2000-11-10

Parallel Phylogenetic Inference

Mark J. Clement
clement@cs.byu.edu

David McLaughlin

See next page for additional authors

Original Publication Citation
Parallel Phylogenetic Inference Quinn Snell, Michael Whiting, Mark Clement and David McLaughlin Published in SC2 Proceedings.

Parallel Phylogenetic Inference[1]

Quinn Snell, Michael Whiting, Mark Clement and David McLaughlin
Brigham Young University
Provo, UT 84602

Abstract

Recent advances in DNA sequencing technology have created large data sets upon which phylogenetic inference can be performed. However, current research is limited by the prohibitive time necessary to perform tree search on even a reasonably sized data set. Some parallel algorithms have been developed but the biological research community does not use them because they don't trust the results from newly developed parallel software. This paper presents a new phylogenetic algorithm that allows existing, trusted phylogenetic software packages to be executed in parallel using the DOGMA parallel processing system. The results presented here indicate that data sets that currently take as much as 11 months to search using current algorithms, can be searched in as little as 2 hours using as few as 8 processors. This reduction in the time necessary to complete a phylogenetic search allows new research questions to be explored in many of the biological sciences.

1 Introduction

Phylogenetic analysis has become an integral part of many biological research programs. These include such diverse areas as human epidemiology [CWNT+98, SHZT92], viral transmission [Cra96], biogeography [DeS95], and systematics [HMM96]. With the advent of Polymerase Chain Reaction (PCR) and automated sequencing technologies, the ability to generate data for inferring evolutionary histories (phylogenies) for a great diversity of organisms has increased dramatically in the last ten years. Researchers are now commonly generating many sequences from many individuals. However, the ability to analyze the data has not kept pace with data generation, and phylogenetics has now reached a crossroads where we cannot effectively analyze the data we generate. The chief challenge of phylogenetic systematics is to develop algorithms and search strategies to effectively analyze large data sets.

The majority of methods used for phylogenetic inference are based upon optimizing character data (e.g., DNA nucleotide sequences) on fully dichotomous, branching topologies (trees). The best tree (or set of trees) is selected by finding that topology which is most optimal based on a given criterion [SOWH96]. Currently in phylogenetics, there exist three major classes of optimality criteria: parsimony, maximum likelihood, and distance methods. Parsimony methods seek solutions that minimize the amount of evolutionary change required to explain the data. Maximum likelihood methods incorporate a specific statistical model of evolutionary change, and then calculate the probability that a given tree topology would give rise to an observed data set [HC97]. In a distance analysis, differences among sequences are first transformed into a pair wise distance matrix. The elements of the matrix may be simple counts of the differences between the sequences, but it is more typical to use a model of evolution to transform these observed counts into estimates of the total percentage of sites that have changed between the two sequences. A least-squares approach is then used to optimize branch lengths and tree topology given this distance matrix [RN92].

The tree or set of trees that is most optimal under a given criterion is selected as the best-supported hypothesis of evolutionary relationships, and is then used for analysis of the particular

[1] 0-7803-9802-5/2000/$10.00 (c) 2000 IEEE.

problem at hand. While there is considerable debate as to the best optimality criterion to use under various conditions, all methods require the evaluation and ranking of multiple trees across a landscape of all possible trees.

1.1 Computational Complexity

The crux of the computational problem is that the actual landscape of possible topologies can be extraordinarily difficult to evaluate with large data sets. This occurs for a number of reasons:

1. With an increase in the number of terminals (taxa), there is an incredible increase in the number of possible bifurcating topologies to be evaluated (see Figure 1). The numbers become so phenomenally large with even a relatively modest number of terminals, that phylogenetic inference is considered an *NP*-complete problem [Kim98, Swo96]. The increase in tree landscape with an increase in terminal numbers is considered the greatest computational challenge. Because of the immensity of the search space, it is impossible to do exhaustive searches that guarantee the most optimal solutions; hence phylogeneticists resort to heuristic algorithms that attempt to effectively search the tree landscape for optimal trees.

