Genetic Algorithms and Parallel Processing in Maximum-Likelihood Phylogeny Inference

Matthew J. Brauer,[1] *Mark T. Holder,†* *Laurie A. Dries,*[2] *Derrick J. Zwickl,** *Paul O. Lewis,†* *and David M. Hillis**

*Section of Integrative Biology and Center for Computational Biology and Bioinformatics, University of Texas; and †Department of Ecology and Evolutionary Biology, University of Connecticut

We investigated the usefulness of a parallel genetic algorithm for phylogenetic inference under the maximum-likelihood (ML) optimality criterion. Parallelization was accomplished by assigning each ''individual'' in the genetic algorithm ''population'' to a separate processor so that the number of processors used was equal to the size of the evolving population (plus one additional processor for the control of operations). The genetic algorithm incorporated branch-length and topological mutation, recombination, selection on the ML score, and (in some cases) migration and recombination among subpopulations. We tested this parallel genetic algorithm with large (228 taxa) data sets of both empirically observed DNA sequence data (for angiosperms) as well as simulated DNA sequence data. For both observed and simulated data, search-time improvement was nearly linear with respect to the number of processors, so the parallelization strategy appears to be highly effective at improving computation time for large phylogenetic problems using the genetic algorithm. We also explored various ways of optimizing and tuning the parameters of the genetic algorithm. Under the conditions of our analyses, we did not find the best-known solution using the genetic algorithm approach before terminating each run. We discuss some possible limitations of the current implementation of this genetic algorithm as well as of avenues for its future improvement.

Introduction

With the advent of large-scale sequencing projects, the amount of phylogenetically useful molecular data has increased by several orders of magnitude. There are intrinsic advantages to using large amounts of data in a phylogenetic analysis, both in terms of number of taxa and number of characters (Wheeler 1992; Lecointre et al. 1993; Hillis, Huelsenbeck, and Swofford 1994; Hillis 1996, 1998; Pollock et al. 2002; Zwickl and Hillis 2002). In addition, many researchers would like to analyze their data with a computationally intensive, model-based optimality criterion such as maximum likelihood (ML). These trends suggest a need for an analytical method that can be scaled to accommodate phylogenetic problems with any number of taxa and characters. One possible way of accomplishing this goal is to develop parallel and distributed algorithms for ML analysis. These methods can potentially take advantage of both the increasingly common dispersed computer resources and the ''big iron'' of supercomputer centers. A natural candidate for such a method is one based on a genetic algorithm.

In nonphylogenetic domains, genetic algorithms have been used to find near-optimal solutions to complex problems in a rapid and efficient manner. A genetic algorithm mimics the adaptation of a ''population'' by including processes analogous to mutation, recombination, and selection. Each ''individual'' in the evolving population is a potential solution to a problem, and the individual's fitness is determined by how well it solves the problem. We used GAML (Lewis 1998), a genetic algorithm designed to solve the problem of finding a phylogenetic tree that maximizes the likelihood of a data set, to investigate the usefulness of applying a parallel approach to genetic algorithms for ML phylogenetic analysis. In this phylogenetic application of a genetic algorithm, an individual in a population is a hypothesis consisting of the tree, branch lengths, and parameter values for the model of sequence evolution; the fitness of the individual is the likelihood score of the hypothesis. The ML score of a tree is usually approximated by a computationally intensive hill-climbing process in which the values of many nuisance parameters must be estimated. GAML uses the genetic algorithm approach to optimize these parameters and to simultaneously search through tree space. In the original article in which the method was described, Lewis (1998) applied GAML to a real data set of 55 sequences. GAML correctly found the best tree and performed substantially faster than a traditional heuristic search in which all parameters were optimized for each tree examined.

The genetic algorithm approach implies an obvious strategy for parallelizing the search process. Each individual in a population can be handled by a single processor or node, which calculates the likelihood of the respective individual (tree, branch lengths, and parameter values). Because this operation takes the largest amount of wall-clock time, an increase in population size up to the number of available nodes should not substantially increase the amount of time needed to solve the problem. Thus, with an efficient parallel algorithm, searches can be scaled to run on either massively parallel processors or on workstation clusters such as Beowulf (Becker et al. 1995) or Legion (Grimshaw and Wulf 1997).

[1] Present address: Department of Genetics, Stanford University School of Medicine.

[2] Present address: Department of Biological Sciences, Ohio University.

Key words: genetic algorithm, phylogeny inference, maximum likelihood, parallel algorithm, nucleotide sequence data.

Address for correspondence and reprints: David M. Hillis, Section of Integrative Biology, University of Texas, Austin, Texas 78712. E-mail: dhillis@mail.utexas.edu.

Mol. Biol. Evol. 19(10):1717–1726. 2002

As with other heuristic search strategies, the genetic algorithm approach does not guarantee finding the best solution; the parameters chosen for the genetic algorithm affect its efficiency and the extent of solution-space covered. Our goal in this study was to evaluate the accuracy and efficiency of a parallelized genetic algorithm in solving very large (>200 taxa) ML problems in phylogenetics. This parallel version of GAML allowed us to examine the performance of the method for a data set whose size and complexity place it beyond the normal range of most other heuristic methods.

Materials and Methods

The Program

We examined the performance of a parallelized version of GAML for solving a large phylogenetic problem, given a range of program parameters. We used the August 1998 revision (pre-1.3) version of GAML modified and compiled to run on the Cray T3E (Lonestar) at the Texas Advanced Computing Center. Lonestar employs a scalable number (88–277 during our study) of Alpha EV5 RISC processors in a three-dimensional toroidal memory interconnect, with 128 Mb of memory per node (further technical specifications of the Cray T3E can be found at the website http://www.tacc.utexas.edu). Parallel routines were written using the message-passing interface, so the implementation is portable to other architectures and operating systems. The program uses the common HKY85 model (Hasegawa, Kishino, and Yano 1985), with inferred transition to transversion rate ratios and empirical base frequency estimates. Other comparative analyses were done as noted using the program PAUP* 4.02b (Swofford 2000).

