UNIVERSITÀ DEGLI STUDI DI TRIESTE

XXXVI CICLO DEL DOTTORATO DI RICERCA IN

INGEGNERIA DELL'INFORMAZIONE

PHYLOES An Evolution Strategies approach for phylogenetic inference based on the Balanced Minimum Evolution Criterion

Settore scientifico-disciplinare: **MAT/09 – RICERCA OPERATIVA**

DOTTORANDO
Andrea Gasparin [illegible]

COORDINATORE
PROF. Fulvio Babich [illegible]

SUPERVISORE DI TESI
PROF. Lorenzo Castelli [illegible]

CO-SUPERVISORE DI TESI
PROF. Daniele Catanzaro [illegible]

ANNO ACCADEMICO 2022/2023

Contents

Abstract

The *Balanced Minimum Evolution* (BME) is a powerful distance based phylogenetic estimation model introduced by Desper and Gascuel and nowadays implemented in popular tools for phylogenetic analyses. It was proven to be computationally less demanding than more sophisticated estimation methods, e.g. maximum likelihood or Bayesian inference while preserving the statistical consistency and the ability to run with almost any kind of data for which a dissimilarity measure is available. BME can be stated in terms of a nonlinear non-convex combinatorial optimisation problem, usually referred to as the *Balanced Minimum Evolution Problem* (BMEP). Currently, the state-of-the-art among approximate methods for the BMEP is represented by FastME (version 2.0), a software which implements several deterministic phylogenetic construction heuristics combined with a local search on specific neighbourhoods derived by classical topological tree rearrangements. These combinations, however, may not guarantee convergence to close-to-optimal solutions to the problem due to the lack of solution space exploration, a phenomenon which is exacerbated when tackling molecular datasets characterised by a large number of taxa.

To overcome such convergence issues, in this work we propose a novel metaheuristic, named PhyloES, which exploits the combination of an exploration phase based on Evolution Strategies, a special type of evolutionary algorithm, with a refinement phase based on two local search algorithms. Extensive computational experiments show that PhyloES consistently outperforms FastME, especially when tackling larger datasets, providing solutions characterised by a shorter tree length but also significantly different from the topological perspective.

Abstract

Il *Balanced Minimum Evolution* (BME) è un potente modello di stima filogenetica basato sulle distanze biologiche fra i genomi, introdotto da Desper e Gascuel e attualmente implementato in efficienti software per le analisi filogenetiche. Il BME è computazionalmente meno impegnativo rispetto a metodi di stima più sofisticati, ad esempio Maximum Likelyhood o inferenza bayesiana, pur conservando la coerenza statistica e la capacità di funzionare con quasi qualsiasi tipo di dati che rappresentano una misura di dissimilarità fra genomi. Il BME può essere formulato in termini di un problema di ottimizzazione combinatoria non lineare e non convessa, denominato *Balanced Minimum Evolution Problem* (BMEP). Attualmente, lo stato dell'arte tra i metodi approssimati per il BMEP è rappresentato da FastME (versione 2.0), un software che implementa diverse euristiche deterministiche di costruzione filogenetica, combinate con algoritmi di ricerca locale basati sulla manipolazione di alberi filogenetici. Tuttavia, questo approccio, sebbene molto efficiente, non garantisce la convergenza a soluzioni prossime all'ottimo, a causa della mancanza di esplorazione dello spazio delle soluzioni, un fenomeno che si aggrava quando si affrontano insiemi di dati molecolari caratterizzati da un grande numero di taxa.

Per ovviare a tali problemi di convergenza, in questo lavoro, proponiamo una nuova metaeuristica, denominata PhyloES, che sfrutta la combinazione di una fase esplorativa, basata su una particolare tipologia di algoritmo evolutivo, denominata Evolution Strategies, con una fase di raffinamento incentrata su due algoritmi di ricerca locale. I nostri esperimenti computazionali mostrano che PhyloES è in grado di superare le prestazioni di FastME, specialmente al crescere delle dimensioni delle istanze, fornendo soluzioni migliori nel senso della funzione obiettivo e significativamente diverse dal punto di vista topologico, fattore di particolare rilevanza dal punto di vista biologico.

Chapter 1

Introduction

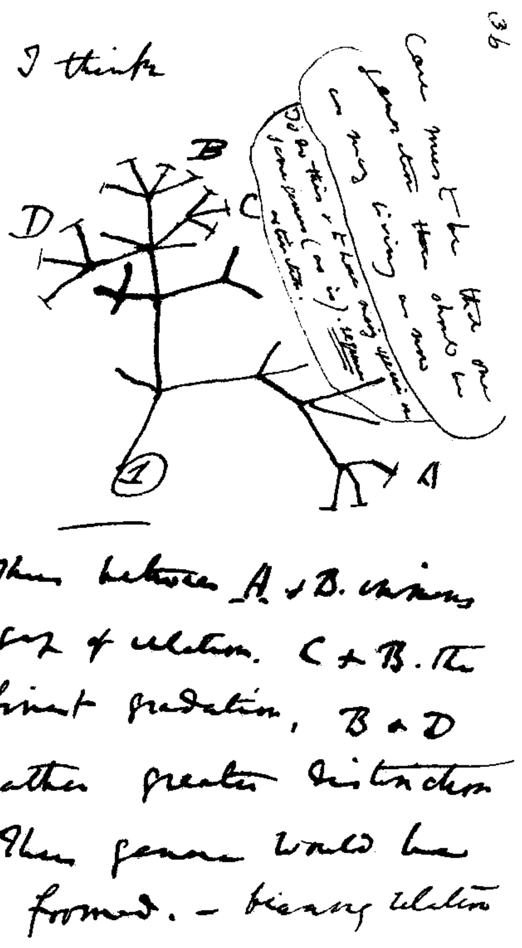
Phylogenetics, the study of evolutionary relationships among organisms, stands as an indispensable cornerstone of modern biology. It aims to study life's interconnectedness by investigating the ancestral connections and evolutionary trajectories that have shaped the diversity of living organisms on Earth. Rooted in the Greek words *phylon* (meaning tribe or ancestry) and *genesis* (representing origin or birth), phylogenetics tries to unearth the ancestral ties binding species.

Charles Darwin from his observations on the phenotypic variation of finches in the Galapagos to the publication of his seminal work *On the Origin of Species* [Darwin, 1859], laid the theoretical groundwork for the concept of a *tree of life*, (see Figure 1.1): the idea for which all organisms evolved from one common ancestor lived approximatively 3.7 billion years ago, (Pearce et al. [2018], Rosing [1999], Ohtomo et al. [2013]). Darwin's curiosity about the genealogical relationships among species pushed the study of phylogenetics to the scientific forefront.

To explore the evolution and connections between genes and living beings, various types of data can be employed. Traditionally, species relationships have been determined by comparing their morphological characters, (Linnaei [1758]). However, with the growing availability of molecular data like nucleotide and amino acid sequences, as well as restriction fragment length polymorphisms (RFLPs), scientists can now deduce phylogenetic relationships using molecular approaches. The use of molecular data to construct phylogenetic trees has gained significant traction across various biological disciplines, often complementing morphological data for a more comprehensive analysis of relationships, as they emerged as game-changing tools that allowed scientists to explore previously inaccessible aspects of the history of life on earth, as emphasised by Nei and Kumar [2000].

In fact, when dealing with extinct species, obtaining molecular data is challenging or impossible, making the reliance on morphological characteristics from mummies or fossils the primary means of estimating their relationships. On the other hand, organisms like viruses do not leave behind fossil records, necessitating the study of their past through the phylogenetic relationships of existing viruses.

By offering fundamental insights into the intricate evolutionary patterns found in various detailed molecular datasets, phylogenies offer crucial support across a wide spectrum of research domains, and their significance extends far beyond the academic community. Their application areas span from systematics, which is the study of the diversification of living forms, both past and present, and the relationships among living things through time, to medical research, encompassing drug discovery, epidemiology, ecology, biodiversity assessment, the analysis of population dynamics, conservation biology, and agriculture. In the realm of medicine, phylogenetics plays a pivotal role in tracing the spread of infectious diseases and uncovering the origins of drug resistance.



Case must be that one generation then should be as many living as now. To do this & to have many species in same genus (as is) requires extinction.

Thus between A & B. immense gap of relation. C & B. the finest gradation, B & D rather greater distinction Thus genera would be formed. — bearing relation

Figure 1.1: Charles Darwin's 1837 sketch, his first diagram of an evolutionary tree from his First Notebook on Transmutation of Species (1837) on view at the Museum of Natural History in Manhattan.
Side text: Case must be that one generation then should be as many living as now. To do this & to have many species in same genus (as is) requires extinction.
Bottom text: *Thus between A & B immense gap of relation. C & B the finest gradation, B & D rather greater distinction. Thus genera would be formed. — bearing relation*

As an illustration of their versatile applications in the medical field, phylogenies have been employed in various ways. These include predicting the evolution of human influenza A (Bush et al. [1999]), comprehending the relationships between the virulence and evolution of HIV (ECastro-Nallar et al. [2012]), detecting emerging viruses like SARS Amiroch et al. [2017], exploring ancestral proteins (Chang and Donoghue [2000]), the evolution of several common human diseases (Pennington et al. [2007]), and in particular carcinomas (Myers et al. [2019]).

Alongside its well-known application to human evolution, phylogenetics helped to uncover parallels in the evolution of multiple languages (Jäger [2018]).

Conservation biology relies on phylogenetics to guide strategies for preserving biodiversity by identifying evolutionarily distinct and vulnerable species (Isaac et al. [2007]). In agriculture, phylogenetics aids in crop improvement and the development of disease-resistant varieties (Piquerez et al. [2014]).

Furthermore, the impact of phylogenetics reverberates across disciplines concerned with climate change and ecosystem dynamics. By reconstructing the evolutionary histories of species within ecosystems, scientists can better predict their responses to changing environmental conditions and human-induced perturbations (Webb et al. [2002]).

This introductory foray into phylogenetics only scratches the surface of this expansive and multidisciplinary field, but we hope to have given a flavor of its importance in modern science.

1.1 Phylogenetic inference

As mentioned earlier, phylogenetics aims at reconstructing the evolutionary relationship between different organisms, usually named in this context **taxa** (sing. *taxon*), and whether possible the history of the underlying process. The natural representation of such a phenomenon among genes and organisms is achieved through the construction of phylogenetic trees, or phylogenies, as they visually convey the proximity of genes or organisms. Notably, the terminologies employed in these diagrams, such as **root**, **branch**, **node**, and **leaf**, evoke similarities to the components of trees and they are largely used in this context.

The external nodes, often referred to as *leaves*, represent taxa. This generic term encompasses a wide array of comparable entities, including families of organisms, individual specimens, or different strains of viruses within a species. **Internal nodes** represent instead the conjectured ancestors. The arrangement of nodes within the tree and its branching pattern is named **topology**.

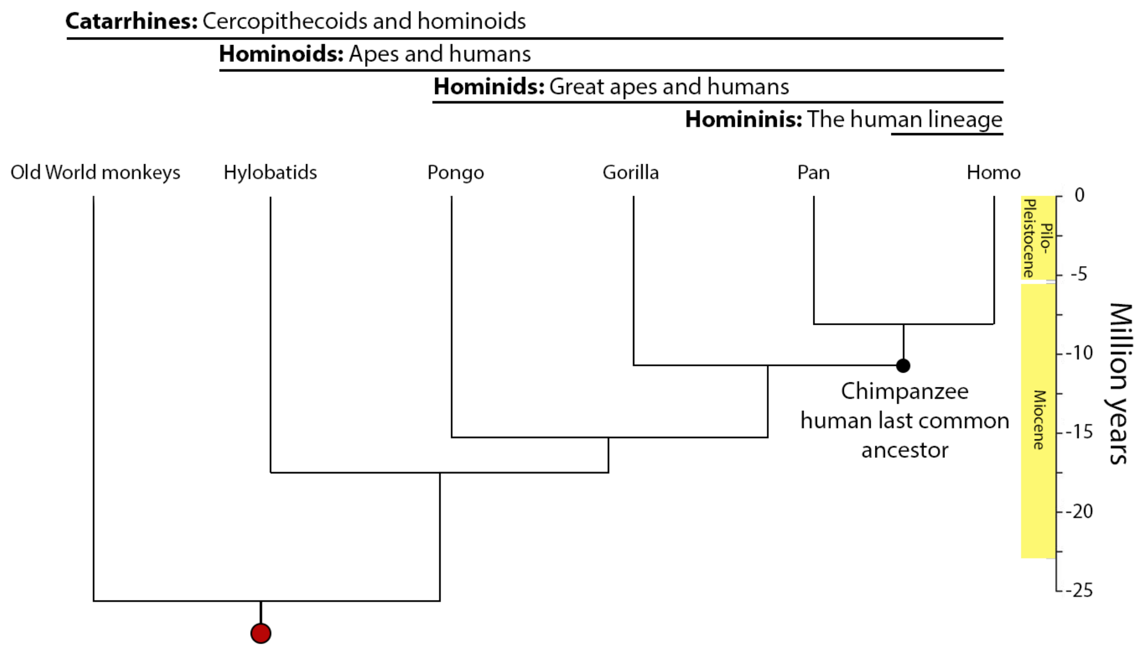
When a group of taxa shares a common branch, indicating a common ancestral origin, they form a *cluster*, which from the taxonomy point of view might suggest the existence of a biological group, such as species, genus, family, order, class, phylum, kingdom or domain (Cain [2023]).

In most applications and in accordance with classical evolutionary theory, internal nodes are also interpreted as *bifurcation* points of the tree, meaning the point in time in which due to both random mutations and a selective pressure from a common ancestor a new form of organism has born, (Darwin [1859]). Even though there are phenomena in nature, such as the explosive evolutionary radiation of HIV or HCV, that might be best represented by a multifurcating tree (Klug et al. [2003]), based on the fact that any *m-ary* tree can be transformed into a bifurcating tree by just adding "dummy edges", (Catanzaro et al. [2022]), and since they represent exceptions to the standard way of representing evolutionary among organisms (Lemey et al. [2009]), for the purpose of this work we will assume to always work with bifurcating trees.

Now that we have introduced all the required elements we can give a broad definition of phylogenetic inference: *given a set of taxa determine the tree's topology, its evolutionary hierarchy and estimate the branches' lengths which represent the evolutionary distance within individuals.*

If enough information about the taxa is provided it is sometimes possible to achieve all three tasks, as in the case of ape's evolution depicted in Figure 1.2. Here, the integration of molecular data with morphological and biomechanical information about extinct taxa allowed the estimation of the tree shape and the taxa arrangements as well as the bifurcating events dating, defining in this way the hierarchical structure between ancestors, and providing a time measurement of the evolutionary distances, which determine the tree branches' length.

It is important to note that in this case, the diversity of the data sources plays an essential role in inferring the hierarchical structure between taxa: the tree is in fact rooted (red node), meaning



Catarrhines: Cercopithecoids and hominoids
Hominoids: Apes and humans
Hominids: Great apes and humans
Homininis: The human lineage
Old World monkeys
Hylobatids
Pongo
Gorilla
Pan
Homo
Chimpanzee human last common ancestor
Plio-Pleistocene
Miocene
0
-5
-10
-15
-20
-25
Million years

Figure 1.2: Evolution of apes, reconstruction from the study of Almécija et al. [2021]

that we are able to establish which was the first speciation event from the oldest ancestor, common to all taxa; consequently, we are also able to deduce the direction of evolution in the tree.

However, such an accurate and complete analysis is not always straightforward nor possible when only molecular data are available. In fact, when taxa are simply represented with their molecular sequence, i.g. nucleotides (DNA or RNA) or amino acid (proteins) sequences, although several techniques to identify the most ancient common ancestor have been developed (Lemey et al. [2009]), the chronological reconstruction of the evolution process is extremely difficult. In these cases, the phylogeny is usually encoded as an unrooted binary tree (see Figure 1.3), in which all taxa are potential roots, and the reconstruction of its topology, which will be the focus of this work, represents the only (but still hard) achievable task. When molecular data is the unique source of information, the concept of evolutionary distance cannot be directly interpreted in a chronological sense, as the only possible measurement available is the differences between the taxa sequences; so the branch lengths in this case measure the genetic differences within individuals (Lemey et al. [2009]).

In this respect, the use of the so-called *substitution models* stands as a pivotal cornerstone. These models provide a quantitative framework for understanding the processes of genetic sequence evolution, delineating how nucleotides or amino acids undergo mutation and replacement over evolutionary time.

A genetic sequence is encoded as a vector of characters, in which each entry represents a distinct nucleotide or amino acid, providing the essential framework for the genetic information that governs an organism's structure, function, and evolution.

The notion of substitution models has its roots in the fundamental concept that genetic sequences accumulate changes over time, driven by a combination of mutation, selection, and random genetic drift. These models are essential to translate raw genetic data into meaningful insights about the evolutionary history of species (Holland [2015]). By capturing and quantifying the patterns of

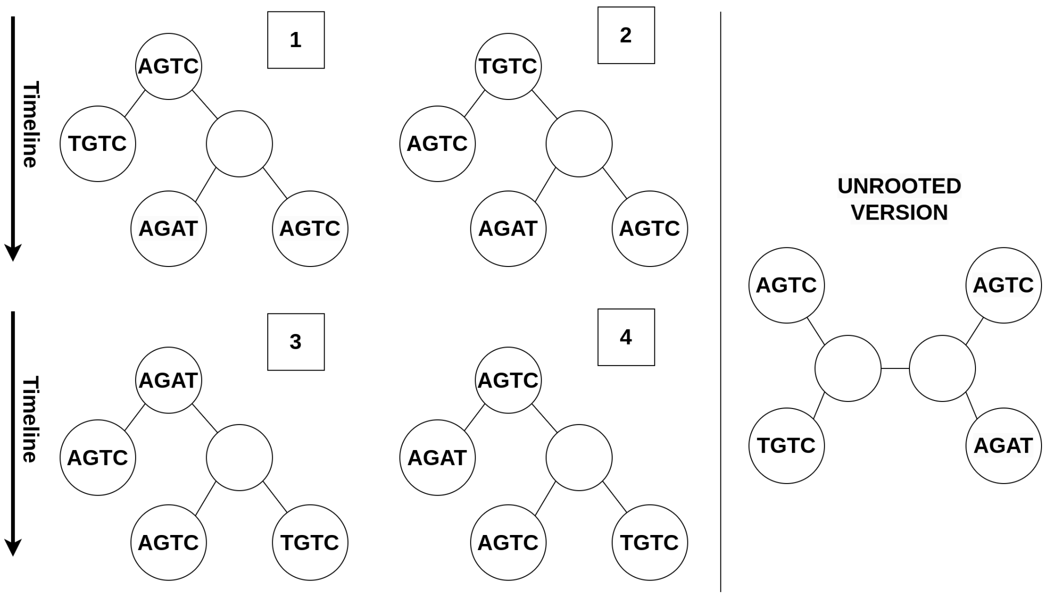


Figure 1.3: Example of the reconstruction of a phylogeny of four taxa: AGTC, TGTC, AGAT, AGCT. The characters A, G, T, and C represent the nucleotides that compose their DNA sequence, and they stand for adenine, guanine, thymine, and cytosine respectively. For the sake of clarity, here we assume to describe the differences between taxa simply by counting the number of character changes needed to obtain one from another. Under this framework, we can say that AGTC and TGTC are "close" since they share 3 out of the 4 characters of their code, and the same consideration holds for AGAT and AGCT. The pairs AGTC-AGAT, AGTC-AGCT, TGTC-AGAT, and TGTC-AGCT instead, share only 2 characters so we could say that they look slightly more different, or "distant". If we further assume that the phylogeny we are looking for is the one characterised by the minimum number of character changes required along its branches to explain the observed states at the terminal nodes (see *Lex pharsimonie*, Section 1.2), we have that trees 1, 2, 3, and 4 represent a plausible reconstruction of the evolutionary process. However, without any additional evidence for preferring one over the others, it is not possible to determine which taxon is the most ancient common ancestor. For this reason, in such cases, the evolutionary process is generally represented with an unrooted tree, which still preserves the relative evolutionary structure between taxa, without suggesting any preference for a particular taxon as the common ancestor of all the others. We can note in fact, that by "unrooting" trees 1, 2, 3, and 4 we obtain the same tree unrooted tree, shown in the right side of the figure.

molecular change, substitution models empower researchers to make informed inferences about the relatedness of species, the rates of molecular evolution, and the mechanisms governing sequence variation (Tavaré [1986]).

They can range from simple, time-homogeneous models like the one proposed by (Jukes and Cantor [1969]), to more complex time-structured models like the GTR (General Time Reversible) model, (Tavaré [1986]). Each of these models carries its own assumptions and mathematical representations to describe the rates and patterns of nucleotide or amino acid substitution and its choice should be carefully made based on the biological characteristic of the dataset under study (Tamura and Nei [1993]).

The field of substitution models has witnessed continuous evolution and refinement, reflecting both advances in computational capabilities and our growing knowledge of molecular evolution. The development of more realistic and flexible models, coupled with sophisticated statistical techniques, has enabled the extraction of deeper insights from genetic data.

A comprehensive understanding of substitution models is out of the scope of this work as it would require a deep analysis of the various types and complexities of these models. Still, it is important to notice that they play a crucial role in phylogenetic inference, representing the bridge between the raw data to the numerical quantities that are at the core of the main phylogenetic reconstruction methods.

1.2 Main phylogenetic inference methods

The estimation of a phylogeny given a set of molecular data is a problem that has been tackled via a wide spectrum of methods, from clustering to optimisation. In this subsection, we provide a brief overview of the main and most common ones.

The first method we describe is the Maximum Parsimony (MP). MP is a character-based method, meaning that it operates directly with the sequences. It draws inspiration from the *Lex parsimoniae*, also known as *Occam's razor principle* in the scientific literature, which is a form of abduction widely acknowledged in modern scientific methodology (Popper [2005]). This principle is firmly rooted in the concept of falsifiability and posits that when faced with multiple competing theories to explain a particular observed phenomenon, one should favour theories that necessitate the fewest assumptions.

Drawing inspiration from the underlying philosophy of this principle, Farris [1970] and Fitch [1971] laid the theoretical foundation for the parsimony criterion in the realm of phylogenetic estimation (Albert [2005]). The MP goal is to determine the most likely tree structure for a given set of aligned sequences while minimizing the number of character changes. When considering a particular tree structure, the MP algorithm deduces the minimum number of character changes necessary for each position within the sequences to account for the observed states at the terminal nodes. The cumulative count of these changes across all positions is termed the "parsimony length" of the tree. This parsimony length is computed for various tree structures, and after assessing a sufficient number of options, the tree requiring the fewest changes is chosen as the Maximum Parsimony tree.

Substitution models can be deployed to estimate evolution probabilities, a fact which is exploited by the two following methods.

Maximum likelihood (ML), (Felsenstein [2004]), shares similarities with the Maximum Parsimony method in its approach to examining various tree topologies and assessing their score according to the given criterion. More precisely, ML algorithms are designed to search for the tree that maximizes the likelihood of observing the data (i.e. the sequences), given a substitution model. When dealing with a particular tree, the likelihood computation involves summing over all possible nucleotide (or amino acid) states in the ancestral (internal) nodes. To achieve this, numerical optimization techniques are applied to identify the combination of branch lengths and evolutionary parameters that yield the highest likelihood. Since the set of possible trees is generally extremely large, depending on the search algorithm employed, the likelihood is computed for a range of tree topologies using this criterion, and the tree that provides the greatest likelihood is selected as the optimal tree. Unfortunately, the process of determining the likelihood of a given tree can be computationally intensive.

Bayesian Inference methods (BI), (Huelsenbeck et al. [2001], Huelsenbeck et al. [2002]), while sharing the notion of character-state analysis and an optimality criterion with MP and ML, exhibit a distinct conceptual difference. Unlike MP and ML, which aim to identify a single best tree, Bayesian methods diverge in their approach. They delve into the realm of probability distributions, searching for a multitude of plausible trees (or hypotheses) that may explain the observed data. This collection of potential trees, known as the *posterior distribution* of trees, inherently yields a measure of confidence in evolutionary relationships. To employ Bayesian methods, a prior *belief* must be specified. Such belief is a distribution encompassing model parameters such as substitution model parameters, branch lengths, and tree topology. This prior belief serves as an *a priori* understanding of the system and data play the role of guiding the update of the prior belief. The determination of posterior probabilities involves traversing the space of potential trees using a sampling technique known as Markov chain Monte Carlo (MCMC). The output of Bayesian methods is usually presented in the form of summary statistics derived from the samples. For continuous parameters, this may involve the mean or median, while for trees, it often results in the presentation of a consensus tree or a maximum *a posteriori* tree. Bayesian methods are useful in providing information about uncertainty, making them particularly robust. On the other hand, they are generally computationally extremely intensive.

