Gaphyl: A Genetic Algorithms Approach to Cladistics

Clare Bates Congdon

Department of Computer Science, Colby College, 5846 Mayflower Hill Drive, Waterville, ME 04901, USA
congdon@colby.edu
http://www.cs.colby.edu/~congdon

Abstract. This research investigates the use of genetic algorithms to solve problems from cladistics – a technique used by biologists to hypothesize the evolutionary relationships between organisms. Since exhaustive search is not practical in this domain, typical cladistics software packages use heuristic search methods to navigate through the space of possible trees in an attempt to find one or more "best" solutions. We have developed a system called Gaphyl, which uses the genetic algorithm approach as a search technique for finding cladograms, and a tree evaluation metric from a common cladistics software package (Phylip). On a nontrivial problem (49 species with 61 attributes), Gaphyl is able to find more of the best known trees with less computational effort than Phylip is able to find (corresponding to more equally plausible evolutionary hypotheses).

1 Introduction

The human genome project and similar projects in biology have led to a wealth of data and the rapid growth of the emerging field of bioinformatics, a hybrid discipline between biology and computer science that uses the tools and techniques of computer science to help manage, visualize, and find patterns in this wealth of data. The work reported here is an application to biology, and indicates gains from using genetic algorithms (GA's) as the search mechanism for the task.

Cladistics [4] is a method widely used by biologists to reconstruct hypothesized evolutionary pathways followed by organisms currently or previously inhabiting the Earth. Given a dataset that contains a number of different species (also called taxa), each with a number of attribute-values (also called character states), cladistics software constructs cladograms (also called phylogenies), which are representations of the possible evolutionary relationships between the given species. A typical cladogram is a tree structure: The root of a tree can be viewed as the common ancestor, the leaves of a tree are the species, and subtrees are subsets of species that share a common ancestor. Each branching of a parent node into offspring represents a divergence in one or more attribute-values of the species within the two subtrees. In an alternate approach, sometimes called "unrooted trees" (and sometimes called "networks"), the root of the tree is not

L. De Raedt and A. Siebes (Eds.): PKDD 2001, LNAI 2168, pp. 67–78, 2001.

assumed to be an ancestral state, and drawing the structures as trees seems to be primarily a convenience for the software authors.

Cladograms are evaluated using metrics such as parsimony: A tree with fewer evolutionary steps is considered better than one with more evolutionary steps. The work reported here used Wagner parsimony, though there are other possibilities, such as Camin-Sokal parsimony. (See [10] or [4], for example, for a discussion of alternatives.) Wagner parsimony is straightforward to compute (requiring only a single pass through the tree) and incorporates few constraints on the evolutionary changes that will be considered. (For example, some parsimony approaches require the assumption that characters will be added, but not lost, through the evolutionary process.) An unrooted tree evaluated with Wagner parsimony can be rooted at any point without altering the evaluation of the tree.

The typical cladistics software approach uses a deterministic hillclimbing methodology to find a cladogram for a given dataset, saving one or more "most parsimonious" trees as the result of the process. (The most parsimonious trees are the ones with a minimum number of evolutionary changes connecting the species in the tree. Multiple "bests" correspond to equally plausible evolutionary hypotheses, and finding more of these competing hypotheses is an important part of the task.) The tree-building approach adds each species into the tree in sequence, searching for the best place to add the new species. The search process is deterministic, but different trees may be found by running the algorithm with different random "jumbles" of the order of the species in the dataset.

The genetic algorithm (GA) approach to problem solving has shown improvements to hillclimbing approaches on a wide variety of problems [5,1,9]. In this approach, a population of possible solutions to the problem "breed", producing new solutions; over a number of "generations", the population tends to include better solutions to the problem. The process uses random numbers in several different places, as will be discussed later.

This research is an investigation into the utility of using the genetic algorithm approach on the problem of finding parsimonious cladograms.