In this parallel version of GAML, a master processor creates a population of individuals and sends each of them to another single processor to be scored. Thus, the number of nodes used equals the number of slave nodes (equal to the population size) plus one master node. During each run, all nodes used were dedicated to this analysis. Each of the slave processors calculates the likelihood of the tree, given a set of parameters. This value becomes the basis for the rank ordering of the population which determines each individual's fitness. The next generation of the population is created by copying the best individual in the current generation a set number of times (referred to as the holdover number). The remaining individuals that will contribute to the subsequent generation are stochastically selected, with the probability of any given individual being selected set by its rank. Next, all individuals, except one copy of the best individual, are altered by mutations in branch length, a change in topology (using a subtree pruning-regrafting operation), or recombination with another member of the population (or all).

Individual runs were stopped on the basis of a subjective criterion incorporating the size of likelihood increase per unit time and the current availability of computational resources. In general, most runs took between 3 and 10 million node-seconds.

Data Sets

Two data sets of approximately equal size were analyzed. The first (simulated) was a Monte Carlo simulation of 5,000 nucleotide characters across a model tree of 228 taxa derived from the most parsimonious solution of a large angiosperm data set (Soltis et al. 1999). Data were simulated using the program Siminator (Huelsenbeck, Hillis, and Jones 1995), using parameters that correspond closely to those estimated for the original data set. As described by Hillis (1996), this simulation used a K2P model with γ-distributed rate variation (the shape parameter α set to 0.5) and a transition-transversion ratio of 2. For this data set, the model tree represents the correct solution to the problem, although not necessarily the tree with the highest likelihood. For the second (observed) data set, 4,822 aligned nucleotides of 12S and 16S mitochondrial ribosomal RNA genes were taken from the same 228 angiosperm taxa that formed the basis of the simulation (Soltis et al. 1999).

Variation of Program Parameter Values

The GAML program allows variation in the parameter values assigned to population size, holdover number, recombination probability, the probability of mutating the tree topology, and a shape distribution for the size of branch length and transition:transversion ratio mutations (these parameters are described at length by Lewis 1998). We serially varied population size, topological mutation rate, and recombination rate to investigate the importance of these parameters on the efficiency of the genetic algorithm. The parameter that might most obviously affect performance is the size of the population. We varied population size among the values 3, 6, 15, and 30. A decline in the ratio of performance–processor time was evident after 15 individuals, so increases in population size much beyond 30 are unlikely to result in dramatic increases in performance above that expected by the simple increase in total processing power. Furthermore, limitations in the number of nodes available for dedicated analysis precluded much larger population sizes. For all experiments, the holdover number was set at one third of the total population size. This parameter is the minimum number of times that the top-ranked individual is represented in the subsequent generation (see Lewis 1998 for a complete description).

Finally, to establish that the values chosen for the per-generation topological mutation and crossover probabilities were reasonable, we jointly varied these two parameters over a range of values. Initial experiments showed that mutation and crossover probabilities less than 0.1 or greater than 0.8 did not improve performance, so these experiments limited crossover and topological mutation probabilities to values within this range. Crossover and topological mutation rates were thus 0.1, 0.2, 0.4, or 0.8 per generation.

Migration Experiments

To determine the extent to which a loss of population genetic variance might limit the performance of

GAML, we conducted migration experiments, in which two independent populations were first evolved for 66,500 generations (by which point the increase in likelihood score had slowed substantially) and were then mixed, with recombination occurring between individuals of different populations. The resulting hybrid populations were allowed to continue evolution under the original conditions, and variation within and among these populations was monitored.

Evaluating Performance

The performance of any algorithm is a function of the accuracy of its solutions and the efficiency of its resource use. We estimated the accuracy of a phylogenetic solution by its likelihood score and also by its topological proximity to one or more ''reference'' trees representing the best available solutions.

The choice of reference tree depended on the data set used. For experiments using the simulated data set, the actual solution is known—it is the tree used as the model for simulation. The model tree is therefore used as the reference. Measuring progress against this ''true'' reference tree (in contrast to the best ML tree used for the real data set) allows some estimate of the objective performance of the method for a given model of evolution.

The solution for the empirically observed (real) data set is unknown. Furthermore, it is not feasible to calculate with certainty the single optimal ML tree for a 228-taxon problem. For these reasons, the reference was taken to be the best ML tree found for this data set under any search strategy. The best ML tree that we have found for this data set was discovered by optimizing the likelihood scores among the 1,728 best trees found by PAUP* in a maximum parsimony heuristic search. For both simulated and real data sets, solutions were also compared with the last solution (i.e., to the best of the trees found when the analysis was stopped). Note that because the respective reference trees differ, progress is not directly comparable between the two data sets.

Progress toward the reference tree was quantified by monitoring the log-likelihood of the best tree in the population. Additionally, the topological distance from the reference tree was calculated in symmetric distance units (Robinson and Foulds 1981; Penny and Hendy 1985). This index sums the number of internal edges (branches) found in each tree that are not found in the other. The symmetric distance is taken here as a rough correlate of the number of mutations (i.e., nearest-neighbor-interchange operations) needed to move between two trees because no exact algorithm has been developed to calculate this number (Swofford 1991). The maximum symmetric distance between two trees is $2(N - 3)$, where N is the number of taxa in each tree (and $N - 3$ is the number of internal edges).

It is possible for two trees to be similar for most taxa but still have a high value for the symmetric distance statistic (in the extreme case of moving just one terminal taxon from one end of a pectinate tree to the other, the symmetric distance metric will be at its maximum value). To obtain a metric of tree similarity that is less sensitive to this problem, we also calculated the size of the maximal agreement subtree (Goddard et al. 1994). The size of the agreement subtree was generally inversely proportional to the topological distance measured in symmetric distance units.

