

Inferring gene order phylogenies by learning ancestral adjacencies

Dan Adkins¹, Jittat Fakcharoenphol², and Satish Rao¹

¹ University of California, Berkeley, Computer Science Division
Berkeley, CA 94720-1776, USA

² Kasetsart University, Department of Computer Engineering
50 Pahonyothin Rd., Ladyao, Chatujak, Bangkok 10900, Thailand

Abstract. As genomes evolve over very long times, genes get rearranged which changes their order along the genome. The resulting differing orders for various species provide evidence of their phylogenetic relationships, particularly, from long ago. Indeed, this type of data is increasingly useful in sorting out evolutionary relationships [16, 17, 23].

In this paper, we give the first polynomial time algorithm for inferring phylogenies from logarithmic length gene-order data. We provide an implementation of a version of this algorithm which is effective in the high mutation regimes which are difficult for previous polynomial time approaches [17]. Our method takes advantage of reconstructing internal sequences as does the branch and bound approach in GRAPPA [16, 17]. Our polynomial time runtime versus GRAPPA's exponential time is dramatic in practice as well as theory.

The heart of our contribution is a method for estimating distance between genomes and an associated learning method to infer gene-order data at internal nodes. This leads to a polynomial time algorithm for reconstructing a phylogeny on a set of n taxa from gene-order data consisting of $N = O(\log n)$ genes. Our algorithm follows the structure of the algorithm by Mihaescu et al. [15] which applies to character data. We replace the estimators and learning subroutines in the Mihaescu algorithm with methods that work with gene order.

We implement and test a version of our method against the best previous polynomial time method on simulated data. Our method does considerably better in high evolution conditions. For low evolution conditions, we don't do as well as previous methods.

Keywords: phylogeny, gene order, whole genome

1 Introduction

A chromosome or genome can be represented by a signed permutation of genes. This ordering can change due to evolutionary events such as inversions (the flipping of subsequences of genes) or transpositions (the flipping and/or the movement of subsequences of genes). These changes are signals which we can use to reconstruct the evolutionary history, or *phylogeny*, of a set of organisms. Gene order phylogenetic reconstruction is increasingly being used by biologists. See, for example, [16–19].

In this paper, we present a polynomial-time algorithm for reconstructing a phylogeny on a set of n taxa from gene-order data consisting of $N = O(\log n)$ genes. We experiment with an implementation of our method and compare results against previous polynomial time and exponential time methods.

1.1 Background

A *genome* consists of a set of N genes. We assume each genome has the same set of N genes and each gene appears exactly once. A genome is described by a signed circular permutation of the set $[N]$. We call this permutation the *gene order* data. Let G be a genome with signed ordering g_1, g_2, \dots, g_N . Since G is circular, any rotations are equivalent (e.g. g_2, \dots, g_N, g_1). Any signed reversals are also equivalent (e.g. $-g_N, -g_{N-1}, \dots, -g_1$).

In the *Nadeau-Taylor* model of genome evolution, Genomes evolve through a series of evolutionary events. In this paper, we only consider one class of genome rearrangement event: inversions. An *inversion* between indices i and j , for $i \leq j$, reverses the interval between g_i and g_j , resulting in the genome

$$g_1, \dots, g_{i-1}, -g_j, \dots, -g_i, g_{j+1}, \dots, g_N.$$

Definition 1. Genes i and j are said to be adjacent in the gene-order if either i is followed immediately by j or $-j$ is followed immediately by $-i$. Such a pair (i, j) is called an adjacency.

Remark 1. We note that (i, j) and $(-j, -i)$ are the same adjacency. We usually view a gene-order as the set of its adjacencies.

In order to reconstruct a phylogeny, we assume that there is an underlying model tree T with n leaves rooted at ρ , and for each edge $e \in T$, a real number $D(e)$ specifying the mutation rate. The genome on every node of T evolves from the genome at ρ . For edge $e = (u, v) \in T$, with u being the parent of v , the genome at v is the result of applying $D(e) \cdot N$ random inversions to the genome at u . Note that $D(\cdot)$ induces a tree metric on nodes in T . Given a set of observed genomes on the leaves, we want to reconstruct T .

In this paper, we address inversions only. Our theoretical results should apply to transpositions without much effort. Empirically, Moret et al. [18] have found a good correlation between performance on inversion-only data and inversion/transposition data.

1.2 Our Approach

The key idea of our algorithm is to not just use the distances between leaves, but to also learn ancestral genomes. For molecular data (i.e. A, C, G, T), the ancestral approach has been shown to be more accurate than distance-based methods in theory and practice. In practice, likelihood method (e.g. RAxML [24] and PhyML [12]) and parsimony methods (e.g. Fitch-Hartigan [11, 13])

which learn internal sequences significantly outperform distance methods. In theory, the recursive majority algorithm for learning ancestral sequences was introduced and analysed by Mossel [20]. This recursive procedure is a key step in the logarithmic-sized phylogeny reconstruction algorithms of Daskalakis, Mossel, and Roch [6] and Mihaescu et al. [15].

The algorithm we are based upon [15] maintains a forest of correct subtrees of T , infers adjacencies on interior nodes of each tree using genomes on its leaves, and uses that information to decide how to join these subtrees. The main tool for joining subtrees is the now classical four-point method which can properly sort out the topology of a quartet given sufficiently accurate distances among the four positions. Our contribution is to show that sufficiently accurate distances can be recovered at internal nodes even using gene-order data.

