Computational Options for Bioinformatics Research in Evolutionary Biology

Michael A. Thomas, Mitch D. Day, and Luobin Yang

Department of Biological Sciences,Idaho State University,
Pocatello, ID 83209-8007 USA
{mthomas, daymitc, yangluob}@isu.edu
http://egg.isu.edu

Abstract. This review will introduce areas of evolutionary research that require substantial computing resources using the examples of phylogenetic reconstruction and homology searching. We will discuss the commonly used analytical approaches and computational tools. We will discuss two computing environments employed by academic evolutionary researchers. We present a simple empirical demonstration of scalable cluster computing using the Apple Xserve solution for phylogenetic reconstruction and homology searching. We conclude with comments about tool development for evolutionary biology and Open Source strategies to promote scientific inquiry.

1 Introduction

An evolutionary perspective is implicit in bioinformatics approaches involving nucleotide and protein sequence analysis. For example, Dayhoff's [1] PAM matrix is based upon the assumption of different rates of amino acid substitution over evolutionary time. It is used to score sequence alignments and as a critical component of BLAST [2]. This evolutionary perspective is most developed in analyses of how families of related protein or nucleic acid sequences have diverged during evolutionary history.

The evolutionary perspective has played an indispensible role in the analysis of the human genome [3, 4] and in other genome projects (e.g., wheat, bacteria and Drosophila). Researchers identify homologs of interesting genes, infer probable gene functions by identifying conserved functional elements within genes, and determine the intensity of natural selection on a given genetic element. These approaches have also been used to explore human evolution by comparing human genes with their homologs in our relatives sharing recent ancestors (such as chimp) [5]; to investigate human disease-related genes [6]; and to identify new human disease-gene candidates through comparisons with model organisms [7].

The computational needs of molecular biologists are increasing more rapidly than our collective ability to manage, analyze, and interpret data. A number of current projects manage vocabularies describing genes and functions [8] and manage data from the numerous genome projects with generic genome database

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3515, pp. 68–75, 2005.

construction sets [9]. The flood of freely available data means that current research requires analysis of dozens of genes and thousands of nucleotides. The added complexity of managing and analyzing this amount of data requires the skills of a bioinformatics collaborator and significant computing power. There are several distinct classes of problems that face the bioinformatics researcher. We present the examples of phylogenetic reconstruction and homology searching.

Reconstructing Evolutionary History. Phylogenetic reconstructions of the evolution of genes or species utilizes historical information contained in protein and nucleotide sequences. Phylogenetic reconstruction has made early and extensive use of bioinformatics approaches and large data sets.

There are several steps involved with phylogenetic reconstruction. First, we select an optimality criterion for selecting the proper phylogeny. Second, we examine likely trees to find those that satisfy the optimality criterion. Heuristic search algorithms are often used since an exhaustive search of all possible topologies is impossible or problematic. Third, statistical metrics determine the probability that the tree(s) found are representative of the true evolutionary relationship. See section 2 for commonly used software tools; comparisons of these tools using different cluster configurations are explored in section 3.1.

Homology Searches. Homologs are genes that share a common evolutionary ancestor. All evolutionary analyses are dependent upon the accurate and meaningful assessment of homology [10]. Large-scale genome comparisons rely on homolog predictions extracted from databases such as NCBI's HomoloGene and TIGR's EGO [5]. Predicted homologs in these databases are based on *reinforced reciprocal best match* criteria, wherein a gene from each of two organisms is shown to be the best match for the other in a genome BLAST (Basic Local Alignment Search Tool)[2].

2 Applications of Bioinformatics Tools for Phylogenetic Reconstruction

Many methods are currently used to reconstruct and evaluate phylogenetic trees [1, 11]. The nucleotides (or amino acids) are resampled with replacement to create a number of new data sets (bootstrap pseudo-replicates). An experiment may use hundreds to tens of thousands of bootstrap pseudo-replicates. For each bootstrap replicate, the tree(s) best meeting the optimality criterion are saved. For each node on the recovered tree using the original data set, the researcher records the frequency that bootstrap replicates recover the same node. Strong nodes are those that are supported by many sites in the original data set. Therefore, the bootstrap approach tests the hypothesis that the data are consistent with the tree recovered.

2.1 Optimality Criteria

The level of computational difficulty depends greatly on the type of phylogenetic analysis chosen and the criteria used to select the best tree.