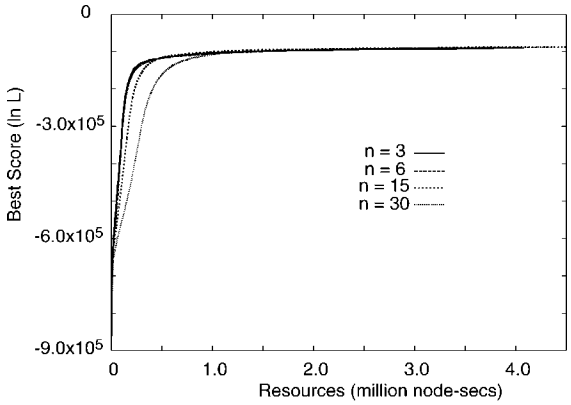

FIG. 1.—Progress of the genetic algorithm across four populations of differing sizes, measured by the log-likelihood of the best solution in the population. Note that the abscissa is measured in millions of node-seconds of processor time. Similar solutions require similar amounts of total processor time for the different population sizes, but the total number of processors used in this parallel algorithm equals the population size plus one. Therefore, the absolute (wall-clock) time for a given analysis decreases in an almost linear fashion with increasing population size.

For experiments in which population size was held constant, we discuss the progress of the algorithm as a function of the number of generations elapsed. When comparing populations of different sizes, however, this measure cannot adequately describe the program's use of computational resources. Therefore, for these experiments, we measured the progress of the algorithm in terms of node-seconds. Number of node-seconds = (elapsed time) × (number of nodes used), or equivalently, (elapsed time) × (population size + 1). All nodes in use were dedicated to the given experiment; hence, the time per generation per node did not change substantially between runs.

The number of node-seconds used takes into account that larger populations distributed across more nodes take more computational resources, as measured by most administrators of supercomputer systems. But if processors are not limiting, total elapsed time is more relevant because it measures the time to completion from the perspective of an end user.

Results and Discussion

For populations of all sizes, the fitness of the best individual in the population follows a characteristic pattern (figs. 1 and 2). There is a rapid increase in the log-likelihood score, followed by a slow approach to an apparent asymptote fairly close to the score of the model or reference tree ($\ln L_{ref} = -139{,}001.17$ for the real data set, $\ln L_{ref} = -84{,}490.0$ for the simulated data set). By this measure, progress toward a solution appears to be

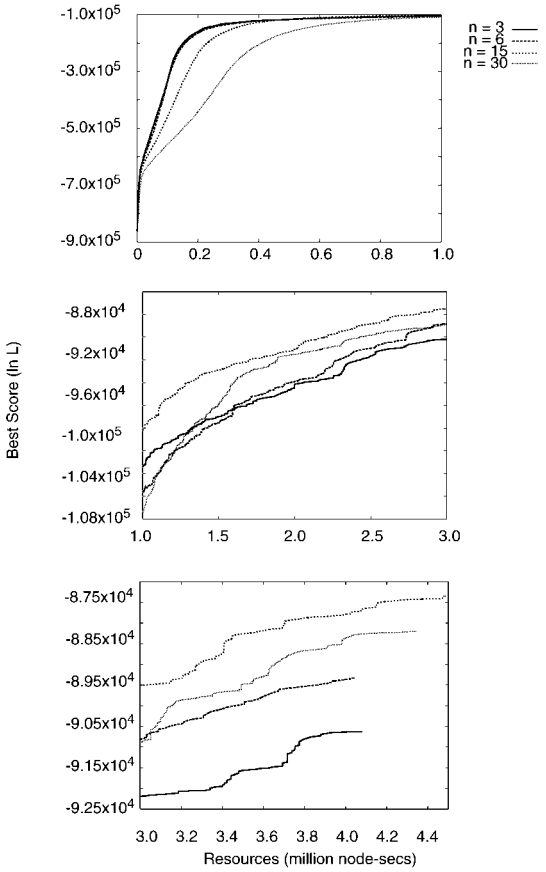

FIG. 2.—Detailed views of the data presented in figure 1, shown across three divisions of the abscissa. Note that there is a slight decrease below linear speedup between a population size of 15 and 30.

extremely rapid in the earliest part of the run, after which it reaches a phase of diminishing returns during which the application of further resources adds little to finding a better solution.

But when the progress of the population is expressed as the topological distance from the reference tree, the conclusion is different. Whether the observed or simulated data sets are being considered, there is a substantial gain in topological accuracy even after the likelihood has neared its likelihood asymptote (fig. 3). Thus, from the standpoint of making an accurate phylogenetic estimate (where accuracy is defined by the number of correct branches in the tree), the algorithm continues to find better solutions, even though the improvement in likelihood scores is relatively small.

Despite this continued improvement, the algorithm never found the best-known solution for either the simulated or real data set during our analyses. After about 58,000 generations a population with a size of 15 was still 94 symmetric distance units away from the reference tree for the simulated data. In the analysis of the observed data set, after about 61,000 generations the best tree found by GAML was still 241 symmetric distance units away from the reference tree (fig. 3).

It is unlikely that the best trees found are topologically very distant from the ''correct'' trees, although it is a possibility. The likelihood scores of the best trees

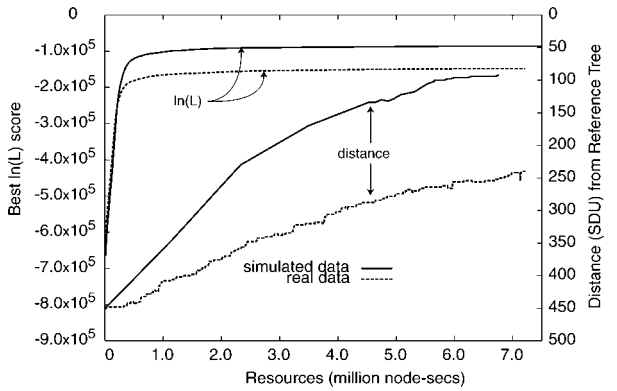

FIG. 3.—Progress of the genetic algorithm for simulated (solid line) and observed (dashed line) data, measured both by log-likelihood of the best solution and symmetric distance between the best solution and the reference tree. Replicate runs (not shown) had nearly identical trajectories.

found by GAML are much lower than those of the reference trees, indicating that GAML did not find a different and better solution. Given that the goal of this study was to examine the effects of parameters on the performance of the algorithm, and not to find the best tree for either of the data sets, the failure to converge after these limited runs is not surprising. In the original demonstration of GAML using a 55-taxon data set, the program found what was believed to be the optimal ML tree in around 8,000 generations.