2 Design Decisions

To hasten the development of our system, we used parts of two existing software packages. Phylip [3] is a cladistics system widely used by biologists. In particular, this system contains code for evaluating the parsimony of the cladograms (as well as some helpful utilities for working with the trees). Using the Phylip source code rather than writing our own tree-evaluation modules also helps to ensure that our trees are properly comparable to the Phylip trees. Genesis [6] is a GA package intended to aid the development and experimentation with variations on the GA. In particular, the basic mechanisms for managing populations of solutions and the modular design of the code facilitate implementing a GA for a specific problem. We named our new system Gaphyl, a reflection of the combination of GA and Phylip source code.

The research described here was conducted using published datasets available over the internet [2], and was done primarily with the families of the superorder of Lamiiflorae dataset, consisting of 23 species and 29 attributes. This dataset was chosen as being large enough to be interesting, but small enough to be manageable for this initial study. A second dataset, the major clades of the angiosperms, consisting of 49 species and 61 attributes, was used for further experimentation.

These datasets were selected because the attributes are binary, which simplified the tree-building process. As a preliminary step in evaluating the GA as a search mechanism for cladistics, "unknown" values for the attributes were replaced with 1's to make the data fully binary. This minor alteration to the data does impact the meaningfulness of the resulting cladograms as evolutionary hypotheses, but does not affect the comparison of Gaphyl and Phylip as search mechanisms.

3 The Genetic Algorithm Approach

There are many variations on the GA approach[1], but a standard methodology proceeds as follows:

1. Generate a population of random solutions to the problem. (These are not assumed to be particularly good solutions to the problem, but serve as a starting point.)
2. The GA proceeds through a number of "generations". In each generation:
   a) Assign a "fitness" to each solution, so that we know which solutions are better than others.
   b) Select a "parent" population through a biased random (with replacement) process, so that higher fitness solutions are more likely to be parents.
   c) Use operators such as crossover, which combines parts of two parent solutions to form new solutions, and mutation, which randomly changes part of a solution, to create a new population of solutions.

The algorithm terminates after a predetermined number of generations or when the solutions in the population have converged within a preset criterion (that is, until they are so similar that little is gained from combining parents to form new solutions).

Several factors should be evaluated when considering the utility of GA's for a particular problem:

1. Is there a more straightforward means of finding a "best" solution to the problem? (If so, there is no point in using the GA approach.)

[1] As is the custom in the evolutionary computation community, the author distinguishes different forms of evolutionary computation, and is working specifically within the "genetic algorithms" framework.

2. Can potential solutions to the problem be represented using simple data structures such as bit strings or trees? (If not, it may be difficult to work with the mechanics of the GA.)
3. Can a meaningful evaluation metric be identified that will enable one to rate the quality of each potential solution to your problem? (Without such a measure, the GA is unable to determine which solutions are more promising to work with.)
4. Can operators be devised to combine parts of two "parent" solutions and produce (viable) offspring solutions? (If the offspring do not potentially retain some of what made the parents "good", the GA will not be markedly better than random trial and error.)

In the cladistics task, there is a standard approach to forming the cladograms, but that process also has a stochastic element, so the standard approach is not guaranteed to find "the best" cladogram for a given dataset. In the cladistics task, solutions to the problem are naturally represented as trees. In addition, a standard metric for evaluating a given tree is provided with the task (parsimony). However, there is a challenge for implementing the cladistics task using the GA approach: devising operators that produce offspring from two parent solutions while retaining meaningful information from the parents.

4 The GA for Cladistics

The typical GA approach to doing "crossover" with two parent solutions with a tree representation is to pick a subtree (an interior or root node) in both parents at random and then swap the subtrees to form the offspring solution. The typical mutation operator would select a point in the tree and mutate it to any one of the possible legal values (here, any one of the species). However, these approaches do not work with the cladistics trees because each species must be represented in the tree exactly once.