In order to deal with gene-order data, we must reconcile the Nadeau-Taylor model for gene-order data with character based models of evolution, which assume that each character evolves independently. We simplify the gene-order data by showing that it's sufficient to represent a genome by its set of adjacencies rather than its full signed permutation. We also bound the error that comes from assuming that these adjacencies evolve independently like characters. This simplification leads to a sufficiently accurate distance estimator and ancestral learning algorithm which we can combine with the algorithm of Mihaescu et al. [15] to produce a polynomial time method that uses logarithmic data.

We implement a variant of this method and compare it against the state of the art polynomial time methods on simulated data using the framework described in [17]. We further demonstrate that for a small real world example that our performance is comparable to the advertised performance of GRAPPA [18], an exponential time approach to gene order phylogenetics.

Again, we have focused on inversions. From previous studies [17] which found that inversion only and transposition data to be very similar, we expect our algorithm to empirically work well for transpositions as well. Furthermore, our theoretical bounds should also extend given a reasonable model of transpositions versus inversions. We leave the analysis out for now.

1.3 Related work

Distance-based methods have been widely used in phylogeny reconstruction. These methods could be carefully applied to run in polynomial time and with polynomial length sequences. Furthermore, it was shown that such methods, in fact, required polynomial length sequences.

In a breakthrough, Daskalakis, Mossel, and Roch [6] shows that sequence data could reconstruct the tree with *logarithmic length* sequences. Later, Mihaescu et al. [15], built on the insight from [6] to give a simpler algorithm, which is the basis of this current work. Both algorithms rely on the recursive majority algorithm shown to be effective in Mossel [20] and work when the mutation probability on each edge is no larger than the phase transition probability.

In practice, methods based on parsimony and maximum likelihood that took advantage of the sequence data were shown to work better than distance methods.

For gene order data, the first attempt by Blanchette, Kunisawa, and Sankoff [4] applies the neighbor joining method to the breakpoint distance matrix. Wang and Warnow [27] analyze the expected number of breakpoints given the number of evolution events and use the derived bounds to find more effective estimates of the true evolution distances. Various subsequent results follow this one by either providing better estimates or using other distance parameters to estimate the true distances, e.g., Wang [25, 26] and Eriksen [9]. We point the reader to surveys by Moret, Tang, and Warnow [18] and by Wang and Warnow [28]. Moret et.al. [17] also show that these methods are quite effective in a program in a simulation study.

Researchers are also interested in using other parameters beside the numbers of breakpoints or adjacencies, notably parameters related to conserved intervals (see, e.g., [2, 5, 3]). Blin, Chauve, and Fertin [5] use these parameters in phylogeny reconstructions. Bergeron, Blanchette, Chateau, and Chauve [3] present an algorithm that reconstructs ancestral gene orders given a tree topology using conserved intervals.

2 Framework and Gene Order Data

In this section, we first describe the algorithmic framework provided by Mihaescu et al. [15] which can be viewed as calling subroutines to compute distances among “objects” and an “object” learning algorithm. As noted their objects were sequence data, ours will be gene order data represented as adjacencies. We then proceed to provide some background for our learning method and distance estimator.

2.1 The framework

We describe an un-optimized version of Mihaescu’s algorithm for the sake of simplicity. We will later discuss how to make it more efficient.

We view taxa as associated with an object, either a sequence or a set of adjacencies.

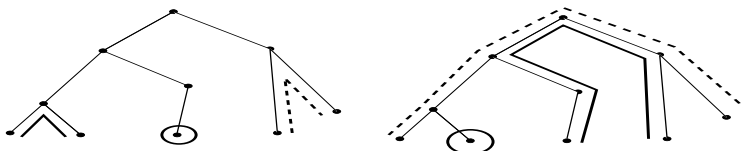


Fig. 1. Two evolutionary trees and forests on their leaves are illustrated. The three trees in the forest are indicated by dotted lines, solid lines and circles. The dotted, and solid line trees in the forest each consist of a single edge. The figures shows the corresponding induced paths through the evolutionary tree. The forest on the right fails to meet the invariant of the algorithm since the induced paths for the dotted and solid trees overlap.

The algorithm begins by forming a forest consisting of each taxa as an isolated tree. At each step it combines two subtrees into a larger tree. A tree, T_f , in the forest corresponds to a set of taxa and an induced subtree in the true evolutionary tree T . An edge in T_f may correspond to a path in T . The algorithm maintains the property that T_f in the forest has the same topology as the induced tree on its taxa in T (viewing paths as edges.) The algorithm further maintains the property that the set of trees in the forest correspond to disjoint induced subgraphs in T . Figure 1 shows examples of valid and invalid forests represented as induced paths in an evolutionary tree.

The algorithm joins trees by locating the shortest path (in evolutionary distance) in the evolutionary tree between two subtrees in the forest. Adding an edge corresponds to joining the two

subtrees, say T_1 and T_2 . That edge corresponds to a path in the evolutionary tree. By choosing the shortest such path and corresponding “edge”, the algorithm maintains the disjointness condition of the forest. By choosing to attach the edge to the appropriate place in T_1 and T_2 , the algorithm maintains the correct topologies on the subtrees of the forest. (Again, in Figure 1 this may correspond to the path in the evolutionary tree in the right figure from the solid circle subtree to the dotted subtree.)