Terminals	*Number of Trees*
10	2×10^{6}
22	3×10^{23}
50	3×10^{74}
100	2×10^{182}
1,000	$2 \times 10^{2,860}$
10,000	$8 \times 10^{38,658}$
100,000	$1 \times 10^{486,663}$
1,000,000	$1 \times 10^{5,866,723}$
10,000,000	$5 \times 10^{68,667,340}$

Fig. **1** The number of distinct, unrooted, bifurcating trees as a function of the number of taxa (Hillis et al., 1996)

2. An increase in the number of terminals and number of characters (e.g., increased sequence lengths) requires an increased time allocation to compute cost for each topology. While this may be a relatively minor problem for distance and parsimony methods, it can become very pronounced with likelihood methods that require more complex models of evolution.

3. Large data sets typically have multiple islands of suboptimal trees, such that search algorithms can become easily trapped in local optima [HSPH88, Mad91].

4. In many cases there are multiple, equally optimal solutions, each of which are typically saved and further evaluated in an attempt to find a more optimal solution. In many cases this requires an enormous number of optimal trees (100,000+) to be saved and evaluated.

2 Phylogenetic Algorithms

There currently are three major software packages that are designed to search for optimal trees:

1. PHYLIP [Fel91] uses algorithms for evaluating topologies under all three optimality criteria.
2. PAUP* [Swo96], currently the most widely used and complete software package, also evaluates trees using distance, parsimony, and maximum likelihood methods.
3. NONA [Gol97] evaluates trees only under parsimony, but includes some unique searching strategies that provide efficient searches with larger data sets.

PAUP* is the most commonly used and trusted software. Much of the research community has come to depend on PAUP* and its results even though other software has shown some speed improvements over PAUP*. Due to the trust in this software package, it is the target sequential application used in this paper.

With parsimony search, sequential applications are usually bogged down (i.e., may require 6+ months for a complete heuristic search) with more than 500 terminals. Using likelihood search methods, it is hard to perform a search of 50 or more terminals (see section 4). All three programs currently run sequentially, and there does not exist any widely used parallel version for computing phylogeny. Some parallel approaches for both maximum likelihood and parsimony have been developed. However, researchers in systematics and biology have not used these tools because they don't trust the results. This is a common problem with new software in an established research community. Our solution to the problem is to create a parallel application based on the trusted sequential algorithms already in use by the research community.

2.1.1 The Ratchet

A new and very effective tree search strategy, called the ratchet, has recently been developed by Kevin Nixon [Nix98]. The ratchet is a search method that uses a statistical approach to sampling tree islands to find the most optimal trees for a data set. Because of the simplicity of the method, it can be readily implemented without the need to modify the tree search software.

The basic ratchet strategy is as follows:

1. *Obtain a starting tree*. This is obtained by first generating a Wagner tree, followed by some minimal level of branch swapping; various alternative ways of obtaining starting trees are available, such as randomly generating trees or using trees from previous analyses.
2. *Randomly perturb the data set*. Originally, the perturbation used was to set all characters weights to 1 (or some original weighting scheme) and randomly select a subset of characters that would be upweighted (usually by 1). However, various modifications of the perturbation method have now been tested, and it turns out that removing (weighting to 0) is probably as effective as upweighting selected characters.
3. Holding a single or a few trees, perform a standard tree search on the perturbed data.
4. Reweight characters to original weights.
5. Using the tree(s) found in step 3 as a starting point, perform SPR or TBR tree search on the UNPERTURBED data, still holding a single (or few) trees.
6. Return to 2) and repeat, using tree(s) from step 5 as a starting point.

Steps 2 through 5 constitute a single iteration. Each of these iterations can be executed in parallel as there are no data dependencies between iterations. More importantly, each of the steps of an iteration can be performed by current sequential software simply by executing the appropriate command. The remainder of this paper describes the parallel programming system used, the parallelization of the ratchet search, and the resulting performance of the algorithm.

