A Practical Algorithm for Recovering the Best Supported Edges of an Evolutionary Tree (*Extended Abstract*)

Vincent Berry* David Bryant† Tao Jiang‡ Paul Kearney§ Ming Li¶
Todd Wareham‖ Haoyong Zhang**

Abstract

It is now routine for biologists to conduct evolutionary analyses of large DNA and protein sequence datasets. A computational bottleneck in these analyses is the recovery of the topology of the evolutionary tree for a set of sequences. This paper presents a practical solution to this challenging problem. In particular, a new technique, called *hypercleaning*, is presented that can be combined with various tree-building algorithms to efficiently reconstruct from sequence data the best supported edges of the evolutionary tree. More precisely, the hypercleaning technique computes from sequence data a small subset of edges that is likely to contain most edges of the correct tree. A tree-building algorithm then attempts to identify edges in the subset that are compatible with each other and hereby produces an evolutionary tree. We also propose a simple greedy agorithm that builds a tree by screening the edges provided by hypercleaning in the decreasing order of support from sequence data. This technique is a substantial improvement over previous algorithms in its ability to recover edges of the evolutionary tree. Hypercleaning also incorporates a detailed error model that relates errors in the data to the topology of the evolutionary tree. The results of a simulation study that strongly support the practicality, efficiency and effectiveness of hypercleaning are also presented.

1 Introduction

Advances in DNA sequencing technology this decade have resulted in an exponential growth in the amount of sequence data available for biological analysis [8]. As a consequence, it is now routine for biologists to conduct evolutionary analyses of large sequence datasets [16, 17, 21]. However, standard methods for inferring evolutionary trees from sequence data, such as maximum likelihood [14] and maximum parsimony [26], are plagued by computational difficulties.

An evolutionary tree T for a set S of sequences is a rooted and edge weighted tree where the leaves of T are labeled bijectively by S. The topology of T (that is, T without edge weights) describes the speciation events resulting in the evolution of sequences in S from the root. The edge weights of T are proportional to the amount of evolution (sequence substitutions, insertions and deletions) between speciation events.

This paper presents new insights on the difficult computational problem of determining the topology of an unknown evolutionary tree T given only the set S of sequences that label the leaves of T. This problem is of practical importance since once the topology of T is known, T can be rooted and edge weights determined, resulting in a hypothesis describing the evolutionary history of the sequences in S. Determining the topology of T is considered to be the most difficult computational step in the reconstruction of T from S.

It is well–known that the topology of an evolutionary tree can be specified by its set of edge-induced bipartitions. An evolutionary tree T labeled by S

*Address: Département d'Informatique Fondamentale et Applications, LIRMM, Université de Montpellier II, France. Part of this work was done at the Département de Mathématiques, EURISE, Université de Saint-Etienne, France. E-mail: vberry@lirmm.fr

†Supported in part by a Bioinformatics Postdoc Fellowship from the CIAR, Evolutionary Biology Program and by NSERC and CGAT grants to D. Sankoff. Address: CRM Université de Montréal. E-mail: bryant@crm.umontreal.ca

‡Supported in part by NSERC Research Grant OGP0046613, a CITO grant, and a UCR startup grant. Department of Computer Science, University of California, Riverside, CA 92521. jiang@cs.ucr.edu. On leave from McMaster University.

§Supported in part by a CITO grant and NSERC Research Grant 160321. Address: Department of Computer Science, University of Waterloo, Waterloo, ON, N2L 3G1, Canada. E-mail: pkearney@math.uwaterloo.ca

¶Supported in part by NSERC Research Grant OGP0046506, a CITO grant, and the Steacie Fellowship. Address: Department of Computer Science, University of Waterloo, Waterloo, ON, N2L 3G1, Canada. E-mail: mli@math.uwaterloo.ca

‖Department of Computer Science, Memorial University of Newfoundland, St. John's, NF A1B 3X5, Canada. Work done at McMaster University. Email: harold@garfield.cs.umn.edu

**Department of Computer Science, University of Waterloo, Waterloo, ON, N2L 3G1, Canada. E-mail: h2zhang@wh.math.uwaterloo.ca

contains the bipartition (X, Y) of S if there is an edge e in T such that $T - \{e\}$ consists of two trees where one is labeled by X and the other by Y. This is denoted $e = (X, Y)$ and we use the terms 'edge' and 'bipartition' interchangeably. A set of bipartitions is *compatible* if there is a tree that contains these bipartitions. Compatibility of a set of bipartitions is characterized by the property that for every pair of bipartitions (A, B) and (C, D) in the set, at least one of A or B is a subset of C or D [11, 22].

One standard method for estimating evolutionary trees is to determine a *support function* that measures how well the sequence data S supports a bipartition (X, Y) and then construct a tree topology from the best supported bipartitions. An example of this approach is the spectral decomposition method of [18]. In this paper we describe a support function based on quartet data.

The set of all possible bipartitions is enormously large, and so, cannot be explicitly computed even for moderately sized S. Instead, we present a polynomial time algorithm for constructing a (polynomial-sized) collection of bipartitions most strongly supported quartet data.

1.1 Measuring the Support for a Bipartition

Recently the *quartet method* for constructing evolutionary trees from sequence data has received much attention in the computational biology community [2, 6, 9, 13, 20, 7, 25]. Given a quartet of sequences $\{a, b, c, d\}$ and an evolutionary tree T, the *quartet topology* induced in T by $\{a, b, c, d\}$ is the path structure connecting a, b, c and d in T. Given a quartet $\{a, b, c, d\}$, if the path in T connecting labels a and b is disjoint from the path in T connecting c and d, the quartet is said to be *resolved* and is denoted $ab|cd$. Otherwise, the quartet is said to be *unresolved* and is denoted $(abcd)$. The four possible quartet topologies induced by a quartet are depicted in Figure 1.

