Quartet-based phylogeny reconstruction from gene orders*

Tao Liu[1], Jijun Tang[2], and Bernard M.E. Moret[1]

[1] Department of Computer Science, U. of New Mexico, Albuquerque, NM 87131
[2] Department of Computer Science & Engineering, U. of South Carolina, Columbia, SC 29208

Abstract. Phylogenetic reconstruction based on gene rearrangements is attracting increasing attention from biologists and computer scientists. Methods used in reconstruction include distance-based methods, parsimony methods using sequence encodings, and direct optimization. The latter, pioneered by Sankoff and extended by us with the software suite `GRAPPA`, is the most accurate approach; however, its exhaustive nature means that it can be applied only to small datasets (of fewer than 15 taxa). While we have successfully scaled it up to 1,000 taxa by integrating it with a disk-covering method, yielding `DCM-GRAPPA`, the recursive decomposition in the DCM may require many levels of recursion to handle datasets with 1,000 or more taxa. In order to handle larger datasets and reduce the need for recursive decomposition, we investigate quartet-based approaches, which directly decompose the datasets into subsets of four taxa each. Such approaches have been well studied for sequence data, but not for gene-order data. We give an optimization algorithm for the NP-hard problem of computing optimal trees for each quartet, present a variation of the dyadic method (using heuristics to choose suitable short quartets), and use both in simulation studies. We find that our quartet-based method can handle more taxa than the base version of `GRAPPA`, thus enabling us to reduce the number of levels of recursion in `DCM-GRAPPA`, but is more sensitive to the rate of evolution, with error rates rapidly increasing when saturation is approached.

1 Introduction

Modern techniques can yield the ordering and strandedness of genes for genomes; each chromosome can then be represented by an ordering of signed genes, where the sign indicates the strand. Rearrangement of genes under inversion, transposition, and other operations such as duplications, deletions and insertions, is an important evolutionary mechanism [7]. Reconstructing phylogenies from gene-order data has been studied intensely since the pioneering papers of Sankoff [2, 23]. Because they capture the complete genome, gene-order data do not suffer from the gene tree vs. species tree problem; and because rearrangements of genes are *rare genomic events* [19], gene-order data enable the reconstruction of evolutionary events far back in time. Many biologists have embraced this new source of data in their phylogenetic work [7, 17, 18], while computer scientists are slowly solving the difficult problems posed by the manipulations of these gene orders [16]. Studies conducted by our group [15, 26, 29–31] confirm that gene-order data support very accurate reconstructions.

* Contact author: Bernard Moret, `moret@cs.unm.edu`

The main software package for analyzing gene-order data is GRAPPA, based on the BPAnalysis software of Sankoff and Blanchette [23]. GRAPPA achieved a billion-fold speed-up over BPAnalysis [15]; it can analyze datasets of 13 organellar genomes in 20 minutes on a laptop. Extensive testing has shown that the trees returned by GRAPPA are better than those returned by other methods based on gene orders, such as distance-based methods and parsimony based on encodings [6, 31]. However, since GRAPPA examines every possible tree, it only handle only small datasets—a 17-taxon analysis would take a month on today's most powerful computers. We integrated GRAPPA with DCM, a divide-and-conquer approach pioneered by Warnow [11], to produce DCM-GRAPPA [29], which can analyze datasets of up to 1,000 taxa without loss of accuracy.

DCM-GRAPPA works in three steps: it first decomposes the dataset into overlapping subproblems (disks), then runs GRAPPA on the subproblems, and finally uses a specialized supertree method [20] to build a tree for the original dataset from the trees returned by GRAPPA for the subproblems. Because the decomposition technique of DCM can still produce subproblems too large for GRAPPA to handle, we call DCM recursively until each subproblem size falls below a given threshold. Because the threshold is small (14 taxa), large problems require many levels of recursive decomposition, which is time-consuming and also risks propagating and amplifying errors in the assembly of the subtrees. On 1,000 taxa, DCM-GRAPPA needs 6–7 levels of recursion if limited to disks of at most 14 taxa, but only 2–3 levels if allowed disks of up to 20 taxa.

