Parsimony Score of Phylogenetic Networks: Hardness Results and a Linear-Time Heuristic

Guohua Jin, Luay Nakhleh, Sagi Snir, and Tamir Tuller

Abstract—Phylogenies—the evolutionary histories of groups of organisms—play a major role in representing the interrelationships among biological entities. Many methods for reconstructing and studying such phylogenies have been proposed, almost all of which assume that the underlying history of a given set of species can be represented by a binary tree. Although many biological processes can be effectively modeled and summarized in this fashion, others cannot: recombination, hybrid speciation, and horizontal gene transfer result in *networks* of relationships rather than trees of relationships. In previous works, we formulated a maximum parsimony (MP) criterion for reconstructing and evaluating phylogenetic networks, and demonstrated its quality on biological as well as synthetic data sets. In this paper, we provide further theoretical results as well as a very fast heuristic algorithm for the MP criterion of phylogenetic networks. In particular, we provide a novel combinatorial definition of phylogenetic networks in terms of "forbidden cycles," and provide detailed hardness and hardness of approximation proofs for the "small" MP problem. We demonstrate the performance of our heuristic in terms of time and accuracy on both biological and synthetic data sets. Finally, we explain the difference between our model and a similar one formulated by Nguyen et al., and describe the implications of this difference on the hardness and approximation results.

Index Terms—Maximum parsimony, phylogenetic networks, horizontal gene transfer, hardness and approximation.

1 Introduction

Phylogenetic networks are a special class of *directed acyclic graphs* (DAGs) that model evolutionary histories when trees are inadequate, such as in the cases of horizontal gene transfer (HGT) and hybrid speciation [24], [29], [26]. Fig. 1a illustrates a phylogenetic network on four species with a single HGT event. In an evolutionary scenario involving horizontal transfer, an organism transfers genetic material to another organism that is not its offspring (i.e., genetic material is transferred from one lineage to another), as in Fig. 1a. In such a case, the origin of certain sites in a DNA sequence may be nonparental (as in Fig. 1c), while all others are inherited from the parent (as in Fig. 1b). Thus, *each site evolves down one of the trees induced by (or contained in) the network*. Similar scenarios arise in the cases of other reticulate evolution events (such as hybrid speciation and interspecific recombination).

Hybrid speciation is a significant evolutionary mechanism in plants, fish, and other groups of species [25], and HGT is believed to be ubiquitous among prokaryotic organisms [6], [7], [23], [22], [4], [17]. Therefore, in order to reconstruct and analyze evolutionary histories of these groups of species, as well as to reconstruct the prokaryotic branch of the Tree of Life, developing accurate criteria for reconstructing and evaluating phylogenetic networks and efficient algorithms for inference based on these criteria is imperative. A large number of publications have been introduced in recent years about various aspects of phylogenetic networks; see [10], [12], [29], [32], [11], [15], [16], [1], [31] for some of the papers introduced in the last three years, and [24], [26] for detailed surveys.

In this work, we address the *maximum parsimony* (MP) of phylogenetic networks. Maximum parsimony is one of the most commonly used and extensively studied criteria for phylogenetic tree inference. Roughly speaking, inference based on this criterion seeks the tree that explains the evolution of a set of sequences with the minimum number of mutations.

In 1990, Jotun Hein proposed using this criterion for inferring the evolution of sequences subject to recombination. Recently, Nakhleh et al. formulated the parsimony criterion for evaluating and inferring general phylogenetic networks [31]. In particular, they formulated two problems based on the MP criterion: the "small" parsimony problem, **PSPN**, which seeks the parsimony score of a fixed phylogenetic network leaf labeled by a set of sequences, and the "big" parsimony problem, **FTMPPN**, which seeks an augmentation of a fixed tree into a network so as to optimize the parsimony score up to a certain threshold.[1] In two recent articles, we demonstrated the quality of the criterion for evaluating phylogenetic networks as well as the appropriateness of the solutions to these two problems for reconstructing phylogenetic networks [18], [20].

In [18], we conjectured the **PSPN** problem to be NP-hard. Recently, Nguyen et al. [33] provided a hardness result for a closely related version of the **PSPN** problem and claimed that the problem cannot be approximated within a factor of

- *G. Jin and L. Nakhleh are with the Department of Computer Science, Rice University, Houston, TX 77005. E-mail: {jin, nakhleh}@cs.rice.edu.*
- *S. Snir is with the Institute of Evolution, University of Haifa, Haifa 31905, Israel. E-mail: ssagi@research.haifa.ac.il.*
- *T. Tuller is with the School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: tamirtul@post.tau.ac.il.*

Manuscript received 8 Sept. 2008; accepted 12 Oct. 2008; published online 20 Oct. 2008.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2008-09-0162.
Digital Object Identifier no. 10.1109/TCBB.2008.119.

1. PSPN stands for Parsimony Score of Phylogenetic Networks and FTMPPN stands for Fixed Tree Maximum Parsimony of Phylogenetic Networks.

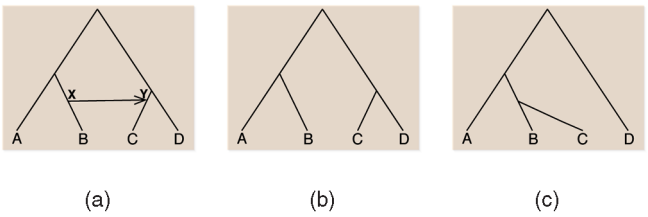


Fig. 1. (a) A phylogenetic network with a single HGT event from X to Y. (b) The underlying organismal (species) tree. (c) The tree of a horizontally transferred gene.

$\log n$, where n is the number of taxa (leaves) in the network. This means that there does not exist a constant c for which a polynomial algorithm can give a c-approximation. Moreover, they claimed that the problem was NP-hard even for *galled networks*, which are a special class of phylogenetic networks in which cycles in the underlying undirected graph of the network are node disjoint [38].

In this paper, we first provide a formal definition of a phylogenetic network that was previously formulated in [29]. We present complete hardness proofs that we had sketched in [18], and extend it to networks of bounded degree. This is essential for establishing the hardness of approximation of the **PSPN** problem. We explain the differences between our results and recent results of Nguyen et al. [33]. We show that while Nguyen et al. did address the small parsimony problem, the phylogenetic networks that their reduction produces do not satisfy the required temporal constraints. On the other hand, our reduction does satisfy these constraints and thus gives different hardness of approximation results. Our reduction implies that the problem is not approximable within any constant. In the conference version of this paper, we presented a 3-approximation algorithm. Unfortunately, the algorithm is erroneous and its 3-approximation is not guaranteed.

As almost every computational task in biology turns to be NP-hard, heuristics play a central role in computational biology. Devising fast and accurate heuristics requires deep insight into the problem under investigation. The central part of this work is a heuristic algorithm for the **PSPN** problem. In Section 4, we devise a very fast heuristic algorithm for the problem and demonstrate its strength on synthetic as well as real biological data. Its high speed and very high accuracy with respect to the exact exhaustive algorithm of [31] make it more practical for analyzing large microbial data sets where HGT is very common.

Finally, we show that the reduction of [33] in the case of galled networks is for a different version of the **PSPN** problem, hence clarifying the seemingly contradictory result to an algorithm that was sketched in [30] for the problem.

2 Parsimony of Phylogenetic Networks

2.1 Preliminaries and Definitions

Let $T = (V, E)$ be a tree, where V and E are the *tree nodes* and *tree edges*, respectively, and let $\mathcal{L}(T)$ denote its leaf set. Further, let $\mathcal{X}$ be a set of taxa (species). Then, T is a phylogenetic tree over $\mathcal{X}$ if there is a bijection between $\mathcal{X}$ and $\mathcal{L}(T)$. Henceforth, we will identify the taxon set with the leaves they are mapped to, and let $[n] = \{1, \ldots, n\}$ denote the set of leaf labels. A tree T is said to be *rooted* if there is a single distinguished internal vertex r with in-degree 0 and all the edges are directed away from it. In this work, we deal with rooted trees.

