The worst case complexity of Maximum Parsimony

Amir Carmel, Noa Musa-Lempel, Dekel Tsur, and Michal Ziv-Ukelson

Department of Computer Science, Ben-Gurion University of the Negev.
Email: {karmela,noamu,dekelts,michaluz}@cs.bgu.ac.il

Abstract. One of the core classical problems in computational biology is that of constructing the most parsimonious phylogenetic tree interpreting an input set of sequences from the genomes of evolutionarily related organisms. We re-examine the classical Maximum Parsimony (MP) optimization problem for the general (asymmetric) scoring matrix case, where rooted phylogenies are implied, and analyze the worst case bounds of three approaches to MP: The approach of Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967), the approach of Hendy and Penny (Hendy and Penny, 1982), and a new agglomerative, "bottom-up" approach we present in this paper. We show that the second and third approaches are faster than the first one by a factor of $\Theta(\sqrt{n})$ and $\Theta(n)$, respectively, where n is the number of species.

1 Introduction

Phylogenetics is the study of evolutionary relationships among groups of organisms (e.g. species, populations), which are discovered through molecular sequencing data and morphological data matrices. A *Phylogeny* (also called a dendogram) is a graph-like structure whose topology describes the inferred evolutionary history among a set of biological entities, such as species or DNA sequences.

Phylogenies are classically modeled as either rooted or unrooted labeled binary trees, where the input entities are assigned to the leaf vertices. An *unrooted phylogeny* is an acyclic connected labeled graph in which every vertex has degree of either three or one. Each vertex of degree one has a distinct label. A *rooted phylogeny*, on the other hand, is similar to an unrooted phylogeny, except that it has one internal vertex of degree two, which is designated as the root. In a rooted phylogeny the edges are directed from the root towards the leaves.

The decision of whether to model phylogenies as rooted versus unrooted trees depends either on the availability of a molecular clock, or on the nucleotide or amino acid substitution scoring matrix representing the evolutionary mutation events. Modeling phylogenies as unrooted trees requires the assumption of symmetric scoring matrices. However, when the symmetry restriction on scoring matrices is removed, the tree rooting becomes meaningful. A simple literature review of current biological research shows that the symmetric scoring matrices, though computationally convenient, do not yield a biologically reliable model (Rodriguez *et al.*, 1990; Takahata and Kimura, 1981; Gojobori *et al.*, 1982; Tajima and Nei, 1984). Various recent biological publications apply asymmetric scoring matrices to the alignment of genomic sequences, and many papers can nowadays be found on the construction of asymmetric scoring matrices consisting of nucleotides (Tamura, 1992; Tamura and Nei, 1993; Blouin *et al.*, 1998) and amino acids (Müller *et al.*, 2001; Bastien *et al.*, 2005). Thus, in this work we do not assume symmetric scoring matrices and therefore construct rooted phylogenies.

Methods for phylogeny reconstruction can be classified into *distance-based* versus *character-based* methods. Given a set of input sequences, a distance-based method, such as UPGMA and neighbor joining (NJ), first computes pairwise distances according to some measure (Felsenstein and Felenstein, 2004). Then, the actual data is discarded and the fixed distances are used in the derivation of trees. In contrast, in character-based methods (such as Maximum Parsimony and Maximum Likelihood) the inference depends upon models describing the evolutionary changes of characters (e.g. nucleotides or amino acids) that led from an original sequence in some common ancestor to the evolution of the observed input sequences. The great advantage of character-based algorithms for phylogenetic reconstruction is that, given a good multiple alignment of the input sequences, they can exploit the potential phylogenetic inferences with great sensitivity. Their weakness, however, is in their computational intensity. In this paper we focus on the classical *Maximum Parsimony* character-based phylogenetic approach.

1.1 Phylogentic Reconstruction based on Parsimony Maximization

Parsimony Maximization (i.e. preferring the simpler of two otherwise equally adequate theorizations) is one of the classical approaches to computationally reconstruct a phylogeny for a given set of biologically related sequences. When applied to computational phylogenetics, the parsimony maximization approach seeks the phylogenetic tree that supposes the least amount of evolutionary change explaining the observed data (Cavalli-Sforza and Edwards, 1967). There are two classical problems inferred from phylogenetic parsimony maximization: Small Parsimony (SP) and Maximum Parsimony (MP), explained below.

Problem 1: Small Parsimony (SP) The Small Parsimony problem is to compute, for a proposed phylogeny, a reconstruction of events leading to the input data with as few changes as possible over the whole tree (see Figure 1). The input to this problem is a multiple alignment of n input sequences of length m each, and a topology in the form of a rooted phylogenetic tree over n leaves, where each leaf is associated with a distinct sequence from the input set. Based on this input, the objective is to compute a labeling of the internal vertices of the input phylogeny which optimizes some predefined scoring scheme.