3 DOGMA

The Distributed Object Group Metacomputing Architecture (DOGMA) [JCS98] is a parallel and distributed programming architecture. It can be used on supercomputers or clusters and it also provides a way for machines to become involved with a computation through a screen saver or through accessing an Internet web page. DOGMA manages machines that are currently available for a computation and becomes a broker that matches application requirements with available resources.

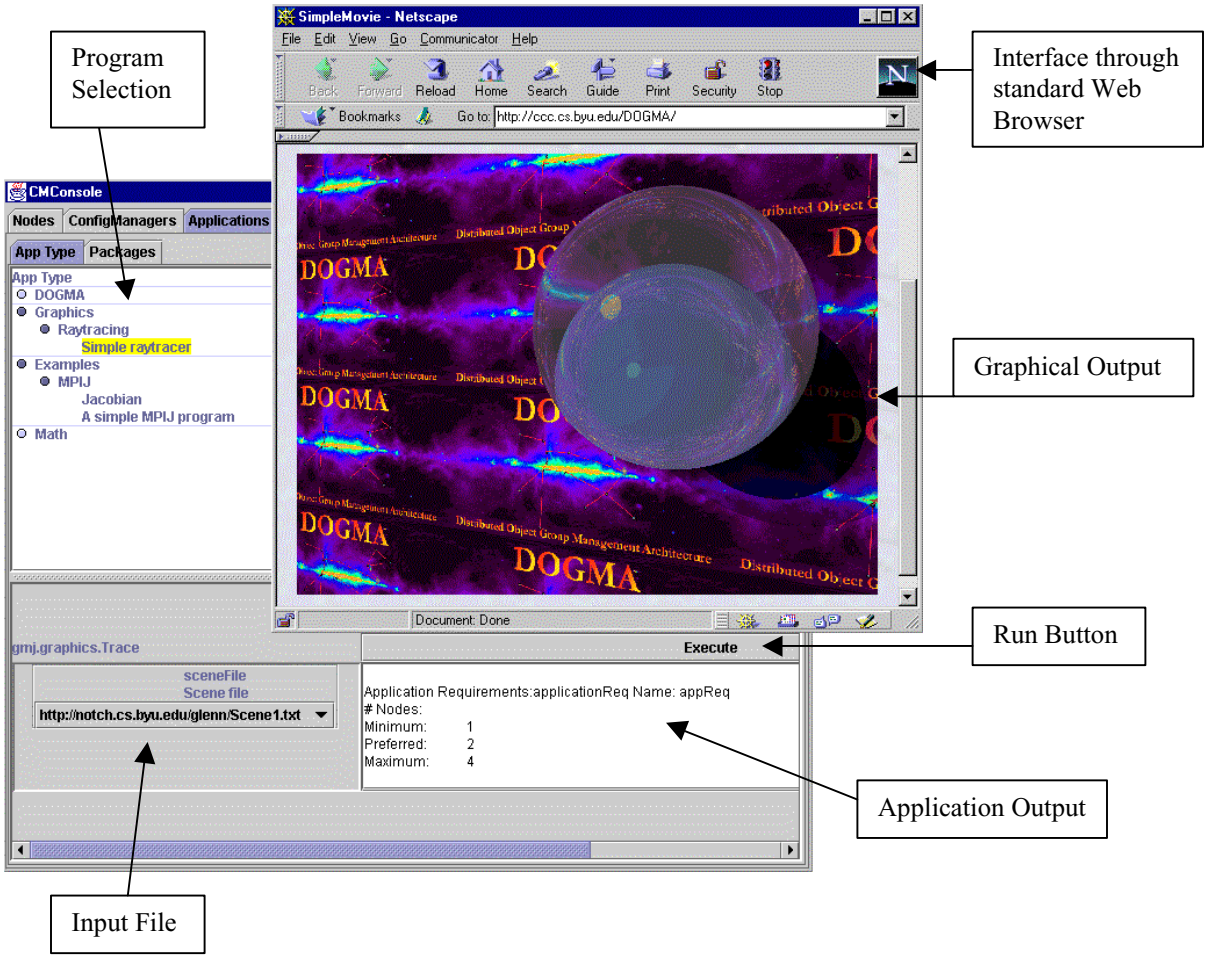

Figure 2: DOGMA application interface

The existing DOGMA interface simplifies the use of parallel applications and systems. Figure 2 shows an existing ray tracing application. The user selects the application type and the specific application from the tree diagram shown on the left. When the application is se lected, the user clicks to select the input file shown in the lower left corner. The application requirements are then shown in the lower right box. The application is run by clicking on the execute button. At this point the DOGMA system launches the program on available nodes. If additional machines become idle, their screen savers may contact the DOGMA system and become involved in the computation. The only requirement placed upon the user is some familiarity with the application and its inputs. Ap plications may also ask the user to specify input files for the DOGMA computation. The development of DOGMA applications has been covered extensively in [JCS98].

3.1.1 Volunteer supercomputing

Imagine a typical evening at any major university between 2am and 8am. The computer laboratories are all closed and as many as 2,000 computers are sitting idle. At the same time, researchers in several biological sciences are waiting for results from phylogenetic inference engines that have been running for months on a single machine. Several avenues of research are

unexplored because it takes so long to complete computations on a single machine. These applications could run several thousand times faster if they were parallelized and the computing resources were available. Since idle machines are available at some location in the world at any given time (it is always night somewhere), it makes sense to employ these machines in useful computations. Although attempts have been made to solve this problem previously, they have generally failed because they have been cumbersome to use and machine dependent. The DOGMA infrastructure described here addresses the concerns that have thwarted prior attempts in order to provide a solution for the computational biology community.

