathbb{C}$, where $n-3$ is the number of inner branches. Then, we apply hill-climbing NNI to the perturbed tree to obtain a new locally optimal tree T^* (Figure 2.1 box c).

If T^* has a higher likelihood than the best tree in $\mathbb{C}$, we replace that tree by T^*. Moreover, the stochastic NNI successfully found a better tree, we then set

the counter (*count*) of the number of perturbations to zero. If T^*'s likelihood is higher than the likelihood for the worst tree in $\mathbb{C}$, then that tree is replaced by T^*. Finally, $\mathbb{C}$ does not change if the likelihood of T^* is smaller than the smallest likelihood for the trees in $\mathbb{C}$. In the last two cases, the tree with the highest likelihood did not change and *count* is increased by one.

The tree search stops, if the current best tree has not changed after $count = 100$ random perturbations. The flowchart of our stochastic tree search is summarized in Figure 2.1.

2.3 Results

2.3.1 Benchmark Setup

Here, we compared the performance of our approach (implemented in IQ-TREE 1.0) with with the default tree searches implemented in PhyML 3.1 (Guindon *et al.*, 2010) and RAxML 7.3.5 (Stamatakis, 2006). To that end, we downloaded multiple sequence alignments from TreeBASE ((Sanderson *et al.*, 1994); accessed December 1, 2012) fulfilling the following criteria. First, the number of sequences must be between 200 and 800 for DNA and between 50 and 600 for AA alignments. Second, the alignment length must be at least four (or two) times the number of sequences in DNA (or AA) alignments. Third, the proportion of gaps/unknown characters must be less or equal than 70%. Identical sequences were discarded from the alignments keeping only one. We obtained 70 DNA and 45 AA alignments (see supplementary tables S1 and S2, Appendix A). The DNA alignment lengths range from 976 to 61,199 sites. The AA alignment lengths were between 126 and 22,426 sites.

For all programs, we used the substitution model GTR (Lanave *et al.*, 1984) and WAG (Whelan and Goldman, 2001) for DNA and AA alignments, respectively. Rate heterogeneity followed the discrete Γ model (Yang, 1994) with four rate categories, where relative rates are computed as the mean of the portion of the Γ distribution falling in the respective category. To avoid numerical discrepancies between different implementations of the likelihood function, we used PhyML to recompute the log-likelihoods of the final trees based on parameters produced by each program. We note that, for 92% of the trees the differences in log-likelihoods

computed by IQ-TREE and PhyML are smaller than 0.01 and the maximal difference is only 0.05 (supplementary Figure A.1, Appendix A). Thus, the numerical discrepancies in the log-likelihood calculation between the programs are negligible. All analyses were performed on the Vienna Scientific Cluster (VSC-2, vsc.ac.at).

2.3.2 Comparison with Equal Running Times

Because RAxML and PhyML are considered as the two high-performance ML tree-inference programs, we first benchmarked IQ-TREE by restricting the running time of IQ-TREE to that required by each RAxML and PhyML run. This is done to study how efficiently IQ-TREE uses its search time compared with the other programs. For each alignment, we ran RAxML ten times and PhyML once (because the default tree search in PhyML is deterministic). Subsequently, we ran IQ-TREE ten times for each alignment with the restricted CPU time. Then, we compared for each alignment the average log-likelihood of trees produced by IQ-TREE with those by the other two programs.

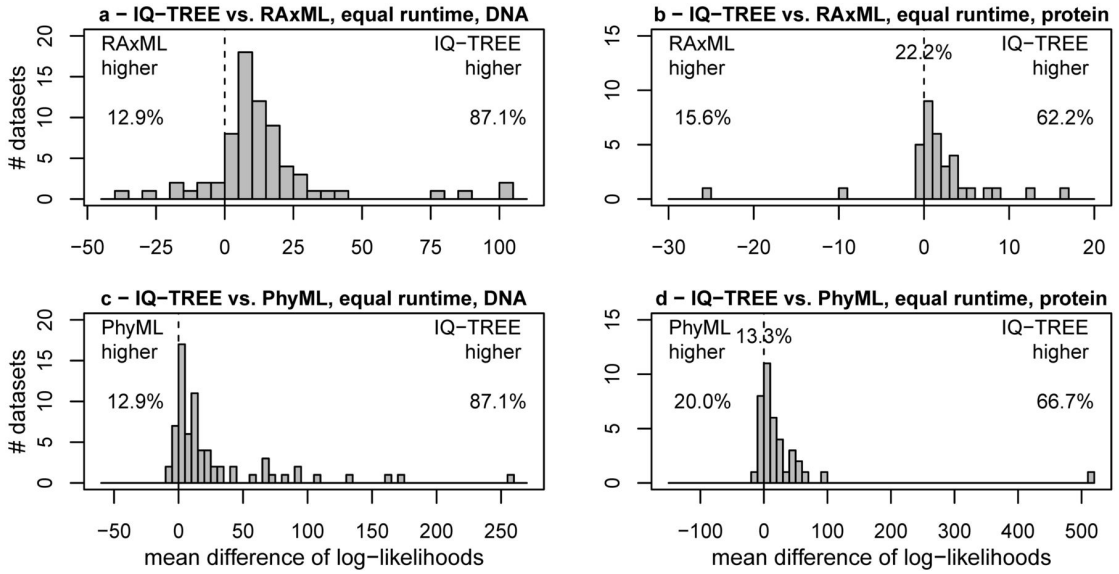

FIGURE 2.2: Performance of IQ-TREE for fixed CPU times: (a, b) Display frequencies of log-likelihood differences for IQ-TREE minus RAxML for 70 DNA (a) and 45 AA (b) alignments. (c) and (d) show the same if IQ-TREE is compared with PhyML. IQ-TREE's CPU times were limited to those required by RAxML and PhyML, respectively. The percentages on the dashed line in (b) and (d) represent the fraction of alignments where log-likelihood differences are smaller than 0.01.

Figure 2.2 (and supplementary Figure A.2, Appendix A) displays the pairwise log-likelihood difference distributions for IQ-TREE versus RAxML (Figure 2.2 a and b) and PhyML (Figure 2.2 c and d). Trees inferred with IQ-TREE for DNA alignments had in 87.1% of the instances a higher likelihood than RAxML-trees or PhyML-trees (Figure 2.2 a and c). Although these percentages are identical, the alignments for which IQ-TREE found better trees if compared with RAxML or PhyML are not the same (see supplementary Figure A.2, Appendix A). For 12.9% of the alignments, RAxML or PhyML found better trees.

For the AA alignments, IQ-TREE found higher likelihoods in 62.2% if compared with RAxML (Figure 2.2 b) and in 66.7% if compared with PhyML (Figure 2.2 d). In 22.2% of the alignments, RAxML and IQ-TREE found trees with negligible log-likelihood differences (smaller than 0.01). This number is 13.3% when comparing PhyML with IQ-TREE. In only 15.6% and 20% of the AA alignments, RAxML and PhyML performed better (with respect to tree log-likelihoods) than IQ-TREE, respectively.

We note that the distributions in Figure 2.2 a, c, and d are skewed to the right. Thus, our tree search strategy sometimes produced substantially better likelihoods.

In summary, based on the analysis of a large collection of alignments, we demonstrate that IQ-TREE shows higher likelihoods in approximately three-quarters of the analyzed data. The improvement is almost the same compared with RAxML or PhyML. Because we fixed IQ-TREE's running time to the time the other programs needed, we conclude that the employed search strategy explores tree-space more efficiently.

2.3.3 Comparison with Different Running Time

We now discuss the performance of IQ-TREE if the CPU time is not determined by RAxML or PhyML but rather by the default stopping rule. Thus, we compared the differences in CPU time and the differences in log-likelihoods (Figure 2.3 and supplementary Figure A.3, Appendix A). Again, the analyses were based on average of ten independent IQ-TREE and RAxML runs. Figure 2.3 is organized like Figure 2.2 (RAxML vs. IQ-TREE results in the first row, PhyML vs. IQ-TREE results in the second row, DNA alignments left column, and AA alignments right column).

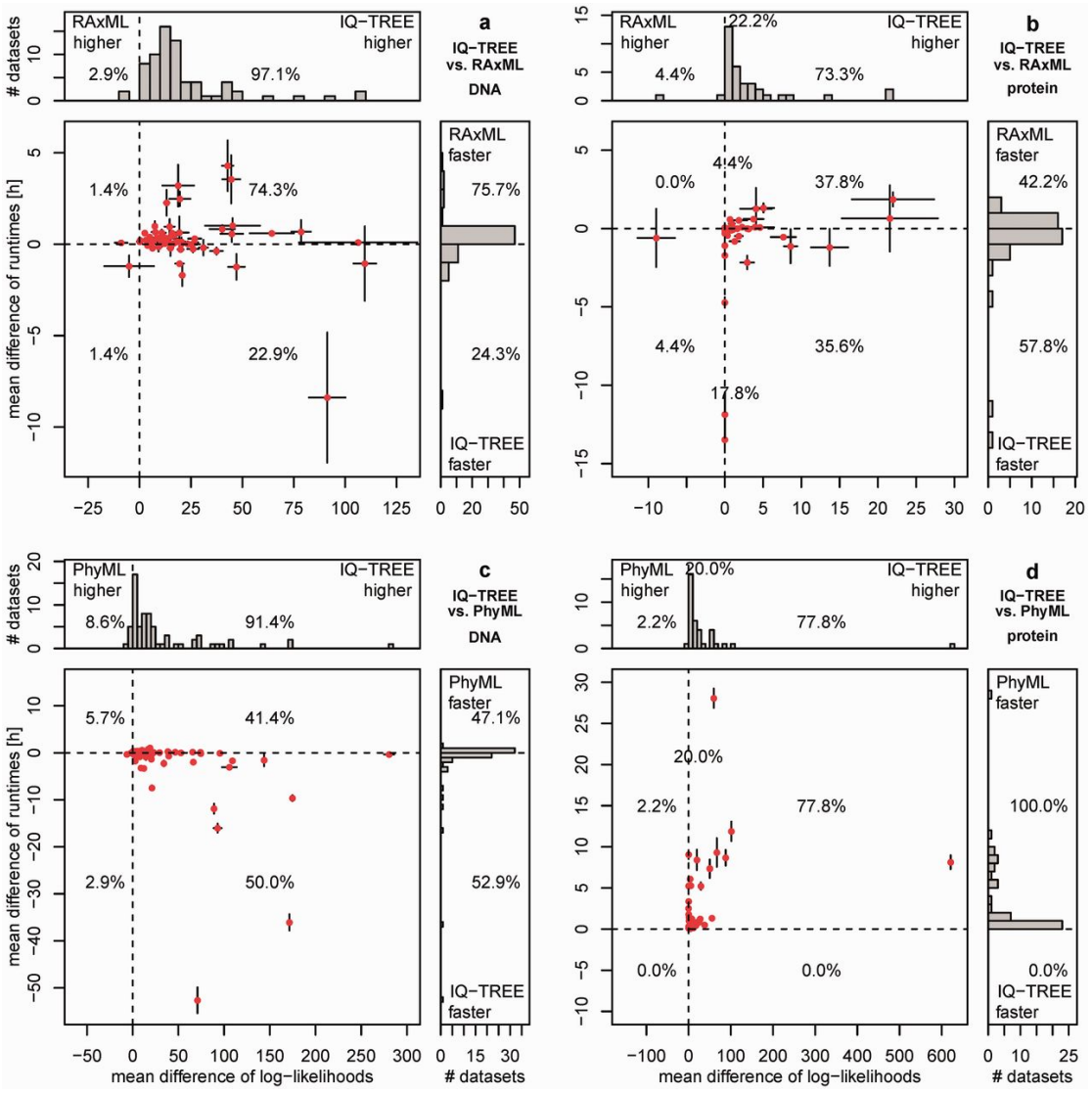

FIGURE 2.3: Performance of IQ-TREE for variable CPU times: The upper plots (a, b) show the performance of IQ-TREE against RAxML using the 70 DNA (a) and 45 AA (b) alignments. The lower plots (c, d) show the same against PhyML. Each dot in the main diagrams represents for one alignment the mean differences of the CPU times (y axis) and of the mean differences of log-likelihoods (x axis) of the reconstructed trees by the programs compared. The whiskers at each point show the standard errors of the differences. The histograms at the top and the side present the marginal frequencies. Dots to the right of the vertical dashed line represent alignments where IQ-TREE found a higher likelihood. If a dot is below the horizontal dashed line, the reconstruction by IQ-TREE was faster. Percentages in the quadrants of histograms denote the fraction of alignments in that region. Percentages on the dashed line reflect the number of alignments where log-likelihood differences are smaller than 0.01 (see [b] and [d]).

By allowing variable CPU time, the number of the alignments for which IQ-TREE found higher-likelihood trees than RAxML or PhyML increases. For 97.1% of the DNA alignments, the likelihood is improved compared with RAxML (Figure 2.3 a). The maximal log-likelihood difference is 109.5 (TreeBASE ID: M7964). For two DNA alignments, IQ-TREE obtained trees with lower likelihoods than RAxML with log-likelihood differences up to -8.9 (M2534).

This success in finding higher likelihoods comes at a cost; IQ-TREE required longer CPU times than RAxML for 75.7% of the DNA alignments. However, the situation is complicated; the differences in average CPU times are highly variable, and for some alignments, one program is much faster than the other. For example, for the alignment M7024 IQ-TREE needed 4.2 h more than RAxML to finish, whereas RAxML required 8.3 h more to produce an ML tree for M14582. To finish all ten repetitions for the 70 DNA alignments, IQ-TREE needed 2,020 CPU hours ($\sim$ 87 CPU days), whereas RAxML needed 1,870 CPU hours ($\sim$ 78 CPU days). This is an average CPU time difference of less than 13 min per run.

Figure 2.3b displays the results for the 45 AA alignments. For ten AA alignments (22.2%, cf. supplementary Figure A.3b, Appendix A), IQ-TREE and RAxML inferred trees with likelihood differences smaller than 0.01 for all ten runs. For 73.3% of the AA alignments, IQ-TREE obtained higher likelihoods than RAxML with a maximal log-likelihood difference of 21.9 (M11012). And for only 4.4% of the AA alignments, the results were in favor of RAxML, with a maximum log-likelihood difference of -8.9 (M3114). In terms of computing time, IQ-TREE obtained the result faster than RAxML in 57.8% of the AA alignments, whereas RAxML was faster in 42.2%. In total, IQ-TREE needed 2,042 CPU hours to complete all 450 runs, whereas RAxML required 2,380 CPU hours. This is an excess of 16.6% compared with the CPU time of IQ-TREE. Thus, on average RAxML needed 45 CPU minutes more per run. The run time ratios between IQ-TREE and RAxML range from 0.6 to 3.2 for DNA and from 0.5 to 1.9 for protein alignments.

Finally, Figure 2.3c and d display the results of IQ-TREE and PhyML for the DNA and AA alignments, respectively. IQ-TREE obtained higher likelihoods than PhyML for 91.4% of the DNA and 77.8% of the AA alignments. PhyML obtained higher likelihoods in 8.6% and 2.2% for DNA and AA, respectively. Notably, the maximal log-likelihood differences in favor of IQ-TREE were 280.5 (M4794) and

621.1 (M8630) for DNA and AA, respectively. The maximal differences in favor of PhyML were -6.3 (M9143) and -0.27 (M8175) for DNA and AA, respectively.

With respect to computing time, PhyML was faster in 47.1% of the DNA and for all AA alignments, whereas IQ-TREE was faster in 52.9% of the DNA alignments. PhyML spent 357 and 61 CPU hours for all DNA and AA alignments, respectively. Whereas IQ-TREE needed, on average, 202 and 204 hours. However, in the shorter run time PhyML produced lower likelihoods for 77.8% of the AA alignments. The run time ratios between IQ-TREE and PhyML range from 0.3 to 2.5 (DNA) and from 2 to 7.5 (protein).

In addition, we ran PhyML ten times per alignment using a random starting tree and SPR search strategy. Supplementary Figure A.4 in Appendix A shows the results. In terms of computing time, PhyML ran slower than IQ-TREE for 98.6% DNA alignments but still faster than IQ-TREE for all AA alignments. With respect to log-likelihoods, IQ-TREE produced higher likelihoods than PhyML for 88.6% DNA and 93.3% AA alignments (an increase by 2.8% for PhyML on DNA, but a decrease by 15.5% on AA). Hence, IQ-TREE performed better than PhyML under both the default and random starting tree options.

2.4 Discussion

We have combined well-known phylogenetic and combinatorial optimization techniques into a fast and effective tree search algorithm. The success of IQ-TREE results from two factors. First, the new tree search strategy helps to escape local optima efficiently and, thus, leads to trees with high likelihood. Second, the use of the phylogenetic likelihood library (Flouri *et al.*, 2015) reduces the computing time of the likelihood. Given the same amount of CPU time, the efficient IQ-TREE implementation of hill-climbing and stochastic NNI operations (see Materials and Methods) computed trees with higher likelihood than RAxML or PhyML in the majority of cases (up to 87.1% of the benchmark data). This improvement is further boosted if the internal stopping rule was used (up to 97.1%). The success of IQ-TREE in finding trees with higher likelihoods is somehow at odds with the discussion in the literature about the inferior effectiveness of NNI compared with SPR (Hordijk and Gascuel, 2005; Guindon *et al.*, 2010; Whelan and Money, 2010). One explanation for the very good performance of IQ-TREE is the introduction

of the stochastic NNI. This random perturbation of locally optimal trees helps to escape local optima. The perturbed trees are then optimized by hill-climbing NNI, thus allowing for the possibility of finding new and higher local optima. The combination of random perturbation with hill-climbing search was also employed in (Vos, 2003) and (Vinh and von Haeseler, 2004) for ML tree searches. In fact, this simple technique was first proposed in the computer science literature and is not commonly refer to as *iterated local search* (Lourenco *et al.*, 2003).