The most basic algorithms for solving SP are Fitch's algorithm (Fitch, 1971) and Sankoff's algorithm (Sankoff, 1975). In its simplest variant (solved by Fitch's algorithm and exemplified in Figure 1.c-d), the scoring scheme is Hamming Distance, and the optimal labeling is one that minimizes the number of mutations. It is standard to assume position-independence between the states (columns of characters in the multiple alignment). Thus, the total SP score for an input multiple sequence alignment is computed as the sum of the SP scores of each state assignment.

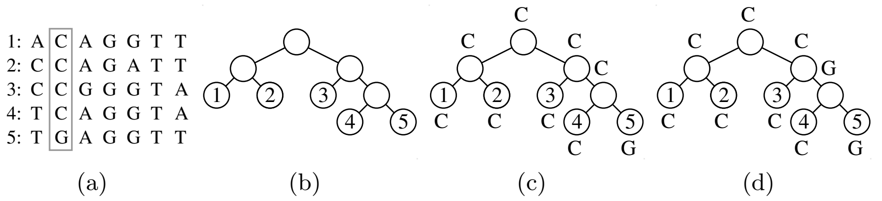

Fig. 1. An example of an instance with Hamming Distance based parsimony score. The input to the problem consists of five species and a multiple alignment of their corresponding sequences (a), and a given phylogeny (b). Note that the phylogeny specification consists of both the tree topology and the assignments of species to its leaves. In this example, $n = 5$, and $m = 7$. Figures (c) and (d) show two examples of internal vertices assignments corresponding to the second state (column) of the exemplified SP instance (highlighted in (a) via a gray rectangle). A binary substitution scoring matrix is assumed in this scoring scheme, where two identical characters are given a score of 0 and two distinct characters a score of 1. Assignment (c) provides the minimal number of mutations, 1, whereas assignment (d) yields two character mutations.

Problem 2: Maximum Parsimony (MP) The Maximum Parsimony (MP) problem is to seek, among all possible phylogenies over a given set of leaves, the phylogeny that yields the best SP score. Similarly to SP, the input to this problem is a multiple alignment of n input sequences of length m each. However, here the topology is not given. The MP problem is NP-Hard (Day and Sankoff, 1986; **?**). A straightforward approach for solving MP is to enumerate all possible phylogenies over the set of leaves and then employ an SP algorithm on each phylogeny. This approach was used by Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967) who showed that the number of phylogenies over n leaves is $(2n-3)!! = 1 \times 3 \times 5 \times \cdots \times (2n-3)$. Several constrained variants of MP were studied over the years (Felsenstein and Felenstein, 2004). The most famous among them is the Perfect Phylogeny problem, which is also NP-Hard (**?**) and for which FPT algorithms were proposed (**??**).

Measuring SP and MP complexity in terms of basic operations. SP and MP algorithms work by computing some information for every internal vertex of the input phylogeny. This information, as well as the complexity of its computation, depend on the scoring scheme employed by the parsimony algorithm. Thus, in what follows, we will use the term *basic operation* to denote the work invested in the computation of the information of a single vertex of a considered phylogeny for a specific scoring scheme. For example, in the Fitch SP algorithm (Fitch, 1971), which computes a minimal Hamming Distance SP score, an $O(m)$-time basic operation is applied, while in the Sankoff algorithm (Sankoff, 1975), which optimizes an SP score of minimal weighted edit distance, an $O(m\Sigma^2)$-time basic operation is applied, where Σ denotes the size of the alphabet spelling the input sequences.

Our Contribution In this work, we examine the complexity of MP in terms of the total number of basic operations executed throughout the run of the algorithm. Using this measure, we analyze the worst case complexity of three approaches to MP. The first, basic approach, is the one proposed by Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967), which performs $(n-1) \cdot (2n-3)!!$ basic operations. The second approach is based on the Hendy and Penny (Hendy and Penny, 1982) MP search space, which interleaves the SP computations within the tree-space development flow. This search space was originally proposed for the purpose of a branch-and-bound MP search, and its theoretical worst-case bound was not previously properly bounded. *Our first result is Theorem 2.2 in Section 2, in which we analyze the basic operations complexity of the Hendy and Penny approach, and show that it is faster than the Cavalli-Sforza and Edwards approach by a factor of $\Theta(\sqrt{n})$.*

Both Hendy and Penny's work as well as follow-up exact branch and bound extensions (Felsenstein and Felenstein, 2004) still kept the same traditional order of search space development, based on the Cavalli-Sforza and Edwards tree enumeration order. *In order to further improve MP efficiency, we propose to turn the state development order of the classical MP search tree "upside down", i.e. from the classical top down incremental tree extension by a single edge per each new state, to a bottom-up agglomerative approach which merges two previous subtrees per each state.* This idea is based on the observation that the "top down" approach to MP search goes "against the grain" of the update operations applied by the Hendy and Penny SP subroutines per each new search space node.

