Inferring Models of Rearrangements, Recombinations, and Horizontal Transfers by the Minimum Evolution Criterion

(extended abstract)

Hadas Birin [a], Zohar Gal-Or [a], Isaac Elias [b], Tamir Tuller [a] ⋆

[a] School of Computer Science
Tel Aviv University
{hadasbir, zohargal, tamirtul}@post.tau.ac.il
[b] School of Computer Science and Communication
KTH
isaac@csc.kth.se

Abstract. The evolution of viruses is very rapid and in addition to local point mutations (insertion, deletion, substitution) it also includes frequent recombinations, genome rearrangements, and horizontal transfer of genetic material. Evolutionary analysis of viral sequences is therefore a complicated matter for two main reasons: First, due to HGTs and recombinations, the right model of evolution is a network and not a tree. Second, due to genome rearrangements, an alignment of the input sequences is not guaranteed. Since contemporary methods for inferring phylogenetic networks require aligned sequences as input, they cannot deal with viral evolution. In this work we present the first computational approach which deals with both genome rearrangements and horizontal gene transfers and does not require a multiple alignment as input. We formalize a new set of computational problems which involve analyzing such complex models of evolution, investigate their computational complexity, and devise algorithms for solving them. Moreover, we demonstrate the viability of our methods on several synthetic datasets as well as biological datasets.

Key words: Phylogenetic network, horizontal gene transfer, genome rearrangements, recombinations, minimum evolution.

1 Introduction

Eukaryotes evolve largely through vertical lineal descent driven by local point mutations and genome rearrangements. Unlike the Eukaryotes, bacteria also acquire genetic material through the transfer of DNA segments across species boundaries—a process known as *Horizontal Gene Transfer* (HGT) [8]. In the presence of HGTs, the evolutionary history of a set of organisms is modelled by a *phylogenetic network*, which is a directed acyclic graph obtained by inferring a set of edges between pairs of edges in the organismal tree to model the horizontal

⋆ Corresponding author

transfer of genetic material [15] (see Figure 1). We call such a network a *rooted* phylogenetic network.

In the case of viruses, the evolutionary model is even more complicated. The viral genomes are usually compact and evolve very rapidly by all the aforementioned mutations in addition to a large number of recombinations [23], and rearrangements. Furthermore, in this case an organismal tree usually does not exist [23], thus the right model is an unrooted tree with an additional small set of undirected edges (between pairs of edges in the initial tree). We call such networks *unrooted* phylogenetic networks.

There are many strategies and models for dealing with non tree-like evolution, here we briefly describe some of them. For example, *Splits networks* (see e.g. [13]) are graphical models that capture incompatibilities in the data due to various factors, not necessarily HGT or hybrid speciation. Some works describe a phylogenetic network as probabilistic models and use maximum likelihood for analyzing it [25, 14], while others use maximum parsimony [15], or deal with the problem by a graph-theoretic approach of reconciling species and gene trees into phylogenetic networks [1]. None of the mentioned works deal with rearrangements.

In this work we advise a distance based method for inferring evolution under complicated models that can involve substitutions, insertions, and deletions of single nucleotides, rearrangement, HGT, and recombination. We believe that in our case, where the models of evolution are complex, distance methods have advantages for three main reasons: First, sometimes the appropriate probabilistic model is not completely clear, thus, using ML is not feasible. Second, by our experience [14, 15], usually ML and MP are more time consuming than distance methods even when considering complete HGT. If the models include HGTs together with rearrangements these methods are not feasible. Finally, MP and ML require multiple alignment as an input, while we want to separate our method from this requirement.

The multiple alignment problem is NP-hard [9], and to the best of our knowledge, at some stage of the processing most methods for inferring evolutionary networks require a multiple alignment. We believe that this requirement is problematic, especially with regard to complete viral genomes. Thus our method takes unaligned sequences as input.

Boc and Makarenkov suggested a distance based method for detecting HGTs [4], however, there are two main differences between this research and the work of Boc and Makarenkov: First, as opposed to their method, our models allow for rearrangements and recombination. Second, our models are sequence oriented (*i.e.* the input in our case is a set of sequences), while the approach of Boc and Makarenkov [4] requires *distance matrices* as input. Consequently, our work considers a more general and realistic setting.

