University of Szeged
Research Group on Artificial Intelligence

Evolutionary Tree Reconstruction and its Applications in Protein Classification

Summary of the PhD Thesis

by

Róbert Busa-Fekete

Supervisors:

Prof. János Csirik and Dr. András Kocsor

Szeged
2008

Introduction

This booklet summarizes the scientific results of the author of the PhD dissertation entitled "Evolutionary Tree Reconstruction and its Applications in Protein Classification". In a wider sense the dissertation concentrates on two key topics, namely artificial intelligence and bioinformatics. Within these fields we focus on evolutionary tree reconstruction and machine learning.

Over a hundred years the theory of evolution has became the most acknowledged model of how animal and plant species have developed over time. The discipline which deals with the modelling of evolution is called phylogenetics (the word is originated by the conjunction of the Greek words: phyle = tribe, race and genesis = birth). The methods which are in widespread use in phylogenetics represent the process of species evolution by a so-called phylogenetic tree, which corresponds to a weighted tree-graph where the leaves represent the biological objects of interest. In connection with the reconstruction of these kinds of trees, several problems arise which are interesting from both a computer scientific and a biological point of view.

Earlier phylogenetics focused just on the evolution of species based on morphological characters, but nowadays the explosive advancement in molecular biology also requires the study of proteins. The wealth of sequenced protein data allows us to perform novel investigations. The possibility of comparing protein sequences has moved research work towards the systematization of the proteins isolated from distinct species. Proteins that share a high sequence identity or similarity support the hypothesis that they share a common ancestor, and therefore we call them evolutionary related or homologous proteins. The analysis of evolutionary-related proteins has become a key question in phylogenetics. After our brief introduction we can state the basic goal of the phylogenetics: to reconstruct an appropriate tree topology based on protein sequences which have a high sequence similarity. We should mention here that the high sequence similarity of proteins usually implies that they share common functionality as well, but it does not logically follow.

As the dissertation consists of two parts, the author's results will also be split into two parts. In the *first part*, we introduce evolutionary tree reconstruction methodologies.

Several tree building method have been worked out and some of them have become widely used, for example the Neighbor Joining (NJ) [1] and the Unweighted Pair Group Method with Arithmetic mean (UPGMA) [2]. These methods belong to the so-called distance-based or distant matrix methods because they reconstruct the evolutionary history of biological objects based only on pre-determined or observed distance values among them. Our Multi-Stack (MS) [3] algorithm methodologically falls also into this category. Broadly speaking, the MS method finds a weighted tree topology that predicts the observed set of distances as closely as possible. More precisely, a weighted tree defines a distance value for all pair of leaves –i.e. the sum of the weights of edges containing the path between them. Thus the output tree of the MS approach we expect from the distances defined by itself will differ from the observed distances as small as possible. To find this tree is an NP-complete problem when we have an arbitrary distance measure [4], hence it can only be applied to heuristical solutions. The idea behind the MS method is that it builds the optimal tree for the subsets of the proteins of interest, and then it joins these subtrees in an iterative manner. We can apply this bottom-up approach efficiently in many test scenarios, and the MS approach often outperforms many traditional tree building methods.

Since there are many tree building methods which produce more than one possible evolutionary history, or the different tree building methods reconstruct different trees, in many cases it is necessary

to use those methodologies which are able to reconstruct one "representative" tree based on several different phylogenetic trees. These kinds of methods are called the consensus tree methods [5], and they are usually applied as the last step of the phylogenetic analysis process.

In general, each inner point in a rooted phylogenetic tree determines a subset of the biological object of interest (i.e. the objects which are represented by those leaves in the tree which lie below the inner point). Exploiting this observation we can see that the concept of a phylogenetic tree and the concept of a hierarchical set system are equivalent. The hierarchical set systems consist of those subsets or, in other words, clusters which are pairwise compatible. Thus each phylogenetic tree corresponds to a pairwise compatible cluster set. Most of the consensus methods determine a compatible subset of the cluster sets of the input trees in different ways, based on the cardinality of clusters' occurrences in the input trees. Their calculations can be done in polynomial time. Our goal is to find the subset of the input clusters for which the total number of the cluster occurrences is maximal. Furthermore, we can also define an arbitrary (not necessarily occurrence-based) weighting function on the clusters of the input trees. We solved this consensus tree building approach efficiently [6], and we showed that it can perform a more precise phylogenetic analysis than the traditional consensus methods such as the Majority-Rule, the Strict and Greedy consensus methods[7].

In the *second part* of this thesis we apply the tree building methods in protein classification problems. Automated protein classification is a crucial task in today's biology. The unknown genes/proteins of the distinct organisms can be retrieved and stored in the form of character sequences that are several hundred in length. Nowadays, it has become routine to compare this data to the sequences of known proteins/genes using a method of *approximate string matching*. Then, applying a machine learning method, the unknown protein can be classified into a known category (e.g. a structural or functional category) [8]. The automated data annotation system of the frequently mentioned genome research is based on this methodology.

In this thesis we seek to develop novel and efficient protein classification algorithms. Our basic assumption is that the structure of the biological datasets can be represented by a phylogenetic tree, and using this representation protein classification can be carried out significantly efficiently [9; 10]. The protein classification methods, which also use phylogenetic information, belong to the field of phylogenomics [11], hence our methods can be viewed as phylogenetic algorithms as well.

I. Evolutionary inference methods

The definition of a phylogenetic tree

The evolutionary history of biological objects (e.g. species, genes, proteins, genomes) in general can be represented as a tree structure, where each leaf corresponds to a biological object of interest. These biological objects will be denoted by X. The inner points of the tree can be regarded as hypothetical ancestors. In the terminology we follow, a phylogenetic tree is a leaf-labelled tree[12].