Due to this, the savings by the Hendy and Penny approach in avoiding overlapping redundant SP computations across subtrees shared by distinct phylogenies at level n, is weakened by the work applied for the sake of search-space information maintenance. Such maintenance requires, per each new node in the search space, the traversal and updates of all vertices along the path from the newly added leaf up to the root of the phylogeny represented by the new node (see Figure 2).

In order to avoid this inefficiency, we suggest a "bottom up" directed MP search, based on subtree merging, which flows along with the natural "bottom up" direction of Small Parsimony. By design, traversing the search space according to our proposed search tree requires performing exactly one basic operation per node of the search tree (the basic operation is applied on the new root of the two merged trees in the agglomerative step). Thus, the complexity of our new MP algorithm is equal to the size of our proposed search space tree. We show that this size is approximately $e \cdot (2n-3)!!$, where e is the base of the natural logarithm. *Thus, our new approach yields a basic operations complexity which is smaller by a factor of $\Theta(\sqrt{n})$ than the complexity of the Hendy and Penny approach (Theorem 3 in Section 4).* We note that the bound obtained by our proposed algorithm is on the same order of the number of topologies considered by MP, and therefore any approach to exact MP that must enumerate all topologies will not improve our result.

We note that the original work of Hendy and Penny assumed a symmetric scoring matrix. Under the assumption of symmetric scoring matrices, it can be shown that rooted MP can be solved via the simpler unrooted MP. However, such is not the case in practice, when the restriction to symmetrical scoring matrices is removed (Rodriguez *et al.*, 1990; Takahata and Kimura, 1981; Gojobori *et al.*, 1982; Tajima and Nei, 1984), and the tree rooting becomes meaningful. The algorithms we propose and analyze in this paper generalize the previous most efficient exact solutions to this problem and do not assume the use of symmetrical scoring matrices.

The rest of the paper proceeds as follows. First, in Section 2 we analyze the basic operations complexity of the algorithm of Cavalli-Sforza and Edwards, and the algorithm of Hendy and Penny. In Section 3 we describe our new approach to search space construction, and in Section 4 we analyze the basic operations complexity of our search space. Finally, we give our concluding remarks in Section 5.

2 Analysis of Previous Approaches

Throughout the paper, a *phylogeny* is a rooted binary labeled tree. Each leaf in the tree has a distinct label from $\{1, \ldots, n\}$, where n is the number of leaves. We also define a *forest* to be a collection of rooted binary labeled trees. Each leaf in the forest has a distinct label from $\{1, \ldots, n\}$, where n is the number of leaves in all the trees.

We will later define another type of trees called search space trees. In order to make the text more clear, we will use different terminology for the two types of trees: we will use *vertex* for phylogenies and *node* for search space trees.

2.1 The algorithm of Cavalli-Sforza and Edwards

The algorithm of Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967) enumerates all phylogenies with n leaves, and then solves the Small Parsimony problem on each tree. Cavalli-Sforza and Edwards showed that the number of phylogenies with n leaves is $(2n-3)!!$. Moreover, each phylogeny has exactly $n-1$ internal vertices. The following complexity bound is obtained.

Theorem 1 (Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967)). *The basic operations complexity of the algorithm of Cavalli-Sforza and Edwards is* $(n-1) \cdot (2n-3)!!$.

The enumeration of all phylogenies can be modeled by a *search space tree*. The search space tree $\mathcal{T}_n^{\mathrm{CSE}}$ consists of n levels (see Figure 2). The nodes of level $i-1$ correspond to all phylogenies with i leaves. For a node v of the search space tree, denote by F_v the phylogeny that corresponds to v. A node v of level $i-1$ has $2i-1$ children defined as follows: For each edge e in F_v, there is a child v_e of v whose corresponding phylogeny F_{v_e} is obtained from F_v by splitting the edge e into two edges connected in a new vertex x, whose additional child is a new leaf with label $i+1$. The node v has an additional child v' whose corresponding phylogeny $F_{v'}$ is obtained from F_v by adding a new root vertex x. The children of x are the root of F_v and a new leaf (with label $i+1$). From the definition of the search space tree, it is clear that level i of the tree contains $(2i-3)!!$ nodes, and in particular, there are $(2n-3)!!$ leaves in the tree. Thus, the number of phylogenies with n leaves is $(2n-3)!!$. The enumeration of the phylogenies with n leaves is achieved by performing a traversal of the search space tree, and building the phylogeny F_v when reaching each node v.