4.1 Crossover Operator

The needs for our crossover operator bear some similarity to traveling salesperson problems (TSP's), where each city is to be visited exactly once on a tour. There are several approaches in the literature for working on this type of problem with a GA, however, the TSP naturally calls for a string representation, not a tree. In designing our own operator, we studied TSP approaches for inspiration, but ultimately devised our own. We wanted our operator to attempt to preserve some of the species relationships from the parents. In other words, a given tree contains species in a particular relationship to each other, and we would like to retain a large degree of this structure via the crossover process.

Our crossover operator proceeds as follows:

1. Choose a species at random from one of the parent trees. Select a subtree at random that includes this node, excluding the subtree that is only the

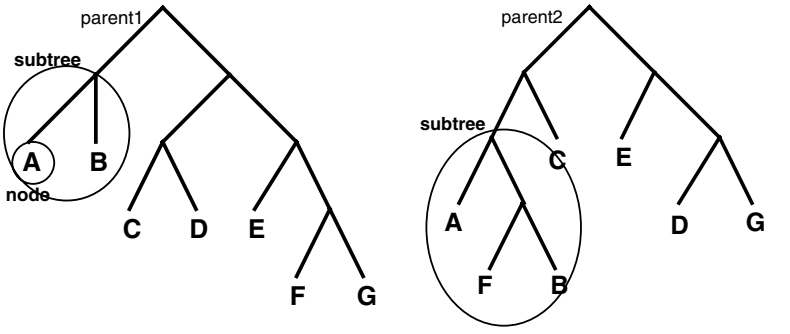

Fig. 1. Two example parent trees for a cladistics problem with seven species. A subtree for crossover has been identified for each tree.

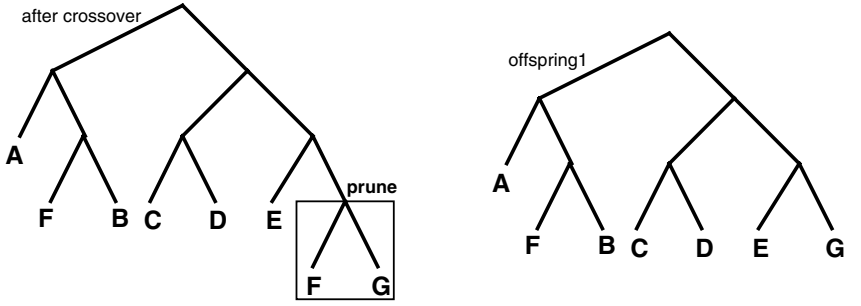

Fig. 2. At the left, the offspring initially formed by replacing the subtree from parent1 with the subtree from parent2; on the right, the offspring tree has been pruned to remove the duplicate species F.

leaf node and the subtree that is the entire tree. (The exclusions prevent meaningless crossovers, where no information is gained from the operation.)

2. In the second parent tree, find the smallest subtree containing all the species from the first parent's subtree.
3. To form an offspring tree, replace the subtree from the first parent with the subtree from the second parent. The offspring must then be pruned (from the "older" branches) to remove any duplicate species.
4. Repeat the process using the other parent as the starting point, so that this process results in two offspring trees from two parent trees.

This process results in offspring trees that retain some of the species relationships from the two parents, and combine them in new ways. An example crossover is illustrated in Figs. 1 and 2. The parents are shown in Fig. 1; Fig. 2 shows first the offspring formed via the crossover operation and identifies the subtree that must now be pruned and second shows the resulting offspring (af-

ter pruning species F). (Note that in the cladograms, swapping the left and right children does not affect the meaning of the cladogram.)

4.2 Mutation Operator

The typical GA "mutation" operator takes a location in the solution at random and mutates it to some other value. Again, the standard operator was not suited to our representation, where each species must appear exactly once in the tree. Instead, for our mutation operator, we selected two leaf nodes (species) at random, and swapped their positions in the tree.

4.3 Canonical Form

The Wagner parsimony metric uses "unrooted" trees, leading to many different possible representations of "the same" cladogram that are anchored at different points. Furthermore, flipping a tree (or subtree) left to right (switching the left and right subtrees) does not alter the parsimony of a cladogram (nor represent an alternative evolutionary hypothesis). Therefore, it soon became clear that Gaphyl would benefit from a canonical form, that could be applied to trees to ascertain whether trees in the population represented the same or distinct cladograms.