We denote by T_v the subtree rooted at v. A function $\lambda : [n] \to \{0, 1, \ldots, |\Sigma| - 1\}$ is called a *state assignment function* over the alphabet Σ for T. We say that function $\hat{\lambda} : V(T) \to \{0, 1, \ldots, |\Sigma| - 1\}$ is an extension of λ on T if it agrees with λ on the leaves of T. Let k denote the sequences' length. In a similar way, we define a function $\lambda^k : [n] \longmapsto \{0, 1, \ldots, |\Sigma| - 1\}^k$ and an extension $\hat{\lambda}^k : V(T) \longmapsto \{0, 1, \ldots, |\Sigma| - 1\}^k$. The latter function is called a *labeling* of T. We write $\hat{\lambda}^k(v) = s$ to denote that sequence s is the label of the vertex v. The ith *site* is an n-tuple where the jth coordinate is the ith state of species (leaf) j.

A fully labeled tree is a tree in which all its nodes have labels from $\{0, 1, \ldots, |\Sigma| - 1\}$. Given a labeling $\hat{\lambda}^k$, let $d_e(\hat{\lambda}^k)$ denote the Hamming distance between the two sequences labeling the two endpoints of the edge $e \in E(T)$. A phylogenetic network $N = N(T) = (V, E \cup H)$ over the taxon set $\mathcal{X}$ is derived from $T = (V, E)$ by adding a set H of directed edges to T, such that each edge $h \in H$ connects two existing nodes in T. Therefore, the set of nodes in N is same as in T and the edge set E is augmented with the set H. For $H = \emptyset$, N is a tree; otherwise (i.e., $H \neq \emptyset$) we say that N is *proper*. From now on, we will refer to proper networks solely.

In the reverse direction, a network gives rise to a set of trees. Each such tree is obtained by the following two steps: 1) for each node of in-degree greater than one, remove all but one of the incoming edges and then 2) suppress all nodes with out-degree one. We denote by $T(N)$ the set of all trees contained inside network N. For a network N and a node $v \in V(N)$, N_v denotes the graph induced by the nodes reachable from v.

Finally, phylogenetic networks must satisfy additional temporal constraints, as described in [29]. First, N must be acyclic. Second, N should respect the time flow property, which we now elaborate on. Since at the scale of evolution, HGT events are instantaneous in time, a reticulation edge between two nodes x and y dictates that the organisms represented by x and y must have coexisted in time. Therefore, having a reticulation edge between x and y serves as a "synchronization point": no pair of nodes u and v, where one precedes x and y and the other succeeds them, can be the endpoints of a reticulation edge.

Fig. 2 illustrates a directed acyclic graph: lines correspond to tree edges, which are directed away from the root, whereas arrows correspond to reticulation edges. While acyclic, this graph does not satisfy the time flow property as it implies that y precedes x, and at the same time that x precedes y—an impossible scenario.

We now give a simpler formal definition of the time flow property given in [29, Definition 3]: Let $\mathcal{A}$ be the ancestry relation in a tree T. This is the ordered pair $\langle u, v \rangle \in \mathcal{A}$ if there is a (directed) path $u \rightsquigarrow v$ in T. It is easy to see that $\mathcal{A}$ is an asymmetric and transitive relation.[2] A network $N(T)$ extends $\mathcal{A}$ as follows: Let $e = (x \to y) \in H$ be a reticulation edge from x to y. Then for every w, $\langle w, y \rangle \in \mathcal{A} \Leftrightarrow \langle w, x \rangle \in \mathcal{A}$. Note that the direction of e is irrelevant in this context (the

2. It is common to treat this relation as partial order; however, in our case, reflexivity is unnecessary.

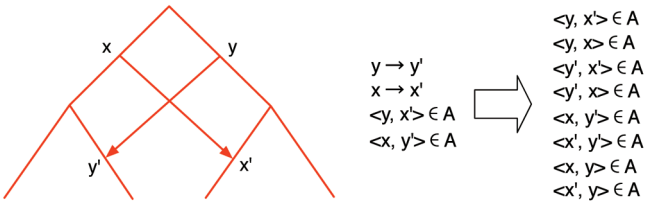


Fig. 2. An example of a phylogenetic network that is a DAG, yet does not satisfy the time flow property. Since reticulation is instantaneous at the scale of evolution, the network implies that x occurs before y and y before x—an impossible scenario.

extension of $\mathcal{A}$). Finally, we augment $\mathcal{A}$ with the transitive closure induced by the newly added elements (i.e., $\langle u,y\rangle \in \mathcal{A} \wedge \langle x,v\rangle \in \mathcal{A} \wedge e = (x \rightarrow y) \Rightarrow \langle u,v\rangle \in \mathcal{A}$).

Definition 1. *A network $N(T)$ is valid (or satisfies the time flow property) if $\mathcal{A}$ is asymmetric.*

In particular, there is no HGT edge between a node and its descendant. We need two more definitions to show some combinatorial properties of phylogenetic networks that will be required later.

Definition 2. *An undirected path P in $N(T)$ is HGT-neutral if tree edges in P are traversed according to their direction and HGT edges arbitrarily.*

As a special case, an *HGT-neutral* path is an *HGT-neutral* cycle if the first node in the path is identical to the last node.

Definition 3. *An HGT-neutral cycle C in $N(T)$ is forbidden if it contains at least one tree edge.*

For example, the cycle x, y', y, x' in Fig. 2 is forbidden since tree edges in the paths $x \rightsquigarrow y'$ and $y \rightsquigarrow x'$ are traversed while considering their direction and reticulation edges $(x \rightarrow x')$ and $(y \rightarrow y')$—arbitrarily.

Observation 1. A network is valid iff it contains no forbidden cycles.

Proof. $\Rightarrow$ We need to show that a network is valid if it contains no forbidden cycles. Assume that the network is not valid, that is, $\mathcal{A}$ is not asymmetric. This implies that there are $x, y \in V$ s.t. $\langle x,y\rangle \in \mathcal{A}$ and $\langle y,x\rangle \in \mathcal{A}$. By the construction, there are HGT-neutral paths from x to y and from y to x. We are left to show that the cycle contains at least one tree edge. Note from the construction that nodes across an HGT edge are unrelated, that is, they are not added to $\mathcal{A}$ as a result of the HGT edge. Therefore, $\langle x,y\rangle \in \mathcal{A}$ implies that there is a path between x and y with at least one tree edge between them. This implies that there must be at least one tree edge in the cycle.

$\Leftarrow$ We need to show that if a network is valid, it contains no forbidden cycles. Assume that there is a forbidden cycle C in $N(T)$. Let $(u \rightarrow v)$ be a tree edge in C. Then $\langle u,v\rangle \in \mathcal{A}$, and since the transitive closure extends the relation to all nodes along an HGT-neutral path, we obtain that $\langle v,u\rangle \in \mathcal{A}$. □

2.2 Maximum Parsimony of Phylogenetic Networks

We begin by reviewing the maximum parsimony criterion for phylogenetic trees.

Problem 1. *Parsimony Score of Phylogenetic Trees (PSPT)*

Input. *A 3-tuple (S,T,λ^k), where T is a phylogenetic tree and λ^k is the labeling of $\mathcal{L}(T)$ by the sequences in S.*

Output. *The extension $\hat{\lambda}^k$ that minimizes the expression $\sum_{e\in E(T)} d_e(\hat{\lambda}^k)$.*

We define the parsimony score for (S,T,λ^k), $pars(S,T,\lambda^k)$, as the value of this sum, and $pars(S,T,\lambda^k,i)$ as the value of this sum for site i. So, $pars(S,T,\lambda^k) = \sum_{1\le i\le k} pars(S,T,\lambda^k,i)$. It is easy to see that optimal value is obtained by optimal solutions for every site $1 \le i \le k$. Polynomial time algorithms, due to Fitch and Sankoff, solve PSPT, as well as its weighted version (sites and substitutions have weights), in polynomial time [8], [37].