The last family of phylogenetic methods we consider is the one so-called *Distance-based methods*, which represents a well-consolidated theoretical and algorithmic framework to carry out practical phylogenetic analyses. They are characterised by a measure of the dissimilarity, or distance, of each pair of taxa to produce a pairwise distance matrix D, which represents the input data used to infer the phylogenetic relationships of the taxa. Distance-matrix methods usually make use of a substitution model to compute the dissimilarities between taxa. The main idea is that the distance between two taxa grows the more their sequences are divergent, as it is likely that in the evolutionary process that led from one to the other, multiple consecutive mutations have occurred.

These methods are typically based on hypotheses and assumptions that are considerably simpler than those at the core of more sophisticated estimation methods such as ML or BI, a fact that might make the ML and BI poor at modelling complex evolutionary processes (Schwartz [2019], Catanzaro et al. [2022]). Another advantage of distance methods is the fact that they can run with almost any kind of data for which a dissimilarity measure is available and prove to be computationally less demanding than more sophisticated estimation methods based on ML and BI.

The Neighbor-Joining (NJ) method proposed by Saitou and Nei [1987], constructs a tree through a sequential process that identifies pairs of neighboring taxa. Neighbours are characterized by being linked by a single internal node. Given a particular branch length function, the NJ's objective is to minimize the length of all internal branches, thereby reducing the overall length of the entire tree. The NJ algorithm initiates with the assumption of a star-like tree in which all taxa are linked to a single internal node. At each step, the algorithm identifies the pair of leaves with the smallest genetic distance. These two leaves are considered "neighbours" and are joined together as a new single (leaf) node in the tree, which represents their common ancestor. The matrix D is then updated to include the newly joined node and delete the selected leaves, which implies the recalculation of the genetic distances from this node to all other leaves. The process is iteratively carried out until no further leaves joining is possible.

Example 1 (Distance computation) *Let us consider the problem described in Figure 1.4 in which taxa $t_1, \ldots, t_5$ (fictional for simplicity) are defined by their DNA sequence. One of the simplest evolutionary distance is the so-called Alignment-Free Eucledian, which is computed by interpreting each sequence as a text composed of words, where each word is defined by a tuple, in this example of length two, of contiguous characters.*

Considering the set of all words appearing in the sequences, one can then construct the vectors $\bar{t}_1, \ldots, \bar{t}_5$ which are counting the occurrences of each word in each sequence, as described in 1.5.

Finally, the Euclidean distance between each pair of taxa t_i, t_j can be computed with the usual formula:

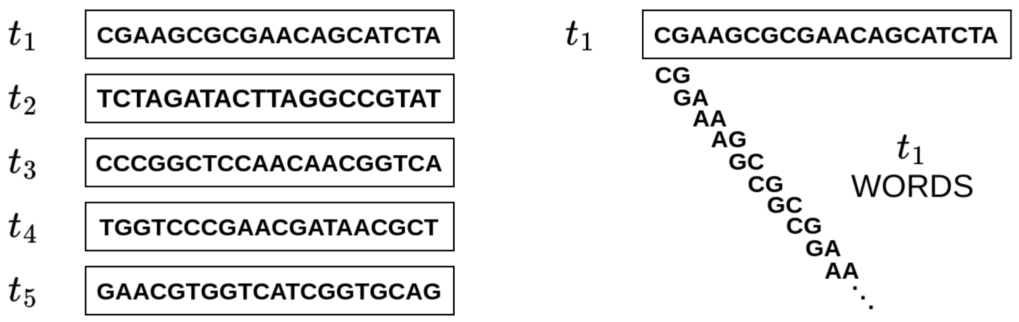
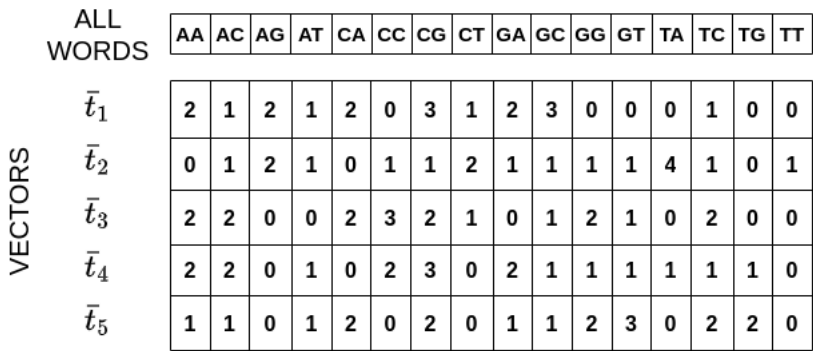

t_1 CGAAGCGCGAACAGCATCTA
t_2 TCTAGATACTTAGGCCGTAT
t_3 CCCGGCTCCAACAACGGTCA
t_4 TGGTCCCGAACGATAACGCT
t_5 GAACGTGGTCATCGGTGCAG
t_1 CGAAGCGCGAACAGCATCTA
CG
GA
AA
AG
GC
CG
GC
CG
GA
AA
t_1 WORDS

Figure 1.4: LEFT. Five DNA sequences defining the respective taxa. RIGHT. Example of word-segmentation of the t_1 sequence.



ALL WORDS	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
$\bar{t}_1$	2	1	2	1	2	0	3	1	2	3	0	0	0	1	0	0
$\bar{t}_2$	0	1	2	1	0	1	1	2	1	1	1	1	4	1	0	1
$\bar{t}_3$	2	2	0	0	2	3	2	1	0	1	2	1	0	2	0	0
$\bar{t}_4$	2	2	0	1	0	2	3	0	2	1	1	1	1	1	1	0
$\bar{t}_5$	1	1	0	1	2	0	2	0	1	1	2	3	0	2	2	0

Figure 1.5: Sequences' word frequency vectors

$$||t_i - t_j||_2, \tag{1.1}$$

providing the definition of the pairwise distance matrix:

$$D = \begin{pmatrix} 0.00000 & 6.16441 & 5.47723 & 4.69042 & 5.47723 \\ 6.16441 & 0.00000 & 6.32456 & 5.47723 & 6.48074 \\ 5.47723 & 6.32456 & 0.00000 & 4.00000 & 4.69042 \\ 4.69042 & 5.47723 & 4.00000 & 0.00000 & 4.47214 \\ 5.47723 & 6.48074 & 4.69042 & 4.47214 & 0.00000 \end{pmatrix} \tag{1.2}$$

1.2.1 Minimum Evolution

Moved by the same motivation which inspired the MP, Kidd and Sgaramella-Zonta [1971] formulated the following conjecture: *given a set of taxa $\mathcal{L}$, the set $\mathcal{T}$ of all possible phylogenies of $\mathcal{T}$ and dissimilarity matrix D with entries d_{ij}, if the evolutionary distances are unbiased estimates of the corresponding true evolutionary distances, then the true phylogeny of taxa has an expected length shorter than any other possible phylogeny $T \in \mathcal{T}$ satisfying the following condition: for each pair of taxa i and j, the sum of the edge length belonging to the path p_{ij} in T from i to j is greater than, or equal to, their estimated evolutionary distance d_{ij}, (Catanzaro et al. [2022]).*

The problem can be formally defined as:

$$\min L_{ME} = \sum_{e \in E(T)} l(e), \tag{1.3}$$

where E is the edge set of T and $l(e)$ is defined by the linear system:

$$\sum_{e \in p_{ij}} l(e) = d_{ij} \quad \forall i, j \in \mathcal{L} \mid i \neq j, \tag{1.4}$$

which is imposing that the edges' length is such that the length of the whole path from i to j (the sum of the length of the edges in the path) must be exactly equal to d_{ij}. In general 1.4 might no have solution, so Cavalli-Sforza and Edwards [1967] proposed to compute an estimation $\bar{l}(e)$ of $l(e)$ via the mean-square method, *id est* by solving:

$$\min \sum_{\substack{i,j \in \mathcal{L} \\ i \neq j}} (d_{i,j} - \bar{l}(e))^2. \tag{1.5}$$

Such a methodology is known in the literature as *Minimum Evolution under Ordinary Least Squares*, (MEOLS). The proposed approach has an important theoretical advantage. The dissimilarities d_{ij} are generally computed based on the observed data (the molecular sequences), which are normally incomplete. Therefore, regardless of the substitution model used, we have that the d_{ij} are only estimates of the true value d^*_{ij}. A desirable property of any phylogenetic inference method is the ability to reconstruct the true tree T^* whenever the d_{ij} are sufficiently close to d^*_{ij}. Such a property is named *statistical consistentency*, (Denis and Gascuel [2003]). Here, it is crucial to remark that by "true tree" we are not meaning the *real* tree, the one describing the actual evolutionary process that occurred in nature, that is the result of stochastic events and which, if no other evidence is provided, it remains unknown. By true tree, we simply mean the one determined by the chosen criterion according to the complete data d_{ij}. Rzhetsky and Nei [1992] demonstrated that the MEOLS is statistically consistent.

However, it has been empirically shown by Gascuel [2000] that MEOLS provides poor edge length estimates when the variances of the d_{ij} are not constant and this leads to a rather low ability to recover the true tree.

It has been in fact shown by several authors (Bulmer [1991], Nei and Jin [1989], Nei et al. [1985], Susko [2003]) that the variance is much higher for long than for short distances and in particular that it grows exponentially as a the function of the d_{ij}.

An estimation model that shares with MEOLS the statistical consistency property but that is able to take into account the behaviour of the distances' variance will be the subject of the next subsection.

1.2.2 The Balanced Minimum Evolution Problem

We now present the distance method which represents the core of this work. The Balanced Minimum Evolution (Gascuel [2005]) is a phylogenetic inference model which can be stated in terms of a discrete nonlinear non-convex optimisation problem defined over unrooted binary trees (Catanzaro et al. [2012], Catanzaro et al. [2020]), specifically:

Problem 1 (Balanced Minumum Evolution) *Consider a set $\mathcal{L} = \{1, 2, \ldots, n\}$ of $n \geq 3$ taxa and the $n \times n$ relative symmetric distance matrix $\mathbf{D}$ with entries d_{ij} representing a measure of the dissimilarity between the pair of taxa $i, j \in \mathcal{L}$. Then, the **Balanced Minimum Evolution***

Problem (BMEP) *consists in finding a* phylogeny T *of* $\mathcal{L}$ *(i.e., an unrooted binary tree T having $\mathcal{L}$ as a leaf set) that minimises the following* length function

$$L(T) = \sum_{\substack{i,j \in \mathcal{L} \\ i \neq j}} \frac{d_{ij}}{2^{\tau_{ij}}}, \tag{1.6}$$

where τ_{ij} represents the path-length *between taxa i and j in T (see Figure 1.6), i.e., the number of edges belonging to the (unique) path in T connecting taxon i to taxon j (Pardi [2009], Catanzaro et al. [2012]).*

Based on an estimation model proposed by Pauplin [2000] to provide a fast analytical branch estimation formula, the BMEP was introduced in the literature on molecular phylogenetics by Desper and Gascuel [2002]. Subsequently, the problem was proven to be statistically consistent by Desper and Gascuel [2004]. The authors have also shown that if the variance of the dissimilarities is described by

$$Var(d_{ij}) = k 2^{\tau_{ij}}, \quad k > 0, \tag{1.7}$$

(*id est* it is proportional to the exponential of the topological distance of i and j in T) and assuming null covariances, the BMEP length function (Equation 1.6) defines also the minimum variance tree length estimator. This result is of particular importance if we note that the assumption defined by Equation 1.7 is consistent with the variance exponential growth of the d_{ij} suggested in the literature (see Subsection 1.2.1).

A great research effort regarding the BMEP has been put into the study of its combinatorics (Semple and Steel [2004], Cueto and Matsen [2011], Haws et al. [2011], Catanzaro et al. [2012], Forcey et al. [2016], Forcey et al. [2017], Catanzaro et al. [2020]), and an important result in this respect is that the BMEP is polynomially solvable if the input distance matrix D is *additive* (Gascuel [2005]), i.e., if its entries satisfy

$$d_{ij} + d_{kr} \leq \max\{d_{ik} + d_{jr}, d_{ir} + d_{jk}\} \quad \forall\, i, j, k, r \in \mathcal{L}. \tag{1.8}$$

However, when this property does not hold for D, the BMEP is generally a hard problem to solve. In fact, Fiorini and Joret [2012] proved, by reducing it to the 3-Colorability Problem (3CP) (Garey and Johnson [1979]), that the BMEP is $\mathcal{NP}$-hard and inapproximable within c^n, for some positive constant $c > 1$, unless $\mathcal{P} = \mathcal{NP}$. The authors also showed that if the input distance matrix $\mathbf{D}$ is just *metric*, i.e., if its entries satisfy the triangle inequality, then the optimal solution to the BMEP can be approximated within a factor of two.

In addition, Frohn [2021] showed that the problem remains APX-hard even when restricted to a simpler case, known in the literature as the Fixed-Tree BMEP (FT-BMEP) (Aringhieri et al. [2011a]), in which the topology of the phylogeny is fixed and one wants to assign the taxa to the leaves in order to minimize the BME length function.

The hardness of the problem limited the effectiveness of the exact approaches proposed so far, either via implicit enumeration algorithms (Pardi [2009], Aringhieri et al. [2011b], Catanzaro et al. [2012]) or via integer programming Catanzaro et al. [2012], which were capable of solving instances up to just 26 taxa.

For this reason, an important effort has been put into the development of efficient heuristics, for example in the works proposed by Desper and Gascuel [2002], Pardi [2009] and Fiorini and Joret [2012].

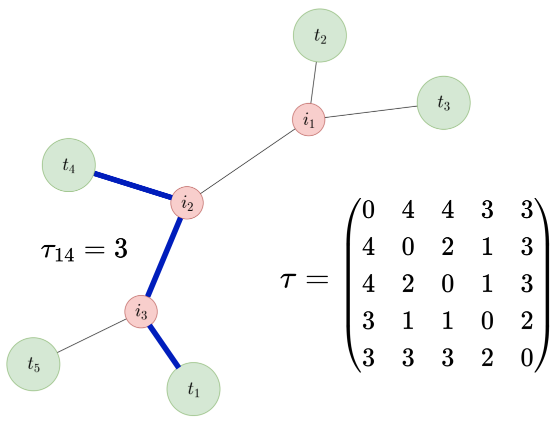

t_2
t_3
i_1
t_4
i_2
$\tau_{14} = 3$
i_3
t_5
t_1
$\tau = \begin{pmatrix} 0 & 4 & 4 & 3 & 3 \\ 4 & 0 & 2 & 1 & 3 \\ 4 & 2 & 0 & 1 & 3 \\ 3 & 1 & 1 & 0 & 2 \\ 3 & 3 & 3 & 2 & 0 \end{pmatrix}$

Figure 1.6: Example of a path-distance matrix τ of a phylogenetic tree

The current state-of-the-art approximate solution algorithm for the BMEP is FastME 2.0 (Lefort et al. [2015a]), whose algorithmic core is constituted by a local search that deterministically explores (part of) the solution space of the problem by means of topological changes (called *tree rearrangements*, Gascuel [2005]) carried out on an initial phylogeny. FastME often proves fast and accurate in practical phylogenetic analyses as well as able to scale to very large molecular datasets.

Example 2 *Let us consider the taxa of example 1 with the distance matrix D of Equation 1.2, and the phylogenetic tree T described in Figure 1.6.*

We first note that, since the diagonal of any distance matrix D is null, we obtain that Equation 1.6 can be rewritten as:

$$L(T) = \sum_{\substack{i,j \in \mathcal{L} \\ i \neq j}} \frac{d_{ij}}{2^{\tau_{ij}}} = \sum_{i,j \in \mathcal{L}} \frac{d_{ij}}{2^{\tau_{ij}}}. \tag{1.9}$$

Then, to compute the BMEP tree length of T one can simply consider the matrix $\hat{\tau}_{ij} = 1/2^{\tau_{ij}}$, perform the element-wise product $D * \hat{\tau}$, and sum the elements of the obtained matrix (operation denoted with the symbol $\overset{\circ}{\Sigma}$), in fact:

$$L(T) = \sum_{ij} d_{ij}/2^{\tau_{ij}} = \sum_{ij} d_{ij}\hat{\tau}_{ij} = \overset{\circ}{\sum} D * \hat{\tau}. \tag{1.10}$$

In our case, we have:

$$\tau = \begin{pmatrix} 0 & 4 & 4 & 3 & 3 \\ 4 & 0 & 2 & 1 & 3 \\ 4 & 2 & 0 & 1 & 3 \\ 3 & 1 & 1 & 0 & 2 \\ 3 & 3 & 3 & 2 & 0 \end{pmatrix} \quad \textit{and} \quad \hat{\tau} = \begin{pmatrix} 1.000 & 0.062 & 0.062 & 0.125 & 0.125 \\ 0.062 & 1.000 & 0.250 & 0.500 & 0.125 \\ 0.062 & 0.250 & 1.000 & 0.500 & 0.125 \\ 0.125 & 0.500 & 0.500 & 1.000 & 0.250 \\ 0.125 & 0.125 & 0.125 & 0.250 & 1.000 \end{pmatrix}, \tag{1.11}$$

which leads to:

$$\overset{\circ}{\sum} D * \hat{\tau} = \overset{\circ}{\sum} \begin{pmatrix} 0.000 & 0.385 & 0.342 & 0.586 & 0.685 \\ 0.385 & 0.000 & 1.581 & 2.739 & 0.810 \\ 0.342 & 1.581 & 0.000 & 2.000 & 0.586 \\ 0.586 & 2.739 & 2.000 & 0.000 & 1.118 \\ 0.685 & 0.810 & 0.586 & 1.118 & 0.000 \end{pmatrix} = 21.665 \; . \tag{1.12}$$

1.3 Motiavations and contributions

Although less accurate than ML and BI in terms of phylogenetic reconstruction, distance-based algorithms are capable of running on large datasets, whereas likelihood-based methods struggle to be applied because of their high computational cost. For this reason, distance-based approaches are still currently widely used as they represent a good compromise between good reconstruction accuracy (Kuhner and Felsenstein [1994]) and practicality, due to their simplicity and speed. Among distance methods, one of the most widely accepted and studied is the Minimum Evolution criterion because of its compliance with Occam's principle of scientific inference and its statistical consistency, which however does not accurately take into account the characteristics of the distances' variance. For this reason, the BMEP represents in phylogenetic inference an extremely valuable tool, as it shares with the other distance methods a good tree reconstruction accuracy and a competitive computational performance while guaranteeing statistical consistency and providing an appropriate description of the distances' variance. Moreover, as shown by Hordijk and Gascuel [2006], there is a strong correlation between the BMEP tree length estimation and the likelihood, meaning that trees with lower values of L (equation 1.6) tend to have higher likelihood value, a fact which makes the BMEP not only a good phylogenetic reconstruction criterion but also a potential tool to develop faster algorithms under the Maximum Likelihood framework. For these reasons, the necessity to develop more and more advanced algorithms capable of finding the most accurate solutions possible according to the BMEP principle, represents an important task in the phylogenetic research field.

Moved by these considerations, this work further adds to the above literature by presenting a novel heuristic, called *PhyloES*, that defines the new reference in approximating the optimal solutions to the BMEP. FastME is characterised by a deterministic exploration strategy of the solution space, which, as we show in this work, turns out to be too restrictive in some circumstances, causing premature convergence to solutions that can be arbitrarily far from the optimum. PhyloES proposes a possible way around this problem that consists of making nondeterministic the search in the solution space of the BMEP. This task is achieved by combining classical local search strategies with the *Evolution Strategy* (ES) framework discussed in Bäck [1996]. Specifically, starting from an initial set of phylogenies, PhyloES first generates a new set of solutions to the problem by using local search strategies similar to those implemented in FastME. Subsequently, PhyloES stochastically recombines the new phylogenies so obtained by means of the so-called *ES operators* (see Chapter 4). The iteration of the local search phase followed by the recombination phase (until a stop condition is met) allows to span the whole solution space to the BMEP by enabling the potential convergence to the optimum on a sufficiently long period. PhyloES can be downloaded at *https://github.com/andygaspar/PHYLOES*, is released to the scientific community under the form of a user-friendly open-source Python library, and makes an extended usage of Pytorch (Paszke et al. [2017]) (enabling a parallel CUDA GPU implementation of the algorithm discussed in the next sections) to improve computational efficiency.

1.4 Dissertation outline

This dissertation is organised as follows. Chapter 2 presents a detailed discussion of some of the existing exact approaches for the BMEP. This will be our starting point, in which we will discuss some important properties of the problem in question and we will set our first benchmark to test the heuristics introduced in the following chapter. In Chapter 3 we will analyse in depth the FastME algorithm, describe its components and assess its performance. In Chapter 4 we will first give a brief introduction to Evolutionary Algorithms and Evolution Strategies, defining and discussing the main concepts necessary to fully understand the PhyloES algorithm. Then, we will dive into the main core of this work. Here, we will introduce, define and analyse in depth the PhyloES algorithm and its components. In Chapter 5 we discuss the methodology adopted for the performance assessment and we present the computational results as well as some robustness analysis. Finally, Chapter 6 presents conclusions and future research considerations.

Chapter 2

Exact algorithms

In this Chapter, we aim to provide a detailed examination of a selection of established exact methodologies tailored for solving the BMEP, that will lay the groundwork for our subsequent analyses developed in the next chapters. We will first dig into a complete enumeration approach that will provide us with a first introduction to the topological properties of unrooted binary trees as well as a glance at the hardness of the BMEP computational challenge. We will also establish a baseline of comparison by means of an Integer Linear Programming that will be discussed in detail.

2.1 Complete enumeration

The first naive strategy one could think to adopt in order to solve the BMEP is via complete enumeration, *id est*, given an $n \times n$ matrix D representing the pairwise estimated biological distance between the element of a set $\mathcal{L}$ of n taxa, to compute the BMEP tree length of any phylogenetic tree in $\mathcal{T}$. However, this raises the problem of developing an algorithm capable of generating all elements of $\mathcal{T}$. A candidate for this task is represented by the so-called *Step-wise addition* algorithm, which flow can be described as follows.

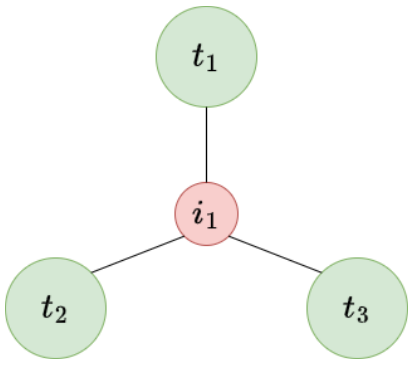
Let $\sigma : \mathcal{L} \to \{1, \dots, n\}$ a one-to-one map which determines an ordering for $\mathcal{L}$. In the initial step, we consider the star tree composed of $t_{\sigma^{-1}(1)}, t_{\sigma^{-1}(2)}, t_{\sigma^{-1}(3)}$ and a single internal node labelled as i_1. At each step $k = 3, \dots, n$ taxon $t_{\sigma^{-1}(k)}$ is inserted in the tree by choosing an edge (v_α, v_β) within the edge set of the partial tree, deleting (v_α, v_β) and adding the new edges $(t_{\sigma^{-1}(k)}, i_k)$, (v_α, i_k) and (v_β, i_k). We remark that v_α and v_β might be both internal nodes or an internal nodes and a taxon.

Figure 2.1 shows an example of step-wise addition procedure in which σ is defined as $\sigma(k) := k, \forall k \in \{1, \dots, n\}$. From the initial star $\{(t_1, i_1), (t_2, i_1), (t_3, i_1)\}$ (Figure 2.1a), (t_3, i_1) is selected, deleted and edges (t_4, i_2), (i_1, i_2) and (t_3, i_2) are added to the tree, (2.1b).