In addition, we employed evolution strategies (Rechenberg, 1973) to allow for a broader exploration of the tree space. To this end, we maintain a small population (candidate tree set) of locally optimal trees initially generated from a large number of maximum parsimony (MP) trees. Throughout the ML tree search, we continuously update the candidate tree set with better trees. Using a pool of candidate trees allows the search to escape local optima more effectively than having a single candidate tree.

We also observed that the improvement in log-likelihood differences is positively skewed for all comparative analyses with the exception of the time constrained IQ-TREE versus RAxML for AA data (Figure 2.2b). Thus, not only the average log-likelihood improved with IQ-TREE but also for some alignments, the improvement is substantial. It would be interesting to find out the characteristics of such alignments to further improve ML tree reconstruction methods. Moreover, by using many large alignments we show that the very good performance of our search strategy is not limited to a few alignments. Based on our benchmark results, we are confident that IQ-TREE will generally work very well.

We would like to point out that it is not enough to run phylogenetic programs with a stochastic component only once. RAxML and IQ-TREE showed some variation in the log-likelihoods, if they were run several times (here ten times) on the same alignment (Figure 2.3 and supplementary Figure A.3, Appendix A). This observation implies that both programs still finish sometimes in local optima and one should rerun the programs as many times as possible. In addition, we also offer the possibility to run IQ-TREE longer by adjusting the corresponding parameter of the stopping rule or by applying the statistical stopping rule suggested by (Vinh and von Haeseler, 2004). Compared with other programs, our results show that with the standard setting of the stopping rule IQ-TREE can produce very good results, with a moderate increase in running time.

To further facilitate large phylogenetic analyses, we also consider future development of IQ-TREE for distributed computing platforms. The highly independent components of our stochastic search algorithm would allow us to implement an efficient parallelization strategy (cf. (Minh *et al.*, 2005) with near-optimal speedup. Thus, the running time of very large phylogenetic analyses would then be greatly reduced.

In conclusion, IQ-TREE is a time and search efficient ML-tree reconstruction program. It complements the collection of available ML-programs and shows a better performance with respect to the ML search than RAxML or PhyML. However, as IQ-TREE is not always better than the other programs, we recommend using all three programs.

Chapter 3

Complex models of sequence evolution require accurate estimators as exemplified with the invariable site plus Gamma model

This chapter examines the unreliable parameter estimates observed in popular phylogenetic inference programs for the invariable site plus Gamma model. We show that the inability to infer the correct parameters is caused by insufficiently effective numerical optimization routines. To overcome this problem, we propose a simple optimization strategy to improve accuracy for maximum-likelihood methods and we recommend exercising care when implementing estimation routines for complex evolutionary models.

3.1 Introduction

In model based phylogenetic analysis, the invariable site plus Γ model (Yang, 1994; Gu *et al.*, 1995), thereafter referred to as I+Γ, is widely used to model rate heterogeneity among sites, because it often fits the data better than the Γ model or the invariable-sites model alone (Sullivan and Swofford, 1997). Thus, the I+Γ model is frequently selected by MODELTEST (Posada and Crandall, 1998). The I+Γ model has two parameters: the proportion of invariable sites p_{inv}

($0 \leq p_{inv} < 1$) and the shape parameter α of the Γ distribution. A small α (< 1) indicates strong rate heterogeneity, whereas a large α (> 1) corresponds to weak rate heterogeneity. Under certain conditions p_{inv} and α can compete with each other for the same phylogenetic signals. For example, $\alpha \leq 1$ already accounts for sites with low rates; that interferes with p_{inv} and causes a correlation between the parameters making reliable estimation of those parameters difficult (Sullivan *et al.*, 1999; Mayrose *et al.*, 2005; Yang, 2014). Consequently, some phylogenetic practitioners discourage the use of the I+Γ model (Jia *et al.*, 2014; Stamatakis, 2015; Yang, 2014). This recommendation is at odds with the recent result that the GTR+I+ continuous Γ model is identifiable for an unrooted phylogenetic tree with three or more distinct interspecies distances (Rogers, 2001; Allman and Rhodes, 2008; Chai and Housworth, 2011).

Since the I+ continuous Γ model is identifiable, reliable parameter estimation for this model should be possible for sufficiently long multiple sequence alignments. However, most phylogenetic software only implement the I+ discrete Γ (Yang, 1994) model. The discrete Γ model is widely used because it requires less computing time than the continuous Γ model. Moreover, there is no reason to prefer the continuous Γ model to the discrete Γ model because none of them has strong biological justifications (Yang, 2014). The identifiability of the GTR+I+ discrete Γ model is unproven (Chai and Housworth, 2011) and it is unclear how well popular phylogenetic software estimate parameters of the I+ discrete Γ model.

Thus, we assessed the accuracy of the I+ discrete Γ estimators implemented in commonly used model-based phylogenetic software.

3.2 Results

We investigated how accurate RAxML (Stamatakis, 2014), PhyML (Guindon *et al.*, 2010), MrBayes (Ronquist *et al.*, 2012) and IQ-TREE (Nguyen *et al.*, 2015) infer invariable proportion, shape of the Γ distribution and tree length for long simulated alignments with 100,000 sites, that guaranteed to recover the correct tree topology.

Figure 3.1, 3.2, 3.3 and 3.4 display the average of shape parameters $\hat{\alpha}$, the fractions of invariable sites $\hat{p_{inv}}$ and the tree lengths estimated for alignments simulated from

trees with 6, 24 and 96 taxa assuming a JC+I+Γ4 model and different combinations of α and p_{inv} by PhyML, RAxMl, MrBayes and IQ-TREE, respectively. The red boxes indicate parameter combinations where the estimates deviate more than 25% from the true values. Yellow rectangles indicate a medium deviation from the true values ($10 - 25\%$), while the green rectangles show correctly estimated parameters ($< 10\%$ deviation).

None of the tested programs estimated all parameter combinations correctly for the 6-taxon alignments. For extreme rate heterogeneity ($\alpha = 0.1$), PhyML and MrBayes recovered the true parameters for $p_{inv} \leq 0.5$ and $p_{inv} \geq 0.5$, respectively, whereas estimates by IQ-TREE and RAxML were all incorrect. For strong rate heterogeneity ($\alpha = 0.5$) the accuracy of the estimates improved. However, the results of the programs differ unsystematically. IQ-TREE and MrBayes estimated six parameter combinations correctly, whereas for RAxML and PhyML only four and two estimates were correct, respectively. For small rate variation ($\alpha = 1.0$), PhyML, RAxML and MrBayes failed for some α and p_{inv} combinations, whereas IQ-TREE estimated all combinations correctly.

With increased taxon sampling (24- and 96-taxon alignments), we observed substantially more correct estimates. These results corroborate a previous study (Sullivan *et al.*, 1999) showing that increased taxon sampling leads to reliable estimates. However under extreme rate heterogeneity, only MrBayes estimated the parameter combinations correctly.

We also noted that inaccurate estimates of α and p_{inv} could lead to a substantial overestimation of the tree lengths. For instance, on the 96-taxon alignments simulated from $\alpha = 0.1$ and $p_{inv} = 0.8$ (true tree length = 18.9) estimated tree lengths produced by PhyML and IQ-TREE were 125.85 and 333.75, respectively. Similarly, MrBayes and RAxML produced tree lengths that were many times longer than the true values for alignments simulated from the 96 taxon tree with $\alpha = 1.0$ and $p_{inv} = 0.9$ and from the 24 taxon tree with $\alpha = 0.1$ and $p_{inv} = 0.3$, respectively.

In terms of computing times, MrBayes needed on average 50 minutes, 3 days and 13 days for the 6-, 24-, and 96-taxon alignments, respectively. In contrast, the ML programs were much faster. For instance, PhyML only needed 25 seconds, 20 minutes and 2 hours for the respective alignments. Computing time of RAxML and IQ-TREE is also of the same order.

PhyML

α \ p_{inv}	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	
6 Taxon Tree (Tree Length = 0.9)											
0.10	0.00	0.10	0.20	0.27	0.39	0.55	0.87	0.89	0.90	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.07	0.09	0.10	0.59	0.54	0.21	0.13	**Est.** α
	0.90	0.90	0.90	0.87	0.88	18.85	1.99	1.78	5.89	0.40	**Est. tree length**
0.50	0.19	0.18	0.18	0.17	0.27	0.37	0.45	0.50	0.75	0.90	**Est.** p_{inv}
	0.82	0.62	0.47	0.36	0.35	0.33	0.28	0.21	0.27	0.50	**Est.** α
	0.90	0.90	0.90	0.89	0.88	0.88	0.86	0.78	1.08	1.09	**Est. tree length**
1.00	0.13	0.19	0.19	0.19	0.18	0.17	0.18	0.30	0.46	0.90	**Est.** p_{inv}
	1.66	1.45	0.97	0.67	0.46	0.31	0.21	0.17	0.14	1.02	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.88	0.85	0.80	0.73	0.89	**Est. tree length**
24 Taxon Tree (Tree Length = 4.5)											
0.10	0.64	0.66	0.66	0.72	0.74	0.54	0.61	0.70	0.80	0.93	**Est.** p_{inv}
	0.72	0.63	0.52	0.32	0.30	0.13	0.09	0.10	0.11	0.21	**Est.** α
	6.31	7.17	11.03	45.18	46.16	13.72	19.59	5.60	3.73	6.74	**Est. tree length**
0.50	0.01	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.51	0.50	0.50	0.50	0.50	0.50	0.49	0.50	0.49	0.50	**Est.** α
	4.49	4.50	4.50	4.50	4.50	4.50	4.53	4.51	7.02	4.53	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.01	1.01	1.00	0.99	0.99	1.00	0.99	1.00	1.00	6.43	**Est.** α
	4.51	4.50	4.50	4.50	4.50	4.50	4.51	4.51	7.57	161.62	**Est. tree length**
96 Taxon Tree (Tree Length = 18.9)											
0.10	0.28	0.43	0.49	0.55	0.62	0.50	0.62	0.73	0.90	0.93	**Est.** p_{inv}
	0.23	0.27	0.28	0.28	0.28	0.10	0.13	0.17	0.32	0.19	**Est.** α
	15.01	13.87	13.98	13.93	13.99	19.00	18.43	21.72	125.85	28.48	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.75	0.89	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.31	0.47	**Est.** α
	18.88	18.89	18.87	18.90	18.96	19.02	19.00	19.17	9.32	17.48	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.01	1.04	**Est.** α
	18.90	18.90	18.90	18.86	18.86	18.91	19.03	18.74	19.00	35.51	**Est. tree length**

FIGURE 3.1: Estimates of p_{inv}, α and tree lengths by PhyML on alignments simulated from 6-, 24- and 96-taxon trees. Estimates are colored according to their differences from the true values: red (more than 25% deviation), yellow (10% to 25% deviation) and green (less than 10% deviation). For the estimated is green if , yellow if and red if .

RAxML

α \ p_{inv}	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	
6 Taxon Tree (Tree Length = 0.9)											
0.10	0.71	0.74	0.77	0.79	0.82	0.84	0.86	0.89	0.90	0.94	**Est.** p_{inv}
	4.79	3.86	2.98	2.11	1.27	0.82	0.58	0.56	0.32	0.32	**Est.** α
	0.93	0.95	0.98	1.05	1.23	1.81	2.08	1.76	3.25	0.50	**Est. tree length**
0.50	0.30	0.36	0.40	0.46	0.51	0.57	0.64	0.73	0.82	0.91	**Est.** p_{inv}
	1.33	1.22	1.08	0.96	0.84	0.75	0.67	0.65	0.72	0.83	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.91	**Est. tree length**
1.00	0.16	0.23	0.30	0.37	0.45	0.53	0.62	0.71	0.80	0.90	**Est.** p_{inv}
	1.91	1.79	1.63	1.48	1.38	1.26	1.18	1.15	1.14	1.03	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	**Est. tree length**
24 Taxon Tree (Tree Length = 4.5)											
0.10	0.64	0.67	0.70	0.72	0.66	0.71	0.77	0.82	0.88	0.94	**Est.** p_{inv}
	0.72	0.64	0.57	0.31	0.15	0.15	0.18	0.24	0.24	0.30	**Est.** α
	6.33	7.22	8.54	51.38	35.24	36.78	28.85	10.93	7.63	3.00	**Est. tree length**
0.50	0.01	0.11	0.21	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.51	0.51	0.51	0.51	0.51	0.51	0.50	0.50	0.50	0.51	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.50	4.52	4.52	**Est. tree length**
1.00	0.01	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.02	1.02	1.01	1.01	1.00	1.00	1.00	1.00	1.00	1.00	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.51	**Est. tree length**
96 Taxon Tree (Tree Length = 18.9)											
0.10	0.37	0.43	0.49	0.65	0.62	0.68	0.74	0.80	0.87	0.94	**Est.** p_{inv}
	0.27	0.27	0.28	0.62	0.28	0.28	0.27	0.25	0.26	0.29	**Est.** α
	13.86	13.89	13.93	13.40	14.02	14.08	14.19	16.01	15.07	11.19	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	**Est.** α
	18.89	18.90	18.90	18.90	18.90	18.90	18.90	18.90	18.90	18.92	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**Est.** α
	18.90	18.90	18.90	18.90	18.89	18.90	18.89	18.90	18.92	18.99	**Est. tree length**

FIGURE 3.2: Estimates of p_{inv}, α and tree lengths by RAxML

MrBayes

α \ p_{inv}	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	
6 Taxon Tree (Tree Length = 0.9)											
0.10	0.08	0.14	0.24	0.32	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.59	0.36	0.33	0.17	0.12	0.10	0.10	0.10	0.10	0.11	**Est.** α
	0.90	0.91	0.91	0.91	0.91	0.90	0.90	0.92	0.99	0.84	**Est. tree length**
0.50	0.10	0.14	0.21	0.29	0.38	0.48	0.59	0.69	0.69	0.76	**Est.** p_{inv}
	0.65	0.59	0.54	0.51	0.48	0.48	0.49	0.48	0.32	0.16	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.77	0.56	**Est. tree length**
1.00	0.04	0.09	0.17	0.25	0.37	0.45	0.55	0.63	0.69	0.70	**Est.** p_{inv}
	1.14	1.01	0.95	0.90	0.95	0.90	0.89	0.86	0.71	0.17	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.89	0.84	0.68	**Est. tree length**
24 Taxon Tree (Tree Length = 4.5)											
0.10	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	**Est.** α
	4.48	4.51	4.50	4.51	4.51	4.51	4.52	4.56	5.33	4.63	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.86	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.22	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.51	4.53	4.44	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.64	**Est.** p_{inv}
	1.01	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.06	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.51	12.70	**Est. tree length**
96 Taxon Tree (Tree Length = 18.9)											
0.10	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	**Est.** α
	18.85	18.92	18.93	18.95	18.93	18.94	19.00	19.06	19.11	18.90	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.77	0.87	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.38	0.34	**Est.** α
	18.88	18.90	18.90	18.90	18.91	18.91	18.92	18.92	12.91	13.18	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.88	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.88	0.70	**Est.** α
	18.90	18.90	18.90	18.90	18.90	18.90	18.90	18.91	17.32	99.26	**Est. tree length**

FIGURE 3.3: Estimates of p_{inv}, α and tree lengths by MrBayes

IQ-TREE

α \ p_{inv}	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	
6 Taxon Tree (Tree Length = 0.9)											
0.10	0.72	0.74	0.77	0.79	0.82	0.84	0.86	0.89	0.90	0.93	**Est.** p_{inv}
	5.04	4.07	3.16	2.31	1.42	0.72	0.56	0.55	0.27	0.26	**Est.** α
	0.93	0.95	0.98	1.04	1.22	1.75	2.06	1.75	2.54	0.44	**Est. tree length**
0.50	0.19	0.25	0.31	0.37	0.43	0.51	0.60	0.71	0.81	0.91	**Est.** p_{inv}
	0.82	0.78	0.70	0.64	0.56	0.53	0.52	0.52	0.57	0.75	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.87	0.74	**Est. tree length**
1.00	0.01	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.03	1.01	1.01	1.00	1.02	1.00	1.00	1.03	1.07	1.03	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.89	0.89	**Est. tree length**
24 Taxon Tree (Tree Length = 4.5)											
0.10	0.64	0.67	0.70	0.72	0.75	0.71	0.76	0.81	0.85	0.95	**Est.** p_{inv}
	0.72	0.64	0.57	0.31	0.30	0.14	0.14	0.14	0.12	0.36	**Est.** α
	6.33	7.22	8.52	50.61	52.95	36.54	38.43	41.53	28.84	18.93	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.51	4.53	4.56	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.58	**Est. tree length**
96 Taxon Tree (Tree Length = 18.9)											
0.10	0.36	0.43	0.59	0.65	0.70	0.75	0.79	0.72	0.89	0.94	**Est.** p_{inv}
	0.27	0.27	0.58	0.62	0.57	0.42	0.43	0.12	0.26	0.19	**Est.** α
	13.84	13.87	13.21	13.44	23.85	44.20	42.81	57.90	333.75	44.51	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.89	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.45	**Est.** α
	18.89	18.90	18.89	18.92	18.93	18.96	19.00	19.06	19.21	18.23	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	**Est.** α
	18.90	18.90	18.91	18.91	18.92	18.94	18.96	19.02	19.15	19.62	**Est. tree length**

FIGURE 3.4: Estimates of p_{inv}, α and tree lengths by IQ-TREE

Because the tested programs performed quite differently with respect to parameter estimation, the number of taxa cannot be the only reason for the lack of accuracy. We suspected that the optimization heuristics as implemented in the programs drive the accuracy of the parameter estimation. Depending on the simulated parameter combinations, we observed that the optimization routines were sometimes stuck in local optima resulting in estimates that differ dramatically from the true combinations (data not shown). Thus, we changed the optimization strategy currently implemented in IQ-TREE to account for plateaus observed in the likelihood surfaces (Sullivan *et al.*, 1999). To this end, we developed IQ-TREE-Improved in which optimization routine is restarted from ten evenly spaced initial $\alpha \in [0.1, 1.0]$

This extended search found the correct parameter estimates for almost all parameter combinations (Figure 3.5). The computational costs and the improvements in log-likelihoods of the new optimization strategy compared to the old strategy are displayed in Figure 3.6. Here, we observed that the log-likelihoods of IQ-TREE-Improved were always better or equal to the old strategy. This substantiates our assumption that the incorrect estimation of the parameters is caused by insufficient optimization. Fortunately, the improvement of the parameter estimation with a more complex optimization heuristic only increased the total computing time by a factor of 1.17.