Since Fitch's algorithm is a basic building block in this paper, we hereby describe it. As mentioned above, the input to the problem is a tree T, a single site C, and its state assignment λ^1. The algorithm returns the tree T with its optimal extension $\hat{\lambda}^1$ in two phases:

1. **Bottom-up phase.** For every node v in the tree, the algorithm computes $A(v)$, the set of states from which the optimal assignment of states to site C at node v is obtained. For a leaf node v, $A(v) = \{\sigma\}$, where $\sigma = \lambda^1(v)$. For a node v whose children are v_1 and v_2, $A(v)$ is computed as
$$A(v) = \begin{cases} A(v_1) \cap A(v_2), & \text{if } A(v_1) \cap A(v_2) \neq \emptyset, \\ A(v_1) \cup A(v_2), & \text{otherwise.} \end{cases}$$
2. **Top-down phase.** For every node v in the tree, the algorithm computes $\hat{\lambda}^1(v)$, which is the optimal assignment of states to site C of all nodes in T. For the root r, $\hat{\lambda}^1(r) = \sigma$, where σ is an arbitrary element of $A(r)$. For a node v whose parent is u, $\hat{\lambda}^1(v)$ is computed as
$$\hat{\lambda}^1(v) = \begin{cases} \sigma \in A(v) \cap \hat{\lambda}^1(u), & \text{if } A(v) \cap \hat{\lambda}^1(u) \neq \emptyset, \\ \sigma \in A(v), & \text{otherwise.} \end{cases}$$

As described here, the algorithm applies only to binary trees. Nonetheless, a straightforward extension to arbitrary k-degree trees can be easily achieved.

The extension of Problem 1 (PSPT) to phylogenetic networks is as follows:

Definition 4. *Parsimony Score of Phylogenetic Networks* (**PSPN**).

Input. *A 3-tuple (S,N,λ^k), where N is a phylogenetic network and λ^k is the labeling of $\mathcal{L}(N)$ by the sequences in S.*

Output. $pars(S,N,\lambda^k) = \sum_{1\le i\le k}\left[\min_{T\in\mathcal{T}(N)} pars(S,T,\lambda^k,i)\right]$.

$A(u)$ for a node u in the network N is a collection of states (i.e., values from $\{0,1,\ldots,|\Sigma|-1\}$), with the construction so that $A(u)$ is the set of optimal states for N_u if N is a tree.

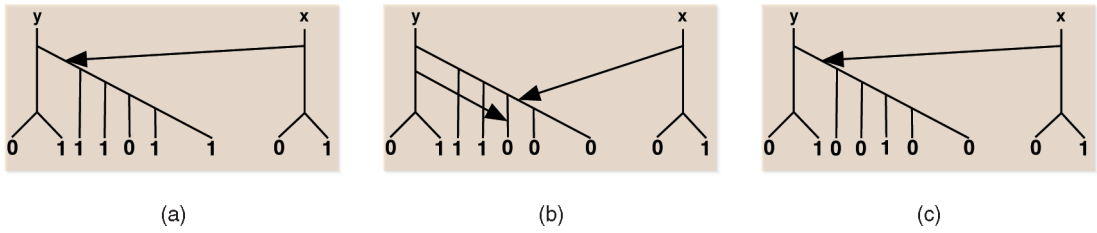


Fig. 3. Part of the reduction from max-2-sat to **PSPN1**. (a) $x \vee y$. (b) $x' \vee y$. (c) $x' \vee y'$.

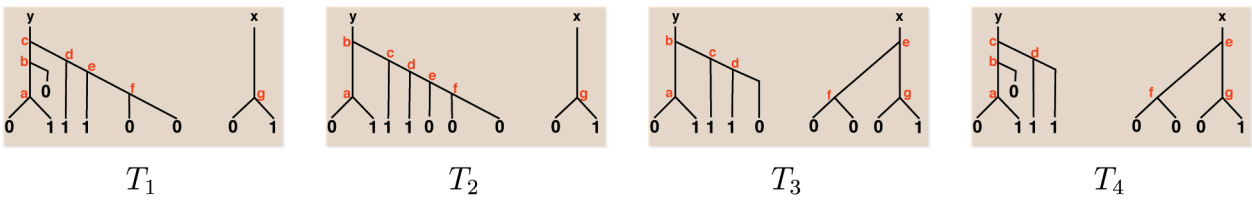


Fig. 4. (Lemma 1) The four possible subforests of the "True-False" network in Fig. 3b.

3 Hardness Results for PSPN

3.1 Hardness of PSPN

In this section, we give a detailed proof of the hardness of **PSPN**, which we had sketched in [19]. Since solving the problem for a given set of sequences entails solving it for every site separately, we formalize the single-site decision version of the problem as follows.

Problem 2. *(PSPN1)*

Input. *A 3-tuple (S, N, λ^1), where N is a phylogenetic network and λ^1 is the labeling of $\mathcal{L}(N)$ by the sequences (each consisting of a single site) in S, and an integer P.*

Question. *Is $pars(S, N, \lambda^1) \leq P$?*

We prove the hardness of the **PSPN1** problem by a reduction from the Maximum 2-Satisfiability (max-2-sat) problem [9], which is formally defined as follows.

Problem 3. *Maximum 2-Satisfiability (max-2-sat)*

Input. *Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 2$, and a positive integer $K \leq |C|$.*

Question. *Is there a truth assignment for U that simultaneously satisfies at least K of the clauses in C?*

We start with a lemma which will be used in our main proof. Let a "True-True" denote a clause that has no negated literals, "True-False" denote a clause that has exactly one negated literal, and "False-False" denote a clause in which both literals are negated. For each of these three types of clauses, we generate subnetworks as shown in Figs. 3a, 3b, and 3c.[3]

Lemma 1. *1) An optimal parsimony score of 3 for a "True-True" network is obtained by labeling $x = 1$ or $y = 1$. Otherwise (i.e., $x = 0 \wedge y = 0$) , the best parsimony score is 4. 2) An optimal parsimony score of 3 for a "True-False" network is obtained by labeling $x = 0$ or $y = 1$. Otherwise, the best parsimony score is 4. 3) An optimal parsimony score of 3 for a "False-False" network is obtained by labeling $x = 0$ or $y = 0$. Otherwise, the best parsimony score is 4.*

3. Note that the subnetwork for the "False-False" case is identical to the "True-True" case but with complementary labeling.

Proof. We provide the full details for case (2) which is the most involved. The proofs for the other cases are similar and hence omitted. Let T_1, T_2, T_3, and T_4 in Fig. 4 be the four subforests of the "True-False" network in Fig. 3b. Let $a, b, \ldots, g$ denote the names of the internal nodes in these trees, as illustrated in the figure. Given the leaf labeling of the four trees, a lower bound on the MP score of each of the trees is 3. Therefore, to establish that the network has an optimal MP score of 3 for a certain labeling, we show that at least one of the four trees attains that score. On the other hand, to establish that the network has an optimal MP score of 4 for a certain labeling, we show that all trees have MP scores higher than 3.

Case 1: $x = 1$ and $y = 1$. If we set $a = b = c = d = g = 1$ and $e = f = 0$, then T_2 has exactly three mutations, and hence, the MP score of the network in this case is 3.

Case 2: $x = 0$ and $y = 1$. If we set $a = b = c = d = 1$ and $e = f = g = 0$, then T_2 has exactly three mutations, and hence, the MP score of the network in this case is 3.

Case 3: $x = 0$ and $y = 0$. If we set $a = b = c = e = f = g = 0$ and $d = 1$, then T_4 has exactly three mutations, and hence, the MP score of the network in this case is 3.

Case 4: $x = 1$ and $y = 0$. A case analysis shows that any labeling to the internal nodes of the four trees results in at least four mutations in each one of them. Hence, the MP score of the network in this case is at least 4. □

We are now in position to prove the main theorem.

Theorem 1. ***PSPN1*** *is NP-hard.*

Proof. Given an input $\langle U, C, K \rangle$ to the max-2-sat problem, we generate the instance to **PSPN1** as follows. We generate a vertex for each variable in U. For each clause in C, we generate a subnetwork and connect it to the variables of the clause (as described in Fig. 3).

A cap (see Fig. 5a) is a subtree that includes three leaves (labeled with 0, 0, and 1) and three internal nodes. One of the internal nodes connects to a variable node, the other two are named *q node* and *j node*.

