Efficiency and Accuracy of Phylogenetic Trees for Large Sequence Datasets

Daniel Keymer, Stanford University

Introduction

Since the early 1990s, the number of sequences in the GenBank database has increased exponentially, and as of February 15, 2007 had reached more than 67 million sequences, with an average size of more than 1000 base pairs (NCBI 2007). This incredible explosion of data acquisition has been fueled by the application of polymerase chain reaction (PCR) technology to environmental samples and the increasing affordability of DNA sequencing for entire microbial genomes. The statistics listed above do not include the 86 billion bases of sequence generated by whole genome shotgun sequencing projects. Since the first microbial genome was sequenced in 1995, the number of sequenced microbial genomes has reached nearly 500 and continues to rise.

As the amount of sequence data increases exponentially, systematic biologists are scrambling to place this information in a biologically meaningful context. This task primarily involves structuring the data in terms of evolutionary relationships as shown in a phylogenetic tree. However, the algorithms traditionally used to build phylogenetic trees often become overwhelmed as the number of sequences for analysis increases, resulting in reduced accuracy of tree structure and absurdly long computation times. The implementation of new or modified algorithms for handling large sequence datasets is paramount to our understanding of evolutionary processes in genomes and gene sequences. This problem is of particular interest to those researchers attempting to reconstruct the organismal Tree of Life, which presents a formidable challenge of many taxa with deeply divergent sequences. This review covers a number of the strategies proposed for increasing efficiency and accuracy in computing phylogenetic trees and provides a critique of what modifications will be necessary to tackle increasingly growing collections of genetic sequences.

Basic Tree-Building Algorithms

In general, phylogenetic reconstruction is based on one of three principles: minimum evolution, maximum parsimony, or maximum likelihood. Minimum evolution (ME) methods find the optimal tree that minimizes the total sum of the branch lengths. These methods compute a distance matrix for the sequences based on a given evolutionary model and then build the tree from this matrix. They include the popular neighbor-joining (NJ) method (Saitou and Nei 1987) and the unweighted pair group method with arithmetic averages (UPGMA) (Sokal and Michener 1958). Similarly, but in a slightly different way, maximum parsimony (MP) methods search for a tree that explains the phylogeny using the minimum number of evolutionary changes (Fitch 1971). A tree length value is computed for each tested topology and the most parsimonious trees have the minimum tree length value. Maximum likelihood (ML) methods compute a log likelihood score for each tested topology and choose the optimal tree achieving the maximum score (Felsenstein 1981). A more thorough description of these families of methods is found in (Felsenstein 1988).

As the number of sequences analyzed increases, the number of possible topologies increases exponentially, so that it rapidly becomes prohibitive to analyze all possible tree topologies for a modestly sized dataset. Since MP and ML algorithms rely on the comparison of multiple trees, one for each possible topology, heuristic searches are typically used to only search a subset of the possible trees to find the optimal tree. Different methods exist for guiding the heuristic search to ensure that the trees searched are closest to the true tee. In addition to the heuristic search algorithms, branch-swapping algorithms are used to test outputted trees, by rearranging the topology of the tree and selecting for the optimal rearrangement. Generally, one of three branch-swapping algorithms are used: nearest-neighbor interchanges (NNI), subtree pruning and regrafting (SBR), and tree bisection-reconnection (TBR). The algorithms are listed here in order of increasing computational complexity. Takahashi and Nei found that MP and ML could be improved greatly by using NNI, with little to no improvement when using other branch-swapping algorithms, but for ME-methods, NJ-NNI was not significantly more accurate than NJ alone (2000).