Observed Versus Simulated Data

It is usually much easier to infer phylogenies from data simulated under any currently used model of substitution than from empirically observed data. Indeed, there is a stark difference in the progress of populations analyzing observed versus simulated data (fig. 3). As measured both by the increase in likelihood and the approach in topology toward the reference tree, progress is much slower for the empirically observed data set. These differences are genuinely due to the different data sets: replicate runs show nearly identical results (not shown).

Effects of Varying Population Parameters

In our experiments, the number of processors used was equal to the population size (of the genetic algorithm); hence, the population size studies are also an assay of how well GAML scales to many processors. Population size did affect the rate at which the optimal fitness is reached (fig. 4). At 4 million node-seconds, a population size of 15 produced the tree with the best likelihood of all of the conditions tried. Fortunately, the population of size 30 was not far behind, indicating that this method of parallelizing the problem works very well. If the parallelization of this scheme were perfect, population size would have no effect and all runs would perform identically with respect to node-seconds. This result is promising because it implies that the amount of time to perform a search should decrease nearly linearly with the number of processors that are used, at

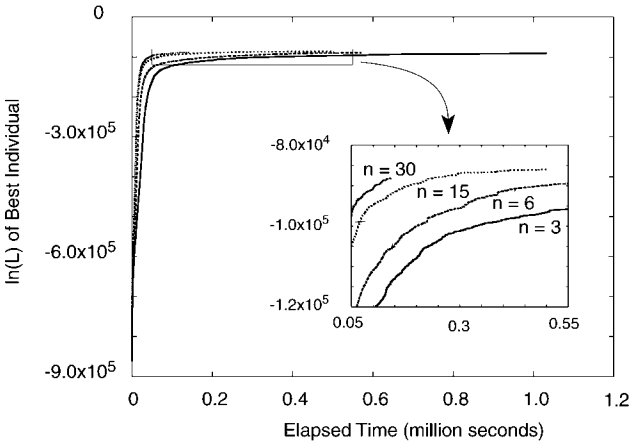

FIG. 4.—Log-likelihood of the best solution versus elapsed (wall-clock) time for four populations of different sizes.

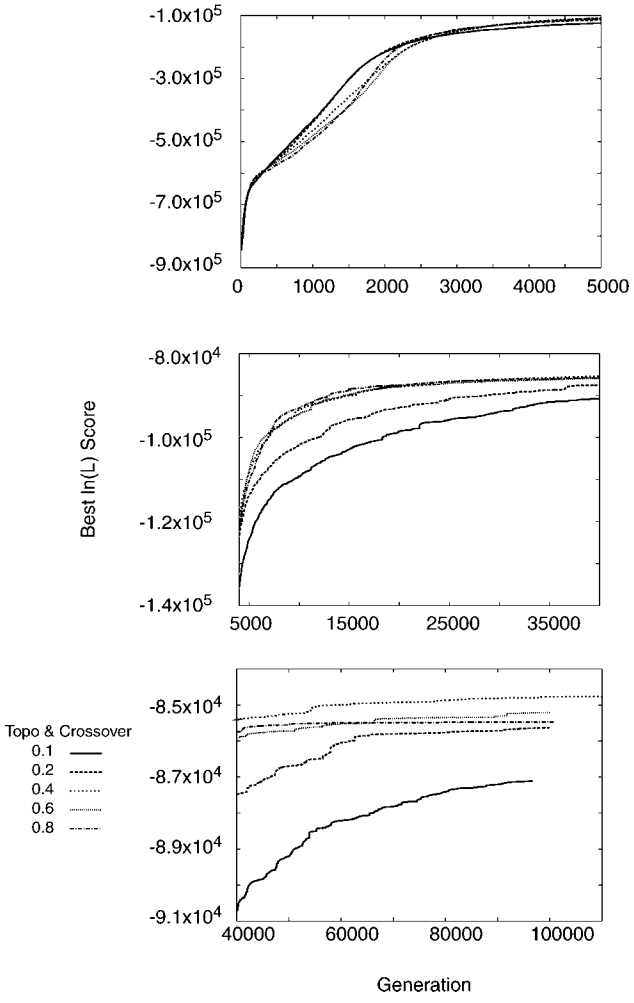

FIG. 5.—Effects of varying mutation and crossover rates on progress of the genetic algorithm. The values for topological mutation rate and crossover rate are absolute probabilities per generation. Population size was 15. Under these conditions one generation corresponds to approximately 117 node-seconds.

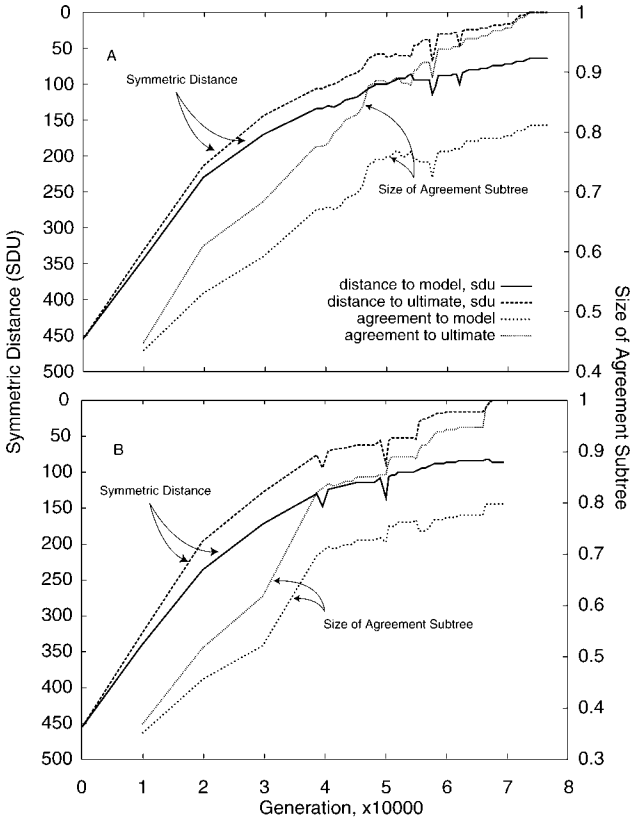

FIG. 6.—Progress of two independent populations (A and B), using simulated data and measured by four indices: (1) symmetric distance between the best tree and the model tree; (2) symmetric distance between the best tree and the final tree found; (3) size of the agreement subtree between the best tree and the model tree; and (4) size of the agreement subtree between the best tree and the final tree found. Mutation and crossover rates were 0.2; population size was 15.