In the general case, the algorithm needs to find the correct pair of trees, and edges $e_1 = (a, b)$ and $e_2 = (c, d)$ in the forest to serve as the endpoints of the new edge. We assume that it tries all pairs of edges and estimates the distance of the connecting edge as follows. It removes the edges $e_1 = (a, b)$ and $e_2 = (c, d)$ breaking their respective trees into two. It then learns internal sequences (adjacencies in our case) at $a, b, c,$ and d from the separate subtrees that they have been split into. These internal sequences can then be used in the four point method to estimate the length of the edge connecting the two edges assuming that this is indeed the correct pair of edges. This follows since the sequences at $a, b, c,$ and d are independent random independent variables since they are learned from entirely separate subtrees in the true evolutionary subtree. If we are at an incorrect pair of edges $(a,b), (c,d)$ one or more of the subtrees will contain a portion of the path in the tree connecting the edges. This creates dependencies between the learning algorithm in the supposedly separate subtrees and makes the distance computation invalid. The algorithm contains a check for this case (essentially the four point method at adjacent edges). The analysis of this case requires dealing carefully reasoning about the dependencies.

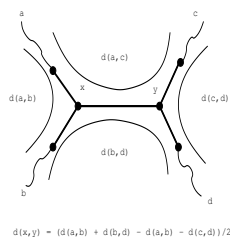


Fig. 2. The classical four point method with internal sequences deals with the noise from learning (indicated by squiggly lines) as well. Notice that the distance from x to a is added in the calculation $d(a, c)$ and subtracted in $d(a, b)$ thus removing that the result. Continuing along these lines, we see that the length of the middle edge remains. In contrast, directly measuring the length of the middle edge using learned sequences at x and y would overstate its length by the noise in the learning process.

The four point method is illustrated in Figure 2. The idea is that while the learned sequences contain noise from learning and the outer edges the noise is cancelled since it appears in distances between the points that are added and subtracted. Still, the pairwise distances estimates need to have small bias and good convergence which is what we will eventually show for our adjacency based methods.

Now, given that the middle edge distance output by the four point method is sufficiently well estimated by a distance estimator and learning method the algorithm finds the shortest possible joining edge and proceeds.

The efficient algorithm optimizes this approach by caching various sequence learning results (carefully since the specific subset of leaves used is important), by turning the checking method that indicates we are joining incorrectly into a direction indicator that allows the algorithm to move toward the correct edge for a pair of trees.

The authors prove that this algorithm works as long as the evolution along any edge is strictly less than $1/\sqrt{2}$ flips per edge for sequences, with $O(\log n)$ length sequences.

A bit more formally, a restatement of their results is as follows.

Theorem 1. *If T is an evolutionary tree with edge weights that correspond to the probability of a mutation event per unit length of the data where with edge lengths at most $\tau - g$ where τ is a fixed threshold for the data and g is arbitrarily small.*

Assume further that there is a learning algorithm for the data, and an associated four point method on the data that

- *genomes of length of some sufficient length which remains $O(\log n)$,*
- *and four nodes $a, b, c,$ and $d,$ and disjoint subtrees $T_a, T_b, T_c,$ and T_d of the evolutionary tree where each T_x has no edge longer than $\tau - g$ where paths in the evolutionary trees are viewed as edges in $T_x,$*

finds the middle edge length in the associated quartet to within ϵ with high probability.

Given such an algorithm, then with high probability, T can be reconstructed using data of length $O(\log n)$.

Mihaescu (following Daskalakis et al. [6]) shows that the above can be done for sequence data for $\tau = 1/\sqrt{2}$. This is the best one can hope for logarithmic data as shown by Mossel [20].

We give a learning algorithm and associated distance function for gene order data for a fixed but small τ . This can be improved by increasing the depth of our recursive majority learning algorithm in the same manner as was done in [6]. Still, for gene order data getting optimal bounds for τ for gene order data remains an interesting open problem.

Finally, we note the dependence on f and g are inversely quadratic and inversely linear respectively. The behavior with respect to these parameters is asymptotically optimal.

In the following section, we give the necessary background for describing and analyzing our learning and associated distance estimation algorithms

2.2 Gene Order Data, Adjacencies, Distances and the Four Point Method

The notion of distance that we use is the natural extension of breakpoint distances to adjacencies. Recall that the set of adjacencies for a gene order is the set of pairs of genes that are next to each other in the gene.

For node $u \in T$, we also let u denote the random variable corresponding to the genome at node u , in particular we let it denote the set of all adjacencies in u . For subtree T' of T rooted at u , such that $\delta T' \subseteq \delta T$, we let $\tilde{u}(T')$ denote the set of learned adjacencies at node u by the recursive majority procedure on tree T' (which we describe in Section 4). Note that $\tilde{u}(T')$ is a random variable. For two random genomes u and v , let

$$\hat{\theta}(u, v) = |u \cap v|/N; \theta(u, v) = \mathbf{E}[|u \cap v|/N]$$

be the fractions of common adjacencies, and its expectation. We also let estimated distance and true distance

$$\hat{d}(u, v) = \frac{\log \hat{\theta}(u, v)}{N \log(1 - 2/N)}; d(u, v) = \frac{\log \theta(u, v)}{N \log(1 - 2/N)}.$$

Intuitively, since $\theta(u, v)$ is approximately $(1 - 2/N)^{D(u,v)N}$ (see Section 3), $d(u, v)$ approximates $D(u, v)$.