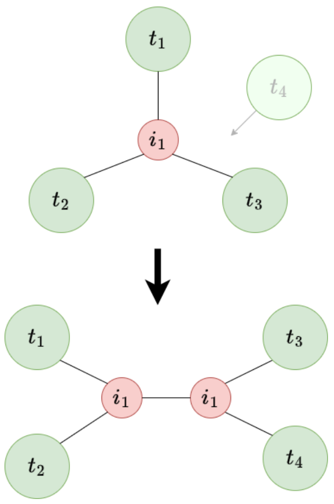
Still with $\sigma(k) := k, \forall k \in \{1, \dots, n\}$, Figure 2.2 shows instead the insertion of t_k in the two cases: the one in which the nodes of the selected edge (v_α, v_β) are both internal nodes and the one where one of them is a taxon.

It is important to remark that at each step the new insertion does not alter the degree of any of the existing nodes, the leaf added corresponds to a taxon, and the new internal edge is of degree three. The generated connected graph is therefore by definition a phylogenetic tree. What is left to prove is that any phylogenetic tree can be constructed with the above-described procedure.

This can be done simply by inverting the step-wise addition algorithm. Let T be a phylogenetic tree in $\mathcal{T}$ and let define a taxa ordering σ. Let also i_γ be the internal node adjacent to $t_{\sigma^{-1}(n)}$ and v_α, v_β, that we will call the insertion nodes, be the other two nodes adjacent to i_γ. If edges



(a) Initial star



(b) Insertion of the first taxa

Figure 2.1: First step of the Step-wise addition algorithm.

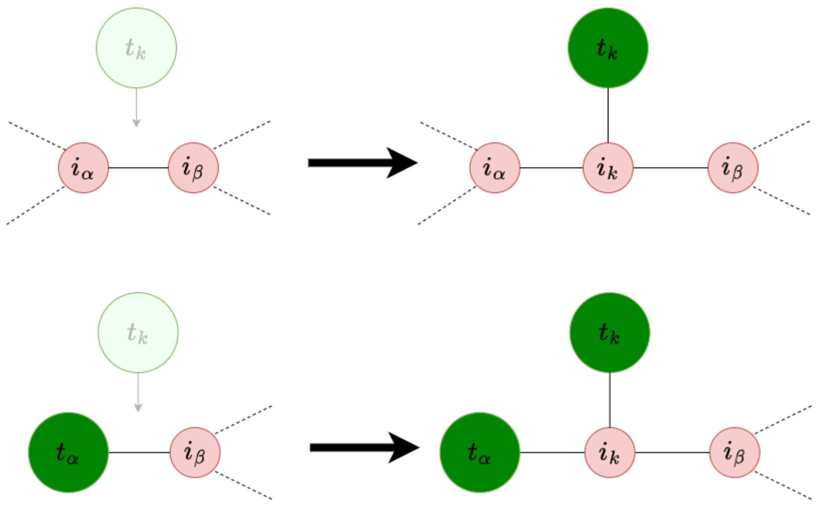
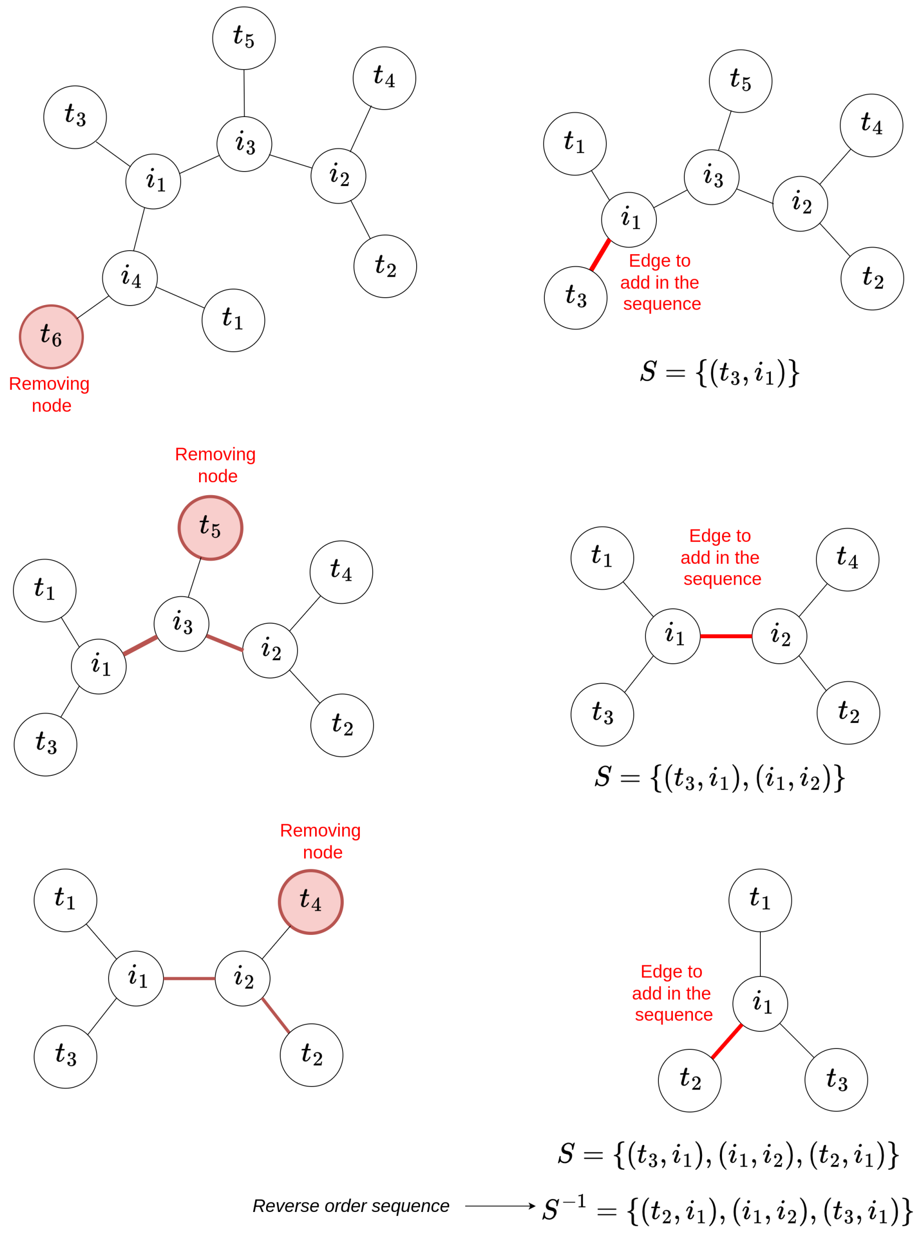


Figure 2.2: Insertion of t_k in the two cases: (UPPER) (v_α, v_β) are both internal nodes, so $(v_\alpha, v_\beta) = (i_\alpha, i_\beta)$: (BOTTOM): one of the nodes, for instance v_α, is a taxon and the other an internal node, so $(v_\alpha, v_\beta) = (t_\alpha, i_\beta)$.

$(t_{\sigma^{-1}(n)}, i_\gamma)$, (v_α, i_γ), $(v_{\beta}, _\gamma)$ are removed from the tree and a new edge connecting the two insertion nodes (v_α, v_β), named insertion edge, is added, we obtain a phylogenetic tree T_{n-1} of the $n-1$ taxa $t_{\sigma^{-1}(1)}, \ldots, t_{\sigma^{-1}(n-1)}$. We remark that if taxon $t_{\sigma^{-1}(n)}$ was then reinserted into T_{n-1} in a step-wise addition fashion by choosing edge (v_α, v_β), we would trivially re-obtain T. Therfore, if all taxa from $t_{\sigma^{-1}(n)}$ to $t_{\sigma^{-1}(4)}$ are removed one after the other storing at each step k insertion edge e_k, we would obtain the sequence $e_1, \ldots, e_{n-3}$. The final step of the proof consists of noticing that its reverse order, $e_{n-3}, \ldots, e_1$ defines the sequence of edges chosen at each step of the step-wise addition



t_5 t_4 t_3 i_3 i_1 i_2 i_4 t_2 t_6 t_1
Removing node
t_5 t_1 t_4 i_3 i_2 i_1 t_3 t_2
Edge to add in the sequence
$S = \{(t_3, i_1)\}$
Removing node
t_5 t_1 t_4 i_3 i_1 i_2 t_3 t_2
Edge to add in the sequence
t_1 t_4 i_1 i_2 t_3 t_2
$S = \{(t_3, i_1), (i_1, i_2)\}$
Removing node
t_1 t_4 i_1 i_2 t_3 t_2
Edge to add in the sequence
t_1 i_1 t_2 t_3
$S = \{(t_3, i_1), (i_1, i_2), (t_2, i_1)\}$
Reverse order sequence
$S^{-1} = \{(t_2, i_1), (i_1, i_2), (t_3, i_1)\}$

Figure 2.3: Reverse procedure to obtain the step-wise sequence from a given phylogeny.

algorithm to construct T starting from the initial star composed by the three taxa $t_{\sigma^{-1}(1)}, t_{\sigma^{-1}(2)}$ and $t_{\sigma^{-1}(3)}$. Figure 2.3 shows an example of the reverse procedure to obtain from a given phylogenetic tree the sequence of edges which determines its construction via the step-wise addition algorithm.

Now that we have a procedure to generate all possible phylogenetic trees we can also compute $|\mathcal{T}|$. In fact, we can observe that when constructing a tree, in the initial step we have 3 choices for the insertion of the first taxon, 5 for the second, 7 for the third and, since at each step two new

n taxa	$\lvert\mathcal{T}\rvert$
n= 4	3
n= 5	15
n= 6	105
n= 7	945
n= 8	10.395
n= 9	135.135
n= 10	2.027.025
n= 11	34.459.425
n= 12	654.729.075
n= 13	13.749.310.575
n= 14	316.234.143.225
n= 15	7.905.853.580.625

Table 2.1: Factorial growth of $\lvert\mathcal{T}\rvert$ with the number of taxa

edges are added to the tree, at step k we have $2k-3$ choices for the insertion of the k-th taxon; which means that:

$$\lvert\mathcal{T}\rvert = 3 \cdot 5 \cdot 7 \cdots 2(n-1) - 3 = (2n-5)!!. \tag{2.1}$$

Unfortunately, this result shows that a complete enumeration approach becomes untractable in practice quite quickly with the increasing number of taxa, as shown in Table 2.1.

Notable attempts to overcome this issue have been made by Pardi [2009], who exploited some lower bounds to exclude non-optimal edge choices for the tree construction within a Branch and Bound framework, and by Aringhieri et al. [2011a], who exploited the topological properties of the phylogenetic trees to partially reduce the size of the search space. However, both approaches have been shown to be applicable to problems up to 20 taxa.

Example 3 *Referring to the problem defined in Example 1 we can apply the complete enumeration method by generating all the* $(2 \cdot \mathbf{5} - 5)!! = 5 \cdot 3 = 15$ *phylogenies and then computing for each tree the BMEP length function. The resulting optimal phylogeny is the one represented as* T_2 *in Figure 2.4, since, according to the BMEP principle, is the one with the smallest length.*

2.2 An Integer Programming Formulation

In this Section, we present an Integer Programming (IP) formulation for the BMEP presented for the first time in Catanzaro et al. [2012]. Toward this aim a brief introduction on the combinatorial properties of the BMEP is needed.

Let us consider for a generic phylogenetic tree $\mathcal{T}$, in which we denote with $\mathcal{L}$ and $\mathcal{I}$ the set of leaves and internal nodes respectively, its length-path matrix τ, which element τ_{ij} represents the minimum path length between vertices i and j. We first notice that for any length-path matrix τ we have:

$$\tau_{ii} = 0 \qquad \forall i \in \mathcal{L}, \tag{2.2}$$

$$\tau_{ij} = \tau_{ji} \qquad \forall i,j \in \mathcal{L}, \tag{2.3}$$

$$\tau_{ij} + \tau_{jk} - \tau_{jk} \geq 2 \qquad \forall i,j,k \in \mathcal{L} |\, i \neq j \neq k, \tag{2.4}$$

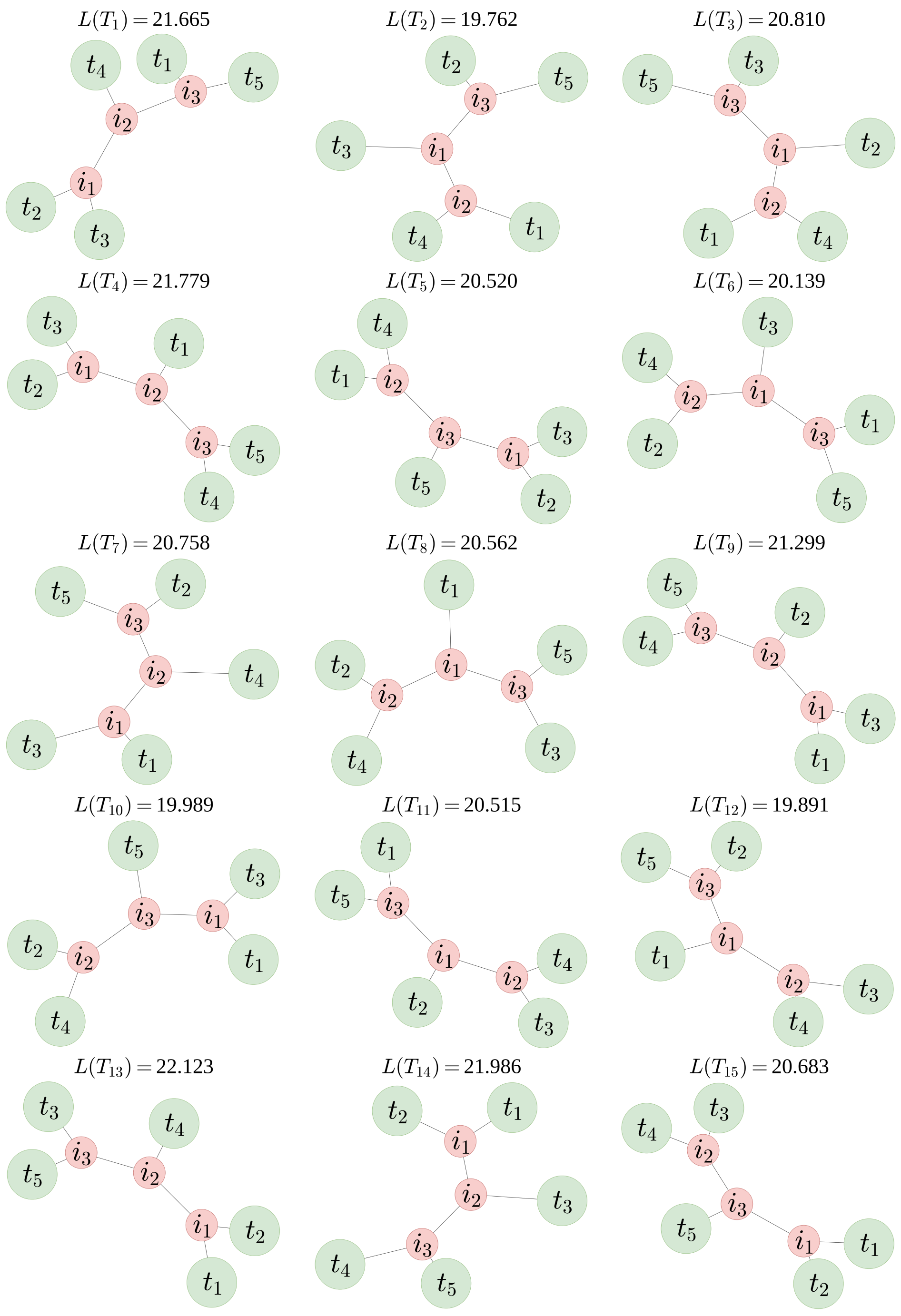


Figure 2.4: All possible phylogenies and their BMEP length of the five taxa case of Example 1.

since: (2.2) self distances are always null, (2.3) path distances are symmetric and (2.4) path distances satisfy the triangular inequality.

When τ describes an unrooted binary tree, Catanzaro et al. [2012] showed that it holds the following nontrivial necessary property, derived from Parker [1996] and known as *Kraft's equality*:

$$\sum_{\substack{j \in \mathcal{L} \\ j \neq i}} \frac{1}{2^{\tau_{ij}}} = \frac{1}{2} \tag{2.5}$$

Moreover, the authors derived from Pauplin [2000] the so-called *phylogenetic manifold* characterization, which is described by the following equality:

$$\sum_{\substack{i,j \in \mathcal{N} \\ i \neq j}} \frac{\tau_{ij}}{2^{\tau_{ij}}} = 2n - 3. \tag{2.6}$$

Finally, in order to fully characterise a phylogenetic tree we must ensure the *four-point Buneman's condition*, which states that for any quartet $i, j, q, t \in \mathcal{N}$ with $i \neq j \neq q \neq t$ it holds:

$$\tau_{ij} + \tau_{qt} \leq \max\{\tau_{iq} + \tau_{jt}, \tau_{it} + \tau_{jq}\}. \tag{2.7}$$

The Catanzaro et al. [2012] IP formulation consists of determining the length-path matrix τ which defines the optimal phylogenetic tree according to the BMEP criterion, given the taxa dissimilarity matrix D.

Denoting with $\mathcal{D}$ the set $\{1, 2, 3, \ldots, (n-1)\}$ of possible topological distances between nodes, we can define the problem's binary decision variables which represent the entries of the matrix τ:

$$x_{ij}^k = \begin{cases} 1 & \text{if } \tau_{ij} = k \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j \in \mathcal{N} | i \neq j, \forall k \in \mathcal{D}$$

In other words, the x_{ij}-s determine the topological distance between nodes. To ensure the four-point conditions (Equation 2.7) we also need to define the set of binary decision variables:

$$y_{ij}^{qt} = \begin{cases} 1 & \text{if } \tau_{it} + \tau_{jq} \geq \tau_{iq} + \tau_{jt} \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j, q, t \in \mathcal{N} | i \neq j \neq q \neq t$$

The BMEP **objective function** 1.6 can be then written as:

$$\min \sum_{\substack{i,j \in \mathcal{L} \\ j \neq i}} d_{ij} \left(\sum_{k \in \mathcal{D} \setminus \{1\}} \frac{x_{ij}^k}{2^k} \right) \tag{2.8}$$

which is subject to the following constraints.

The distance between nodes i and j is defined by a single value:

$$\sum_{k \in \mathcal{D}} x_{ij}^k = 1 \quad \forall i, j \in \mathcal{N} | i \neq j. \tag{2.9}$$

The distance is symmetrical:

$$x_{ji}^k = x_{ij}^k \quad \forall i, j \in \mathcal{N} | i < j, k \in \mathcal{D}. \tag{2.10}$$

The Kraft equality (Equation 2.5) must be ensured:

$$\sum_{i\in\mathcal{L},k\in\mathcal{D}\setminus\{1\}} \frac{2x_{ji}^k}{2^k} = \frac{1}{2} \quad \forall i \in \mathcal{N}. \tag{2.11}$$

The solution must be consistent with the *phylogenetic manifold* characterisation (Equation 2.6):

$$\sum_{\substack{k\in\mathcal{D}\\k\neq 1}} \frac{k}{2^k} \sum_{\substack{i,j\in\mathcal{L}\\i\neq j}} x_{ij}^k = (2n-3). \tag{2.12}$$

The four-point condition must be ensured:

$$\sum_{k\in\mathcal{D}} k\left(x_{ij}^k + x_{qt}^k\right) \leq \sum_{k\in\mathcal{D}} k\left(x_{iq}^k + x_{jt}^k\right) + (2n-2)y_{ij}^{qt} \qquad \forall i,j,q,t \in \mathcal{N} | i \neq j \neq q \neq t, \tag{2.13}$$

$$\sum_{k\in\mathcal{D}} k\left(x_{ij}^k + x_{qt}^k\right) \leq \sum_{k\in\mathcal{D}} k\left(x_{it}^k + x_{jq}^k\right) + (2n-2)\left(1 - y_{ij}^{qt}\right) \quad \forall i,j,q,t \in \mathcal{N} | i \neq j \neq q \neq t. \tag{2.14}$$

Each leaf is adjacent to only one internal node:

$$x_{ij}^1 = 0 \quad \forall i,j \in \mathcal{L} | i \neq j \tag{2.15}$$

$$\sum_{j\in\mathcal{I}} x_{ij}^1 = 1 \quad \forall i \in \mathcal{L}. \tag{2.16}$$

Each internal node is adjacent to exactly three nodes:

$$\sum_{\substack{j\in\mathcal{N}\\j\neq i}} x_{ij}^1 = 3 \quad \forall i \in \mathcal{I} \tag{2.17}$$

No triangles between internal nodes are allowed (Constraints 2.15 and 2.16 already ensure this condition for leaves):

$$x_{ij}^1 + x_{ir}^1 + x_{rj}^1 \leq 2 \quad \forall i,j,r \in \mathcal{I} | i \neq j \neq r. \tag{2.18}$$

If leaf j is adjacent to the internal node r, the distance between any other leaf i and j is greater than the distance between i and r (Figure 2.5):

$$x_{ij}^k + 1 \geq x_{ir}^{(k-1)} + x_{rj}^1 \quad \forall i,j \in \mathcal{L} | i \neq j, r \in \mathcal{I}, k \in \mathcal{D}\setminus\{1, n-1\}. \tag{2.19}$$

If node j is adjacent to node q and the distance between node r and i is equal to $k-1$, then the distance between i and r must be equal to k or $k-2$ (Figure 2.6):

$$x_{ij}^k + x_{ij}^{(k-2)} + 1 \geq x_{ir}^{(k-1)} + x_{rj}^1 \quad \forall i,j,r \in \mathcal{N} | i \neq j \neq r, k \in \mathcal{D}\setminus\{1, n-1\}. \tag{2.20}$$

Variables are binary:

$$x_{i,j}^k \in \{0,1\} \quad \forall i,j \in \mathcal{N}, k \in \mathcal{D} \tag{2.21}$$

$$y_{i,j}^{q,t} \in \{0,1\} \quad \forall i,j,q,t \in \mathcal{N} \tag{2.22}$$

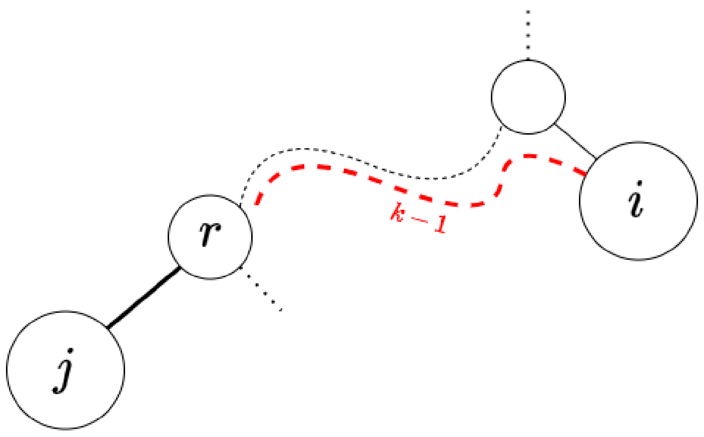


Figure 2.5: Graphical representation of Constraint 2.19.

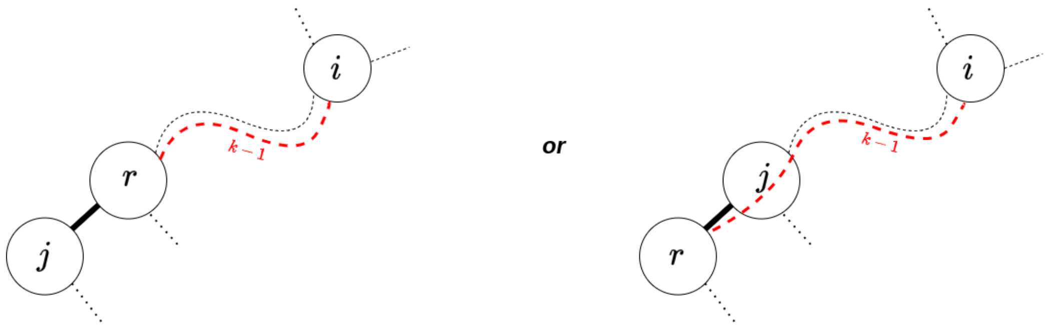


Figure 2.6: Graphical representation of Constraint 2.20.

The solution times of such formulation prove to be particularly slow, mainly due to the large number of y variables ($\binom{n}{4}$). The authors proposed the use of a branch-and-bound (B&B) approach based on the step-wise algorithm presented in Section 2.1 in which the lower bound for a generic node of the B&B tree is obtained by computing the linear relaxation but considering only constraints (2.9)-(2.12).