We connect all the variable nodes as follows. We first connect each variable node to a cap (different cap for each variable node). Then, we connect all the q nodes of

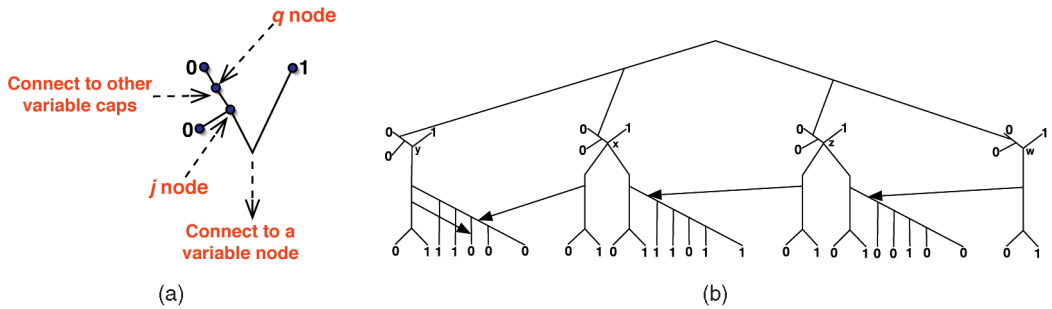


Fig. 5. (a) Part of reduction from max-2-sat to **PSPN1**: the cap of each variable node. (b) Example of a reduction from max-2-sat to **PSPN1**. The input to the max-2-sat problem contains three clauses $y \vee x'$, $x \vee z$, and $z' \vee w'$ on four variables.

these caps and form an arbitrary binary tree where the q *nodes* are its leaves (see Fig. 5a and an example in Fig. 5b); we call this last subtree "caps' subtree." We choose $P = 4 * (|C| - K) + 3 * (K) + |U| = 4 * |C| - K + |U|$.

The resulting network satisfies the time flow constraint since: 1) each clause subnetwork satisfies the time flow constraint and 2) each reticulation edge involves two vertices from the same clause subnetwork (i.e., there are no reticulation edges between two different clause subnetworks).

$\Rightarrow$ Suppose there is a truth assignment for U that simultaneously satisfies at least K of the clauses in C. We choose the labeling of the variable nodes to be their assignment. By Lemma 1, the parsimony score for each of these K clauses' subnetworks (below its variables' nodes) is 3, and the parsimony of the other clauses is at most 4. There must be exactly one mutation between each of the variable nodes and its neighbors which are part of the cap; this increases the parsimony score by $|U|$. By choosing the labeling "0" for each of the nodes in the caps' subtree, we do not increase the parsimony score of the tree. Thus, the total parsimony score will be less than or equal to $4 * |C| - K + |U|$.

$\Leftarrow$ Suppose the parsimony score of the network is less than or equal to $4 * |C| - K + |U|$. By Fitch's algorithm (phase 1), $A(j) =$ "0" or "$\{0, 1\}$"; thus, always $A(q) =$ "0." By Fitch's algorithm, the best assignment to the internal nodes of the caps' subtree assigns zero to all these nodes. Thus, the total number of mutations in caps' subtree is zero.

In any case, there are exactly $|U|$ mutations between the variable nodes and their two neighbor leaves which are part of the cap (the label of one is "0," while the label of the other is "1"), and the contribution to the parsimony score from the clauses' subnetworks is at most $4 * |C| - K$.

By the definition of these subnetworks and by Lemma 1, the labeling of the variables' nodes totally determines the parsimony cost of these subnetworks. By Lemma 1, a clause's subnetwork has parsimony score 3 if and only if the assignment (labeling) to the clause's nodes satisfies the clause, otherwise it has parsimony score 4.

Thus, by choosing the assignment to U being equal to the labeling of these nodes in the network, we will satisfy at least K clauses. □

Lemma 2. *Max-2-sat is NP-hard even for inputs where each variable is restricted to appear at most 12 times.*

Proof. From [34], 3-sat is NP-hard even for a restricted version in which each variable is restricted to appear at most three times. Applying the same reductions from 3-sat to max-2-sat as in [34], with the initial instance to the 3-sat problem being the restricted version, will generate a max-2-sat instance where each variable appears at most 12 times. □

Corollary 1. ***PSPN1*** *is NP-hard even for networks of bounded degrees, where each node has at most 12 children.*

3.2 Hardness of Approximation of PSPN

Using the results of the above reduction, we can now provide a hardness of approximation result for PSPN. The gap version (see [14] for the definition of gap problems) of PSPN1, $gap - PSPN1[Q1, Q2]$ is defined as follows:

Problem 4. $gap - PSPN1[Q1, Q2]$

Input. *A 3-tuple (S, N, λ^1), where N is a phylogenetic network and λ^1 is the labeling of $\mathcal{L}(N)$ by the sequences (each consisting of a single site) in S, and two integers $Q1$ and $Q2$.*

Output. *If $pars(S, N, \lambda^1) \leq Q1$, answer "Yes"; if $pars(S, N, \lambda^1) > Q2$, answer "No"; otherwise, answer either "Yes" or "No."*

Our reduction is from $gap - max - 2sat[P1, P2]$ which is defined as follows:

Problem 5. $gap - max - 2sat[P1, P2]$

Input. *Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 2$, and two positive integers $P1$ and $P2$.*

Output. *If there is a truth assignment for U that simultaneously satisfies at least $P2$ of the clauses in C, then answer "Yes"; if no truth assignment for U simultaneously satisfies more than $P1$ of the clauses in C, then answer "No"; otherwise answer either "Yes" or "No."*

We show that if $gap - max - 2sat[P1, P2]$ is NP-hard, then by our reduction, $gap - PSPN1[4 * |C| - P2 + |U|, 4 * |C| - P1 + |U|]$ is NP-hard.

By [13], there is a constant ζ such that there is no polynomial time algorithm for max-2-sat with performance ratio better than ζ. Thus, there is such a constant also for **PSPN1** also.

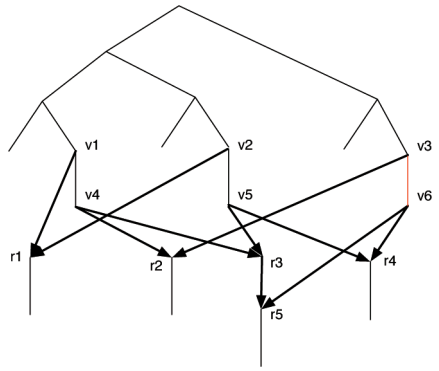


Fig. 6. A reduction from Set Cover to **PSPN**, from the work of Nguyen et al. [33]. Directed edges represent recombination edges. Nodes $r1$, $r2$, $r3$, $r4$, and $r5$ are recombination nodes. Clearly, the network does not satisfy the time flow property.

Corollary 2. *There is a constant ζ' such that there is no polynomial time algorithm for **PSPN1** with performance ratio better than ζ'.*

Corollary 3. ***PSPN1** is NP-hard to approximate even for networks of bounded degrees, where each node has at most 20 children.*

This result follows from the fact that $gap - max - 3sat$, where every variable appears five times, is NP-hard.

We end this section with a note on the reduction to **PSPN1** that Nguyen et al. devised [33]. Nguyen et al. devised a reduction from Set Cover to a problem similar to **PSPN**, and showed that it cannot be approximated with ratio $c \log n$. However, their model does not require the time flow property, and hence, their reduction generates phylogenetic networks that do not satisfy the time flow property (see Fig. 6), while our reduction from $max - 2sat$ does satisfy these constraints (as described above).

4 The Linear Time Heuristic Algorithm

In this section, we describe our heuristic algorithm for the **PSPN1** problem. The general structure of the heuristic is based on the following lemma.

Definition 5. *A reticulation edge $(u \rightarrow v)$ is called a lowest reticulation edge (or, lowest edge) if there is no reticulation edge that is incident with a node in either $T_u \setminus \{u\}$ or $T_v \setminus \{v\}$ (see Fig. 7a).*

We comment that edges comprising a cycle of solely HGT edges (necessarily, since otherwise it is a forbidden cycle by Definition 3) is possible and each such edge can be a lowest edge. In particular, in the case of two reticulation edges $(u \rightarrow v)$ and $(v \rightarrow u)$, if $(u \rightarrow v)$ is a lowest edge, then also $(v \rightarrow u)$ (see Fig. 7b).

Lemma 3. *For every (proper) phylogenetic network, there exists a lowest edge.*

Proof. By Definition 5, it can be seen that a network without a lowest edge contains either a forbidden cycle or an infinite HGT neutral path. □

Observation 2. Let $(u \rightarrow v)$ be a lowest edge. Then both N_u and N_v are trees, and there is no reticulation edge entering both N_u or N_v.

Proof. The observation follows directly from Definition 5 of lowest edges. □

By Lemma 3, there exists a lowest edge $e = (u \rightarrow v)$ in N, and by Observation 2, the in-degree of each node in the subnetworks reachable from both endpoints u and v is one. Therefore, we can compute $A(u)$ and $A(v)$ by Fitch's algorithm.