Our methods are based on the following basic biological observations: 1) In phylogenetic networks each nucleotide evolves according to a tree (which may be different from the organismal tree) [12]. 2) Closely positioned nucleotides are more likely to have evolved according to the same tree than distantly positioned nucleotides [23]. Therefore, our method infer different trees to different subset of sequences, and partition the genomes into subsequences (each of at least a few dozens bp) and constrain the nucleotides in each subsequence to have the same

evolution.
Given an organismal tree and a set of sequences[1], our method finds families of homologue subsequences and reconstructs their evolutionary history by adding reticulation edges to the organismal tree while optimizing the minimum evolution criteria. This work does not handle gene duplication or deletion; dealing with these operations has been deferred to future works. However, as we demonstrate in this work, there are many interesting datasets that do not involve events such as duplication or deletion.

2 Definitions

Let $T = (V, E)$ be a tree, where V and E are the *tree nodes* and *tree edges*, respectively, and let $L(T)$ denote its leaf set. Further, let X be a set of taxa (species). Then T is a *phylogenetic* tree over X if there is a bijection between X and $L(T)$. Henceforth, we identify the taxa with their associated leaves and denote the set of leaf-labels with $[n] = \{1, .., n\}$. A tree T is said to be *rooted* if the set of edges E is directed and there is a single distinguished internal vertex r with in-degree 0. Let $S = [s_1, s_2, s_3, \ldots, s_n]$ be the sequences corresponding to the n taxa (note that these sequences may be of different lengths). A *family* over the set of sequences S is a set of sequences $S' = [s'_1, s'_2, s'_3, \ldots, s'_n]$, such that for all i, s'_i is a subsequence of s_i. The definition of the ACS distance between two sequences appear in [26].

Let $D(\cdot, \cdot)$ denote a distance measure between pairs of sequences. In this paper $D(\cdot, \cdot)$ is either the cost of the pairwise alignment or the ACS distance. Two sequences, s_1 and s_2, are considered *d-homologous* with respect to a block length L, if each sequence is longer than L, and every window of length L in their pairwise alignment has evolutionary distance $< d$; we denote this property $D_L(s_1, s_2) < d$.

A family of d-homologous subsequences is defined as a set of subsequences S' with the following property:
$\forall s'_1, s'_2 \in S' \quad D_L(s'_1, s'_2) < d$. A *non-overlapping* set of families is a set of families such that in each sequence, subsequences from different families do not overlap. The subsequence $s' \subseteq s$ that is part of the family f is denoted by $f(s)$. We call the set of subsequences that are induced by a set of families a *partitioning*.

A rooted phylogenetic network $N = N(T) = (V', E')$ over the taxa set X is derived from a rooted tree T by inferring *reticulation edges* between pairs of edges in T. That is, each reticulation edge is inferred by adding two new vertices on two edges of E and thereafter joining the two new vertices with the directed reticulation edge. A tree edge can take part in more than one reticulation event. In a similar way, an unrooted phylogenetic network is derived from an unrooted tree by adding undirected edges to the tree. Each family f of subsequences is related to a subset of the reticulation edges, denoted by $M(f)$, which describes the evolution of the family. If a family, f, evolves along the organismal tree then $M(f) = \emptyset$.

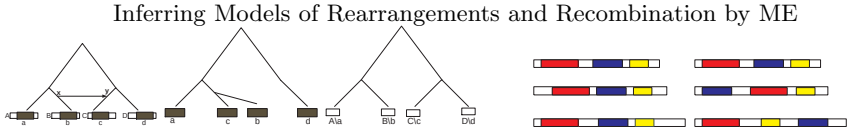
A rooted phylogenetic network must satisfy additional temporal constraints, such as acyclicity [14, 15]. Such temporal constraints do not exist in an unrooted

[1] If the organismal is not part of input we estimate it from the input sequences.

network. Finally, we denote the set of all trees contained inside the network N (rooted or unrooted) by $\mathcal{T}(N)$. In the case of rooted network, each such tree is obtained by the following two steps:
(1) for each node of in-degree 2, remove one of the incoming edges, and then (2) for every node x of in-degree and out-degree 1, whose parent is u and its child is v, remove node x and its two adjacent edges, and add a new edge from u to v. In the case of unrooted networks, a tree is obtain by removing an edge from each cycle of the tree, removing each node, x, with exactly two neighbors u and v, removing the two edges that include the node x, and adding a new undirected edge, (u, v). In our setting the tree, $T_f \in \mathcal{T}(N)$ that includes exactly all the reticulation edges in $M(f)$ describes the evolution of the family f.