3.3 Discussion

Our simulations revealed a major issue for parameter estimation of the I+ discrete Γ model as implemented in many phylogenetic software. Despite using very long alignments, none of the tested programs recovered the true α, p_{inv} and tree length for all parameter combinations. Often, the estimates deviated heavily from the true values and different programs estimated different values for the same simulated evolutionary parameters, although all of them inferred the true tree. Thus, the optimization heuristics in ML programs are responsible for the varying, sub-optimal estimates. The good performance of MrBayes is likely attributed to the Bayesian sampling of the parameter space that can escape local optima more effectively. However, it comes at a cost of excessive computing time.

IQ-TREE-Improved

α \ p_{inv}	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	
6 Taxon Tree (Tree Length = 0.9)											
0.10	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.13	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.91	0.85	0.46	**Est. tree length**
0.50	0.06	0.12	0.20	0.30	0.39	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.59	0.55	0.51	0.52	0.50	0.50	0.50	0.50	0.52	0.50	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.89	0.92	**Est. tree length**
1.00	0.01	0.09	0.18	0.28	0.39	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.03	0.98	0.96	0.94	0.99	0.99	0.99	1.01	1.01	1.01	**Est.** α
	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.90	**Est. tree length**
24 Taxon Tree (Tree Length = 4.5)											
0.10	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	**Est.** α
	4.49	4.50	4.49	4.50	4.50	4.51	4.52	4.55	4.35	3.99	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.52	4.53	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**Est.** α
	4.50	4.50	4.50	4.50	4.50	4.50	4.50	4.51	4.51	4.57	**Est. tree length**
96 Taxon Tree (Tree Length = 18.9)											
0.10	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	**Est.** α
	18.88	18.93	18.93	18.95	18.96	18.96	19.01	19.11	20.25	20.77	**Est. tree length**
0.50	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	**Est.** α
	18.89	18.90	18.89	18.92	18.94	18.96	19.00	18.95	18.89	19.30	**Est. tree length**
1.00	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	**Est.** p_{inv}
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	**Est.** α
	18.90	18.90	18.91	18.91	18.91	18.93	18.96	19.01	19.15	19.55	**Est. tree length**

FIGURE 3.5: (a) Estimates of α, p_{inv} and tree lengths by the improved version of IQ-TREE. The color code is explained in Figure 3.1.

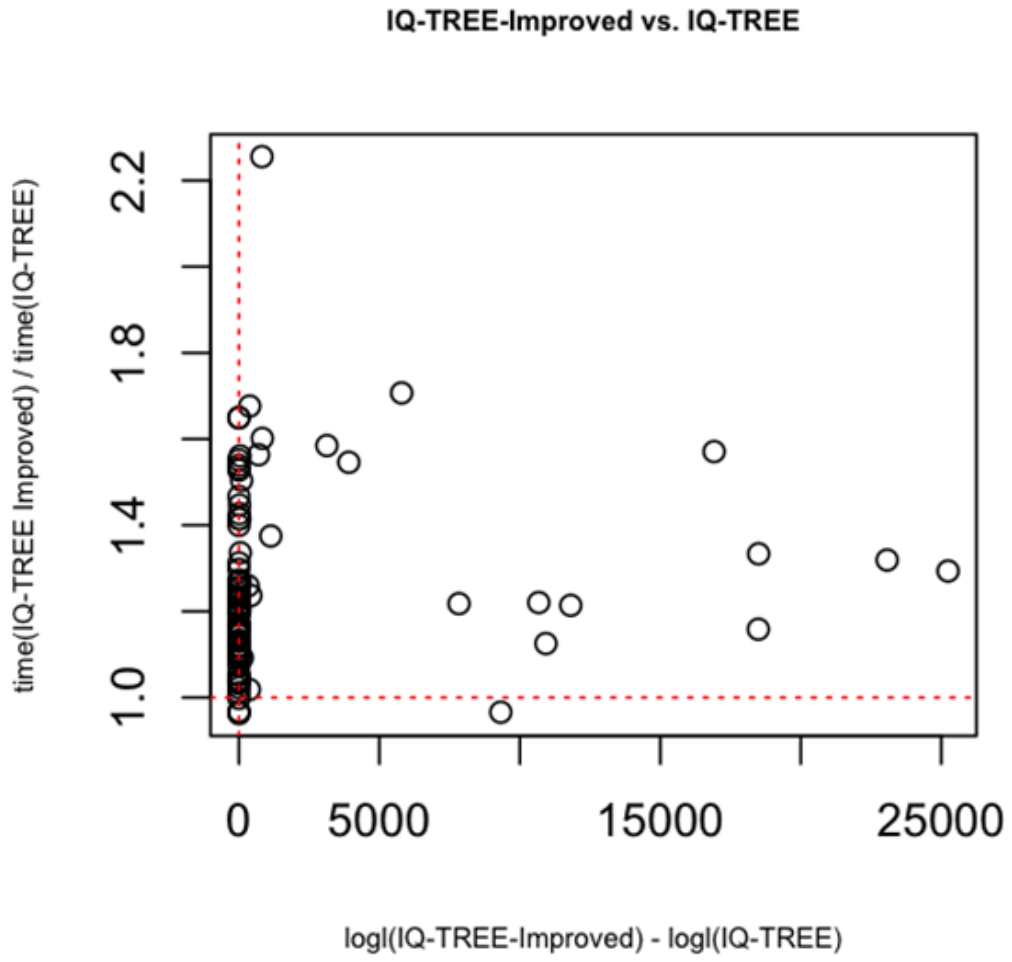

FIGURE 3.6: Run time ratios and log-likelihood differences when IQ-TREE-Improved was compared with IQ-TREE. Each open circle represents the average log-likelihood differences and run time ratios computed from the 100 alignments simulated for each pair of α and p_{inv}.

We showed that a more thorough exploration of the parameter space lead to correct ML estimates. Thus, one may speculate that the GTR+I+ discrete Γ model is also identifiable like the GTR+I+ continuous Γ model.

Moreover, we want to emphasize that developing good parameter estimation method is equally important to proposing new and complex models of sequence evolution. It is not enough to recover the true tree, if one wants to understand how evolutionary forces shaped contemporary genomes. The effect of wrong parameter

estimates for the substitution model on the total tree length is sometimes dramatic (see Figure 3.1, 3.2, 3.3 and 3.4). Thus, one should critically scrutinize the heuristics implemented in popular programs and a more thorough evaluation of phylogenetic inference programs allowing for very complicated models of sequence evolution is necessary, but beyond the scope of this thesis.

3.4 Materials and Methods

We used Seq-Gen (Rambaut and Grass, 1997) to simulate 100,000-bp alignments along three balanced trees containing 6, 24 and 96 taxa, each has equal branch lengths of 0.1 substitutions per site. We assumed the Jukes-Cantor model (Jukes and Cantor, 1969) and the rate heterogeneity model I+ discrete Γ using four rate categories. For each tree and each combination of α and p_{inv} where $p_{inv} \in 0.0, 0.1, ..., 0.9$ and $\alpha \in 0.1, 0.5, 1.0$ we simulated 100 alignments. For each alignment, we performed tree reconstruction with RAxML version 8.2.2, IQ-TREE version 1.3.7 and MrBayes version 3.2.2 (compiled with the BEAGLE library (Ayres *et al.*, 2012)), using the JC+I+Γ model. We used the default tree search options in IQ-TREE, RAxML and PhyML. The IQ-TREE-Improved version is invoked via the -test_param option. With MrBayes we ran one million generations using four chains. We then computed the mean of the 100 estimates for α, p_{inv} and tree lengths (sum of branch lengths)

Chapter 4

IQ-TREE-SP: A data-driven heuristic for constraining tree space

4.1 Introduction

The influx of biological data generated by Next Generation Sequencing technologies has created many new challenges in the so-called phylogenomic era. Multiple sequence alignments containing hundreds thousands sites are being used to resolve long-standing phylogenetic uncertainties (e.g. McCormack *et al.* (2013); Misof *et al.* (2014)). Although such large datasets help to improve the accuracy of phylogenetic inference by providing more phylogenetic signals, they require enormous amount of computing time. It would be of benefit if the added information can also be used to improve the time efficiency of maximum likelihood methods.

Many heuristics for speeding up the maximum-likelihood tree inference have been introduced over the years. Guindon and Gascuel (2003) proposed the simultaneous NNI modification in PhyML that substantially reduced the computing time of the NNI search. Stamatakis (2006) used many computational shortcuts and low-level optimization techniques to economize the SPR search in RAxML. Also in the realm of SPR search, Hordijk and Gascuel (2005) used the maximum parsimony method to quickly filter out non-promising SPR moves, thus reducing the number of moves needed to be evaluated by the maximum-likelihood method.

Unlike RAxML or PhyML which produces a single SPR-optimal tree, IQ-TREE samples many NNI-optimal trees. It is reasonable to assume that trees with high likelihood cluster together. In other words, they share common splits with the maximum-likelihood tree. To this end, we propose IQ-TREE-SP, a complementary data-driven heuristic for the IQ-TREE algorithm. IQ-TREE-SP infers so-called stable splits that are present in the best locally optimal trees and use those splits to constrain the search space. In what follows, we describe in detail the mechanism of IQ-TREE-SP.

4.2 Methods

Let $\mathbb{T}$ be the set of the current 20 best locally optimal trees. A split is considered stable if it is present in at least 90% of the trees in $\mathbb{T}$. We denote by $\mathbb{S}$ the set of stable splits. $\mathbb{T}$ and $\mathbb{S}$ are continuously updated during the tree search. The general idea of IQ-TREE-SP is to avoid doing NNI on the stable splits. Here, we only perform NNIs on the stable splits with probability of 10% and the number of random NNIs performed in the stochastic NNI is reduced from $(n-3)/2$ to $(n-3-|\mathbb{S}|)/2$, where n is the number of taxa.

In addition, we also employ techniques from the tabu search (Glover, 1989). Here, we maintain a list $\mathbb{B}$ of so-called tabu splits. When a random NNI is performed, we add the new split into $\mathbb{B}$. In the first round of the hill-climbing NNI step, we do not perform NNIs on the tabu splits. In the next rounds, we empty $\mathbb{B}$ so that the tabu restriction is abolished. The rational for using the tabu list is as follows. The number of random NNIs performed in the stochastic step becomes smaller when $|\mathbb{S}|$ gets larger. Thus, if a only small number of random NNIs is applied, the subsequent hill-climbing NNI has little chance of escaping the current local optima. In fact, it will likely undo the random NNIs and the search will return to the previous local optimum. The tabu list $\mathbb{B}$ prevents the hill-climbing NNI to prematurely undo the random NNIs and help the search move far away from the current local optimum.

The following pseudo-code describes how IQ-TREE-SP is integrated into the main IQ-TREE algorithm:

Initial tree generation step

1. Initialize $\mathbb{T}$ with the 20 locally optimal trees generated in the initial tree generation step.
2. Initialize $\mathbb{S}$ with stable splits derived from $\mathbb{T}$.

Stochastic NNI step

1. Set $i = 0$; $\mathbb{B} = \emptyset$.
2. Randomly choose an internal branch b:
   - IF $b \notin \mathbb{B}$ AND $s \notin \mathbb{S}$:
     (a) Perform a random NNI on b.
     (b) Add b to $\mathbb{B}$.
     (c) Set $i = i + 1$.
   - IF $b \notin \mathbb{B}$ and $s \in \mathbb{S}$: do (a)-(c) with probability 10%.
   - ELSE go to 2.
3. STOP if $i = (n - 3 - |\mathbb{S}|)/2$ where n is the number of taxa; otherwise go to 2.

Hill-climbing NNI step

1. For each internal branch b:
   - IF $b \notin \mathbb{B}$ AND $s \notin \mathbb{S}$:
     Evaluate the likelihood of the two NNI-trees.
   - IF $b \notin \mathbb{B}$ AND $s \in \mathbb{S}$:
     Evaluate the likelihood of the two NNI-trees with probability 10%.
2. IF no improving NNI found:
   - IF $\mathbb{B} \neq \emptyset$: GOTO 4.
   - ELSE: set $\mathbb{B} = \emptyset$ AND GOTO 1.
3. Apply all improving NNIs; set $\mathbb{B} = \emptyset$ and GOTO 1.
4. If the likelihood of the current tree is better than that of the worst tree in $\mathbb{T}$:
   - Replace the worst tree in $\mathbb{T}$ with the current tree.
   - Update $\mathbb{S}$ based on the new $\mathbb{T}$

ID_Type	# Taxa	# Sites	Source
1_DNA	128	29,198	(Stamatakis and Alachiotis, 2010)
2_DNA	180	14,912	(van der Linde *et al.*, 2010)
3_DNA	237	43,834	(Nyakatura and Bininda-Emonds, 2012)
4_DNA	298	5,074	(Bouchenak-Khelladi *et al.*, 2008)
5_DNA	372	61,199	(Springer *et al.*, 2012)
6_DNA	404	13,158	(Stamatakis and Alachiotis, 2010)
7_DNA	435	16,016	(Hinchliff and Roalson, 2013)
8_DNA	767	5,714	(Pyron *et al.*, 2011)
9_AA	69	8,546	(Dell'Ampio *et al.*, 2014)
10_AA	70	11,789	(Dell'Ampio *et al.*, 2014)
11_AA	72	12,548	(Dell'Ampio *et al.*, 2014)

TABLE 4.1: Benchmark alignments

4.3 Results

We tested the performance of IQ-TREE-SP against the standard version IQ-TREE on 11 large alignments (Table 4.1). For each alignment, we ran each version ten times, always assuming the same evolutionary model (GTR+G for DNA and LG+G for amino acids). We then measured the average speedups obtained by IQ-TREEE-SS over IQ-TREE for each alignment. In addition, we compared the tree log-likelihoods produced by each method.

To account for the stochastic nature of the stopping rule which leads to varying running time, for each alignment we used the average computing time for one search iteration as a basis for comparison:

$$\text{time(version)} = \frac{1}{10}\sum_{i=1}^{10}\frac{t_i}{n_i},$$

where t_i and n_i is the CPU time and the number of iterations required for each run, respectively. The speedup provided by IQ-TREE-SP is then computed as time(iq-tree)/time(iq-tree-ss).

Figure 4.1 shows the average speedup obtained by IQ-TREE-SP over IQ-TREE. We found an average speedup of 2.4 times, ranging from 1.6 to 3.9 times. The median speedup is 2.2 times. For alignments 1_DNA, 4_DNA, 5_DNA IQ-TREE-SP achieved substantial speedup of more than 3 times.