When a machine enters screen saver mode, it contacts the DOGMA broker and accepts tasks that correspond to those specified by the user in the screen saver setup. Master tasks are started from a web page that will allow the users to select the user group that they belong to and the parameters for the inference engine. The master task will then use all available machines to complete the task. If one of the machines leaves screen saver mode, the task will terminate on that node and be spawned on the next available machine. The system software and phylogenetic search algorithms are designed so that any machine can be terminated without effecting the computation. As additional machines become available, they are added to the computation.

4 The Parallel Ratchet

As explained earlier, each iteration of the ratchet may occur in parallel as there is no direct dependency between iterations. In the sequential ratchet, each iteration operates on the best available data. However, due to the random weighting of character data that takes place, each iteration has an equally likely opportunity to find the next most optimal tree. This behavior means that each iteration that occurs in parallel has an equally likely chance to return the next most optimal tree regardless of the fact that they may have each started with different length trees.

Our implementation of the parallel ratchet is master-worker based. A master process is launched in the DOGMA system that creates a set of tasks and then launches worker processes on available machines (see Figure 3). Each worker, in this case, is simply wrapper code that is used to interact with the newest release version of PAUP* or NONA. The wrapper application is directly connected to the standard input and output of the sequential application. When a parallel tree search is performed, each worker gives the search commands to the unmodified sequential application. When the command is complete, the wrapper code parses the output to determine the length of the shortest trees found and then sends this length along with the actual trees back to the master task. When a worker requests a task, the master process selects one of the best available trees and sends it to worker machines for the next iteration. A benefit of this approache is that as improvements are made to the sequential search engines (PAUP*, NONA etc), these improvements will also be seen by the parallel implementation.