Maximum Parsimony. Maximum parsimony (MP) approaches are rooted in the simple philosophical assumption that the best tree is the simplest tree that can be explained by the data. This approach ignores those nucleotide or amino acid sites that are uninformative. The researcher examines each possible (or probable) tree arrangement and calculates a tree length, which is equal to the sum of the number of changes at each site for that tree. The most parsimonious tree has the lowest tree length, but, frequently, there is more than one most parsimonious tree.

Distance Methods. Distance methods are evolutionary models intended to represent the process of evolution as it occurred on the given sequences. This method relies on a distance matrix composed of pair-wise distances for every pair of sequences in the data set using any of a number of models of nucleotide evolution. We then infer a tree from the distance matrix. Computationally, this approach is extremely efficient for most datasets.

Maximum Likelihood. Maximum Likelihood (ML) estimates are computationally intensive. Likelihood approaches calculate the likelihood that a given model is fit by a given data set. In the maximum likelihood approach to tree building, we calculate the likelihood of the data given a specific model and tree. The ML approach searches all possible tree topologies and finds the tree(s) with the highest likelihood score, a taxing task even on powerful machines.

Bayesian Analysis. A Bayesian analysis calculates the probability of the model and tree given the sequence data. We calculate this using Bayes' theorem, but the implementation is computationally intense. The Metropolis-Hastings algorithm [12, 13] can surmount this problem by using a Markov chain Monte Carlo (MCMC) search through tree space. This variation of their famous algorithm prevents the search from becoming trapped at local optima. Fortunately, this particular approach lends itself very well to parallel computing. This strategy is encoded in the *MrBayes* package (available from `http://morphbank.ebc.uu.se/mrbayes`).

2.2 Bioinformatics Tools for Phylogenetic Reconstruction

PAUP: Phylogenetic Analysis Using Parsimony (* and other methods).* PAUP is available from Sinauer (`http://paup.csit.fsu.edu/`). As the title implies, PAUP* was originally designed to implement parsimony-based approaches but has since evolved into a more comprehensive package. It is arguably the most-used and most-published bioinformatics tool for phylogenetics reconstruction. The Unix-based (command-line) portable version that can run on any Unix system, including MaxOS X. The PAUP* authors are preparing a Message Parsing Interface (MPI) designed specifically for clusters (Swofford, pers. comm.).

Phylip: Phylogeny Inference Package. Phylip is available free from (`http://evolution.genetics.washington.edu/phylip`). Phylip is a collection of dozens of open source bioinformatics applications with a unified structure and common file formats. Many developers use this open standard for structure and file formats for their own programs, building a community of users and increasing the utility of the package as a whole. Phylip has been ported to a number of platforms, but is most at home as a command-line package on a Unix machine or cluster. This characteristic makes Phylip very amenable to implementations on bioinformatics computer clusters, adding power and convenience to analyses. Phylip applications have also been integrated into BioPerl modules allowing developers to combine and extend existing tools for novel applications.

One useful tool in the Phylip package is MPI-FastDNAml, an implementation of the maximum likelihood for phylogenetic reconstruction. This program allows the user to specify a model of nucleotide evolution and searches for a phylogenetic tree that maximizes likelihood under that model. The MPI version of this program takes advantage of multiple processors to search different segments of tree space. See section 3.1 for simulation studies using this package.

Searching for Homologs. The Basic Local Alignment Search Tool (BLAST), originally created by the National Center for Biotechnology Information (NCBI), quickly approximates alignments that optimize a measure of local similarity, allowing the user to use a nucleotide or amino acid sequence to rapidly search for similar sequences in *very* large databases [2]. BLAST calculates scores of the statistical significance of alignments, providing the user a mechanism to find homologous sequences or regions. BLAST can be implemented for straightforward DNA and protein sequence database searches, motif searches, gene identification searches, and in the analysis of multiple regions of similarity in long DNA sequences. BLAST has also been extended for specialized applications.

We compare the performance of standard BLAST this program compared to three extensions and modifications (see section 3.1 for results).

3 Using Clusters for Bioinformatics

Bioinformatic problems usually have a high algorithmic complexity. ClustalW, the most commonly used multiple sequence alignment algorithm has a complexity of (n^2) where n is the number of sequences in the alignment. Consider also the number of times a project needs to be run before all the kinks and bugs are worked out. These factors combine to make the use of parallel processing power a necessity in bioinformatics research.