The resulting algorithm represents the current state-of-the-art among the exact approaches for the BMEP [Catanzaro et al., 2022]. However, its capabilities remain quite limited so far, as the optimal solution can be provided in a reasonable amount of time ($\approx$ 1 hour) for problems with up to 26 taxa. Still, we have to notice that, as we will see in 3.3.1, the computation time highly varies from instance to instance. Further research might be needed to fully understand the dependency of the computation time on the structure of D. Since many of the real-world instances include hundreds, or even thousands, of taxa the need for a heuristic approach seems to be therefore necessary.

Chapter 3

The FastME algorithm

FastME, which stands for Fast Minimum Evolution, is a powerful algorithm that represents, at the current, the state-of-the-art for the BMEP. Developed as an efficient alternative to traditional tree-building methods, FastME employs a heuristic approach to rapidly construct phylogenetic trees.

It leverages powerful local search operators defined by tree rearrangement moves to improve on any initial solution provided by the user. FastME achieves its speed first by generating a feasible solution to an instance of the BMEP by means of an *initial heuristic* and then iteratively improves it by using tree rearrangements (*local search phase*) until the length of the best-so-far tree stops improving.

This allows FastME to quickly generate phylogenetic trees, making it an attractive choice for researchers working with large datasets or when time a is critical factor.

In this chapter, we will look in detail at the various components of the algorithm in order to understand the reason for its efficiency as well as to identify its weaknesses.

3.1 Topological Tree Rearrangements

In the local search phase, FastME uses two tree rearrangements, called the *Balanced Nearest Neighbours Interchanges* (BNNI) (Desper and Gascuel [2002]) and the *Balanced Subtree Pruning and Regrafting* (BSPR) (Hordijk and Gascuel [2006]), which have been shown to provide important consistency properties within the BMEP framework (Bordewich et al. [2009]). The BNNI and BSPR are based, respectively, on the *Nearest Neighbours Interchanges* (NNI) and *Subtree Pruning and Regrafting* (SPR) tree rearrangements. FastME extensively uses these two techniques, along with formulas that allow to speed up the evaluation of neighbourhood elements thus making FastME extremely computationally efficient.

3.1.1 Nearest Neighbour Interchanges

The NNI operation consists of interchanging two subtrees that are adjacent to an internal edge e (Semple and Steel [2003]), where by internal edge we refer to an edge connecting two internal nodes. Figure 3.1 shows two subtrees that can be obtained by applying NNI to edge e: in the first case, the subtree A is swapped with C, while in the second with D. The other possible combinations of exchanges are not considered since they yield *symmetric phylogenies*, i.e., phylogenies topologically equivalent to the ones already taken into account. We will refer to A, B, C, D as the **NNI neghbours** subtrees, or simply NNI neighbours.

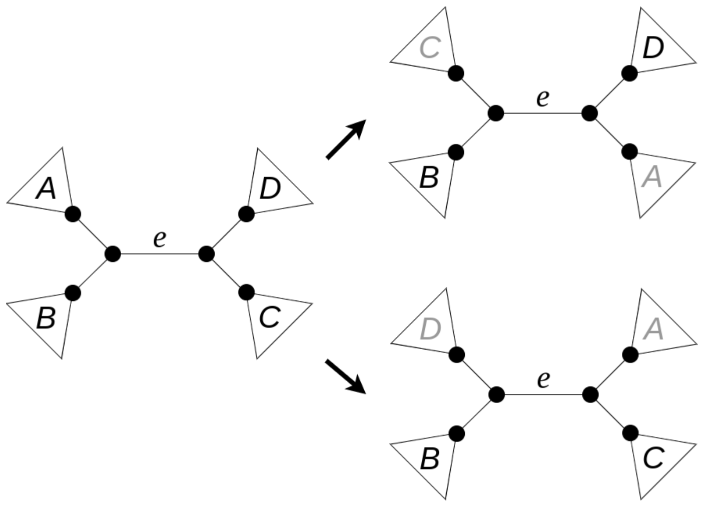


Figure 3.1: Representation of an NNI move. When edge e is selected, two new trees are created by swapping subtrees A-D and A-C, respectively.

3.1.2 BNNI

The idea behind the BNNI can be summarised as follows.

- Generate an initial tree
- Until no further improvement is achievable:
    1. test all possible NNI swaps of the initial tree
    2. perform, if possible, the one that improves the most the tree length

The computation of the length of all possible trees resulting from the NNI swaps would be however very expensive as it would imply recalculating the distance matrix τ of each tree ($O(n^2 \log(n)$, Desper and Gascuel [2002]). This can be avoided by exploiting an appropriate decomposition of the length function. Let us define the distance between two non-intersecting subtrees A and B as:

$$l_{AB} = l_{BA} := \sum_{i,j \in A \cup B} \frac{d_{ij}}{2^{\tau_{ij}}}, \tag{3.1}$$

and the subtree A length as:

$$l_A = \sum_{i,j \in A} \frac{d_{ij}}{2^{\tau_{ij}}} \tag{3.2}$$

Let us consider for example the tree T of figure 3.2 and suppose that we have already computed the matrix M^T of dimension $(2n-3) \times (2n-3)$ which element M^T_{AB} represents the distance between subtrees A and B. The length of T can be then decomposed in the following manner:

$$L(T) = l_A + l_B + l_C + l_D + l_{AB} + l_{AC} + l_{AD} + l_{BC} + l_{BD} + l_{CD}. \tag{3.3}$$

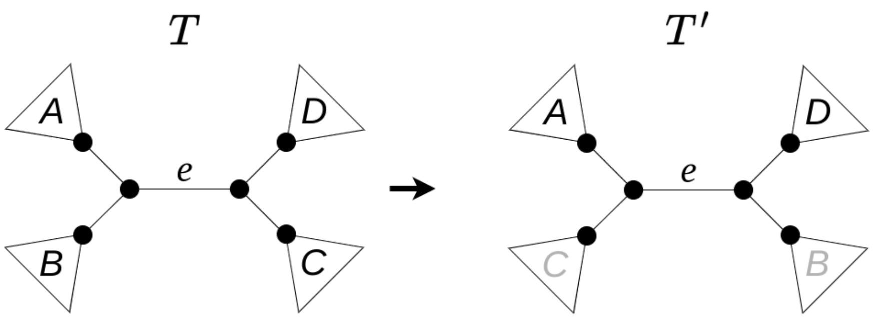


Figure 3.2: NNi swap between subtrees B and C.

T' is generated via the swap between subtrees A and C performed. Hence the length contributions l_A, l_B, l_C, l_D do not change as the structures of the corresponding subtree remain unaltered by the swap. Similarly, the same consideration holds for l_{AD}, l_{BC} as the swap does not change the corresponding subtrees' distance, which is the case instead of the distance between A and C, and between B and D. Denoting with $\bar{l}$ the value of modified subtrees' distances, we have:

$$L(T') = l_A + l_B + l_C + l_D + \overline{l_{AB}} + \overline{l_{AC}} + l_{AD} + l_{BC} + \overline{l_{BD}} + \overline{l_{CD}}. \tag{3.4}$$

Since in T' A and B are one edge further away than their distance in T, denoting τ_{ij}^{T} as the distance between taxa i and j in T, and $\tau_{ij}^{T'}$ as the distance between taxa i and j in T', we obtain:

$$\tau_{ij}^{T'} = \tau_{ij}^{T} + 1 \quad \forall i \in A, j \in B, \tag{3.5}$$

which gives:

$$\overline{l_{AB}} = \sum_{i,j \in A \cup B} \frac{d_{ij}}{2^{\tau_{ij}^{T'}}} = \sum_{i,j \in A \cup B} \frac{d_{ij}}{2^{\tau_{ij}^{T}+1}} = \frac{1}{2}\Big(\sum_{i,j \in A \cup B} \frac{d_{ij}}{2^{\tau_{ij}^{T}}}\Big) = \frac{1}{2} l_{AB}. \tag{3.6}$$

With the same reasoning, we get:

$$\overline{l_{CD}} = \frac{1}{2} l_{CD}. \tag{3.7}$$

Conversely, in T', A and C are an edge closer with respect to their distance in T, so:

$$\tau_{ij}^{T'} = \tau_{ij}^{T} - 1 \quad \forall i \in A, j \in C, \tag{3.8}$$

which leads to:

$$\overline{l_{AC}} = 2 l_{AC}, \tag{3.9}$$

and, similarly to

$$\overline{l_{BD}} = 2 l_{BD}. \tag{3.10}$$

In conclusion, we obtain that the difference in tree length between T (3.3) and T'(3.4) simply reduces to:

$$\Delta T = L(T) - L(T') = \frac{1}{2}(l_{AB} + l_{CD}) - (l_{BD} + l_{AC}), \tag{3.11}$$

which allows us to determine whether an NNI swap leads to a length improvement without the need to calculate τ for the new tree T'. Furthermore, if the swap is performed the matrix $M^{T'}$ can be obtained from M^{T} in $O(n \log(n))$ (Desper and Gascuel [2002]) updating the elements affected

Algorithm 1 BNNI

```
Require: $M$, $T$
  $T_{best} = T$
  $\Delta = -1$
  while $\Delta < 0$ do
      $\Delta, T_{best} = BNNIstep(T_{best}, M)$
      update $M$
  end while
  return $T_{best}$
```

Algorithm 2 BNNIstep

```
Require: $T_{best}, M$
  $T_{best} = T$
  $\Delta = 0$
  for internal edge in $T$ do
      for swap in NNI swaps do
          $T'$ = swapped tree
          $\Delta' = L(T) - L(T')$
          if $\Delta' < \Delta$ then
              $T_{best} = T'$
              $\Delta = \Delta'$
          end if
      end for
  end for
  return $\Delta, T_{best}$
```

by the swap. Algorithms 1 describes the detail of the BNNI algorithm, in which the sub-routine *BNNIstep* (Algorithm 2) is called until no further tre length improvement is achieved. In view of the above considerations, Algorithm 1 requires $O(n^2 + pn\log(n)$, where $O(n^2)$ is due to the initial computations of M and $pn\log(n)$ is the cost of performing p times the *while* loop.

3.1.3 Subrtree pruning and regrafting

The SPR neighbours are instead obtained by *pruning* (i.e. removing), a subtree from the initial phylogeny, and by *regrafting* (i.e. reattaching), the pruned subtree into one of the remaining branches in the phylogeny (Semple and Steel [2003]). Figure 3.3 shows a simple example of an SPR move where the red subtree is pruned and regrafted onto the target yellow edge.

More in general, as depicted in Figure 3.4, an SPR move can be defined as follows:

- pick an edge s.
- Pick a vertex x of s. Of the two subtrees rooted in s consider the one (S) opposite to x.
- Pick a target edge e within the two subtrees S_1, S_2, which are the ones rooted in z_1 and z_2 respectively and which are not intercepting S. Since e_1 and e_2 are connected to x they also have to be excluded from S_1 and S_2 as the insertion of S into any of them would lead to the initial tree.

- Detach S as well as s and x from the tree
- Split e by generating the edges e_1 and e_2 in order to insert x in between them
- Merge e_z^1 and e_z^2 to form a single edge e_z connecting z_1 and z_2.

We will refer to the SPR move just described with $S \mapsto e$.

3.1.4 Balanced SPR

Now that we have defined an SPR move, we can describe the flow, similar to the BNNI ones, of the BSPR:

- Generate an initial tree
- Until no further improvement is achievable:
    1. test all possible SPR swaps of the initial tree
    2. perform, if possible, the one that improves the most the tree length

Similar to the BNNI case, the length computation of all possible trees resulting from the SPR swaps would be however very expensive. This time, since the number of choices for any SPR move is $O(n)$ for any possible subtree, the recalculation of the distance matrix τ of each tree would cost $(O(n^4 \log(n))$. Fortunately, Hordijk and Gascuel [2006] provides a very efficient workaround for such a hurdle.

In fact, we can first notice that any SPR move can be obtained via multiple applications of NNI swaps. As shown in Figure 3.5, let us consider a general SPR move where subtree S has to be inserted on edge e. Since any tree is a connected graph there is a path $p_1, \dots, p_k$ between s and e. Let us define the subtree S_i as the one not including p_i and p_{i+1} and with root the node between p_i and p_{i+1}. If we now consider *Step 1* of Figure 3.5 the swap between S and S_1 is the NNI operation of Figure 3.2 which moves S one edge closer on its path to e. The same consideration holds for any $i = 2, \dots, k$, allowing at each step i to swap S with S_i, ultimately resulting in the target SPR move.

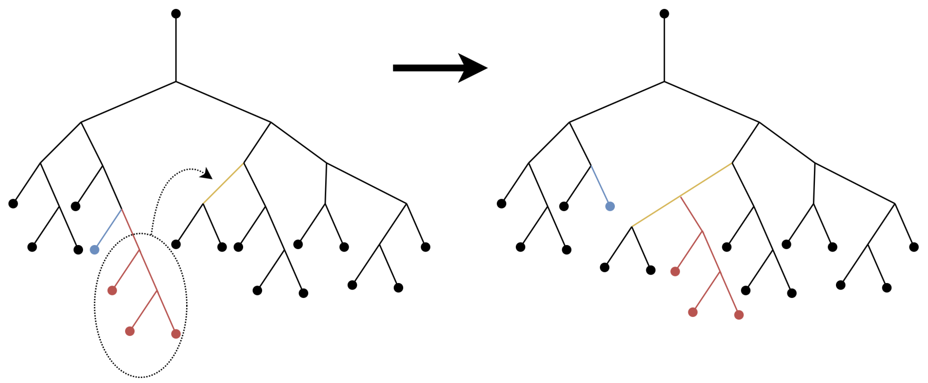


Figure 3.3: Representation of an SPR move. The red subtree is detached (pruned) from the tree and reattached (regrafted) into the target yellow edge by splitting it into two edges. In order to fix the tree the blue subtree has to be instead reattached to its grandparent.

If the subtree distance matrix M is known, the difference in terms of tree length can be then obtained by recursively computing equation 3.11. We further notice that in this case after each NNI swap it is not necessary to update the whole M but only the terms required by 3.11. Referring to Figure 3.5, the subtree distance update at step i can be performed in constant time as follows.

Let us indicate with a bar the new NNI neighbours obtained after the swap. After the swapping S and S_i the new A, $\overline{A}$, is composed of the old A and S_i. B instead does not change, $\overline{B} = S = B$. The distance between $\overline{A}$ and $\overline{B}$ is therfore:

$$l_{\overline{AB}} = \frac{1}{2} l_{AB} + l_{SS_i}. \tag{3.12}$$

Except for S_{i+1}, all subtrees of D still belong to $\overline{D}$, but they are an edge closer to S, so:

$$l_{\overline{BD}} = 2(l_{BD} - l_{SS_{i+1}}). \tag{3.13}$$

In order to reapply Equation 3.12 at the following step, the distance between S and S_{i+1} must be updated:

$$\bar{l}_{SS_{i+1}} = 2 l_{SS_{i+1}}. \tag{3.14}$$

$l_{\overline{CD}}$ is known and its value is stored in M, as both $\overline{C} = S_2$ and $\overline{D}$ have not be altered so far. Instead, $l_{\overline{AC}}$ must be computed. Referring to step i of Figure 3.5, we notice that the new $\overline{A}$ includes

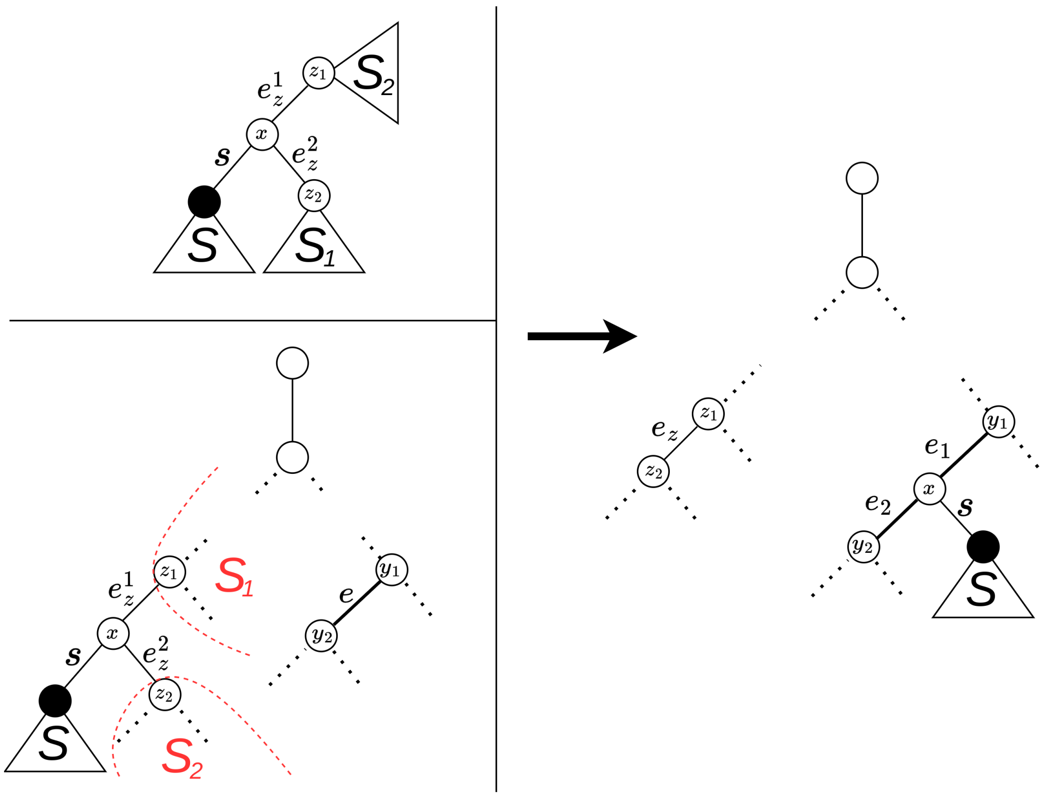


Figure 3.4: Representation of $S \mapsto e$.

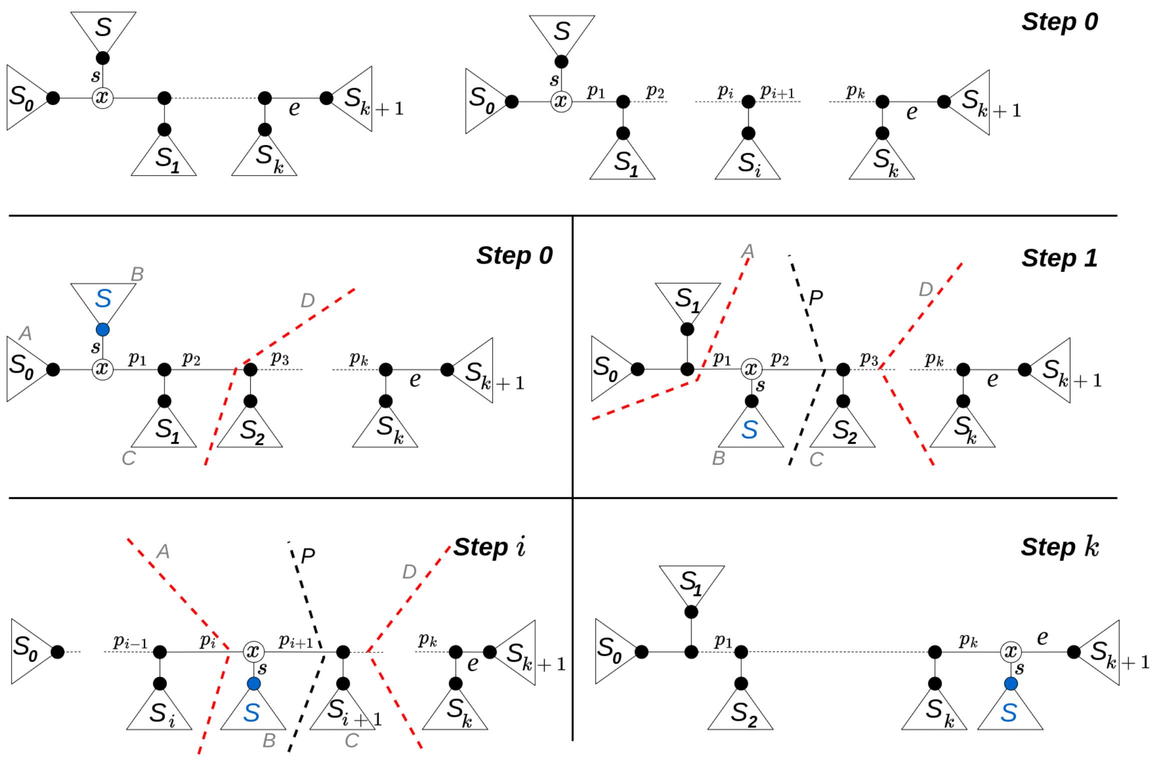


Figure 3.5: Representation of an SPR move performed by multiple applications of NNI swaps.

the same subtrees of P except for S. $l_{S_{i+1}P}$ is known, but we have to take into account that apart from S_0, all subtrees S_j, with $j = 1, \dots, i$ are now one edge further away. We then obtain:

$$l_{\overline{AC}} = \frac{1}{2}(l_{S_{i+1}P} - l_{SS_{i+1}}) + \frac{1}{2} l_{S_0 S_{i+1}}. \tag{3.15}$$

Finally, denoting the tree obtained after the swap with S_i as T_{S_i}, we notice that:

$$L(T) - L(T_{S_k}) = \big(L(T) - L(T_{S_1})\big) + \big(L(T_{S_1}) - L(T_{S_2})\big) + \cdots + \big(L(T_{S_k-1}) - L(T_{S_k})\big), \tag{3.16}$$

which provides the recursive formula to compute the final tree length difference.

We further remark that each swap i performed in the process corresponds to the SPR move $S \mapsto p_{i+1}$. Therefore, this procedure provides the additional advantage of computing the length difference for all the SPR moves of S included in the path between s and e.

As shown in Figure 3.6, all SPR moves of S can be recursively computed following the tree structure from s to all leaves not included in S and keeping track of the best tree length improvement.

The whole procedure is described in Algorithm 3, in which equations 3.12, 3.13, 3.14 and 3.15 have been included in the function *UpdateDistances*.

Finally, we can now in Algorithms 5 provide the definition of the BSPR, which consists of consecutive applications of the sub-routine *BSPRstep* (Algorithm 6) until no further tree length improvement is achieved.

Since there are $O(n)$ possible moves per each of the $O(n)$ subtrees, the complexity of *BSPRstep* (Algorithm 6) is $O(n^2)$. The initial M computation as well as its update requires $O(n^2)$. Therefore the whole BSPR requires $O(pn^2)$, where p is the number of iterations of the while loop.

3.2 FastME Algorithm

Heaving provided a heuristic that calculates an initial tree, which will be the focus of the next sections, we can now give the complete definitions of the FastME algorithm (7).

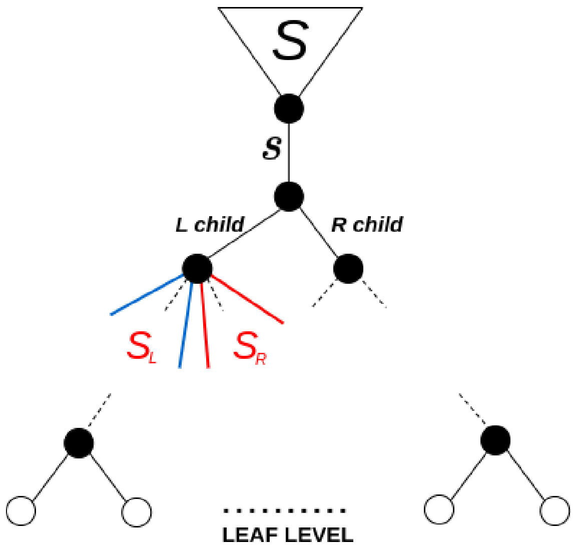


Figure 3.6: Recursion tree for subtree S SPR moves computations.