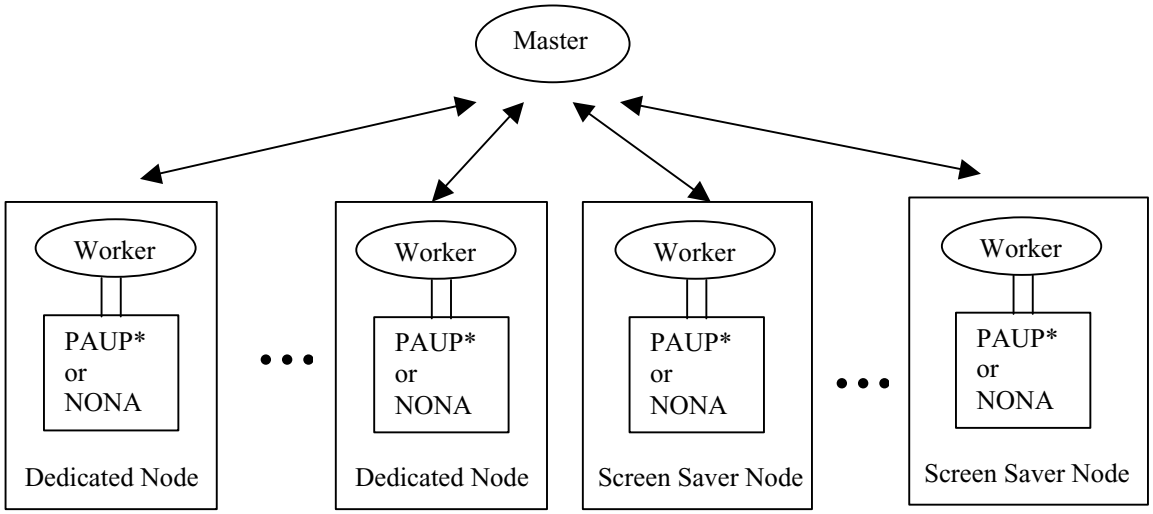

Figure 3: DOGMA implementation of the parallel parsimony ratchet

The master-worker algorithm is implemented using DOGMA Dynamic Object Group programming. This model of parallel programming allows the DOGMA system to create and launch a master program and workers on available machines. As more machines become available, the DOGMA system is free to launch more workers up to the user specified limit. If a machine suddenly becomes unavailable because it is being used, the worker dies, but the parallel program continues. Master-worker algorithms such as the parallel ratchet can benefit from this type of behavior. Although some machines accept work and then die, there are many that are able to complete the computation and add more optimal trees that will be used to further the computation. A more traditional algorithm written in MPI, for instance, must use a constant and predetermined number of processors. The number of nodes must not decrease and, more importantly in this case, is not allowed to increase during the execution of the application. DOGMA distributed object group programs can easily take advantage of having more nodes become available; thus increasing the parallelism and performance.

5 Performance

The following results have also been obtained comparing PAUP*, NONA and the parallel ratchet code. Table 1 summarizes the performance statistics for the phylogenetic tree searches. The reported times represent the time to find the first shortest tree on 266 MHz Pentium II processors. The reported lengths are measures of tree optimality with lower lengths indicating greater optimality. All data sets reporting 200 hours time were terminated before completion of the search. Entries with an asterisk indicate suboptimal solutions. For NONA and PAUP*, trees were searched using 10 random addition sequences combined with TBR branch swapping and up to 100,000 trees were retained. The cluster times were generated with the parallel parsimony ratchet algorithm on 8 processors.

Table 1: Performance results for phylogenetic search strategies

#Taxa	SOURCE	PAUP* Time (seconds)	PAUP* Length	Parallel Ratchet Time (seconds)	Parallel Ratchet Length
200	Simulated	180	3015	7	3015
1500	Simulated	200 hrs	10294*	500	10290
100	HIV	59	581	2.7	581
200	HIV	200 hrs	766*	21	765
500	HIV	200 hrs	2040*	1274	2028
100	18S	180	2054	18	2054
200	18S	200 hrs	4539*	200	4537
500	Zilla	200 hrs	16222*	300	16218

Under all scenarios the parallel ratchet algorithm found the shortest tree in considerably less time. There is a greater differential in performance (in terms of speed and the ability to find shorter trees) with large data sets. Notice that for the 500 taxa HIV data set, the parallel ratchet algorithm found a tree 12 steps shorter than the tree found by PAUP*. For this example, PAUP* took an unreasonable amount of time to find trees that were less optimal than those found by the parallel ratchet in a fraction of an hour. As is shown by the table, this is a common occurrence. The parallel ratchet made it possible to find trees that were unable to be found by the sequential algorithms. Based on our initial tests, the parallel ratchet always finds a tree at least as optimal as those reported by NONA and PAUP*. Most often it finds a tree that is more optimal and finds that more optimal tree in less time. Other research at BYU shows that the parallel ratchet more effectively covers the search space than traditional methods and appears to work much like a simulated annealing search.