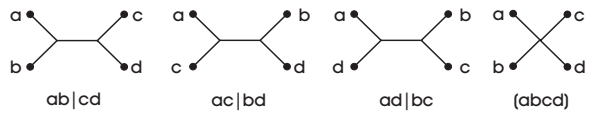

Figure 1: The four quartet topologies for quartet $\{a, b, c, d\}$.

The quartet method first estimates the quartet topology induced in T by each quartet in S. There are many methods for estimating quartet topology, including maximum likelihood [14], maximum parsimony [15], neighbor joining [23] and the ordinal quartet method [20]. Although many of these methods cannot be feasibly applied to the entire dataset S to infer the topology of T directly, they can be applied feasibly to infer tree topologies of size four. Restricting the analysis to smaller subsets also allows the alignment of a greater number of sites, while combining a large number of smaller analyses can lead to a more stable phylogenetic estimation [27].

Let Q be the set of these $\binom{n}{4}$ inferred quartet topologies. The quartet topologies in Q can be viewed as pieces of a larger puzzle T. The second step of the quartet method, *quartet recombination*, is to recombine the quartet topologies into an evolutionary tree topology T' that is an estimate of T. A number of heuristics for quartet recombination are available. There are methods based on clustering [24, 1, 4], leaf insertion [25, 27], greedy selection and quartet closure rules [13], semi-definite programming [2], and smooth polynomial integer programming [19]. Exact polynomial time algorithms for various restricted cases are given by [6] and [10].

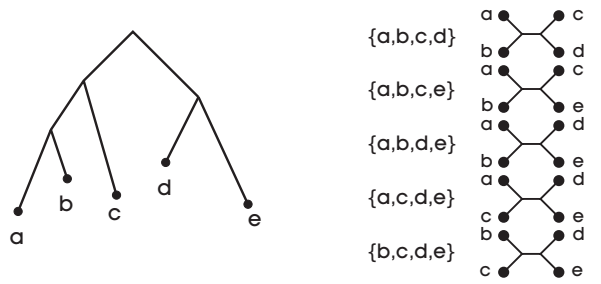

Figure 2: An evolutionary tree T and its set Q_T of induced quartet topologies.

The computational challenge of quartet recombination derives from the fact that the set Q of estimated quartet topologies can contain *quartet errors*. Define Q_T to be the set of quartet topologies induced by sequence quartets from S (see Figure 2). The quartet $\{a, b, c, d\}$ is a quartet error if $ab|cd \in Q_T$ but $ab|cd \notin Q$. Furthermore, $\{a, b, c, d\}$ is a quartet error *across* edge e in T if the removal of e from T disconnects a and b from c and d. Effectively, this defines the distribution of quartet errors throughout T. Quartet errors also permit us to assess how well Q supports an edge e by counting the number of quartet errors across e. This measure can be extended to measure the support for any bipartition. For a bipartition (X, Y) define $Q_{(X,Y)}$ to be the set of quartet topologies of the form $xx'|yy'$ where $x, x' \in X$ and $y, y' \in Y$. The *distance* from a set of quartets Q to a bipartition (X, Y) is defined to be

$$|Q_{(X,Y)} - Q|.$$

Note that the number of quartet topologies in $Q_{(X,Y)}$ is $|X|(|X|-1)|Y|(|Y|-1)/4$. In order to compare the support for two bipartitions, the distance function must be normalized. We define the *normalized distance* from Q to (X,Y) by

$$\delta(Q,(X,Y)) = \frac{4|Q_{(X,Y)} - Q|}{|X|(|X|-1)|Y|(|Y|-1)}.$$

When (X,Y) is *trivial* ($|X| = 1$ or $|Y| = 1$), the normalized distance is defined to be 0. We will be using δ as our bipartition support function: the bipartitions with smaller distances from Q are those better supported by Q.

1.2 Interesting Neighborhoods of Q

Using the normalize distance δ defined above, we can implicitly define a bipartition list that orders all bipartitions of S by increasing distance from Q. Assuming that bipartitions of T are well–supported by Q (a claim supported by the simulation study presented in Section 3) and appear near the start of the list, the task is to generate a bipartition neighborhood of Q of the form

$$\{(X,Y) \mid \delta(Q,(X,Y)) \leq r\}$$

which is called the closed r–neighborhood of Q. When the inequality is strict it is called the open r–neighborhood of Q.

When $r = 0$ the closed r–neighborhood of Q corresponds to those bipartitions that have 0 quartet topology differences with Q. There is an $O(n^4)$ time algorithm, called the Q^* method, for recovering this set of bipartitions [6]. However, the Q^* tree is a very conservative estimate of T since it includes only those bipartitions with 0 quartet topology difference with Q. We must search for bipartitions that are a greater distance from Q.

When $r = \frac{2}{|X||Y|}$ the open r–neighborhood of Q is compatible [7]. An algorithm that constructs this r–neighborhood is called a *local edge cleaning* algorithm. In this paper we provide an $O(n^5)$ time local edge cleaning algorithm, the first polynomial time algorithm for this problem. We observe that the closed r–neighborhood of Q is not necessarily compatible [19].

Although a local cleaning algorithm is guaranteed to return a set of compatible bipartitions, it is not guaranteed to return all $n-3$ compatible, nontrivial bipartitions of the underlying evolutionary tree T. In fact, the simulation study presented in Section 3 demonstrates that local edge cleaning often does not obtain all bipartitions of T. We widen the search by introducing a parameter $m > 0$ and defining

$$Best(Q,m) \quad = \quad \{(X,Y) \mid \delta(Q,(X,Y)) < \frac{2m}{|X||Y|}\}.$$

Thus the set $Best(Q,m')$ contains the set $Best(Q,m)$ for all $m' \geq m$. Note that $Best(Q,1)$ is the set obtained by local edge cleaning and the limit of $Best(Q,m)$ as m tends to zero is the set of bipartitions in the Q^* tree.