Definition 1 *A* **phylogenetic tree** *is an ordered pair $\mathcal{T} = (T; \phi)$, where T is a tree (acyclic connected graph) whose vertex set $V(T)$ contains at most one element which is of degree two and $\phi : X \to L(T)$ is a bijection between the leaf set $L(T)$ of T and the set X. The function ϕ is called the labelling function. A phylogenetic tree is a rooted phylogenetic tree if it has a vertex $r \in V$ which is of degree two. The vertex r is called as the root of the phylogenetic tree.*

Definition 2 *If every non-root interior point of a rooted phylogenetic tree $\mathcal{T}$ is of degree three, then $\mathcal{T}$ is called a* **binary phylogenetic tree**.

Definition 3 *A phylogenetic tree is a* **weighted phylogenetic tree** *if there is a non-negative real function on its edge set: $w : E(T) \to \mathbb{R}_{\geq 0}$.*

Any two vertices of a tree can be joined by a path. If we demand that any edge on a path occurs at most once, then the path is a simple path. In a tree there is precisely one simple path for two vertices which connects them.

Corollary 1 *Let us denote a weighted phylogenetic tree by $\mathcal{T}$. A pair of elements will be denoted by $x, y \in X$ and the simple path which joins them in $\mathcal{T}$ will be denoted by $p(x, y)$. Then the weight function of $\mathcal{T}$ can be used to define a distance function on the set X:*

$$d_{\mathcal{T}}(x, y) = \sum_{e \in p(x,y)} w(e)$$

This distance function is called the patristic distance or leaf distance over the set X.

Next, we look for that evolutionary history or phylogenetic tree which can represent the evolutionary relationship for a given criteria in a better way. The interior vertices of a phylogenetic tree can be interpreted in many ways, depending on the type of the biological object studied. For example, if we reconstruct a tree for a set of genes, then each interior points can be interpreted as a so-called evolutionary event like a gene duplication or gene specialization.

We should mention here that if $RB(n)$ stands for the set of all rooted phylogenetic tree over the set $|X| = n$, then $|RB(n)| = (2n - 3)!!$ [13]. This means that the size of tree space grows at a superexponential rate with the size of the taxon set. This is why elementary approaches, such as exhaustive search, are hardly suitable even for a small taxon set.

Distance-based approach

Numerous approaches have been developed which seek to reconstruct phylogenetic trees. The explosive advancement in molecular biology requires the development of the phylogenetic methods. The phylogenetic tree reconstruction methods can be roughly classified into three main types: distance-, sequence and quartet-based methods [1; 14; 15]. In this dissertation we present a novel distance-based algorithm for this task.

In the case of the distance-based approach, we shall assume that just the similarity distances between the biological objects are available [13]. Thus, before we perform a phylogenetic analysis we need a distance function. Using this function we can express numerically the 'similarities' or even the 'dissimilarities' between the set of objects X. If we have a formal definition for a 'similarity' measure, which can express the similarities of sequences of fixed length, then a monotone decreasing transformation of the similarities can be used as a distance-like measure, and vice-versa. Several distance functions are now commonly used which define distance values between character strings over a given alphabet.

In the course of evolution the changes in the sequences are called mutations. For the assessment of how two sequences are related from an evolutionary point of view, we need to identify these changes in the sequences. To do this the most commonly used method is the general alignment model, which in bioinformatics is also known as the Needleman-Wunsch algorithm. The method itself carries out a pairwise alignment for the sequences. Based on this alignment, we can determine the number of positions where two sequences are identical or differ from each other. Several alternative alignment methods have been developed for this, like the Smith-Waterman and the Gotoh algorithms [16; 17].

After the alignment step, the distances of sequences are calculated based on a time-continuous Markov Chain [18; 19]. These models determine the evolutionary distances of sequences based on the number and type of mutations which were identified during the pairwise alignment of sequences. This approach can take into account the case when in a certain position duplicated mutation takes place. For example, $A \to C$ occurs in one position, and then there is a $C \to A$ mutation again.

Alignment-free sequence distances are also applied in different areas of bioinformatics, but they are not so common as the alignment-based ones [20]. Perhaps the simplest alignment-free sequence distance, which also has low computational requirements, is the relative entropy of the distribution of nucleic or amino acids in a given set of sequences of interest. In early automatic protein classification research this was mainly applied as the sequence comparison method.

The distance-based tree building algorithms have a close connection with the clustering methods. The main goal here is also to find groupings of target objects based on a predefined similarity measure/distance function. The elements of the groupings are called clusters. Numerous traditional agglomerative clustering methods are in use for phylogenetic analysis like the Single Linkage (SL), Complete Linkage (CL) and UPGMA algorithms. The application of these methods in tree building is very useful because the clustering we get is represented as a tree structure. Here the known distance-based tree building method is the Neighbor-Joining method, which is methodologically very similar to the above-mentioned SL and CL clustering methods. The NJ method can be considered as a divisive method, namely it constructs the phylogenetic tree via a bottom-up clustering procedure. The reasons for the success of the NJ method are being studied nowadays, over 20 years after it was first presented. Several of its advantages and disadvantages have been recently pointed out[21; 22].