Algorithm 3 SPR

```
Require: $S, T$
  $\Delta = 0$
  $\Delta_{best} = 0$
  $T_{best} = T$
  $\Delta, \Delta_{best}, T_{best} = Recursion(S, \Delta_{best})$
  return $\Delta_{best}, T_{best}$
```

Algorithm 4 Recursion

```
Require: $S, \Delta, \Delta_{best}, T_{best}$
  if $S =$ leaf then
      Return
  end if
  if $LChild \neq$ leaf then
      $T_{S_R}$ = tree obtained by swapping $S$ and $S_R$
      $UpdateDistances$
      $\Delta = \Delta + L(T_S) - L(T_{S_R})$
      if $\Delta < \Delta_{best}$ then
          $\Delta_{best} = \Delta$
          $T_{best} = T_{S_R}$
      end if
      $Recursion(S_R, \Delta, \Delta_best, T_{best})$
  end if
  if $RChild \neq$ leaf then
      $T_{S_L}$ = tree obtained by swapping $S$ and $S_L$
      $UpdateDistances$
      $\Delta = \Delta + L(T_S) - L(T_{S_L})$
      if $\Delta < \Delta_{best}$ then
          $\Delta_{best} = \Delta$
          $T_{best} = T_{S_L}$
      end if
      $Recursion(S_L, \Delta, \Delta_{best}, T_{best})$
  end if
  return $\Delta_{best}, T_{best}$
```

Algorithm 5 BSPR

```
Require: $M$, $T$
  $T_{best} = T$
  $\Delta = -1$
  while $\Delta < 0$ do
      $\Delta, T_{best} = BSPRstep(T_{best}, M)$
      update $M$
  end while
  return $T_{best}$
```

3.3 Standard Initialisation Algorithms

As mentioned in section 3.2, FastME requires an initial tree to start the BNNI and the BSPR subroutines. The heuristics described in the literature for determining a tree given a taxa distance matrix D can be classified into two main categories: *constructive* and *agglomerative* (Lemey et al. [2009]). The constructive heuristics build a phylogeny for a given set $\mathcal{L}$ of $n \geq 3$ in a step-wise fashion (see Section 2.1) choosing the insertion edge according to any given criterion. The agglomerative heuristics, instead, akin clustering techniques, construct a phylogeny for $\mathcal{L}$ by starting from a star tree that connects all the n taxa in $\mathcal{L}$ and by iteratively aggregating pair of taxa according to a

Algorithm 6 BSPRstep

```
Require: $T_{best}, M$
  $T_{best} = T$
  $\Delta = 0$
  for subtree $S$ in $T$ do
    $\Delta', T' = SPR(S)$
    if $\Delta' < \Delta$ then
      $T_{best} = T'$
      $\Delta = \Delta'$
    end if
  end for
  return $\Delta, T_{best}$
```

Algorithm 7 FastME

```
Require: $D$, $InitHeuristic$
  $T = InitHeuristic(D)$
  $T_{nni} = BNNI(T)$
  $T_{spr} = BSPR(T)$
  $T = \arg\min\{l(T), l(T_{nni}), l(T_{spr})\}$
  return T
```

specific selection criterion (Sokal [1958], Saitou and Nei [1987]).

FastME allows the use of several *Standard Initialisation Algorithms (SIA)* to generate the initial phylogeny. Examples include constructive heuristics that use *Greedy Balanced Minimum Evolution algorithm* (GBME) and *Ordinary Least-Square for Minimum Evolution* (OLSME) as an edge selection criterion (Desper and Gascuel [2002], Rzhetsky and Nei [1992]) and agglomerative heuristics based on the *Neighbor-Joining* (NJ) algorithm (Saitou and Nei [1987]), the *Unweighted Neighbor-Joining* (UNJ) algorithm (Gascuel [2002]), and Gascuel's BioNJ [Gascuel, 1997]. Despite their detailed analysis is beyond the scope of this work, it is essential to discuss here some empirical evidence arising from our preliminary tests.

3.3.1 Comparison with IP model and SIA tests

For the following tests, we considered the Ribosomal Database Project RDPII, one of the benchmark datasets used in this work and consisting of aligned and annotated rRNA gene sequences for which the distance between taxa has been computed using the F84 substitution model (Felsenstein and Churchill [1996]).

The first performance assessment is naturally the comparison with an exact method. For this task, we made use of the IP formulation presented in Section 2.2, as it represents the state-of-the-art among the exact approaches. Due to its computational limitations, the instance sizes considered were quite small: 15, 20, and 25 taxa. For each of them, we generated 10 problems by randomly picking the corresponding number of taxa.

The C++ implementation has been provided by the authors and it makes use of the FICO-Xpress Library. The time limit for the ILP model has been set to 6 hours and the max gap tolerance to 10^{-10}. In order to speed up the computations, the solver was provided with the initial solution computed with PhyloES (see Chapter 4). The experiments have been run on an Intel Core(TM) i7-10700 (2.90GHz) 16 cores machine. Table 3.1 illustrates the results of the experiments, showing the comparison between FastME and the IP model in terms of objective values and computing time (in seconds). FastME proved extreme efficiency, providing the optimal solution in all instances in which the IP was able to terminate the computation within the time limit. In these cases, FastME was on average $99,5\%$ times faster than the ILP (std $0,01\%$). We also report that in only 2 instances the IP solver outperformed FastME in terms of solution quality, but with a very limited gap.

run	n	FastME obj	IP obj	FastME time	IP time	gap %
1	15	1.458490	1.458490	0.012	0.248	0.000000
2	15	1.734989	1.734989	0.011	11.615	0.000000
3	15	1.818631	1.818631	0.014	6.543	0.000000
4	15	1.799283	1.799283	0.012	55.476	0.000000
5	15	1.542442	1.542442	0.012	2.103	0.000000
6	15	1.816193	1.816193	0.012	103.122	0.000000
7	15	1.659289	1.659289	0.012	0.573	0.000000
8	15	1.771357	1.771357	0.013	0.715	0.000000
9	15	1.785821	1.785821	0.012	3.135	0.000000
10	15	1.728345	1.728345	0.012	5.075	0.000000
11	20	2.247273	2.247273	0.015	964.173	0.000000
12	20	2.254319	2.254319	0.014	58.561	0.000000
13	20	2.304339	2.304339	0.014	154.638	0.000000
14	20	2.367511	2.367511	0.015	63.619	0.000000
15	20	2.322821	2.322821	0.014	366.301	0.000000
16	20	2.175853	2.175853	0.014	966.346	0.000000
17	20	2.252254	2.252254	0.014	337.923	0.000000
18	20	2.298822	2.298822	0.014	4043.536	0.000000
19	20	2.406584	2.406584	0.014	231.219	0.000000
20	20	1.855247	1.855247	0.014	10.893	0.000000
21	25	2.607170	2.607170	0.017	Time Limit	0.000000
22	25	2.859623	2.859411	0.018	Time Limit	0.000074
23	25	2.576427	2.576427	0.018	Time Limit	0.000000
24	25	2.705412	2.705412	0.017	737.774	0.000000
25	25	2.608054	2.608054	0.018	Time Limit	0.000000
26	25	2.705522	2.704784	0.017	Time Limit	0.000273
27	25	2.851618	2.851618	0.018	Time Limit	0.000000
28	25	2.631415	2.631415	0.018	18575.847	0.000000
29	25	2.461560	2.461560	0.018	Time Limit	0.000000
30	25	2.765643	2.765643	0.017	Time Limit	0.000000

Table 3.1: Comparison between FastME and the ILP model (time in seconds)

Another important remark arises from the analysis of the IP computation time. Its high standard deviation ($33,9$ and $1218,6$ respectively within the 15 and the 20 taxa instances) and the differences between the maximum time and the minimum ($102,8$ for the 15 taxa case and $4032,6$ for the 20 taxa case) indicate how the problem's hardness highly variates from instance to instance, as also shown in Figure 3.7.

The second performance assessment we discussed here, concerns the SIAs. In order to study

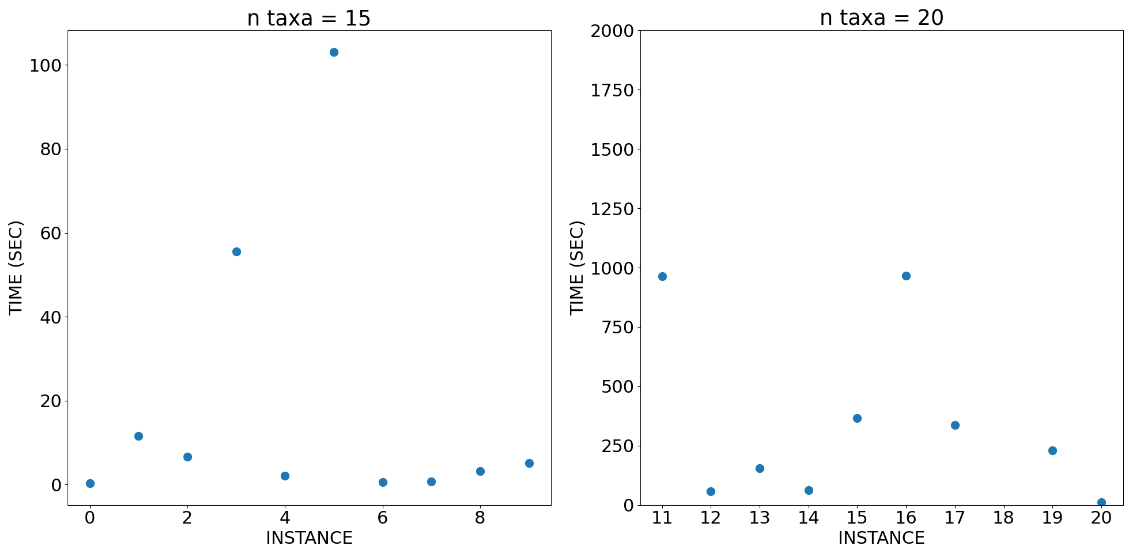


Figure 3.7: Computation time of the IP in the 15 and 20 taxa case.

how the choice of the initialisation heuristic affects the FastME performance, we tested all the SIA in 1000 instances of 150 randomly sampled taxa. As shown in Figure 3.8 the best performance is obtained 36, 1% of the times with BioNJ which seems to be the most reliable choice. This is in line with the developers' recommendation (Lefort et al. [2015b]) to use BioNJ as the initialisation algorithm. However, we have to remark that for the remaining 63, 9% of the instances, a better result would have been obtained with a different choice. In view of these results, one might wonder whether an optimal strategy for the SIA selection exists or whether it is possible to develop a stronger initialisation heuristic capable of outperforming the SIAs.

3.4 Random initialisation

To answer the question that arose at the end of section 3.3, we can start trying the simplest possible approach, which consists of initialising the BNNI and the BSPR with randomly generated trees. If we perform the same experiment made for testing the SIA, initialising 100 random trees per instance we obtain the interesting result shown in Figure 3.9. In the 95, 2% of the cases, we have that the best result is obtained with the random initialisation.

This fact is of particular importance, because it indicates that with a different initialisation approach the solution obtained by FastME can be improved.

Moreover, if we focus our attention on a single run, for example the last of the experiment, and we analyse the results of the 100 random initialisations, we find out that several iterations provided better solutions with respect to the best SIA (Figure 3.10).

However, a pure random initialisation approach (RI) might turn out quite weak, as if on the one hand it favours the exploration of the search space on the other it does not allow the exploitation of any information collected during the previous runs, since with the RI all the generated trees are independent. The natural research question is therefore if it is possible to design a strategy that is capable of finding a good trade-off between the exploration of the tree space and the exploitation of

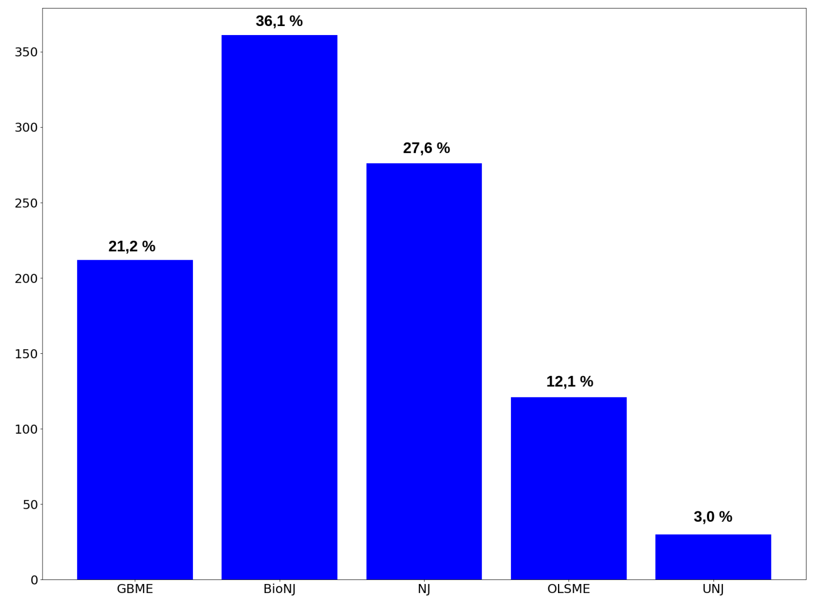


Figure 3.8: Number of instances in which the corresponding SIA provided the best FastME performance over 1000 runs on the *RDPII_F84* dataset considering for each run a random sample of 150 taxa.

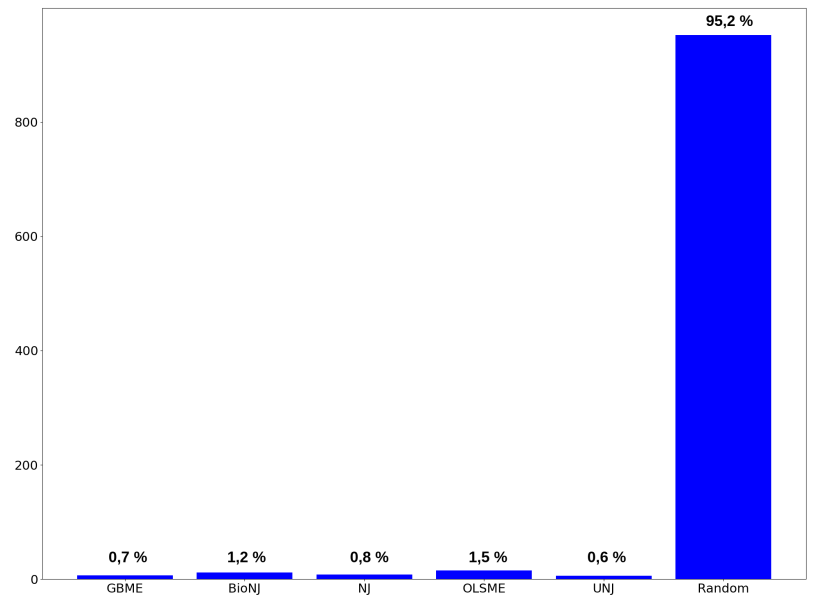


Figure 3.9: Statistics of the best FastME performance over 1000 runs on the *RDPII_F84* dataset considering for each run a random sample of 150 taxa.

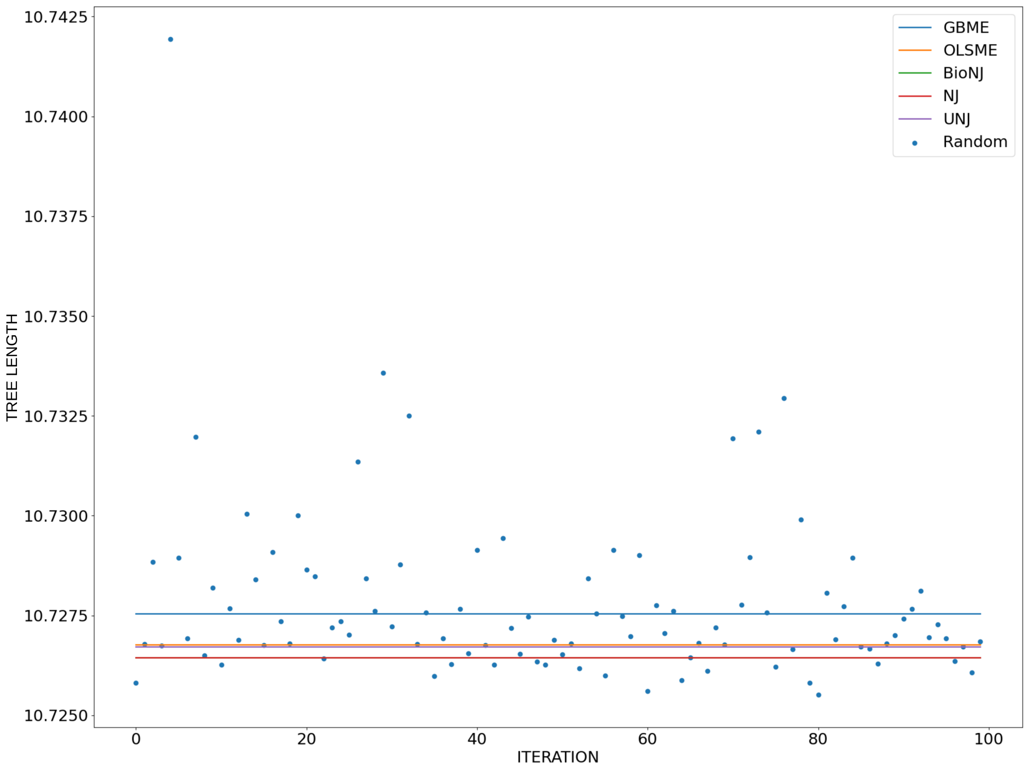


Figure 3.10: Statistics of the best FastME performance over 1000 runs on the *RDPII_F84* dataset considering for each run a random sample of 150 taxa.

the information retrieved during such exploration.

Chapter 4

The PhyloES algorithm

Evolutionary Algorithms (EA) are a class of metaheuristics that are inspired by natural processes and often used to solve complex real-life optimisation problems [Katoch et al., 2021]. EAs represent a powerful class of optimisation and search techniques inspired by the principles of biological evolution by simulating the process of natural selection and the survival of the fittest. Over the past few decades, EAs have gained widespread popularity and have been applied to a diverse range of domains due to their ability to adapt, explore, and exploit solution spaces which make them valuable tools for tackling real-world challenges. Their applications in fact cover a wide spectrum of fields, from engineering and biology to finance and machine learning as well as several others.

The concept of evolutionary algorithms can be traced back to the early 1960s when researchers like John Holland [Holland, 1975] and Ingo Rechenberg [Rechenberg, 1973] began exploring computational models inspired by natural evolution. John Holland's work on Genetic Algorithms (GAs) and Ingo Rechenberg's development of Evolution Strategies (ES) were pioneering efforts in this field. These early efforts laid the foundation for the development of various evolutionary algorithms.

In their general framework, EA are characterised by a set of key components that collectively drive the optimisation process:

- Population: EAs maintain a population of candidate solutions to the optimisation problem. Each candidate solution is represented as a potential solution to the problem.
- Fitness Function: A fitness function evaluates how well each candidate solution performs in solving the problem. It assigns a fitness score to each solution, which reflects its quality or suitability.
- Selection: The selection process simulates the natural selection mechanism, favouring solutions with higher fitness scores. Solutions with better fitness have a higher chance of being selected to form the next generation.
- Recombination (Crossover): In this step, pairs of selected solutions are combined to create new solutions, mimicking the genetic recombination observed in nature. This introduces diversity into the population.
- Mutation: Mutation introduces small, random changes to selected solutions, further diversifying the population and preventing premature convergence to suboptimal solutions.
- Stop Criteria: EAs continue evolving the population for a specific number of generations or until predefined termination criteria are met, such as a target fitness level or a time limit.

Evolutionary Algorithms encompass several sub-types, each tailored to specific problem domains and requirements. Genetic Algorithms (GAs) use a binary string representation for solutions and are well-suited for combinatorial optimisation problems. Genetic Programming (GP) extends the concept of GAs to evolve computer programs or symbolic expressions, making it suitable for symbolic regression and program synthesis. Differential Evolution (DE) instead, is a population-based optimisation algorithm particularly effective in continuous parameter space problems.

4.1 $(\mu + \lambda) Evolution Strategies$

The EA metaheuristics class is said to be *population-based*, i.e., it operates on a set (the *population*) of candidate solutions (the *individuals*) which are *evolved* (i.e., modified) iteratively over time. Their workflow is, as mentioned at the beginning, inspired by biological phenomena and can be summarised as follows: starting from an initial population they define ways to generate new individuals (*offspring*) through an evolutionary process at each iteration, called a *generation*, and a criterion to define the next generation of individuals (*selection*). These operations are repeated at each generation, by evaluating individuals according to a *fitness function* that represents the quality of a candidate solution, until a stopping criterion is met [Luke, 2013]. In this work, we will focus on *Evolution Strategy* (ES), which is a subclass of EAs that was first introduced in Rechenberg [1973] and that is characterised by two parameters μ and λ. In a $(\mu + \lambda)$-ES, the population is initialised by randomly sampling λ individuals; then, by means of a *mutation operator* (i.e. a procedure to convert a single individual into a new one by random changes) λ/μ children are created for each of the parents. Subsequently, the parents and the generated offspring undergo a so-called *truncation selection*, where only the μ fittest individuals survive and form the next generation while the rest are discarded. The process is then repeated with the previous generation's best individuals as the new parents. Algorithm 8 outlines the main steps carried out by the $(\mu + \lambda)$-ES [Luke, 2013].

Despite being very similar, ESs and GAs differ in the way in which they handle the selection

Algorithm 8 The $(\mu + \lambda)$ Evolution Strategy

```
Require: $\mu$, $\lambda$
  $P \leftarrow \{\}$
  for $\lambda$ times do
      $p_i \leftarrow$ random individual
      $P \leftarrow P \cup \{p_i\}$
  end for
  Best $\leftarrow$ None
  while stop criterion is False do
      $Q \leftarrow P$
      for each $p_i \in P$ do
          for $\lambda/\mu$ times do
              $Q \leftarrow Q \cup \{\text{Mutate}(\text{Copy}(p_i))\}$
          end for
      end for
      $P \leftarrow$ the $\mu$ individuals in $Q$ whose fitness function are smallest
      Best $\leftarrow$ BestIndividual($P$)
  end while
  return Best
```

phase: ESs first select the parents and then create the children: instead, the GAs step by step select two parents and generate multiple children until enough children have been created (Luke [2013]).

The main advantage of such an approach is represented by the fact that in our case we have at our disposal two fast local search heuristics, the BNNI and the BSPR, that can be exploited to guide the population development toward the most promising regions of the search space. This can be simply achieved by performing BNNI and BSPR on each new individual and adding the resulting tree to the population. In this manner, while the offspring favours the exploration of the search space, the local search algorithms ensure keeping a good *quality* of the population accelerating the improvement of the current best solution.

We remark that while EAs have been extensively applied in the literature on phylogenetics, including the works of Noutahi and El-Mabrouk [2018], Zwickl [2006], Kosakovsky Pond et al. [2006], it is worth noting that most of these works refer to genetic algorithms that have been designed for phylogeny estimations under the ML framework, (Felsenstein [2004], Lewis [1998], Matsuda [1996], Poladian and Jermiin [2006], Brauer et al. [2002], Skourikhine [2000], Helaers and Milinkovitch [2010], Zwickl [2006]). To the best of our knowledge, the algorithm that we present here is the first application of ES in phylogeny estimation, and the first EAs approach for distance methods, in particular for the BMEP.

4.2 PhyloES

As seen in section 3.4 an iterative RI for the FastME heuristic leads to better solutions with respect to the SIA initialisation. If on one hand, this fact represents itself an improvement, on the other a random initialisation approach makes it hard to establish the number of iterations required to obtain a satisfactory solution. Indeed, it does not allow the exploitation of any information acquired in the previous iterations, as all random tree samples are independently generated. In order to provide a workaround to the random approach limitations and find a compromised approach between exploration and exploitation, in this chapter we discuss in detail *PhyloES*, an ES-based algorithm that is capable of consistently improving both the FastME and the RI performance.

Evolutionary algorithms often make use of an encoded representation of the individuals in order to simplify the recombination and mutation operations. Since it is not trivial how to implement such operations on trees, in the next section we will define a tree encoding that will provide a simple rule to define the mutation and the recombination of trees.

4.3 Tree Encoding

The problem of encoding trees has been widely studied in the literature [Caminiti et al., 2007, Catanzaro and Pesenti, 2019] and several tree codes have been proposed [Prüfer, 1918, Neville, 1953]. However, the encoding method proposed in Rohlf [1983] is specific for unrooted binary trees and it offers two main advantages: it provides a straightforward method for the random generation of phylogenies, and it allows to manipulate trees operating directly on the encodings, by ensuring that when a component of the code of a phylogeny is altered, the result is still a phylogeny. Rohlf [1983]'s encoding defines a map ϕ between the edge set E of a tree and the set of coding vectors H whose elements are of the form $(h_1, \dots, h_{n-3})$, where n is the number of leaves and $h_k \in \{1, 2, \dots, 2k-3\}$. Such a function ϕ is proven to be well-defined and bijective as long as a well-defined labelling of the edges is given [Rohlf, 1983].

