FASTER SOLUTION TO THE MAXIMUM QUARTET CONSISTENCY PROBLEM WITH CONSTRAINT PROGRAMMING *

GANG WU, GUOHUI LIN †, JIA-HUAI YOU, AND XIAOMENG WU

Department of Computing Science, University of Slberta. Edmonton, Alberta T6G 2E8, Canada.
Email: wgang,ghlin,you,xiaomeng@cs.ualberta.ca.

Evolution is an important sub-area of study in biological science, whereby the evolutionary history, or phylogeny, would shed light on the genetic linkage and the functional correlation for the species under consideration. Many kinds of species data can be deployed for the task and many phylogeny reconstruction methods have been examined in the literature. A quartet approach is to build a local phylogeny for every 4 species, which is called a quartet for these 4 species, and then to assemble a phylogeny for the whole set of species satisfying the topological constraints imposed by these quartets built. In practice, those predicted quartets might not agree each other and the optimization problem, the well-known Maximum Quartet Consistency (MQC) problem, is to construct a phylogeny to satisfy a maximum number of the predicted quartets. An equivalent representation for the MQC problem through searching for a certain ultrametric matrix via Answer Set Programming has recently been proposed. This paper follows the approach and presents a number of optimization techniques to speed up the searching process. The experimental results on both the simulated and real datasets suggest that the new representation combined with Constraint Programming presents a unique perspective to the MQC problem.

1. Introduction

A fundamental problem in computational biology is to retrieve the history of a set of species by reconstructing their evolutionary tree. Such a tree, also called a *phylogeny*, has its leaves labeled with the given species, while the internal nodes represent extinct or hypothesized ancestors. If the phylogeny is rooted, then its root represents a common ancestor of all the species. Species data used to reconstruct a phylogeny often consist of DNA or protein sequences, besides their morphological characteristics. In many cases, the huge amount of genomic data limit the number of species that can be analyzed at one time. In the last two decades, quartet based methods for reconstructing phylogenies have received a considerable amount of attention in the computational biology community.[9,12] The quartet-based phylogeny reconstruction is to first build a subtree of phylogeny for every subset of 4 species (*quartet*) and then rely on some combinatorial algorithms to construct a phylogeny on the entire set of species. Such methods are based on the principle that constructing small phylogenies is easier, and often more reliable as they allow for more intensive analysis. Tree criteria like maximum likelihood, which are computationally horrendous on larger

*This work is supported by NSERC grants and CFI.
†To whom correspondence should be addressed. Fax: (780) 492-1071. Email: ghlin@cs.ualberta.ca.

trees, can be solved exactly on quartets (note that there are only three possible resolved trees to consider for every 4 species).

In the ideal case where all quartets are "correct", the task of assembling an overall phylogeny is easy and can be done in $O(n^5)$ time,[6] where n is the number of species under consideration. In practice, however, some quartets might be ambiguous (and thus missing) or even erroneous. Therefore, the set of quartets might be incomplete and might contain conflicting quartets. These properties complicate the overall phylogeny construction but also raise the computational interest. The parsimony goal is to construct a phylogeny which respects as many quartets as possible. However, such an optimization problem turns out to be hard.[12] To the best of our knowledge, existing exact algorithms[13] are all of exhaustive search nature, which however is generally infeasible as the size of search space is huge (there are $\frac{(2n-5)!}{(n-3)!\ 2^{n-3}}$ unrooted resolved phylogenies on n leaves to choose from[8]). There are a lot of efforts put on the approximation side. To name a few, the heuristics of Sattath and Tversky[15] and Bandelt and Dress[2] combine some clustering procedures with a pairwise similarity or neighborliness scores derived from the quartets; one novel variation on the scoring approach is described by Ben-Dor et al.,[3] where instead of constructing a similarity score for clustering, they embed those n leaves as points in the n-dimensional Euclidean space R^n using semi-definite programming and then apply a nearest neighbor clustering procedure to finish the task; Dekker[4] proposed another method for constructing phylogenies from quartets and other sub-phylogenies using some quartet inference rules. The *Short Quartet Method* proposed in Erdos et al.[5] constructs phylogenies using some inference rules and greedy selection of quartets.