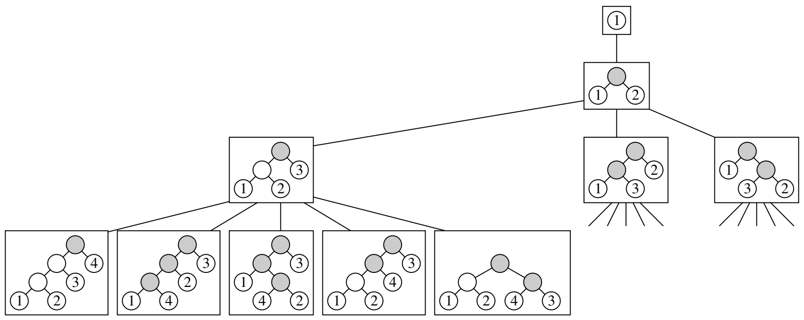

Fig. 2. An example of the search space tree $\mathcal{T}_4^{\mathrm{CSE}}$. Only some of the nodes of the search space tree are shown. Highlighted in gray within the phylogenies in each node of the search tree are the vertices updated by basic operations when applying Hendy and Penny's MP approach on $\mathcal{T}_4^{\mathrm{CSE}}$.

2.2 The algorithm of Hendy and Penny

Hendy and Penny (Hendy and Penny, 1982) interleaved the Small Parsimony operations within search space tree traversal. Their algorithm traverses the tree $\mathcal{T}_n^{\mathrm{CSE}}$ and when reaching a node v, it solves the Small Parsimony problem for the phylogeny F_v. However, the Small Parsimony algorithm does not need to compute the required information for all internal vertices of F_v since the information computed for the phylogeny of v's parent can be used. More precisely, let u be the parent of v, and let x be the newly added leaf in F_v. If y is an internal vertex of F_v that is not an ancestor of x, then the information for the vertex y in F_v is identical to the information for the vertex y in F_u. Since the latter was already computed, in order to solve the Small Parsimony problem on F_v, only vertex information for the ancestors of x needs to be computed. For a vertex v in $\mathcal{T}_n^{\mathrm{CSE}}$, let $\textsc{NumAnc}(v)$ denote the number of ancestors of x in F_v, where x is the newly added leaf in F_v. Therefore, the basic operations complexity of the algorithm of Hendy and Penny is $\sum_{v\in\mathcal{T}_n^{\mathrm{CSE}}} \textsc{NumAnc}(v)$ (the summation is performed over nodes from all levels of the search tree). We next give a formula for the summation above.

Definition 1. *Let H_i be the sum of $\textsc{NumAnc}(v)$ for all nodes v in level $i+1$ of $\mathcal{T}_n^{\mathrm{CSE}}$.*

By definition, $\sum_{v\in\mathcal{T}_n^{\mathrm{CSE}}} \textsc{NumAnc}(v) = \sum_{i=1}^{n-1} H_i$.

Lemma 1. *$H_i = (2i)!! - (2i-1)!!$.*

Proof. Let v be a node at level $i+1$ in $\mathcal{T}_n^{\mathrm{CSE}}$, and let x be the newly added leaf in F_v. Let F' be the forest obtained from F_v by removing x and its ancestors, and removing all the edges incident on these vertices. Note that every tree in F' is a subtree of a distinct ancestor of x in F_v, and every ancestor of x corresponds to exactly one tree in F'. Therefore, $\textsc{NumAnc}(v)$ is equal to the number of trees in F' (see Figure 3 for an example). The latter number is called the cover number for $\{1,\ldots,i\}$ in F_v in (Ochiumi *et al.*, 2011). It is shown in (Ochiumi *et al.*, 2011) that the sum of the cover number for $\{1,\ldots,i\}$ in all phylogenies with $i+1$ leaves is $(2i)!! - (2i-1)!!$. $\square$

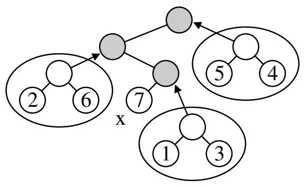

Fig. 3. An example for the proof of Lemma 1. The leaf x has 3 ancestors, which is equal to the number of trees in the forest obtained by removing x and its ancestors.

Theorem 2. *The basic operations complexity of the algorithm of Hendy and Penny is* $\Theta(\sqrt{n}(2n-3)!!)$.