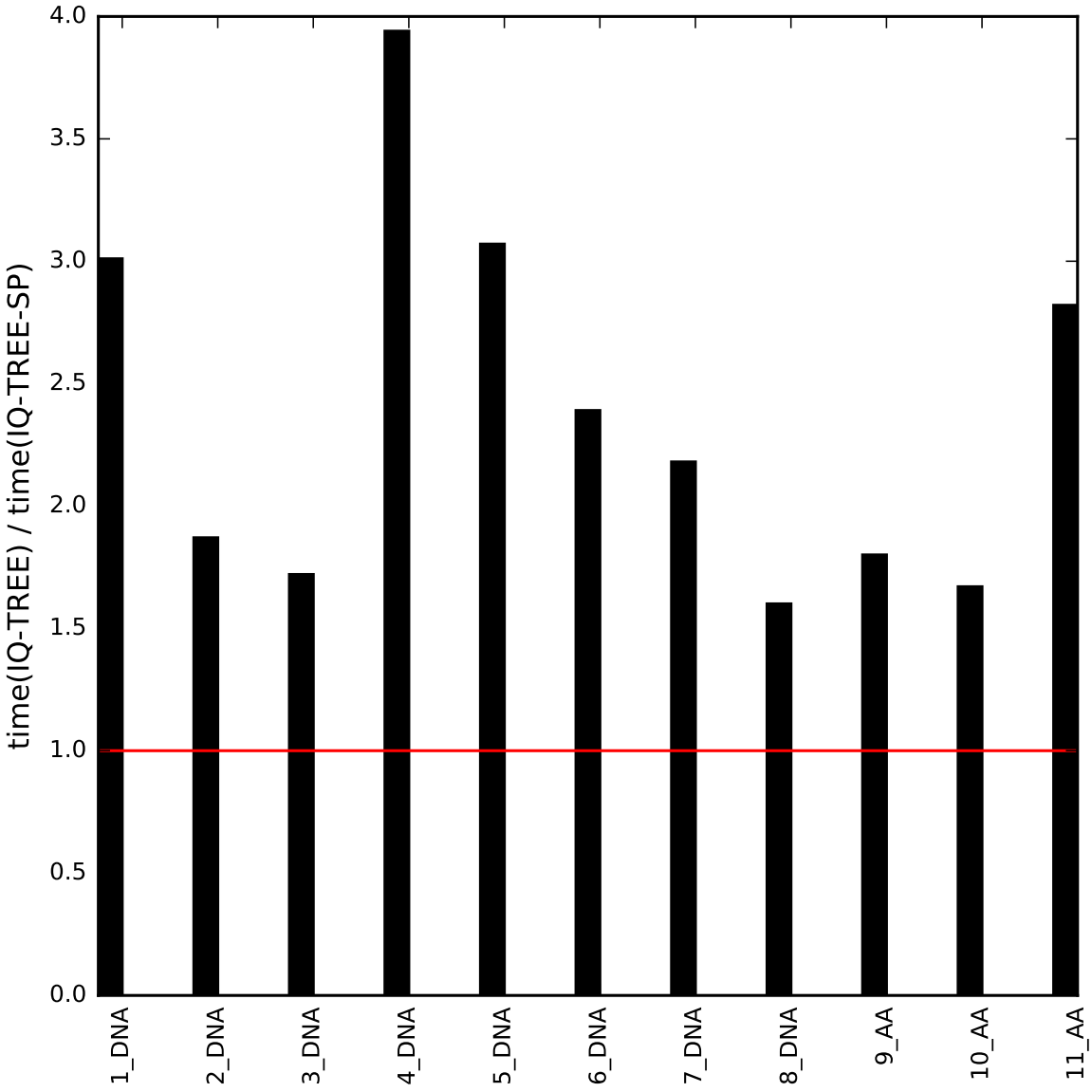

FIGURE 4.1: Speedup provided by IQ-TREE-SP over IQ-TREE

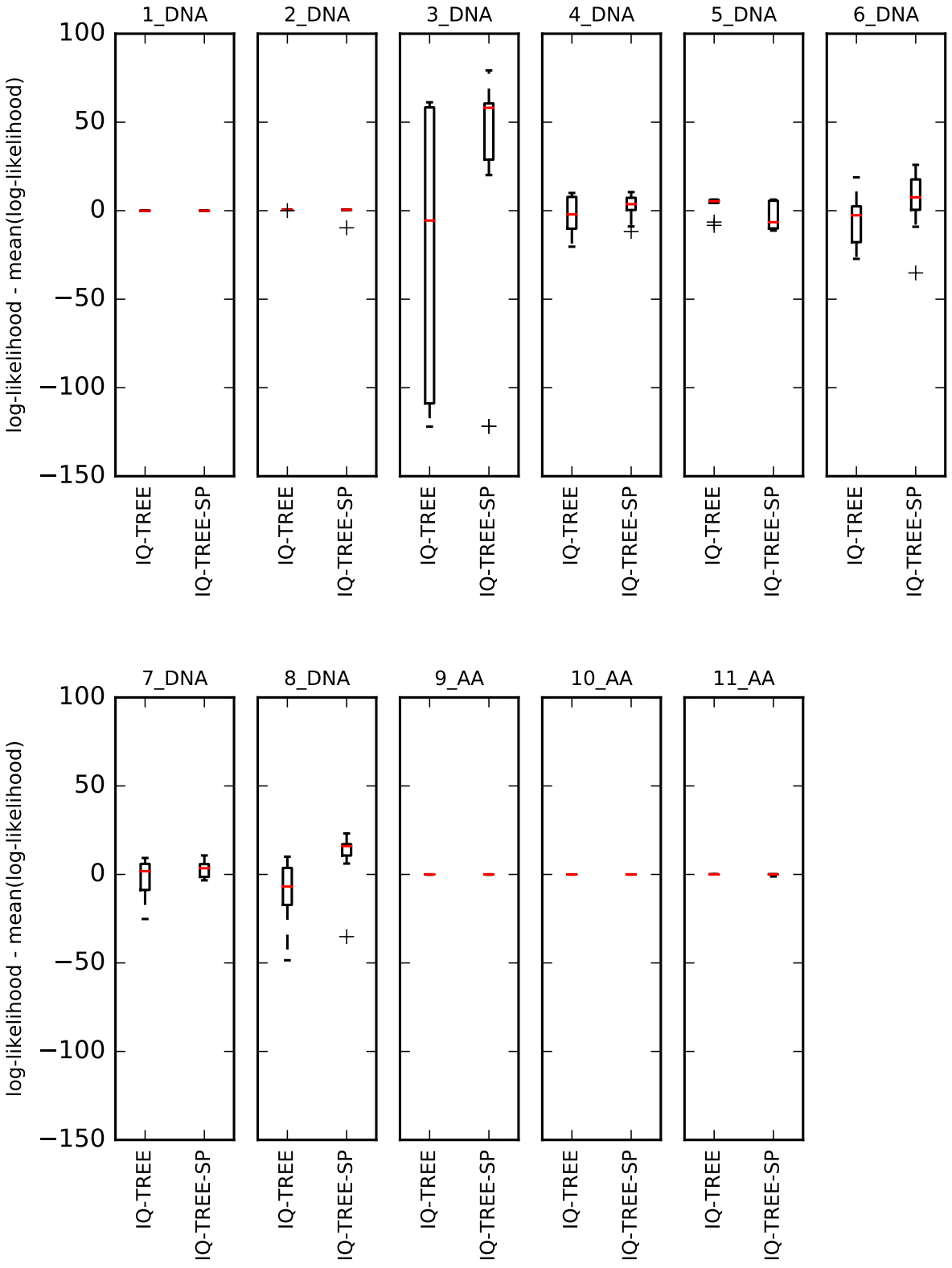

FIGURE 4.2: Log-likelihood comparison between IQ-TREE and IQ-TREE-SP

To compare the log-likelihoods among different alignments, we normalized the log-likelihood as follows. First, we computed the average log-likelihood from the 20 runs (10 for each IQ-TREE version) for each alignment. We computed then for each IQ-TREE version and for each of the 10 inferred trees the deviation of the corresponding log-likelihood from the global mean (Figure 4.2).

For four alignments (1_DNA, 9_AA, 10_AA and 11_AA), IQ-TREE and IQ-TREE-SP produced trees with same log-likelihood. IQ-TREE-SP produced better log-likelihoods than the standard algorithm on five alignments: 3_DNA, 4_DNA, 6_-DNA, 7_DNA and 8_DNA. Moreover, IQ-TREE-SP found trees with the highest log-likelihood on those alignments. On the two alignments 2_DNA and 5_DNA, the log-likelihoods produced by IQ-TREE are more congruent than those found in IQ-TREE-SP. However, the best log-likelihoods were found by both IQ-TREE and IQ-TREE-SP for those alignments.

4.4 Conclusions

We have proposed an improvement heuristic for the IQ-TREE search algorithm to make the tree search more efficient. Our heuristic employs a data-driven approach for constraining the search space. We use the topological congruency among the locally optimal trees as an indication for the strength of the phylogenetic signals provided by the data. Therefore, we avoid doing tree rearrangements on splits shared by the best trees. Moreover, the new heuristic helps to navigate the tree search in a more systematic way, which increases the chance of finding better trees. Our idea was inspired by the technique termed *reduction* proposed by Lin and Kernighan (1973) in their seminal paper on the traveling salesman problem. For the test alignments, the new heuristic achieved speedups of up to 3.9 times while the obtained tree log-likelihoods are equal or better than those produced by the standard IQ-TREE algorithm.

Chapter 5

IQ-TREE-MPI: An efficient parallel tree search algorithm using the Message Passing Interface

5.1 Introduction

With the advent of cheap and ubiquitous multi-core and multi-processor computers, more and more phylogenetic software have utilized parallelization to shorten the computing time. Approaches to parallelization often fall under the following three categories: fine-grained, coarse-grained, and embarrassing parallelism (Foster, 1995). These types of parallelism differ in the amount of communication taking place among the processes. Fine-grained parallelism requires the most communication, whereas there is no communication in embarrassing parallelism.

In phylogenetic inference, fine-grained parallelism is employed by distributing the calculation of site likelihoods to different processes (Stamatakis and Ott, 2008). The results are then collected to compute the final likelihood. Implementation of this approach is straightforward. With the help of application programming interface such as OpenMP, adding fine-grained parallelization to existing software can be accomplished with very little coding. However, fine-grained parallelization does not scale very well (Stamatakis, 2015). During tree search, likelihood calculation

is performed millions of times. Every time the tree likelihood is computed, communication among the processes is required. Thus, if many processes are involved, communication overhead will eventually outweighs computing time, diminishing the advantage of using parallel computing.

Coarse-grained parallelism typically requires refactoring the tree search algorithm into large and independent computational subtasks so that communication overhead is minimized. Thus, not every search algorithm can be efficiently parallelized using the coarse-grained approach. Because of the moderate communication requirement, a well designed coarse-grained parallel algorithm might achieve very good scaling performance (Minh *et al.*, 2005).

Embarrassing parallelism is often used to execute multiple tree searches or perform non-parametric bootstrap analysis (Felsenstein, 1985) that involves tree inferences for a large number of pseudo-replicated alignments. Because of its simplicity, embarrassing parallelism is widely used (Zwickl, 2006; Pfeiffer and Stamatakis, 2010; Guindon *et al.*, 2010).

The memory consumption of a coarse-grained or embarrassing parallel program is often proportional to the number of processes, whereas a fine-grained parallel program needs only the same amount of memory as the sequential version. Therefore, the requirements of the phylogenetic analysis and the capacity of the computing platform are the deciding factors for choosing the most suitable parallelization scheme. Since fine-grained parallelization can be added to any existing tree search algorithm, one can also employ a hybrid approach (Rabenseifner, 2003) that combines fine-grained with coarse-grained parallelism to increase parallel efficiency.

The IQ-TREE algorithm is well suited for coarse-grained parallelization because it mainly consists of loosely coupled computing components, whose task is to sample local optima in the tree space. In this chapter, we present IQ-TREE-MPI, an efficient parallel tree search algorithm for IQ-TREE.

5.2 Methods

We used the Message Passing Interface (MPI) to distribute the computations of the parsimony and locally optimal trees among the processes. When a new tree is computed by a process, its topology and likelihood are sent to other processes so

that every process has the same copy of the generated trees. To avoid idle time, we use asynchronous messaging. Every process has a so-called *inbox*, in which trees received from other processes are stored. When a message is sent, it goes directly into the recipient's inbox and no immediate response is required. Thus, the sending process can quickly resume its current activity and computation in the receiving process is not interrupted. Figure 5.1 and Figure 5.2 show the parallelization schemes for the initial tree generation and optimization step, respectively (see also Section 2.2.2 and 2.2.3).

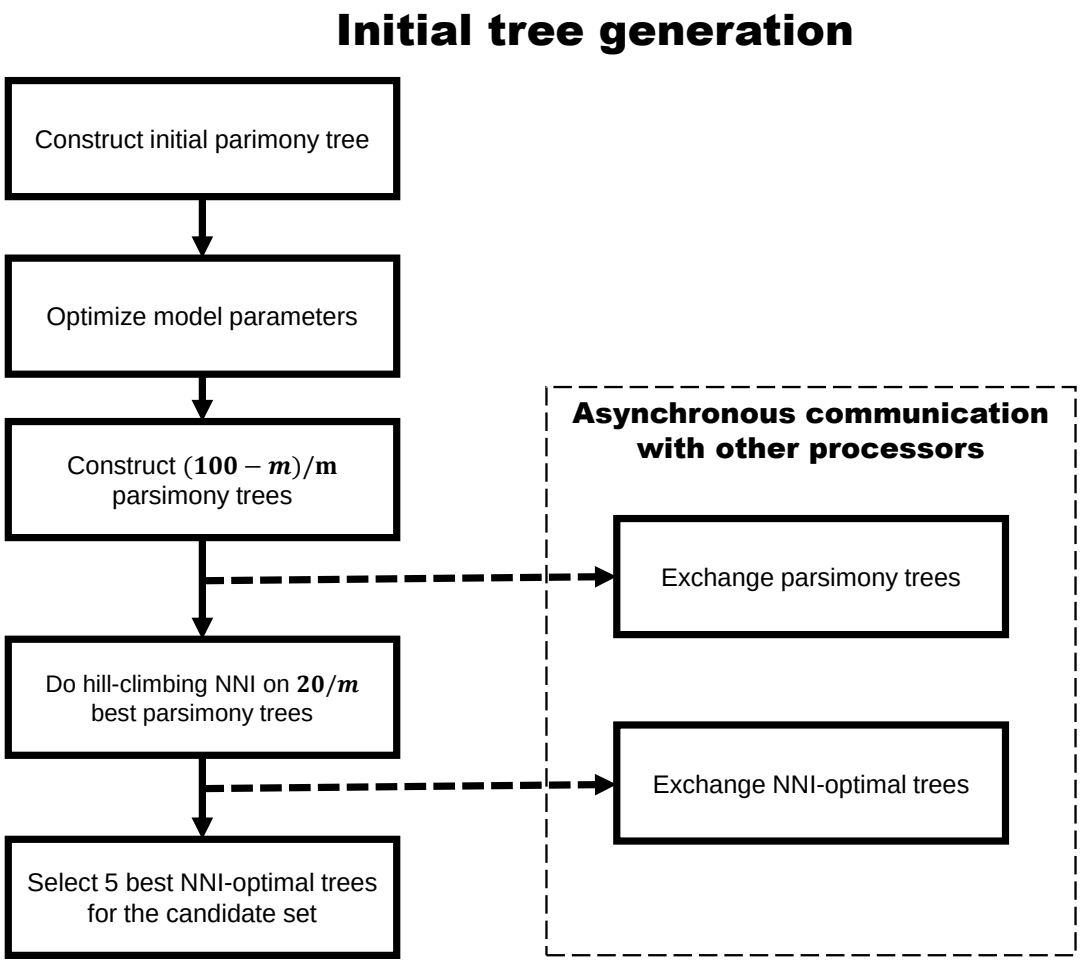

FIGURE 5.1: Parallelization scheme for the initial tree generation step (m is the number of processes).

In the initial tree generation step, the computations of the 100 parsimony and initial NNI-optimal trees are distributed equally among the processes. First, each process generates an initial parsimony tree and performs model optimization on

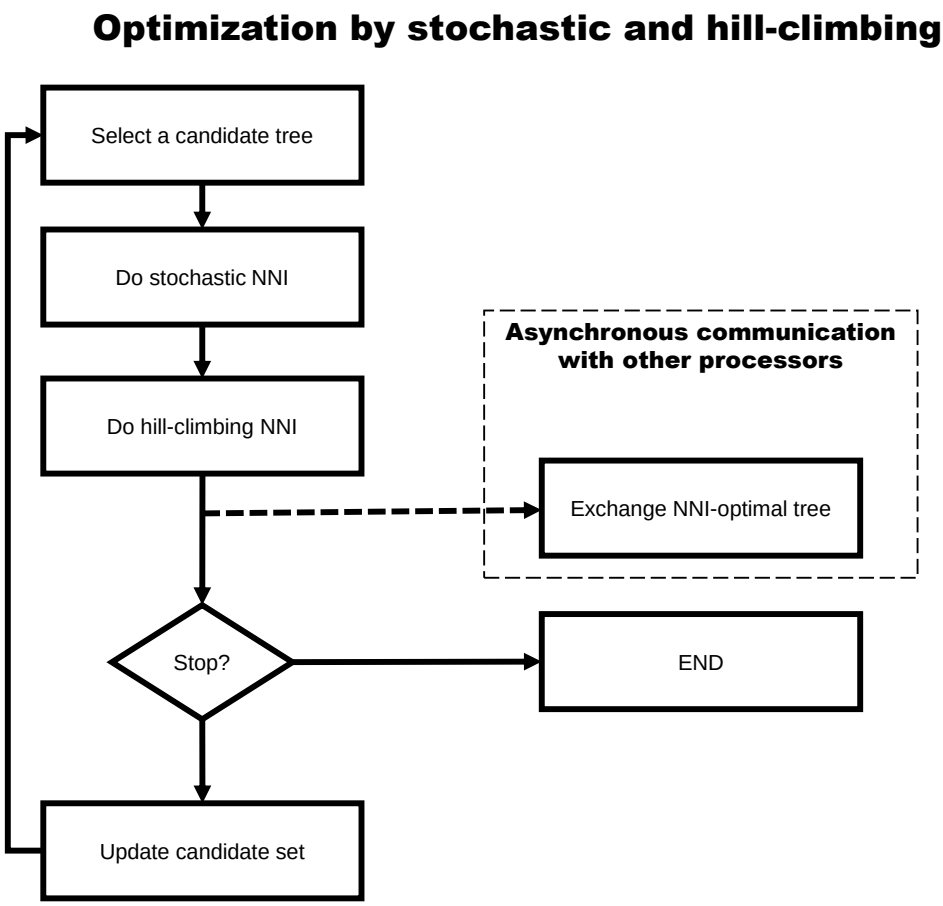

FIGURE 5.2: Parallelization scheme for the optimization step.

that tree. Then, it will compute $\lceil (100 - m)/m \rceil$ parsimony trees, where m is the number of processes. The processes then exchange the parsimony trees with each other until each of them has the same copy of the 100 parsimony trees. We then divide up the computations of the initial 20 NNI-optimal trees (from the top 20 parsimony trees) among the processes. The resulting NNI-optimal trees are then sent around so that in the end each process has the same copy of the candidate set.

In each iteration of the optimization step, each process performs the stochastic and hill-climbing NNI on a randomly selected candidate tree like in the sequential algorithm. Afterwards, the new NNI-optimal tree is sent to other processes and trees in the inbox are collected. Each process then updates its candidate set with better trees, if found. We use a so-called master process to keep track of the stopping condition. It increases the iteration counter every time a new tree is generated or received from other process. We use the same stopping condition

as in the sequential version. The search is stopped if a predefined number of iterations is reached or no better tree is found in the last 100 iterations. If one of the stopping conditions is met, the master process will send out a stop message to other processes informing them to finish.