The canonical form we instituted picks the first species in the data set to be an offspring of the root, and "rotates" the tree (and flips, if necessary) to keep the species relationships in tact, but to reroot the tree at a given species. (To simplify comparisons, we followed the default Phylip assumption of making the first species in the dataset the direct offspring of the root of the tree.) Secondly, the subtrees are (recursively) rearranged so that left subtrees are smaller (fewer nodes) than right subtrees and that when left and right subtrees have the same number of nodes, a preorder traversal of the left subtree is alphabetically before a preorder traversal of the right subtree. This process is carried out when saving the "best" trees found in each generation, to ensure that no equivalent trees are saved among the best ones. Canonical form is illustrated in Fig. 3.

4.4 A Second Mutation Operator

The addition of a canonical form suggested the design of a second mutation operator. The relationships between species in a subtree is potentially useful information for offspring to inherit from parents. But perhaps the subtrees should be connected differently. The second mutation operator picks a random subtree and a random species within the subtree. The subtree is rotated to have the species as the left child of the root and reconnected to the parent.

4.5 Immigration

Early runs with Gaphyl on the larger dataset yielded trees with a parsimony of 280, but not 279 (lower parsimony is better). Reflection on the process and

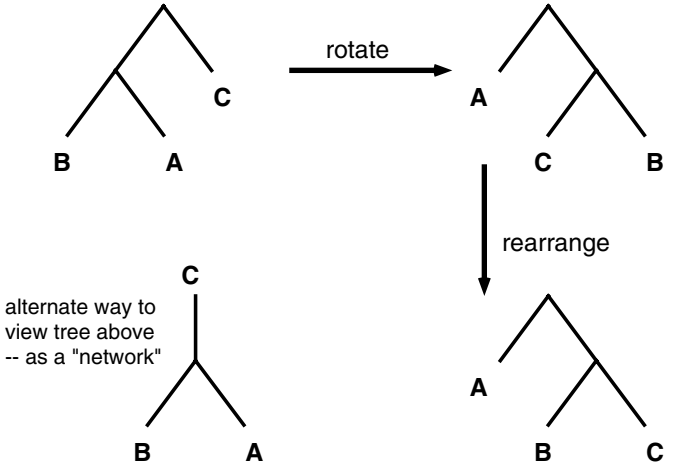

Fig. 3. An illustration of putting a tree into canonical form. The tree starts as in the top left; an alternate representation of the tree as a "network" is shown at the bottom left. First, the tree is rotated, so that the first species is an offspring of the root. Second, subtrees are rearranged so that smaller trees are on the left and alphabetically lower species are on the left.

inspection of the population determined that the process seemed to be converging too rapidly – losing the diversity across individuals that enables the crossover operator to find stronger solutions. "Premature convergence" is a known problem in the GA community, and there are a number of good approaches for combatting it. In Gaphyl, we opted to implement parallel populations with immigration. Adding immigration to the system allowed Gaphyl to find the trees of fitness 279.

The immigration approach implemented here is fairly standard. The population is subdivided into a specified number of subpopulations which, in most generations, are distinct from each other (crossovers happen only within a given subpopulation). After a number of generations have passed, each population migrates a number of its individuals into other populations; each emmigrant determines at random which population it will move to and which tree within that population it will uproot. The uprooted tree replaces the emmigrant in the emmigrant's original population. The number of populations, the number of generations to pass between migrations, and the number of individuals from each population to migrate at each migration event are, of course, all determined by parameters to the system.

5 Experimental Results

Recall that both Gaphyl and Phylip have a stochastic component, which means that evaluating each system requires doing a number of runs. In Phylip, each

distinct run first "jumbles" the species list into a different random order. In Gaphyl, there are many different effects of random number generation: the construction of the initial population, parent selection, and the selection of crossover and mutation points. For both systems, a number of different runs must be done to evaluate the approach.