One can also decompose the set of taxa into the smallest possible subsets for which meaningful answers exist, namely *quartets*, sets of four taxa. (Sets of two or three taxa can produce only one tree, but a quartet can give rise to three distinct unrooted trees.) While there are many such quartets, their tiny size should make them easy to compute. If every quartet tree is computed correctly from noiseless data, then there is a single tree compatible with all $\binom{n}{4}$ quartet trees and that tree is the true tree [4]; in practice, of course, many of the quartet trees are in error and no single tree is compatible with all quartet trees. With sequence data, biologists have long used the heuristic method known as quartet-puzzling [25], while computer scientists have developed several theoretical methods, such as quartet cleaning [1, 3, 12]—see [24] for a review and experimental comparison of these methods. In the case of gene orders, however, finding the best quartet tree is NP-hard—it includes finding the median of three genomes, a known NP-hard problem [5], as a special case.

We present algorithmic and experimental results that lead to a reconstruction method from gene-order data which overcomes some of the problems associated with quartet methods. After some background review, we describe in Section 3 our exact method to compute optimal quartet trees under breakpoint distances and present experimental results on our approach to resolving the quartets as well as on our use of the dyadic inference rule [10] to obtain a sufficient set of quartets from a selected subset of short quartets (inspired from the short-quartet method [8, 32]). In Section 5, we summarize our experiments on simulated and biological datasets, the results of which suggest that our method can produce accurate topologies (much more accurate than neighbor-joining) for datasets of up to 25 taxa within reasonable time, provided that the genomes are large enough to avoid saturation.

2 Definitions and Notation

A quartet is a quadruple of taxa; a quartet tree is an unrooted binary tree with four leaves labelled by these taxa. Given a quartet $\{a,b,c,d\}$, we say that a quartet tree on this set is *unresolved* if it is a star (four edges, each touching a leaf) and denote it by $(abcd)$. If the quartet tree has an internal edge separating two pairs of leaves we say that it is *resolved* and, if the pairs are a,b and c,d, denote it by $ab \mid cd$. The four possible quartet trees induced by a quartet are depicted in Figure 1. We will use quartet in lieu of quartet tree when the sense is clear.

Quartet $ab \mid cd$ *agrees* with tree T if all four of its taxa are leaves of T and the path from a to b in T does not intersect with the path from c to d in T. Equivalently, $ab \mid cd$ agrees with a tree if the subtree induced in T by the four-taxon subset $\{a,b,c,d\}$ is the quartet tree itself. Quartet $ab \mid cd$ is *in error* with respect to the tree T if it does not agree with T. If Q_T denotes the set of all quartets that agree with T, then T is uniquely characterized by Q_T; moreover, T can be reconstructed in polynomial time from Q_T [8]. (Of course, the set Q of $\binom{n}{4}$ quartets that we can construct is only an approximation of Q_T.) In Figure 2, quartet $ac \mid bd$ would be in error, since it does not appear in Q_T.

Quartet-based methods operate in two phases. First, they construct a set Q of resolved quartets—usually by determining the preferred tree for each of the $\binom{n}{4}$ quartets. Because each dataset has size 4, any phylogenetic method can be used to estimate the quartet tree, including maximum likelihood and maximum parsimony (see [27]), neighbor-joining [21], the relaxed four-point method [9], and the ordinal quartet method [13]. In the second phase, the resolved quartets

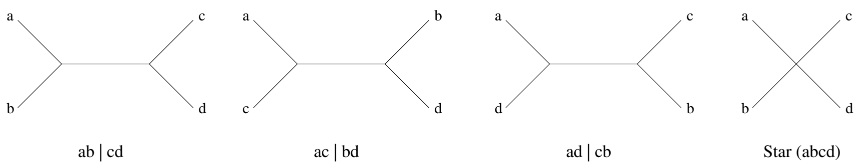
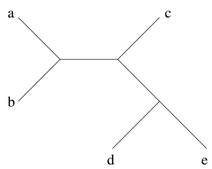


Fig. 1. The four possible quartet trees for quartet $\{a,b,c,d\}$.



ab | cd
ab | ce
ab | de
ac | de
bc | de

Fig. 2. An evolutionary tree T and its set Q_T of induced quartet trees.

are used to build a single tree on the full set of taxa. This second phase is simple when all quartets are compatible, but a challenge when some of the quartets conflict with others—a common occurrence when insufficient data are present [24].