5.3 Results

We benchmarked the performance of IQ-TREE-MPI using one DNA (218 taxa, 4182 sites) and one amino acid (74 taxa, 4013 sites) alignment. For both alignments we performed ten independent tree searches using IQ-TREE-MPI and the sequential IQ-TREE. To avoid running time fluctuation caused by the default stopping rule, the number of iterations is set to 1000. The speedup of IQ-TREE-MPI was computed as the ratio between the average run time of IQ-TREE and that of IQ-TREE-MPI. The computations were performed on computational nodes of the Vienna Scientific Cluster 3. Each node has two processors and each processor has eight cores. We tested IQ-TREE-MPI using 2, 4, 8, 16 and 32 cores.

Figure 5.3 shows an almost linear speedup of IQ-TREE-MPI. With 32 CPU cores, IQ-TREE-MPI is 26.4 and 28 times faster than the sequential version for the DNA and amino acid alignment, respectively. Moreover, we observed minimal communication overheads among the processes. Figure 5.4 shows the percentage of wall-clock time spent on communication with regard to the total running time for different number of CPU cores. As expected, the communication time rises with the number of CPU cores. However, it only makes up a negligible portion of the total running time. With 32 CPU cores the percentage of communication time for the DNA and amino acid alignment was only 0.08% and 0.022%, respectively. Thus, the efficiency of our parallel algorithm is not affected by the communication overhead.

We also investigated the performance of IQ-TREE-MPI with regard to the obtained tree log-likelihoods. Figure 5.5 compares the log-likelihood produced by IQ-TREE-MPI with different number of CPU cores with that by the sequential version. For the DNA alignment, IQ-TREE-MPI produced higher median log-likelihood in most settings. Only in the runs with 4 CPU cores, the median log-likelihood of IQ-TREE-MPI was lower than that of the sequential version.

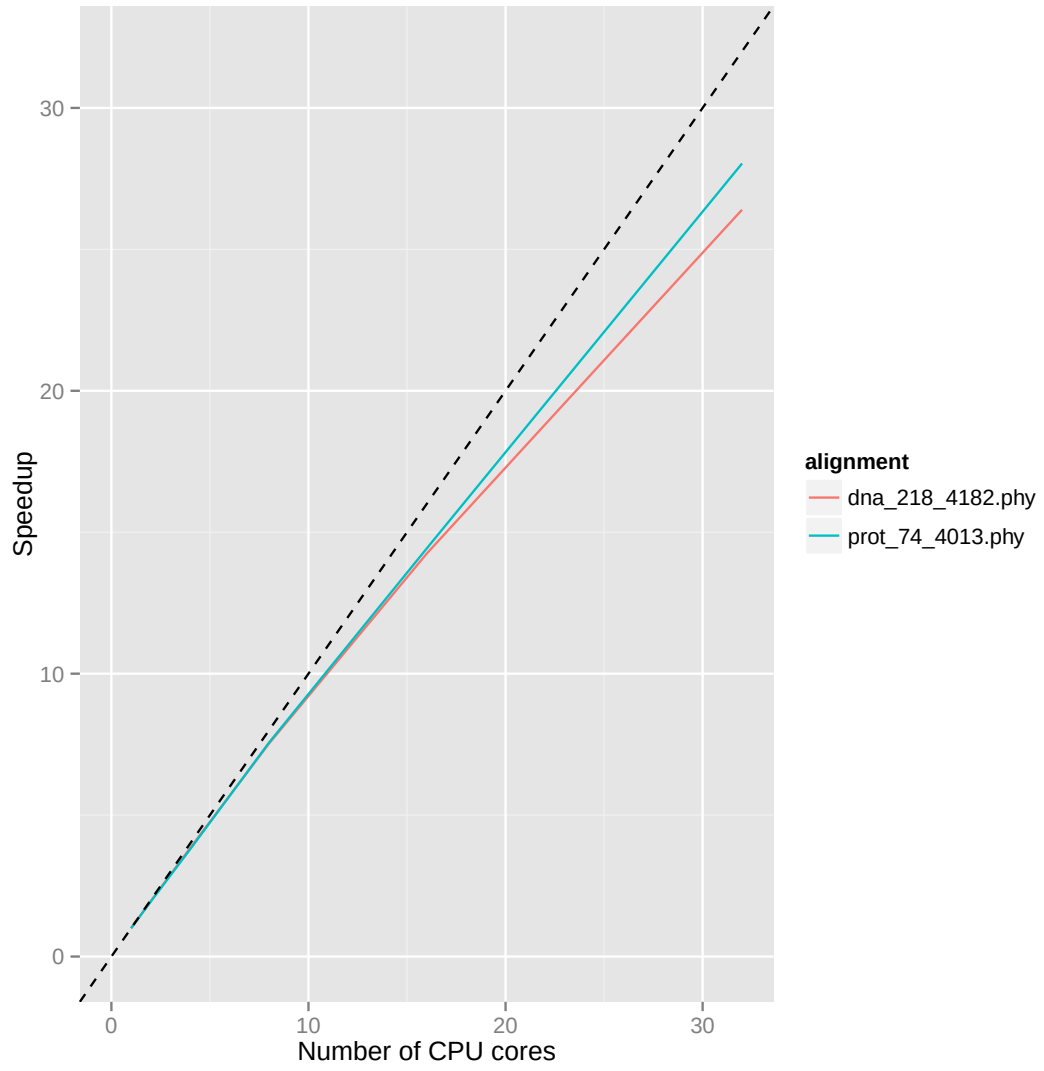

FIGURE 5.3: Speedups by IQ-TREE-MPI using different number of CPU cores. The dotted line represents the theoretical linear speedups

However, in these runs IQ-TREE-MPI found the log-likelihood. For the amino acid alignment, all versions produced the same results.

5.4 Discussions

We have presented an efficient parallelization of the IQ-TREE search algorithm. IQ-TREE-MPI provides very good speedup which scales well with the number of CPU cores. With regard to the tree log-likelihood, it also exhibited better performance than the sequential algorithm in most test instances. This result is

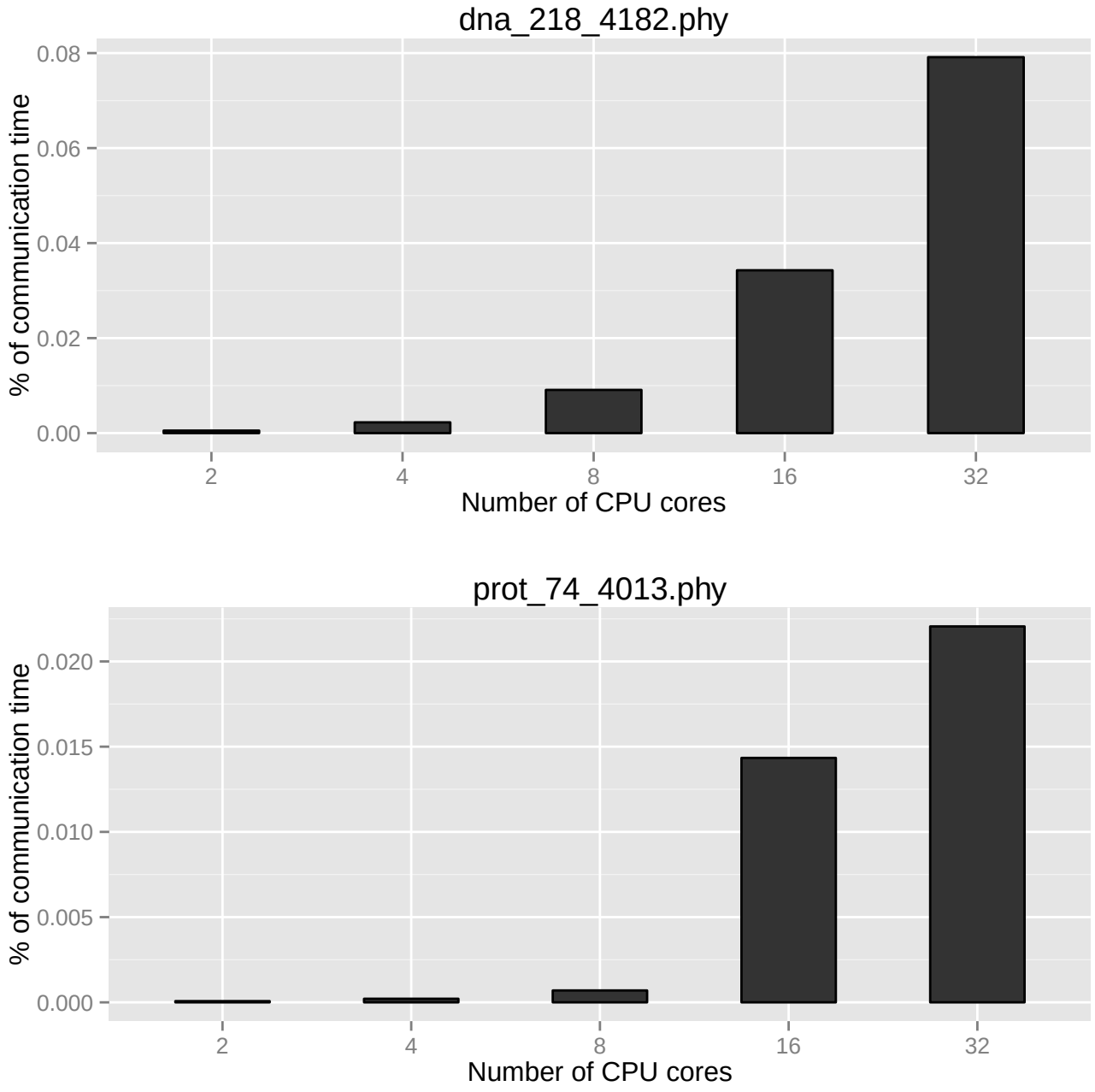

FIGURE 5.4: Percentage of communication time over the total running time

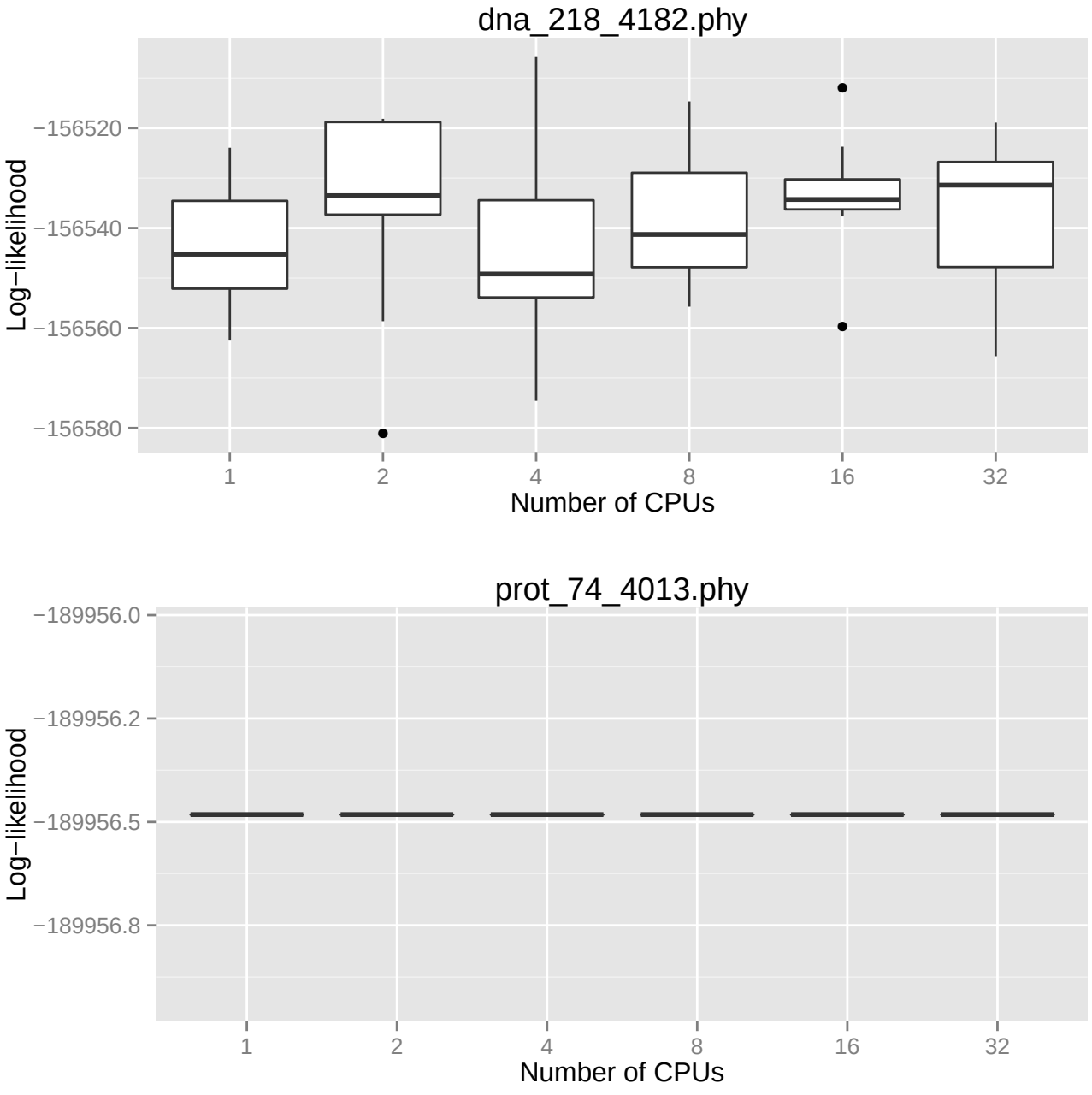

FIGURE 5.5: Comparison of tree log-likelihoods produced by IQ-TREE using different numbers of CPU cores. Each box plot displays the distribution of the log-likelihoods produced by ten replicate runs.

in accordance with previous studies showing that parallel evolutionary algorithm can be more effective than its sequential counterpart in solving combinatorial optimization problems (Alba and Tomassini, 2002).

We note that the performance of our parallel algorithm depends on the technical configuration of the underlying computer system. Since the memory consumption of IQ-TREE-MPI grows proportionally with the number of processes, care must be taken when running IQ-TREE-MPI on shared memory multi-core processors. For large alignments, the computer might run out of memory if too many processes are used. Even with sufficient main memory, runtime performance can still be negatively affected if a large portion of the cache memory is share among the processors. Moreover, as observed by Stamatakis and Ott (2008), using MPI on shared memory computers could result in degrading performance because many MPI implementations are not optimized for such systems. A hybrid parallelization might be most beneficial to shared memory machines. Therefore, we are planning to implement the hybrid parallelization in the next release of IQ-TREE.

Chapter 6

Summary

I have presented four contributions to the field of maximum-likelihood phylogenetic inference:

The IQ-TREE algorithm (Chapter 2)

I proposed the stochastic search algorithm IQ-TREE for the maximum likelihood framework. I showed that NNI tree rearrangement can be used effectively when coupled with a properly designed search strategy. IQ-TREE employs both up-hill and down-hill NNI moves to examine the tree space and at the same time avoid being stuck in local optima. Moreover, the search technique is embedded within an evolutionary algorithm so that the search space can be explored more efficiently. With extensive computational experiments, I show that IQ-TREE performs favorably other against state of the art methods (RAxML and PhyML), while requiring comparable computing time.

Accurate estimation of rate heterogeneity (Chapter 3)

I showed that conventional methods for estimating parameters of the rate heterogeneity model +I+Γ are not accurate. More specifically, the optimization routines implemented in most phylogenetic software fail to find the maximum likelihood estimates. Given the popularity of the +I+Γ model, having a resolution for the problem is crucial. To this end, I proposed an alternative optimization strategy that can effectively infer the correct parameters. While the use of complex models of sequence evolution is gaining momentum, my finding substantiates the importance of developing accurate estimation methods.

IQ-TREE-SP (Chapter 4)

The massive influx of biological data generated every year keeps pushing phylogenetic methods to the boundary. This fact motivated me to further improve the IQ-TREE search algorithm. Here, I extended the IQ-TREE algorithm with the heuristic IQ-TREE-SP. IQ-TREE-SP constraints the search space based on the concept of stable splits. It speeds up the tree search up to 3.9 times while delivering comparable or better results than the standard algorithm.

IQ-TREE-MPI (Chapter 5)

As multi-core and multi-processor computing systems are becoming ubiquitous, phylogenetic software need to be adapted to take advantage of the added computing power. Here, I developed IQ-TREE-MPI, a coarse-grained parallelization of the IQ-TREE algorithm. IQ-TREE-MPI shows very good scaling performance and also improves the search quality.