In this work, we deal with the Minimum Evolution (ME) criteria [18]. It is known to be consistent when using the least-squares criterion [21] (as in this work); meaning that it converges to the correct tree for long enough sequences. In the case of evolutionary trees, the decision variant of the problem of finding the minimum evolution tree is defined as follows:
Problem 1 [7] Input: Set of n sequences, S, that induces a distance matrix B and a real number e.
Output: A tree, T, with total edge lengths less than e, while the edge lengths are least squares estimated from B.
The sum of edge lengths of a tree T is the ME score of a tree; Let $E(T, S, D)$ denotes the ME score for a tree T, with a set of sequences S corresponding to its leaves, and when D is used as a distance measure between pair of sequences. In our setting, we use the minimum evolution criterion to select the additional reticulation edges that best explain the evolution of each family of subsequences. That is, given a set of sequences S and a phylogenetic tree T, our goal is to find a set of non-overlapping families, a set of reticulation edges, and a mapping relating each family to a subset of the reticulation edges (*i.e.* one tree for each family). These are selected with the objective of minimizing the sum of minimum evolution scores for each family and associated tree. If the set of families is $F = [f_1, f_2, .., f_h]$, the set of reticulation edges is H, the mapping is M, and the pairwise distance measure between sequences is D , then we denote this score by $E(T, F, H, M, D) = \sum_i E(T_{f_i}, f_i, D)$. Let s'_1 and s'_2 denote two subsequences of the sequence s. We say that s'_1 precedes s'_2 ($s'_1 \prec s'_2$) if s'_1 ends before s'_2 begins. Under a non-rearrangement assumption, there is an order of the families, $f_1, f_2, .., f_h$, such that in each sequence, s_i: $f_1(s_i) \prec f_2(s_i), .., f_{h-1}(s_i) \prec f_h(s_i)$ (see figure 1), but this assumption is not always justified.

Here we deal with three variants of the problem, each related to different assumptions about the input: **1.** The first variant, *Non Rearrangement Given Tree* (NRGT), assumes an organismal tree and that subsequences have not been rearranged. An example of such input is a set of proteins and an organismal tree of bacteria. **2.** The second variant, *Rearrangement Given Tree* (RGT), assumes an organismal tree and that subsequences may be rearranged. An example of such input is a set of genomes and an organismal tree of bacteria. **3.** The third variant, *Rearrangement No Tree* (RNT), does not assume an organismal tree and subsequences may be rearranged. An example of such input is a set of viral genomes.
The output for the first two variants is a set of homologous non-overlapping



(a) (b) (c) (d) (e)

Fig. 1. (a)-(c): A simple example of a HGT or recombination event between the ancestral black sequences in the ancestral taxa x and y. (a) A phylogenetic network with a single HGT event (the directed edge) which describes the evolution of the black family of sub-sequences. (b) The tree of the horizontally transferred family. (c) The underlying organismal (species) tree which describe the rest of the sequences. (d)-(e): Families.(d) Set of families under the non-rearrangement assumption. (e) Set of families without the non-rearrangement assumption. The white parts in the two cases denotes families that evolve along the organismal tree, T, while the evolution of the colored families is along trees that are different than T.

families, a set of reticulation edges, and a mapping from each family to a subsets of reticulation edges (that is related to that family). In the third variant, the organismal tree is also part of the output.

3 Hardness Issues

In this section, we deal with the computational hardness of some variants of the problems that were mentioned in the previous section, and other related problems. Roughly, the problem can be divided into two subproblems [2]: (i) Dividing the set of input sequences into non-overlapping d-homologous families and (ii) Finding the best set of reticulation edges for each family. By the results that are presented in this section, it seems that these two problems are NP-hard.

A related problem is Binary Minimum Common String Partitioning (BMCSP); we will use the hardness result of this problem for establishing the hardness result of our problems. A *minimum common partitioning* of two binary strings s_1 and s_2 is given by the least number of blocks that s_1 has to be cut into such that these blocks can be reconcatenated to form s_2. Formally, $BMCSP$ is defined as follows:

Problem 2 [BMCSP] Input: Two binary strings, s_1 and s_2, an integer B.
Output: Can the sequence s_2 be formed from the sequence s_1 by cutting it into less than B subsequences and subsequently reconcatinating them.