To extend the distance notation to sets of learned adjacencies, we let

$$d(\tilde{u}(T'), \tilde{v}(T')) = \frac{\log(\mathbf{E}[|\tilde{u}(T') \cap \tilde{v}(T')|]/N)}{N \log(1 - 2/N)}; \hat{d}(\tilde{u}(T'), \tilde{v}(T')) = \frac{\log(|\tilde{u}(T') \cap \tilde{v}(T')|/N)}{N \log(1 - 2/N)}.$$

For a fixed set A of adjacencies, we also define notions of distances with respect to A as follows. Let

$$\theta_A(u, v) = \mathbf{E}[|u \cap v \cap A|/N],$$

and

$$d_A(u, v) = -\log \theta_A(u, v).$$

We define $\hat{\theta}_A(\cdot, \cdot)$ and $\hat{d}_A(\cdot, \cdot)$ accordingly.

We use these concepts to implement the four point method required in Theorem 1. Denote by $(a, b|c, d)$ a quartet Q such that a and b form a cherry and c and d form a cherry with respect to this quartet.

Recall that the four-point method FPM, given distance metric \hat{d} on the four leaves, returns the quartet topology $Q = (x, y|z, t)$ that minimizes the sum $\hat{d}(x, y) + \hat{d}(z, t)$ over all permutations (x, y, z, t) of (a, b, c, d) .

Let,

$$\text{ME}(\hat{d}; x, y|z, t) = \frac{1}{4}(\hat{d}(x, z) + \hat{d}(x, t) + \hat{d}(y, z) + \hat{d}(y, t) - 2\hat{d}(x, y) - 2\hat{d}(z, t)).$$

If d_Q is any tree metric corresponding to the quartet topology Q , $\text{FPM}(d_Q) = Q$. Let e be the length of the middle edge of Q . We also have that $\text{ME}(d_Q; a, b|c, d)$ returns e .

If we are given another approximate metric \hat{d} such that for all pairs $x, y \in \{a, b, c, d\}$, $|\hat{d}(x, y) - d_Q(x, y)| < \delta$, then $|\text{ME}(\hat{d}; x, y|z, t) - e| < 2\delta$, and if $\delta < e/2$, we get that $\text{FPM}(\hat{d}) = Q$.

In our algorithm, the four point method is presented with learned adjacencies \tilde{a} , \tilde{b} , \tilde{c} and \tilde{d} . Thus, the actual quartet consists of the original quartet in the evolutionary tree and the noise from learning as illustrated in Figure 2.

If the noise introduced by using \tilde{a} instead a is bounded and all the edges are small enough (less than a constant fraction of the adjacencies change), $O(\log 1/\delta)$ bits is enough to estimate $\hat{d}(x, y)$ to within δ . To get a high probability result (i.e., for the method to work on every one of the $O(n)$ edges), we need to increase the number of bits by an $O(\log n)$ factor.

The following theorems us to conclude that our estimator are sufficiently unbiased enough and our learning adds small enough noise so that the four point method to work. The proofs of these theorems are sketched in Section 3 and 4. (Fuller proofs will be available in the journal version.)

Theorem 2. *For any $\epsilon > 0$ and $M > 0$ there exists an $N = O(\log n)$ such that for two random genomes G and G' the following are true:*

1. $\Pr[|\hat{d}(G, G') - D(G, G')| > \epsilon] < 1/n^3$ when $D(G, G') \leq M + \epsilon$;
2. $\Pr[\hat{d}(G, G') < M] < 1/n^3$ when $D(G, G') > M + \epsilon$.

Theorem 3. *For any $\epsilon_2, \epsilon_3 > 0$, there exists $N = O(\log n)$ such that for tree $T' \subseteq T$, rooted at u , where $\delta T' \subseteq \delta T$,*

- (1) *For a set of adjacencies A , such that $|A \cap u| \geq \beta N$ for some $\beta <= 1$, let $\mu'_u = \mathbf{E}[|\tilde{u}(T') \cap u \cap A|]$. We have*

$$\Pr[| |\tilde{u}(T') \cap u \cap A| - \mu'_u | > \epsilon_2 \mu'_u] < 1/n^3,$$

(2) $\Pr [|\tilde{u}(T') - u| > \epsilon_3 N] < 1/n^3$.

The following central lemma shows that with these parameters the four-point method succeeds with high probability for quartets with diameter, under D , at most M . The proof uses Theorem 2 and 3 to bound the sizes of $\tilde{u} \cap \tilde{v}$ for pairs of sets of learned adjacencies.

Lemma 1. *Let a, b, c, d be nodes in a tree T and let $Q = (a, b|c, d)$ be the subtree they induce on T . Let T_a, T_b, T_c, T_d be subtrees of T rooted at a, b, c, d respectively, such that $\delta(T_x) \subseteq \delta(T)$ for all $x \in \{a, b, c, d\}$. If the diameter of Q under D is at most $M = O(\tau + g + f)$ and the length of the middle edge of Q is e , then*

$$\text{FPM}(\hat{d}; \tilde{a}(T_a), \tilde{b}(T_b), \tilde{c}(T_c), \tilde{d}(T_d)) = Q$$

and

$$|\text{ME}(\hat{d}; \tilde{a}(T_a), \tilde{b}(T_b)|\tilde{c}(T_c), \tilde{d}(T_d)) - e| < f/10,$$

with high probability.

Since the middle edges are measured to within a tenth of its length, then the correct quartet is returned and the middle edge that is returned is reasonably accurate.