An algorithm that constructs the set $Best(Q,m)$ is called a *hypercleaning* algorithm[1]. In this paper we give a hypercleaning algorithm that takes $O(n^5 f(2m) + n^7 f(m))$ time where $f(m) = 4m^2(1+2m)^{4m}$. For bounded values of m, the hypercleaning algorithm runs in polynomial time making it fixed parameter tractable. The simulation study presented in Section 3 indicates that small values of m are sufficient to recover all bipartitions of T. In particular, we have the following accuracy guarantee:

THEOREM 1.1. *The hypercleaning algorithm recovers all bipartitions (X,Y) in the underlying evolutionary tree T with fewer than $m(|X|-1)(|Y|-1)/2$ quartet errors.*

Hence, if Q is reasonably correlated to T then hypercleaning is a powerful tool for estimating evolutionary trees.

1.3 Constructing trees from strongly supported bipartitions

The focus of this paper is the efficient generation of $Best(Q,m)$, the set of bipartitions strongly supported by Q. However, $Best(Q,m)$ may contain incompatible bipartitions. How can a tree be obtained from this set? Several options are considered here.

The first is a simple greedy algorithm, that attempts to produce a tree that is as resolved as possible and that maximizes the agreement with the input set Q. More precisely, it tries to select a maximal set of compatible bipartitions (X,Y), with the smallest distance to Q, *i.e.*, minimizing $\sum_{(X,Y)} \delta(Q,(X,Y))$. Let (X_1,Y_1), (X_2,Y_2), ..., (X_k,Y_k) be the bipartitions in $Best(Q,m)$ ordered by increasing normalized distance to Q. The greedy algorithm selects the following subset, called $Comp(Q,m)$, of $Best(Q,m)$:

- $(X_1,Y_1) \quad \in \quad Comp(Q,m)$
- $(X_j,Y_j) \quad \in \quad Comp(Q,m)$ if (X_j,Y_j) is compatible with all $(X_i,Y_i) \in Comp(Q,m)$ where $i < j$.

[1] The term "hyper" indicates that m can take on values greater than 1.

Observe that $Comp(Q,m)$ is a set of compatible bipartitions and can be easily obtained from $Best(Q,m)$. The simulation study of Section 3 demonstrates that this simple algorithm obtains systematically and significantly better results than the local cleaning algorithm even when $m = 2$ illustrating the potential of hypercleaning.

Note that the above simple algorithm is only a heuristic to select the maximal set of compatible bipartitions that minimizes the sum of normalized distances to Q. For other criteria there are exact polynomial time algorithms. For example, having inferred a set $Best(Q,m)$ with enough edges to construct a tree, we can aim at selecting the maximal set of compatible bipartitions (X,Y) minimizing

$$max_{(X,Y)}\{\delta(Q,(X,Y))\},$$

which is the L_∞ norm on bipartitions. For this criterion, we can use the exact polynomial time algorithm of [9] which runs in $O(|Best(Q,m)|^2)$ time.

Other criteria for which we have no exact polynomial time algorithms also make sense. For example, we may consider selecting the maximal compatible subset T of bipartitions in $Best(Q,m)$ that minimizes $|Q_T - Q|$. This criterion differs from the one used by the greedy heuristic presented above in the sense that each incorrect quartet topology inferred is here accounted for only once and not with every bipartition (X,Y) by which it is induced. This criterion is also strongly related to the principle of the well-known $O(n^4)$ time quartet puzzling heuristic [25] (whose elementary step aims at producing a tree minimizing the number of quartets of Q it contradicts), since in a fully resolved tree every incorrect quartet inferred implies a correct quartet not recovered. However, hypercleaning has an advantage over quartet puzzling in that hypercleaning preselects the set $Best(Q,m)$ of bipartitions to which we can confidently restrict ourselves (cf simulation results). When removing the maximal resolution constraint, *i.e.* not seeking a tree as resolved as possible, the $O(n^3+|Q|)$ time heuristic and bootstrap procedure of [3] can be consulted.

1.4 Summary of Results

In this paper we present an $O(n^5)$ time local edge cleaning algorithm. This is the first polynomial time algorithm for this problem [7].

A more powerful and flexible technique than local edge cleaning, called hypercleaning, is introduced. We give a polynomial bound on the number of bipartitions inferred by the hypercleaning technique, namely $O(n^3 f(2m))$, where $f(m) = 4m^2(1+2m)^{4m}$.

We present an $O(n^5 f(2m)+n^7 f(m))$ time hypercleaning algorithm. When m is bounded this yields a polynomial time algorithm.

We present a simulation study showing the significant increase in accuracy due to hypercleaning (even with $m = 2$) over the local edge cleaning technique. However, due to its efficiency, local edge cleaning remains useful for processing large data sets. Results also show that hypercleaning combined with a greedy bipartition selection algorithm almost always recovers the underlying evolutionary tree.