3 Inferring Quartet Topologies

We can identify maximally parsimonious quartet trees by examining each of the three possible trees, assigning gene orders to the two internal nodes so as to minimize the score (the sum of the lengths of the five edges) of each tree, and returning the tree with the lowest score. However, identifying such gene orders is NP-hard even for just one internal node and the simplest distance measures [5].

An easy approach is to use `GRAPPA` to construct a tree for each quartet; although the result need not be optimal, our experiments, as well as earlier ones [14], show that optimality is reached in most realistic cases. If we limit ourselves to breakpoint distances, we can solve the NP-hard optimization problem directly, using variants of the reduction to TSP devised by Sankoff and Blanchette [22]; we devised two such variations, which we call `Qtsp` and `Qedge`.

Sankoff's reduction to TSP can be summarized as follows. Given three genomes with n genes, we build the complete graph K_{2n} on the $2n$ vertices g_1, $-g_1$, g_2, $-g_2$, …, g_n, $-g_n$. Edge (g,h) in this graph is assigned a weight as follows: if we have $g = -h$, then we set the weight to a large negative value to ensure that (g,h) is part of any solution, otherwise we set it to $3 - \mathrm{adj}(g,h)$, where $\mathrm{adj}(g,h)$ is the number of times that $-g$ and h are adjacent in the given genomes. If $s = s_1, -s_1, s_2, -s_2, \cdots s_n, -s_n$ is the solution to the TSP, then the median of the given genomes is $g = s_1, s_2, \cdots, s_n$.

This result applies to one unknown genome, but we need to identify two such for quartets. Our first algorithm views the two as forming a pair and remaps the problem into a universe where pairs form the unit of computation and where a single tour (of pairs) defines both genomes; our second algorithm retains the original formulation, but looks for a pair of tours.

3.1 The `Qtsp` Method

We look for one permutation of size n, each entry of which consists of a pair of genes (g_i, g_j). Let the two desired internal genomes be e and f, where genome

e is connected to genomes a, b, and f, and genome f is connected to genomes c, d, and e. Build a complete graph, here on $(2n)^2$ vertices, each a pair of signed genomes. For each edge $\{(g_{i1}, g_{i2}), (g_{j1}, g_{j2})\}$ in this graph, we set the weight of the edge as follows: if we have $g_{i1} = g_{j1}$ and $g_{i2} = g_{j2}$, then we set the weight to a large negative value to ensure that this edge is part of any solution, otherwise, we set it to $4 - u_1(g_{i1}, g_{j1}) - u_2(g_{i2}, g_{j2})$, where $u_1(g_{i1}, g_{j1})$ is the number of times $-g_{i1}$ and g_{j1} are adjacent in the genomes a and b and $u_2(g_{i2}, g_{j2})$ is the number of times $-g_{i2}$ and g_{j2} are adjacent in the genomes c and d.

Proposition 1. *If* $s = (s_{11}, s_{12}), (-s_{11}, -s_{12}), \cdots, (s_{n1}, s_{n2}), (-s_{n1}, -s_{n2})$ *is the solution to the TSP on G, then an optimal assignment for the internal genomes is* $e = s_{11}, s_{21}, \cdots, s_{n1}$ *and* $f = s_{12}, s_{22}, \cdots, s_{n2}$.

After transforming the problem, we can use the efficient TSP routine of `GRAPPA` to search for the optimal solution (after modifying it to introduce one more bucket of costs, since now the edge costs of interest are 1, 2, and 3, not just 1 and 2). The problem is that the instance thus created is of size quadratic in the number of genes; combined with the extra bucket of costs, the large instance size makes it difficult to obtain solutions to sizeable instances.