least over a moderate range of processors. As larger clusters of machines become available to biologists, ML searches may become feasible on large data sets, such as the ones investigated in this study. As might be expected, performance of GAML was improved with increasing mutation and crossover rates up to a point. The best performance was obtained from intermediate values: about 0.4 for both (fig. 5). Varying the rate of recombination independent of the mutation rate had little effect, indicating that recombination of individuals was not having a large effect on the performance of the algorithm (not shown). In two replicate runs, the progress (as measured by distance from the reference and ultimate trees) occurred in a similar manner (fig. 6). In all cases where populations of identical size and program parameters were run, the fitnesses of the best solutions in each population followed trajectories that were nearly indistinguishable. As discussed in the next section, this similarity in fitness trajectory is not the result of the populations finding the same sequence of solutions.

The effect of varying holdover number was not examined. Changing this parameter should affect the rate at which a population becomes more homogeneous: a very large value would lead to a near-monomorphic population after one generation. A very small value would result in fewer lower-fitness individuals being

Table 1
Increase in Log-likelihood Score (lnL) as a Result of Branch-Length Optimization

Tree	lnL	Change in lnL
Best tree after 70,000 generations of GAML search (n = 15)	−85,699	—
… after subsequent branch-length optimization	−85,562	137
At the end of TBR search in PAUP*, using the last tree found in GAML as starting tree	−84,491	1,071

displaced. The significance of this parameter to the progress of the program is currently unclear.

Contributions of Branch Lengths to Performance

GAML encodes branch-length data within the ''chromosomes'' (the parameters associated with particular individuals) of the potential solutions. Preferred branch lengths evolve over the course of a run, but the branches are not optimized for every tree. This is in contrast to the heuristic search algorithms of, for example, PAUP*, which applies a branch-length optimization step after every new topology is found. Branch-length optimization is computationally intensive. PAUP* deals with the problem by approximating the branch lengths, and if the score of a new tree is not within a certain percentage (5% is the default setting) of the best tree, the new tree is rejected without wasting resources on further optimization (Rogers and Swofford 1998). Incorporating such a branch-length optimization might increase the rate at which the genetic algorithm approaches the correct solution but at an undetermined per-generation cost in speed.

We examined the degree to which suboptimal branch lengths reduced the likelihood of the best tree found by GAML after 70,000 generations (at a population size of 15) by optimizing the branch lengths in PAUP*. We found that the increase in log-likelihood score due to branch-length optimization was substantial but much less than the increase obtained from a heuristic search in PAUP*, using the last tree found by GAML as the starting tree (table 1). We also examined the effect of branch-length optimization on the distribution of branch lengths across the tree (fig. 7). The average change in the branch length is quite small; before optimization the average branch length for the tree was 0.0050 and the average branch changed less than 0.0005, so on average each branch changed less than 10%. Under the settings of our experiment, 20% of the branches changed each generation. When the branches are close to the optimum (as they are for this tree), improvements from this massive level of mutation will be rare. One way to avoid this effect would be to alter dynamically the mechanism the genetic algorithm uses to mutate branch lengths. During the early logarithmic phase of increase in log-likelihood, the genetic algorithm method of determining branch lengths seems to work quite well. The population starts at a large distance from the optimum, so large changes are most useful. As the population approaches an optimum, the genetic algorithm could lower the probability of branch-length mutation to zero or switch to a Rogers and Swofford (1998) branch-length optimization strategy.

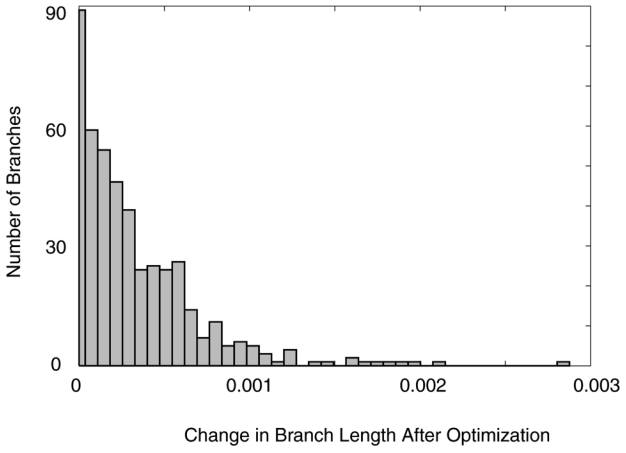

FIG. 7.—The distribution of changes in branch length from optimizing the branch lengths of the last tree found.

Treating branch lengths as parameters to be optimized by the genetic algorithm might make tree estimation more difficult by adding more local optima to the fitness landscape. If one topology has been in the population for a long time, the branch lengths will approach the ML estimates of the branch lengths for that tree. A new topology might have a lower likelihood when it is produced by mutation because most of the branches do not change in length during the branch-swap operation. Because variation is not maintained for long within the population (at least not using the population sizes that we investigated), the new tree may be lost from the population before its branch lengths are sufficiently optimized (see additional discussion regarding this point in the next section). At one point during the analysis with a population size of 15, the best topology did not change for several generations. Using this tree as a starting point of a search in PAUP*, we were able to find a topology that was one subtree pruning-and-regrafting swap away (see Swofford et al. 1996) and had a higher optimized likelihood. But the likelihood of the new tree was lower than that of the previous tree if the branch lengths from GAML were evaluated (table 1). It appeared that the branch-length optimization scheme of GAML had produced a local optimum from which the algorithm might not escape. In subsequent generations, the genetic algorithm did find a better topology, but similar local apparent ''optima'' (that result from not optimizing branch lengths on each topology) may be slowing the search for the best topology toward the end of the search. Salter and Pearl (2001) addressed