The hardness of $BMCSP$ can be proved by a reduction from the APX-complete problem 2-MCSP [11], which is defined as follows (due to lack of space the full details of the proof are deferred to the full version of the paper):

Problem 3 [2-MCSP] [11] Input: Two strings of integers, s_1 and s_2, where each integer appears exactly twice in each sequence, and an integer B.
Output: Can the sequence s_2 be formed from the sequence s_1 by cutting it into less than B subsequences and subsequently reconcatinating them.

The hardness of the BMCSP problem implies the hardness of our problem. The decision variant of the RGT problem, which is defined as follows:

Problem 4 [RGT] [3] Input: A set of binary sequences S, a phylogenetic tree T,

[2] In practice these two problems are not independent.

[3] The problem NGT is defined in a similar way while the input does not include a tree T, the problem $NRGT$ is defined in a similar way but the families must be in order.

two integers h, and k, a real number c, and a distance measure between pairs of sequences, D.
Question: Is there a set, F, of h non-overlapping families $S'_1, .., S'_h : \forall_i S'_i \subset S$, a set, H, of k reticulation edges, and a mapping, M, from each family to subset of H , such that the score $E(T, F, H, M, D) \leq c$.

A reduction from $BMCSP$ can show that RGT and RNT are hard even when there are 0 reticulation edges (details are deferred to the full version of the paper).

Theorem 1. *RGT and RNT are NP-hard.*

As mentioned, in this work we deal with minimum evolution criteria (minimum evolution tree, or MET, see Problem 1. This problem is probably NP-hard for trees (it is still an open problem). It is easy to see that the $NRGT$ problem (even if there is only one family) is NP-hard if Problem 1 is NP-hard (details are deferred to the full version of the paper).

Observation 1 *NP-hardness of the problem MET implies NP-hardness of $NRGT$ (when there is no rearrangement and the tree is given).*

4 Algorithms and Parameters

In this section, we describe our method, *Find Net-Families*. As can be seen in Figure 2 this method consists of three stages, each of which solves a seperate computational problem. As was just shown, most of the problems we deal with are NP-hard and consequently the algorithms presented here are heuristics.
The input to Find Net-Families is an organismal tree. This tree is either provided by the user or generated by computing a distance matrix with the ACS [26] method and then building the associated neighbor joining tree [22]. In the first stage we generate good partitions to each of the $\binom{n}{2}$ input sequences. Thereafter, we greedily expand each of the partitionings to get a set of non-overlapping families that covers all of the input sequences. In the final stage, we improve the total minimum evolution score of the families by greedily adding reticulation edges to the organismal tree. In the final stage, we also adjust the boundaries of the subsequences representing the families .
Due to running time considerations we used a parameter L that constrains the length of each subsequence in each family to be a multiple of L [4]. The typical length of a gene is a few hundred nucleotides, usually only complete genes are horizontally transferred [6]. In the case of partial HGT or recombination, the lengths have similar order of magnitude [17, 3] (very short horizontal transfer are hard and somtimes impossible to detect by any method). Indeed using L of few hundreds nucleotides gave good results (usually changing L from one hundred to few hundreds does not change the results dramatically).

Finding d-Homologous Families in two Sequences Given two sequences s_1 and s_2 of length ℓ, our goal is to find d-homologous families where each block

[4] In practice the adjusting procedure allows lengths that are up to 10% different than this constraint.

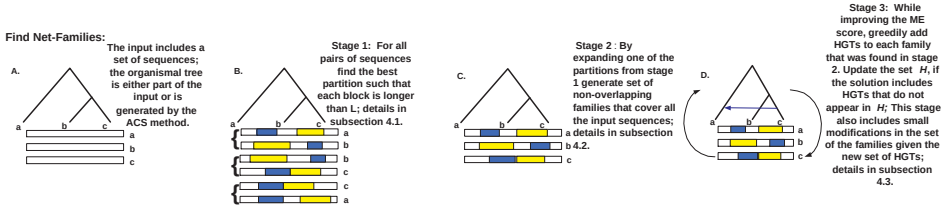


Fig. 2. A sketch of the main algorithm for finding families and a set of reticulation edges for each family.

should be longer than L, and such that d is minimal. Namely, we wish to match each block in one sequence with exactly one as similar as possible block in the other sequence. In this work we assume that there is one unique such matching. In practice, for large enough L (*i.e* more than few dozen characters) and when duplications are not present, this is indeed the case.