3 Estimating the true evolutionary distance

Let G and G' be genomes such that G evolves to G' according to the Nadeau-Taylor model with k evolution events. Let $A(G, G') = |G \cap G'|$ be the number of their common adjacencies. Let $\mu = \mathbf{E}[A(G, G')]$ be its expectation. There have been studies on μ , e.g., see [9, 25]. One can compute the exact value of μ given k . Here, we are interested in the question of how close is $A(G, G')$ and μ . Given k , we will prove the following bounds on μ :

$$N \left(1 - \frac{2}{N}\right)^k \leq \mu \leq 1 + (N - 1) \left(1 - \frac{2}{N - 1}\right)^k$$

From these bounds, it can be seen that our distance

$$d(G, G') = \frac{-\log(\mu/N)}{N} \cdot \log(1 - (1 - 2/N))$$

is approximately k/N . Hence, if $A(G, G')$ is close to μ , our distance $\hat{d}(G, G')$ will be close to $k/N = D(G, G')$ as well.

In this section we show that $A(G, G')$ is concentrated around its mean. Hence, our approximation of k is accurate, i.e., $\hat{d}(G, G')$ is close to $D(G, G')$ with high probability for $N = O(\log n)$. More precisely, this section was devoted to proving Theorem 2.

We put a sketch of the proof in the appendix.

4 Learning Ancestral Adjacencies

In this section we will show how to recover, up to an *a priori* bounded error, the set of adjacencies at the interior nodes of a phylogenetic tree from the adjacencies at the leaves of the tree, by means of a recursive algorithm.

Definition 2. Let $T = (V, E)$ be a tree rooted at ρ with boundary (leaf-set) δT .

Let T' denote a rooted binary tree with n nodes. Let B denote the tree induced by the first two top-most levels of T' . Let r denote B 's root, let a, b, c and d denote its 4 leaves, and let r_1 and r_2 denote two internal nodes, where r_1 (r_2) is the parent of a and b (c and d). Let T'_u denote a subtree of T' rooted at u . Recall that $\tilde{a}(T'_a), \tilde{b}(T'_b), \tilde{c}(T'_c)$, and $\tilde{d}(T'_d)$ denotes the predicted adjacencies at nodes a, b, c , and d , respectively. Since T' is clear in this section, for brevity, we omit T' and write $\tilde{a}, \tilde{b}, \tilde{c}, \tilde{d}$, and \tilde{r} for $\tilde{a}(T'_a), \tilde{b}(T'_b), \tilde{c}(T'_c), \tilde{d}(T'_d)$, and $\tilde{r}(T')$.

Our prediction uses the recursive majority algorithm on these quartets. The predicted adjacency set \tilde{r} is

$$\{(x, y) : (x, y) \text{ is an adjacency in at least 3 predicted sequences at the leaves}\}.$$

With this definition, we never get inconsistency, i.e., if $(x, y) \in \tilde{r}$, for any $y' \neq y$, $(x, y') \notin \tilde{r}$. The goal is to correctly recover a large fraction of r .

Definition 3. Adjacencies in $\tilde{r} \cap r$ are recovered adjacencies, while we adjacencies in $\tilde{r} - r$ are accidental adjacencies. We make a particular note for adjacencies in \tilde{r} that, after being destroyed, get recreated during the evolution process. An adjacency i is recovered in the box model if it gets recovered even if we discard all recreated adjacencies.

To analyze \tilde{r} , we consider three types of adjacencies in \tilde{r} , (1) adjacencies recovered in the box model, (2) other recovered adjacencies, called recreated adjacencies, and (3) accidental adjacencies. Let X_b, X_r , and X_a denote the number of adjacencies in each type, respectively. Later we shall see that mostly \tilde{r} consists of adjacencies in the first type.

4.1 Adjacencies recovered in the box model

We first analyze X_b . This gives the lower bound on the expected number of recovered adjacencies. We assume the box model. Let θ denote the lower bound on the probability that a given adjacency is preserved by the evolution process on one edge of the tree. If we assume that along the edge of the tree, at most k evolutionary events occurs, $\theta \geq (1 - 2/n)^k$. The next lemma analyzes the probability that an adjacency is recovered. It shows that for some θ it is possible to recover an adjacency with constant probability.

Lemma 2. For some θ , there is a constant α such that, if an adjacencies at node v , for $v \in a, b, c, d$ is recovered recursively with probability at least α , an adjacency i at r is recovered with probability at least α .

Lemma 2 implies the following corollary.

Corollary 1. For some θ , there is a constant α such that, $\mathbf{E}[X_b] \geq \alpha N$.

We not only want X_b to be large, since we use the set of learned adjacencies to estimate the closeness of two nodes in the tree, we want X_b to be near its expectation with high probability. We now show that X_b does not deviate much from $\mathbf{E}[X_b]$. For adjacency $i \in r$, let indicator random variable $W_i = 1$ iff adjacency i is recovered in the box model. Note that the W_i 's are not independent. However, it can be shown that they are negatively dependent. As observed in Dubhashi and Ranjan [8], the Chernoff bound still applies. Therefore, we have the following lemma.