Proof. By Lemma 1, the complexity is $\sum_{i=1}^{n-1}((2i)!! - (2i-1)!!)$. We have

$$\begin{aligned}
&\lim_{n\to\infty} \frac{\sum_{i=1}^{n-1}((2i)!! - (2i-1)!!)}{\sqrt{n}(2n-3)!!} \\
&\qquad = \lim_{n\to\infty} \left(\frac{\sum\limits_{i=1}^{n-2}((2i)!! - (2i-1)!!)}{\sqrt{n}(2n-3)!!} + \frac{(2n-2)!! - (2n-3)!!}{\sqrt{n}(2n-3)!!} \right) \\
&\qquad \overset{(1)}{=} \lim_{n\to\infty} \frac{(2n-2)!! - (2n-3)!!}{\sqrt{n}(2n-3)!!} = \lim_{n\to\infty} \frac{(2n-2)!!}{\sqrt{n}(2n-3)!!} \\
&\qquad = \lim_{n\to\infty} \frac{(2n)!!}{\sqrt{n}(2n-1)!!} \overset{(2)}{=} \lim_{n\to\infty} \frac{(2n)!}{\sqrt{n}((2n-1)!!)^2} \\
&\qquad \overset{(3)}{=} \lim_{n\to\infty} \frac{(2n)!}{\sqrt{n}\left(\frac{(2n)!}{2^n \cdot n!}\right)^2} = \lim_{n\to\infty} \frac{(2^n n!)^2}{\sqrt{n}(2n)!} \overset{(4)}{=} \lim_{n\to\infty} \frac{2\pi n(\frac{2n}{e})^{2n}}{\sqrt{n}\sqrt{4\pi n}(\frac{2n}{e})^{2n}} = \sqrt{\pi}.
\end{aligned}$$

Equality (1) is true since

$$\sum_{i=1}^{n-2}((2i)!! - (2i-1)!!) = (2n-4)!! - 1!! + \left(\sum_{i=2}^{n-2}(2i-2)!! - (2i-1)!! \right) \leq (2n-4)!!.$$

Therefore, the limit of the summation above divided by $\sqrt{n}(2n-3)!!$ is 0. Equality (2) follows from the equality $(2n)! = (2n)!!(2n-1)!!$. Equality (3) follows from the equalities $(2n)!! = 2^n \cdot n!$ and $(2n)! = (2n)!!(2n-1)!!$. Finally, equality (4) follows from Stirling's formula. □

3 A new, more efficient search space tree

In this section we present a new, "bottom up" search space enumerating MP. In order to define our new search space tree, we first build a directed acyclic graph $\mathcal{G}_n$. We will later transform $\mathcal{G}_n$ into a tree $\mathcal{T}_n$ by removing some of the edges.

We define a *merge operation* on a forest F to be an operation that generates a new forest F' by adding a new vertex to the forest and hanging two trees of F on this vertex. In the graph $\mathcal{G}_n$, every node v corresponds to a forest with n leaves. The graph contains a single node at level 0 whose corresponding forest consists of n singletons. The nodes of level i in the graph correspond to all the forests that can be obtained from the forests that correspond to the nodes of level $i-1$ by single merge operations. There is an edge (u, v) in the graph if and only if F_v is obtained from F_u by a merge operation.

Note that a node in $\mathcal{G}_n$ can have several incoming edges. We next transform $\mathcal{G}_n$ into a tree $\mathcal{T}_n$, by selecting exactly one edge from the edges entering a node, and removing all other edges.

For a tree T in a forest define the *label* of T to be

$$\text{label}(T) = \min\{\text{label}(v) : v \text{ is a leaf in } T\}.$$

The label of a forest F is

$$\text{label}(F) = \min\{\text{label}(T) : T \text{ is a non-singleton tree in } F\}.$$

For a node u in the graph $\mathcal{G}_n$, let T_u denote the tree in F_u for which $\text{label}(T_u) = \text{label}(F_u)$. Using the definitions above we can define the search space tree $\mathcal{T}_n$ (see Figure 4).

Definition 2. *The search space tree $\mathcal{T}_n$ is a tree whose nodes are the nodes of $\mathcal{G}_n$. A node v is the parent of a node u in $\mathcal{T}_n$ if F_v is obtained from F_u by deleting the root of T_u.*

The definition above gives an implicit characterization for the children of a node. We now show explicit characterization.