The tree reconstruction methods outlined above belong to the class of agglomerative hierarchical clustering algorithms, since during each iteration they divide a cluster (top-down) or join two clusters (bottom-up approach) based on a function defined on the clusters. Furthermore, the tree building methods assign edge weights to the edges of the output tree. Since the vertices of a phylogenetic tree represent biological objects (the inner points represent only hypothetical objects), the edge lengths or edge weights represent evolutionary distances of objects. If we would like to compare the outputs of the distance-based methods, then which tree should we select? One way of answering this is by investigating the difference between the predefined distance values on X and the patristic distance defined by a phylogenetic tree over X. Hence let us define the tree error $e_{\mathcal{T}}$ of a phylogenetic tree $\mathcal{T}$ for a distance function d by the following formula:

$$e_{\mathcal{T}} = \sum_{x,y \in X} (d(x,y) - d_{\mathcal{T}}(x,y))^2 \tag{1}$$

Then the so-called path-edge incidence matrix $P_{\mathcal{T}}$ for a phylogenetic tree $\mathcal{T}$ having n leaves is given by:

$$P_{\mathcal{T}}(p,e) = \begin{cases} 1 & \text{, if } \quad e \in p \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

whose columns correspond to the edges of T, while the rows correspond to the paths between the leaves of T. Clearly this matrix has $n-1$ columns and $\binom{n}{2}$ rows. Figure 1 shows a simple example for the construction of the path-edge incidence matrix. Then by solving an optimization problem we can obtain the minimal tree error, where the x vector contains the optimal edge weights of T, and the vector d contains the distance values according to the P_T topology matrix [1]:

$$e_{\mathcal{T}} = \min_{\mathbf{x} \in \mathbb{R}_+^{\mathbf{n-1}}} \| \left(P_T \mathbf{x} - \mathbf{d} \right) \| \tag{3}$$

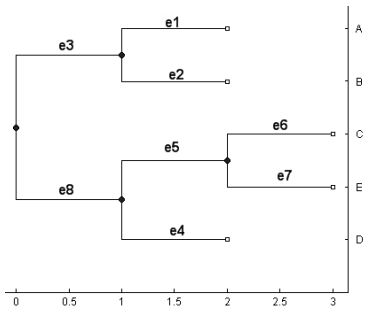
For a given tree $\mathcal{T}$ having n leaves, the minimal tree error $e_{\mathcal{T}}$ can be calculated in $O(n^4)$ time using the Least Squares methods[15]. But later it was shown that the tree error can be calculated in $O(n^2)$ time [23]. In the literature this distance-based criteria was applied to get an optimal edge weighting after a phylogenetic analysis [24].

Based on the minimal tree errors the weighted phylogenetic trees over a set X can be ranked. Day showed that the problem of finding tree over a set X which has a minimal tree error is in general NP-complete [4]. This is why we applied the so-called Multi-Stack (MS) approach in our work [3]. This approach is used in the field of speech recognition, and we adopted this method to reconstruct phylogenetic trees because the MS approach is very suitable for finding the heuristic solution of problems which have an enormous solution space.

The initial step of our MS method includes the exploration of all tree topologies with at most three leaves, since for arbitrary distance values and for any tree with fewer than three leaves there is an edge weighting such that the tree error will be zero (so the optima in Eq. 3 will be zero). After the initial step, the method iteratively joins the examined subtrees whose tree error has been calculated.

[1] The arrangement according to the topology matrix P_T determines an unambiguous ordering among the $\binom{n}{2}$ entries of the vector $\mathbf{d}$.

In the kth iteration the MS algorithm joins the trees which have k, or fewer than k leaves using a tree joining operator. Repeatedly carrying out this iterative step we gradually get the kinds of trees which have an increasing number of leaves. In the MS algorithm we assign a separate stack to the trees which have the same number of leaves and store the K best candidate subtrees in the stack according to their tree errors. Hence we can explore a relevant part of the tree space quite efficiently.



	e1	e2	e3	e4	e5	e6	e7	e8
(A,B)	1	1	0	0	0	0	0	0
(A,C)	1	0	1	0	1	1	0	1
(A,D)	1	0	1	1	0	0	0	1
(A,E)	1	0	1	0	1	0	1	1
(B,C)	0	1	1	0	1	1	0	1
(B,D)	0	1	1	1	0	0	0	1
(B,E)	0	1	1	0	1	0	1	1
(C,D)	0	0	0	1	1	1	0	0
(C,E)	0	0	0	0	0	1	1	0
(D,E)	0	0	0	1	1	0	1	0

Figure 1: A phylogenetic tree and its edge-path incidence matrix, where $X = \{A, B, C, D, E\}$.

Consensus tree methods

Let us introduce the consensus tree methods and then actively exploit the analogy between the rooted phylogenetic trees and the hierarchical set systems.

Definition 4 *Let M be a finite set. A subset $\mathcal{H}$ of subsets of M is a* hierarchy *if for all $A, B \in \mathcal{H}$,*

$$A \cap B \in \{\emptyset, A, B\} \tag{4}$$

We call the elements of hierarchy $\mathcal{H}$ pairwise compatible.

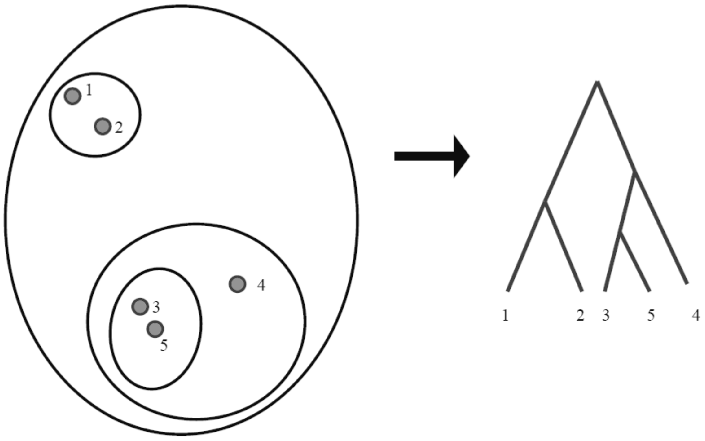


Figure 2: The analogy between the phylogenetic trees and the hierarchical set systems.