We denote by a *conflict* a node with in-degree bigger than one. Obviously these conflicts are caused only by reticulation edges. In this section, we describe an algorithm that when given a network as an input, proceeds recursively, aiming at removing conflicts while computing assignments to all the network nodes. Finally, when there are no conflicts, the network is a tree and the parsimony score can be computed easily. Note that removing a single reticulation edge does not necessarily remove a conflict as there can be multiple edges entering the same node. Also note that removing a conflict can be achieved either by removing a reticulation edge or by removing a tree edge. The intuition behind the algorithm is that reticulation edges

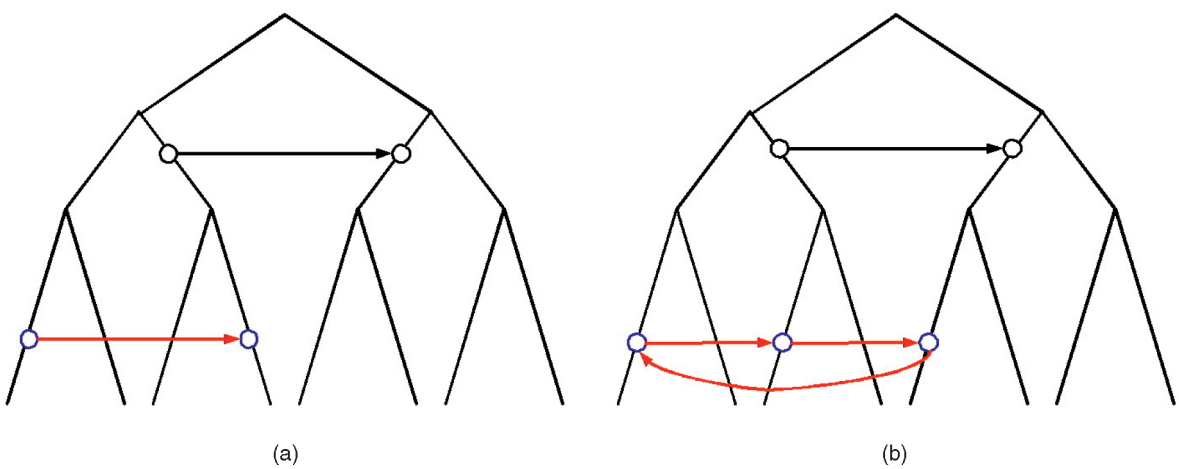


Fig. 7. (a) A lowest edge (the lower left). (b) A set of three lowest edges.

Linear-PSPN$(N = (V, E))$

1. If N is not a tree
   (a) Find a lowest reticulation edge $e = (u \to v)$ in N; let H_v denote the set of reticulation edges entering v.
   (b) Let e' be the edge between v and its ancestral node on the tree edge.
   (c) By Fitch's algorithm compute an optimal bottom-up assignment to all the nodes in N_u and N_v.
   (d) If $A(u) \cap A(v) = \emptyset$ then $\hat{\lambda}_1 =$ Linear-PSPN$(V, E \setminus e)$.
   (e) else $\hat{\lambda}_1 =$ Linear-PSPN$(V, E \setminus (e' \cup H_v))$.
   (f) return $\hat{\lambda}_1$.
2. Continue first phase of Fitch's algorithm on the tree N without changing internal labels that have already determined.
3. Perform second phase of Fitch's algorithm on the tree N.
4. Return the resultant fully labeled tree.

Fig. 8. The Linear-PSPN algorithm.

are assumed to be sparse, compared to the tree edges. This implies that values computed along the tree branches are usually correct. We formulate the following observation.

Observation 3. Let $e = (u \to v)$ be a lowest edge in a network N, and let $A(u)$ and $A(v)$ be the optimal values for the parsimony problem at trees N_u and N_v, respectively. Consider any tree $T \in T(N)$ in which edge e is retained and the tree edge incident into v is deleted. Then if the values at u and v are taken from $A(u)$ and $A(v)$, respectively, we obtain

1. If $A(u) \cap A(v) = \emptyset$, there will be a mutation on e.
2. If $A(u) \subseteq A(v)$, for any $x \in A(u) \cap A(v)$, setting the value at each endpoint u and v to x results in no change on edge e.

Note that Observation 3 is not a proof for the optimality of the heuristic. However, Observation 3 provides the intuition for the good performances of the heuristic algorithm in practice (empirically). If case 1 holds, the reticulation edge e is removed. In all other cases, e is selected and all other edges entering v are removed. The algorithm proceeds recursively until there are no conflicts in the graph. A formal description of the algorithm is given in Fig. 8.

Claim 1. *Let $E(N)$ be the set of edges in N. Then the algorithm Linear-PSPN terminates and runs in time $O(|E(N)|)$.*

Proof. Fitch's algorithm runs in linear time on disjoint subtrees. Additionally, every reticulation edge is considered at most once in the algorithm. □

5 Experimental Results

We have implemented our heuristic algorithm with the following change: We include an additional final step where we modify the internal labeling on the tree topology found by $Linear - PSPN$. This is simply done by running Fitch's algorithm on the resultant tree returned by $Linear - PSPN$, and it can improve the final MP score. As demonstrated in this section, in practice, this usually gives trees with parsimony scores that are very close to the optimal ones (i.e., the ones found by the exact algorithm in [31]).

We evaluated the algorithm on biological as well as synthetic data sets. The experiments were performed on a 2.4 GHz Intel Pentium 4 PC. Accuracy of the heuristic algorithm was measured as the difference of the parsimony scores computed by the heuristic algorithm and the exact algorithm normalized by the parsimony score computed by the exact algorithm, presented as percentage. Execution times of both the heuristic algorithm and the exact algorithm were measured and speedups of the heuristic algorithm over the exact algorithm were reported.

5.1 Synthetic Data Sets

For the simulated data sets, we first used the `r8s` tool [36] to generate a random birth-death phylogenetic tree on 20 taxa. The `r8s` tool generates molecular clock trees; since we wanted to simulate general trees, we multiplied each branch length by a number randomly drawn from an exponential distribution with a rate of 1. The resulting tree was taken as the species tree. The expected evolutionary diameter (longest path between any two leaves in the tree) was 0.2. A model phylogenetic network was generated by adding five HGT edges to the model tree.

Based on the model network, we used the Seq-gen tool [35] to evolve 26 data sets of DNA sequences of length 1,500 down the "species" tree and DNA sequences of length 500 down the tree, contained inside the network, which exhibits all HGT events. Both sequence data sets were evolved under the GTR+Γ+I model of evolution, using the parameter settings of [39]. Finally, we concatenated the two data sets.

5.2 Biological Data Sets

We have included experimental results on four biological data sets, of which three were previously studied [20]. The first data set is the rubisco gene *rbcL* of a group of 46 plastids, cyanobacteria, and proteobacteria, which was analyzed by Delwiche and Palmer [5]. This data set consists of 46 aligned amino acid sequences (each of length 532), 40 of which are from Form I of rubisco and the other 6 are from Form II of

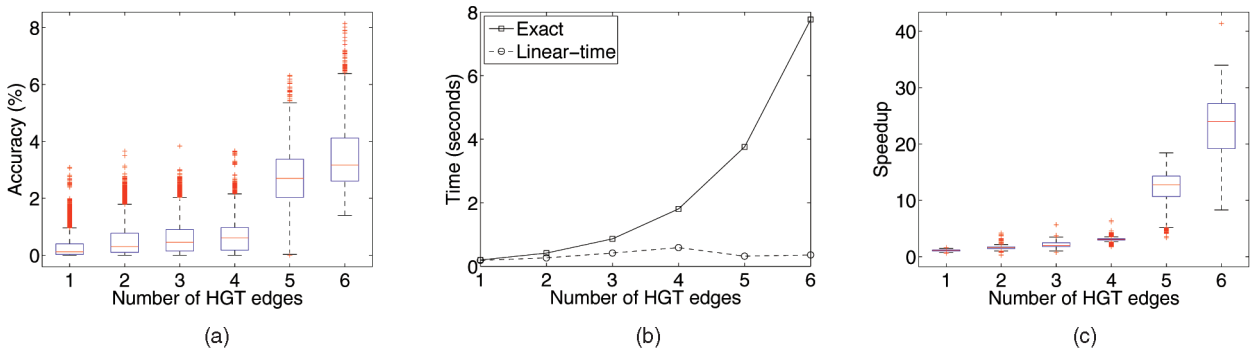


Fig. 9. Results for the simulated data sets. (a) Accuracy computed by $((MP_{linear} - MP_{exact})/MP_{exact})$ shown as percentage. (b) Running times of the exact and heuristic algorithms. (c) Speedup computed as the result of the execution time of the exact algorithm divided by the execution time of the heuristic algorithm.

rubisco. The first 21 and the last 14 sites of the aligned sequences were excluded from the analysis, as recommended by the authors. The species tree for the data set was created based on information from the ribosomal database project (http://rdp.life.uiuc.edu) and the work of [5].