5.1 Comparison of Gaphyl and Phylip

1. With the Lamiiflorae data set, the performance of Gaphyl and Phylip is comparable. Phylip is more expedient in finding a single tree with the best parsimony (72), but both Gaphyl and Phylip find 45 most parsimonious cladograms in about twenty minutes of run time.
2. With the angiosperm dataset, a similar pattern emerges: Phylip is able to find one tree with the best fitness (279) quite quickly, while Gaphyl needs more run time to first discover a tree of fitness 279. However, in a comparable amount of runtime, Gaphyl is able to find 250 different most parsimonious trees of length 279 (approximately 24 hours of runtime). Phylip runs for comparable periods of time have not found more than 75 distinct trees with a parsimony of 279.

In other words, Gaphyl is more successful than Phylip in finding more trees (more equally plausible evolutionary hypotheses) in the same time period.

The first task is considerably easier to solve, and Gaphyl does not require immigration to do so. Example parameter settings are a population size of 500, 500 generations, 50% elitism (the 250 best trees are preserved into the next generation), 100% crossover, 10% first mutation, and 100% second mutation. Empirically, it appears that 72 is the best possible parsimony for this dataset, and that there are not more than 45 different trees of length 72.

The second task, as stated above, seems to require immigration in order for Gaphyl to find the best known trees (fitness 279). Successful parameter settings are 5 populations, population size of 500 (in each subpopulation), 2000 generations, immigration of 5% (25 trees) after every 500 generations, 50% elitism (the 250 best trees are preserved into the next generation), 100% crossover, 10% first mutation, and 100% second mutation. (Immigration does not happen following the final generation.)

We have not yet done enough runs with either Phylip or Gaphyl to estimate the maximum number of trees at this fitness, nor a more concise estimate of how long Phylip would have to run to find 250 distinct trees, nor whether 279 is even the best possible parsimony for this dataset.[2]

Based on these initial experiments, the pattern that is emerging is that as the problems get more complex, Gaphyl is able to find a more complete set of trees with less work than what Phylip is able to find. The work done to date

[2] We note that we inadvertently capped the number of trees that Gaphyl is able to find in setting our elitism rate. With a population size of 500 and 50% elitism, 250 is the maximum number of distinct trees that will be saved from one generations into the next.

illustrates that Gaphyl is a promising approach for cladistics work, as Gaphyl finds a wider variety of trees on this problem than Phylip does. This further suggests that Gaphyl may be able to find solutions better than Phylip is able to find on datasets with a larger number of species and attributes, because it appears to be searching more successful regions of the search space.

5.2 Evaluation of Contribution of Operators

To evaluate the contributions of the GA operators to the search, additional runs were done with the first data set (and no immigration). Empirically, crossover and the second mutation operator had been found to be the largest contributors to successful search, so attention was focused on the contributions of these operators.

In the first set of experiments, the first mutation rate was set to be 0%. First, the crossover rate was varied from 0% to 100% at increments of 10% while the second mutation rate was held constant at 100%. Second, the second mutation rate was varied from 0% to 100% at increments of 10% while the crossover rate was held constant at 100%. 20 experiments were run at each parameter setting; 500 generations were run.

Figure 4 illustrates the effects of varying the crossover rate (solid line) and second mutation rate (dashed line) on the average number of generations taken to find at least one tree of the known best fitness (72). Experiments that did not discover a tree of fitness 72 are averaged in as taking 500 generations. For example, 0% crossover was unable to find any trees of the best fitness in all 20 experiments, and so its average is 500 generations.

This first experiment illustrates that in general, higher crossover rates are better. There is not a clear preference, however, for high rates of the second form of mutation. To look at this operator more closely, the final populations of the 20 experiments were looked at to determine how many of the best trees were found in each run.

Figure 5 illustrates the effects of varying the crossover rate (solid line) and second mutation rate (dashed line) on the average number of best trees found. Experiments that did not discover a tree of fitness 72 are averaged in as finding 0 trees. For example, 0% crossover was unable to find any trees of the best fitness in all 20 experiments, and so its average is 0 of the best trees.