The procedure has two stages; in the first stage we search for common subsequences of length at least W, where W has to be tuned with respect to the input sequences. In general, too small W (*e.g.* $W = 2$) is not specific enough, since we expect that both non-homologue and homologue sequences share common subsequences of length W. On the other hand too large W (*e.g.* $W = 100$) is also problematic, since even homologue sequences do not share such long subsequences. In practice $12 < W < 18$ gave good results for nucleotides, and $7 < W < 12$ gave good results in the case of amino acids.

Let $S_i(s_1, s_2)$ denote the longest substring that starts in position i in s_1 and appears in the two sequence (we assume that $|S_i(s_1, s_2)| = O(1)$). In the first stage, we performed the following steps: **1.** Generate the suffix array for s_1. **2.** Scan s_2, in each position i, find the longest subsequence that starts in that position and appears in both sequence, $S_i(s_1, s_2)$. **3.** If $|S_i(s_1, s_2)| > W$ keep the position and the length of the matching substring.

In our implementation we used the "lightweight suffix array" of [5, 26] which is constructed in time $O(\ell \log(\ell))$. Step 2 of the algorithm above, for each position, i, can be accomplished in $\log(\ell)$ time by performing lexicographic binary search for $s_2(i)$ in the suffix array of s_1.

After the first stage we have a set of position-pairs for each common substring longer than W. In the second stage, we map each overlapping window of length L in the first sequence with the window in the second sequence which has the maximal sum of lengths of common substrings. We call each such match the *core of a family* $f \in F$. Finally, we greedily adjust the boundaries of each family by adding/removing small blocks at the ends of the windows while optimizing $\min_{F; s'(i), s'(j) \in F} D_L(s'(i), s'(j))$, such that in the end of this stage the two strings have been partitioned into families that cover all of the sequences. The runtime complexity of this stage is $O(\ell^2)$ for each pair of sequences. Thus the total runtime complexity for n sequences is $O(\ell^2 \cdot \binom{n}{2})$.

Finding a Family of d-homologous Subsequences From the previous stage we have a d-homologous partitioning for each $\binom{n}{2}$ pairs of sequences. In this stage the aim is to expand these pairwise matchings to families of d-homologous

subsequences with minimal d that cover all the n input sequences. As mentioned before, we assume that each window of length close to L in a sequence has *exactly one* homologue in each of the other sequences, an assumption which is supported by our biological inputs.
We examine the expansion of each of the $\binom{n}{2}$ partitionings of pairs of sequences to a partitioning over all the n sequences. This is done by the following steps:
1. For each of the $\binom{n}{2}$ partitionings of pairs of sequences.
a. Start with one partitioning.
b. The k-th ($k \leq n-1$) step: Greedily add another sequence to the partitioning of $k-1$ sequences that was generated in the previous step. This is done by checking consecutive overlapping windows of length L, and for each family choosing a non-overlapping window(s) (*i.e.* a subsequence of the new sequence) that includes the maximal sum of lengths of common subsequences that appear in the other members of this family in the $k-1$ previous sequences.
2. Chose the expansion that minimizes $\min_{F;s'(i),s'(j)\in F} D_L(s'(i), s'(j))$.
The runtime complexity of this stage is $O(\ell \cdot n^2)$ for each pair of sequences. Thus the total runtime complexity for n sequences is $O(\ell \cdot n^2 \cdot \binom{n}{2})$.
Adding Reticulation Edges and Refining the Partitioning to Families In this subsection we describe how to find the set of reticulation edges that are related to each family. In this stage we assume a given initial (organismal) tree and a set of d-homologous families. Each family induces a distance matrix. Our procedure greedily chooses one of the families and adds a new reticulation edge that is related to that family. In each such step the size of the set of reticulation edges that is related to one of the families is increased by one.
We plot a graph of the improvement in the ME score after each such step. Such a graph can help biologists to decide the actual number of reticulation edges. As is described in the simulation study, usually after adding the actual number of reticulation edges the improvement in the ME score is insignificant.
Given a tree topology (an organismal tree and a set of reticulation edges) and a set of sequences at its leaves (a family). We use the least square estimation to calculate the edge lengths of the tree. This can be done in the time complexity of an $n \times n$ matrix inversion [22], less than $O(n^3)$. By using the more sophisticated method of [10] the least square estimation of the edge lengths of a given tree and distance matrix can be done in $O(n^2)$.
After each stage of adding a reticulation edge we perform a stage of greedily adjusting the boundaries of the families (by increasing or decreasing the boundaries of each subsequences in each family) while improving the ME criteria. Since after each such stage the ME criteria is improved, a convergence to a local optima is guaranteed. The time complexity of this stage is $(h^2 \cdot f \cdot n) \cdot n^2$.
Total Time Complexity Suppose the input includes n sequences of length ℓ, and the result includes h families each with f reticulation edges. The total runtime complexity of our method is $\binom{n}{2} \cdot \ell^2 + \ell \cdot n^2 \cdot \binom{n}{2} + (h^2 \cdot f \cdot n) \cdot n^2 = O(n^2 \cdot (\ell^2 + n \cdot h^2 \cdot f + n^2 \cdot \ell))$.