Criteria for Evaluating Algorithms

Some factors that influence the accuracy and efficiency of tree building algorithms are sequence length, the number of taxa, and model of evolution. Generally, accuracy increases as the sequence length increases and the number of taxa decreases (Nakhleh et al. 2002; Sanderson and Driskell 2003). When collecting data, it is valuable to know what accuracy is required for your application since the scaling of sequence length required for a given number of sequences improves with decreasing accuracy threshold (Bininda-Emonds et al. 2001). Most studies have shown that simpler models of evolution provide for more accurate trees, but there is some uncertainty about how appropriate different models are for real datasets. Besides characteristics of the dataset in question, fundamental traits of the algorithm in question should be considered. Statistical consistency is the property that the optimal estimation of a method approaches the true value as the amount of data included increases. For phylogenetic trees this would mean that the method would produce an optimal tree closer in topology to the true tree as the length of the sequences analyzed increases. Although there is some disagreement among statisticians as to the relative importance of this property in validating an algorithm, it is generally accepted that consistency should be proven for all algorithms. Users should also decide what output information from a phylogeny is most useful for their analysis. While many methods are designed to compute branch lengths along with the branching structure of the tree, there is often a trade-off between accurately depicting branch lengths versus topology (Takahashi and Nei 2000). Computation speed is a factor controlling the ability of certain methods to be used for a given dataset, but faster methods are being developed which should overcome these problems as explained below.

Simulations

Computer simulations are often used to evaluate algorithms for several reasons. First, for a given set of sequences, the true phylogeny may not be known, so there is no gold standard to compare results in computing the accuracy. Simulations allow the computation of synthetic datasets from a given tree topology, so the true topology is thus known. Second, we would often like to test algorithms for sequences with specific properties. By constructing synthetic sequence datasets, the tester can manipulate factors such as evolutionary rate or overall evolutionary distance, while controlling for other factors that could influence the analysis. Finally, when considering the use of an algorithm on increasingly large datasets, real data of the appropriate size may not be available or continuous over a range in question. Fortunately, synthetic datasets can be constructed for any given size of interest. Of course, simulated data may not respond in the same manner as real data in the analysis and should be verified whenever possible. Many researchers have noted that optimal trees appear to be more easily found for synthetic data as compared to real data (Williams et al. 2006).

Genes Versus Genomes

Most sequence information in biological databases falls into a bimodal distribution, with complete genomes from individual organisms in one peak and individual genes from many organisms in the second peak. The former group is characterized by relatively few sequences on the order of 10^6 or more bases, while the latter is characterized by many sequences on the order of 10^3 or fewer bases. This discrepancy means that algorithms that work well for one of these groups may not work well for the other group. The main reason is that as the length of sequences increases, so does the ease with which an accurate phylogeny is returned. In contrast, as the number of taxa increases, the accuracy of the optimal tree generally declines. Therefore, different algorithms for trees from long sequences (genomes) are more likely to comparable results, but differ in computational complexity. In this situation, the user is most likely to select the fast algorithm due to little sacrifice in accuracy. Furthermore, building phylogenies for genomes is likely to be more restricted by multiple sequence alignment problems. For shorter sequences (genes) and large numbers of taxa, there is a trade-off between speed and accuracy that demands optimization. Although building phylogenies for large numbers of genomes is likely to be of interest in the future, this review pertains more to the immediate problem of constructing phylogenies from large numbers of short sequences (< 2kb).

Horizontal Gene Transfer

Evolution of eukaryotic organisms is dominated by vertical descent of genetic information from parents to progeny. In some bacteria and archaea, the evolutionary process can be greatly influenced by the horizontal transfer of genes among often distantly related species or even genera. The result of this process is that gene sequences from one bacterium may be more closely related to sequences from another species than to sequences within its own species. If ignored, horizontal gene transfer (HGT) can confound phylogenetic inference and lead to false conclusions. An important distinction must be made as to how to handle phylogenies for one or more genes in organisms with high rates of HGT. Phylogenetic trees may be used for individual genes or short sequences without violating the assumption that all parts of the sequence evolved together in a vertical fashion. By comparing these trees for several short sequences, it may be possible to detect HGT between organisms and identify which genes do not match the evolutionary history of the rest of a genome. In contrast, phylogenetic trees cannot be used to construct phylogenies from longer sequences that contain a subset of horizontally transferred genetic information. Under these circumstances, the user must employ techniques that allow separate evolutionary paths for different parts of a sequence (Jin et al. 2006). Several tools using modifications of the algorithms described in this review will construct a phylogenetic network for such data instead of a tree. For a review of these methods see (Morrison 2005).