All methods described in this thesis were implemented in the phylogenetic software IQ-TREE, which can be freely downloaded at:

http://www.cibiv.at/software/iqtree

IQ-TREE is a command-line program supporting all three major operating systems: Linux, Windows and MAC OS X. We also provide an user-friendly web interface (Figure 6.1) available at:

http://iqtree.cibiv.univie.ac.at

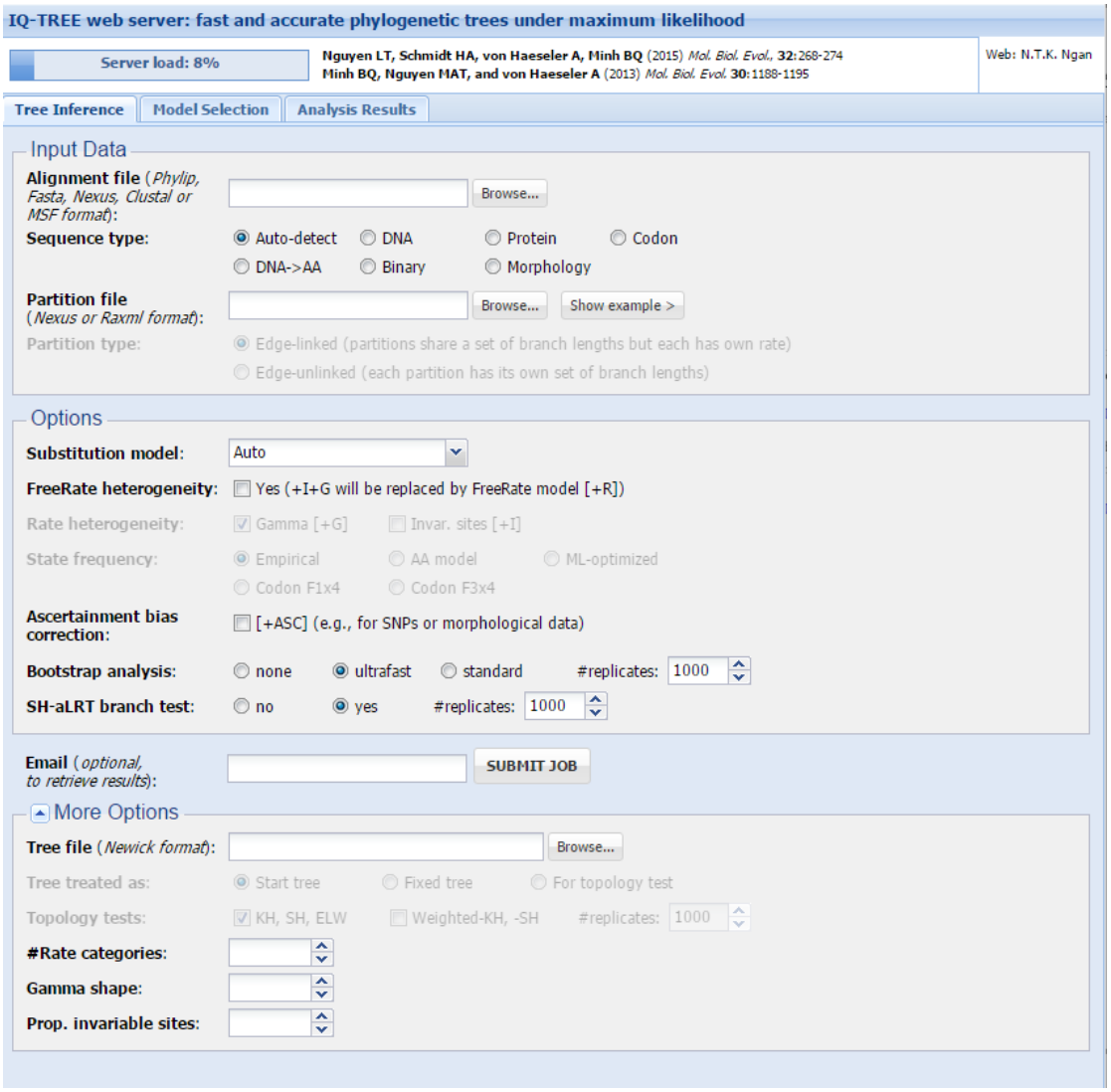

FIGURE 6.1: Screen shot of the IQ-TREE web interface.

Appendix A

Supplementary data for chapter 2

A.1 Supplementary tables

Table S1 TreeBASE DNA alignments and average log-likelihoods. In bold face are the highest log-likelihood found for each alignment.

TreeBASE ID	#Taxa	#Sites	% Gaps / ambiguous characters	IQ-TREE with default stopping rule	RAxML	IQ-TREE restricted to RAxML time	PhyML	IQ-TREE restricted to PhyML time
M214	295	1836	13.32	**-38,670.97**	-38,681.75	-38,678.08	-38,687.82	-38,679.68
M667	218	1002	4.67	**-45,048.88**	-45,052.75	-45,049.03	-45,060.45	**-45,048.88**
M980	249	1604	19.23	**-19,611.37**	-19,619.88	-19,614.52	-19,629.07	-19,613.97
M1110	330	1711	10.14	**-56,377.21**	-56,382.67	-56,377.39	-56,379.22	**-56,377.21**
M1224	210	8235	40.95	**-241,142.99**	-241,162.43	-241,143.12	-241,163.34	-241,144.92
M1838	228	1131	2.1	**-76,967.34**	-77,073.62	-76,971.09	-77,041.43	-76,972.60
M2307	318	1434	3.87	**-35,722.34**	-35,737.57	-35,726.57	-35,745.25	-35,724.83
M2534	207	976	10.23	-23,626.38	**-23,617.43**	-23,627.83	-23,665.68	-23,634.39
M2902	220	1117	1.14	-7,206.09	-7,212.21	-7,210.28	**-7,204.67**	-7,207.63
M2931	229	4722	23.7	**-54,290.51**	-54,310.48	-54,291.02	-54,294.23	-54,290.86
M3031	276	1518	14.47	**-20,237.53**	-20,249.97	-20,243.53	-20,266.95	-20,245.69
M3198	216	2578	11.09	**-147,473.15**	-147,498.71	-147,478.56	-147,483.62	-147,475.27
M3514	217	3665	41.73	**-141,010.38**	-141,030.79	-141,037.88	-141,063.13	-141,044.94
M3605	260	5315	34.05	**-55,863.43**	-55,870.86	-55,865.33	-55,876.41	-55,864.25
M3777	363	1707	3.44	-19,365.75	-19,378.47	-19,365.80	**-19,364.62**	-19,365.76
M4170	258	2559	5.51	**-16,585.71**	-16,602.40	-16,594.25	-16,592.69	-16,590.74
M4324	206	4543	25.32	**-117,989.19**	-118,026.61	-118,003.45	-118,098.31	-117,992.93
M4326	227	4055	9.73	**-29,395.83**	-29,411.69	-29,396.27	-29,400.38	-29,395.86
M4399	205	8913	57.25	**-59,907.83**	-59,915.50	-59,915.65	-59,918.08	-59,912.89
M4720	201	2899	30.81	**-53,732.74**	-53,744.97	**-53,732.74**	-53,734.79	**-53,732.74**
M4727	350	6464	52.26	**-206,129.75**	-206,208.32	-206,132.97	-206,150.81	-206,130.16
M4794	204	12113	52.47	**-148,388.80**	-148,434.07	-148,453.76	-148,669.35	-148,413.50

TreeBASE ID	#Taxa	#Sites	% Gaps / ambiguous characters	IQ-TREE with default stopping rule	RAxML	IQ-TREE restricted to RAxML time	PhyML	IQ-TREE restricted to PhyML time
M4806	222	3995	26.7	**-73,850.56**	-73,865.86	**-73,850.56**	-73,851.22	**-73,850.56**
M4850	328	2166	6.15	-23,425.32	-23,427.86	-23,425.57	-23,426.00	**-23,425.01**
M4900	206	3074	31.27	-62,389.10	-62,403.73	-62,389.11	**-62,388.47**	-62,389.10
M4927	268	1275	37.89	**-15,275.77**	-15,302.56	-15,289.19	-15,314.22	-15,288.21
M4938	213	1250	8.13	**-36,243.38**	-36,261.75	-36,245.11	-36,258.73	-36,244.48
M5078	265	9768	14.48	**-317,361.67**	-317,382.43	-317,361.69	-317,370.73	-317,362.15
M5235	298	11596	67.51	**-66,573.65**	-66,588.26	-66,580.11	-66,592.51	-66,577.79
M5381	413	3632	10.42	**-224,667.49**	-224,712.33	-224,685.74	-224,733.80	-224,668.63
M5731	242	9626	62.94	**-79,593.92**	-79,608.95	-79,594.80	-79,596.64	-79,594.50
M5931	298	4948	47.67	**-71,241.57**	-71,261.17	-71,251.78	-71,281.02	-71,250.53
M6134	219	5158	28.53	**-30,201.02**	-30,265.32	-30,228.78	-30,266.62	-30,222.20
M6414	209	2000	29.9	**-36,707.55**	-36,717.07	-36,709.87	-36,713.05	-36,709.22
M6625	247	1812	32.17	**-17,190.07**	-17,203.34	-17,193.05	-17,205.32	-17,193.46
M7024	767	5814	66.73	**-381,126.07**	-381,168.90	-381,161.16	-381,269.87	-381,135.11
M7054	222	6237	35.63	**-79,139.05**	-79,143.97	-79,143.80	-79,155.57	-79,141.71
M7165	357	4475	17.65	**-76,166.48**	-76,177.78	-76,168.62	-76,168.94	-76,167.32
M7210	204	1701	64.48	**-14,803.61**	-14,810.03	-14,804.95	-14,804.61	-14,804.49
M7211	201	1519	50.61	**-16,465.75**	-16,470.14	-16,465.85	-16,466.11	-16,466.29
M7292	213	7572	44.14	**-192,598.51**	-192,629.55	-192,601.89	-192,673.02	-192,604.30
M7929	428	15016	57.7	**-410,939.26**	-410,983.84	-410,949.34	-411,028.18	-410,945.26
M7964	640	25260	41.43	-1,327,204.51	-1,327,314.04	-1,327,210.58	-1,327,375.92	**-1,327,202.44**
M8012	213	2333	5.76	**-67,060.36**	-67,070.90	-67,060.78	-67,072.40	**-67,060.36**
M8385	212	19972	63.57	**-347,340.89**	-347,360.26	-347,376.10	-347,515.64	-347,353.57
M8619	246	11829	65.9	**-93,597.01**	-93,606.25	**-93,597.01**	-93,600.99	-93,597.29
M8692	395	3583	14.73	**-24,993.81**	-25,033.99	-25,014.37	-25,014.32	-25,009.85
M8703	215	1038	0.31	**-17,399.39**	-17,416.29	-17,399.58	-17,403.57	-17,399.41

TreeBASE ID	#Taxa	#Sites	% Gaps / ambiguous characters	IQ-TREE with default stopping rule	RAxML	IQ-TREE restricted to RAxML time	PhyML	IQ-TREE restricted to PhyML time
M8975	405	3027	1.78	**-92,430.53**	-92,454.00	-92,431.41	-92,525.99	-92,431.59
M8982	297	6954	42.72	**-141,323.67**	-141,349.66	**-141,323.67**	-141,328.39	**-141,323.67**
M8984	201	3931	29.43	**-21,997.97**	-22,004.58	-21,999.10	-22,005.05	-21,999.24
M9033	300	1394	7.1	**-23,867.40**	-23,871.18	-23,869.91	-23,869.90	-23,870.76
M9142	235	1854	9.07	**-18,985.30**	-18,985.57	-18,989.29	-18,987.18	-18,992.48
M9143	228	1223	11.52	-8,917.67	-8,917.71	-8,923.28	**-8,911.35**	-8,917.71
M9915	504	2757	54.26	-483,690.03	**-483,684.95**	-483,720.07	-483,795.81	-483,738.35
M10243	203	1771	0.4	-16,344.85	-16,348.02	-16,344.91	**-16,344.57**	-16,344.86
M10434	544	5681	7.2	**-139,349.11**	-139,362.30	-139,352.27	-139,363.43	-139,352.25
M10467	202	4074	45.87	**-91,948.98**	-91,962.16	-91,949.77	-91,952.53	-91,949.00
M10933	229	2696	22.58	**-94,316.14**	-94,342.72	-94,323.67	-94,362.75	-94,320.81
M11113	344	9778	15.7	**-487,985.38**	-488,032.50	-487,987.70	-488,019.47	-487,992.29
M11745	316	1494	49.58	**-19,004.08**	-19,014.98	-19,005.73	-19,015.87	-19,004.84
M11762	208	2468	0.06	-20,050.65	-20,061.46	-20,050.65	**-20,048.61**	-20,050.65
M12051	699	6914	40.96	-384,988.31	-385,007.09	-385,032.28	-385,081.21	**-384,987.48**
M12098	231	4108	38.01	**-129,824.04**	-129,830.33	-129,824.09	-129,841.30	-129,824.07
M12388	324	1405	42.11	**-9,609.98**	-9,620.77	-9,610.81	-9,612.25	-9,610.15
M13718	235	2309	37.25	-100,203.82	-100,215.25	-100,206.13	-100,224.21	**-100,202.73**
M14164	204	5549	59.37	**-40,356.16**	-40,367.22	-40,358.50	-40,371.17	-40,357.91
M14165	204	5611	41.84	**-144,455.80**	-144,476.21	-144,458.86	-144,467.93	-144,456.17
M14582	372	61199	68.64	**-656,241.64**	-656,332.90	-656,243.70	-656,312.55	-656,241.69
M14678	225	2673	25.51	**-47,166.74**	-47,182.26	-47,167.72	-47,174.37	-47,167.41

Table S2 TreeBASE protein alignments and average log-likelihoods. In bold face are the highest log-likelihood found for each alignment.

TreeBASE ID	#Taxa	#Sites	% Gaps / ambiguous characters	IQ-TREE with default stopping rule	RAxML	IQ-TREE restricted to RAxML time	PhyML	IQ-TREE restricted to PhyML time
M510	57	430	9.38	**-8,164.14**	-8,165.75	**-8,164.14**	-8,175.26	-8,164.17
M1118	137	348	26.31	**-12,837.91**	-12,840.76	-12,838.72	-12,845.97	-12,839.32
M1726	50	1000	37.18	**-68,153.56**	-68,154.33	-68,153.61	-68,167.93	-68,154.15
M2358	55	714	20.72	**-14,670.25**	-14,670.31	**-14,670.25**	-14,677.96	**-14,670.25**
M2593	56	386	2.98	**-8,945.24**	-8,948.11	-8,945.25	-8,948.04	-8,948.29
M2926	105	899	43.08	**-85,026.88**	-85,026.90	-85,026.90	-85,027.77	-85,026.94
M3113	77	9918	55.33	**-361,280.00**	-361,297.06	-361,281.64	-361,347.19	-361,294.78
M3114	77	11234	55.77	-434,506.92	**-434,504.13**	-434,519.03	-434,614.68	-434,525.05
M3807	82	591	41.38	**-38,706.56**	-38,709.34	-38,706.87	-38,722.00	-38,707.97
M3810	55	271	21.31	**-11,061.54**	-11,061.76	**-11,061.54**	-11,061.55	-11,063.98
M4249	153	455	45.47	**-57,701.68**	-57,704.83	-57,702.29	-57,718.75	-57,703.21
M4318	78	2295	3.97	**-185,004.73**	**-185,004.73**	**-185,004.73**	-185,058.96	-185,005.08
M4325	73	230	1.05	**-21,106.15**	**-21,106.15**	-21,106.15	-21,112.61	-21,106.55
M4539	59	12428	19.66	**-473,611.99**	-473,622.19	**-473,611.99**	-473,662.13	-473,622.59
M4780	90	583	45.96	**-45,725.34**	-45,726.37	-45,725.36	-45,742.72	-45,726.26
M4860	62	11544	18.61	**-426,418.62**	**-426,418.62**	**-426,418.62**	-426,418.65	**-426,418.62**
M4884	50	11827	34.80	**-242,169.69**	**-242,169.69**	**-242,169.69**	-242,174.90	-242,173.32
M5379	60	7776	13.91	-279,270.38	**-279,270.36**	-279,270.39	-279,299.37	-279,282.01
M6416	57	126	7.10	**-5,181.98**	-5,182.79	-5,182.93	-5,186.26	-5,184.03
M7078	77	12457	13.04	**-379,457.82**	-379,463.94	-379,458.13	-379,477.72	-379,459.36
M7729	62	2973	29.31	**-110,353.19**	-110,361.23	-110,353.19	-110,379.37	-110,353.20
M8175	194	665	29.40	-26,504.23	-26,506.99	-26,504.27	**-26,503.94**	-26,504.89
M8461	89	5699	29.89	**-307,038.70**	-307,047.10	-307,040.16	-307,126.42	-307,056.90

TreeBASE ID	#Taxa	#Sites	% Gaps / ambiguous characters	IQ-TREE with default stopping rule	RAxML	IQ-TREE restricted to RAxML time	PhyML	IQ-TREE restricted to PhyML time
M8569*	164	383	44.67	**-40,594.84**	-40,595.64	-40,595.27	-40,595.81	-40,596.53
M8630	50	21154	0.22	**-588,675.92**	-588,675.92	**-588,675.92**	-589,298.44	-588,831.51
M9973	60	327	3.42	**-13,077.17**	-13,077.98	-13,077.19	-13,077.18	-13,077.26
M10236	59	164	62.23	**-5,113.01**	-5,113.90	-5,113.93	-5,117.14	-5,114.20
M10273	169	11009	17.16	**-650,541.12**	**-650,541.12**	**-650,541.12**	-650,541.14	**-650,541.12**
M10372	169	22426	15.93	**-1,359,736.89**	-1,359,736.90	**-1,359,736.89**	-1,359,796.74	-1,359,736.90
M10866	88	3329	3.26	-150,463.11	**-150,463.11**	-150,463.11	-150,463.12	-150,463.11
M11012	55	11500	19.55	**-495,773.15**	-495,787.73	-495,778.73	-495,824.51	-495,778.26
M11013	55	8741	19.14	**-251,347.86**	**-251,347.86**	**-251,347.86**	-251,347.87	**-251,347.86**
M11335	99	696	27.19	**-41,474.35**	-41,475.04	-41,474.41	-41,499.34	-41,476.80
M11336	95	268	0.00	**-14,847.24**	-14,849.76	**-14,847.24**	-14,847.24	-14,853.97
M11338	100	567	48.18	**-32,213.69**	-32,217.12	-32,215.33	-32,251.85	-32,217.06
M11341	100	699	26.36	**-53,103.85**	-53,105.13	-53,104.67	-53,131.42	-53,113.88
M11342	91	323	0.00	**-24,050.26**	-24,050.42	**-24,050.26**	-24,056.54	**-24,050.26**
M11344	84	691	50.36	**-42,681.83**	-42,683.91	-42,681.85	-42,687.96	-42,683.34
M11595	66	463	12.87	**-27,727.92**	-27,727.92	**-27,727.92**	**-27,727.92**	**-27,727.92**
M11596	62	439	17.46	**-34,244.66**	**-34,244.66**	**-34,244.66**	-34,244.66	**-34,244.66**
M11740	138	4427	2.62	-376,122.31	-376,125.26	**-376,121.96**	-376,125.96	-376,128.09
M12103	97	199	0.00	**-17,447.60**	-17,448.40	-17,449.18	-17,448.93	-17,459.79
M12104	93	349	0.00	**-16,139.84**	-16,144.58	-16,140.05	-16,144.77	-16,145.06
M12376	116	327	37.23	**-20,453.87**	-20,454.54	-20,453.88	-20,455.30	-20,453.91
M13804	78	1889	16.89	**-113,431.86**	-113,433.54	**-113,431.86**	-113,439.18	**-113,431.86**

* We removed the sequence *Homo_sapiens_GSTA3* from the original multiple sequence alignment because it was obviously unaligned.