5 Experimental Results

For evaluating our methods we performed three tests. First, we implemented our method on two biological dataset (bacterial *rbcL* proteins, and the plants' gene

rps11) that underwent horizontal gene transfer. In the second test we simulated evolution that included HGT/recombination, rearrangements, and local point mutations, our method were used for reconstructing the simulated evolution. Finally, we implementation of our method on two datasets of viruses' genomes.

5.1 Biological Inputs: Proteins and Genes

Proteins of Bacteria The first input includes the rubisco gene *rbcL* of a group of 14 plastids, cyanobacteria, and proteobacteria, which were first analyzed by Delwiche and Palmer [6] (they and other suggest that it includes HGTs). This dataset consists of amino acid sequences, part of them are from Form I of rubisco, and the other six are from Form II of rubisco. We used exactly the same sequences that Delwiche and Palmer used in their paper. The species tree was based on information from the ribosomal database project (http://rdp.life.uiuc.edu) and the work of [6]. We checked two distance matrices, $PAM250$ and $Blosum62$, both with gap penalty -8. Since this dataset includes a set of proteins we constrained the families to be ordered (the NRGT problem). We checked various sizes of L and got similar results (due to lack of space the more details about the results are deferred to the full version of this paper). We got similar results for the two distance matrices; this indicates that our method is robust to changes in the distance matrix. In general, our results support previous results that analyzed this dataset [6, 4, 15, 14]. For example, we and previous methods discovered reticulation edge between α and β *proteobacteria*, and reticulation edge between *proteobacteria* and the plastid.

Genes of Plants The second database includes the ribosomal protein gene *rps11* of a group of 47 flowering plants, which was first analyzed by Bergthorsson *et al.* [3] (they and others suggest that this dataset includes partial HGT). The species tree was reconstructed based on various sources, including the work of [20] and [16]. We used exactly the same sequences that Bergthorsson *et al.* used in their paper. Due to lack of space more details about the results are deferred to the full version of the paper. By Bergthorsson *et al.* these species underwent *chimeric* HGT (*e. g.* partial HGT), this conjecture is supported by our results which relate all the HGT to the family in positions 150 trough 300. In general, our HGTs suggest transfer of genetic material between *Liliopsida* and *Dipsacales*, *Liliopsida* and *Papaveraceae*, and *Ranunculales* and *Dipsacales*. The first two HGTs are similar to HGTs reported in previous works (for example, in [15]), while the third is new and suggests further biological research.

Simulated Data: Simulating HGT/Recombination, Rearrangements, and Local Point Mutations Here we evaluate the accuracy of our method on simulated data. The data consists of sequences which have evolved through substitutions, insertions, deletions, and lateral transfers. We generated 20 data sets with 10 leaves and 20 data sets with 20 leaves using the following recipe (see figure 3). (1) The *species tree* was generated using a regular birth death process from the Beep software package [2]. These trees are ultra metric with a root to leaf distance of 1. (2) Three *transfer trees* were independently created from the species tree by applying two random lateral transfers. Each transfer event was chosen to occur at time $t \in [0, 1]$ with probability $P(t) = \frac{\#\text{ concurrent lineages at time } t}{\int_0^1 P(r)}$, i.e, the probability increases linearly

with the number of concurrent lineages. Once t was selected the transfer was selected uniformly at random from the possible transfer events at time t. (3)*Species sequences*, the sequences which have evolved according to the species tree, of expected length 4000 (similar to the typical length of genome virus, which is is few kbp) were generated using the ROSE [24] software package. Each nucleotide evolved according to the Jukes-Cantor model with a substitution probability of 0.2 from the root down to any leaf. Moreover, in each nucleotide insertions and deletions of up to 7 nucleotides[5] occurred with probability 0.01 from the root down to any leaf. (4) *Transfer blocks*, the sequences which have evolved according to the transfer trees, of expected length 500 were generated using the same process as for the species sequences (The typical length of a gene is few hundreds nucleotides, usually complete gene are horizontally transferred [6]. In the case of partial HGT or recombination, the lengths are in the order of magnitude of at least half a gene [17, 3], *i.e. few hundreds bp*. Thus, we transfers blocks with similar size.). (5) The *combined sequences*, the sequences containing both the species sequences and the transfer blocks, where created by inserting the transfer blocks uniformly at random into the species sequences such that no evolutionary block in the sequences was shorter than 500.