Since each inner point of a phylogenetic tree $\mathcal{T}$ over a set X assigns a subset of the set X (the elements can be found below it), we can naturally map the inner points of $\mathcal{T}$ onto the elements of a hierarchy over X (Figure 2). In this way we get a bijection between the two structures. In the latter $\mathcal{T}^C$ stands for the hierarchy belonging to the rooted phylogenetic tree $\mathcal{T}$. As the consensus tree methods result in one 'representative' tree from many phylogenetic tree, we can describe the basis of these methods such that they determine a pairwise compatible subset from the union of many hierarchies.
The most widely used consensus tree methods are briefly described below. Their detailed description and some of their properties can be found in the seminal paper by David Bryant [7]:

1. Strict consensus: it simply collects those subsets from the input set which are common to all input phylogenetic trees.

2. Majority-rule consensus: it simply collects those subsets from the input set which can be found in more than the half of the input phylogenetic trees.

3. Greedy consensus: first, it sorts the subsets in descending order according to their frequencies (i.e. the number hierarchy of trees they appeared in). Then it iteratively adds the subset with the highest frequency to our consensus subset set if it obeys the restriction of being compatible with all previously added subsets.

All of these methods determine a compatible subset in different ways based on the cardinality of subset occurrences in the input hierarchies. However, we can devise a more general approach as well. Let us assume that there is a real function w^C defined on $\mathcal{C} = \bigcup_i \mathcal{T}_i^C$ such that it maps $\mathcal{C}$ onto the non-negative real numbers. Our goal here is to determine a $\mathcal{C}^{'} \subseteq \mathcal{C}$ for which $\sum_{c \in \mathcal{C}'} w(c) \rightarrow \max$. This approach is known as the Max Clique Consensus problem, and it is NP-complete when the number of input trees is more than two [25].

We created a binary integer programming formulation for the MCC problem. After, we solved it using the well-known Branch&Bound algorithm [26] efficiently. Thus the MCC approach turned out to be suitable for phylogenetic analysis.

In our study [6] we investigated the performance of this methodology using many evolutionary models. We also tested this method when the input trees were obtained from many different tree building methods, then we introduced a Maximum Likelihood (ML) based weighting function.

The experiments clearly show that it is worth using this ML based weighting with the MCC consensus tree method, as in numerous test scenarios this outperforms the other consensus tree methods. However, we should mention here that the Strict Consensus method was better than the other consensus method in cases where the input trees were very different in size and type.

The assessment of the accuracy of the phylogenetic analysis

The evolutionary history of proteins is usually unknown, and often there is no consensus among the experts. Therefore it is very important to work out methods which can assess the performance of a phylogenetic analysis, and allow us to compare the phylogenetic tree reconstruction methods. Here we introduce a simple testing protocol that is suitable for this purpose. When we implemented the testing

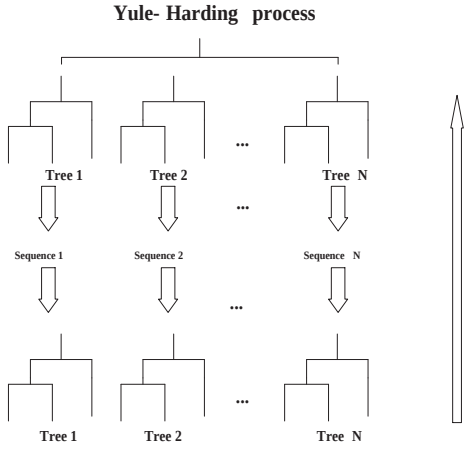


Figure 3: The process of testing phylogenetic tree reconstruction methods.

protocol we took into account many biological observations, hence the advantages and disadvantages of the tree building methods can be easily seen.

The core of the testing process consists of three main steps:

1. We construct a model tree having N leaves whose edge lengths are generated according to some distribution (e.g. a standard exponential distribution).

2. Based on an evolutionary model (e.g. Kimura-2-parameter model) we construct N sequences according to the branching pattern of the model tree.

3. We build a tree using the tree building method of interest for the generated sequences. Then we compare this output tree to the model tree using a tree similarity measure. In this way we can evaluate the performance of the tree reconstruction method.

In step 1 we could choose a model tree based, for example, on an uniform distribution or we could apply the Yule-Harding process [27; 28]. This process has many attractive features. For instance, it is more likely to produce balanced trees, so generating a caterpillar tree has a higher probability than if we use a uniform distribution in the tree space. In the second step we can use some of the evolutionary models to generate sequences. Quite a few evolutionary models are available for this purpose [13]. In the last step we build a tree using the tree building method of interest. By comparing the output tree to the model tree we can infer the accuracy of the tree building. There are many tree similarity metrics in use. If we are only interested in the similarity of the topology then we should use the Robinson-Foulds distance[29]. If we apply this testing protocol repeatedly, we can get a reliable measurement for the accuracy of the tree building method. Hence we can benchmark the different tree reconstruction methods using this testing protocol.

I/1. Thesis

The author developed a Multi-Stack based phylogenetic tree building method which makes use of the least-squares criteria. In this way he produced a novel algorithm which is competitive with the conventional used distance-based tree building methods, and it can reconstruct the evolutionary history of

datasets in a better way when the biological objects (sequences of interest) have a lower similarity [3]. This improvement can be shown using evolutionary distances and alignment-free sequence distances. In addition, the MS method achieves better results in many test scenarios than those obtained using the Fitch-Margoliash algorithm, which also applies the least-squares criteria.

I/2. Thesis

The author solved the Max Clique Consensus *problem via a binary integer programming task. With this approach one can find the compatible subsets of an arbitrary weighting of subsets that have maximal weights. In addition, the author introduced a novel Maximum Likelihood weighting scheme, which leads to an efficient phylogenetic reconstruction technique. He tested this method with different evolutionary models and found that this approach in many cases outperforms the standard consensus tree building methods [6]. The trees in the tests were generated by the widely-used PAUP program package[30], and the consensus methods were compared with each other on these trees. After, the author compared the consensus methods on a real-life database.*

I/3. Thesis

The author created a testing framework where the different phylogenetic reconstruction techniques could be compared using different evolutionary models over a wide range of data sizes [3; 6]. In this testing methodology, the biological sequences (DNA or protein) are generated based on a predetermined model evolutionary tree. Next, the tree-building method of interest is applied on this set of sequences, and it produces an output tree, which is then compared to the predetermined model tree. Based on the similarity values of these trees we can estimate the accuracy of the given tree reconstruction method. This testing framework provides a more comprehensive testing environment than the bootstrap method [13] because here we can examine the efficiency of the tree-building method using different evolutionary models.