In next section, we briefly introduce the definitions for a number of objects used in this paper. We then describe the target combinatorial optimization problems. Section 3 gives the outline of the new representation for the phylogeny reconstruction problem to satisfy the maximum number of quartets. For the details, the readers may refer to a preceding paper.[17] In Section 4, we present a number of nice structural properties of the new representation which can be taken advantage of to prune the search space more efficiently. Section 5 presents the experimental results with comparisons made to the phylogenies constructed using Phylogeny Inference Package (PHYLIP).[7] We conclude the paper in Section 6.

2. Problem Descriptions

For a set of 4 species $S = \{s_1, s_2, s_3, s_4\}$, there are three possible resolved quartets. These three quartets are shown in Figure 1. For simplicity, we use $[12 \mid 34]$ to denote the quartet in which the path connecting s_1 and s_2 does not intersect the path connecting s_3 and s_4 (as shown in Figure 1(a)). Let Q denote the set of quartets built in the first step of a quartet based phylogeny reconstruction, which can be done by various approaches. If there exists one overall phylogeny T such that the quartet $q \in Q$ is the same as the quartet derived from T for the 4 species, then we say T *satisfies* q or q is *consistent* with T;

The *Maximum Quartet Consistency* (MQC) problem can be stated as follows:

INSTANCE: A set Q of quartets on species set $S = \{s_1, s_2, \ldots, s_n\}$.
GOAL: Find a phylogeny T to satisfy a maximum number of quartets in Q.

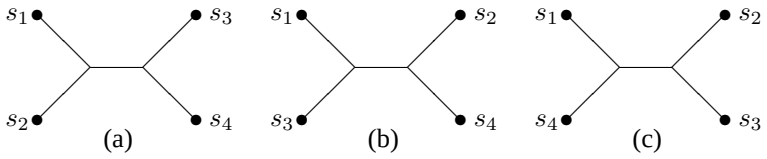

Figure 1. Three possible resolved quartets for subset $\{s_1, s_2, s_3, s_4\}$.

It is known that when Q is complete, MQC is NP-hard but admits a PTAS.[11] When Q isn't complete, MQC is MAX SNP-hard.[11]

Given a rooted phylogeny T on $S = \{s_1, s_2, \ldots, s_n\}$, the *least common ancestor* of two leaf nodes s_i and s_j in T is the common ancestor of s_i and s_j furthest away from the root, denoted as LCA(s_i, s_j). A *labeling scheme* for phylogeny T is a mapping from the set of internal nodes in T to the set of integers $\{1, 2, \ldots, n-1\}$. Note that there are exactly $n-1$ internal nodes in T and each node can be labeled by any number in the set $\{1, 2, \ldots, n-1\}$. Let $M(i, j)$ denote the label of the internal node LCA(s_i, s_j). (Without loss of generality, $M(i, i)$ is set to 0.) Consequently, for two pairs of leaf nodes (s_i, s_j) and (s_k, s_ℓ), we have $M(i, j) = M(k, \ell)$ if and only if LCA(s_i, s_j) = LCA(s_k, s_ℓ). A labeling scheme is *ultrametric* if along any root to leaf path, the labels of the internal nodes on the path is strictly decreasing. One phylogeny together with an ultrametric labeling scheme is called an *ultrametric phylogeny*.

Let M be an $n \times n$ symmetric matrix with its entry values taken from $\{0, 1, 2, \ldots, n-1\}$. M is *ultrametric* if $M(i, i) = 0$ for every i, $M(i, j) > 0$ for every pair $i \neq j$, and for every triplet (i, j, k) there are two equal values among $M(i, j)$, $M(j, k)$, and $M(i, k)$, and they are greater than the third value.

Theorem 2.1. [17] *Given a set of species $S = \{s_1, s_2, \ldots, s_n\}$ and a phylogeny T on S, there exists an ultrametric labeling scheme for T and the resultant matrix M is ultrametric.*

3. Solving MQC via Constraint Programming

The following two theorems tell that constructing a phylogeny to satisfy a maximum number of quartets is equivalent to the search for an ultrametric matrix to satisfy the maximum number of quartets.