3.2 The `Qedge` Method

This method uses the original TSP formulation, of size linear in the number of genes, but seeks simultaneously to optimize tours in two separate graphs. Let a, b, c, d, e, and f be as before. We set up two complete graphs on $2n$ vertices each—one for e and one for f. For each edge $\{g, h\}$, we set its weight as follows: if we have $g = -h$, we set the weight to a large negative value to force its inclusion; otherwise, we set it to $2 - u(gh)$, where $u(gh)$ is the number of times $-g$ and h are adjacent in the genomes a and b (for an edge in the first graph) or in the genomes c and d (for an edge in the second graph). Now, when adding an edge to the two tours under construction, we can either pick different edges from the two graphs, each with the minimum weight in its own graph, and add one breakpoint between e and f to the total cost; or pick the same edge in both graphs, even if not locally optimal, thereby saving a breakpoint between e and f; our algorithm computes the cost of each choice and picks the choice of lower cost.

3.3 Experimental Results for `Qtsp` and `Qedge`

We ran tests for these two methods and `GRAPPA` on quartets of 10, 20, 30, 40, and 50 genes under evolutionary rates (the expected numbers of events per edge of the model tree) of $r = 1$, 2, 3, 4, and 5. The running times of `Qtsp` and `Qedge` are shown in Figure 3. As expected, `Qedge` runs much faster than `Qtsp`; however, its running time depends strongly on the quartet score, so that it may prove unusable in the reconstruction of large trees, where many of the quartets will have large scores. Both `Qtsp` and `Qedge` reconstructed the same optimal quartet

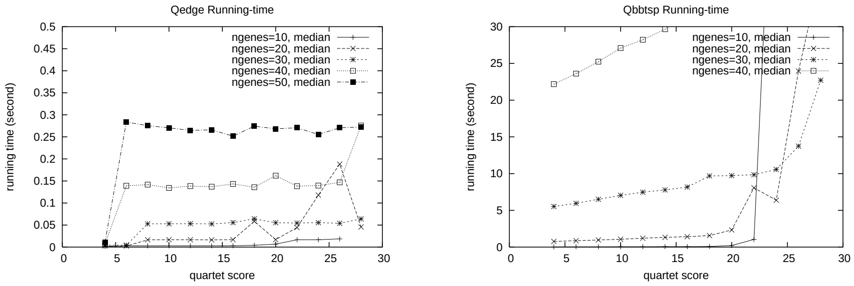


Fig. 3. Running times of `Qtsp` (left) and `Qedge` (right) as a function of quartet score

trees (except for ties), which were the same as the model tree in 97% of cases (although their scores were usually lower than the model tree scores); most of the 3% came from the 10-gene case, which gets quickly saturated for evolutionary rates above $r = 3$. `GRAPPA` did almost as well: 96% of the quartets it returned matched the model tree—and it returned answers in a few microseconds. Since reconstructing phylogenies from quartets requires the computation of a large set of quartets, the running time is critical. Therefore, based on our results, we chose `GRAPPA` as the method to resolve the quartets.

4 Phylogenetic Reconstruction from Quartets

The computational challenge of quartet recombination (building a single tree from the collection of quartet trees) is how to deal with quartet errors. Most optimization problems related to tree reconstruction from quartets are NP-hard, such as the Maximum Quartet Compatibility problem [12], which seeks a tree T for a given set of quartet Q such that $|Q_T \cap Q|$ is maximized. Various methods have been designed to handle quartet errors. The dyadic-closure method simply issues an error message and quits [10]. The $Q*$ method seeks the maximally resolved tree T' that obeys $Q(T') \subseteq Q$, a very conservative method that generally produces many polytomies [4]. Quartet-cleaning methods establish a bound on the number of quartet errors around each reconstructed tree edge [1, 3, 12]. None of these methods produces satisfactory results on sequence data [24]. It is theoretically possible to produce the true tree by selecting a subset of short quartets and adding further quartets according to an inference rule [8], but this result assumes perfect data and gives no simple method by which to select the subset of quartets. We thus set out to design a selection rule and investigate its performance, using the dyadic inference rules [10]:

1. If $ab \mid cd$ is a valid quartet, so are $ba \mid cd$ and $cd \mid ab$.
2. If $ab \mid cd$ and $ac \mid de$ are valid quartets, so are $ab \mid ce$, $ab \mid de$, and $bc \mid de$.
3. If $ab \mid cd$ and $ab \mid ce$ are valid quartets, so is $ab \mid de$.