5.1 Parallel Performance and Scalability

The ratchet is based on a quasi-random search heuristic. Due to this, the performance of the algorithm exhibits many search anomalies; it can be as slow as a single processor or exhibit superlinear speedup. Thus it is impossible to characterize the performance of the algorithm with data where each data point is a single observed execution time. For the analysis of the parallel ratchet, we have chosen to use average execution time. More performance data is being gathered that improves the accuracy of the expected execution times and our understanding of the performance of the parallel ratchet.

Scalability experiments were run using three different sizes of data sets: an HIV virus data set consisting of 200 individuals, the well-known Zilla data set of 500 individuals, and another HIV data set containing 1600 individuals. The HIV data sets consist of empirical data from the envelope and protease genes of the HIV virus. The Zilla data set is a well-known data set that has long been used as a benchmark for search algorithms.

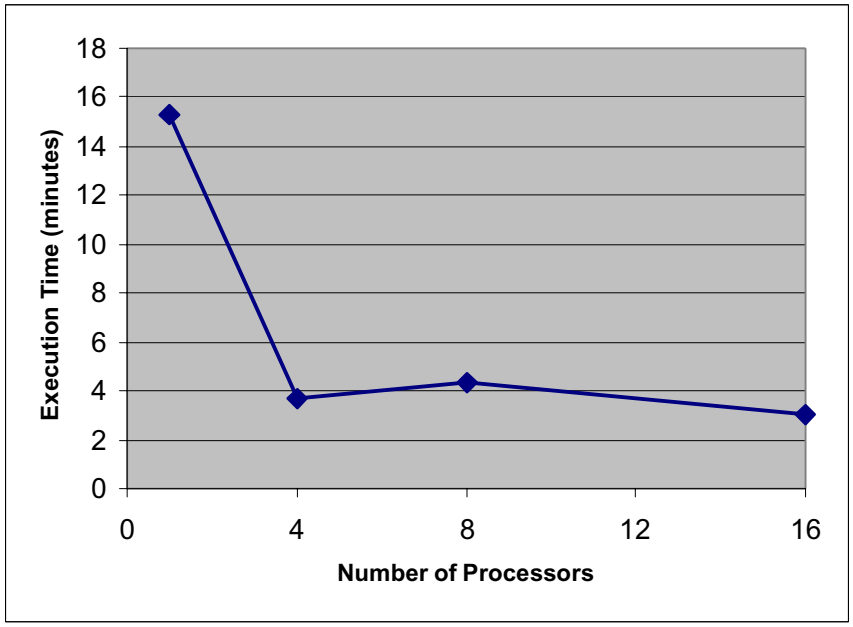

Figure 4: Performance for Parallel Ratchet on the small HIV data set

Interestingly, with the small data set of 200 individuals the number of trees in the search space is greater than $2x10^{182}$. Yet parallelism beyond 4 processors appears to be ineffective; even though it is impossible to search even that size of a search space (see Figure 4).