Theorem 3.1. [17] *A quartet $[ab \mid cd]$ is consistent with a phylogeny T if and only if any ultrametric labeling scheme M of T satisfies:*

$$\min\{M(a, c), M(b, d)\} > \min\{M(a, b), M(c, d)\}.$$

An $n \times n$ ultrametric matrix M satisfies a quartet $[ab \mid cd]$, or the quartet is *consistent* with M, if $\min\{M(a, c), M(b, d)\} > \min\{M(a, b), M(c, d)\}$ holds.

Theorem 3.2. [17] *Given a set Q of quartets on a set of species $S = \{s_1, s_2, \ldots, s_n\}$ and an ultrametric phylogeny T on S, T satisfies a maximum number of quartets in Q if and only*

if the corresponding ultrametric matrix M on S satisfies the maximum number of quartets in Q.

According to Theorem 3.2, the MQC problem is equivalent to the problem of finding an ultrametric matrix that satisfies the maximum number of quartets, which can be formulated into a *Constraint Programming* problem.[17] In the problem, the input consists of n^2 variables $M(i,j)$ whose domain is $\{0,1,\ldots,n-1\}$, and a set Q of quartets on S. The constraint set contains "symmetry constraints", "ultrametric constraints" $ultra(i,j,k)$, and "quartet constraints" $q(i,j,k,\ell)$ for quartet $[ij \mid k\ell] \in Q$. The goal is to find a solution to the set of variables such that all symmetry and ultrametric constraints are satisfied and a maximum number of quartet constraints are satisfied.

For every triplet (i,j,k) of distinct indices, among $M(i,j)$, $M(j,k)$, and $M(i,k)$ there must be two equal values which is greater than the third value. This is the *ultrametric constraint* involving (i,j,k) and is denoted as $ultra(i,j,k)$. $ultra(i,j,k)$ is satisfied if and only if one of the following three constraints is satisfied:

- $M(i,j) = M(i,k) > M(j,k)$;
- $M(i,j) = M(j,k) > M(i,k)$;
- $M(j,k) = M(i,k) > M(i,j)$.

According to Theorem 3.1, one quartet $[ij \mid k\ell]$ is satisfied if and only if at least one of the following two constraints is satisfied:

- $M(i,k) > M(i,j)$ and $M(j,\ell) > M(i,j)$;
- $M(i,k) > M(k,\ell)$ and $M(j,\ell) > M(k,\ell)$.

This is the *quartet consistency constraint* on $[ij \mid k\ell]$ and is denoted $q(i,j,k,\ell)$.

4. Optimizations

In a constraint programming problem, there are generally two ways to speed up the computation, one is to reduce the number of variables and the other is to reduce the size of the domain for each variable. We present three speedup strategies specific to the problem in the follow three subsections, each of which takes advantage of some structural properties of the optimal phylogeny. Our experimental results show that they all help reduce the running time significantly.

4.1. Breaking the Symmetry

This might not be quartet specific but rather ultrametric matrix specific. The observation is that an ultrametric matrix M is symmetric and therefore instead of putting the symmetry as constraints, we would rather use it to reduce the number of variables. Only $M(i,j)$ with $1 \le i < j \le n$ becomes a variable, which gives only $\frac{1}{2}(n^2-n)$ variables at the end. Consequently, we remove all symmetry constraints from the constraint set. Similarly, we would only consider ultrametric constraints $ultra(i,j,k)$ such that $1 \le i < j < k \le n$ and quartet consistency constraints $q(i,j,k,\ell)$ such that $1 \le i < j \le n$, $1 \le k < \ell \le n$, and $1 \le i < k \le n$.

4.2. Reducing the Number of Species

The implementation of this strategy depends on the quality of the quartet set Q. Observe that in an optimal phylogeny T if two species s_i and s_j are siblings, then any T-induced quartet involving both s_i and s_j must have the form of $[ij \mid **]$. The question we ask is, under what kind of condition, we can infer that in any (or one) optimal phylogeny for Q species s_i and s_j are siblings? The importance of the ability to answer the question is that once we conclude species s_i and s_j to be siblings, we can "merge" them into a super-species, remove all quartets involving both of them, suitably replace quartets involving exactly one of them, and thus reduce the original problem to a problem with one less species. We remark that reducing the number of species by one is about to reduce the computational time by one magnitude.