4.1 Selecting a Subset of Quartets

Figure 4 shows two possible resolutions for quartet $\{a,b,c,d\}$. In the first topology, the two pairs of genomes $\{a,b\}$ and $\{c,d\}$ are far apart from each other, but in the second topology the two pairs $\{a,c\}$ and $\{b,d\}$ are quite close: the first topology is more likely to be correct, an observation supported by the relaxed four-point method [9]:

Compute pairwise distances among a, b, c, and d; return ab | cd if we have $d_{ab} + d_{cd} < min(d_{ac} + d_{bd}, d_{ad} + d_{bc})$*, but return a star if all sums are equal.*

Fig. 4. Two quartet trees; the left has a higher probability of correctness

Since computing all $\binom{n}{4}$ quartets takes too long, we can use this relaxed four-point method to choose quartets and reduce the overall running time. After resolving the quartets, we can assign a weight to each resolved quartet to measure our confidence in that quartet: for example, we can use the inversion distance between the two internal nodes.

4.2 Fixing Quartet Errors

Although GRAPPA is reliable and although we can pick only quartets of larger weight, quartet errors still arise, especially when we are forced to select some quartets of low weight in order to resolve every internal tree edge. We propose a simple new method, quite distinct from quartet cleaning, to handle errors. Since the quartets are weighted and since we place more trust in quartets of higher weight, we examine the source of quartet errors and, whenever two quartets are incompatible, we remove the one with lower weight.

Starting from a large initial set of resolved quartets only returns us to a version of the NP-hard problem Maximum Quartet Compatibility. Instead, we proceed incrementally. We select a high weight threshold and only retain quartets (computed on the fly with GRAPPA) with weights above that threshold; if we find quartet errors, we remove the incompatible quartets of lower weight. We then apply the dyadic inference rules to augment our collection of compatible quartets. Finally, if the resulting set of quartets fully resolves the tree, we are done (a method like $Q*$ will recover the tree), otherwise we lower the threshold and add to our set the newly eligible quartets. By controlling the decrease in the weight threshold, we can control the tradeoff between running time and quality.

Since we do not know the weight of a quartet until we resolve it, but want to avoid resolving useless quartets, we need a fast method to select quartets to resolve. Given quartet $q = \{a,b,c,d\}$, define the *width* of q as
$q_w = max(d_{ab}+d_{cd}, d_{ac}+d_{bd}, d_{ad}+d_{bc}) - min(d_{ab}+d_{cd}, d_{ac}+d_{bd}, d_{ad}+d_{bc})$
As q_w increases, the two pairs of genomes move further apart and the weight increases: hence we can decide which quartets to resolve by comparing their width with the weight threshold. Even if the threshold is lowered to zero, the set of compatible quartets may remain inadequate to resolve the tree—in which case we have no choice but to leave these unresolved polytomies in the output.

5 Experimental Results

If the true tree has an edge defining a bipartition with no equivalent in the reconstructed tree, that edge is a *false negative (FN)*; conversely, if the reconstructed tree has an edge with no equivalent in the true tree, that edge is a *false positive (FP)*. FP edges are more problematic than FN edges.

We generated model tree topologies from the uniform distribution on binary trees, each with 12, 16 and 20 leaves respectively. On each tree, we evolved signed permutations of 40, 60 and 80 genes, using evolutionary rates (the expected numbers of events along a tree edge) of 2, 4, 6. For each combination of parameters, we generated 20 trees; the final results are averaged on the 20 datasets. We computed quartets using `GRAPPA` and built the resulting tree using our algorithms for selecting quartets of high weight, eliminating conflicting quartets, and expanding the set with the dyadic rules. Figure 5 shows FP and FN rates for datasets with 80 genes. Our method did well, but saturation (high evolutionary rates leading to ill-defined estimates of distances) causes a small increase in the error rate. This observation is confirmed by our results on datasets with 40 genes, where saturation occurs much sooner and the results are unacceptable, as seen in Figure 6. It can also be seen in Figure 5 that, with very low evolutionary rates, many quartets cannot be satisfactorily resolved (we get equalities and thus star topologies), leading to poor resolution and many false negatives.