In order to develop efficiently our ES algorithm we made use of an *ad hoc* representation of the phylogenetic trees inspired by the work of Rohlf [1983]. The encoding exploits the fact that any phylogeny can be constructed in a step-wise fashion, starting from an initial star tree composed of only three taxa and a single internal node, and adding one by one all the remaining taxa by selecting at each step an insertion edge (see figure 4.1). If an edge labeling is well defined it is possible to represent a tree with the edge selected for its construction. Formally: we encode any tree of n taxa with a vector $(h_1, \ldots, h_{n-3})$ of $n-3$ elements, where h_i represents the label of the edge selected at the $i-th$ step for the insertion of the $(i+3)$-th taxon. In our case, for a tree T with taxa $t_1, \ldots t_n$ and internal nodes $i_{n+1}, \ldots i_{2n-3}$ where i_k is the internal node inserted at the k-th step of the tree construction, the label of an edge is defined with its index in the edge list E, which is unique as we order the elements of E first according to the lowest index of the two nodes defining each edge, and in case of ambiguity according to the greatest.

The advantage of using such an encoding approach is that it provides a straightforward method for the random generation of phylogenies, and it allows the tree manipulation operating directly on the encodings, by ensuring that when a component of the code of a phylogeny is altered, the result is still a phylogeny.

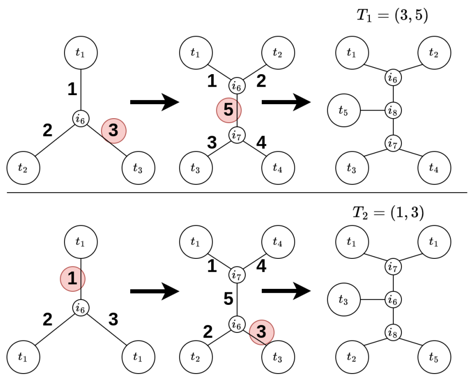


Figure 4.1: Two examples of tree encoding. (TOP) The ordered edge list at the initial step is $E = \{(t_1, i_6), (t_2, i_6), (t_3, i_6)\}$. The edge selected for the insertion of t_4 is (t_3, i_6), which is the 3-rd in the list. At the next step E becomes $\{(t_1, i_6), (t_2, i_6), (t_3, i_7), (t_4, i_7), (i_6, i_7)\}$; t_5 is then iserted in the edge t_5 is (i_6, i_7), 5-th in the list, so the resulting code is $T_1 = (3, 5)$. (BOTTOM) The initial edge list is $E = \{(t_1, i_6), (t_2, i_6), (t_3, i_6)\}$. t_4 is inserted in (t_1, i_6), 1-st in the list. At the next step E becomes $\{(t_1, i_7), (t_2, i_6), (t_3, i_6), (t_4, i_7), (i_6, i_7)\}$; the edge selected is (t_3, i_6) , 3-rd in the list, providing the resulting code is $T_2 = (1, 3)$.

4.3.1 Internal edges labels

For our encoding, we first assume to work with an ordered set X of n taxa, $t_1, t_2, \ldots, t_n$. We define the label of an edge as its index in the edge list E of a phylogenetic tree, which is ordered first according to the lowest index of the two nodes defining each edge, and in case of ambiguity according to the greatest. To finally guarantee the uniqueness of the edge labels we have to ensure that a well-defined labelling of the internal nodes is given. We achieve so by considering the step-wise tree construction method that, starting from an initial star tree composed of the first three taxa and a single internal node, adds one by one all the remaining taxa. All internal nodes are labelled according to their insertion order, starting from the index $k = n + 1$ up to the index $k = 2n - 2$ (where n is the number of taxa).

4.3.2 Decoding

ϕ^{-1}, the decoding map described in algorithm 9, allows us to retrieve the edge set of a tree T of n taxa given its code h. Since in our case h represents the instructions for the construction of T we simply need to build E step-by-step according to h, starting from the initial star composed by the first three taxa. At each step k we remove the selected edge h_k, we add the two new edges and we fix the order of E. Algorithm 9 shows the details of the decoding procedure, which we will now analyse in depth. From now onwards we will indicate a taxa with the letter t and internal nodes with the letter i.

Algorithm 9 ϕ^{-1}, Decoding

```
Require: $h$, $\{t_1, \dots, t_n\}$,
1: $E := \{(t_1, i_{n+1}), (t_2, i_{n+1}), (t_3, i_{n+1})\}$ (the initial star tree)
2: for $k = 1, k \leq n-3, k++$ do
3:     $(v_\alpha, i_\beta) = E[h_k]$
4:     InsertTaxonEdge($(t_{k+3}, i_{n+1+k})$)
5:     Set $(v_\alpha, i_\beta) = (v_\alpha, i_{n+1+k})$
6:     FixPosition($(v_\alpha, i_{n+1+k})$)
7:     Insert($(i_\beta, i_{n+1+k})$)
8: end for
9: return E
```

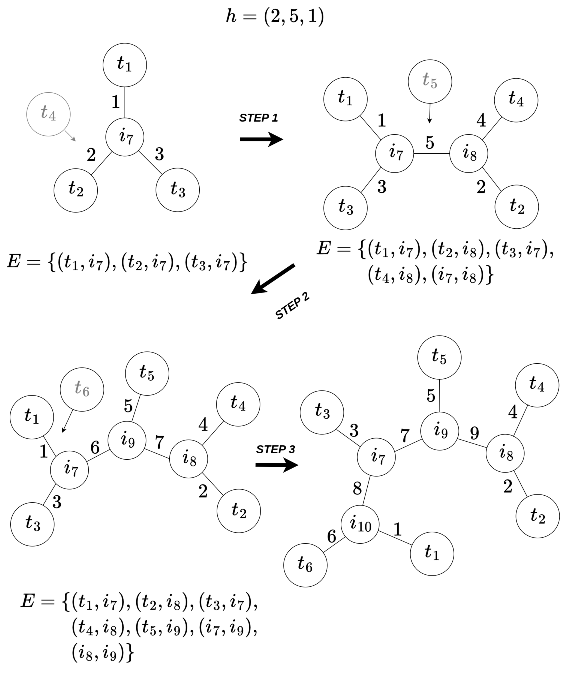
At the initial step $E = \{(t_1, i_{n+1}), (t_2, i_{n+1}), (t_3, i_{n+1})\}$; h, which elements are in $\{1, 2, 3\}$, indicates the index in E of the edge selected for the insertion of t_4 and the internal node i_{n+2}. Referring to the example of Figure 4.2, in which $h = (2, 5, 1)$, the edge selected is $E[2] = (t_2, i_7)$. The function **InsertTaxonEdge** (algorithm 10) adds the new edge in the list after the only edge connecting the previous taxa t_3, (t_3, i_7), obtaining $E = \{(t_1, i_7), (t_2, i_7), (t_3, i_7), (t_4, i_8)\}$. This insertion preserves the correct ordering of E since the edges having taxa as the first component, $(t_*, *)$, always come before the others, which are of the form (i_α, i_β). This is always true as we established that taxa are indexed from 1 to n and internal nodes form $n+1$ to $2n-2$.

Algorithm 10 InsertTaxonEdge((t_{k+3}, i_{n+1+k}))

```
$(t_{k+2}, *)$ = unique edge connecting leaf $t_{k+2}$
Insert $(t_{k+3}, i_{n+1+k})$ in $E$ after $(t_{k+2}, *)$
```

Then, the selected edge (t_3, i_7), has to be removed and its nodes will be connected with the new internal node i_8 forming two new edges. Therefore, (t_3, i_7) can be simply replaced by (t_3, i_8), (line 5 in algorithm 9). In this case, E is still ordered so no fixing is needed. However, in general, it might be necessary to adjust the position of the new edge, a task which is handled by the function **FixPosition** (algorithm 11). In fact, as shown in figure 4.3, if the selected edge (i_α, i_β) is connecting two internal nodes i_α and i_β, and if i_α is linked with some other internal node i_γ with $\gamma > \beta$ and the node $*$ is either a leaf or an internal node with index greater than γ, we have that the order of $E = \{\dots, (i_\alpha, i_{n+1+k}), (i_\alpha, i_\gamma,), \dots\}$ after the replacement, must be fixed swapping (i_α, i_{n+1+k}) and (i_α, i_γ), since $i + 1 + k > \gamma$. In the case in which $*$ is also an internal node with index greater than γ, an additional swap between (i_α, i_{n+1+k}) and $(i_\alpha, *)$ has to be performed.

The last line of the loop of ϕ^{-1} is the insertion in E of the edge (i_β, i_{n+1+k}), handled by the function **Insert** (algorithm 12). Here there are three cases: if $k = 1$, the edge (i_{n+1}, i_{n+2})



$h = (2, 5, 1)$
STEP 1
STEP 2
STEP 3
$E = \{(t_1, i_7), (t_2, i_7), (t_3, i_7)\}$
$E = \{(t_1, i_7), (t_2, i_8), (t_3, i_7), (t_4, i_8), (i_7, i_8)\}$
$E = \{(t_1, i_7), (t_2, i_8), (t_3, i_7), (t_4, i_8), (t_5, i_9), (i_7, i_9), (i_8, i_9)\}$

Figure 4.2: Example of the tree construction and encoding for a phylogeny with six taxa. Starting from a star-tree with three taxa connected to a single inner node, the remaining taxa are added to the tree iteratively and both the edge set E and the encoding H are updated at each addition.

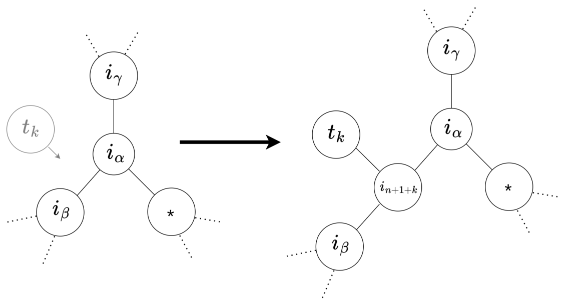


Figure 4.3: In case of $\gamma < \beta$, the order of E must be adjusted.

Algorithm 11 FixPosition((v_α, i_{n+1+k}))

```
$(v_\alpha, *)$ = last occurrence in $E$ of an edge with first component $i_\beta$
insert $(v_\beta, i_{n+1+k})$ in $E$ after $(v_\alpha, *)$
```

is simply added to the end of the list. If $k > 1$ and no edge in E has i_β as first component, (i_β, i_{n+1+k}) must be inserted in E after $(i_\gamma, *)$, where i_γ is the internal node with the greatest index such that $\gamma < \beta$ and $*$ is the internal node connected to i_γ with the greatest index. This is the case of (i_8, i_9) in the second step of the example of figure 4.2; before its insertion, $E = \{(t_1, i_7), (t_2, i_8), (t_3, i_7), (t_4, i_8), (t_5, i_9), (i_7, i_9)\}$; since no edge E has i_8 as first component and i_7 is the internal node with the greatest index lower than 8, (i_8, i_9) is inserted after (i_7, i_9). The last case is the one in which there are in E some edges with i_β as the first component. If so, (i_β, i_{n+1+k}) has to be inserted after the last of them, a task performed by function 11.

Algorithm 12 Insert((i_{n+1+k}, i_β))

```
1: if k=1 then
2:     insert $(i_{n+1}, i_{n+2})$ at the end of $E$
3: else if any $(i_\beta, *)$ not in $E$ then
4:     $(i_\gamma, *)$ = edge in $E$ s.t. $\gamma$ is the greatest index with $\gamma < \beta$
5:     insert $(i_\beta, i_{n+1+k})$ after $(i_\gamma, *)$
6: else
7:     FixPosition($(i_\beta, i_{n+1+k})$)
8: end if
```

Decoding complexity

The decoding algorithm, ϕ^{-1} is composed of a *for* loop of $n-3$ ($O(n)$) iterations. The function **InsertTaxonEdge** (algorithm 10) requires constant time since the position of each node in the list can be traced out and reached in O(1). **FixPosition** (algorithm 11) needs to iterate through E starting from (v_α, i_{n+k+1}) to find the last occurrence of an edge with first component equal to v_α; since any node is linked to at most three other nodes, the function is $O(1)$. Concerning function **Insert** (algorithm 12), apart from the one in line 4, all operations require $O(1)$. Since in line 4 the index γ is unknown, it must be searched in an ordered list, an operation that requires $O(\log(n))$ operations. Due to the above reasoning the whole complexity of ϕ^{-1} is $O(n \log(n))$.

4.3.3 Coding

The coding algorithm, ϕ, maps the edge set of a phylogenetic tree into its coding vector h. We remark that in order to obtain the correct code of a tree two conditions must be ensured:

1. The internal node labels must be consistent with the one chosen for our encoding.
2. E must be ordered primarily according to the edges' first component index and, when equal, according to the second component index.

Concerning 1), we notice that our internal node labels are determined by the phylogeny step-wise tree construction, that is the one followed in the decoding algorithm and for which the label of the internal node inserted at the construction step k is $n+k+1$. Consequently, the edge list E of a phylogeny should always contain the edge (t_n, i_{2n-2}), which is the one added to the list in the last step. If we have instead that t_n is linked to i_α with $i_\alpha \neq i_{2n-2}$ we can fix the inconsistency by swapping the labels of the two internal nodes. If we then remove the edge (t_n, i_{2n-2}) from E we should have that t_{n-1} is linked i_{2n-3}: if this is not the case, we can again fix the inconsistency by performing the index swap between i_{2n-3} and the internal node connected to t_{n-1}. To complete the adjustment of the internal node labels, as described in algorithm 13 this procedure has to be performed for all the remaining steps. We remark that at each step, when we remove the edge (t_k, i_{n+k+1}), the tree must be reconnected. This is done connecting the two nodes v_α and i_β, with $\alpha < \beta$, previously linked to i_{n+k+1} (figure 4.4).

Algorithm 13 FixInternalNodesLabels

```
Require: $E$
 1: for $k = n, k \geq 4, k--$ do
 2:     $i_\gamma$ = internal node connected to $t_k$
 3:     $v_\alpha, i_\beta$ = nodes connected to $i_\gamma$ different to $t_k$
 4:     if $\gamma = n + k - 2$ then
 5:         swap indexes between $i_\gamma$ and $i_{n+k-2}$
 6:     end if
 7:     Remove $(t_k, i_{n+k-2})$ and $(i_\beta, i_{n+k-2})$ from $E$
 8:     Set $(v_\alpha, i_{n+k-2}) = (v_\alpha, i_\beta)$
 9: end for
10: return E
```

Once ensured condition 1), condition 2) can be guaranteed by sorting E.

The idea behind the coding procedure (algorithm 14) is to invert the flow of the decoding algorithm in a similar fashion to algorithm 13, and to retrieve h deconstructing the tree by removing one-by-one all taxa from t_n to t_4. The main difference with algorithm 13 is that each time that an edge is removed the order of E must be restored. This might happen when reconnecting nodes v_α and i_β, in the case in which v_α is an internal node connected with some other internal node i_γ such that $\gamma > \beta$. In that case, the function 15 ensures the correct ordering by applying the appropriate swaps. Since finally E is correctly ordered, the component h_k is given by the index of (v_α, i_β) in E which is the edge between the nodes v_α and i_β previously connected to i_{n+k+1}.

Algorithm 14 ϕ

```
Require: $E$
 1: for $k = n, k \geq 4, k--$ do
 2:     $v_\alpha, i_\beta$ = nodes connected to $i_{n+k-2}$
 3:     Remove $(t_k, i_{n+k-2})$ and $(i_\beta, i_{n+k-2})$ from $E$
 4:     Set $(v_\alpha, i_{n+k-2}) = (v_\alpha, i_\beta)$
 5:     FixPosition2($(v_\alpha, i_\beta)$)
 6:     $h_k$ = index of $(v_\alpha, i_\beta)$ in $E$
 7: end for
 8: return E
```

Coding complexity

The whole coding procedure is composed of three parts: the function **FixInternalNodesLabels** (algorithm 13), the sorting of E and the function ϕ (algorithm 14). **FixInternalNodesLabels** consists of a for loop composed of only O(1) operations, so it requires $O(n)$ operations. Sorting E instead requires $O(n\log(n))$. In the for loop of ϕ all operations but the one of line 6 require $O(1)$; the index retrieval of line 6 requires instead $O(\log(n))$ operations since it is a search in an ordered

Algorithm 15 FixPosition2((v_α, i_β))

```
$v_{\gamma1}, v_{\gamma2}$ = two nodes connected to $v_\alpha$ different from $i_\beta$, with $\gamma1 < \gamma2$
if $\gamma2 > \beta$ then
    swap $(v_\alpha, v_{\gamma2})$ and $(v_\alpha, i_\beta)$ in $E$
    if $\gamma1 > \beta$ then
        swap $(v_\alpha, v_{\gamma1})$ and $(v_\alpha, i_\beta)$ in $E$
    end if
end if
```

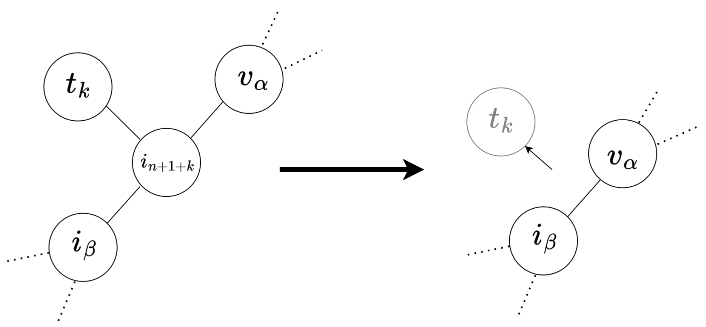


Figure 4.4: Removal of taxon t_k

list. The whole cost of ϕ is therefore $O(n\log(n))$. In conclusion, all three steps of the coding require in total $O(n\log(n))$.

4.3.4 PhyloES encoding advantages

As mentioned here in section 4.3, the encoding we developed turns out particularly handy to generate and manipulate trees. The decoding algorithm (14) provides in fact a constructive procedure to generate a random tree of n taxa, by sampling a vector of $n-3$ components such that:

$$\begin{aligned} h_1 &= \{1,2,3\} \\ h_2 &= \{1,2,3,4,5\} \\ &\dots \\ h_k &= \{1,\dots,2k-3\} \\ &\dots \\ h_{n-3} &= \{1,\dots,2n-6\} \end{aligned} \tag{4.1}$$

In addition, in the PhyloES offspring phase, a new individual is generated defining its coding vector h, which components are obtained by sampling from the components of the whole population. Since any vector of the form described in equation 4.1 is a valid tree we are ensured that such a new individual is a valid tree. Figure 4.5 shows an example of the generation of individuals in the case of a population composed of three trees of eight taxa.

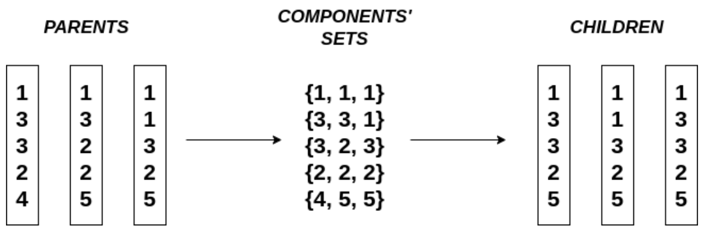


Figure 4.5: Example of generation of a new individual by sampling each component from the population.

4.4 Phylogenetic Inference with ES

To harness the exploration power of ES (Section 4.1) and the efficiency of the BNNI and BSPR tree rearrangement algorithms (Section 3.1) in this section we describe a novel algorithm, PhyloES, for the BMEP. PhyloES is an ES, with μ and λ both set equal to the population size, specifically designed for the BMEP that introduces crucial modifications to Algorithm 8 allowing an efficient exploration of the search space by using tree rearrangement operators to perform *local search*, with the aim of steering the mutation process in order to obtain a better solution. Algorithm 16 illustrates the proposed algorithm and the following paragraphs describe in detail each of its key components.

Population Initialisation As in Algorithm 8, the population is initialised by sampling μ random phylogenies and applying both BNNI and BSPR to each tree. Furthermore, to foster the initial exploration of the search space we employ a population size decreasing over time according to a halving strategy (Hallam et al. [2010]).

Offspring In the usual ES algorithm blueprint described in Section 4.1, the mutation operator is used to generate new offspring. Mutation yields individuals by introducing random modifications in the encoding of a member of the previous generation's population. Since PhyloES aims to exploit the similarities between trees resulting from the BNNI and BSPR, the mutation operator is replaced by a *tree generation* operator which takes into account not just a single member but all trees in the population and uses their encoding to construct a new tree. More in detail: each element j of a new individual's encoding is determined by sampling from the set composed by the j-th element of all the trees of the previous generation. In mathematical terms, let $\mathcal{T}^g = \{T_1^g, \ldots, T_k^g\}$ be the set of k trees with n taxa which represent the population at generation g, and let $e(T_i^g) = (h_{i1}^g, \ldots, h_{in-3}^g)$ be the encoding of T_i^g, with h_{ij} its j-th component. Then, a mutated individual T_i^{g+1} is obtained from $\mathcal{T}^g$ by sampling each of its components as

$$h_{ij}^{g+1} = \text{random sample from } \{h_{1j}^g, \ldots, h_{kj}^g\} \tag{4.2}$$

After offspring have been generated, local search is employed to improve the new phylogenies.

Fitness function The fitness is simply defined as the BMEP tree length in Equation (1.6).

Selection and Stop Criteria In PhyloES truncation selection is performed as described in Section 4.1: the μ individuals with the lowest fitness value (BMEP length function value) among the previous population and the newly created offspring become the next generation. Regarding the stop criteria, we consider three possibilities: 1) *convergence*, the algorithm runs until all the individuals

Algorithm 16 PhyloES

```
Require: $\mu$, $\max_{\text{iter}}$, $tol$
  $P \leftarrow \{\}$
  for $\lambda$ times do
      $p_i \leftarrow$ random phylogeny
      $p_i \leftarrow$ RearrangeTree($p_i$)
      $P \leftarrow P \cup \{p_i\}$
  end for
  Best $\leftarrow$ BestIndividual(P)
  Worst $\leftarrow$ WorstIndividual(P)
  $g \leftarrow 0$
  while $|P| > 1$ and $g < \max_{\text{iter}}$ and $l(\text{Worst}) - l(\text{Best}) > tol$ do
      $Q \leftarrow P$
      for each $p_i \in P$ do
          $q_i \leftarrow$ GenerateTree($P$)
          $q_i \leftarrow$ RearrangeTree($q_i$)
          $Q \leftarrow Q \cup \{q_i\}$
      end for
      $P \leftarrow$ the $\mu$ individuals in $Q$ whose fitness function are smallest
      $P \leftarrow$ IndividualReplacement($P$)
      Best $\leftarrow$ BestIndividual(P)
      Worst $\leftarrow$ WorstIndividual(P)
      $g \leftarrow g + 1$
  end while
  return Best
```

in the population are identical; 2) *maximum iterations*, a fixed number of iterations (max_{iter}) is performed, which is the most common approach in evolutionary computation; 3) *tolerance*, execution stops when the difference, in terms of tree length, between the current best and worse individuals in the population is lower than a given threshold tol.

4.5 Individuals replacement

In the procedure described above, the simple truncation selection of the best μ individuals within P to define the new generation sometimes leads the algorithm to suffer from "stagnation", i.e., the situation in which the new generation remains identical to the previous one, consequently slowing down convergence. Furthermore, this phenomenon tends to occur when the population P has multiple instances of its worst individual, which increases the probability of replicating it and decreases the chances of producing better individuals. To overcome this issue, we introduced a simple adjustment on top of the usual truncation selection criterion, similar to the one introduced by Bartoli et al. [2019]. The authors propose to reduce the number of duplicate individuals in the population by modifying the reproduction phase: whenever they generate a new individual, they check if it is unique in the merged set of parents and already generated offspring; if not, they drop it and reapply the genetic operator. Instead, at each generation of PhyloES, we check in the set P for multiple occurrences of the worst individual in terms of tree length and, if any, we replace one of them with a copy of the second worst element. This modification helps avoid stagnation as, if no improvement in the population is gained after one generation cycle, it favours a gradual shift to the replication of the best individual and drives the algorithm towards reaching the convergence stop criteria.