Choosing A Method for Phylogenetic Inference

The three main categories of algorithms for building phylogenies described above each have benefits and shortcomings that will influence the choice of a particular algorithm for a given dataset. All of the methods for constructing phylogenetic trees require the user to make assumptions about the evolutionary processes involved in generating the dataset of interest (Felsenstein 1988). We are still learning how changes in many factors affect the behavior of these algorithms and are incrementally increasing our knowledge of how reliable a given method can be. Simulation studies have greatly increased our ability to test the effects of different factors and evaluate the accuracy and efficiency of different methods. Studies have shown that distance methods perform optimally for sequences with larger evolutionary distances relative to the optimal datasets for maximum parsimony (Guindon and Gascuel 2003). This is primarily due to the fact that the number of informative sites underestimates the actual number of evolutionary changes as the evolutionary distances between sequences increases. Simulations indicate that criteria-based methods such as MP and ML are more likely than ME methods to find an optimal tree approximating the true tree, despite the fact that the former algorithms generally operate on a heuristic search rather than extensive searches for the optimal tree (Pupko et al. 2000; Guindon and Gascuel 2003). MP methods also have been shown to suffer when sequences have evolved at different evolutionary rates (Williams et al. 2006).

Evolutionary Models of Sequence Divergence

The construction of any distance-based or maximum likelihood phylogeny requires the assumption of a model for how the sequences analyzed changed over time. The simplest models (uncorrected “p”, Jukes-Cantor) assume that all substitutions have equal probability at all sites, and despite the fact that these assumptions are probably invalid, these models behave almost equally well as more complicated models (Takahashi and Nei 2000; Bininda-Emonds et al. 2001). However, analysis of real sequence datasets has revealed the presence of heterogeneity in substitution rates from site to site and also from sequence to sequence. To improve on the error and bias found in these models, Ninio et al. created iterative Bayesian methods that estimate evolutionary rates across the entire tree by using all sequences and the current estimate of the phylogeny to estimate the rates

(2007). Despite modifications in models of substitution that better represent real sequence data, there is still plenty of room for improvement in this area of research.

Minimum Evolution Algorithms

The neighbor-joining (NJ) method is an agglomerative method, which progressively builds a tree by connecting the two sequences with the smallest distance, starting at the leaves of the tree and progressing along internal nodes toward the root (Saitou and Nei 1987). While the presence of a single optimal tree for the specified distance matrix keeps the computation time short, the mistaken mispairing of sequences early in the agglomeration can significantly reduce the accuracy of the tree. This fact makes the distance-based ME methods highly susceptible to improper use of a model of evolution for the sequences. Topological rearrangements of ME trees can overcome the shortcoming of agglomerative methods by testing trees with swapped branches for improvement in total branch length minimization over the original tree. The major benefit to using NJ is the computational speed with which a dataset can be analyzed: on the order of n^3 for computing the initial tree and pn^3 for branch-swapping, where n is the number of sequences and p is the number of rearrangements (Desper and Gascuel 2002).

Although, NJ was originally thought to operate by minimization of the ordinary least squares (OLS) estimate of the tree length, Gascuel and Steel showed that NJ instead performs a local optimization in selecting sequence pairs that is comparable to heuristic searches in other methods (2006). However, the same study notes that algorithms truly optimizing the OLS criteria do not necessarily have more accurate topologies (Gascuel and Steel 2006). As a faster alternative to NJ, Desper and Gascuel designed a new algorithm called FastME that operates on a “balanced minimum evolution” criteria, where subtrees have equal weights irrespective of the number of sequences contained in each, as opposed to the OLS criteria that weighs each sequence equally (2002). Gascuel showed through simulations that this balanced criteria was better suited than OLS to biological sequence data (2000). This intuitively makes sense since OLS assumes equal variances in all sequences, a principle known to be false. Simulations with FastME, NJ, and other ME-based algorithms showed the greedy and balanced ME algorithms with balanced nearest-neighbor interchanges (BNNI) to outperform the other methods in terms of topological accuracy (Desper and Gascuel 2002). The greatest contribution from this method is the BNNI algorithm for branch-swapping, which greatly improved tree topology in all ME methods tested and had a computational complexity on the order of n^2 for most datasets. FastME (GME + BNNI) improves on the computational speed of NJ by only requiring a calculation of the tree length after the final topology has been reached, and scales much better than NJ for increasing number of sequences.