II. Phylogenetic tree-based protein classification methods

Eisen first made use of evolutionary trees in protein classification, and he called this novel approach phylogenomics [11]. Up to now many methods have been presented in the field of bioinformatics. But most of these methods cannot cope with large-scale datasets in many cases, because in general they are not just needed for pairwise sequence similarities but, for example, they are needed to identify the gene duplication events or they require information from a multiple alignment. This is why the application of these kind of methods in a real-life protein classification system is very time consuming and difficult.

In our work we present many methods which seek to solve the protein classification task using phylogenetic trees. In our approach we require just the similarities of the given sequences. We store these similarity relationships in a weighted phylogenetic tree structure. All of the methods we introduce here are based on the assumption that we can represent the similarity relations of proteins in a better way using a weighted phylogenetic tree.

When we introduce the methods below, we assume that we have a dataset with a known class labels. This dataset will also be called an a priori database or known dataset. These classes have a biological meaning, e.g. they represent a functional category or a structural class. Moreover, we will also assume that we have an element with an unknown class label which is called the query element or simply the unknown element, which will be classified.

TreeInsert and TreeNN methods

The TreeInsert method is based on the assumption that we can construct the real phylogenetic tree of a given known protein database. This rooted weighted phylogenetic tree $\mathcal{T}$ is constructed using the similarities $d(x, y)$ between the proteins contained the known database. Afterwards we try to place the query element q into the phylogenetic tree $\mathcal{T}$. But to find the best place for q we need to measure the "amount of fitting" of q into the original tree $\mathcal{T}$. Following the usual conventions of phylogenetics, we insert the q query element into the tree $\mathcal{T}$ as a leaf node. Below Figure 4 shows an insertion of a the query element, where L_i stands for the leaf which represents the ith proteins from the known dataset, and p_i denotes the parent of L_i before the insertion. After the insertion of a q query element, p_i' will be the common ancestor of L_i and L_q will represent the query element. The variables x, y and z are the unknown edge lengths.

With this line of thinking, we need to determine the amount of fitting of an unknown element into the phylogenetic tree $\mathcal{T}$. This is why we define the insertion cost of q for a leaf L_i. This insertion cost will depend on the difference between the patristic distance of $\mathcal{T}$ after the insertion of the unknown element and the d similarity values. Now let us calculate the optimal fitting of the patristic distance between q and the other leaves and the similarities $d(x, y)$:

$$\min_{0 \leq x,y} \left(\sum_{j=1}^{n} \left(d\left(L_j, L_q\right) - d_{\mathcal{T}}\left(L_i, L_q\right) \right) \right)^2 \tag{5}$$

$$\text{s.t.} \quad y + z = d_{\mathcal{T}}\left(p_i, L_i\right)$$

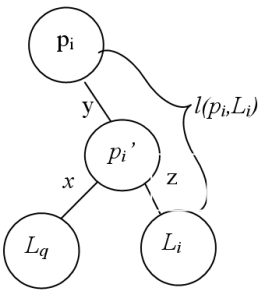


Figure 4: The insertion of an unknown q element as a leaf node of $\mathcal{T}$ next to the leaf L_i.

These constraints ensure that the leaf-distance between L_i and its parent remains unchanged. The insertion cost for a leaf L_i shall be denoted by $IC(L_q, L_i)$, and it will be the optimal value of x, when the formula in Eq. 5 is minimal.

The TreeInsert algorithm inserts the q query elements next to each leaf into the tree separately then, based on the minimal insertion cost, it determines the optimal place of q. In this way we can place the unknown element into the evolutionary history of the a prior dataset. After, the TreeInsert method assigns a class label of that leaf to the query where the insertion cost was minimal.

The TreeNN method is a simpler method than TreeInsert. In this approach we also assume that the pairwise similarities between the elements of the a prior dataset D are known. The q query elements will be compared to each elements of the known dataset using the same similarity measure. Next, TreeNN constructs a weighted phylogenetic tree $\mathcal{T}$ using a tree building method for D and q together . Based on the patristic distance of $\mathcal{T}$ we can perform the classification of q by using the Nearest Neighbor method[31], or another like it.

Investigating these methods from a protein classification point of view, the TreeInsert method first builds a weighted tree $\mathcal{T}$ for the a prior dataset, and then it looks for the place of each test element in $\mathcal{T}$ where the insertion cost is minimal. In contrast with this, the TreeNN method reconstructs a tree for a known dataset and each test element separately. TreeInsert achieves good results in protein classification, but we need to perform n optimization tasks for the calculation of the insertion cost of a test elements, where n denotes the size of the a prior dataset. We compared these methods with several model evaluation techniques, and they achieved quite similar results[9].

TreeProp-N

Several methods are in use where the similarities of studied objects are stored in a special graph structure, such as a complete graph or tree graph. Then these methods carry out a so-called propagation on this graph structure, which means that the adjacent graph nodes send a message to each other. These messages usually correspond to real numbers. These kinds of propagational methods are, for example, the PageRank [32], Message Passing [33], Affinity Propagation [34] algorithms. In our study we used a special form of the PageRank method, namely the Personalized PageRank algorithm [35; 36], as our starting point. This method has been mostly applied in Information Retrieval purposes, and it has been used with great success by the Google WEB surfer perhaps being the best

known example. The main idea behind the Personalized PageRank method is the following: if we have a query element whose similarities to the elements of known dataset are known, then let us also take into account the similarity relationships of the dataset at the classification of the query element.