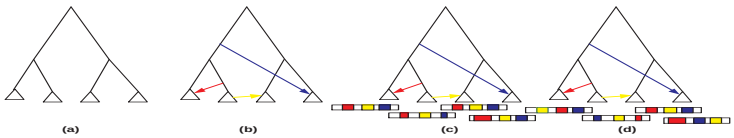


Fig. 3. Illustration of our simulation. We generated synthetic data by the following steps. (a) Generated a random tree. (b) Add three random reticulation edges to the tree. (c) Evolve sequences along the trees, most of the positions evolve according the organismal tree, three blocks were evolve according to the organismal tree plus subset of the reticulation edges. (d) Randomly rearrange the blocks in each of the leaves.

We ran our algorithm with $380 < L < 600$, and with $W = 15$ (the results for $12 \leq W \leq 18$ where similar). For each of the 20 dataset of each size, there are 3 blocks of length about 500 that were transferred (while the rest of the sequences evolve in the original tree). Thus, there were 7 families for each dataset with a total of 140 families (for both the 10 and 20 leaf test sizes). Moreover, each family had been affected by two HGT events. Thus, there was a total of 120 HGT events (for both the 10 and 20 leaf test sizes).

The results were similar for the two datasets, while the results for the 20 leaves datasets were a bit better. Due to lack of space we describe only the results of the 10 leaf datasets, while the results for the 20 leaf datasets are deferred to the full version of the paper: Out of the 140 families our algorithm did not completely miss any family. Only four families were shifted; three by 300 positions and one by 200 positions. Our algorithm identified 93 of the total 120 HGT events. Four of the edges were identified but with reversed direction. Only 23 edges were different than the original edges. However, in this case the edges our algorithm found were very close to the original edges.

[5] The standard insertion and deletion functions in ROSE were used.

According to our results the accuracy of the algorithm improves when the number of leafs increases. One important goal of the method is its ability to infer the right number of HGT events. According to the results, our method performed very well in achieving this goal. Usually after adding the correct number of reticulation edges the improvement in the score is negligible. This is a major advantage compared to methods such as MP or ML when sites are independent [15, 14], where usually there is less clearer change in the slope of the score graph.
Genome of viruses Our last datasets include complete genomes of two RNA viruses, one of HIV the other of Hepatitis C. We checked our method on these two typical inputs. The genomes were downloaded from [19], and each dataset included 10 genomes. We used our method to check if the datasets include HGTs/recombination and/or rearrangement. For the HIV dataset, our method did not find HGT events nor did it find rearrangement events. In the case of Hepatitis C we found two possible reticulation edges that may suggest an ancient recombination or horizontal gene transfer events. Due to lack of space more details about the virus datasets and results are deferred to the full version of the paper.

6 Concluding Remarks and Further Research

In general, genomic material evolves through local point mutations (insertion, deletion, substitution), genome rearrangements, horizontal gene transfers, recombinations, duplications, and deletions. This work is a step towards developing a method for inferring evolution under all these types of operations, and it is mainly a proof of concept. We showed that our method, which is based on the ME criterion, is useful for inferring partial or complete HGT events, and can infer rearrangements together with HGTs or recombinations.

One work on this new topic is clearly not enough for solving all the problems. Further research in this direction will include: extending the set of operations to include duplications, deletions, and inversions; developing a more sophisticated simulator of virus evolution; investigating the hardness of $NRGT$ (in this work we proved the hardness RGT and RNT); and improving the running time of our heuristic. are currently aimed at using our approach for exploring the evolution of various groups of viruses and bacteria.

Acknowledgment

We thank Prof. Benny Chor for helpful discussions. T.T. was supported by the Edmond J. Safra Bioinformatics program at Tel Aviv University.