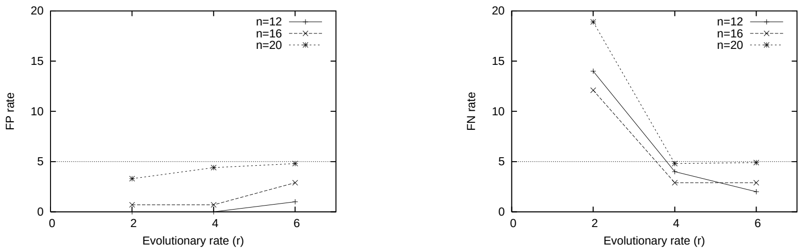
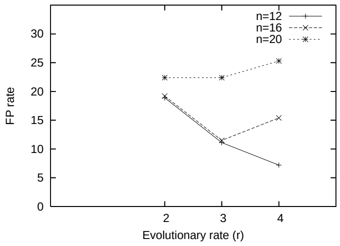


Fig. 5. Performance of our method on genomes of 80 genes.



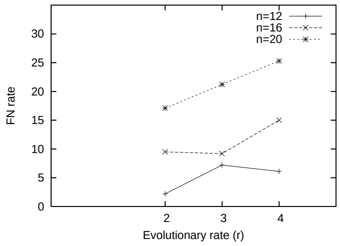


Fig. 6. Performance of our method on genomes of 40 genes.

Our tests verified that a small subset of quartets suffices to infer the complete set of quartets. For datasets with 12 genomes, only 75 quartets (15% of the total) are needed; with 20 genomes, only 270 quartets (6%) are needed. Our selection rule worked well: of the quartets selected, fewer than 2% overall were found to be incompatible. Interestingly, the set of resolved quartets produced by our method produced very accurate reconstructions, while the set produced directly by the relaxed four-point method gave very poor results.

We compared the results obtained by our method with those obtained by simply running the (very fast) neighbor-joining (NJ) method on breakpoint and inversion distance matrices (computed by GRAPPA) for each dataset. For 80-gene genomes, the Robinson-Foulds rate (the average of FP and FN rates) for NJ varied from 20% (for $r = 2$) down to 2–5% (for $r = 6$, with lower rates for 12 genomes and larger rates for 20 genomes), compared to a maximum of 10% (for $r = 2$) down to 1.5–4.5% (for $r = 6$) for our method. For 40-gene genomes, as we observed, our method suffers from saturation effects with 16 or 20 genomes, where its error rate roughly matches that of NJ (10-20%); for 12 genomes, where saturation is less of a problem, our method again easily surpasses NJ, with a median error rate of 8.5% compared to NJ's rate of 14%.

6 Conclusions

We have presented a quartet-based phylogeny reconstruction method for gene-order data and reported its performance on simulated datasets. Our method produces accurate topologies for trees with up to 25 leaves in reasonable time when the datasets do not exhibit significant saturation. The results we have obtained promise well, especially because we have many possible avenues of improvement. For instance, we have recently developed a linear-programming method that can accurately estimate the edge lengths of fairly small trees [28]; by using this method to estimate the length of quartet edges, we can further improve our quartet selection, in terms of both speed and accuracy. Such improvement should also enable us to handle datasets with larger pairwise distances. We designed this method to extend the range of base methods that can be used in conjunction with a disk-covering method: thus the limitation to sets of 20–30 taxa is not an issue, but in fact a potentially significant gain over the direct use of GRAPPA as a base method, since this last is limited to 12–15 taxa.

7 Acknowledgments

This work is supported by the US National Science Foundation under grants EF 03-31654, IIS 01-13095, IIS 01-21377, and DEB 01-20709, by the US National Institutes of Health under grant 2R01GM056120-05A1 (through a subcontract to the U. of Arizona), and by the Dept. of Computer Science and Engineering at the U. of South Carolina.