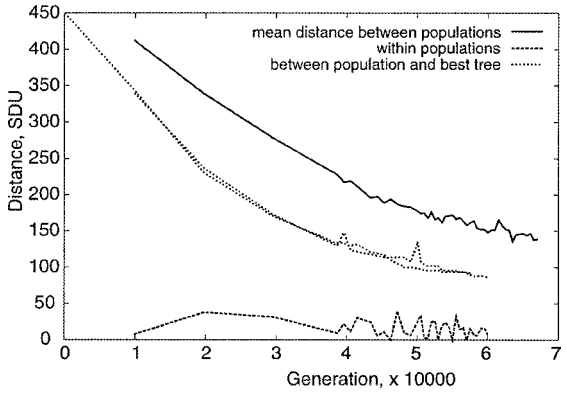

FIG. 8.—Variation within and between two independently evolving populations, measured as average symmetric distance among all solutions. Populations were evolved under identical conditions. Mutation and crossover rates were 0.2; population size was 15.

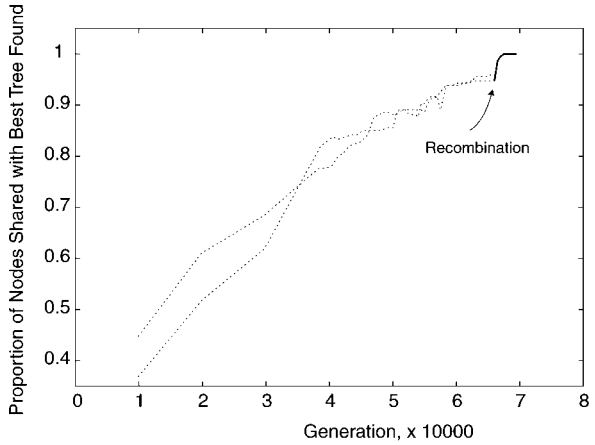

FIG. 9.—Effect of mixing and recombining two independently evolved populations. The arrow indicates the point at which a new population, comprising random samples from the independent populations, was started. Mutation and crossover rates were 0.2; population size was 15.

a similar problem in developing a simulated annealing search strategy to phylogeny reconstruction. In that case, when a topological change occurred, the branches involved in the swap were optimized using the Newton-Raphson optimization (Salter and Pearl 2001). This local optimization may offer an efficient middle ground between random alterations of branches and full ML estimates of the branches.

Variation Within and Between Populations

We monitored the average pairwise distance (in symmetric distance units) within two populations of size 15, as well as the mean distance between the populations, and between the population and the reference tree (fig. 8). Variation within each population is low from early in the run, as might be expected for conditions of such strong selection. Note that at several points the variation within a population is 0, indicating that selection frequently purges the populations of all variation in topology. This may explain why varying the recombination rate had little effect on GAML.

Variation between populations from different analyses remains substantial over the length of the runs, indicating that each population explores a different region of the solution-space. This suggests that it might be useful to evolve independent populations until reasonably stable solution sets are found and then to mix and recombine them. To examine this idea, we allowed two independent populations to evolve to near the fitness plateau, chose 15 random individuals from both, and allowed this mixed population to recombine and evolve further. This experiment showed a dramatic and substantial increase in the accuracy of the solution, taking place at the point at which mixing occurs (fig. 9). Although the result is encouraging, the significance of this increase, its generality across data sets and experimental conditions, and how it can best be used to increase the accuracy of the ultimate solution will require further experiments.

The Stopping Criterion Problem

As with any heuristic search strategy, it is not necessarily clear when the problem is solved. With Bayesian methods, such as those based on Markov-chain Monte Carlo (MCMC) simulation (Rannala and Yang 1996; Larget and Simon 1999), the stability of the distribution of posterior probabilities can be used as the basis of a stopping criterion. Unfortunately, this approach is not available for the genetic algorithm strategy. But our results suggest some logical criteria for stopping a GAML search. The early rapid progress of the genetic algorithm quickly slows as the best solution found approaches an asymptote in log-likelihood score and in distance to the reference tree. At some point, further investments in resources give increasingly small returns, indicating a logical stopping point. The variation among estimates made by independent populations of solutions might be used to evaluate the distance from the ultimate solution, thereby suggesting another possible stopping point. Until the behavior of many independent genetic-algorithm populations is explored, the stopping point will more likely be determined by availability of resources rather than by any objective function of the genetic algorithm performance.

Conclusions

Our results demonstrate the usefulness as well as some limits of the genetic algorithm approach for estimating phylogenetic relationships among many taxa. The performance of this ''first-generation'' parallel version of GAML is particularly encouraging when one considers both the ease of and potential for further scaling of the algorithm and the rapidly increasing availability of computational power. For solving large instances of the phylogeny problem, wall-clock time is likely to be more significant of a limiting resource than of processor time. The fact that the process can be so effectively sped up by addition of processors (fig. 4, for example) indicates that problems of this size or greater

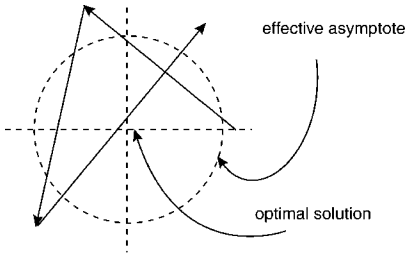

FIG. 10.—Illustration of Fisher's geometrical model and one source of difficulty for a genetic algorithm approaching an optimal solution. Arrows represent phenotypic effects of mutations for a population that is near a phenotypic fitness optimum. If mutational effects remain constant with decreasing distance to the optimum, further increases in fitness will be rare.

may be solved in reasonable amount of time. Although we saw a slightly less than linear speedup from 15 to 30 processors (fig. 2), the nearly linear scaling of computation time (from the standpoint of the end user) with number of available processors across the range we investigated indicates that this approach to parallel genetic algorithms is highly effective.