Maximum Likelihood Methods

The maximum likelihood criterion is arguably the most accurate means for building a phylogenetic tree. Unfortunately, the methods employing this criterion are also computationally very complex and require long execution times for any datasets containing more than one dozen sequences. Recently, several faster algorithms using ML have been proposed, and two prominent examples will be described here. In 1994, Olsen et al. introduced fastDNAml (1994), which made several modifications to an existing algorithm by Joe Felsenstein (fastML) to increase its computational efficiency. First, the authors substituted an unproven, but empirically good and fast algorithm for optimizing branch lengths. Second, optimization of branch lengths was changed from focusing on individual branches to optimizing all branches in a tree simultaneously. Finally, locations of new added sequences are chosen without reoptimizing the tree, rather the tree is refined after the new branch is added. The modifications reduced computation time with no observed decrease in accuracy, allowing up to 100 sequences to be analyzed in a reasonable time frame. This most recent version is implemented as part of PHYLIP (Felsenstein 1989).

Although fastDNAml was able to reduce the overall computation time for ML datasets, practically this method is still limited to datasets of 100 sequences or fewer. Guindon and Gascuel realized that a large contribution to the long computation times is due to the separate optimization of branch lengths and topology (2003). By simultaneously optimizing topology and branch lengths, these authors were able to maintain topological accuracy while vastly improving the speed with which the analysis is completed for large numbers of sequences. Their improved ML algorithm is implemented in the PHYML package (Guindon et al. 2005). First, a tree is constructed for the dataset based on a rapid distance-based algorithm. Then, the maximum likelihood of the tree is maximized through the independent adjustment of branch lengths and branch swapping. Construction of the initial tree takes on the order of n^3 time, and the likelihood optimization step for topology and branch lengths is on the order of *pns* time, where p is the number of refinements, n is the number of taxa, and s is the sequence length. Since p is always much smaller than n, this step does not take significantly longer than most distance-based tree building algorithms. Indeed, when tested on synthetic datasets of 100 taxa and 500 base pairs in length, PHYML had an average run time of 12 seconds, while NJ and fastDNAml had average computing times of 2.3 seconds and 32.6 minutes, respectively (Guindon and Gascuel 2003). The PHYML algorithm also computed a ML tree for 500 plant plastid sequences (1428 bp) in less than 12 minutes. These results demonstrate that modifications to the ML algorithms can make them nearly as fast as other distance-based methods, while providing a more accurate depiction of the "true" topology.

Disk-Covering Methods

Recently, Tandy Warnow and colleagues have created a suite of programs called Disk-Covering Methods (DCMs) that can be coupled with existing tree building programs to aide in more rapid convergence to the optimal tree through a divide-and-conquer approach. DCMs operate by first decomposing the dataset into a number of overlapping subsets of sequences called disks, then building trees for each subset using any phylogenetic tree program, and finally constructing a supertree from the amalgamation of the subset trees using a strict consensus merger. Consequently, the DCMs can be seen as boosters of the phylogenetic tree methods and are not restricted to any single optimization criterion.

The first version, DCM1, was designed to boost distance methods and operated by choosing one optimal supertree from many supertrees generated for different decompositions of the dataset (Huson, Nettles, and Warnow 1999). Simulations showed that combining DCM1 with neighbor-joining resulted in highly significant reductions in both false positives and false negatives compared to NJ alone, especially for shorter sequences. Warnow et al. were also able to prove that a specific sequence length will result in high probability of finding the true tree for Markov models of evolution (2001). However, while DCM1 does produce small rapidly solvable subproblems, the structure of the decomposition was found to be poor in some circumstances (Roshan et al. 2004).

To correct this, the second version, DCM2, used a fixed structure to decompose the dataset (Huson, Vawter, and Warnow 1999). DCM2 was designed to improve heuristic searches using MP, and when combined with PAUP* with TBR branch swapping resulted in improved accuracy with less computation time on simulated datasets. This method was also combined with GRAPPA, a direct optimization method for gene order data (Tang and Moret 2003). GRAPPA is computationally expensive, so it is only feasible for subproblems of fewer than 15 taxa. Optimization with DCM2 allowed for large synthetic datasets to be analyzed, and improved on NJ and DCM-NJ over a range up to 640 taxa, particularly for higher evolutionary rates of change.