A.2 Supplementary figures

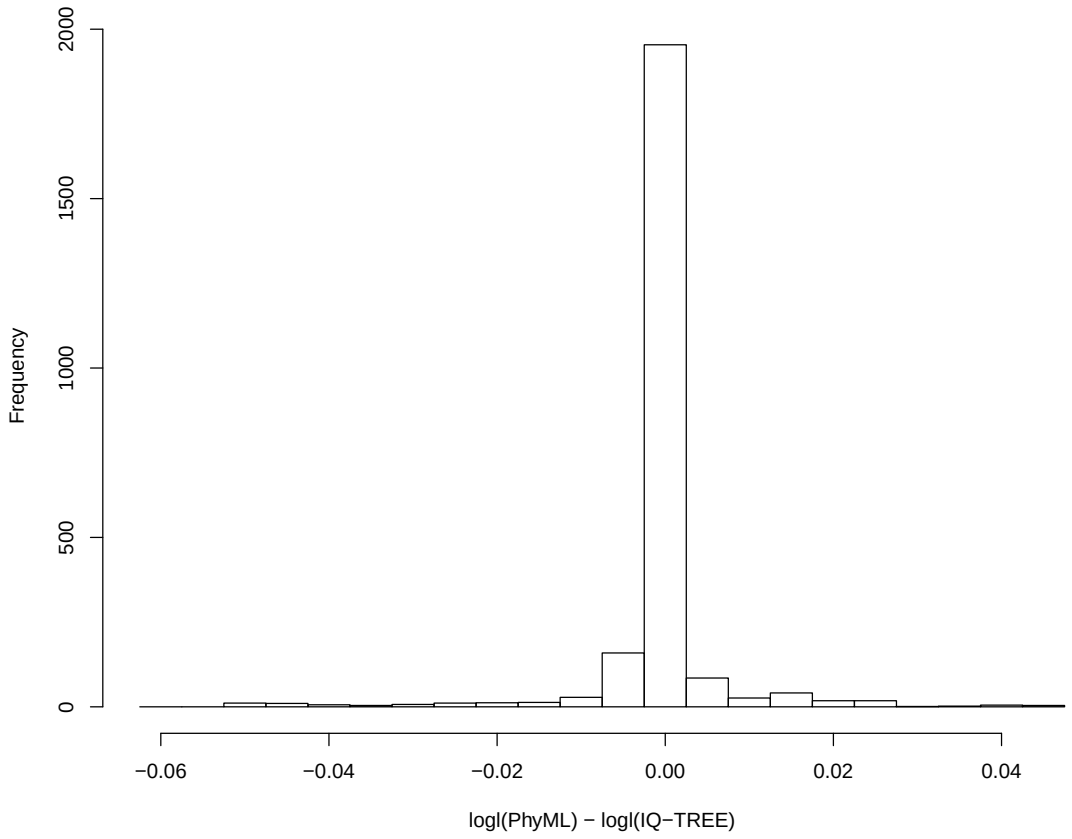

FIGURE A.1: Distribution of differences between log-likelihood computed by PhyML and IQ-TREE for the same tree and model parameters. Trees and model parameters are taken from results used in Figure 2.3

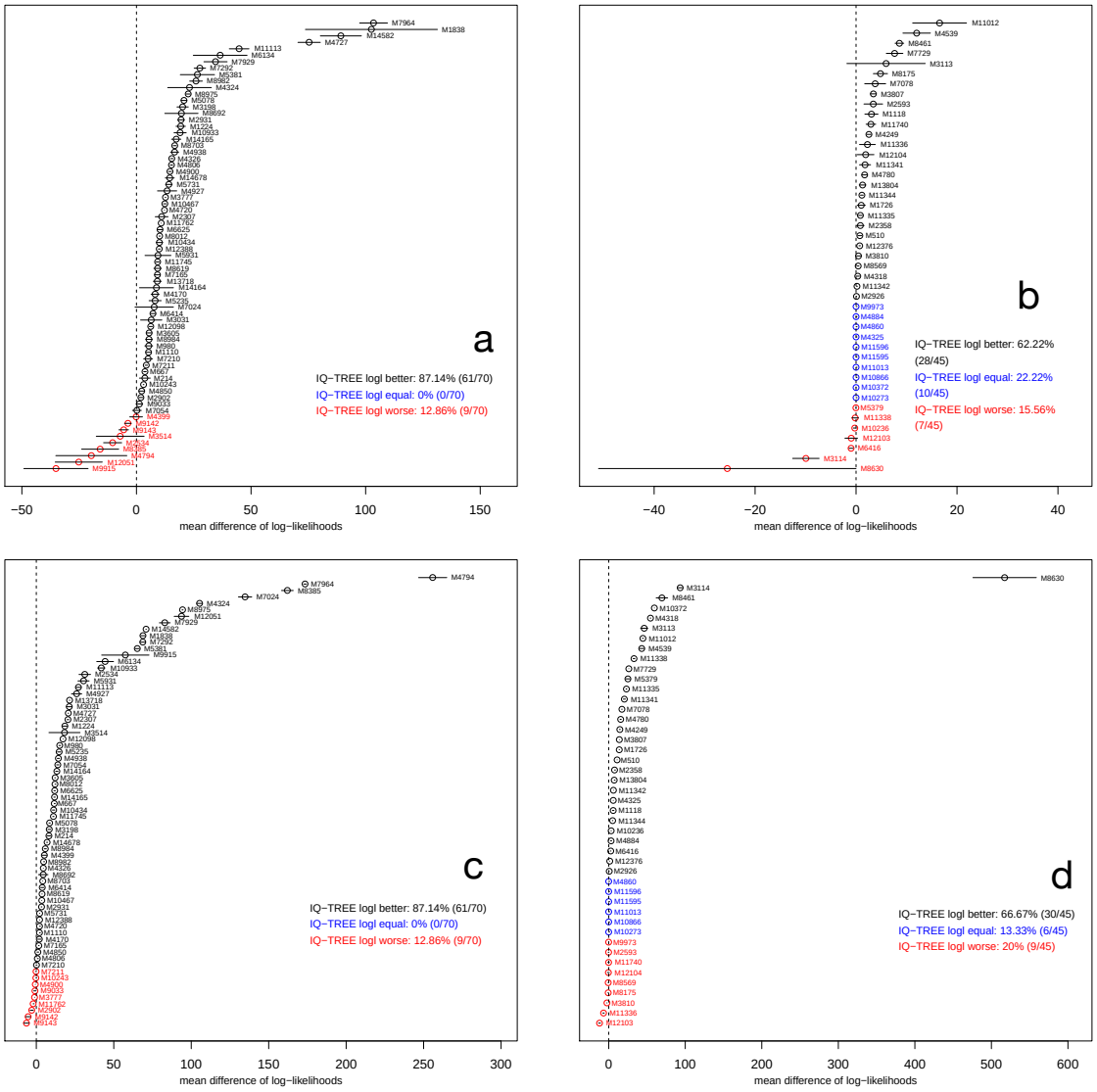

FIGURE A.2: Log-likelihood differences for IQ-TREE with equal CPU times as RAxML times (a,b) or PhyML times (c,d): Subfigure a and b show the log-likelihood differences of IQ-TREE against RAxML using the 70 DNA and 45 AA alignments. Subfigure c and d show the same against PhyML. The alignments are ordered by the average log-likelihood differences. The whiskers at each point show the standard errors of the differences. If the log-likelihood differences are smaller than 0.01 the results are regarded as being equal. Subfigure a to d relate to the subfigure a to d in Figure 2.2 in the main text.

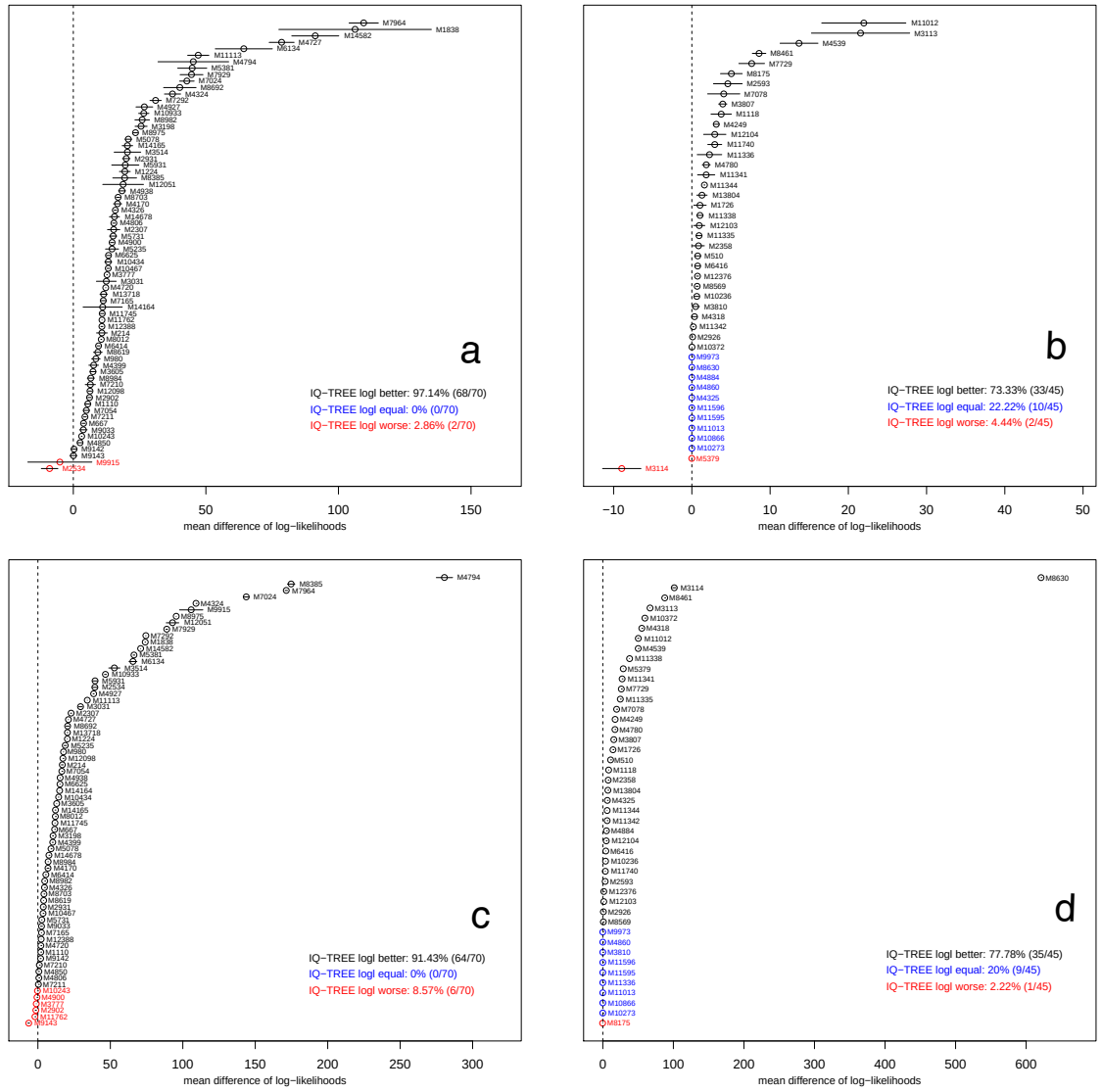

FIGURE A.3: Log-likelihood differences for IQ-TREE with CPU times determined by its stopping rule: Subfigure a and b show the log-likelihood differences of IQ-TREE against RAxML using the 70 DNA and 45 AA alignments. Subfigure c and d show the same against PhyML. The alignments are ordered by the average log-likelihood differences. The whiskers at each point show the standard errors of the differences. If the log-likelihood differences are smaller than 0.01 the results are regarded as being equal. Subfigure a to d relate to the subfigures in Figure 2.3 in the main text.

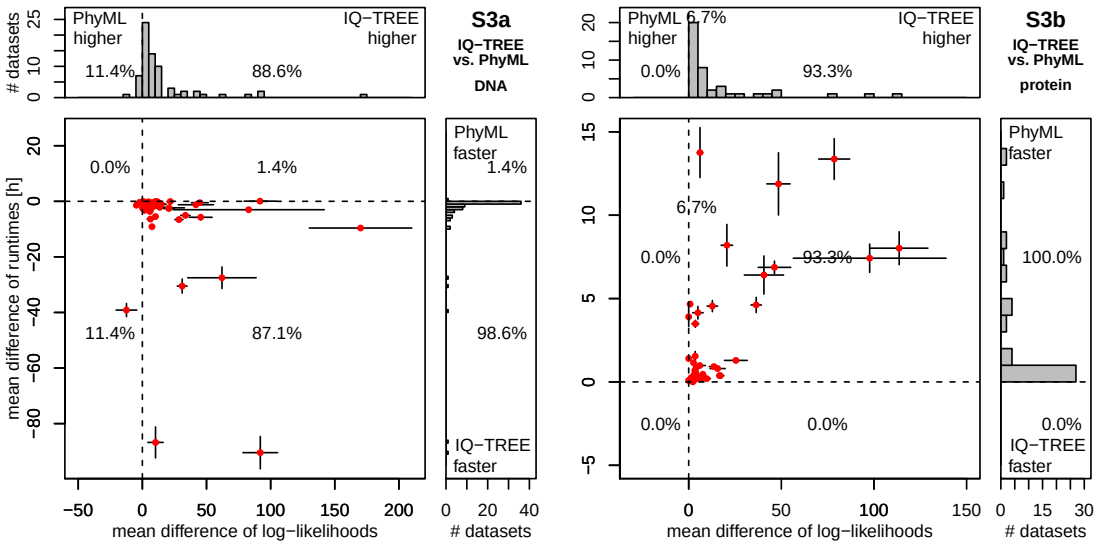

FIGURE A.4: Performance of IQ-TREE versus PhyML with random starting trees for 70 DNA (a) and 45 AA (b) alignments. Each dot in the main diagrams represents for one alignment the mean differences of the CPU times (y-axis) and of the mean differences of log-likelihoods (x-axis) of the reconstructed trees by the programs compared. The whiskers at each point show the standard errors of the differences. The histograms at the top and the side present the marginal frequencies. Dots to the right of the vertical dashed line represent alignments where IQ-TREE found a higher likelihood. If a dot is below the horizontal dashed line the reconstruction by IQ-TREE was faster. Percentages in the quadrants of histograms denote the fraction of alignments in that region. Percentages on the dashed line reflect the number of alignments where log-likelihood differences are smaller than 0.01. To allow this analysis (`--rand_start --n_rand_start=1 --s spr`) the source code of PhyML had to be fixed to perform and output a proper tree search from a random starting tree.

Bibliography

Alba, E. and Tomassini, M. (2002) Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, **6**, 443–462.

Allman, E. S. and Rhodes, J. A. (2008) Identifying evolutionary trees and substitution parameters for the general markov model with invariable sites. *Mathematical Biosciences*, **211**, 18 – 33.

Ayres, D. L., Darling, A., Zwickl, D. J., Beerli, P., Holder, M. T., Lewis, P. O., Huelsenbeck, J. P., Ronquist, F., Swofford, D. L., Cummings, M. P., Rambaut, A. and Suchard, M. A. (2012) BEAGLE: An application programming interface and high-performance computing library for statistical phylogenetics. *Systematic Biology*, **61**, 170–173.

Bouchenak-Khelladi, Y., Salamin, N., Savolainen, V., Forest, F., van der Bank, M., Chase, M. W. and Hodkinson, T. R. (2008) Large multi-gene phylogenetic trees of the grasses (poaceae): progress towards complete tribal and generic level sampling. *Molecular Phylogenetics and Evolution*, **47**, 488–505.

Caceres, A., Castillo, J., Lee, J. and St.John, K. (2013) Walks on SPR neighborhoods. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, **10**, 236–239.

Chai, J. and Housworth, E. A. (2011) On Rogers' proof of identifiability for the GTR+ Γ+ I model. *Systematic Biology*, **60**, 713–718.

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)*, volume 3500 of *Lecture Notes in Computer Science*, pp. 296–310, ACM Press, New York, USA.

Crick, F. *et al.* (1970) Central dogma of molecular biology. *Nature*, **227**, 561–563.

Darwin, C. (1859) *On the origin of species by means of natural selection: or the preservation of favoured races in the struggle for life*. John Murray.

Dell'Ampio, E., Meusemann, K., Szucsich, N. U., Peters, R. S., Meyer, B., Borner, J., Petersen, M., Aberer, A. J., Stamatakis, A., Walzl, M. G. *et al.* (2014) Decisive data sets in phylogenomics: lessons from studies on the phylogenetic relationships of primarily wingless insects. *Molecular Biology and Evolution*, **31**, 239–249.