It is also encouraging that the GAML algorithm progresses toward a better solution throughout what appears as nearly a plateau in likelihood score in later generations, both for observed and simulated data (fig. 3). Despite the dramatic reduction in the rate of likelihood increase for the best solution, the algorithm continues to find trees that are topologically closer to the correct tree. Given additional time, it would be reasonable to expect that the best solutions found here would be greatly improved upon. Below, we discuss a few possible improvements to the algorithm.

Dynamic Mutation Effects

The slowdown in performance of the algorithm is similar in concept to the slowdown in adaptation in Fisher's Geometrical Theory (Fisher 1930, pp. 38–41) (fig. 10). As a population approaches a fitness optimum in a multidimensional phenotypic space, the larger mutations do not bring the population nearer to the optimum. As the complexity of the problem (and the dimensionality of the phenotype-space) increases, the distribution of mutations that are beneficial becomes skewed toward those that are small in effect, relative to the distance of the population to the optimum (Orr 1998). For this reason, a mutation rate and spectrum that are constant will result in an asymptote in fitness. One means of addressing this problem would be to scale the effects of each mutation by the approximate distance to the optimum. This might be done by periodically modifying the branch-length mutation effect so that it is a function of the per-generation change in likelihood scores. In other words, the slope of the likelihood score versus time plot could be used to scale the size of mutation effect.

The Model of Parallelization

One obvious limit to parallelizing a genetic algorithm relates to the redundancy of calculations. If the variation among solutions is low, that is, if most of the potential solutions are similar or identical, then the processors will in part be duplicating each other's calculations. Furthermore, there will be redundancy in transmitting the tree specifications, as well as the results of the likelihood calculations. Because communication among processors is often slow relative to computational time, this transmission of duplicate data may significantly slow the process.

Furthermore, in the scheme described here, all the processors operate on all the data simultaneously. This means that the efficiency of memory use does not scale with the number of processors. As data sets grow in size, it may become necessary to partition the data by subsets of characters so that each processor works on only a subset of the total data. For example, if each of n processors were given $1/n$ columns of the data matrix, the efficiency of memory use would then scale linearly with processor number. In this case, each processor would calculate the sum of the log-likelihoods for each of its characters for a given tree. The log-likelihood for the tree would then be the sum of log-likelihoods gathered from each of the processors. The parallel implementation of a ''gather-with-sum'' operation is very efficient in comparison with the processor to processor communication required to transmit the individual results of a processor's calculation. But the number of communication operations per tree would increase from one to the number of processors used. There would be disadvantages to dividing the problem by groups of characters, as described above. The processors would not be operating independently and would have to be synchronized at each generation. Branch-length optimization algorithms use likelihood information from all characters; if this procedure were added to a program parallelized by character, the communication costs would be high. Parallelization by characters might therefore be more appropriate for a large parallel computer, whereas the ''classic'' method, implemented in GAML, is probably better suited for a highly distributed network of workstations.

Another way that parallelization might be achieved is by separating the populations into several moderately sized demes with each processor controlling an entire population. The performance gain from recombining among populations does not appear to be phenomenal, but repeated introgression of independently evolved solutions may speed up the progress toward the optimum. A further advantage is that relatively little communication would be required: each processor could run several generations uninterrupted and then report back to a server to receive migrants before continuing their evolution.

Performance Relative to Other Approaches

GAML's approach to phylogeny reconstruction is sufficiently distinct from either the heuristic methods of PAUP* or the MCMC methods of BAMBE (Larget and Simon 1999) and MrBayes (Huelsenbeck and Ronquist 2001) that substantial performance differences might be expected. For estimates to approximate independence,

MCMC methods must discard the vast majority of the sampled solutions. These methods also discard an initial set of solutions (those found in the ''burn-in'' period) that can be quite large. As with genetic algorithms and other heuristic methods, the optimum program parameters must be determined empirically for an MCMC method. The costs of these factors relative to those incurred by the use of a genetic algorithm have not been determined, and the relative performance of these methods is not known.

The potential benefits of parallelizing genetic algorithms of course apply to MCMC methods as well. Determining whether the latter can be as easily and efficiently adapted to parallel architectures as can genetic algorithms, and the extent to which this affects overall performance, will require further research.

Statistical Confidence and Hypothesis Testing

The GA approach to statistical hypothesis testing in phylogenetic inference will probably remain limited to the bootstrap methods developed for other heuristic tree-finding algorithms (Hillis and Bull 1993; Huelsenbeck, Hillis, and Jones 1995). In contrast to MCMC methods, genetic algorithms do not retain and use intermediate solutions for quantitative estimates of a posterior probability density function. The results of a bootstrap resampling using heuristic procedures thus lack the clear statistical interpretation obtained from results of Bayesian estimation procedures. In addition, the performance trade-offs of a Bayesian method versus multiple bootstrap runs of a heuristic method have yet to be determined.

The Future of Genetic Algorithms in Phylogenetics

The class of large-scale problems represented by the angiosperm phylogeny will always require substantial computational resources as long as thorough searches of tree-space are conducted. Although some recent authors have suggested that simple point-estimation methods are adequate to find reasonable estimates of phylogenetic trees under certain limited conditions (e.g., Nei, Kumar, and Takahashi 1998), other studies show that thorough searches of tree space markedly increase phylogenetic accuracy and become increasingly important to estimate accurate trees of many taxa (Zwickl and Hillis 2002). Therefore, efficient and thorough phylogenetic algorithms are increasingly important for the field of phylogenetic analysis. In most cases, finding solutions to large phylogenetic problems is more limited by absolute time than by available computational resources. A requirement for millions of node-seconds of computer time may seem extreme, but many universities and other research institutions have large numbers of computers, in computer labs or on desktops, that remain idle for many hours. Therefore, a program that distributes a phylogenetic problem across hundreds of processors (e.g., during periods when the computers would otherwise be idle) may be feasible, whereas one running 10 times more quickly, but limited to a single very powerful processor, may not be. With computer resources no longer at such a premium and distributed computation schemes becoming more common, parallel algorithms will increasingly be used to capture and exploit a large amount of networked computer resources. Our results indicate that genetic algorithms are particularly well suited for such a distributed environment. The existence of a rich set of possible refinements (e.g., migration among multiple populations, dynamic changes in mutation and recombination rates, and periodic branch-length optimization) indicates that rapid improvements in genetic algorithms for phylogenetic analysis are possible.