2 The Hypercleaning Algorithm

Let $S = \{s_1, s_2, \ldots, s_n\}$, $S_k = \{s_1, s_2, \ldots, s_k\}$, and Q_k be the subset of Q induced by S_k. The hypercleaning algorithm proceeds by first computing sets of the form

$$Best_{xy}(Q_k,m) = \{ (X,Y) \mid x \in X, y \in Y, \text{ there are fewer than } m \text{ quartet errors across } (X,Y) \text{ involving } x \text{ and } y\}$$

for all $x, y \in S$ and $1 \le k \le n$ in time $O(n^5 f(m))$ where $f(m) = 4m^2(1+2m)^{4m}$. These sets are then combined to form the sets

$$Best(Q_k,m) = \{(X,Y) | \delta(Q,(X,Y)) < 2m/(|X||Y|)\}$$

for all $1 \le k \le n$ in $O(n^5 f(2m)+n^7 f(m))$ time. Observe that $Best(Q,m) = Best(Q_n,m)$. Hence, the time required to construct is $O(n^5 f(2m)+n^7 f(m))$. We note that, unlike quartet puzzling [25], the resulting set of bipartitions does not depend on the ordering $s_1, s_2, \ldots, s_n$.

2.1 Computing $Best_{xy}(Q_k,m)$

Let x and y be two sequences in S and $1 \le k \le n$. We define a recurrence for $Best_{xy}(Q_k,m)$ by the following theorem:

THEOREM 2.1. *If $k = 1$ then $Best_{xy}(Q_k,m) = \emptyset$. If $k \ge 2$ then*

$$Best_{xy}(Q_k,m) \subseteq L_{xy} \cup R_{xy} \cup M_{xy}$$

where

$$\begin{aligned} L_{xy} &= \{(X \cup \{s_k\}, Y) \mid (X,Y) \in Best_{xy}(Q_{k-1},m)\} \\ R_{xy} &= \{(X, Y \cup \{s_k\}) \mid (X,Y) \in Best_{xy}(Q_{k-1},m)\} \\ M_{xy} &= \{(\{s_k\}, S_{k-1})\} \end{aligned}$$

Proof. When $k = 1$ there are not enough sequences to form a bipartition, and so, $Best_{xy}(Q_k,m) = \emptyset$. Suppose $k \ge 2$ and $(X \cup \{s_k\}, Y) \in Best_{xy}(Q_k,m)$.

If $(X \cup \{s_k\}, Y)$ is trivial then either $Y = \{s_i\}$ for $i < k$ or $X = \emptyset$. In the former case $(X, Y) \in Best_{xy}(Q_{k-1}, m)$. In the latter case $(X \cup \{s_k\}, Y) \in M_{xy}$. Suppose $(X \cup \{s_k\}, Y)$ is not trivial. Since $(X \cup \{s_k\}, Y) \in Best_{xy}(Q_k, m)$ there are fewer than m quartet errors across $(X \cup \{s_k\}, Y)$ involving x and y. It follows that there are fewer than m errors across (X, Y) involving x and y, and so, $(X, Y) \in Best_{xy}(Q_{k-1}, m)$. ■

The algorithm for computing $Best_{xy}(Q_k, m)$ for all $x, y \in S$ and $1 \leq k \leq n$ follows from Theorem 2.1: For all $x, y \in S$, for k ranging from 1 to n, and for all $(X, Y) \in L_{xy} \cup R_{xy} \cup M_{xy}$, place (X, Y) in $Best_{xy}(Q_k, m)$ if there are fewer than m quartet errors across (X, Y) involving x and y.

In order to analyze the complexity of this algorithm a bound on the size of each set $Best_{xy}(Q_k, m)$ is obtained. To begin, define

$$f(m) = 4m^2(1 + 2m)^{4m}.$$

THEOREM 2.2. *The number of bipartitions in $Best_{xy}(Q_k, m)$, for any $1 \leq k \leq n$, is $O(nf(m))$.*

Proof. Let (X, Y) be a bipartition in $Best_{xy}(Q_k, m)$ with $x \in X$ and $y \in Y$. We first bound the number of bipartitions $(X', Y') \in Best_{xy}(Q, m)$ with $x \in X'$ and $y \in Y'$ that are incompatible with (X, Y). There exists nonempty sets $A = X \cap Y'$ and $B = X' \cap Y$ since (X, Y) and (X', Y') are not compatible.

Let G be a directed graph with vertex set X and for each $u, v \in X$, (u, v) is an edge of G if $ux|vy \in Q$. Notice that an edge from $a \in A$ to $v \in X - A$ indicates that $\{a, v, x, y\}$ is a quartet error across (X', Y') involving x and y since $v, x \in X'$ and $a, y \in Y'$. It follows that the number of edges from vertices in A to vertices in $X - A$ is less than m.

Analogously, a directed graph can be constructed on vertex set Y with the result that the number of edges from vertices in $Y - B$ to B is less than m.

Notice that, for each $a \in A$ and $b \in B$, $\{a, b, x, y\}$ is a quartet error across either (X, Y) or (X', Y'). It follows that $|A||B| < 2m$. Consequently, $|A| < 2m$ and $|B| < 2m$.

The number of bipartitions (X', Y') incompatible with (X, Y) is bounded by the number of choices for $A \subseteq X$ such that $|A| < 2m$ and there are less than m edges from A to $X - A$ in the above graph with vertex set X. Similarly, the number of choices for $B \subseteq Y$ such that $|B| < 2m$ and there are less than m edges from $Y - B$ to B in the above graph with vertex set Y. We use the following lemma to give us this bound where for a directed graph G, $\text{in}(v)$ and $\text{out}(v)$ denote the out-degree and in-degree of a vertex. Similarly, let $\text{in}(V')$ and $\text{out}(V')$ denote the number of edges leaving or entering a subset V' of the vertex set.