Similar to the notation used above, let us denote the unknown elements by q, and the known dataset by $D = \{y_1, y_2, \ldots, y_n\}$. The vector

$$y(0) = (d(q, y_1), d(q, y_2), \ldots, d(q, y_n))$$

contains the original similarity values between the a prior database and the unknown element. The similarities within the database D will be rewritten into a matrix form $S = d(y_i, y_j)$. And the $y(t)$ vector will contain the similarity values after the tth iteration. Next, using the propagation rule described in Eq. 6, we can get updated similarity values for the query elements where the similarity relationships within the database D has been considered as well. The matrix S contains these similarity relationships, and this matrix can be used in a simple way in the propagation rule:

$$y(t+1) = (1-\alpha)y(0) + \alpha S y(t) \tag{6}$$

This iterative method is quite slow in practice, because in many cases we have to work with large-scale networks, and we have to carry out the propagation for each query element. This is why we substitute the network by a more sparse structure, namely an unrooted weighted binary phylogenetic tree $\mathcal{T}$ which is built up for the D and q, as we did in the case of TreeNN. Since we construct a phylogenetic tree for n elements, our network will have $2n-1$ points, but each point is of degree of three at most.

Let $y_i(0)$ denotes the patristic distance of the query q and the ith element of D. In addition, the set of points $N(p_i)$ will be the neighbors of p_i in $\mathcal{T}$. The propagation rule we shall has the following form[2]:

$$y_i(t+1) = (1-\alpha)y_i(0) + \alpha \sum_{p \in N(p_i)} d_{\mathcal{T}}(p, p_i) y_p(t). \tag{7}$$

This approach is very similar to the Personalized PageRank method. There the similarities within the database are represented by a weighted complete graph, but here we represent the similarity relationships by a weighted binary phylogenetic tree.

This method can be applied easily in protein classification. Let us denote the y^* limit point of the $y_1, y_2, \ldots, y_T, \ldots$ convergent point series[3] defined in Eq. 7. The classification can be carried out based on the maximal component of y^*. If we set the number of iterations to zero, then this method will be equivalent to the TreeNN method, because y_0 consists of patristic distances.

TreeProp-E

In our study we also have developed another extension of TreeProp-N. The TreeProp-E method carries out the propagation on the weight of edges based on a very similar propagation rule like TreeProp-N. Let us denote the edge weights of a weighted phylogenetic tree $\mathcal{T}$ by $y(0)$ vector, whose ith

[2]Since this propagation can be performed on similarities, we need to use a monotone decreasing function on the patristic distance of $\mathcal{T}$.

[3]The convergence can be easily obtained using the Perron-Frobenius theorem.

component will be $y_i(0)$. We say two edges are adjacent in a tree when they have a common point. The adjacent edges of an edge e will be denoted by $N(e)$. After this, we shall adapt the following propagational rule for the edge of the tree $\mathcal{T}$:

$$y_i(t+1) = (1-\alpha)y_i(0) + \frac{\alpha}{|N(e_i)|}\sum_{e_j \in N(e_i)} y_{e_j}(t). \tag{8}$$

The idea behind this method is very similar to the motivation of the original PageRank algorithm. In a weighted phylogenetic tree the edge weights represent similarities, and an edge will be elongated –it will get a bigger edge weight– if it has more adjacent edges which represent bigger similarities.

We can use this method in protein classification: we will reconstruct a tree for the query element q and the known database D, and then we will carry out the tree-based propagation on the tree in the way described in Eq. 8. After the propagation[4] we can calculate the patristic distance between the query element and the known database, and we can perform the classification using a Nearest Neighbor classifier. In many cases in our experiments TreeProp-E outperformed the traditional machine learning algorithms like Artificial Neural Networks [37], Support Vector Machines [38].

Receiver Operating Characteristic (ROC) analysis

ROC analysis is the most commonly used evaluation technique for protein classification models, which is applied in a wide range in bioinformatics [39] as well as in signal processing [40]. In protein classification there are usually multi-class classification tasks. But a multi-class task can be considered as many one versus all classification tasks, where one particular class is considered as the positive class while the rest of the samples are treated as negatives. A binary classification task can be handled by a binary classifier, which has trained on the training set. This binary classifier assigns a so-called class conditional probability to all elements to be classified. Based on the class conditional probability of the positive class, we can rank the test elements. Then each real value $t \in [0,1]$ splits the test set into two in a natural way: we consider a test element as negative if it has a value below t, otherwise it will be considered as positive. So each t can be treated as a decision threshold, and this decision threshold influences the prediction of the binary classifier. When we also take into account the real class labels of the test elements, then we can divide the test elements into four groups, depending on the value of t:

1. The binary classifier predicts the test element as positive, and its real class label is positive. Then this element is a true positive element. The set of these test elements shall be denoted by $TP(t)$.

2. The binary classifier predicts the test element as positive, but its real class label is negative. Then this element is a false positive element. The set of these test elements shall be denoted by $FP(t)$.

3. The binary classifier predicts the test element as negative, but its real class label is positive. Then this element is a false negative element. The set of these test elements shall be denoted by $FN(t)$.

[4]The experiments showed clearly that the value of $y_i(t)$ does not really change after 20 iteration.

4. The binary classifier predicts the test element as negative, and its real class label is negative. Then this element is a true negative element. The set of these test elements shall be denoted by $TN(t)$

Using these notations we can define the False Positive Rate (FPR) and the True Positive Rate (TPR), which depend on the value of t as well:

$$TPR(t) = \frac{TP(t)}{TP(t) + FN(t)}, \qquad FPR(t) = \frac{FP(t)}{FP(t) + TN(t)} \tag{9}$$

Then the definition of the ROC curve is the following.