In light of the shortcomings in the first two versions, Roshan et al. introduced DCM3, which chooses a decomposition of the dataset based on a consistently updated guide tree (2004). This change results in small subproblems that are directed within the tree space closest to the current phylogeny estimate. The authors also combined DCM3 with a recursion step that reduces subproblems to a manageable size and an iteration step that successively refines the guide tree toward the optimal topology (Roshan et al. 2004). This new booster algorithm, Rec-I-DCM3, was combined with MP and shown to statistically outperform the maximum parsimony method TNT over all tested datasets and time intervals. Detsenko et al. (2006) showed that running Rec-I-DCM3 in a parallel master-slave model produced better MP scores in a fraction of the time, with the optimal arrangement being 8 worker CPUs.

Summary and Conclusions

The optimization of criteria for both maximum parsimony and maximum likelihood becomes prohibitively time consuming as the number of sequences analyzed increases. Coupling of branch length and topology optimization in the PHYML package appears to make this implementation of maximum likelihood competitive in terms of computational time required. Additional improvements may be possible by merging this approach with one of the disk-covering methods described above. Another benefit of this type of optimization approach is that the model of evolution can also be modified during the procedure to better represent the model reflected by the resultant topology. This flexible model of evolution will probably be a much better estimate of the true evolutionary mechanisms, and can be tailored to each dataset, instead of trying to fit one model to all the genes studied. This is especially important since sequence representation in databases used for generating models will definitely have bias toward certain gene families.

Currently, the best approaches for reducing computational complexity involve the decomposition of the dataset into smaller overlapping subsets that can be solved individually. Recursion with DCM3 provides the most promising technique for decomposition by using the most recently updated guide tree to divide the data into appropriately sized subproblems. Constructing and merging the resulting subtrees into a consensus supertree may also provide a bottleneck in the computation time required, especially for a small percentage of the subtrees that are difficult to optimize (Tang and Moret 2003). A primary advantage of the PHYML and DCM methods, which should be considered for all new algorithms, is the ability to escape local optima and converge quickly on the globally optimal tree.

Future improvements on the DCM-based methods should focus on identifying the most difficult subproblems and possibly avoiding them. This could involve a new local decomposition that results in overlapping subproblems of lower complexity or relaxing the optimization criteria. Alternatively, it may be possible to iteratively solve the subproblems using a simpler criterion and providing an estimate of the statistical significance of the branching structure in the trees. In other words, it may not be worth the time to find an optimal solution, if we can instead find a near-optimal solution with bounds for our confidence in the given topology. The combination of iteration and recursion will be necessary for high accuracy and efficiency of any method building phylogenies for large datasets, since the effect of these processes is to avoid local optima and narrow the search to the best subsets of the tree space. When possible, multiple CPUs running in parallel can greatly improve efficiency of the tree building programs, especially since it is complementary to the divide-and-conquer approach of the DCMs. The use of supertree methods for assembling divide-and-conquer subtrees have become popular in newly designed methods for phylogenetic analysis (Sanderson and Driskell 2003), but the robustness of supertree construction should be further evaluated to ensure all elements of the phylogenetic tree are properly represented.

Finally, the use of simulated datasets has been tantamount to the verification of phylogenetic inference methods for large datasets and for testing the impact of individual factors to the accuracy of such methods. However, the usability of any method is dependent on its robust performance on various different types and quantities of real sequence data. Therefore, it is of utmost importance that simulated datasets accurately represent the characteristics of real data. As the amount of sequence data in public databases increases, a concerted effort should be made to construct large phylogenetic trees from this data and to use these actual topologies when generating simulated data for use in verification studies.

References

Bininda-Emonds, O. R., S. G. Brady, J. Kim, and M. J. Sanderson. 2001. Scaling of accuracy in extremely large phylogenetic trees. Pac Symp Biocomput:547-558.

Desper, R., and O. Gascuel. 2002. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. J Comput Biol **9**:687-705.

Felsenstein, J. 1989. PHYLIP -- Phylogeny Inference Package (Version 3.2). Cladistics **5**:164-166.