LEMMA 2.1. *Let $G = (V, E)$ be a tournament. There are at most $(k + 2m/k)^k$ subsets V' of V such that $|V'| = k$ and $in(V') \leq m$. Thus, by symmetry, there are at most $(k + 2m/k)^k$ subsets V' of V such that $|V'| = k$ and $out(V') \leq m$.*

Proof. Omitted. ■

When $a = |A|$ and $b = |B|$, the number of bipartitions (X', Y') is bounded by $(a + 2m/a)^a(b + 2m/b)^b$. Summing over all set sizes we have the bound

$$\begin{aligned} & \sum_{a=1}^{2m} \sum_{b=1}^{2m} (a + 2m/a)^a (b + 2m/b)^b \\ \leq \; & 2m(1 + 2m)^{2m} \sum_{a=1}^{2m} (a + 2m/a)^a \\ \leq \; & 4m^2(1 + 2m)^{4m} \end{aligned}$$

Now let $(X_1, Y_1), (X_2, Y_2), \ldots, (X_h, Y_h)$ be a maximal compatible collection of bipartitions in $Best_{xy}(Q, m)$. Every other bipartition in $Best_{xy}(Q, m)$ is incompatible with at least one (X_i, Y_i). Since $h = O(n)$ we have that $|Best_{xy}(Q, m)|$ is $O(nf(m))$. ■

There are $O(n^3)$ sets $Best_{xy}(Q_k, m)$ to construct. Constructing each set $Best_{xy}(Q_k, m)$ takes time proportional to the size of $Best_{xy}(Q_{k-1}, m)$ times the complexity of testing if a bipartition has fewer than m quartet errors involving x and y. Without loss of generality suppose the bipartition is $(X \cup \{s_k\}, Y)$ and is not trivial (trivial bipartitions have no quartet errors). It follows from Theorem 2.1 that $(X, Y) \in Best_{xy}(Q_{k-1}, m)$. At this point assume that on the previous iteration we have counted and stored the number of quartet errors across (X, Y) involving x and y. It then suffices to count the number of quartet errors across (X, Y) involving x, y and s_k. There are $O(n)$ of these quartets to examine. It follows that the complexity of constructing all sets $Best_{xy}(Q_k, m)$ is $O(n^5 f(m))$.

2.2 Computing $Best(Q_k, m)$

We define a recurrence for $Best(Q_k, m)$ by the following theorem:

THEOREM 2.3. *If $k = 1$ then $Best(Q_k, m) = \emptyset$. If $k \geq 2$ then*

$$Best(Q_k, m) \subseteq L \cup R \cup M$$

where

$$\begin{aligned} L &= \{(X \cup \{s_k\}, Y) : (X,Y) \in Best(Q_{k-1}, m)\} \\ R &= \{(X, Y \cup \{s_k\}) : (X,Y) \in Best(Q_{k-1}, m)\} \\ M &= \cup_{x \in S_{k-1}} Best_{xs_k}(Q_k, m) \end{aligned}$$

Proof. Suppose $(X \cup \{s_k\}, Y) \in Best(Q_k, m)$ but $(X,Y) \notin Best(Q_{k-1}, m)$. It follows that there are less than $m|X|(|Y|-1)/2$ quartet errors across $(X \cup \{s_k\}, Y)$ but at least $m(|X|-1)(|Y|-1)/2$ quartet errors across (X,Y). Let s_{in} denote the number of quartet errors across $(X \cup \{s_k\}, Y)$ involving s_k and let s_{out} denote the number of quartet errors across $(X \cup \{s_k\}, Y)$ not involving s_k:

$$\begin{aligned} s_{\text{in}} + s_{\text{out}} &< \frac{m|X|(|Y|-1)}{2} \\ s_{\text{out}} &\geq \frac{m(|X|-1)(|Y|-1)}{2} \\ s_{\text{in}} &< \frac{m(|Y|-1)}{2} \end{aligned}$$

A key to the design of the hypercleaning algorithm is that quartet errors are relatively sparse for small m: there are $|X|(|X|-1)|Y|(|Y|-1)/4$ quartets across $e = (X,Y)$ but only $m(|X|-1)(|Y|-1)/2$ quartet errors. This ratio suggests that we consider the average number of quartet errors across e involving a pair of labels $x \in X$ and $y \in Y$.

Each quartet error across $(X \cup \{s_k\}, Y)$ involving s_k also involves two elements y_1 and y_2 from Y. Hence, the average number of quartet errors across $(X \cup \{s_k\}, Y)$ involving s_k per element of Y is less than

$$2m(|Y|-1)/2|Y| < m.$$

This implies that there exists $y \in Y$ such that s_k and y are involved in less than m quartet errors across $(X \cup \{s_k\}, Y)$. In particular, $(X \cup \{s_k\}, Y) \in Best_{ys-k}(Q_k, m)$ for some $y \in S_{k-1}$. ■

The algorithm for computing $Best(Q_k, m)$, for $1 \leq k \leq n$ follows from Theorem 2.3: For k ranging from 1 to n, and for all $(X,Y) \in L \cup R \cup M$, place (X,Y) in $Best(Q_k, m)$ if there are fewer than $m(|X|-1)(|Y|-1)/2$ quartet errors across (X,Y).

In order to analyze the complexity of this algorithm a bound on the size of each set $Best(Q_k, m)$ is obtained. Let f be defined as in Section 2.1.