As Fig. 5 illustrates, runs with a higher second mutation rate tend to find more of the best trees than runs with a lower second mutation rate.

The impact of the first mutation operator had seemed to be low based on empirical evidence. So another set of experiments was done to assess the contribution of this operator. Figure 6 illustrates two sets of experiments. In both, the crossover rate was set at 100%; in one, the second mutation rate was set at 0% and in the other, the second mutation rate was set at 100%. The figure illustrates the effect of changing the first mutation rate on the average number of generations to find at least one of the best trees.

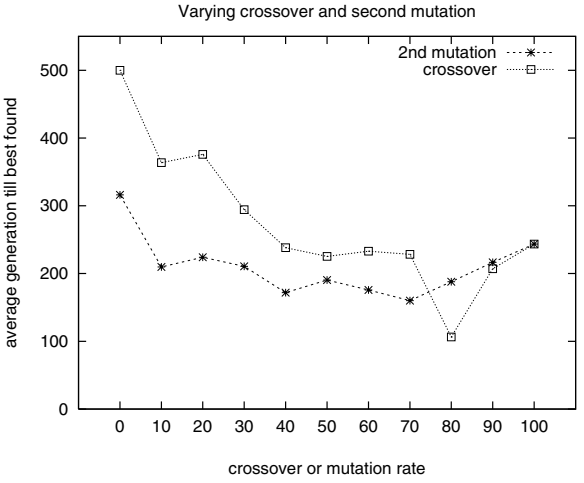

Fig. 4. The effect of varying crossover rate while holding second mutation constant and of varying the second mutation rate while holding the crossover rate constant. The average generation at which the best fitness (72) was found is illustrated.

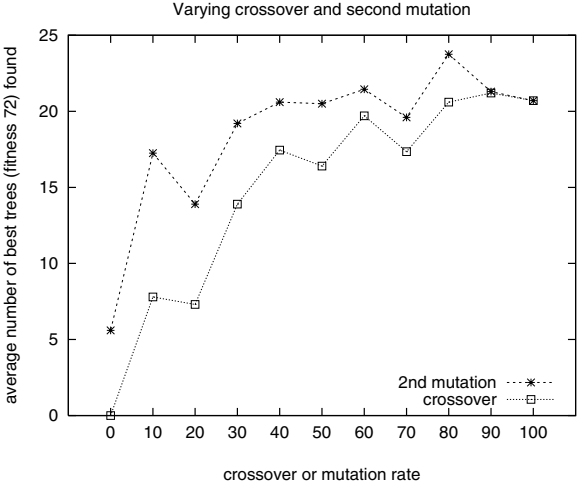

Fig. 5. The effects of varying crossover rate while holding second mutation constant and of varying the second mutation rate while holding the crossover rate constant. The average number of best trees (45 max) found by each parameter setting is illustrated

The results of this experiment clearly indicate that higher rates of this form of mutation are not beneficial. Furthermore, this operator is not clearly contributing to the search.

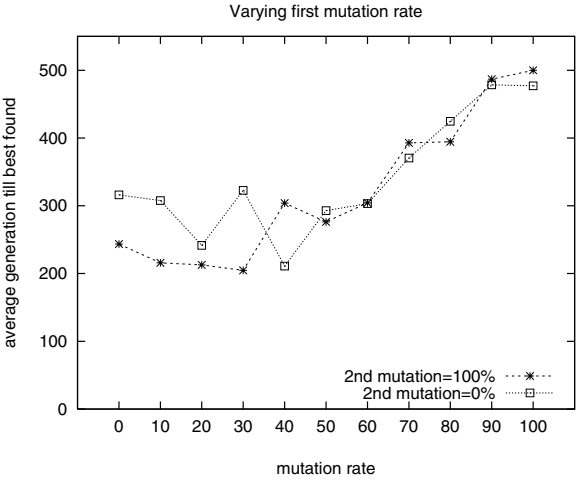

Fig. 6. The effect of varying the first mutation rate while holding crossover and second mutation constant. The crossover rate is 100% for both graphs; second mutation rates of 100% and 0% are shown. The average generation at which the best fitness (72) was found is illustrated.