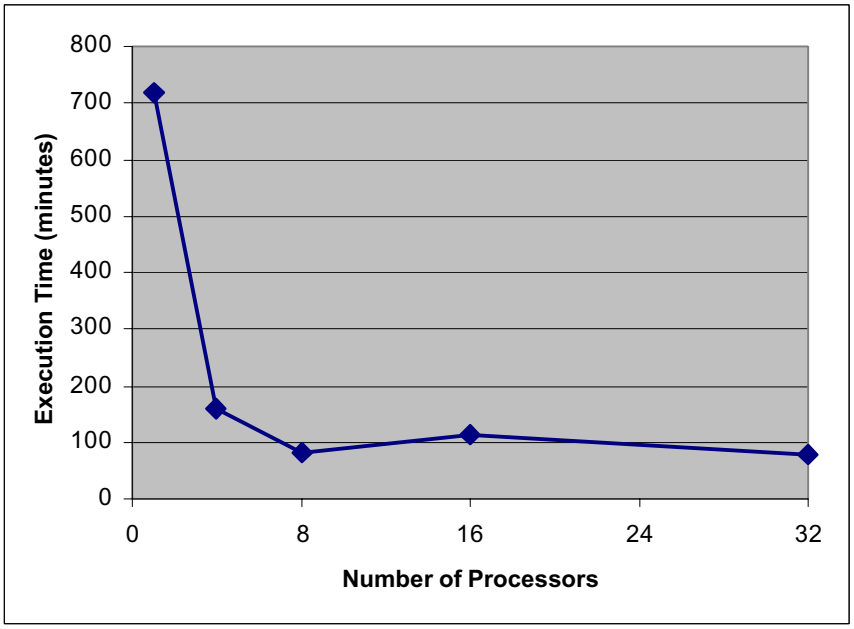

Figure 5: Performance for Parallel Ratchet on the Zilla data set

The Zilla data set shows a similar trend after 8 processors as is shown in Figure 5. The results from the large HIV data set indicate that 32 processors found a tree of cost 4326 in approximately 56 hours and the same size tree was found in 125 hours using 16 processors. Smaller numbers of processors were unable to find a tree of similar size.

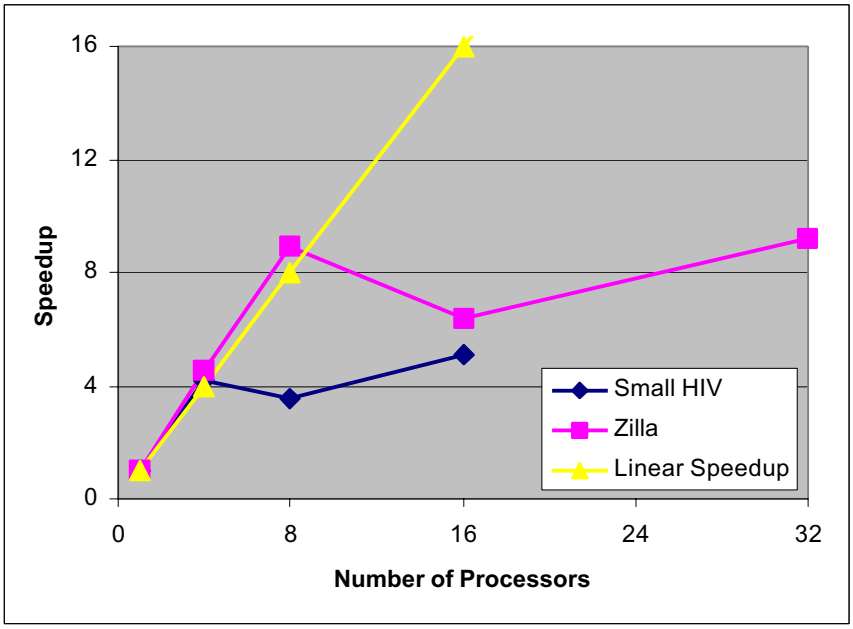

Figure 6: Parallel Ratchet Speedup

The speedup graph in Figure 6 clearly shows the relationship of the number of processors to data set size. The research community is creating larger data sets and it remains to be seen if larger data sets will continue the trend shown here. We speculate that for larger data sets, the knee of the performance curve will continue to move to the right. This indicates that as the data set grows larger, a larger number of processors can be effectively used to analyze the data. Although the size of the search space for even small data sets is exponentially large, the amount of parallel processing that is effective for the parallel ratchet seems to be related more to the data set size. The subject of current research is to determine the relationship between data set size and the knee of the performance curve.