THEOREM 2.4. *The number of bipartitions in $Best(Q_k, m)$, for any $1 \leq k \leq n$, is $O(n^3 f(2m))$.*

Proof. Let $(X,Y) \in Best(Q_k, m)$. Each quartet error across (X,Y) involves 4 pairs of the form $x \in X$ and $y \in Y$. Suppose every pair $x \in X$ and $y \in Y$ is involved in at least $2m$ quartet errors across (X,Y). This requires at least $2m|X||Y|/4 = m|X||Y|/2$ quartet errors across (X,Y). However, this is a contradiction since $(X,Y) \in Best(Q_k, m)$. We conclude that $(X,Y) \in Best_{xy}(Q_k, 2m)$ for some pair $x \in X$ and $y \in Y$. This yields the following:

$$|Best(Q_k, m)| < \sum_{x,y \in S} |Best_{xy}(Q_k, 2m)|$$

By Theorem 2.2 it follows that $Best(Q_k, m)$ is $O(n^3 f(2m))$. ■

There are $O(n)$ sets $Best(Q_k, m)$ to construct. If $(X,Y) \in Best(Q_{k-1}, m)$ then to determine if $(X \cup \{s_k\}, Y) \in Best(Q_k, m)$ (or to determine if $(X, Y \cup \{s_k\}) \in Best(Q_k, m)$) we can assume that on the previous iteration we have counted and stored the number of quartet errors across (X,Y). To this we add the number of quartet errors across $(X \cup \{s_k\}, Y)$ that involve s_k. There are $O(n^3)$ of these quartets to examine. Finally, there are $|Best(Q_{k-1}, m)| = O(nf(2m))$ bipartitions (X,Y). Otherwise, if $(X,Y) \in Best_{xs_k}(Q_k, m)$ then we can use an adaption of the path covering algorithms of [5, 6] in $O(n^4)$ time. Finally, there are $|Best_{xs_k}(Q_k, m)| = O(nf(m))$ bipartitions (X,Y) and x takes on $O(n)$ values. This results in the time complexity of $O(n^5 f(2m) + n^7 f(m))$.

The overall complexity to compute $Best(Q, m)$ is $O(n^5 f(m) + n^5 f(2m) + n^7 f(m)) = O(n^5 f(2m) + n^7 f(m))$. This establishes the problem of determining $Best(Q, m)$ as fixed parameter tractable [12]. Like other fixed parameter tractable problems, we expect more efficient implementations to be achieved.

2.3 Local edge cleaning

To improve the complexity of the hypercleaning algorithm when $m = 1$ we introduce an intermediary set of bipartitions. Let x be a sequence in S and define the set of bipartitions $Best_x(Q, 1) = \{(X,Y) \mid x \in X$ and there are less than $(|Y|-1)/2$ quartet errors across (X,Y) involving $x\}$. Then $Best_x(Q, 1)$ is compatible, and can be constructed in $O(n^4)$ time by first computing sets of the form $Best_{xy}(Q_k, 1)$, for all $x, y \in S$ and $1 \leq k \leq n$ and then computing sets of the form $Best_x(Q_k)$, for all $x \in S$ and $1 \leq k \leq n$ using the following recurrence:

THEOREM 2.5. *If $k = 2$ then $Best_x(Q_k, 1) = \{(\{x\}, \{y\})\}$. When $k \geq 3$ we have*

$$Best_x(Q_k, 1) \subseteq L_x \cup R_x \cup M_x$$

where

$$\begin{aligned} L_x &= \{(X \cup \{s_k\}, Y) : (X, Y) \in Best_x(Q_{k-1}, 1)\} \\ R_x &= \{(X, Y \cup \{s_k\}) : (X, Y) \in Best_x(Q_{k-1}, 1)\} \\ M_x &= Best_{xs_k}(Q_k, 1) \end{aligned}$$

Proof. Omitted. ■

Finally, $Best(Q_k, 1)$ can be constructed for $1 \leq k \leq n$ using the following recurrence in $O(n^5)$ time:

THEOREM 2.6. *If* $k = 2$ *then* $Best(Q_k, 1) = \{(\{x\}, \{y\})\}$. *When* $k \geq 3$ *we have*

$$Best(Q_k, 1) \subseteq L' \cup R' \cup M'$$

where

$$\begin{aligned} L' &= \{(X \cup \{s_k\}, Y) : (X, Y) \in Best(Q_{k-1}, 1)\} \\ R' &= \{(X, Y \cup \{s_k\}) : (X, Y) \in Best(Q_{k-1}, 1)\} \\ M' &= Best_{s_k}(Q_k, 1) \end{aligned}$$

Proof. Omitted. ■

3 Simulation Results

The simulation study was designed to address the following questions:

- Does hypercleaning obtain more edges of the unknown evolutionary tree than local edge cleaning?
- How large must m be so that $Best(Q, m)$ contains all edges of the unknown evolutionary tree?

To answer these questions, DNA sequences were artificially evolved using the Kimura 2 parameter model of evolution [26] with transition/transversion ratio of 5 : 1 on an evolutionary tree T sampled from the Ribosomal Database Project prokaryotic tree [21]. This tree T represents the evolutionary history of the TRP saccharophilum subgroup and contains 10 leaves. The tree appears in Figure 3. Site–to–site rate variance was simulated using the gamma function with parameter 1.

To better explore the parameter space, sequence length and edge length were varied. More specifically, the sequence length was varied over values 100, 200, 500 and 1000. The evolutionary tree T was scaled by factors 0.25, 0.5, 1.0 and 2.0 so that trees with recently diverged sequences and trees with distantly diverged sequences were examined. For each set of sequences generated, a quartet set Q was obtained by applying the ordinal quartet method[20] to the sequences. The hypercleaning algorithm was then applied to Q to obtain $Best(Q, 0)$, $Best(Q, 1)$, ..., $Best(Q, 5)$. Similarly, the sets $Comp(Q, 0)$, $Comp(Q, 1)$, ..., $Comp(Q, 5)$ were obtained.

The results appear in Table 1. Each cell of the table is indexed by an edge length scaling factor, a value of m between 0 and 5 and a sequence length. Each cell contains four values computed as the average over 200 trials.