The second data set consists of the ribosomal protein *rpl12e* of a group of 14 Archaeal organisms, which was analyzed by Matte-Tailliez et al. [27]. This data set consists of 14 aligned amino acid sequences, each having 89 sites. The authors constructed the species tree using Maximum Likelihood, once on the concatenation of 57 ribosomal proteins (7,175 sites) and another on the concatenation of SSU and LSU rRNA (3,933 sites). The two trees are identical, except for the resolution of the *Pyrococcus* three-species group; we used the tree based on the ribosomal proteins.

The third data set consists of the ribosomal protein gene *rps11* of a group of 47 flowering plants, which was analyzed by Bergthorsson et al. [2]. This data set consists of 47 aligned DNA sequences, each with 456 sites. The authors analyzed the 3′ end of the sequences separately; this part of the sequences contains 237 sites. The species tree was reconstructed based on various sources, including the work of [28] and [21].

The fourth data set consists of the mitochondrial gene *cox2* of a group of 25 seed and nonseed plants, which was analyzed by Bergthorsson et al. [3]. This data set consists of 28 aligned DNA sequences, including four copies of the *Amborella* gene. Each aligned sequence is 311 bases long. Ten regions including primer sites and editing sites were excluded from the analysis, as suggested by the authors. The authors generated a maximum parsimony tree from which a maximum likelihood tree was built based on estimated parameters. The maximum likelihood tree was further refined until a stable topology was obtained. Seed and nonseed plants were analyzed separately. We used a species tree for the data set based on information at NCBI (http://www.ncbi.nih.gov) and analyzed the entire data set with both seed and nonseed plants together.

5.3 Results and Analysis

We evaluated the performance of the algorithm in terms of accuracy and speedup. Fig. 9 shows the results of the 26 simulated data sets for candidate networks with up to six HGT edges. We added the sixth HGT edge in each of the candidate networks to see the impact of the extra HGT edge on parsimony scores (the decrease in parsimony scores should become much slower after all five HGT edges in the model networks are identified and added). We made sure that the HGT edges do not violate the time constraints. The results were collected from 1,000 sampled valid networks for each case of the multiple gene transfers. HGTs in each network are distributed differently. Fig. 9a shows the accuracy of the heuristic algorithm. Overall, the heuristic algorithm is very accurate with the statistical mean being up to 3 percent difference in the parsimony scores computed, compared with the exact algorithm. All parsimony scores computed by the heuristic algorithm were within 8 percent of the optimal scores. For the networks with less than five HGTs, the heuristic algorithm achieves about the same accuracy of the exact algorithm in most of the networks. Fig. 9b shows averaged execution time in seconds for computing parsimony score of a network using the exact and heuristic algorithms. Speedups of the heuristic algorithm over the exact algorithm are shown in Fig. 9c. The heuristic algorithm is up to 40 times faster than the exact algorithm, with statistical mean of speedups being over 25. The improved execution time of the heuristic algorithm came from the fewer number of trees or subtrees for which parsimony scores are computed. The number of trees or subtrees processed increases as the number of HGTs increases. For each network with six HGTs, the exact algorithm computes parsimony scores of up to 2^6 trees contained in the network.

For the rubisco gene *rbcL* data set, we tested networks with up to eight HGTs. In each case of the multiple gene transfers, we selected 500 valid networks with HGTs being placed differently. Fig. 10a shows the accuracy of parsimony scores computed with the heuristic algorithm. As the results show, the heuristic algorithm is almost as accurate as the exact algorithm with statistical mean of the difference in accuracy being almost 0. Very few outliers exist across different numbers of HGTs. On the other hand, the heuristic algorithm performs very efficiently. It performs up to a factor of 35 faster than the exact algorithm, as shown in Figs. 10b, 10c. The statistical mean of the improvement increases as the number of HGTs increases.

Similar trends are observed with the other three biological data sets, as shown in Figs. 11, 12, 13. Fig. 11a shows that the parsimony scores computed by the heuristic algorithm are less than 4 percent different in statistical mean from the exact algorithm for the *rpl12e* gene data set. Fig. 12a shows that the

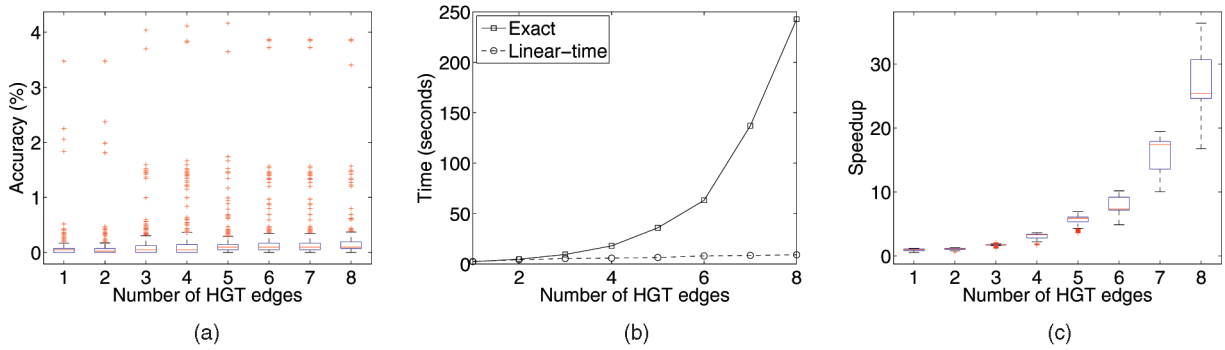


Fig. 10. Results for the *rbcL* gene data set. (a) Accuracy computed by $((MP_{linear} - MP_{exact})/MP_{exact})$ shown as percentage. (b) Running times of the exact and heuristic algorithms. (c) Speedup computed as the result of the execution time of the exact algorithm divided by the execution time of the heuristic algorithm.

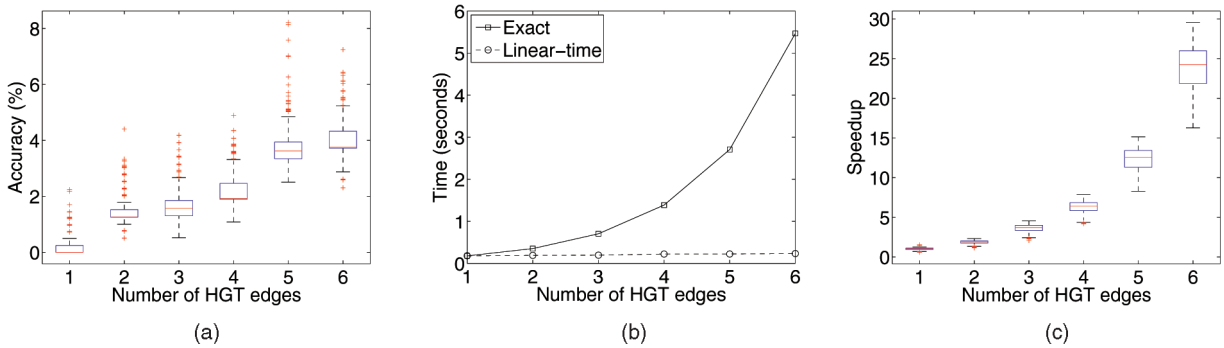


Fig. 11. Results for the *rpl12e* gene data set. (a) Accuracy computed by $((MP_{linear} - MP_{exact})/MP_{exact})$ shown as percentage. (b) Running times of the exact and heuristic algorithms. (c) Speedup computed as the result of the execution time of the exact algorithm divided by the execution time of the heuristic algorithm.