Lemma 2. *A node u is a child of a node v in $\mathcal{T}_n$ if and only if F_u is obtained from F_v by merging two trees T_1 and T_2 from F_v with $\text{label}(T_1) < \text{label}(T_2)$ and either*

1. *$T_1 = T_v$ (and in particular, T_1 is not a singleton), or*
2. *T_1 is a singleton and $\text{label}(T_1) < \text{label}(T_v)$.*

Proof. ($\Rightarrow$) Let u be a child of v in $\mathcal{T}_n$. By definition, deleting the root of T_u gives the forest F_v. In other words, if we denote by T_1 and T_2 the two trees formed from T_u by removing its root, then F_u is obtained from F_v by merging T_1 and T_2. Without loss of generality, assume $\text{label}(T_1) < \text{label}(T_2)$. By definition, $\text{label}(T_u) = \min(\text{label}(T_1), \text{label}(T_2)) = \text{label}(T_1)$. The tree T_u is the non-singleton tree of F_u with minimum label. It follows that in F_u, the leaves with labels $1, \ldots, \text{label}(T_1) - 1$ are each inside a singleton. Since F_u is obtained from F_v by a single merge operation, these leaves are also in singletons in F_v. If T_1 is non-singleton then it is the non-singleton tree in F_v with minimum $\text{label}(\cdot)$ value. Thus, $T_1 = T_v$, namely case 1 of the lemma occurs. If T_1 is a singleton then, since the leaves with labels $1, \ldots, \text{label}(T_1)$ are in singletons in F_v, we obtain that $\text{label}(T_v) > \text{label}(T_1)$ and case 2 of the lemma occurs.

($\Leftarrow$) Suppose F_u is obtained by merging two trees T_1 and T_2 from F_v, and let T' denote the tree obtained by this merge. In both cases of the lemma we have that the leaves with labels $1, \ldots, \text{label}(T_1) - 1$ are each inside a singleton in F_v, and therefore these leaves are also in singletons in F_u. Therefore, T' is the non-singleton tree in F_u with minimum label, namely $T' = T_u$. Thus, v is the parent of u in $\mathcal{T}_n$. □

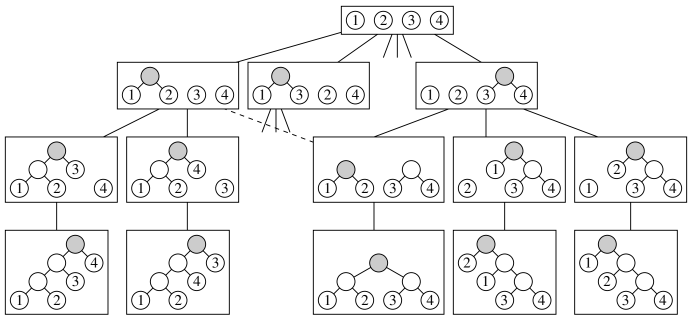

Fig. 4. An example of the search space tree $\mathcal{T}_4$. Only some of the nodes of the tree are shown. The dashed line is an edge of $\mathcal{G}_4$ that is not an edge of $\mathcal{T}_4$. Highlighted in gray within the phylogenies in each node of the search tree are the vertices updated by basic operations when applying our MP approach on $\mathcal{T}_4$.

4 Complexity of the new search space

We have proposed a new search method for MP. We now wish to bound the basic operations complexity of our approach. By design, traversing the search space according to our search tree requires performing exactly one basic operation per node: For a node v, the basic operation is applied on the new root of the two merged trees in F_v. Thus, the basic operations complexity is equal to the size of $\mathcal{T}_n$. We will next show that the size of $\mathcal{T}_n$ is approximately $e \cdot (2n-3)!!$.

Definition 3. *Let A_i^n denote the number of nodes in level i of $\mathcal{T}_n$, and let $L_{i,k}^n$ denote the number of nodes v in level i for which* $\text{label}(F_v) = k$.

Example 1. $A_0^n = 1$, $A_1^n = \binom{n}{2}$, and $A_{n-1}^n = (2n-3)!!$.

Note that for a node v in level i of $\mathcal{T}_n$, the forest F_v contains $n - i$ trees.

Observation 1 *If v is a node in level i then* $\text{label}(F_v) \leq n - i$.

Proof. The forest F_v is obtained from the forest of n singletons by a sequence of i merge operations. At least one merge operation must involve a tree containing a leaf with label at most $n-i$. Thus, the label of the resulting tree is at most $n-i$. This tree can participate in other merge operations, which either decrease or do not change the label of the tree. At the end of the merge operations the label of the tree is at most $n-i$ and thus $\text{label}(F_v) \leq n-i$. □

Observation 2 $A_i^n = \sum_{k=1}^{n-i} L_{i,k}^n$.

Lemma 3. $L^n_{i+1,k} = (n-i-k)\sum_{l=k}^{n-i} L^n_{i,l}$.