- The first value is the percentage of the nontrivial edges in T that appear in $Best(Q, m)$. It is clear from the results that as m increases $Best(Q, m)$ approaches 100%. In particular, $Best(Q, 5)$ almost always contains every nontrivial edge of T. Notice that $Best(Q, 3)$ approaches 100% for sequence lengths greater than 200. This clearly demonstrates that hypercleaning is effective for small values of m.
- The second value is the number of bipartitions in $Best(Q, m)$ expressed as a percentage of the number of nontrivial edges in T. The ratio of the first value to the second value is the density of nontrivial edges of T in $Best(Q, m)$. As m increases it is clear that this density decreases rapidly supporting the assumption that edges of T cluster near the beginning of the bipartition list defined by the support function d.
- The third value is the percentage of the nontrivial edges in T that appear in $Comp(Q, m)$. The third value measures the accuracy of the local cleaning algorithm ($m = 1$) and of the greedy algorithm ($m > 1$) that selects edges of T from $Best(Q, m)$. In all cases, there is a significant improve of hypercleaning with only $m = 2$ over local cleaning (*i.e.*, when $m = 1$). However, it's lower time complexity makes local cleaning still useful for dealing with larger data sets.

  For moderate sequence lengths of 200 to 500, $Comp(Q, m)$ contains approximately 85% of T's nontrivial edges. Since T has 7 nontrivial edges this implies that $Comp(Q, m)$, on average, contains all edges of T except 1. The model tree T used in the simulation contains a very short edges that is difficult to recover. For a less challenging tree it is expected that accuracy will only improve. Note that as sequence length increases so does accuracy. This is to be expected since as sequence length increases so does the amount of information available for recovering edges of T.

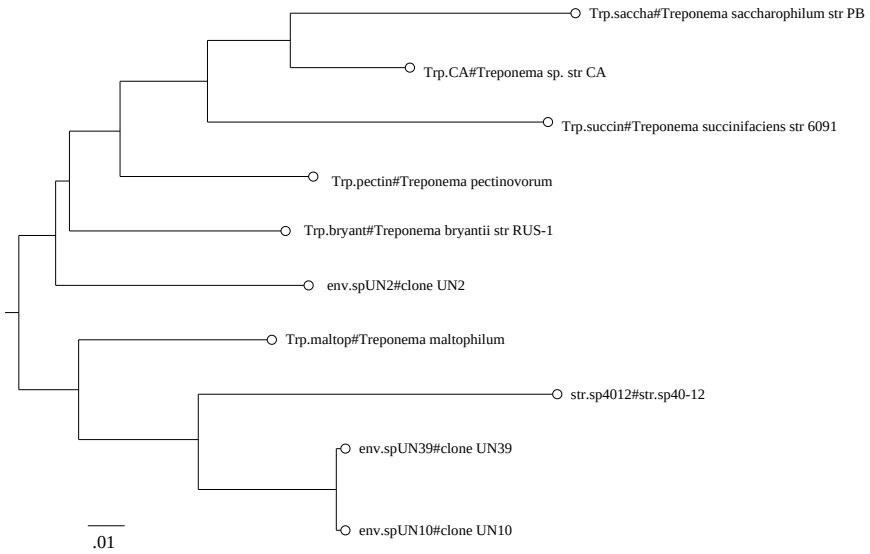

Figure 3: The TRP saccharophilum subgroup tree.

The values for $m = 1$ indicate the number of nontrivial edges of T obtained by local edge cleaning. Comparing these to the $m = 5$ values it is clear that hypercleaning in conjunction with the greedy algorithm is much more effective at obtaining nontrivial edges of T.

- The fourth value is the number of bipartitions in $Comp(Q, m)$ expressed as a percentage of the number of nontrivial edges in T. This provides a measure of the degree of resolution of the tree inferred by the greedy heuristic. In all cases, $Comp(Q, 2)$ gives an almost fully resolved tree (in some cases the tree is already fully resolved), which explains that few more improvements are observed when resorting to higher values of m, *i.e.* going from $m = 2$ to $m = 5$.

  In the study, $Best(Q, 5)$ most often contains all nontrivial edges of T and $Comp(Q, 5)$ is 100%. However, $Comp(Q, 5)$ does not contain all nontrivial edges of T. This indicates that for some bipartition (X', Y') not in T and some bipartition (X, Y) in T, that $\delta(Q, (X', Y')) < \delta(Q, (X, Y))$. Consequently, the greedy algorithm is fooled since an incorrect bipartition is ranked ahead of a correct edge of T. Future research includes the development of less simplistic algorithms for selecting a set of compatible bipartitions from $Best(Q, m)$.

References

[1] H. J. Bandelt and A. Dress. Reconstructing the shape of a tree from observed dissimilarity data. *Advances in Applied Mathematics*, 7:309–343, 1986.

[2] A. Ben-Dor, B. Chor, D. Graur, R. Ophir, and D. Pelleg. From four–taxon trees to phylogenies: The case of mammalian evolution. *Proceedings of the Second Annual International Conference on Computational Molecular Biology*, pages 9–19, 1998.

[3] V. Berry., G. Gascuel and G. Caraux Choosing the tree which best explains the data: another look at the bootstrap in phylogenetic reconstruction. *Computational Statistics and Data Analysis*, to appear, 2000.

[4] V. Berry. Phyloquart v1.3 - quartet programs for phylogeny reconstruction. Technical report RR 99150, LIRMM, 1999.

[5] V. Berry and D. Bryant. Faster reliable phylogenetic analysis. *Proceedings of the Third Annual International Conference on Computational Molecular Biology*, pages 59–68, 1999.

[6] V. Berry and O. Gascuel. Inferring evolutionary trees with strong combinatorial evidence. *Proceedings of the Third Annual International Computing and Combinatorics Conference*, pages 111–123, 1997.

[7] V. Berry, T. Jiang, P. Kearney, M. Li, and T. Wareham. Quartet cleaning: Improved algorithms and simulations. In *European Symposium on Algorithms*, 1999.