Acknowledgments

We thank the National Science Foundation, the National Partnership for Advanced Computational Infrastructure, and the Center for Computational Biology and Bioinformatics at the University of Texas for support of this research. D.J.Z. was supported in part by a fellowship from an NSF IGERT graduate training grant in Computational Phylogenetics and Applications to Biology. We also thank Keith Crandall and two anonymous reviewers for their suggestions on the manuscript.

LITERATURE CITED

BECKER, D. J., T. STERLING, D. SAVARESE, J. E. DORBAND, U. A. RANAWAK, and C. V. PACKER. 1995. Beowulf: a parallel workstation for scientific computation. Pp. 11–14 *in* T. Y. FENG, ed. Proceedings of the 24th International Conference on Parallel Processing (ICPP), Vol. 1. CRC Press, Oconomowoc, Wis.

FISHER, R. A. 1930. The genetical theory of natural selection. Oxford University Press, London.

GODDARD, W., E. KUBICKA, G. KUBICKI, and F. R. MCMORRIS. 1994. The agreement metric for labeled binary trees. Math. Biosci. **123**:215–226.

GRIMSHAW, A. S., and W. A. WULF. 1997. The Legion vision of a worldwide virtual computer. Comm. ACM **40**:39–45.

HASEGAWA, M., K. KISHINO, and T. YANO. 1985. Dating the human-ape split by a molecular clock of mitochondrial DNA. J. Mol. Evol. **22**:160–174.

HILLIS, D. M. 1996. Inferring complex phylogenies. Nature **383**:130–131.

———. 1998. Taxonomic sampling, phylogenetic accuracy, and investigator bias. Syst. Biol. **47**:3–8.

HILLIS, D. M., and J. J. BULL. 1993. An empirical test of bootstrapping as a method for assessing confidence in phylogenetic analysis. Syst. Biol. **42**:182–192.

HILLIS, D. M., J. P. HUELSENBECK, and D. L. SWOFFORD. 1994. Hobgoblin of phylogenetics? Nature **369**:363–364.

HUELSENBECK, J. P., D. M. HILLIS, and R. JONES. 1995. Parametric bootstrapping in molecular phylogenetics: applications and performance. Pp. 19–45 *in* J. D. FERRARIS and S. R. PALUMBI, eds. Molecular zoology: advances, strategies, and protocols. Wiley-Liss, New York.

HUELSENBECK, J. P., and F. RONQUIST. 2001. MRBAYES: Bayesian inference of phylogenetic trees. Bioinformatics **17**:754–755.

LARGET, B., and D. SIMON. 1999. Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. Mol. Biol. Evol. **16**:750–759.

LECOINTRE, G., H. PHILIPPE, H. L. VAN LE, and H. LE GUYADER. 1993. Species sampling has a major impact on phylogenetic inference. Mol. Phylogenet. Evol. **2**:205–224.

LEWIS, P. O. 1998. A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. Mol. Biol. Evol. **15**:277–283.

NEI, M., S. KUMAR, and K. TAKAHASHI. 1998. The optimization principle in phylogenetic analysis tends to give incorrect topologies when the number of nucleotides or amino acids used is small. Proc. Natl. Acad. Sci. USA **95**:12390–12397.

ORR, H. A. 1998. The population genetics of adaptation: the distribution of factors fixed during adaptive evolution. Evolution **52**:935–949.

PENNY, D., and M. D. HENDY. 1985. The use of tree comparison metrics. Syst. Zool. **34**:75–82.

POLLOCK, D. D., D. J. ZWICKL, J. A. MCGUIRE, and D. M. HILLIS. 2002. Increased taxon sampling is advantageous for phylogenetic inference. Syst. Biol. **51**:590–597.

RANNALA, B., and Z. H. YANG. 1996. Probability distribution of molecular evolutionary trees: a new method of phylogenetic inference. J. Mol. Evol. **43**:304–311.

ROBINSON, D. F., and L. R. FOULDS. 1981. Comparison of phylogenetic trees. Math. Biosci. **53**:131–147.

ROGERS, J. S., and D. L. SWOFFORD. 1998. A fast method for approximating maximum likelihoods of phylogenetic trees from nucleotide sequences. Syst. Biol. **47**:77–89.

SALTER, L. A., and D. K. PEARL. 2001. Stochastic search strategy for estimation of maximum likelihood phylogenetic trees. Syst. Biol. **50**:7–17.

SOLTIS, P. S., D. E. SOLTIS, P. G. WOLF, D. L. NICKRENT, S. M. CHAW, and R. L. CHAPMAN. 1999. The phylogeny of land plants inferred from 18S rDNA sequences: pushing the limits of rDNA signal? Mol. Biol. Evol. **16**:1774–1784.

SWOFFORD, D. L. 1991. When are phylogeny estimates from molecular and morphological data incongruent? Pp. 295–333 *in* M. M. MIYAMOTO and J. CRACRAFT, eds. Phylogenetic analysis of DNA sequences. Oxford University Press, New York.

———. 2000. PAUP*: phylogenetic analysis using parsimony (*and other methods). Sinauer Associates, Sunderland, Mass.

SWOFFORD, D. L., G. J. OLSEN, P. J. WADDELL, and D. M. HILLIS. 1996. Phylogenetic inference. Pp. 407–514 *in* D. M. HILLIS, C. MORITZ, and B. K. MABLE, eds. Molecular systematics. 2nd edition. Sinauer Associates, Sunderland, Mass.

WHEELER, W. 1992. Extinction, sampling, and molecular phylogenetics. Pp. 205–215 *in* M. J. NOVACEK and Q. D. WHEELER, eds. Extinction and phylogeny. Columbia University Press, New York.

ZWICKL, D. J., and D. M. HILLIS. 2002. Increased taxon sampling greatly reduces phylogenetic error. Syst. Biol. **51**: 540–550.

KEITH CRANDALL, reviewing editor

Accepted May 27, 2002