References

1. V. Berry, T. Jiang, P. Kearney, M. Li, and T. Wareham. Quartet cleaning: improved algorithms and simulations. In *Proc. Europ. Symp. Algs. (ESA99)*, volume 1643 of *Lecture Notes in Computer Science*, pp. 313–324. Springer Verlag, 1999.
2. M. Blanchette, G. Bourque, and D. Sankoff. Breakpoint phylogenies. In S. Miyano and T. Takagi, editors, *Genome Informatics 1997*, pp. 25–34. Univ. Academy Press, 1997.
3. D. Bryant, V. Berry, T. Jiang, P. Kearney, M. Li, T. Wareham, and H. Zhang. A practical algorithm for recovering the best supported edges of an evolutionary tree. In *Proc. 11th Ann. ACM/SIAM Symp. Discrete Algs. (SODA'00)*, pp. 287–296. ACM Press, New York, 2000.
4. P. Buneman. *The recovery of trees from measures of dissimilarity*. Edinburgh University Press, 1971.
5. A. Caprara. Formulations and hardness of multiple sorting by reversals. In *Proc. 3rd Ann. Int'l Conf. Comput. Mol. Biol. (RECOMB'99)*, pp. 84–93. ACM Press, New York, 1999.
6. M.E. Cosner, R.K. Jansen, B.M.E. Moret, L.A. Raubeson, L. Wang, T. Warnow, and S.K. Wyman. An empirical comparison of phylogenetic methods on chloroplast gene order data in Campanulaceae. In D. Sankoff and J.H. Nadeau, editors, *Comparative Genomics*, pp. 99–122. Kluwer Academic Publishers, 2000.
7. S.R. Downie and J.D. Palmer. Use of chloroplast DNA rearrangements in reconstructing plant phylogeny. In P. Soltis, D. Soltis, and J.J. Doyle, editors, *Plant Molecular Systematics*, pp. 14–35. Chapman and Hall, 1992.
8. P. Erdős, M. A. Steel, L. A. Székely, and T. Warnow. A few logs suffice to build (almost) all trees I. *Random Structs. and Algs.*, 14:153–184, 1997.
9. P. L. Erdős, M. A. Steel, L. A. Székely, and T. Warnow. Constructing big trees from short sequences. In *Proc. 24th Int'l Colloq. on Automata, Languages, and Programming (ICALP97)*, volume 1256 of *Lecture Notes in Computer Science*. Springer Verlag, 1997.
10. P. L. Erdős, M. A. Steel, L. A. Székely, and T. Warnow. Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule. *Computers and Artif. Intell.*, 16(2):217–227, 1997.
11. D. Huson, S. Nettles, and T. Warnow. Disk-covering, a fast converging method for phylogenetic tree reconstruction. *J. Comput. Biol.*, 6(3):369–386, 1999.
12. T. Jiang, P.E. Kearney, and M. Li. A polynomial-time approximation scheme for inferring evolutionary trees from quartet topologies and its application. *SIAM J. Computing*, 30(6):1942–1961, 2001.
13. P.E. Kearney. The ordinal quartet method. In *Proc. 2nd Ann. Int'l Conf. Comput. Mol. Biol. (RECOMB'98)*, pp. 125–134. ACM Press, New York, 1998.
14. B.M.E. Moret, A.C. Siepel, J. Tang, and T. Liu. Inversion medians outperform breakpoint medians in phylogeny reconstruction from gene-order data. In *Proc. 2nd Int'l Workshop*

Algs. in Bioinformatics (WABI'02), volume 2452 of *Lecture Notes in Computer Science*, pp. 521–536. Springer Verlag, 2002.