For a pair of species (s_i, s_j) and a quartet q involving both of them (and two other species), the pair *conflicts* q if q is not in the form of $[ij \mid **]$. For a pair of species (s_i, s_j) and the quartet on $\{s_i, s_a, s_b, s_c\}$, if we change s_i to s_j in the quartet and the resultant quartet is same as the given quartet on $\{s_j, s_a, s_b, s_c\}$, then 4-subsets $\{s_i, s_a, s_b, s_c\}$ and $\{s_j, s_a, s_b, s_c\}$ are *exchangeable* on pair (s_i, s_j); otherwise, they are *nonexchangeable* on pair (s_i, s_j).

Theorem 4.1. *Given a complete set Q of quartets on species set $S = \{s_1, s_2, \ldots, s_n\}$, a pair of species s_i and s_j must be siblings in any optimal phylogeny if the number of nonexchangeable pairs on (s_i, s_j) plus the number of quartets conflicting (s_i, s_j) is strictly less than $\lfloor \frac{(n-3)}{2} \rfloor$.*

Proof. Let n_1 denote the number of of quartets conflicting (s_i, s_j) and n_2 denote the number of nonexchangeable pairs on (s_i, s_j). We partition the quartet set Q into three parts (see Table 1). Every quartet in Part 1 does not involve species s_i neither s_j; every quartet in Part 2 involves exactly one species of s_i and s_j; every quartet in Part 3 involves both species s_i and s_j. Quartets in Part 2 can be paired up to have three other species in common, that is, they have the form of $\{s_i, s_a, s_b, s_c\}$ and $\{s_j, s_a, s_b, s_c\}$. Each such pair is either exchangeable or nonexchangeable.

Table 1. Partition Q into three parts.

Part	Quartets
1	quartets not involving any of s_i and s_j $\{s_a, s_b, s_c, s_d\}$
2	quartets involving exactly one of s_i and s_j $\{s_i, s_a, s_b, s_c\}$, $\{s_j, s_a, s_b, s_c\}$
3	quartets involving both of s_i and s_j $\{s_i, s_j, s_a, s_b\}$

An argument to show the contradiction is done by proving that for any phylogeny T_1 in which s_i and s_j are not siblings, a new phylogeny T_2 in which s_i and s_j are siblings can be constructed to satisfy more quartets than T_1, using the fact that $n_1 + n_2 < \frac{n-3}{2}$. For the page limit we omit the detailed proof here. □

4.3. Reducing the Domain Size

We define the *height* of an internal node v in a rooted phylogeny as the maximum number of internal nodes along any path from v to a leaf node in the subtree rooted by v. For any rooted phylogeny T, we can label each internal node by its height. This gives an ultrametric labeling scheme for T. Suppose we know in advance height of the root, denoted by h. Then the domain of variables for the target ultrametric matrix can be reduced to $\{1, 2, \ldots, h\}$. What complicates the search of the target ultrametric matrix is that we do not know h in advance.

Theorem 4.2. *Given a quartet set Q on species set $S = \{s_1, s_2, \ldots, s_n\}$, there exists a rooted phylogeny T which satisfies a maximum number of quartets in Q and the height of the root is at most $\lceil \frac{n}{2} \rceil$.*

Proof. The proof is done by re-rooting phylogeny T, if its height is greater than $\lceil \frac{n}{2} \rceil$, while maintaining the number of quartets satisfied. The process is done by first discarding the root of T to get an unrooted phylogeny denoted as T'. Note that an unrooted phylogeny can be rooted on any of the possible $2n - 3$ edges without changing the satisfied subset of quartet. In T', the longest path between any two leaf nodes has a maximum number of n internal nodes. We can root T' on the edge which is in the middle of the longest path. This way, every path from root to a leaf node has a maximum number of $\lceil \frac{n}{2} \rceil$ internal nodes. In such a way, we obtain a new rooted phylogeny T' which satisfies the same number of quartets as T and the height of its root is at most $\lceil \frac{n}{2} \rceil$. □

From Theorem 4.2, we conclude that in the search of a target ultrametric matrix, we can limit the domain of variables to be $\{1, 2, \ldots, \lceil \frac{n}{2} \rceil\}$. Furthermore, we can restrict that only the least common ancestor of two leaf siblings can be labeled by 1, and if two species (s_i, s_j) can not be siblings in any optimal phylogeny, the domain of matrix variable $M(i, j)$ is reduced to $\{2, 3, \ldots, \lceil \frac{n}{2} \rceil\}$.