6 Conclusions and Future Work

The GA search process as implemented in Gaphyl represents an improvement over Phylip's search process in its ability to find more trees than Phylip in the same runtime. One possible facet of this success is that the Gaphyl search process is independent of the number of attributes (and attribute-values); the complexity of the search varies with the number of species (which determines the number of leaf nodes in the tree). Phylip uses attribute information in its search process.

The first mutation operator is perhaps the "obvious" form of mutation to implement for this problem, and yet, its use (at high levels) appears to detract from the success of the search. This points to the importance of evaluating the contributions of operators to the search process.

There is obviously a wealth of possible extensions to the work reported here. First, more extensive evaluations of the capabilities of the two systems must be done on the angiosperms data set, including an estimate of the maximum number of trees of fitness 279 (and, indeed, whether 279 is the most parsimonious tree possible). This would entail more extensive runs with both approaches. Furthermore, as evidenced by the unexpected result with the mutation operator, the effect of the immigration operator in Gaphyl must be explored further.

Second, more work must be done with a wider range of datasets to evaluate whether Gaphyl is consistently able to find a broader variety of trees than Phylip, and perhaps able to find trees better than Phylip is able to find.

Third, Gaphyl should be extended to work with non-binary attributes and to handle data with unknown values. Since the ability to work with missing values

and a number of alternative metrics are already part of the Phylip implementation, these changes should be straightforward in the Gaphyl system. This is particularly important in that phylogenetic trees are increasingly used by biologists primarily with the A, C, G, T markers of genetic data. It should also be extended to implement and evaluate alternative evaluation metrics to Wagner parsimony.

Finally, we need to compare the work reported here to other projects that use GA approaches with different forms of cladistics, including [7] and [8]. Both of these projects use maximum likelihood for constructing and evaluating the cladograms. The maximum likelihood approach (which is known as a "distance-based method") is not directly comparable to the Wagner parsimony approach (which is known as a "maximum parsimony" approach).

Acknowledgments. I would like to thank Emily F. Greenfest, who worked with me in the initial design and implementation of this project. Thanks also to Judy L. Stone and Randall Downer for sharing their knowledge of cladistics theory and software. Thanks to Randolph M. Jones, Joshua R. Ladieu, and the anonymous reviewers for comments on previous drafts of this paper.

References

1. L. Davis. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York, NY, 1991.
2. M. J. Donaghue. Treebase: A database of phylogenetic knowledge. web-based data repository, 2000. Available at `http://phylogeny.harvard.edu/treebase`.
3. J. Felsenstein. Phylip source code and documentation, 1995. Available via the web at `http://evolution.genetics.washington.edu/phylip.html`.
4. P. L. Forey, C. J. Humphries, I. L. Kitching, R. W. Scotland, D. J. Siebert, and D. M. Williams. *Cladistics: A Practical Course in Systematics.* Number 10 in The Systematics Association. Clarendon Press, Oxford, 1993.
5. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.
6. J. J. Grefenstette. A user's guide to GENESIS. Technical report, Navy Center for Applied Research in AI, Washington, DC, 1987. Source code updated 1990; available at `http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/genetic/ga/systems/genesis/`.
7. P. O. Lewis. A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. *Mol. Biol. Evol.*, 15(3):277–283, 1998.
8. H. Matsuda. Protein phylogenetic inference using maximum likelihood with a genetic algorithm. In L. Hunter and T. E. Klein, editors, *Pacific Symposium on Biocomputing '96*, pages 512–523. World Scientific, London, 1996.
9. M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, Cambridge, MA, 1996.
10. D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M. Hillis. *Molecular Systematics*, chapter Phylogenetic Inference, pages 407–514. Sinauer Associates, Inc., Sunderland, MA, 1996.