Definition 5 *Given a square $N = [0,1]^2$, and the values of $TPR(t)$ and $FPR(t)$ for all $t \in [0,1]$, the ROC curve is the point set of $(TPR(t), FPR(t))$.*

The Area Under Curve (AUC) can be interpreted as the probability of the event when our classifier predicts an element as positive and its real class label is also positive[40]. This is why this model evaluation technique is more reliable than, for example, the simple accuracy one. Figure 6 shows a simple example of the application of an ROC analysis.

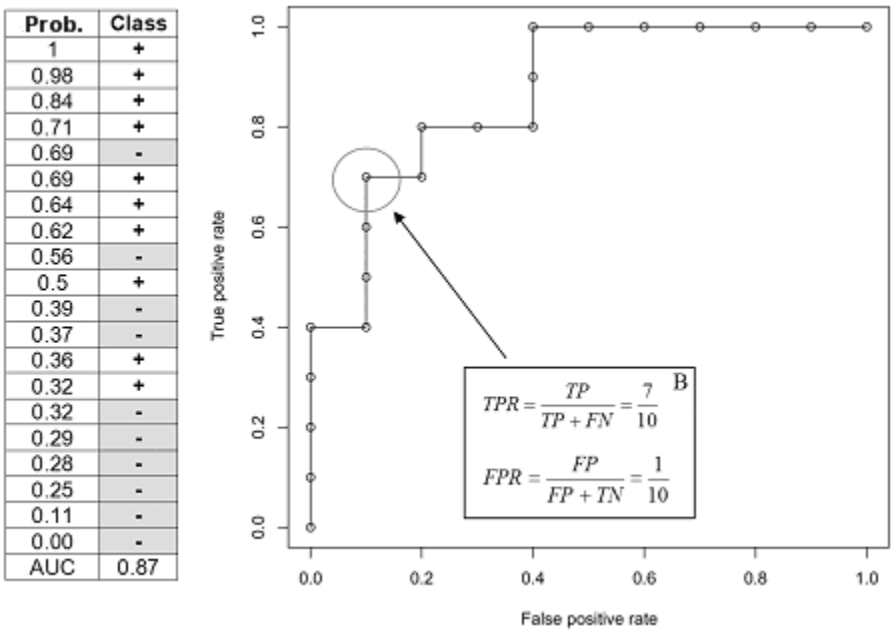


Figure 5: The calculation of the ROC curve on ranked objects. On the left hand side of the picture one can see the probabilities obtained by the binary classifier and beside it are the real class labels of the corresponding objects. Using these the ROC curve can be plotted, as we have done on the right hand side.

In protein sequence classification ROC analysis is also a commonly applied technique, but here we need to evaluate sequence similarities. Hence we compare the query element to the known dataset using a sequence similarity, and then based on these similarities we can arrange the elements of the known dataset in decreasing order. Afterwards we can apply ROC analysis for this ranking. Here the top list of this ranking is more important from the point of evaluation, because we expect that there

will be more positive elements at the top of the list than later on in the raking. This is why we can limit the top list so as to include n negatives (where n is usually taken as some plausible number like 10, 50 etc.) [39]. These are the so-called ROC_n (e.g. ROC_{10}, ROC_{50}) values.

The main difficulties with the protein datasets are that the known classes (structural of functional classes) are very different in size and in inner class similarities. So it is very hard to find the appropriate n values for the ROC_n analysis. We propose a method for the setting of the value of n [41]. Due to this method we can then get a more reliable AUC value for the problematic protein classes, and ROC analysis will not be so sensitive to the size of the training set. This method can be very useful during the development of the protein classification databases.

II/1. Thesis

The author introduced the TreeInsert and TreeNN methods, which are novel tree-based protein classification algorithms. In contrast to the earlier methods, the algorithms he introduced here make use of just the sequence similarities. Thus they are readily applicable in a wide range on protein classification tasks. The author compared the tree-based methods on many protein classification tasks using ROC analysis, and they were often significant better. The experiments showed that it is worth applying phylogenetic information in protein classification. [9].

II/2. Thesis

The author devised two tree-based propagational methods, namely TreeProp-N and TreeProp-E. These methods may be regarded as extensions of TreeNN, because all of these methods update the sequence similarities using the topology of a phylogenetic tree. In experiments these propagational algorithms usually gave a better performance in protein classification comparing to the former systems [10].

II/3. Thesis

The author created a ROC analysis-based evaluation method which is a more reliable model evaluation technique than the original ROC analysis when the distribution of the classes is imbalanced. Applying it, a model selection could be carried out more reliably than with the other approaches[41]. He tested this approach on several large-scale datasets.

Conclusions

For understanding the language of genes and proteins we have to find a suitable model of how they have evolved in the course of evolution. Because of this we need to develop tree building methods which discover the process of evolution. These kinds of methods have gained importance with the advent of molecular biology in the 1970's. Thereafter the implosive advancement in biology allows us to investigate the sequences of the proteins, genes, as well as species/genomes. This is why the microbiological research requires novel and novel phylogenetic analysis tools.

In the first part of this thesis we provided two phylogenetic tree reconstruction methodologies. In the second part, to demonstrate the application of phylogenetic tree reconstruction methods in automatic protein classification, then we introduced protein classification algorithms which make use of phylogenetic tree building methods as well.

The main goal of the first part was to introduce methodologies which can perform a highly accurate phylogenetic analysis. The Multi-Stack algorithm categorically is a distance-based method. Thus it uses only the distance values of the sequences of interest to build a phylogenetic tree. This method is suitable, for example, in constructing a guide tree before multiple alignment.

The second phylogenetic analysis tool was a consensus tree building method, namely the Max Clique Consensus method. It is obvious from the results that the MCC consensus outperforms many widely-used procedures, and it was easy to implement. The time requirement of this method is reasonable (proportional to the tree building method itself), so it can be employed efficiently in a post-processing phase of a phylogenetic analysis tool.