6 Conclusion

This software infrastructure and the parallel ratchet allows researchers to explore new areas that were previously impractical because of the long compute times. The parallel ratchet is based on well-known and trusted software, which will make it easier for the research community to adopt. Once other researchers see the success and ease of use offered by parallel processing, we feel that there will be widespread use of the system throughout the Biology community. We have observed that improvements in search algorithms are currently being stifled by the lack of a high performance, public domain software base for researchers to modify when trying new algorithms.

Five years ago the Internet was a realm for a few technical people who could deal with remembering Internet addresses and obscure command line driven programs. Internet Web Browsers changed the face of the Internet to allow many more people access to a wealth of information. This research has the potential of making similar changes in the biological sciences. The web site at http://ccc.cs.byu.edu/phylo/demo.html provides a working version of the preliminary implementation. Anyone with a web browser will be able to make use of these tools.

7 Bibliography

[Cra96] Crandall, K. A. 1996. Multiple interspecies transmissions of human and simian T -cell leukemia/lymphoma virus type I sequences. Molecular Biology and Evolution 13: 115-131.

[CWNT+98] Clark, A. G., Weiss, K. M., Nickerson, D. A., Taylor, S. L., Buchanan, A., Stengard, J., Salomaa, V., Vartiainen, E., Perola, M., Boerwinkle, E. and Sing, C. F. 1998. Haplotype structure and population genetic inferences from nucleotide-sequence variation in human lipoprotein lipase. American Journal of Human Genetics 63: 595-612.

[DeS95] DeSalle, R. 1995. Molecular approaches to biogeographic analysis of Hawaiian Drosophilidae. Hawaiian Biogeography (ed. by W. L. Wagner and V. A. Funk), pp. 72--89. Smithsonian Institution Press.

[Fel91] Felsenstein, J. 1991. PHYLIP: Phylogenetic Inference Package. 3.4. University of Washington, Seattle, WA

[Gol97] Goloboff, P.A. 1997. NONA, version 1.5 (32 bit) (and more recent versions). Available via FTP with registration from Willi Hennig Society.

[HMM96] D. M. Hillis, C. Moritz, and B. K. Mable. 1996. Molecular Systematics. Sinauer Assc. Sunderland.

[HSPH88] Hendy, M. D., Steel, M. A., Penny, D. and Henderson, I. M. 1988. Families of trees and consensus. In: Classification and Related Methods of Data Analysis (Bock, H. H., eds). pp. 355-362. Elsevier, Amsterdam.

[JCS98] Judd, G., Clement, M. and Snell, Q. 1998. The DOGMA Approach to High -Utilization Supercomputing', Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing (HPDC-7), Chicago, Illinois, July 1998.

[JCS98] Judd, G., Clement, M. and Snell, Q. 1998. DOGMA: Distributed Object Group Metacomputing Architecture, Concurrency: Practice and Experience, Vol 10(1),1-7 (1998).

[Kim98] Kim, J. 1998. Large-scale phylogenies and measuring the performance of phylogenetic estimators. Syst. Biol. 47:43-60.

[Mad91] Maddison, D. R. 1991. The discovery and importance of multiple islands of most-parsimonious trees. Systematic Zoology 40: 315-328.

[Nix98] Nixon, K., 1998. The parsimony ratchet, a new method for rapid parsimony analysis and broad sampling of tree islands in large data sets. Program of the 17th meeting of the Willi Hennig Society, Sao Paulo, Brazil, Abstracts, p. 59.

[RN92] Rzhetsky, A. and Nei, M. 1992. A simple method for estimating and testing minimum -evolution trees. Molecular Biology and Evolution 9: 945-967

[SHZT92] Sing, C. F., M. B. Haviland, K. E. Zerba and A. R. Templeton 1992. Application of cladistics to the analysis of genotype-phenotype relationships. Eur. J. Epidemiol. 8: 3-9.

[Swo96] Swofford, D.L. 1996. PAUP: Phylogenetic analysis using parsimony (and other methods), vers. 4.0. Sinauer Associates, Sunderland MA.