Dobzhansky, T. (1973) Nothing in biology makes sense except in the light of evolution. *The American Biology Teacher*, **35**, 125–129.

Edgar, R. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, **32**, 1792.

Farris, J. S. (1970) Methods for computing Wagner trees. *Systematic Biology*, **19**, 83–92.

Felsenstein, J. (1978a) Cases in which parsimony or compatibility methods will be positively misleading. *Systematic Biology*, **27**, 401–410.

Felsenstein, J. (1978b) The number of evolutionary trees. *Systematic Biology*, **27**, 27–33.

Felsenstein, J. (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, **17**, 368–376.

Felsenstein, J. (1985) Confidence limits on phylogenies: An approach using the bootstrap. *Evolution*, **39**, 783–791.

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

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

Fitch, W. M. and Margoliash, E. (1967) A method for estimating the number of invariant amino acid coding positions in a gene using cytochrome c as a model case. *Biochemical Genetics*, **1**, 65–71.

Flouri, T., Izquierdo-Carrasco, F., Darriba, D., Aberer, A., Nguyen, L.-T., Minh, B., Von Haeseler, A. and Stamatakis, A. (2015) The phylogenetic likelihood library. *Systematic Biology*, **64**, 356–362.

Foster, I. (1995) *Designing and building parallel programs.* Addison Wesley Publishing Company.

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

Gill, P. E., Murray, W. and Wright, M. H. (1981) *Practical optimization.* Academic press.

Glover, F. (1989) Tabu search - Part I. *ORSA Journal on computing*, **1**, 190–206.

Grenfell, B. T., Pybus, O. G., Gog, J. R., Wood, J. L., Daly, J. M., Mumford, J. A. and Holmes, E. C. (2004) Unifying the epidemiological and evolutionary dynamics of pathogens. *Science*, **303**, 327–332.

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

Guindon, S., Dufayard, J.-F., Lefort, V., Anisimova, M., Hordijk, W. and Gascuel, O. (2010) New algorithms and methods to estimate maximum-likelihood phylogenies: Assessing the performance of PhyML 3.0. *Systmatic Biology*, **59**, 307–321.

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

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

Hall, B. G. (2005) Comparison of the accuracies of several phylogenetic methods using protein and dna sequences. *Molecular Biology and Evolution*, **22**, 792–802.

Harris, J. W. and Stöcker, H. (1998) *Handbook of mathematics and computational science.* Springer-Verlag New York.

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

Helaers, R. and Milinkovitch, M. C. (2010) MetaPIGA v2. 0: maximum likelihood large phylogeny estimation using the metapopulation genetic algorithm and other stochastic heuristics. *BMC Bioinformatics*, **11**, 379.

Hinchliff, C. E. and Roalson, E. H. (2013) Using supermatrices for phylogenetic inquiry: an example using the sedges. *Systematic Biology*, **62**, 205–219.

Hordijk, W. and Gascuel, O. (2005) Improving the efficiency of SPR moves in phylogenetic tree search methods based on maximum likelihood. *Bioinformatics*, **21**, 4338–4347.

Jia, F., Lo, N. and Ho, S. Y. W. (2014) The impact of modelling rate heterogeneity among sites on phylogenetic estimates of intraspecific evolutionary rates and timescales. *PLoS ONE*, **9**, e95722.

Jin, L. and Nei, M. (1990) Limitations of the evolutionary parsimony method of phylogenetic analysis. *Molecular Biology and Evolution*, **7**, 82–102.

Jukes, T. H. and Cantor, C. R. (1969) Evolution of protein molecules. *Mammalian Protein Metabolism*, **3**, 21–132.

Katoh, K., Misawa, K., Kuma, K.-i. and Miyata, T. (2002) MAFFT: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Research*, **30**, 3059–3066.

Kimura, M. (1980) A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, **16**, 111–120.

L. L. Cavalli-Sforza, A. W. F. E. (1967) Phylogenetic analysis: Models and estimation procedures. *Evolution*, **21**, 550–570.

Lanave, C., Preparata, G., Sacone, C. and Serio, G. (1984) A new method for calculating evolutionary substitution rates. *Journal of Molecular Evolution*, **20**, 86–93.

Lewis, P. O. (1998) A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. *Molecular Biology and Evolution*, **15**, 277–283.

Li, W.-H., Gouy, M., Sharp, P. M., O'hUigin, C. and Yang, Y.-W. (1990) Molecular phylogeny of rodentia, lagomorpha, primates, artiodactyla, and carnivora and

molecular clocks. *Proceedings of the National Academy of Sciences*, **87**, 6703–6707.

Lin, S. and Kernighan, B. W. (1973) An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, **21**, 498–516.

van der Linde, K., Houle, D., Spicer, G. S. and Steppan, S. J. (2010) A supermatrix-based molecular phylogeny of the family drosophilidae. *Genetics Research*, **92**, 25–38.

Lourenco, H. R., Martin, O. C. and Stützle, T. (2003) Iterated local search. In Glover, Fred W., K. G. A. (ed.), *Handbook of Metaheuristics*, pp. 320–353, Springer US.

Löytynoja, A. and Goldman, N. (2008) Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis. *Science*, **320**, 1632–1635.

Mayrose, I., Friedman, N. and Pupko, T. (2005) A gamma mixture model better accounts for among site rate heterogeneity. *Bioinformatics*, **21**, 151–ii158.

McCormack, J. E., Harvey, M. G., Faircloth, B. C., Crawford, N. G., Glenn, T. C. and Brumfield, R. T. (2013) A phylogeny of birds based on over 1,500 loci collected by target enrichment and high-throughput sequencing. *PLoS One*, **8**, e54848.

Meyer, S. and von Haeseler, A. (2003) Identifying site-specific substitution rates. *Molecular Biology and Evolution*, **20**, 182–189.

Minh, B. Q., Nguyen, M. A. T. and von Haeseler, A. (2013) Ultrafast approximation for phylogenetic bootstrap. *Molecular Biology and Evolution*, **30**, 1188–1195.

Minh, B. Q., Vinh, L. S., Von Haeseler, A. and Schmidt, H. A. (2005) pIQPNNI: parallel reconstruction of large maximum likelihood phylogenies. *Bioinformatics*, **21**, 3794–3796.

Misof, B., Liu, S., Meusemann, K., Peters, R. S., Donath, A., Mayer, C., Frandsen, P. B., Ware, J., Flouri, T., Beutel, R. G. *et al.* (2014) Phylogenomics resolves the timing and pattern of insect evolution. *Science*, **346**, 763–767.

Money, D. and Whelan, S. (2012) Characterizing the phylogenetic tree-search problem. *Systematic Biology*, **61**, 228–239.

Morrison, D. A. (2007) Increasing the efficiency of searches for the maximum likelihood tree in a phylogenetic analysis of up to 150 nucleotide sequences. *Systematic Biology*, **56**, 988–1010.

Needleman, S. B. and Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, **48**, 443–453.

Nguyen, L.-T., Schmidt, H. A., von Haeseler, A. and Minh, B. Q. (2015) IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular Biology and Evolution*, **32**, 268–274.

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

Nyakatura, K. and Bininda-Emonds, O. R. (2012) Updating the evolutionary history of carnivora (mammalia): a new species-level supertree complete with divergence time estimates. *BMC Biology*, **10**, 12.

Olsen, G. (1987) Earliest phylogenetic branchings: comparing rrna-based evolutionary trees inferred with various techniques. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 52, pp. 825–837, Cold Spring Harbor Laboratory Press.

Palumbi, S. R. (1989) Rates of molecular evolution and the fraction of nucleotide positions free to vary. *Journal of Molecular Evolution*, **29**, 180–187.

Pfeiffer, W. and Stamatakis, A. (2010) Hybrid mpi/pthreads parallelization of the raxml phylogenetics code. In *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, pp. 1–8.

Posada, D. and Crandall, K. A. (1998) Modeltest: testing the model of dna substitution. *Bioinformatics*, **14**, 817–818.

Press, W. H. (2007) *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press.

Pyron, R. A., Burbrink, F. T., Colli, G. R., De Oca, A. N. M., Vitt, L. J., Kuczynski, C. A. and Wiens, J. J. (2011) The phylogeny of advanced snakes (colubroidea), with discovery of a new subfamily and comparison of support

methods for likelihood trees. *Molecular Phylogenetics and Evolution*, **58**, 329–342.

Rabenseifner, R. (2003) Hybrid parallel programming on HPC platforms. In *Proceedings of the Fifth European Workshop on OpenMP, EWOMP*, volume 3, pp. 185–194.

Rambaut, A. and Grass, N. C. (1997) Seq-Gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Computer applications in the biosciences: CABIOS*, **13**, 235–238.

Rechenberg, I. (1973) *Evolutionsstrategie Optimierung technischer systeme nach prinzipien der biologischen evolution*. Friedrich Frommann Verlag.

Rogers, J. S. (2001) Maximum likelihood estimation of phylogenetic trees is consistent when substitution rates vary according to the invariable sites plus gamma distribution. *Systematic Biology*, **50**, 713–722.

Ronquist, F., Teslenko, M., van der Mark, P., Ayres, D. L., Darling, A., Höhna, S., Larget, B., Liu, L., Suchard, M. A. and Huelsenbeck, J. P. (2012) MrBayes 3.2: efficient bayesian phylogenetic inference and model choice across a large model space. *Systematic Biology*, **61**, 539–542.

Rzhetsky, A. and Nei, M. (1992) A simple method for estimating and testing minimum-evolution trees. *Molecular Biology and Evolution*, **9**, 945.

Saitou, N. and Nei, M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, **4**, 406–425.

Salipante, S. J. and Horwitz, M. S. (2006) Phylogenetic fate mapping. *Proceedings of the National Academy of Sciences*, **103**, 5448–5453.

Salter, L. A. and Pearl, D. K. (2001) Stochastic search strategy for estimation of maximum likelihood phylogenetic trees. *Systematic Biology*, **50**, 7–17.

Sanderson, M., Donoghue, M., Piel, W. and Eriksson, T. (1994) Treebase: a prototype database of phylogenetic analyses and an interactive tool for browsing the phylogeny of life. *American Journal of Botany*, **81**, 183.

Smith, T. F. and Waterman, M. S. (1981) Identification of common molecular subsequences. *Journal of Molecular Biology*, **147**, 195–197.

Springer, M. S., Meredith, R. W., Gatesy, J., Emerling, C. A., Park, J., Rabosky, D. L., Stadler, T., Steiner, C., Ryder, O. A., Janečka, J. E., Fisher, C. A. and Murphy, W. J. (2012) Macroevolutionary dynamics and historical biogeography of primate diversification inferred from a species supermatrix. *PLoS ONE*, **7**, e49521.

Stamatakis, A. (2006) Raxml-vi-hpc: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, **22**, 2688–2690.

Stamatakis, A. (2014) Raxml version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, **30**, 1312–1313.

Stamatakis, A. (2015) RAxML manual version 8.2.x. *Heidelberg Institute for Theoretical Studies (Distributed by the author).*

Stamatakis, A. and Alachiotis, N. (2010) Time and memory efficient likelihood-based tree searches on phylogenomic alignments with missing data. *Bioinformatics*, **26**, i132–i139.

Stamatakis, A. and Ott, M. (2008) Exploiting fine-grained parallelism in the phylogenetic likelihood function with mpi, pthreads, and openmp: a performance study. In *Pattern Recognition in Bioinformatics*, pp. 424–435, Springer.

Strimmer, K. and von Haeseler, A. (1996) Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies. *Molecular Biology and Evolution*, **13**, 964–969.

Strimmer, K. and von Haeseler, A. (2009) Genetic distances and nucleotide substitution models. In Lemey, P., Salemi, M. and Anne-Mieke, V. (eds.), *The Phylogenetic Handbook: a Practical Approach to Phylogenetic Analysis and Hypothesis Testing*, 2nd edition, pp. 111–141, Cambridge University Press, Cambridge.

Sullivan, J. and Swofford, D. L. (1997) Are guinea pigs rodents? the importance of adequate models in molecular phylogenetics. *Journal of Mammalian Evolution*, **4**, 77–86.

Sullivan, J., Swofford, D. L. and Naylor, G. J. (1999) The effect of taxon sampling on estimating rate heterogeneity parameters of maximum-likelihood models. *Molecular Biology and Evolution*, **16**, 1347–1356.

Swofford, D. L. (2003) *PAUP*: Phylogenetic analysis using parsimony (* and other methods). Version 4*. Sinauer Associates.

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*, 2nd edition, pp. 407–514, Sinauer Associates.

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. *Molecular Biology and Evolution*, **11**, 261–277.

Tavaré, S. (1986) Some probabilistic and statistical problems on the analysis of DNA sequences. In Miura, R. M. (ed.), *DNA Sequence Analysis (Proceedings of the 1984 Symposium Some Mathematical Questions in Biology)*, volume 17 of *Lectures on Mathematics in the Life Sciences*, pp. 57–86, American Mathematical Society, Providence, Rhode Island, USA.

Thompson, J., Higgins, D. and Gibson, T. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, **22**, 4673.

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

Vandamme, A.-M. (2003) Basic concept of molecular evolution. In Lemey, P., Salemi, M. and Anne-Mieke, V. (eds.), *The Phylogenetic Handbook: a Practical Approach to Phylogenetic Analysis and Hypothesis Testing*, 1st edition, pp. 1–23, Cambridge University Press, Cambridge.

Vinh, L. S. and von Haeseler, A. (2004) IQPNNI: Moving fast through tree space and stopping in time. *Molecular Biology Evolution*, **21**, 1565–1571.

Vos, R. (2003) Accelerated likelihood surface exploration: the likelihood ratchet. *Systematic Biology*, **52**, 368–373.

Waddell, P. J. and Steel, M. A. (1997) General time-reversible distances with unequal rates across sites: mixing γ and inverse gaussian distributions with invariant sites. *Molecular Phylogenetics and Evolution*, **8**, 398–414.

Wang, L. and Jiang, T. (1994) On the complexity of multiple sequence alignment. *Journal of Computational Biology*, **1**, 337–348.

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

Whelan, S. and Money, D. (2010) The prevalence of multifurcations in tree-space and their implications for tree-search. *Molecular Biology and Evolution*, **27**, 2674–2677.

Yang, Z. (1993) Maximum-likelihood estimation of phylogeny from dna sequences when substitution rates differ over sites. *Molecular Biology and Evolution*, **10**, 1396–1401.

Yang, Z. (1994) Maximum likelihood phylogenetic estimation from dna sequences with variable rates over sites: approximate methods. *Journal of Molecular evolution*, **39**, 306–314.

Yang, Z. (1996) Among-site rate variation and its impact on phylogenetic analyses. *Trends in Ecology & Evolution*, **11**, 367–372.

Yang, Z. (2000) Maximum likelihood estimation on large phylogenies and analysis of adaptive evolution in human influenza virus a. *Journal of molecular evolution*, **51**, 423–432.

Yang, Z. (2006) *Computational molecular evolution*. Oxford University Press.

Yang, Z. (2014) *Molecular evolution: a statistical approach*. Oxford University Press.

Yang, Z. and Rannala, B. (2012) Molecular phylogenetics: principles and practice. *Nature Reviews Genetics*, **13**, 303–314.

Zwickl, D. J. (2006) *Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion*. Ph.D. thesis, University of Texas, Austin, USA.

Curriculum Vitae

Contact information

Lam-Tung Nguyen
Center for Integrative Bioinformatics Vienna (CIBIV)
Max F.Perutz Laboratories
Dr.Bohr Gasse 9
A-1030 Vienna, Austria
Email: tung.nguyen@univie.ac.at

Research interests

- Phylogenetic Inference.
- Optimization and Metaheuristics.
- High Performance Computing.

Education

- 1995-2002 : Hanoi-Amsterdam High School, *class for gifted students in mathematics*.
- 2002-2003 : Hanoi University of Science and Technology, *Computer Science* (discontinued).

- 2004-2008: Vienna University of Technology, *Software & Information Engineering (Computer Science)*.
  Degree: Bachelor of Science.

- 2008-2011: Vienna University of Technology, *Information & Knowledge Management (Computer Science)*.
  Degree: Diplom-Ingenieur (Master of Science).

Professional Experience

- 6.2011-present: Center for Integrative Bioinformatics Vienna, *Research Associate*.

- 2008-2011: Center for Integrative Bioinformatics Vienna, *Scientific Programmer*.

- 2009-6.2011: Base19 - Homes for Students, *Computer Administrator*.

- 2006-2008: Vienna University of Economics and Business, *Tutor*.

- 3.2006-7.2006: Siemens A.G, *Intern*.

Publications

- L.-T. Nguyen, H.A. Schmidt, A. von Haeseler, and B.Q. Minh (2014) *IQ-TREE: A fast and effective stochastic algorithm for estimating maximum likelihood phylogenies*. Molecular Biology and Evolution.

- T. Flouri, F. Izquierdo-Carrasco, D. Darriba, A.J. Aberer, L.-T. Nguyen, B.Q. Minh, A. von Haeseler, and A. Stamatakis (2014) *The phylogenetic likelihood library*. Systematic Biology.

- L.-T. Nguyen, A. von Haeseler, and B.Q. Minh (2016) *Complex models of sequence evolution require accurate estimator as exemplified with the invariable site plus Gamma model* (in preperation).

- Tobias S. Kaiser, Birgit Poehn, David Szkiba, Marco Preussner, Fritz J. Sedlazeck, Alexander Zrim, Tobias Neumann, Lam-Tung Nguyen, Andrea

J. Betancourt, Thomas Hummel, Heiko Vogel, Silke Dorner, Florian Heyd, Arndt von Haeseler, Kristin Tessmar-Raible (2015) *Insight into the adaptive evolution of circadian and circalunar timing from the genome of a marine insect.* Nature (in review).