[8] M. Boguski. Bioinformatics – a new era. In S. Brenner, F. Lewitter, M. Patterson, and M. Handel, editors, *Trends Guide to Bioinformatics*, pages 1–3. Elsevier Science, 1998.

[9] D. Bryant. *Structures in Biological Classification.* Ph.D. Thesis, Department of Mathematics and

Edge Length Scaling Factor	m	Sequence Length 100		200		500		1000	
0.5	0	28 (31)	28 (31)	35 (37)	35 (37)	50 (51)	50 (51)	63 (64)	63 (64)
0.5	1	50 (64)	50 (64)	62 (73)	62 (73)	82 (87)	82 (87)	89 (93)	89 (93)
0.5	2	68 (102)	63 (87)	80 (112)	74 (93)	92 (119)	88 (98)	97 (115)	94 (100)
0.5	3	84 (216)	65 (98)	93 (231)	76 (100)	98 (238)	89 (100)	100 (237)	94 (100)
0.5	4	92 (387)	66 (99)	98 (415)	76 (100)	100 (439)	89 (100)	100 (450)	94 (100)
0.5	5	96 (666)	66 (100)	99 (703)	76 (100)	100 (725)	89 (100)	100 (733)	94 (100)
1.0	0	28 (29)	28 (29)	35 (37)	35 (37)	51 (52)	51 (52)	61 (61)	61 (61)
1.0	1	57 (66)	57 (66)	68 (76)	68 (76)	83 (87)	83 (87)	90 (92)	90 (92)
1.0	2	74 (109)	70 (92)	84 (117)	78 (96)	95 (118)	90 (99)	97 (116)	95 (100)
1.0	3	87 (225)	73 (100)	95 (233)	80 (100)	99 (238)	91 (100)	100 (237)	95 (100)
1.0	4	95 (408)	73 (100)	99 (417)	80 (100)	100 (438)	91 (100)	100 (448)	95 (100)
1.0	5	98 (700)	73 (100)	100 (715)	80 (100)	100 (724)	91 (100)	100 (727)	95 (100)
2.0	0	24 (25)	24 (25)	30 (31)	30 (31)	46 (47)	46 (47)	63 (63)	63 (63)
2.0	1	50 (60)	50 (60)	68 (76)	68 (76)	82 (86)	82 (86)	92 (93)	92 (93)
2.0	2	69 (107)	63 (89)	85 (118)	78 (96)	94 (120)	90 (99)	99 (114)	96 (100)
2.0	3	88 (227)	67 (99)	96 (234)	80 (100)	100 (239)	90 (100)	100 (234)	96 (100)
2.0	4	96 (399)	68 (100)	99 (414)	80 (100)	100 (431)	90 (100)	100 (453)	96 (100)
2.0	5	99 (704)	68 (100)	100 (711)	80 (100)	100 (715)	90 (100)	100 (727)	96 (100)

Table 1: Summary of the Simulation Study Results

Statistics, University of Canterbury, 1997.

[10] D. Bryant and M. Steel. Fast algorithms for constructing optimal trees from quartets. *Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 147–155, 1999.

[11] P. Buneman. The recovery of trees from measures of dissimilarity. In F.R. Hodson, D.G. Kendall, and P. Tautu, editors, *Mathematics in the Archaeological and Historical Sciences*, pages 387–395. Edinburgh University Press, Edinburgh, 1971.

[12] P. Downey and M. Fellows, *Parameterized Complexity*, Springer-Verlag, 1998.

[13] P. Erdös, M. Steel, L. Székely, and T. Warnow. Constructing big trees from short sequences. *Proceedings of the 24th International Colloquium on Automata, Languages, and Programming*, 1997.

[14] J. Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.

[15] W. M. Fitch. Toward defining the course of evolution: Minimal change for a specific tree topology. *Systematic Zoology*, 20:406–41, 1971.

[16] V. A. Funk and D. R. Brooks. *Phylogenetic Systematics as the Basis for Comparative Biology*. Smithsonian Institution Press, 1990.

[17] R. Gupta and B. Golding. Evolution of HSP70 gene and its implications regarding relationships between Archaebacteria, Eubacteria and Eukaryotes. *Journal of Molecular Evolution*, 37:573–582, 1993.

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

[19] T. Jiang, P. E. Kearney, and M. Li. Orchestrating quartets: Approximation and data correction. *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 416–425, 1998.

[20] P. E. Kearney. The ordinal quartet method. *Proceedings of the Second Annual International Conference on Computational Molecular Biology*, pages 125–134, 1998.

[21] Bonnie L. Maidak, James R. Cole, Charles T. Parker, George M. Garrity Jr, Niels Larsen, Bing Li, Timothy G. Lilburn, Michael J. McCaughey, Gary J. Olsen, Ross Overbeek, Sakti Pramanik, Thomas M. Schmidt, James M. Tiedje, and Carl R. Woese. A new version of the RDP (Ribosomal Database Project). *Nucleic Acids Research*, 27:171–173, 1999.

[22] C. A. Meacham and G. F. Estabrook. Compatibility methods in systematics. *Ann. Rev. Ecol. Syst.*, 16:431–446, 1985.

[23] N. Saitou and M. Nei. The neighbor–joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.

[24] S. Sattath and A. Tversky. Additive similarity trees. *Psychometrika*, 42:319–345, 1977.

[25] K. Strimmer and A. von Haeseler. Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies. *Molecular Biology and Evolution*, 13(7):964–969, 1996.

[26] D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M Hillis. Phylogenetic inference. In D. M. Hillis, C. Moritz, and B. K. Mable, editors, *Molecular Systematics, 2nd Edition*, pages 407–514. Sinauer

Associates, Sunderland, Massachusetts, 1996.

[27] S. Willson. Measuring inconsistency in phylogenetic trees. *Journal of Theoretical Biology*, 190:15–36, 1998.