In the second part of this thesis we sought to develop novel and efficient protein classification algorithms. Our basic assumption was that the structure of the biological datasets could be represented by a phylogenetic tree, and using this representation protein classification could be carried out significantly more efficient. This new field of bioinformatics is a very promising area of research.

References

[1] Saitou N. and Nei M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4(4):406–425, 1987.

[2] Rohlf F. J. Classification of aedes by numerical taxonomic methods (diptera: Culicidae). *Ann Entomol Soc Am*, 56:798–804, 1963.

[3] Busa-Fekete R., Kocsor A., and Bagyinka Cs. A multi-stack based phylogenetic tree building method. *Lencture Notes in Bioinformatics*, 4463:49–60, 2007.

[4] Day W.H.E. Computational complexity of inferring phylogenies from dissimilarity matrices. *Bulletin of Mathematical Biology*, 49:461–467, 1986.

[5] Adams E.N. Consensus techniques and the comparison of taxonomic trees. *Systematic Zoology*, 21:390–397, 1972.

[6] Busa-Fekete R., Bánhalmi A., Kocsor A., and Bagyinka Cs. A binary integer programming relaxation for the max clique consensus. *European Journal of Biophysics*, 2008.

[7] Bryant D. A classification of consensus methods for phylogenetics. *Bioconsensus,Discrete Mathematics and Theoretical Computer Science*, 61:163–184, 2001.

[8] Altschul S. F., Gish W., Miller W., Myers E. W., and Lipman D. J. Basic local alignment search tool. *J Mol Biol*, 215(3):403–410, October 1990.

[9] Busa-Fekete R., Kocsor A., and Pongor S. Tree-based algorithms for protein classification. *Computational Intelligence in Bioinformatics, Studies in Computational Intelligence*, 7:0–0, 2008.

[10] Kocsor A., Busa-Fekete R., and Pongor S. Protein classification based on propagation on unrooted binary trees. *Protein and Peptide Letters*, page in press, 2008.

[11] Eisen J.A. Phylogenomics: improving functional predictions for uncharacterized genes by evolutionary analysis. *Genome Res.*, 8:163–7, 1998.

[12] Semple C. and Steel M. *Phylogenetics*. Oxford University Press, 2003.

[13] Felsenstein J. *Inferring Phylogeneties*. Sinauer, 2004.

[14] Felsenstein J. Evolutionary trees from dna sequences: a maximum likelihood approach. *J Mol Evol*, 17(6):368–376, 1981.

[15] Fitch W. M. and Margoliash E. Construction of phylogenetic trees. *Science*, 155:279–284, 1967.

[16] Smith T. F. and Waterman M. S. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, March 1981.

[17] Gotoh O. An improved algorithm for matching biological sequences. *J Mol Biol*, 162(3):705–708, December 1982.

[18] Jukes T. H. and Cantor C. R. Evolution of protein molecules. *Mammalian Protein Metabolism, Academic Press, New York*, edited by H. N. MUNRO:21–132, 1969.

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

[20] Vinga S. and Almeida J. Alignment-free sequence comparison-a review. *Bioinformatics*, 19(4):513–523, March 2003.

[21] Gascuel O. and Steel M. Neighbor-joining revealed. *Molecular Biology and Evolution*, 23(11):1997–2000, November 2006.

[22] Bryant D. On the uniqueness of the selection criterion in neighbor-joining. *Journal of Classification*, (22):3–15, 2005.

[23] Bryant D. and Waddell P. Rapid evaluation of least-squares and minimum-evolution criteria on phylogenetic trees. *Journal of Biochemical and Biophysical Methods*, 15(10):1346–1359, 1998.

[24] Grunewald S., Forslund K., Dress A., and Moulton V. Qnet: An agglomerative method for the construction of phylogenetic networks from weighted quartets. *Molecular Biology and Evolution*, 24(2):532–538, February 2007.

[25] Bryant D. *Hunting for trees, building trees and comparing trees: theory and method in phylogenetic analysis*. PhD thesis, Dept. Mathematics, University of Canterbury, 1997.

[26] Land A.H. and Doig A.G. An automatic method for solving discrete programming problems. *Econometrica*, 28:497–520, 1960.

[27] Yule G. A mathematical theory of evolution. *Based on the conclusions of Dr. J. C. Willis. Philos. Trans. Roy. Soc. London Ser. B, Biological Sciences*, 213:21–87, 1925.

[28] McMorris F. R. and Powers R. C. Consensus weak hierarchies. *Bulletin of Mathematical Biology*, 53:679–684, 1991.

[29] Robinson D. F. and Foulds L. R. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1-2):131–147, 1981.

[30] Swofford D. Paup program package. *http://paup.csit.fsu.edu/index.html*, 2007.

[31] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000.

[32] Brin S. and Page L. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998.

[33] Lauritzen S. L. and Spiegelhalter D. J. Local computations with probabilities on graphical structures and their application to expert systems.

[34] Brendan J J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, January 2007.

[35] Page L., Brin S., Motwani R., and Winograd T. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[36] Zhou D., Weston J., Gretton A., Bousquet O., and Schölkopf B. Ranking on data manifolds.

[37] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[38] Vapnik V. N. *Statistical Learning Theory*. John Wiley and Son, 1998.

[39] Gribskov M. and Robinson N. Use of receiver operating characteristic (roc) analysis to evaluate sequence matching, 1996.

[40] Egan J.P. *Signal Detection theory and ROC Analysis*. New York: Academic Press, 1975.

[41] Busa-Fekete R., Kertsz-Farkas Attila, Kocsor A., and Pongor S. Balanced roc analysis (baroc) protocol for the evaluation of protein similarities. *Journal of Biochemical and Biophysical Methods*, page doi:10.1016/j.jbbm.2007.06.003, 2007.