Proof. First, we count the number of children with label k for a node in level i with label l (we will soon show that this number does not depend on the topology of the corresponding forest). Let v be some node in level i with label l. Note that the forests corresponding to the children of v have labels at most l. For a child u of v for which $\text{label}(F_u) = k$, the forest F_u is obtained by merging two trees T_1 and T_2 from F_v such that $\text{label}(T_1) = k$ and $\text{label}(T_2) > k$ (note that $T_1 = T_v$ if $k = l$, and otherwise T_1 is a singleton). It follows that the number of children of v with label k is the number of trees T in F_v with $\text{label}(T) > k$. Since $k \le l$ and the leaves with labels $1, \dots, l-1$ are in singletons, it follows that there are exactly k trees with label at most k. The total number of trees in F_v is $n-i$ and therefore F_v contains $n-i-k$ trees with label greater than k. Thus, the number of children of v with label k is $n-i-k$ (not depending on the topology nor on l).

From the above we can deduce a recursive formula for $L^n_{i+1,k}$. A node in level $i+1$ with label k is the child of a node in level i with label $l \ge k$. The lemma follows. □

Lemma 4. $L^n_{i,k} = (2i-1)!!\binom{n-k+i-1}{2i-1}$.

Proof. We prove the lemma using induction on i. For the base of the induction we need to show that $L^n_{1,k} = n-k$ for all $k = 1, \dots, n-1$. Fix some k. The forest corresponding to a node in level 1 with label k is obtained from the forest of n singletons by merging a singleton with label k with a singleton with label larger than k. Therefore, $L^n_{1,k} = n-k$.

We now prove the induction step. Assume that the lemma holds for $i' < i$. By the induction hypothesis and Lemma 3,

$$L^n_{i,k} = (n-i-k+1)\sum_{l=k}^{n-i+1} L^n_{i-1,l} = (n-i-k+1)\sum_{l=k}^{n-i+1}(2i-3)!!\binom{n-l+i-2}{2i-3}$$
$$= (n-i-k+1)(2i-3)!!\sum_{m=0}^{n-i-k+1}\binom{2i-3+m}{2i-3}.$$

Using the equality $\sum_{m=0}^{n-k}\binom{k+m}{k} = \binom{n+1}{k+1}$ we obtain

$$L^n_{i,k} = (n-i-k+1)(2i-3)!!\binom{n+i-k-1}{2i-2} = (2i-1)!!\binom{n+i-k-1}{2i-1}. \square$$

Lemma 5. $A^n_i = (2i-1)!!\binom{n+i-1}{2i}$.

Proof. From Lemma 3 and Observation 2 we have $L^n_{i,1} = (n-i)\sum_{l=1}^{n-i+1} L^n_{i-1,l} = (n-i)A^n_{i-1}$. Therefore, $A^n_i = \frac{L^n_{i+1,1}}{n-i-1}$. By Lemma 4, $A^n_i = \frac{(2i+1)!!\binom{n+i-1}{2i+1}}{n-i-1} = (2i-1)!!\binom{n+i-1}{2i}$. □

Let μ_n be the number of nodes in a search space tree $\mathcal{T}_n$. Then, $\mu_n = \sum_{i=0}^{n-1} A^n_i = \sum_{i=0}^{n-1}(2i-1)!!\binom{n+i-1}{2i}$. The series (μ_n) has already been studied, and the following result was obtained (Grosswald, 1978).

$$\lim_{n\to\infty}\frac{\mu_n}{e(2n-3)!!} = 1 \qquad (1)$$

We now compare the size of our search tree to the size of the search tree $\mathcal{T}_n^{\mathrm{CSE}}$. Let η_n be the number of nodes in $\mathcal{T}_n^{\mathrm{CSE}}$. Then, $\eta_n = \sum_{i=1}^{n}(2i-3)!!$. It is easy to verify that

$$\lim_{n\to\infty} \frac{\eta_n}{(2n-3)!!} = 1 \qquad (2)$$

From equations (1) and (2) we obtain that $\lim_{n\to\infty} \frac{\mu_n}{\eta_n} = e$.

Finally, since solving MP using our search tree requires performing exactly one basic operation per node of the search tree, we obtain the following theorem.

Theorem 3. *The basic operations complexity for solving MP using the new search space is* $(1+o(1))\cdot e\cdot(2n-3)!!$. *Moreover, this complexity is* $\Theta(\sqrt{n})$ *times smaller than the complexity of the algorithm of Hendy and Penny.*

Proof. The theorem follows from Theorem 2 and Equation (1). □

5 Conclusions

We studied the classical problem of exact Maximum Parsimony (when, in the worst case, all phylogenies need to be enumerated), focusing on the worst case time complexity of various algorithms for the problem.