Lemma 3. Let $\mu_b = \mathbf{E}[X_b]$. The following are true:

$$\begin{aligned} \Pr[X_b < (1 - \delta)\mu_b] &< \exp(-\mu_b\delta^2/2); \\ \Pr[X_b > (1 + \delta)\mu_b] &< \exp(-\mu_b\delta^2/3). \end{aligned}$$

5 Implementation and Experiments

5.1 Implementation

We implement the algorithm described above, except that the learning algorithm is not recursive majority. Our learning algorithm estimates edge lengths (again with the four point method) and then for each possible adjacency computes a value that is the weighted sum of its children's value for that adjacency. The leaves are initialized with a 1 for each existing adjacency and 0 for others. The set of adjacencies that the learning program outputs correspond to the largest valued adjacency for each gene, i.e., each gene gets one guy to be on the right of him.

We also briefly experimented with using bayes rule to compute the most likely adjacency for each gene. The method appeared to work a bit worse than the weighted value algorithm above. Note that both of these methods, like the recursive majority method of Section 4, do not necessary produce a consistent set of adjacencies.

All of our runs that we report here are with the weighted value algorithm.

5.2 Simulations

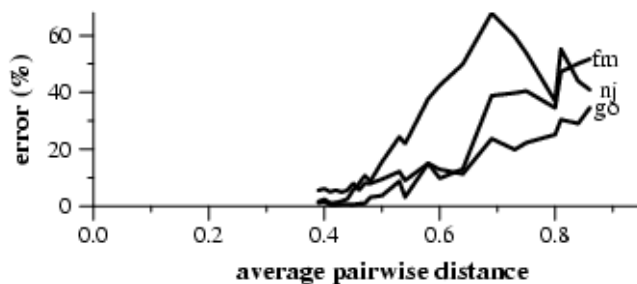


Fig. 3. Experimental results for 37-gene dataset; nj=neighbor joining, fm=FastME, go=our program.

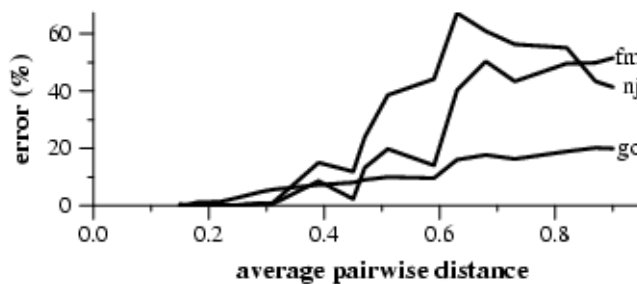


Fig. 4. Experimental results for 120-gene dataset; nj=neighbor joining, fm=FastME, go=our program.

We followed the outlines of the simulation study presented in [17]. We generate trees using `r8s` [22] and then perturb the edge lengths by choosing a random number s in the range $[-c, c]$

and multiplying by e^s . We try values of s in 1, 2, 3, 4, 5. The higher the number, the further we get from the ultrametric tree that is handed to us by `r8s`.

We then scale the resulting trees by multiplying each edge by a constant factor from 0.025, 0.05, 0.1, 0.2, 0.4, 0.8. The purpose of this step is to give us trees with a wide range of inversion distances between pairs of nodes. Average pairwise distance turns out to be a good proxy for the difficulty of reconstructing a tree.

We run 3 trees for each parameter setting and generate 10 gene order datasets for each tree. We then use a Poisson model using the edge weight as the mean to pick a number of inversions and choose uniformly among them. We further ensure that every edge has at least one inversion event. Thus empirical tree is fully resolved.

We compared our program to distance methods using distances computed with the EDE correction [17]. We ran both the FastME algorithm [7], and the neighbor joining algorithm [21] from the PHYLIP package [10].

We processed the data and sorted by our choice of c , and then sorting according to average pairwise inversion distance among the genomes (which tops out at the length of the genomes). We plot the results for c equal to 2 in Figures 5.2 and 5.2. We chose s -factor 2 since it gave the broadest range of possible branch lengths. The results for the other s -factors are similar and can be provided upon request. Essentially, as indicated in the graph we do a bit worse for very low average distance (i.e., most branches have one inversion) and better for large average distance (i.e., when the evolutionary rate is quite high.)

Our programs, and scripts for generating data, and the data and results are all available online [1].

5.3 A brief comparison with GRAPPA

We analyzed the dataset provided with GRAPPA with our program and got one edge wrong when compared to the accepted tree. An examination of the edge indicates that it is small and perhaps not well supported in any case. This is comparable with results described in Moret et al. [16].

6 Discussion

As noted above, our method does considerably better than state of the art distance based methods for large evolutionary distances and a bit worse for very small evolutionary distances. The improvement in the high evolution regime is expected as the learning of internal sequences allows more accurate resolution of deeper edges. If the evolutionary distance is low, however, even deep edges are sufficiently close to known sequences so that distance methods are fine.

Our loss in the low evolution regime is frankly surprising. We would have expected similar performance since this case is easy (and all programs do relatively well). Perhaps learning of internal sequences is unwarranted and introduces noise in this case.

Another possibility is that our variant of sequence learning could be improved. While exploring the differences between bayesian learning and weighted learning, we noticed that the edges that were wrong varied significantly in the reconstructed trees. In one example, weighted value learning got 22 false positives compared to 25 for bayesian learning, but the two trees differed by 13 edges. Thus, they are wrong on a substantially different set of edges.

In any case, in low evolution regimes, it may simply be better to use distance methods. Our contribution provides a tool for high evolution regimes.