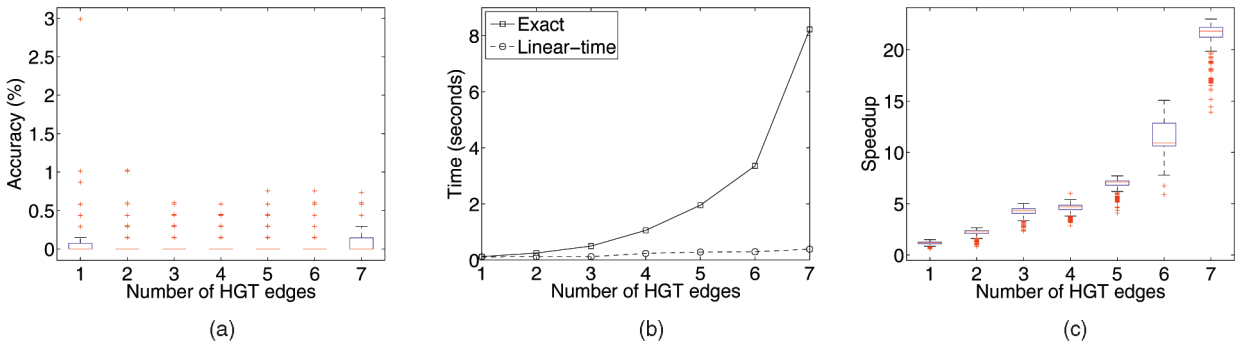


Fig. 12. Results for the *rps11* gene data set. (a) Accuracy computed by $((MP_{linear} - MP_{exact})/MP_{exact})$ shown as percentage. (b) Running times of the exact and heuristic algorithms. (c) Speedup computed as the result of the execution time of the exact algorithm divided by the execution time of the heuristic algorithm.

statistical mean of the difference in accuracy is almost 0 for the *rps11* gene data set, which indicates that the heuristic algorithm computes almost identical scores as the exact algorithm, in most cases. The heuristic algorithm also performs accurately on the *cox2* gene data set with the statistical mean of the difference in accuracy being 0 or 0.5 percent. In all cases except the *cox2* gene, the heuristic algorithm performs significantly faster than the exact algorithm with speedups being up to 30 for the *rpl12e* gene data set and up to 22 for the *rps11* gene data set. The heuristic algorithm performs slower than the exact algorithm in the *cox2* gene case due to the relatively high cost of tree operations performed for each site and the short time of computing the parsimony score of a single tree for short DNA sequences. However, the difference of the execution time decreases as the number of HGTs increases as shown in Figs. 13b, 13c.

We expect that for larger data sets the gains in performance (speedup) will be even more pronounced. If one hopes to detect HGT events in large prokaryotic groups, for example, such a speedup is essential.

5.4 Differences from Recombination Networks

Galled networks are discussed by Nguyen et al. [33], where they show that a version of **PSPN** is NP-hard for galled network. However, their model, recombination networks, is different from ours as they enforce the following constraint on the required solution. Let v denote a node where the two edges that are directed into it are from the nodes w and u

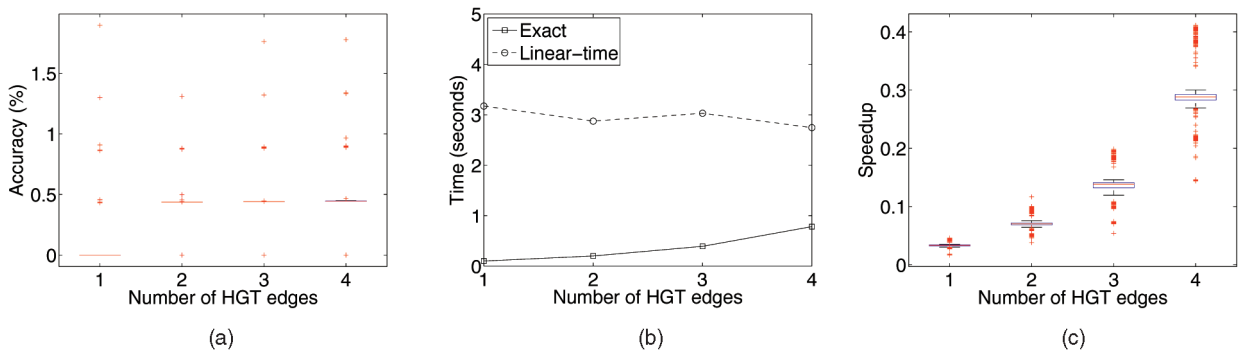


Fig. 13. Results for the *cox2* gene data set. (a) Accuracy computed by $((MP_{linear} - MP_{exact})/MP_{exact})$ shown as percentage. (b) Running times of the exact and heuristic algorithms. (c) Speedup computed as the result of the execution time of the exact algorithm divided by the execution time of the heuristic algorithm.

(the parents of v). Let $s_v = \lambda^k(v)$, $s_w = \lambda^k(w)$, and $s_u = \lambda^k(u)$ denote the labeling of these three nodes. Let s_i^j denote the subsequence of the sequence s that includes positions $i \ldots j$. In the model of Nguyen et al. (a recombination network), the internal labeling of each node, v, must be divided into two continuous nonoverlapping substrings, $(s_v)_1^m$ and $(s_v)_{m+1}^k$, where $1 \leq m \leq k$ and either $((s_v)_1^m = (s_u)_1^m) \wedge ((s_v)_m^k = (s_w)_m^k)$ or $((s_v)_1^m = (s_w)_1^m) \wedge ((s_v)_m^k = (s_u)_m^k)$ (i.e., one of the subsequences s_1^m and s_{m+1}^k is "inherited" from one of its parents and the second subsequence is inherited from the second parent). On the other hand, in our model, each position is independent (as was defined above). Indeed, a simple polynomial algorithm for galled networks under our model was described in [30].

6 Conclusions

In this work, we analyzed the complexity of **PSPN**. We showed that the problem is NP-hard and also hard to approximate within any constant. Next, we developed a very efficient linear-time heuristic for **PSPN**. This algorithm, apart from being very efficient, appears to provide very good results on synthetic as well as real biological data.

There still remain many theoretical problems open. In particular, the "big" parsimony problem, the **FTMPPN**, where the input is a tree and the task is to find the optimal set of HGT edges. We intend to tackle this problem in the future.

Acknowledgments

The authors would like to thank Raphy Yuster for some helpful discussion. This work was supported in part by the Rice Terascale Cluster funded by the US National Science Foundation (NSF) under grant EIA-0216467, Intel, and HP. Luay Nakhleh was supported in part by the US Department of Energy under grant DE-FG02-06ER25734, NSF under grant CCF-0622037, the George R. Brown School of Engineering Roy E. Campbell Faculty Development Award, and the Department of Computer Science at Rice University. Tamir Tuller was supported by the Edmond J. Safra Bioinformatics program at Tel Aviv University.

References

[1] V. Bafna and V. Bansal, "Improved Recombination Lower Bounds for Haplotype Data," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '05)*, pp. 569-584, 2005.

[2] U. Bergthorsson, K.L. Adams, B. Thomason, and J.D. Palmer, "Widespread Horizontal Transfer of Mitochondrial Genes in Flowering Plants," *Nature*, vol. 424, pp. 197-201, 2003.

[3] U. Bergthorsson, A. Richardson, G.J. Young, L. Goertzen, and J.D. Palmer, "Massive Horizontal Transfer of Mitochondrial Genes from Diverse Land Plant Donors to Basal Angiosperm Amborella," *Proc. Nat'l Academy of Sciences USA*, vol. 101, pp. 17747-17752, 2004.

[4] J.R. Brown, "Ancient Horizontal Gene Transfer," *Nat. Rev. Genet.*, vol. 4, pp. 121-132, 2003.

[5] C.F. Delwiche and J.D. Palmer, "Rampant Horizontal Transfer and Duplication of Rubisco Genes in Eubacteria and Plastids," *Mol. Biol. Evol.*, vol. 13, no. 6, pp. 873-882, 1996.

[6] 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.*, vol. 358, pp. 39-57, 2003.

[7] J.A. Eisen, "Assessing Evolutionary Relationships among Microbes from Whole-Genome Analysis," *Curr. Opin. Microbiol*, vol. 3, pp. 475-480, 2000.

[8] W. Fitch, "Toward Defining the Course of Evolution: Minimum Change for a Specified Tree Topology," *Syst. Zool*, vol. 20, pp. 406-416, 1971.