Here we provide the pseudo-code of the individual replacement algorithm used in PhyloES.

Algorithm 17 IndividualReplacement

```
Require: $P$, the current tree population
  sort $P$ by BMEP length value
  $w$ = worst individual in $P$
  if $w$ has multiple occurrences in $P$ then
      $rep$ = second worst individual in $P$ s.t. $w \neq rep$
      replace the first occurrence of $w$ with $rep$
  end if
  return $P$
```

To better understand the flow of algorithm 17, let us suppose to have the following population of trees $P = a, b, c, d, e, f, f$, where elements are ordered by tree length. The worst individual f has two occurrences, so its first occurrence in P is replaced e, which is the second worst individual, obtaining $P = a, b, c, d, e, e, f$. In this manner, any new individual is less likely to be similar to f decreasing the chances of generating full or partial replicas of f, a fact which would slow down the convergence of the algorithm. The benefit of this adjustment has been empirically proven in our experiments, but a detailed analysis of the advantages of such a technique is beyond the scope of this article.

4.6 Implementation details

The PhyloES Python interface provides not only a simple tool to set up and run phylogenetic analyses but also allows efficient exploitation of computational resources. In particular, the algorithm can be conceptually divided into two main tasks: the ES handling and the local search computations. The first consists of the tree coding and decoding and the offspring operations. All these sub-tasks are characterised by the fact that they can be easily vectorised and performed in batches, i.e. multiple instances can be processed at the same time, allowing the exploitation of GPU resources, particularly suitable for the parallelisation of batch vector operations. For this reason, we implemented the entire ES workflow with Pytorch. The second task instead consists of the BNNI and the BSPR computations. As long as both algorithms make use of tree data structures and perform the tree rearrangement operations via recursive functions, we developed a C++ extension of BNNI and BSPR, that optimises their parallelisation. The extension is called by the python interface via the python-c binding library *ctypes*. Such a hybrid Phython-C++ approach allowed us to customise for our purposes the open-source BNNI and BSPR implementations provided by the FastME software and to adapt the two algorithms in order to enable the process of several trees in parallel while keeping a user-friendly interface.

Chapter 5

Results

In this chapter, we report on the results obtained by running a number of computational experiments on a set of benchmark instances of the BMEP. These experiments were designed to answer the following *research questions*:

RQ1 How far is PhyloES from the optimal solution?

RQ2 Can PhyloES improve on the solutions found by FastME in terms of tree length also for larger datasets?

RQ3 Is the proposed ES approach effective in exploring the search space?

RQ4 How do PhyloES tree solutions differ topologically from those of FastME?

RQ5 Are the observed results reliable from the numerical precision perspective?

In order to evaluate the performances of PhyloES we used as a reference the length of the solutions provided by FastME. Moreover, in order to assess the effectiveness of our evolutionary strategy we compared it versus a pure random initialisation approach (RI), consisting of initialising the BNNI and BSPR with random trees.

5.1 Experimental Setting

For RQ1 the tests have been made with the same methodology and data as described in Section 3.3.1. In particular, we remind that in order to speed up the computation of the IP model, as in Section 3.3.1, the solver has been provided with the solution obtained with PhyloES. Instead, the benchmark that we used to address RQ2, RQ3, RQ4, and RQ5 consists of two datasets used in Stamatakis [2005] and in Guindon et al. [2010], and available at *https://github.com/stamatak/test-Datasets*. More in detail, we extrapolated three datasets, named 100_RDPII (selecting the first 100 taxa from RDPII), 200_RDPII (selecting the first 200 taxa from RDPII) and 300_ZILLA (selecting the first 300 taxa from ZILLA). For each dataset, we generated four distance matrices computed via the FastME software and using respectively the JC69 [Jukes and Cantor, 1969], K2P [Kimura, 1980], F81 [Hasegawa et al., 1981] and F84 [Felsenstein and Churchill, 1996] substitution models, always performing pairwise gap removal. All experiments have been run on an Intel Core(TM) i7-10700 (2.90GHz) 16 cores machine with a GeForce RTX 2070 SUPER GPU. Concerning the PhyloES hyperparameters, $\max_{\text{iter}}$ and *tol* have been respectively set to 1000 and 10^{-12}. In addition, to favour a wider initial exploration of the search space the population size has been set to 64 for the first 5 iterations, 32 from the 5-th to the 25-th iteration and 16 from the 25-th iteration onward.

In order to ensure a fair comparison, for each instance we tested all the SIA of FastME and the best solution obtained out of the five initialisation algorithms was the one taken into account for our analysis. We remark that this procedure was performed only once per instance as regardless of the chosen SIA, FastME is a deterministic algorithm and it does not benefit from multiple runs. Instead, the RI algorithm and PhyloES have been run 10 times for each of the 12 problems and the collected data have been used to produce the analysis and the statistics shown in this chapter. Furthermore, when comparing PhyloES to the RI, we first run our algorithm and count the amount of generated trees, and then we perform the same number of RI iterations.

5.2 RQ1, Comparison with the IP model

Looking at Table 5.1 we observe that PhyloES finds the optimal solution in all instances in which the IP was able to terminate the computation within the time limit. In these cases, PhyloES was on average 97, 3% times faster than the IP (std 0.06%). In addition, we also notice that in the remaining 8 instances, the IP solver reached the time limit without being able to provide any improvement with respect to the initial solution provided by PhyloES. We also report that even though by a very limited gap, in 2 instances PhyloES outperformed FastME (runs 22 and 26).

5.3 RQ2, Tree length analysis

In Table 5.2 we report, for each of the considered datasets, the performance of FastME, PhyloES, and the RI. For FastME, we indicate the BMEP length function value of the best solution found and the SIA that led to the solution. Instead, for PhyloES and the RI, we outline the average BMEP length value of the best solution, its standard deviation, the number of unique and distinct best phylogenies across the different runs and the average improvement with respect to FastME (where a negative value indicates a lower BMEP length value than FastME). The results in Table 5.2 show that PhyloES, on average, either matches or outperforms FastME in terms of tree length. In the 100 taxa problems, PhyloES achieves a better solution with the F81 and JC69 datasets while in the other cases performs on par with FastME. As the number of taxa increases, the difference between the two methods becomes more significant with an average improvement of 0.04% over FastME. Again, with the F81 and JC69 datasets, there is a larger gap, yet the same can be stated for the RI which probably indicates that these instances are easier to solve by exploring the solution space utilizing the considered local strategies. However, if we consider the single substitution models, there is a clear trend in the improvement which confirms the effectiveness of exploration strategies for large search spaces. This observation is confirmed also by the RI behaviour which, in terms of tree length improvement with respect to the FastME solutions, follows a similar trend.

5.4 RQ3, Search space exploration

Comparing PhyloES with the RI we can see that, except for the 100 taxa datasets in which they attain the same performance, the evolutionary approach we propose leads to a clear improvement over the simple RI strategy. Additionally, our method shows more consistency in the final length value with respect to the RI. In fact, from Table 5.2 it is noticeable that on half the problems PhyloES produces the same final solution throughout the 10 runs whereas, on the other problems, a limited number of different solutions is found. On the contrary, the RI yields more distinct phylogenies in the 100 taxa problems while in the 200 and 300 taxa datasets show no consistency at all, providing at each run a different solution. Furthermore, we can also inspect this phenomenon by turning our attention to the standard deviation of the best length value attained by each method. In those cases where PhyloES leads to multiple solutions, their tree length lies in a very small interval, hence a standard deviation in the order of 10^{-5}. Conversely, the RI standard deviation tends to be an order of magnitude greater, which indicates a larger variability in the quality of the results.

To conclude, we also report that, except for a single run on the 200_RDPII F81 dataset in which the algorithm stopped due to the minimum tolerance criteria, in all the remaining runs PhyloES stopped due to convergence.

To analyse the scalability of our approach, we study the dependency of the computational effort required by PhyloES, with respect to the instance size, and its variability, as well as how it compares with the RI. It is worth mentioning that since PhyloES consists of multiple runs of BNNI and BSPR it cannot outperform FastME in terms of computational time. In Table 5.3 we detail, for each problem instance, the number of trees evaluated by both algorithms and for each one we also indicate the average and standard deviation of the execution time (in seconds), the average total number of NNI and SPR calls per run as well as the average number of generations carried out by PhyloES. Instead, in Figure 5.1 we display the trend of the PhyloES's number of NNI and SPR calls along the generations. From Table 5.3 we see how PhyloES significantly outperforms RI also in terms of computational time, showing a lower average time to solution and a smaller standard deviation. A straightforward explanation for such a difference can be found in the number of NNI and SPR calls reported in the table. On average, given the same amount of initial trees, PhyloES performs fewer iterations of the local search operators with respect to RI resulting in a much smaller computation time even though there is overhead due to the ES approach, *id est* the coding, decoding and tree recombination phase. The reason for this phenomenon lies in the fact that at each generation, the initial population is composed of individuals that are the recombination of previous BNNI and BSPR neighbours hence they are much more likely to be already quite balanced (having lower tree length) compared to a set of randomly generated trees, as in the case of the RI initial trees. This fact is even more evident from the curves in Figure 5.1, in which we can notice how, for all datasets, the number of BNNI and BSPR iterations drops immediately after the first generation and then gradually decreases at each generation. These results show the effectiveness of the ES we adopt, by employing an evolutionary approach PhyloES is able to efficiently explore the solution space by requiring fewer NNI and SPR iterations.

5.5 RQ4, Topological structure analysis

To further investigate the difference of the solutions provided by PhyloES and FastME we now focus our attention on the quality of the obtained solution trees, meaning that we aim to analyse how much the phylogenetic trees obtained differ with the two algorithms. This is of particular importance as long as our target is not the tree length per se but the underlying phylogeny. In particular, in those cases in which several taxa present a similar distance within each other, different phylogenetic reconstructions might be characterised by a very close tree length while heaving a quite different tree structure.

Toward this aim, we make use of the Robinson-Fould (RF) Distance [Robinson and Foulds, 1981], which defines the distance between two phylogenetic trees as the minimum number of edit operations, edge contractions or/and edge extensions, needed to convert one into the other. An example of the RF computation is given in figure 5.2

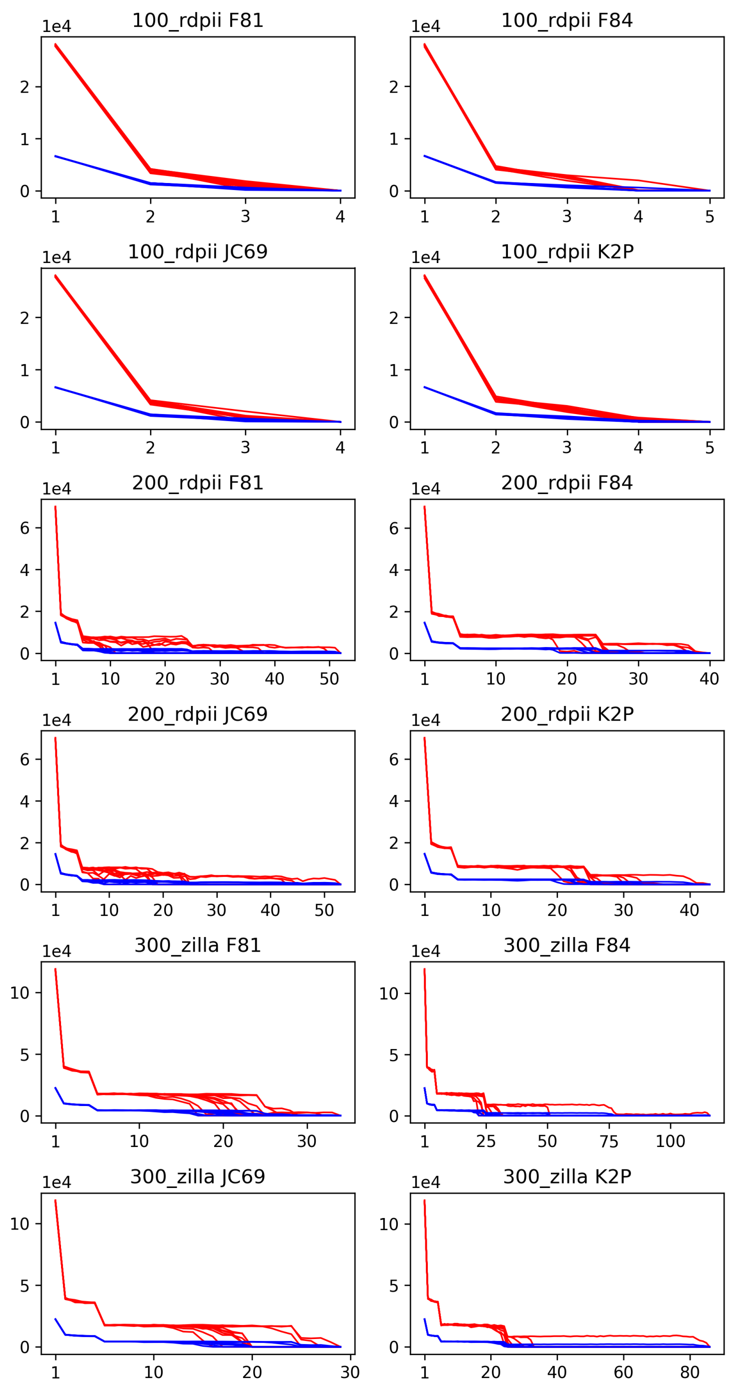


Figure 5.1: Number of BNNI (red line) and BSPR (blue line) iterations per generation in PhyloES.

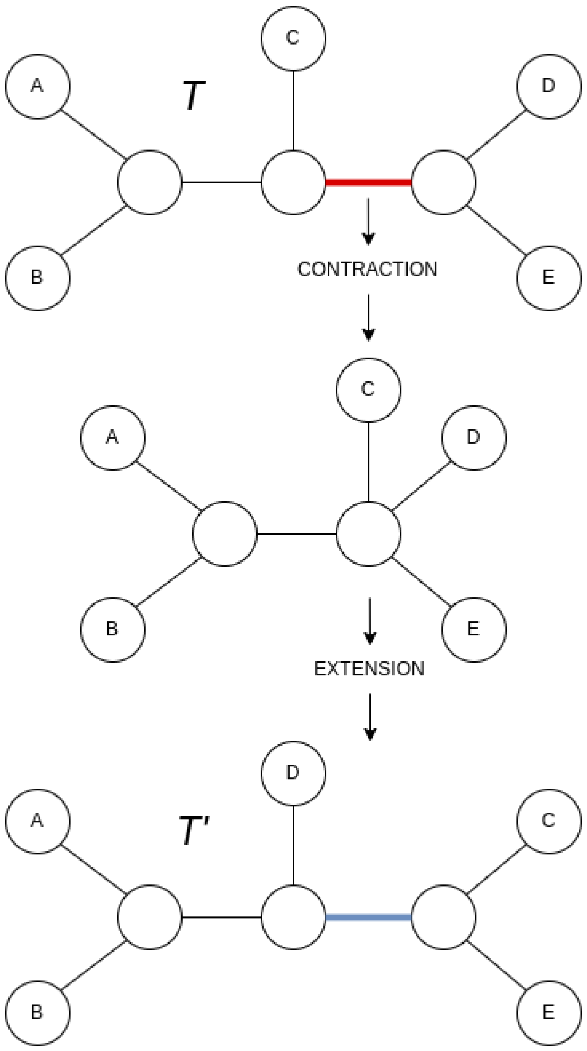


Figure 5.2: Transformation of a tree T into a tree T' via edit operations on the edges. Here the distance between T and T' is 2 as to obtain T' from T were required one contraction and one extension.

In Table 5.4 we indicate the RF distance between the best solution found by FastME and PhyloES as well as the number of different solutions found by PhyloES. The reported values clearly remark how the RF distance tends to increase with the number of taxa. In particular, for the 200 and 300 taxa cases, its high-value range (64-74) indicates a significant difference in terms of tree topology. This analysis confirms the added value of the evolutionary approach we developed suggesting that the observed relatively small changes in tree length (in this case around the order of 10^{-4}) imply notable differences in terms of the resulting tree structure. Hence the impact of the strategy employed in PhyloES leads to a significant difference in terms of the phylogenetic inference problem we aim to solve.

5.6 RQ5, Numerical precision analysis

In principle, the BMEP might suffer from numerical precision issues, a problem which should be carefully taken into account. In fact, with a number of taxa approximately greater than 50, some of the terms of equation (1.6) might take values below the machine epsilon precision. This obstacle, on the one hand clearly represents a theoretical limitation of the BMEP, as it does not allow to guarantee a correct comparison between trees in terms of tree length, but on the other, it turns out to have quite a small impact in real applications. In fact, if we consider a phylogeny T of n taxa, $n \geq 50$, and with a given topology τ, we can split its length (equation 1.6) in two terms:

$$\begin{aligned} l(T) = \sum_{i,j \in X} 2^{1-\tau_{ij}} d_{ij} &= \sum_{i,j \in A} 2^{1-\tau_{ij}} d_{ij} + \sum_{i,j \in B} 2^{1-\tau_{ij}} d_{ij} \\ &= l_A(T) + l_B(T) \end{aligned} \tag{5.1}$$

where $A := \{\tau_{ij} \in \tau | \tau_{ij} < 50\}$ and $B := \{\tau_{ij} \in \tau | \tau_{ij} \geq 50\}$. For the right-hand side $l_B(T)$ of equation 5.1, which represents the summation of the terms affected by numerical precision issues, we can easily derive the following inequality:

$$\begin{aligned} l_B(T) = \sum_{i,j \in B} 2^{1-\tau_{ij}} d_{ij} &\leq \sum_{i,j \in B} 2^{1-\tau_{ij}} \max_{i,j \in X} d_{ij} \\ &\leq |B|^2 2^{1-50} \max_{i,j \in X} d_{ij} < n^2 2^{-49} \max_{i,j \in X} d_{ij} = \omega, \end{aligned} \tag{5.2}$$

in which if the elements d_{ij} are not all identical, as in most of the real applications, the first inequality can be clearly turned into a strong inequality. The last strong inequality instead, holds due to the fact that $|B| < n$, as trivially an unrooted binary tree must have at least two cherries, where cherry indicates a pair of leaves with a mutual topological distance equal to 2. The value ω of equation 5.2 represents a bound for the contribution of the terms of B in $l(T)$ and it can be used as an error tolerance for the trees' length comparison. In other words, once defined $\tilde{l}$ as the numerical computation of l and given two phylogenies T_1 and T_2, equation 5.2 allows to state that:

$$\tilde{l}(T_1) + \omega < \tilde{l}(T_2) \implies l(T_1) < l(T_2). \tag{5.3}$$

In Table 5.5, we show the ω values computed for the different datasets used in the experimental phase. As long as the improvements provided by PhyloES with respect to the FastME solution (Table 5.2) are within 10^{-5} and 10^{-6} we can certify that inequality 5.3 is satisfied and therefore safely confirm the robustness of our results.

We further remark that if on one hand inequality 5.2 tends to grow quadratically with the number of taxa, on the other it actually represents a quite pessimistic overestimation for the value of $l_B(T)$. In fact, the bound basically assumes that when the mutual distance of a set of taxa is greater than 49 its value is exactly 50. Furthermore, in our experiments, all the solution trees provided by each of the tested algorithms were characterised by a maximum topological distance between taxa lower than 50.

		Obj			Time			Gap %	
run	n	FME	IP	PES	FME	IP	PES	FME	PES
1	15	1.458490	1.458490	1.458490	0.012	0.248	0.066	0.000000	0.000000
2	15	1.734989	1.734989	1.734989	0.011	11.615	0.051	0.000000	0.000000
3	15	1.818631	1.818631	1.818631	0.014	6.543	0.144	0.000000	0.000000
4	15	1.799283	1.799283	1.799283	0.012	55.476	0.049	0.000000	0.000000
5	15	1.542442	1.542442	1.542442	0.012	2.103	0.054	0.000000	0.000000
6	15	1.816193	1.816193	1.816193	0.012	103.122	0.059	0.000000	0.000000
7	15	1.659289	1.659289	1.659289	0.012	0.573	0.083	0.000000	0.000000
8	15	1.771357	1.771357	1.771357	0.013	0.715	0.062	0.000000	0.000000
9	15	1.785821	1.785821	1.785821	0.012	3.135	0.060	0.000000	0.000000
10	15	1.728345	1.728345	1.728345	0.012	5.075	0.058	0.000000	0.000000
11	20	2.247273	2.247273	2.247273	0.015	964.173	0.083	0.000000	0.000000
12	20	2.254319	2.254319	2.254319	0.014	58.561	0.069	0.000000	0.000000
13	20	2.304339	2.304339	2.304339	0.014	154.638	0.111	0.000000	0.000000
14	20	2.367511	2.367511	2.367511	0.015	63.619	0.069	0.000000	0.000000
15	20	2.322821	2.322821	2.322821	0.014	366.301	0.185	0.000000	0.000000
16	20	2.175853	2.175853	2.175853	0.014	966.346	0.145	0.000000	0.000000
17	20	2.252254	2.252254	2.252254	0.014	337.923	0.065	0.000000	0.000000
18	20	2.298822	2.298822	2.298822	0.014	4043.536	0.073	0.000000	0.000000
19	20	2.406584	2.406584	2.406584	0.014	231.219	0.074	0.000000	0.000000
20	20	1.855247	1.855247	1.855247	0.014	10.893	0.072	0.000000	0.000000
21	25	2.607170	2.607170	2.607170	0.017	Time Limit	0.100	0.000000	0.000000
22	25	2.859623	2.859411	2.859411	0.018	Time Limit	0.186	0.000074	0.000000
23	25	2.576427	2.576427	2.576427	0.018	Time Limit4	0.090	0.000000	0.000000
24	25	2.705412	2.705412	2.705412	0.017	737.774	0.090	0.000000	0.000000
25	25	2.608054	2.608054	2.608054	0.018	Time Limit	0.138	0.000000	0.000000
26	25	2.705522	2.704784	2.704784	0.017	Time Limit	0.138	0.000273	0.000000
27	25	2.851618	2.851618	2.851618	0.018	Time Limit	0.093	0.000000	0.000000
28	25	2.631415	2.631415	2.631415	0.018	18575.847	0.137	0.000000	0.000000
29	25	2.461560	2.461560	2.461560	0.018	Time Limit	0.187	0.000000	0.000000
30	25	2.765643	2.765643	2.765643	0.017	Time Limit	0.132	0.000000	0.000000

Table 5.1: Comparison between PhyloES, FastME, and the ILP model (time in seconds)

Table 5.2: BMEP length function value and number of solutions analysis for FastME, PhyloES and the RI

	FastME		PhyloES				RI			
Dataset	Best SIA	obj	obj avg	obj std	n sol	improvement %	obj avg	obj std	n sol	improvement %
100_rdpii F81	OLSME	8.415707	8.415707	0.000000	1	0.0000	8.415707	0.000000	4	-0.0000
100_rdpii F84	OLSME	8.461857	8.461710	0.000000	1	-0.0017	8.461710	0.000000	4	-0.0017
100_rdpii JC69	OLSME	8.406178	8.406178	0.000000	1	0.0000	8.406178	0.000000	3	-0.0000
100_rdpii K2P	OLSME	8.453968	8.453811	0.000000	1	-0.0018	8.453811	0.000000	2	-0.0018
200_rdpii F81	BioNJ	13.987183	13.985146	0.000095	2	-0.0146	13.985354	0.000190	10	-0.0131
200_rdpii F84	BioNJ	14.074191	14.068298	0.000048	2	-0.0419	14.068800	0.000301	10	-0.0383
200_rdpii JC69	BioNJ	13.973130	13.971048	0.000059	2	-0.0149	13.971330	0.000130	10	-0.0129
200_rdpii K2P	BioNJ	14.062630	14.056749	0.000031	2	-0.0418	14.057467	0.000368	10	-0.0367
300_zilla F81	NJ	4.482169	4.480371	0.000008	3	-0.0401	4.480839	0.000187	10	-0.0297
300_zilla F84	BioNJ	4.506561	4.504317	0.000001	2	-0.0498	4.504646	0.000213	10	-0.0425
300_zilla JC69	NJ	4.479926	4.478132	0.000000	1	-0.0400	4.478463	0.000200	10	-0.0327
300_zilla K2P	BioNJ	4.501083	4.498846	0.000000	1	-0.0497	4.499206	0.000196	10	-0.0417

Table 5.3: Comparison of the computational time in seconds and the number of NNI and SPR calls in PhyloES and the RI.