References

1. Dan Adkins. Tar file for recomb submission. www.cs.berkeley.edu/~satishr/private-tar-for-recomb.tar.
2. Séverine Bérard, Anne Bergeron, and Cedric Chauve. Conservation of combinatorial structures in evolution scenarios. In *Comparative Genomics*, pages 1–14, 2004.
3. Anne Bergeron, Mathieu Blanchette, Annie Chateau, and Cedric Chauve. Reconstructing ancestral gene orders using conserved intervals. In *WABI*, pages 14–25, 2004.
4. M. Blanchette, T. Kunisawa, and D. Sankoff. Gene order breakpoint evidence in animal mitochondrial phylogeny. *Journal of Molecular Evolution*, 49:193–203, 1999.
5. Guillaume Blin, Cedric Chauve, and Guillaume Fertin. Genes order and phylogenetic reconstruction: Application to *amma*-proteobacteria. In *Comparative Genomics*, pages 11–20, 2005.
6. Constantinos Daskalakis, Elchanan Mossel, and Sébastien Roch. Optimal phylogenetic reconstruction. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on the Theory of Computing (STOC 2006)*, pages 159–168, 2006.
7. Richard Desper and Olivier Gascuel. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *Journal of Computational Biology*, 9(5):687–705, 2002.
8. D. Dubhashi and D. Ranjan. Balls and bins: A study in negative dependence. *Random Structures and Algorithms*, 13(2):99–124, 1998.
9. Niklas Eriksen. Approximating the expected number of inversions given the number of breakpoints. In *Proceedings of the 2nd Workshop on Algorithms in Bioinformatics (WABI 2002)*, volume 2452 of *Lecture Notes in Computer Science*, pages 316–330, 2002.
10. J. Felsenstein. PHYLIP (Phylogeny Inference Package) version 3.6, 2005. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle.
11. W. M. Fitch. Toward defining the course of evolution: Minimum change for a specific tree topology. *Systematic Zoology*, 20:406–416, 1971.
12. Stéphane Guindon and Olivier Gascuel. A simple, fast and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52(5):696–704, 2003.
13. J. A. Hartigan. Minimum mutation fits to a given tree. *Biometrics*, 29:53–65, 1973.
14. Anil Kamath, Rajeev Motwani, Krishna Palem, and Paul Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. *Random Structures and Algorithms*, 7(1):59–80, 1995.
15. Radu Mihaescu, Cameron Hill, and Satish Rao. Fast phylogeny reconstruction through learning of ancestral sequences. *CoRR*, [arxiv:abs/0812.1587](https://arxiv.org/abs/0812.1587), 2008.
16. Bernard M. E. Moret, Adam C. Siepel, Jijun Tang, and Tao Liu. Inversion medians outperform breakpoint medians in phylogeny reconstruction from gene-order data. In *Proceedings of the 2nd Workshop on Algorithms in Bioinformatics (WABI 2002)*, volume 2452 of *Lecture Notes in Computer Science*, pages 521–536, 2002.
17. Bernard M. E. Moret, Jijun Tang, Li-San Wang, and Tandy Warnow. Steps toward accurate reconstruction of phylogenies from gene-order data. *Journal of Computer and System Sciences*, 65(3):508–525, 2002.
18. Bernard M. E. Moret, Jijun Tang, and Tandy Warnow. Reconstructing phylogenies from gene-content and gene-order data. In Olivier Gascuel, editor, *Mathematics of Evolution and Phylogeny*, pages 321–352. Oxford University Press, 2005.
19. Bernard M. E. Moret, Li-San Wang, Tandy Warnow, and Stacia K. Wyman. New approaches for reconstructing phylogenies from gene order data. *Bioinformatics*, 17(1):165–173, 2001.
20. Elchanan Mossel. On the impossibility of reconstructing ancestral data and phylogenies. *Journal of Computational Biology*, 10:669–678, 2003.
21. N. Saitou and M. Nei. The neighbor-joining method: a new method, for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.
22. Michael J. Sanderson. r8s: Inferring absolute rates of molecular evolution and divergence times in the absence of a molecular clock. *Bioinformatics*, 29(2):301–302, January 2003.
23. David Sankoff, Chunfang Zheng, Adriana Mu noz, Zhenyu Yang, Zaky Adam, Robert Warren, Vicky Choi, and Qian Zhu. Issues in the reconstruction of gene order evolution. *Journal of Computer Science and Technology*, 25(1):10–25, January 2010.
24. Alexandros Stamatakis. RAxML-VI-HPC: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, 22(21):2688–2690, November 2006.
25. Li-San Wang. Exact-IEBP: A new technique for estimating evolutionary distances between whole genomes. In *Proceedings of the First Workshop on Algorithms in Bioinformatics (WABI 2001)*, volume 2149 of *Lecture Notes in Computer Science*, pages 175–188, 2001.
26. Li-San Wang. Genome rearrangement phylogeny using Weighbor. In *Proceedings of the 2nd Workshop on Algorithms in Bioinformatics (WABI 2002)*, volume 2452 of *Lecture Notes in Computer Science*, pages 112–125, 2002.

27. Li-San Wang and Tandy Warnow. Estimating true evolutionary distances between genomes. In *Proceedings of the Thirty-Third Annual ACM Symposium on the Theory of Computing*, pages 637–646, 2001.
28. Li-San Wang and Tandy Warnow. Distance-based genome rearrangement phylogeny. In Olivier Gascuel, editor, *Mathematics of Evolution and Phylogeny*. Oxford University Press, 2005.