[9] M.R. Garey, R. Garey, and D.S. Johnson, *Computer and Intractability*. Bell Lab, 1979.

[10] P. Gorecki, "Reconciliation Problems for Duplication, Loss and Horizontal Gene Transfer," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '04)*, pp. 316-325, 2004.

[11] D. Gusfield and V. Bansal, "A Fundamental Decomposition Theory for Phylogenetic Networks and Incompatible Characters," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '05)*, pp. 217-232, 2005.

[12] M. Hallett, J. Lagergren, and A. Tofigh, "Simultaneous Identification of Duplications and Lateral Transfers," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '04)*, pp. 347-356, 2004.

[13] J. Hastad, "Some Optimal Inapproximability Results," *Proc. ACM Symp. Theory of Computing (STOC '97)*, pp. 1-10, 1997.

[14] D.S. Hochbaum, *Approximation Algorithms for NP-Hard Problems*. PWS Pub., 1997.

[15] D.H. Huson, T. Klopper, P.J. Lockhart, and M. Steel, "Reconstruction of Reticulate Networks from Gene Trees," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '05)*, pp. 233-249, 2005.

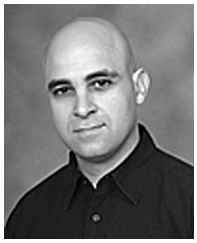
[16] T.N.D. Huynh, J. Jansson, N.B. Nguyen, and W.K. Sung, "Constructing a Smallest Refining Galled Phylogenetic Network," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '05)*, pp. 265-280.

[17] R. Jain, M.C. Rivera, J.E. Moore, and J.A. Lake, "Horizontal Gene Transfer Accelerates Genome Innovation and Evolution," *Mol. Biol. Evol.*, vol. 20, no. 10, pp. 1598-1602, 2003.

[18] G. Jin, L. Nakhleh, S. Snir, and T. Tuller, "Efficient Parsimony-Based Methods for Phylogenetic Network Reconstruction," *Bioinformatics*, vol. 23, pp. e123-e128, 2006.

[19] G. Jin, L. Nakhleh, S. Snir, and T. Tuller, "Maximum Likelihood of Phylogenetic Networks," *Bioinformatics,* vol. 22, no. 21, pp. 2604-2611, 2006.
[20] G. Jin, L. Nakhleh, S. Snir, and T. Tuller, "Inferring Phylogenetic Networks by the Maximum Parsimony Criterion: A Case Study," *Mol. Biol. Evol.,* vol. 24, no. 1, pp. 324-337, 2007.
[21] W.S. Judd and R.G. Olmstead, "A Survey of Tricolpate (Eudicot) Phylogenetic Relationships," *Am. J. Botany,* vol. 91, pp. 1627-1644, 2004.
[22] C.G. Kurland, "Something for Everyone—Horizontal Gene Transfer in Evolution," *Embo Reports,* vol. 1, no. 2, pp. 92-95, 2000.
[23] J.A. Lake, R. Jain, and M.C. Rivera, "Mix and Match in the Tree of Life," *Science,* vol. 283, pp. 2027-2028, 1999.
[24] C.R. Linder, B.M.E. Moret, L. Nakhleh, and T. Warnow, "Network (Reticulate) Evolution: Biology, Models, and Algorithms," *Proc. Pacific Symp. Biocomputing,* tutorial, 2004.
[25] C.R. Linder and L.H. Rieseberg, "Reconstructing Patterns of Reticulate Evolution in Plants," *Am. J. Botany,* vol. 91, pp. 1700-1708, 2004.
[26] V. Makarenkov, D. Kevorkov, and P. Legendre, "Phylogenetic Network Reconstruction Approaches," *Applied Mycology and Biotechnology (Bioinformatics),* vol. 6, 2006.
[27] O. Matte-Tailliez, C. Brochier, P. Forterre, and H. Philippe, "Archaeal Phylogeny Based on Ribosomal Proteins," *Mol. Biol. Evol.,* vol. 19, no. 5, pp. 631-639, 2002.
[28] 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," *Am. J. Botany,* vol. 90, pp. 93-106, 2003.
[29] B.M.E. Moret, L. Nakhleh T. Warnow, C.R. Linder, A. Tholse, A. Padolina, J. Sun, and R. Timme, "Phylogenetic Networks: Modeling, Reconstructibility, and Accuracy," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 1, no. 1, pp. 13-23, 2004.
[30] L. Nakhleh, A. Clement, T. Warnow, C.R. Linder, and B.M.E. Moret, "Quality Measures for Phylogenetic Networks," Technical Report TR-CS-2004-06, University of New Mexico.
[31] L. Nakhleh, G. Jin, F. Zhao, and J. Mellor-Crummey, "Reconstructing Phylogenetic Networks Using Maximum Parsimony," *Proc. Int'l IEEE CS Computational Systems Bioinformatics Conf. (CSB '05),* pp. 440-442, 2005.
[32] L. Nakhleh, T. Warnow, and C.R. Linder, "Reconstructing Reticulate Evolution in Species: Theory and Practice," *Proc. Int'l Conf. Computational Molecular Biology (RECOMB '04),* pp. 337-346, 2004.
[33] C.T. Nguyen, N.B. Nguyen, W.K. Sung, and L. Zhang, "Reconstructing Recombination Network from Sequence Sata: The Small Parsimony Problem," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* 2006.
[34] C.H. Papadimitriou, *Computational Complexity.* Addison-Wesley Pub., 1993.
[35] A. Rambaut and N.C. Grassly, "Seq-Gen: An Application for the Monte Carlo Simulation of DNA Sequence Evolution along Phylogenetic Trees," *Comp. Appl. Biosci.,* vol. 13, 1997.
[36] M. Sanderson, "r8s Software Package," http://loco.ucdavis.edu/r8s/r8s.html, 2008.
[37] D. Sankoff, "Minimal Mutation Trees of Sequences," *SIAM J. Appl. Math.,* vol. 28, pp. 35-42, 1975.
[38] L. Wang, K. Zhang, and L. Zhang, "Perfect Phylogenetic Networks with Recombination," *J. Computational Biology,* vol. 8, no. 1, pp. 69-78, 2001.
[39] D. Zwickl and D. Hillis, "Increased Taxon Sampling Greatly Reduces Phylogenetic Error," *Systematic Biology,* vol. 51, pp. 588-598, 2002.

Guohua Jin received the BSc, MSc, and PhD degrees in computer science from Changsha Institute of Technology in 1984, 1989, and 1993, respectively. He was a visiting assistant professor at the University of Minnesota between August and March 1997. Since April 1997, he has been working in the Computer Science Department at Rice University, where he is a research scientist. His current research interests include high-performance computing and computational biology. He is a member of the ACM.



Luay Nakhleh received the BSc degree in computer science in 1996 from the Technion—Israel Institute of Technology, the master's degree in computer science from Texas A&M University in 1998, and the PhD degree in computer science from The University of Texas at Austin. He is an assistant professor of computer science at Rice University. His research interests fall in the general areas of computational biology and bioinformatics; in particular, he works on computational phylogenetics, comparative genomics, and biological network analysis. He received the Roy E. Campbell Faculty Development Award from Rice University in May 2006 and the DOE Early Career Award in August 2006.

Sagi Snir received the BA degree from Bar Ilan University at Israel majoring in computer science and economics, and the MSc and PhD degrees in computer science from the Technion, Israel. After PhD, he spent two years as a postdoctoral researcher in the Computer Science and Math Departments at the University of California at Berkeley. He is now at the Institute of Evolution at Haifa University, Israel. Before working on the PhD, he worked in various information technologies companies, including IBM Haifa Research Lab. His main research interest is computational biology and, in particular, phylogenetics.

Tamir Tuller received the BSc and MSc degrees in electrical engineering from Tel-Aviv University and the Technion—Israel Institute of Technology, respectively, and the PhD degree in computer science from Tel-Aviv University in 2006. He is a postdoctoral fellow in the School of Computer Science and the Department of Molecular Microbiology and Biotechnology at Tel-Aviv University. He is a fellow of the Edmond J. Safra Bioinformatics Program at Tel-Aviv University and the Yeshaya Horowitz Association through the Center for Complexity Science. His research interests fall in the general areas of computational biology, systems biology, and bioinformatics; in particular, he works on computational phylogenetics, analysis of coevolution, and autoimmunity and bioinformatics of diseases.