To determine that two species (s_i, s_j) can not be siblings in any optimal phylogeny, we take advantage of some existing fast phylogeny construction heuristics (such as neighbor-joining[14] to get a near-optimal phylogeny on the input quartet set Q, which gives a lower bound of the MQC problem. In other words, suppose the near-optimal phylogeny can satisfy k quartets in a complete quartet set Q on $S = \{s_1, s_2, \ldots, s_n\}$; then any optimal phylogeny will have a maximum $\left(\binom{n}{4} - k\right)$ unsatisfied quartets in Q.

Theorem 4.3. *Given a complete quartet set Q on species set $S = \{s_1, s_2, \ldots, s_n\}$, a pair of species s_i and s_j must not be siblings in any optimal phylogeny if the number of conflicting quartets plus the number of nonexchangeable pairs on (s_i, s_j) is greater than $\left(\binom{n}{4} - k\right)$.*

Proof. Let n_1 be the number of of quartets conflicting (s_i, s_j) and n_2 be the number of nonexchangeable pairs on (s_i, s_j).

Suppose s_i and s_j are siblings in an optimal phylogeny T. To achieve this phylogeny, we need change at least those quartets that conflict (s_i, s_j) and at least one quartet of each

nonexchangeable pair on (s_i, s_j). This gives at least $n_1 + n_2$ quartets that are not satisfied by T, which is a contradiction. □

5. Computational Results

We have proposed a number of speedup strategies for the Constraint Programming formulation for the MQC problem. To investigate the performance and usefulness of these strategies and the formulation, we performed experiments on both artificial datasets and a real dataset. The experiments were done on an IBM P690 computer with a Power 4 1.7 GHz processor.

5.1. Artificial Dataset

We describe first how we generated the artificial datasets. For a given number n of species, we generated a random binary tree to act as their phylogeny and extract one quartet for every four species from the phylogeny. This gives a compatible set of $\binom{n}{4}$ quartets. We then changed some arbitrarily selected quartets to give the set Q. The number of changed quartets is a given percentage (p, ranging from 5% to 20%) of the total number of quartets. (We also set the number of unchanged quartets to be a lower bound k of our solution.)

The computational results are summarized in Table 2, which show that our program was able to reconstruct an optimal phylogeny for up to 25 species in 102 hours with quartet error rate as large as 20%. For significance comparison purpose, consider the computational results reported by Ben-Dor *et al.*,[3] who also solve the MQC problem optimally. They were able to solve instances containing up to 20 species and a computational time of 128 hours for the most complicated case on a SUN Ultra 4 with 300MHz, which by a rough scaling is not as fast as we can do. Further more fair comparisons between our method and Ben-Dor's method are undergoing, and will be reported elsewhere.

Table 2. Computational time on artificial datasets.

n	p	time	n	p	time
10	5%	0.03 seconds	15	5%	0.05 seconds
10	10%	0.05 seconds	15	10%	0.54 seconds
10	15%	0.05 seconds	15	15%	0.68 seconds
10	20%	0.06 seconds	15	20%	0.81 seconds
20	5%	40 minutes	25	5%	6 hours
20	10%	6 hours	25	10%	89 hours
20	15%	8 hours	25	15%	102 hours
20	20%	8 hours	25	20%	102 hours

Table 3 shows how Theorems 4.1 and 4.3 work in our computations. In the table, p_1 is the percentage of siblings found by Theorem 4.1 and p_2 is the percentage of non-sibling species found by Theorem 4.3. Intuitively, the less number of unsatisfied quartets, the more number of siblings can be discovered. In our experiments, we found that when the number of unsatisfied quartets is less than 5% of total number of quartets, we could find all the siblings in the optimal phylogeny. We could also find all the species that can not be a sibling to any other species due to the good quality lower bound used in our computations.

Table 3. Performance of speedup strategies.

p	p_1	p_2
5%	100%	100%
10%	95%	100%
15%	82%	100%
20%	41%	100%

Tables 2 and 3 also show that the quartet error rate may affect the performance greatly, typically when the number of species under consideration exceeds 15. When the error rate is less than 10%, the problem can be solved more efficiently. Therefore, a better quality quartet inference technique would be very helpful in the quartet based phylogeny reconstruction process.