Theorem 4. *For any $\epsilon > 0$ and $M > 0$ there exists an $N = O(\log n)$ such that for two random genomes G and G' the following are true:*

1. $\Pr[|\widehat{d}(G, G') - D(G, G')| > \epsilon] < 1/n^3$ when $D(G, G') \leq M + \epsilon$;
2. $\Pr[\widehat{d}(G, G') < M] < 1/n^3$ when $D(G, G') > M + \epsilon$.

First we compute the probabilities that a random inversion will create or break a certain adjacency. Using the notation from Wang and Warnow [27], we compute the probabilities of randomly destroying an arbitrary adjacency (separation) and randomly recreating a previously destroyed adjacency (unification). Given that an inversion cuts a circular permutation in two distinct places, and that there are N places to cut, the total number of possible inversions is $\binom{N}{2}$. An adjacency can be broken by an inversion that cuts at the adjacency and at any of the other $N - 1$ places. Thus, the probability of separating an arbitrary adjacency is $s = 2/N$.

Now we consider reconstructing a broken adjacency (i, j) . Assume without loss of generality that i is positive and precedes j . In general, there are two possibilities. If j is positive, it is impossible to rejoin i and j in one inversion. If j is negative, there is precisely one inversion that brings j next to i , namely $(i + 1, j)$. Thus the unification probability for an arbitrary broken adjacency in the second case is $u = 2/(N(N - 1))$. When dealing with unsigned genomes, it is always possible to recreate a given adjacency. Thus the above equation holds without any condition on the signs of the genes involved.

We start with the lower bound. We define the random variable Z^L to be the number of adjacencies remaining after k steps, assuming that they only break and are never reconstructed. This is clearly a lower bound on the true number of adjacencies remaining. We can compute the expected value of Z^L by examining one adjacency. At each time step, that adjacency will be broken with probability $2/N$. It survives k steps with probability $(1 - 2/N)^k$. By linearity of expectation,

$$\mu_L = \mathbf{E}[Z^L] = N \left(1 - \frac{2}{N}\right)^k.$$

To get a concentration bound is not so simple, though, because the adjacency breaking events are not independent. At any time, at most two adjacencies can be broken. Even though the events are not mutually independent, it can be shown that they have negative dependencies. Therefore, as observed by Dubhashi and Ranjan [8], we can still use Chernoff's bound to obtain the following result.

Lemma 4. *Let $\mu_L = \mathbf{E}[Z^L] = N(1 - 2/N)^k$. Then $\Pr[Z^L < (1 - \delta)\mu_L] < \exp(-\mu_L\delta^2/2)$.*

Now we consider the upper bound. We define the random variable Z^H to be the number of adjacencies remaining after k steps *ignoring* the signs. The analysis of the upper bound is more involved since we must consider recreated adjacencies.

We first analyze the expected value $\mathbf{E}[Z^H]$, paying attention to how it changes over time. We consider genomes in this analysis *unsigned* circular genomes. We define events regarding the presence of adjacencies in the genomes. Because of linearity of expectation, it suffices to analyze one particular adjacency. Consider a fixed pair of consecutive genes $a = (g_i, g_{i+1})$. Let E_k be the event that there is no breakpoint at a in G_k , i.e., g_i and g_{i+1} are adjacent in G_k . The following

claim calculates the probabilities $\Pr[E_k|E_0]$ and $\Pr[E_k|\bar{E}_0]$ by solving recurrences as in Wang and Warnow [27]. This calculation appears more detailed than necessary to analyze the expected value of Z^H , but we shall use the results again later to prove a concentration bound.

Claim. The following are true for any $k \geq 0$:

$$\Pr[E_k|E_0] = \left(1 - \frac{1}{N}\right) \left(1 - \frac{2}{N-1}\right)^k + \frac{1}{N} \quad (1)$$

$$\Pr[E_k|\bar{E}_0] = \frac{1}{N} \left(1 - \left(1 - \frac{2}{N-1}\right)^k\right) \quad (2)$$

Let random variable Y_t denote the number of adjacencies between G_0 and G_t ignoring the signs. Let $Z_t^H = \mathbf{E}[Z^H|Y_t]$. Note that Z_0, Z_1, \dots is a martingale sequence. Using these two expressions derived above, we have that

$$Z_t^H = Y_t(\Pr[E_{k-t}|E_0]) + (N - Y_t)(\Pr[E_{k-t}|\bar{E}_0]) = 1 + (Y_t - 1) \left(1 - \frac{2}{N-1}\right)^{k-t}.$$

By plugging in $Y_0 = N$ at $t = 0$ in the previous expression, we get the expected value of Z^H .

Corollary 2.

$$\mu_H = \mathbf{E}[Z^H] = 1 + (N - 1) \left(1 - \frac{2}{N-1}\right)^k.$$

We now state the concentration lemma, whose proof, following the balls and bins analysis of Kamath et al. [14], uses Azuma's inequality.

Lemma 5.

$$\Pr[|Z^H - \mu_H| \geq \lambda] \leq 2 \cdot \exp\left(-\frac{\lambda^2(N-1)}{8(N^2 - \mu_H^2)}\right).$$

To prove Theorem 2, we need two concentration lemmas for Z^L and Z^H . For Z^L , we prove that the underlying random variables for Z^L is negatively dependent [8] and then use Chernoff's bound. For Z^H , following the balls-and-bins analysis of Kamath et al. [14], we use Azuma's inequality to show the concentration result.