Felsenstein, J. 1981. Evolutionary trees from DNA sequences: a maximum likelihood approach. J Mol Evol **17**:368-376.

Felsenstein, J. 1988. Phylogenies from molecular sequences: inference and reliability. Annu Rev Genet **22**:521-565.

Fitch, W. M. 1971. Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology. Systematic Zoology **20**:406.

Gascuel, O. 2000. On the optimization principle in phylogenetic analysis and the minimum-evolution criterion. Mol Biol Evol **17**:401-405.

Gascuel, O., and M. Steel. 2006. Neighbor-joining revealed. Mol Biol Evol **23**:1997-2000.

Guindon, S., and O. Gascuel. 2003. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. Syst Biol **52**:696-704.

Guindon, S., F. Lethiec, P. Duroux, and O. Gascuel. 2005. PHYML Online--a web server for fast maximum likelihood-based phylogenetic inference. Nucleic Acids Res **33**:W557-559.

Huson, D. H., S. M. Nettles, and T. J. Warnow. 1999. Disk-covering, a fast-converging method for phylogenetic tree reconstruction. J Comput Biol **6**:369-386.

Huson, D. H., L. Vawter, and T. J. Warnow. 1999. Solving large scale phylogenetic problems using DCM2. Proc Int Conf Intell Syst Mol Biol:118-129.

Jin, G., L. Nakhleh, S. Snir, and T. Tuller. 2006. Maximum likelihood of phylogenetic networks. Bioinformatics %R 10.1093/bioinformatics/btl452 **22**:2604-2611.

Morrison, D. A. 2005. Networks in phylogenetic analysis: new tools for population biology. Int J Parasitol **35**:567-582.

Nakhleh, L., B. M. Moret, U. Roshan, K. St John, J. Sun, and T. Warnow. 2002. The accuracy of fast phylogenetic methods for large datasets. Pac Symp Biocomput:211-222.

NCBI. 2007. NCBI-GenBank Flat File Release 158.0.

Ninio, M., E. Privman, T. Pupko, and N. Friedman. 2007. Phylogeny reconstruction: increasing the accuracy of pairwise distance estimation using Bayesian inference of evolutionary rates. Bioinformatics %R 10.1093/bioinformatics/btl304 **23**:e136-141.

Olsen, G. J., H. Matsuda, R. Hagstrom, and R. Overbeek. 1994. fastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. Comput. Appl. Biosci. %R 10.1093/bioinformatics/10.1.41 **10**:41-48.

Pupko, T., I. Pe'er, R. Shamir, and D. Graur. 2000. A fast algorithm for joint reconstruction of ancestral amino acid sequences. Mol Biol Evol **17**:890-896.

Roshan, U. W., B. M. Moret, T. Warnow, and T. L. Williams. 2004. Rec-I-DCM3: a fast algorithmic technique for reconstructing large phylogenetic trees. Proc IEEE Comput Syst Bioinform Conf:98-109.

Saitou, N., and M. Nei. 1987. The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol **4**:406-425.

Sanderson, M. J., and A. C. Driskell. 2003. The challenge of constructing large phylogenetic trees. Trends Plant Sci **8**:374-379.

Sokal, R. R., and C. D. Michener. 1958. A statistical method for evaluating systematic relationships. University of Kansas, Lawrence, KS.

Takahashi, K., and M. Nei. 2000. Efficiencies of fast algorithms of phylogenetic inference under the criteria of maximum parsimony, minimum evolution, and maximum likelihood when a large number of sequences are used. Mol Biol Evol **17**:1251-1258.

Tang, J., and B. M. Moret. 2003. Scaling up accurate phylogenetic reconstruction from gene-order data. Bioinformatics **19 Suppl 1**:i305-312.

Warnow, T., B. M. Moret, and K. St John. 2001. Absolute convergence: True trees from short sequences. Pp. 186-195. Proc. 12th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA'01). SIAM Press.

Williams, T. L., D. A. Bader, M. Yan, and B. M. Moret. 2006. Chapter 16: High-performance phylogeny reconstruction under maximum parsimony. Pp. 369-394 *in* A. Y. Zomaya, ed. Parallel Computing for Bioinformatics and Computational Biology. John Wiley & Sons.