5.2. A Prokaryote Dataset

In our experiments, we computed an optimal phylogeny for a set of species containing 20 Prokaryote and 5 Eukaryote species. We adopt the naming and abbreviation convention in the paper of Hao et al.[10]

The "Bergey Code" of every Prokaryote species is a shorthand of the classification given in the 2001 edition of *Bergey's Manuals of Systematics Bacteriology*[16], which collects the most comprehensive taxonomic information of Prokaryote. For the first letter of Bergey Code, 'A' means Archaea and 'B' means 'Bacteria'. The following digits give the code of species phylum, class, order, family and genus. For example, *Ureaplasma urealyticum (urepa)* is listed under Phylum BXIII (*Firmicutes*) - Class II (*Mollicutes*) - Order I (*Mycoplasmatales*) - Family I (*Mycoplasmataceae*) - Genus IV (*Ureaplasma*). We changed all Roman numerals to Arabic and wrote the lineage as B13.2.1.1.4, dropping the taxonomic units and the Latin names.[10]

The input data to this experiment were the whole genome sequences of these 25 species we downloaded from NCBI.[1] Briefly, we used their whole genome sequences to compute a distance matrix using a measure proposed in the paper of Hao et al.[10] We then apply the *Four-Point Method*[6] on the distance matrix to infer a quartet for every subset of 4 species. This gives a compatible set of $\binom{25}{4} = 12650$ quartets.

Using the distance matrix for the 25 species alone, a phylogeny can be constructed by calling a neighbor-joining executable provided by PHYLIP package.[7] The output phylogeny and the phylogeny constructed by our program are shown in Figure 2. It can be seen that both phylogenies support the classifications provided by *Bergey's Manuals of Systematics Bacteriology*[16] quite well in the overall structure and in many details. During our construction, we found six of the eight pairs siblings in the optimal phylogeny before doing Constraint Programming computation. Each pair of found siblings are very close based on the Bergey's classification. This shows that our optimization methods can not only reduce the computational time, but also give a good preview of the relationships among species.

It is interesting to see that out of the total amount of 12650 quartets, the PHYLIP phylogeny satisfies only 10750 of them, and ours satisfies 216 more (86.7%). Looking more

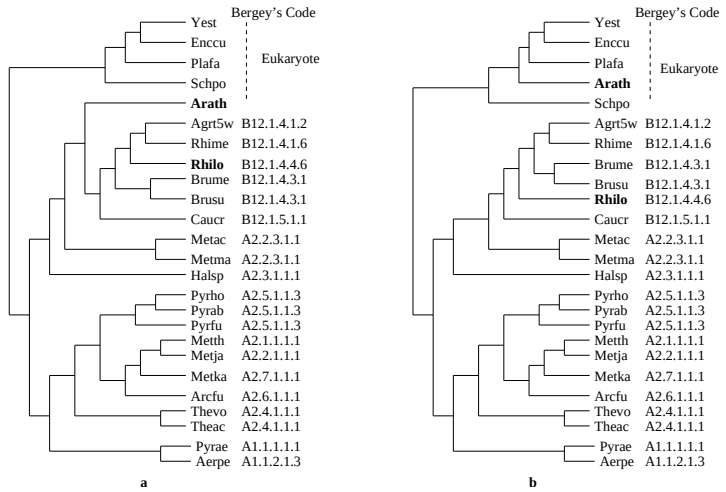

Figure 2. (a) Phylogeny produced by PHYLIP package; (b) Phylogeny produced by our program.

closely, there are two main differences between these two phylogenies. First, the phylogeny generated by PHYLIP falsely put one Eukaryote species into Prokaryote category. This is corrected by the phylogeny generated by our method. Second, the phylogeny generated by PHYLIP package put `Rhilo` very close to `Argt5w` and `Rhime`. By examining the distance matrix directly, we found that the distance between `Rhilo` and `Rhime` is very small compared to other distance entries; and this was probably the reason why PHYLIP put these three species together in its phylogeny. On contrary, the phylogeny produced by our program not only considered this small distance, which led to the fact that the path connecting `Argt5w` and `Rhime` only contains three internal nodes, but also considered all the other distance entries involving one of them. Therefore, it gave a more accurate position of `Rhilo`. Compared to the phylogeny generated by PHYLIP, our phylogeny seems more accurate and reflects the true relationships among these species.