References

1. L. Addario-Berry, M. Hallett, and J. Lagergren. Towards identifying lateral gene transfer events. In *PSB03*, pages 279–290, 2003.
2. L. Arvestad, A. Berglund, J. Lagergren, and B. Sennblad. Beep software package, 2006.

3. U. Bergthorsson, K. Adams, B. Thomason, and J. Palmer. Widespread horizontal transfer of mitochondrial genes in flowering plants. *Nature*, 424:197–201, 2003.
4. A. Boc and V. Makarenkov. New efficient algorithm for detection of horizontal gene transfer events. *WABI*, pages 190–201, 2003.
5. S. Burkhardt and J. Krkkinen. Fast lightweight suffix array construction and checking. *In Proc. 14th Symposium on Combinatorial Pattern Matching (CPM03)*, pages 55–69, 2003.
6. C. Delwiche and J. Palmer. Rampant horizontal transfer and duplicaion of rubisco genes in eubacteria and plastids. *Mol. Biol. Evol*, 13(6), 1996.
7. R. Desper and O. Gascuel. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *J. Comp. Biol.*, 9(5):687–705, 2002.
8. W.F. Doolittle, Y. Boucher, C.L. Nesbo, C.J. Douady, J.O. Andersson, and A.J. Roger. How big is the iceberg of which organellar genes in nuclear genomes are but the tip? *Phil. Trans. R. Soc. Lond. B. Biol. Sci.*, 358:39–57, 2003.
9. I. Elias. Settling the intractability of multiple alignment. In *ISAAC*, pages 352–363, 2003.
10. O. Gascuel. Concerning the NJ algorithm and its unweighted version UNJ, 1997.
11. A. Goldstein, P. Kolman, and J. Zheng. Minimum common string partition problem: Hardness and approximations. In *Algorithms and Computation, 15th International Symposium, ISAAC 2004*, volume 3341 of *Lecture Notes in Computer Science*, pages 484–495. Springer, 2004.
12. J. Hein. A heuristic method to reconstruct the history of sequences subject to recombination. *J. Mol. Evol.*, 36:396–405, 1993.
13. D. H. Huson and D. Bryant. Application of phylogenetic networks in evolutionary studies. *Mol. Biol. Evol*, 23(2):254–267, 2006.
14. G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Maximum likelihood of phylogenetic networks. *Bioinformatics*, 22(21):2604–2611, 2006.
15. G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Efficient parsimony-based methods for phylogenetic network reconstruction. *Bioinformatics*, 23(2):123–128, 2007.
16. W.S. Judd and R.G. Olmstead. A survey of tricolpate (eudicot) phylogenetic relationships. *American Journal of Botany*, 91:1627–1644, 2004.
17. O. Kalinina, H. Norder, and L. O. Magnius. Full-length open reading frame of a recombinant hepatitis c virus strain from St Petersburg: proposed mechanism for its formation. *J. Gen. Virol.*, 85:1853–1857, 2004.
18. K.K. Kidd and L.A. Sgaramella-Zonta. Phylogenetic analysis: concepts and methods. *Am. J. Hum. Genet.*, 23(3):235–252, 1971.
19. C. Kuiken, K. Yusim, L. Boykin, and R. Richardson. The los alamos hcv sequence database. *Bioinformatics*, 21(3):379–84, 2005.
20. F.A. Michelangeli, J.I. Davis, and D.Wm. Stevenson. Phylogenetic relationships among Poaceae and related families as inferred from morphology, inversions in the plastid genome, and sequence data from mitochondrial and plastid genomes. *American Journal of Botany*, 90:93–106, 2003.
21. A. Rzhetsky and M. Nei. Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Mol. Biol. Evol.*, 10:1073–1095, 1993.
22. N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol.*, 4(4):406–25, 1987.
23. J. Sinkovics, J. Horvath, and A. Horak. The origin and evolution of viruses (a review). *Acta Microbiol Immunol Hung.*, 45(3-4):349–390, 1998.
24. J. Stoye, D. Evers, and F. Meyer. Rose: generating sequence families. *Bioinformatics*, 14:157–163, 1998.
25. K. Strimmer and V. Moulton. Likelihood analysis of phylogenetic networks using directed graphical models. *Mol. Biol. Evol.*, 17(6):875–881, 2000.
26. I. Ulitsky, D. Burstein, T. Tuller, and B. Chor. The average common substring approach to phylogenomic reconstruction. *J. Comp. Biol.*, 13(2):336–350, 2006.