15. B.M.E. Moret, J. Tang, L.-S. Wang, and T. Warnow. Steps toward accurate reconstructions of phylogenies from gene-order data. *J. Comput. Syst. Sci.*, 65(3):508–525, 2002.
16. B.M.E. Moret, J. Tang, and T. Warnow. Reconstructing phylogenies from gene-content and gene-order data. In O. Gascuel, editor, *Mathematics of Evolution and Phylogeny*, pp. 321–352. Oxford University Press, 2005.
17. J.D. Palmer. Chloroplast and mitochondrial genome evolution in land plants. In R. Herrmann, editor, *Cell Organelles*, pp. 99–133. Springer Verlag, 1992.
18. L.A. Raubeson and R.K. Jansen. Chloroplast DNA evidence on the ancient evolutionary split in vascular land plants. *Science*, 255:1697–1699, 1992.
19. A. Rokas and P.W.H. Holland. Rare genomic changes as a tool for phylogenetics. *Trends in Ecol. and Evol.*, 15:454–459, 2000.
20. U. Roshan, B.M.E. Moret, T. Warnow, and T.L. Williams. Performance of supertree methods on various dataset decompositions. In O.R.P. Bininda-Edmonds, editor, *Phylogenetic Supertrees: Combining information to reveal the Tree of Life*, pp. 301–328. Kluwer Academic Publishers, 2004.
21. N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4:406–425, 1987.
22. D. Sankoff and M. Blanchette. The median problem for breakpoints in comparative genomics. In *Proc. 3rd Int'l Conf. Computing and Combinatorics (COCOON'97)*, volume 1276 of *Lecture Notes in Computer Science*, pp. 251–264. Springer Verlag, 1997.
23. D. Sankoff and M. Blanchette. Multiple genome rearrangement and breakpoint phylogeny. *J. Comput. Biol.*, 5:555–570, 1998.
24. K. St. John, T. Warnow, B.M.E. Moret, and L. Vawter. Performance study of phylogenetic methods: (unweighted) quartet methods and neighbor-joining. *J. Algorithms*, 48(1):173–193, 2003. A preliminary version appeared in SODA'01, pp. 196–205.
25. K. Strimmer and A. von Haeseler. Quartet puzzling: A quartet maximum likelihood method for reconstructing tree topologies. *Mol. Biol. Evol.*, 13:964–969, 1996.
26. K.M. Swenson, M. Marron, J.V. Earnest-DeYoung, and B.M.E. Moret. Approximating the true evolutionary distance between two genomes. In *Proc. 7th SIAM Workshop on Algorithm Engineering & Experiments (ALENEX'05)*. SIAM Press, Philadelphia, 2005.
27. D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis. Phylogenetic inference. In D.M. Hillis, B.K. Mable, and C. Moritz, editors, *Molecular Systematics*, pp. 407–514. Sinauer Assoc., Sunderland, MA, 1996.
28. J. Tang and B.M.E. Moret. Linear programming for phylogenetic reconstruction based on gene rearrangements. In *Proc. 16th Ann. Symp. Combin. Pattern Matching (CPM'05)*, Lecture Notes in Computer Science, 2005.
29. J. Tang and B.M.E. Moret. Scaling up accurate phylogenetic reconstruction from gene-order data. In *Proc. 11th Int'l Conf. on Intelligent Systems for Mol. Biol. (ISMB'03)*, volume 19 of *Bioinformatics*, pp. i305–i312. Oxford U. Press, 2003.
30. J. Tang, B.M.E. Moret, L. Cui, and C.W. dePamphilis. Phylogenetic reconstruction from arbitrary gene-order data. In *Proc. 4th IEEE Symp. on Bioinformatics and Bioengineering BIBE'04*, pp. 592–599. IEEE Press, Piscataway, NJ, 2004.
31. L.-S. Wang, R.K. Jansen, B.M.E. Moret, L.A. Raubeson, and T. Warnow. Fast phylogenetic methods for genome rearrangement evolution: An empirical study. In *Proc. 7th Pacific Symp. on Biocomputing (PSB'02)*, pp. 524–535. World Scientific Pub., 2002.
32. T. Warnow, B.M.E. Moret, and K. St. John. Absolute convergence: true trees from short sequences. In *Proc. 12th Ann. ACM/SIAM Symp. Discrete Algs. (SODA'01)*, pp. 186–195. SIAM Press, 2001.