From this experiment on a real dataset, we can see that the solution of the MQC problem has the ability of correcting phylogenies from other heuristic phylogeny construction methods. Although those heuristic methods are fast and could generate phylogenies on larger numbers of species, their phylogenies may not be as accurate as ours, despite the fact that our method needs more computational time.

6. Conclusions and Future Work

We have proposed a number of optimization strategies for a new formulation of the MQC problem through Constraint Programming. The formulation, together with our speedup strategies, might lead us to a new perspective of the problem, as our preliminary experiments on both simulated and read datasets showed that the proposed approach outperforms previous exact algorithms proposed for the MQC problem. Although in the worst case our approach of solving the MQC problem still takes exponential time, it allows the incorporation of the domain knowledge into the search process. In the ideal case, we might be able to encode the target matrix variables such that the exponential behavior becomes a rare

occurrence, and the average behavior is acceptable for practical use.

Currently, our encoding scheme can solve instances containing up to 25 species in around 4 days in a 1.7GHz processor. One of the most important future work we want to pursue is to improve our encoding scheme to further speed up the computation. Our goal is set for solving instances containing 80 species within a day, and thus to provide another fast way to optimal phylogeny construction.

We have mentioned that there exist quite a number of algorithms and heuristics in the literature for solving the MQC problem either optimally or approximately. Our immediate goal is to conduct an extensive comparison between our method and these existing ones, on both phylogeny quality and computation time. With these tasks set, our ultimate goal is to fully explore the structure properties of the MQC problem and to design a quartet specific solver.

References

1. *NCBI Taxonomy Browser*. Accessible through `http://www.ncbi.nlm.nih.gov/Taxonomy/`.
2. H. Bandelt and A. Dress. Reconstructing the shape of a tree from observed dissimilarity data. *Advance in Applied Mathematics*, 7:309–343, 1986.
3. A. Ben-Dor, B. Chor, D. Graur, R. Ophir, and D. Pelleg. From four-taxon trees to phylogenies: the case of mammalian evolution. In *Proceedings of the RECOMB*, pages 9–19, 1998.
4. M. C. H. Dekker. Reconstruction methods for derivation trees. Master's thesis, Vrije University, Amsterdam, 1986.
5. P. Erdos, M. Steel, L. Szekely, and T. Warnow. Inferring big trees from short sequences. In *Proceedings of the International Congress on Automata, Languages, and Programming*, volume 1256 of *Lecture Notes in Computer Science*, pages 827–837, 1997.
6. P. Erdos, M. Steel, L. Szekely, and T. Warnow. A few logs suffice to build (almost) all trees (Part 1). *Random Structures and Algorithms*, 14:153–184, 1999.
7. J. Felsenstein. *PHYLIP*. Accessible through `http://evolution.genetics.washington.edu/phylip.html`.
8. J. Felsenstein. The number of evolutionary trees. *Systematic Zoology*, 27:27–33, 1978.
9. D. Graur. Mammalian phylogeny: using every available molecule. In *Proceedings of the Training Course in Molecular Evolution*, pages 6–8, Bari Italy, 1996.
10. B. Hao and J. Qi. Prokaryote phylogeny without sequence alignment: from avoidance signature to composition distance. In *Proceedings of the 2003 IEEE Bioinformatics Conference (CSB 2003)*, pages 375–385, 2003.
11. T. Jiang, P. E. Kearney, and M. Li. Orchestrating quartets: approximation and data correction. In *Proceedings of the 39th FOCS*, pages 416–425, 1998.
12. T. Jiang, P. E. Kearney, and M. Li. Some open problems in computational molecular biology. *Journal of Algorithms*, 34:194–201, 2000.
13. D. Pelleg. Algorithms for constructing phylogenies from quartets. Master's thesis, Israel Institute of Technology, 1998.
14. N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.
15. S. Sattath and A. Tversky. Additive similarity trees. *Psychometrika*, 42:319–345, 1977.
16. Bergey's Manual Trust. *Bergey's Manual of Systematic Bacteriology*. Springer-Verlag, New York, 2nd edition, 2001.
17. G. Wu, G.-H. Lin, and J. You. Quartet based phylogeny reconstruction with answer set programming. In *Proceedings of the 16th ICTAI*, 2004.