		PhyloES					RI				FastME
dataset	n trees	avg time	time std	iterations	nni	spr	avg time	time std	nni	spr	time
100_rdpii F81	256,0	1,7	0,0	4,0	32.852,5	8.364,1	2,8	0,1	111.432,2	26.462,1	0,2
100_rdpii F84	275,2	2,0	0,2	4,3	34.978,8	9.073,4	3,6	0,4	119.981,3	28.477,1	0,2
100_rdpii JC69	256,0	1,7	0,1	4,0	32.613,4	8.286,4	3,0	0,3	111.477,9	26.468,5	0,2
100_rdpii K2P	281,6	2,0	0,2	4,4	34.992,1	9.044,9	3,7	0,4	122.765,2	29.141,5	0,2
200_rdpii F81	660,8	48,6	18,5	27,3	241.433,8	60.035,5	83,5	28,9	589.174,6	122.665,9	1,4
200_rdpii F84	632,0	63,8	7,5	31,7	320.680,4	81.820,5	85,2	18,3	592.632,8	123.768,6	1,4
200_rdpii JC69	683,2	51,9	13,4	28,9	254.344,9	63.662,8	95,3	27,9	664.862,5	138.599,6	1,3
200_rdpii K2P	588,8	64,0	6,2	31,8	321.755,8	82.422,4	79,9	14,3	553.963,2	115.777,8	1,4
300_zilla F81	636,8	224,4	24,3	27,0	550.774,2	125.557,9	377,9	87,4	1.059.653,6	200.291,1	4,4
300_zilla F84	748,8	280,3	80,5	39,0	690.211,0	157.725,7	484,7	279,9	1.370.971,3	259.037,1	4,4
300_zilla JC69	718,4	210,7	30,0	22,4	524.163,3	119.632,6	383,1	68,8	1.077.795,2	203.781,7	4,5
300_zilla K2P	620,8	271,2	72,4	33,8	671.155,2	154.428,8	353,9	197,8	1.000.720,6	189.174,5	4,3

Table 5.4: Topological RF distances between FastME and PhyloES solutions

Dataset	avg RF	std RF	n solutions
100_rdpii F81	0.00	0.00	1
100_rdpii F84	6.00	0.00	1
100_rdpii JC69	0.00	0.00	1
100_rdpii K2P	6.00	0.00	1
200_rdpii F81	44.40	2.06	2
200_rdpii F84	67.80	2.89	2
200_rdpii JC69	45.60	1.26	2
200_rdpii K2P	66.60	1.89	2
300_zilla F81	82.80	12.83	3
300_zilla F84	62.80	5.43	2
300_zilla JC69	74.80	1.03	1
300_zilla K2P	60.80	1.03	1

Table 5.5: Values of the error tolerance for the considered datasets

taxa	dataset	F81	F84	JC69	K2P
100	rdpii	9.64E-12	9.77E-12	9.60E-12	9.74E-12
200	rdpii	4.03E-11	4.07E-11	4.01E-11	4.06E-11
300	zilla	2.52E-11	2.56E-11	2.52E-11	2.55E-11

Chapter 6

Conclusions and future works

In this dissertation, we analysed the Balanced Minimum Evolution Problem, a pivotal and challenging conundrum that lies at the intersection of computational biology, phylogenetics and mathematical optimisation. In comparison with other phylogenetic methodologies, the BMEP offers distinctive advantages. Unlike some more computationally intensive methods, the BMEP provides an excellent trade-off between accuracy and computational efficiency, making it a suitable choice to tackle a wide array of phylogenetic inference problems. Since the BMEP is defined by an optimisation problem, in which the optimal solution defines the true phylogenetic tree describing the evolutionary relationship between a set of taxa, it is essential to have at disposal algorithmic tools to solve it. However, its strong combinatorial nature makes it yet impossible to solve via exact algorithms, except in the case of very small instances. Hence the need to develop heuristics capable of tackling problems with a large number of taxa. Considering that within the BMEP framework the closer a tree length is to the optimal value the closer its structure to the true tree, it becomes clear from the phylogenetic inference perspective the necessity of providing the best solution possible.

With this in mind, in this extensive discussion, we introduced a novel Evolutionary Strategy (ES) approach designed specifically for the Balanced Minimum Evolution Problem (BMEP). Notably, our ES approach has demonstrated remarkable performance, consistently surpassing the capabilities of FastME, which represents the current state-of-the-art algorithm for the BMEP. The key takeaway from our computational experiments is that, while our method does entail some additional computational time, it leverages the power of local search operators such as Balanced Nearest Neighbor Interchange (BNNI) and Balanced Subtree Pruning and Regrafting (BSPR), executed multiple times within a well-crafted exploration framework. The results consistently yield solutions that outperform those produced by FastME, proving the effectiveness of our approach in enhancing phylogenetic analysis via the BMEP.

One of the notable findings of our study pertains to the profound impact of small reductions in the tree length, which can lead to significant differences in the resulting topologies. From the perspective of phylogenetic inference, these variations hold considerable importance. By analysing the behaviour of our algorithm across multiple runs, we have observed a high degree of stability, even in the face of its stochastic nature. Remarkably, for all the BMEP instances under consideration, a singular solution or a limited set of unique solutions consistently emerged across multiple runs on the same problem instance. This stability is a significant asset, offering a degree of reliability and consistency in the solutions provided by PhyloES.

Our approach is proven to be robust also from the numerical point of view. We have in fact addressed the numerical precision challenges inherent in the BMEP, and our analysis has yielded a valuable inequality. This inequality serves as a numerical tolerance criterion for the exact comparison of the lengths of two phylogenetic trees. The application of this criterion empowers us to confidently assert the robustness of the solutions generated by PhyloES, enhancing the trustworthiness of the results in the face of numerical challenges.

While our current implementation of PhyloES has proven highly effective, it is important to acknowledge its limitations in terms of scalability. The computational demands imposed by the BNNI and BSPR algorithms constrain the application of PhyloES to larger problem instances, necessitating high-performance computing resources. However, we identified a promising line of development for future research. The increase in biological data availability and the substantial advancements in Machine Learning techniques, as demonstrated by Azouri et al. [2021], suggest that it may be possible to replace the BNNI and BSPR algorithms with learned approximations. These learned approximations have the potential to not only match the performance of the original algorithms but also dramatically reduce the computational effort required. This opens the exciting possibility of employing PhyloES on larger and more complex instances, thereby expanding its applicability in the field of phylogenetic analysis. Furthermore, the possibility of extending PhyloES to greater-size instances would allow its exploitation also for phylogenetic inference under the Maximum Likelihood framework. The ML approach requires a computational effort far more costly than the one required by the BMEP, and this makes its deployment for large problems extremely challenging and computationally intense. As introduced in chapter 1, the correlation between enhancements in BMEP tree length and improvements in likelihood has been demonstrated by Hordijk and Gascuel [2006]. Consequently, it would be quite natural to consider integrating these two approaches. More specifically: the core of the complexity with the ML is the tree likelihood evaluation; since the evaluation of the BMEP lenght function is much cheaper, one might think of using PhyloES as a proxy for the likelihood computation, to have a faster tool to explore the search space, limiting the number of likelihood evaluations while using the BMEP tree length function as a local search criterion. Another important hurdle of the BMEP is represented by the numerical stability. Although the inequality for the numerical precision reliability proposed in section 5.6 allowed us to prove the robustness of our results, it still represents a quite loose bound, providing a large overestimation of the numerical error of the tree length computation. A first focus of interest for future research might then be the investigation on the possibility of tightening the inequality, increasing the tolerance range around the numerical tree length value needed to confidently compare two trees and by so ultimately improving the accuracy of the algorithm. As opposed to the ILP approach, a further limitation of PhyloES (and FastME) is the inability to provide optimality gaps, precluding the possibility of proving optimality nor measuring the quality of the solution obtained. If on the one hand the tree length optimal gap remains an important missing indicator from the optimisation point of view, on the other, from the phylogenetic perspective, it would be of great importance to have at disposal information regarding the topological differences between the solution obtained by PhyloES and the optimal one, for instance, their RF distance or the knowledge of common substructures. Unfortunately, such research questions have not been successfully addressed yet and they will therefore certainly represent the focus of future developments.

In conclusion, despite the stochastic nature of our proposed algorithm, our extensive research and experiments have undoubtedly proven its quality and stability, consistently delivering a single or a limited number of solutions across multiple runs on the same problem instance. With its enhanced performance over existing methods and the potential for future scalability improvements through the integration of Machine Learning techniques, PhyloES emerges as an excellent tool in the realm of phylogenetic analysis, offering robust and reliable solutions to complex evolutionary questions in the field of biology and beyond.

Bibliography

V.A. Albert. *Parsimony, phylogeny, and genomics.* OUP Oxford, 2005.

S. Almécija, A.S. Hammond, N.E. Thompson, K.D. Pugh, S. Moyà-Solà, and D.M Alba. Fossil apes and human evolution. *Science*, 372(6542):eabb4363, 2021.

S. Amiroch, M.S. Pradana, M.I. Irawan, and M.I. Mukhlash. Multiple alignment analysis on phylogenetic tree of the spread of sars epidemic using distance method. *Journal of Physics: Conference Series*, 890(1):012080, sep 2017.

R. Aringhieri, Catanzaro. D., and M. Di Summa. Optimal solutions for the balanced minimum evolution problem. *Computers & Operations Research*, 38(12):1845–1854, 2011a. ISSN 0305-0548.

R. Aringhieri, Catanzaro D., and M. Di Summa. Optimal solutions for the balanced minimum evolution problem. *Computers and Operations Research*, 38:1845–1854, 2011b.

D. Azouri, S. Abadi, Y. Mansour, I. Mayrose, and T. Pupko. Harnessing machine learning to guide phylogenetic-tree search algorithms. *Nature Communications*, 12, 2021.

T. Bäck. *Evolutionary algorithms in theory and practice: Evolution strategies, evolutionary programming, genetic algorithms.* Oxford University Press, Oxford, UK, 1996.

A. Bartoli, A. De Lorenzo, E. Medvet, and G. Squillero. Multi-level diversity promotion strategies for grammar-guided genetic programming. *Applied Soft Computing*, 83:105599, 2019. ISSN 1568-4946.

M. Bordewich, O. Gascuel, K. Huber, and V. Moulton. Consistency of topological moves based on the balanced minimum evolution principle of phylogenetic inference. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 6:110–7, 2009.

M.J. Brauer, M.T. Holder, L.A. Dries, D.J. Zwickl, P.O. Lewis, and D.M. Hillis. Genetic algorithms and parallel processing in maximum-likelihood phylogeny inference. *Molecular Biology and Evolution*, 19(10):1717–1726, 2002.

M.G. Bulmer. Use of the method of generalized least squares in reconstructing phylogenies from sequence data. *Molecular Biology and Evolution*, 8:868–883, 1991.

R.M. Bush, C.A. Bender, K. Subbarao, N.J. Cox, and W.M. Fitch. Predicting the evolution of human influenza a. *Science*, 286(5446):1921–1925, 1999.

A. Cain. A classification of living organisms. `https://www.britannica.com/science/taxonomy`, 2023. Accessed: 2023-10-14.

S. Caminiti, I. Finocchi, and R. Petreschi. On coding labeled trees. *Theoretical Computer Science*, 382(2):97–108, 2007. ISSN 0304-3975. Latin American Theoretical Informatics.

D. Catanzaro and R. Pesenti. Enumerating vertices of the balanced minimum evolution polytope. *Computers & Operations Research*, 109:209–217, 2019. ISSN 0305-0548.

D. Catanzaro, M. Labbé, R. Pesenti, and J.J. Salazar-González. The balanced minimum evolution problem. *INFORMS Journal on Computing*, 24(2):276–294, 2012.

D. Catanzaro, R. Pesenti, and L. A. Wolsey. On the balanced minimum evolution polytope. *Discrete Optimization*, 36:100570, 2020.

D. Catanzaro, M. Frohn, and R. Gascuel, O.and Pesenti. A tutorial on the balanced minimum evolution problem. *European Journal of Operational Research*, 300(1):1–19, 2022.

L.L. Cavalli-Sforza and A.W.F. Edwards. Phylogenetic analysis. models and estimation procedures. *American journal of human genetics*, 19(3 Pt 1):233, 1967.

B.S.W. Chang and M.J. Donoghue. Recreating ancestral proteins. *Trends in Ecology & Evolution*, 15(3):109–114, 2000. ISSN 0169-5347.

M. A. Cueto and F. A. Matsen. Polyhedral geometry of phylogenetic rogue taxa. *Bulletin of Mathematical Biology*, 73(6):1202–1226, 2011.

C. Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.* J. Murray, 1859.

F. Denis and O. Gascuel. On the consistency of the minimum evolution principle of phylogenetic inference. *Discrete Applied Mathematics*, 127:63–77, 04 2003.

R. Desper and O. Gascuel. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *Journal of computational biology : a journal of computational molecular cell biology*, 9 5:687–705, 2002.

R. Desper and O. Gascuel. Theoretical foundations of the balanced minimum evolution method of phylogenetic inference and its relationship to the weighted least-squares tree fitting. *Molecular Biology and Evolution*, 21(3):587–598, 2004.

E. ECastro-Nallar, M. Pérez-Losada, G.F. Burton, and K.A Crandall. The evolution of hiv: Inferences using phylogenetics. *Molecular Phylogenetics and Evolution*, 62(2):777–792, 2012. ISSN 1055-7903.

J.S. Farris. Methods for computing wagner trees. *Systematic Biology*, 19(1):83 – 92, 1970. Cited by: 875.

J. Felsenstein. *Inferring Phylogenies.* Sinauer Associates, Sunderland, MA, 2004.

J. Felsenstein and G.A. Churchill. A Hidden Markov Model approach to variation among sites in rate of evolution. *Molecular Biology and Evolution*, 13(1):93–104, 01 1996.

S. Fiorini and G. Joret. Approximating the balanced minimum evolution problem. *Operations Research Letters*, 40(1):31–35, 2012.

W.M. Fitch. Toward defining the course of evolution: minimum change for a specific tree topology. *Systematic Biology*, 20(4):406–416, 1971.

S. Forcey, L. Keefe, and W. Sands. Facets of the balanced minimal evolution polytope. *Mathematical Biology*, 73:447–468, 2016.

S. Forcey, L. Keefe, and W. Sands. Split-facets for balanced minimal evolution polytopes and the permutoassociahedron. *Bulletin of Mathematical Biology, in press*, 2017.

M. Frohn. On the approximability of the fixed-tree balanced minimum evolution problem. *Optimization Letters*, 15:2321–2329, 2021.

M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Mathematical Sciences Series. Freeman, 1979. ISBN 9780716710448.

O. Gascuel. BIONJ: An improved version of the NJ algorithm based on a simple model of sequence data. *Molecular biology and evolution*, 14:685–95, 08 1997.

O. Gascuel. On the optimization principle in phylogenetic analysis and the minimum-evolution criterion. *Molecular biology and evolution*, 17(3):401–405, 2000.

O. Gascuel. Concerning the NJ algorithm and its unweighted version, UNJ. *Mathematical Hierarchies and Biology*, 05 2002.

O. Gascuel. *Mathematics of evolution and phylogeny.* Oxford University Press, New York, 2005.

S. Guindon, J.F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, and O. Gascuel. New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0. *Systematic Biology*, 59(3):307–321, 05 2010. ISSN 1063-5157.

J. W Hallam, O. Akman, and F. Akman. Genetic algorithms with shrinking population size. *Computational Statistics*, 25:691–705, 2010.

M. Hasegawa, H. Kishino, and T. Yano. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.

D. C. Haws, T. L. Hodge, and R. Yoshida. Optimality of the neighbor joining algorithm and faces of the balanced minimum evolution polytope. *Bulletin of Mathematical Biology*, 73(11):2627–2648, 2011.

R. Helaers and M. Milinkovitch. Metapiga v2.0: Maximum likelihood large phylogeny estimation using the metapopulation genetic algorithm and other stochastic heuristics. *BMC bioinformatics*, 11:379, 07 2010.

B.R. Holland. Molecular Evolution: A Statistical Approach. — By Ziheng Yang. *Systematic Biology*, 64(3):545–546, 01 2015. ISSN 1063-5157.

J. H. Holland. Adaptation in natural and artificial systems. *Ann Arbor: University of Michigan Press*, 1975.

W. Hordijk and O. Gascuel. Improving the efficiency of SPR moves in phylogenetic tree search methods based on maximum likelihood. *Bioinformatics (Oxford, England)*, 21:4338–47, 01 2006.

J.P. Huelsenbeck, F. Ronquist, R. Nielsen, and J.P. Bollback. Bayesian inference of phylogeny and its impact on evolutionary biology. *Science*, 294(5550):2310 – 2314, 2001.

J.P. Huelsenbeck, B. Larget, R.E. Miller, and F. Ronquist. Potential applications and pitfalls of bayesian inference of phylogeny. *Systematic Biology*, 51(5):673 – 688, 2002. Cited by: 679.

N.J. Isaac, S.T. Turvey, B. Collen, C. Waterman, and J.E. Baillie. Mammals on the edge: conservation priorities based on threat and phylogeny. *PLoS One*, 2007.

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

G. Jäger. Global-scale phylogenetic linguistic inference from lexical resources. *Scientific Data*, 5, 10 2018.

S. Katoch, S.S.h Chauhan, and V. Kumar. A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, 80:8091–8126, 2021.

K.K. Kidd and L.A. Sgaramella-Zonta. Phylogenetic analysis: concepts and methods. *American journal of human genetics*, 23(3):235, 1971.

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

W. S Klug, M. R Cummings, et al. *Concepts of genetics*, volume Ed. 7. Pearson Education, Inc, 2003.

S.L. Kosakovsky Pond, D. Posada, M.B. Gravenor, C.H. Woelk, and D.D.W. Frost. GARD: A genetic algorithm for recombination detection. *Bioinformatics*, 22(24):3096–3098, 2006.

M.K. Kuhner and J. Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular biology and evolution*, 11(3):459–468, 1994.

V. Lefort, R. Desper, and O. Gascuel. FastME 2.0: A comprehensive, accurate, and fast distance-based phylogeny inference program. *Molecular Biology and Evolution*, 32(10):2798–2800, 2015a.

V. Lefort, R. Desper, and O. Gascuel. Fastme 2.0 user guide. `http://www.atgc-montpellier.fr/fastme/usersguide.php`, 2015b. Accessed: 2023-10-09.

P.e Lemey, M. Salemi, and A.M. Vandamme. *The phylogenetic handbook: A practical approach to phylogenetic analysis and hypothesis testing.* Cambridge University Press, 2009.

P.O. Lewis. A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. *Molecular Biology and Evolution*, 15(3):277–283, 03 1998. ISSN 0737-4038.

C. Linnaei. *Systema naturae*, volume v.1. Holmiae, Impensis Direct. Laurentii Salvii, 1758-1759, 1758.

S. Luke. *Essentials of metaheuristics*, volume 2. Lulu Raleigh, 2013.

H. Matsuda. Protein phylogenetic inference using maximum likelihood with a genetic algorithm. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 512–523, 1996.

M. Myers, G. Satas, and B. Raphael. Calder: Inferring phylogenetic trees from longitudinal tumor samples. *Cell Systems*, 8, 06 2019.

M. Nei and L. Jin. Variances of the average number of nucleotide substitutions within and between populations. *Molecular biology and evolution*, 6:290–300, 06 1989.

M. Nei and S. Kumar. *Molecular Evolution and Phylogenetics.* Oxford University Press, 2000.

M. Nei, J.C. Stephens, and N. Saitou. Methods for computing the standard errors of branching points in an evolutionary tree and their application to molecular data from humans and apes. *Molecular biology and evolution*, 2 1:66–85, 1985.

E. H. Neville. The codifying of tree-structure. *Mathematical Proceedings of the Cambridge Philosophical Society*, 49(3):381–385, 1953.

E. Noutahi and N. El-Mabrouk. GATC: A genetic algorithm for gene tree construction under the duplication-transfer-loss model of evolution. *BMC genomics*, 19:97–107, 2018.

Y. Ohtomo, T. Kakegawa, A. Ishida, T. Nagase, and M. Rosing. Evidence for biogenic graphite in early archaean isua metasedimentary rocks. *Nature Geoscience*, 7, 12 2013.

F. Pardi. *Algorithms on Phylogenetic Trees.* PhD thesis, University of Cambridge, UK, 2009.

D.S. Parker. *The Construction of Huffman is a Submodular (" convex") Optimization Problem Over a Lattice of Binary Trees.* University of California (Los Angeles). Computer Science Department, 1996.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS-W*, 2017.

Y. Pauplin. Direct calculation of a tree length using a distance matrix. *Journal of Molecular Evolution*, 51:41–47, 2000.

B.K.D. Pearce, A.S. Tupper, R.E. Pudritz, and P.G. Higgs. Constraining the time interval for the origin of life on earth. *Astrobiology*, 18(3):343–364, 2018.

G. Pennington, C.A. Smith, S. Shackney, and R. Schwartz. Reconstructing tumor phylogenies from heterogeneous single-cell data. *Journal of Bioinformatics and Computational Biology*, 5(02a): 407–427, 2007.

S. Piquerez, S. Harvey, J. Beynon, and V. Ntoukakis. Improving crop disease resistance: Lessons from research on arabidopsis and tomato. *Frontiers in plant science*, 5:671, 12 2014.

L. Poladian and L.S. Jermiin. Multi-objective evolutionary algorithms and phylogenetic inference with multiple data sets. *Soft Computing*, 10:359–368, 2006.

K. Popper. *The logic of scientific discovery.* Routledge, 2005.

H. Prüfer. Neuer beweis eines satzes über permutationen. *Archiv für Mathematik und Physik*, pages 742–744, 1918.

I. Rechenberg. Evolutionsstrategie. *Optimierung technischer Systeme nach Prinzipien derbiologischen Evolution*, 1973.

D.F. Robinson and L.R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53 (1):131–147, 1981. ISSN 0025-5564.

J. Rohlf. Numbering binary trees with labeled terminal vertices. *Bulletin of Mathematical Biology*, 45:40, 01 1983.

M.T. Rosing. 13c-depleted carbon microparticles in $>$ 3700-ma sea-floor sedimentary rocks from west greenland. *Science*, 283(5402):674–676, 1999.

A. Rzhetsky and M. Nei. A Simple Method for Estimating and Testing Minimum-Evolution Trees. *Molecular Biology and Evolution*, 9(5):945–945, 09 1992. ISSN 0737-4038.

N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 07 1987.

R. Schwartz. Computational models for cancer phylogenetics. In Tandy Warnow, editor, *Bioinformatics and Phylogenetics*. Springer, NY, 2019.

C. Semple and M. Steel. *Phylogenetics*, volume 24. Oxford University Press on Demand, 2003.

C. Semple and M. Steel. Cyclic permutations and evolutionary trees. *Advances in Applied Mathematics*, 32(4):669–680, 2004.

A. Skourikhine. Phylogenetic tree reconstruction using self-adaptive genetic algorithm. In *Proceedings IEEE International Symposium on Bio-Informatics and Biomedical Engineering*, pages 129–134, 2000.

R.R. Sokal. A statiscal method for evaluating systematic relationships. *Univ Kans sci bull*, 38: 1409–1438, 1958.

A. Stamatakis. An efficient program for phylogenetic inference using simulated annealing. In *Parallel and Distributed Processing Symposium, International*, volume 9, page 198b, Los Alamitos, CA, USA, apr 2005. IEEE Computer Society.

E. Susko. Confidence regions and hypothesis tests for topologies using generalized least squares. *Molecular biology and evolution*, 20(6):862–868, 2003.

K. Tamura and M. Nei. Estimation of the number of nucleotide substituions in the control region of mitochondrial dna in humans and chimpanzees. *Molecular biology and evolution*, 10:512–26, 06 1993.

S. Tavaré. Some probabilistic and statistical problems on the analysis of DNA sequences. *Lectures on Mathematics in the Life Sciences*, 17:57–86, 1986.

C.O. Webb, D.D. Ackerly, M.A. McPeek, and M.J. Donoghue. Phylogenies and community ecology. *Annual Review of Ecology and Systematics*, 33(1):475–505, 2002.

D.J. Zwickl. Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion. Technical report, School of Biological Sciences, University of Texas, Austin, 2006.