The first approach we analyzed, proposed by Cavalli-Sforza and Edwards (Cavalli-Sforza and Edwards, 1967), yields a basic operations complexity of $(n-1)\cdot(2n-3)!!$.

The second approach we analyzed was proposed by Hendy and Penny. Its theoretical worst-case complexity has not been previously analyzed. We showed that the basic operations complexity of this approach is smaller by a factor of $\Theta(\sqrt{n})$ than the complexity of the Cavalli-Sforza and Edwards approach.

We also proposed a new, faster MP approach, whose basic operations complexity is smaller by a factor of $\Theta(\sqrt{n})$ than the complexity of the Hendy and Penny approach, and by a factor of $\Theta(n)$ than the complexity of the Cavalli-Sforza and Edwards approach. We note that the bound obtained by our proposed algorithm is on the same order of the number of topologies considered by MP, and therefore any approach to exact MP that must enumerate all topologies will not improve our result.

Note that our analysis is done for asymetrical scoring matrices. When the matrices are symmetrical, the algorithm of Hendy and Penny also achieves a running time of $\Theta((2n-3)!!)$ as the traversal to the root is no longer necessary. We emphasize again that this is not the case in practice, and asymmetrical matrices are often used today in various applications of phylogenetics.

Further note that, due to the complexity of MP, when the input data is big it is common in practice to apply either heuristic search or exact optimization methods (such as e.g. branch and bound or intelligent search), in combination with tree-scoring functions, in order to identify a reasonably good tree that fits the data. However, the theoretical questions we ask in this paper, even though focused on a linear factor within the complexity of an NP-Hard problem, are at the core of the analysis of phylogenetic search. Thus we believe that our results could also be reflected, after further study of appropriate heuristics and search optimizations, to the practical, sparsified search-space solutions applied by current state-of-the-art character-based phylogenetic reconstruction tools.

Acknowledgments

The research of A.C, N.M-L and M.Z-U was partially supported by ISF grant 478/10. The research of A.C and D.T was partially supported by ISF grant 981/11. We thank Nimrod Milo for helpful discussions in early stages of this work.

Bibliography

Bastien, O., Roy, S., and Maréchal, É., 2005. Construction of non-symmetric substitution matrices derived from proteomes with biased amino acid distributions. *Comptes rendus biologies* 328, 445–453.

Blouin, M. S., Yowell, C. A., Courtney, C. H., and Dame, J. B., 1998. Substitution bias, rapid saturation, and the use of mtDNA for nematode systematics. *Molecular biology and evolution* 15, 1719–1727.

Cavalli-Sforza, L. L. and Edwards, A. W., 1967. Phylogenetic analysis. models and estimation procedures. *American journal of human genetics* 19, 233.

Day, W. H. and Sankoff, D., 1986. Computational complexity of inferring phylogenies by compatibility. *Systematic Biology* 35, 224–229.

Felsenstein, J. and Felenstein, J., 2004. *Inferring phylogenies*, volume 2. Sinauer Associates Sunderland.

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

Gojobori, T., Ishii, K., and Nei, M., 1982. Estimation of average number of nucleotide substitutions when the rate of substitution varies with nucleotide. *Journal of Molecular evolution* 18, 414–422.

Grosswald, E., 1978. *Bessel Polynomials*.

Hendy, M. and Penny, D., 1982. Branch and bound algorithms to determine minimal evolutionary trees. *Mathematical Biosciences* 59, 277–290.

Müller, T., Rahmann, S., and Rehmsmeier, M., 2001. Non-symmetric score matrices and the detection of homologous transmembrane proteins. *Bioinformatics* 17, S182–S189.

Ochiumi, N., Kanazawa, F., Yanagida, M., and Horibe, Y., 2011. On the average number of nodes covering a given number of leaves in an unordered binary tree. *JCMCC-Journal of Combinatorial Mathematicsand Combinatorial Computing* 76, 3.

Rodriguez, F., Oliver, J. L., Marin, A., and Medina, J. R., 1990. The general stochastic model of nucleotide substitution. *Journal of theoretical biology* 142, 485–501.

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

Tajima, F. and Nei, M., 1984. Estimation of evolutionary distance between nucleotide sequences. *Molecular biology and evolution* 1, 269–285.

Takahata, N. and Kimura, M., 1981. A model of evolutionary base substitutions and its application with special reference to rapid change of pseudogenes. *Genetics* 98, 641–657.

Tamura, K., 1992. The rate and pattern of nucleotide substitution in drosophila mitochondrial DNA. *Molecular Biology and Evolution* 9, 814–825.

Tamura, K. and Nei, M., 1993. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular biology and evolution* 10, 512–526.