Beowulf for Large or Specialized Projects. A common high-performance computing solution for academic settings is a cluster of commodity-off-the-shelf (COTS) PC's yoked together in various configurations. The Beowulf approach is most commonly in academic or governmental research labs settings, but is not always ideal or cost-effective. The Beowulf approach has a low initial cost for hardware

and software, but may require a large amount of technician time for maintenance and expansion. If the effort of maintaining a Beowulf cluster itself serves an educational or research goal this effort is justifiable. However, if the primary desired product is output from high-demand computing problems the equation changes. Effort spent on care and feeding of a COTS cluster is time spent away from biological research.

Apple Xserve Clusters with iNquiry. Apple Computer sells an Apple Workgroup Cluster for Bioinformatics composed of rack-mounted Xserve G5 units with a standard, extendible platform for bioinformatics applications (iNquiry, by The Bioteam: (`http://bioteam.net`)). This system costs about $40k (education pricing) for 6 nodes including all software and hardware, rack, admin tools, etc, and requiring much less system administration experience than comparable Linux solutions. The initial hardware cost is higher, but long term costs for maintenance are lower. At our institution, this solution has proven to have the best combination of flexibility, an open, extensible platform and lower maintenance costs for our program.

3.1 Simulation Studies

The data set for these simulation studies contains 56 bacterial taxa each with 820 nucleotides from a protein-coding gene (available on request). The first study compares the performance of two programs conducting a search for the Maximum Likelihood (ML) tree, PAUP and MPI-FastDNAml. The second simulation compares four homology search programs based on the BLAST algorithm [2], NCBI-BLAST, A/G-BLAST, BT-BLAST, and MPI-BLAST. Both demonstrations were performed on a nine-node Apple Xserve computer cluster with nine nodes running the Mac OS X server platform. Five Xserve nodes were G4-based (dual 1.33Ghz with 2Gb RAM) and four were G5-based (dual 2Ghz with 4Gb RAM).

Maximum Likelihood Tree Search. MPI-fastDNAml and PAUP* searched for the Maximum Likelihood (ML) tree for our data set. Recall, the ML tree is that topology and set of branch lengths that maximizes the likelihood given the dataset and a specific model of evolution. Both programs used the simplest nucleotide substitution model that corrects for multiple substitutions (Jukes-Cantor [14]) and an heuristic search to find the ML tree.

It took PAUP* just under 40 minutes to find the tree while it took MPI-FastDNAml about 7 minutes to find the tree with 1 processor and about 3 minutes with 6 processors. FastDNAml employs an optimized heuristic search and likelihood calculation, but is less flexible in using specific models of evolution and can not take advantage of other PAUP features, such as integrating other phylogentic methods. An anticipated new version of PAUP will include refined search and likelihood algorithms and will take advantage of an MPI environment.

A/G-BLAST vs. NCBI-BLAST. We used A/G-BLAST and NCBI-BLAST to query the data set against two different large databases, NT & EST. The NT

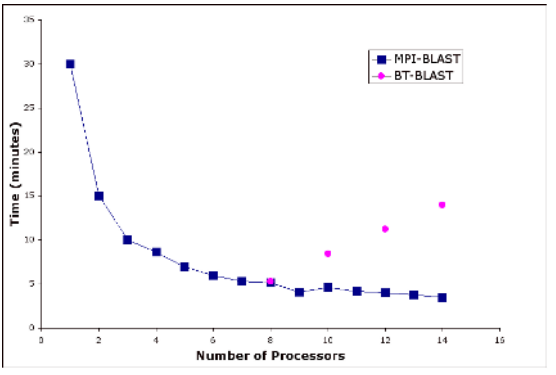

Fig. 1. MPI-BLAST vs. BT-BLAST

database includes all entries from NCBI's GenBank nucleotide database (totaling 10Gb & 2,718,617 sequences) that is nominally (but not completely) non-redundant. NCBI's EST database (8Gb & 12,582,220 sequences) includes expressed sequence tags from all species except for human and mouse, and is populated with sequences of much shorter length than the NT database. The result shows that A/G-BLAST improves the speed over NCBI-BLAST, with a much greater improvement for searches involving the small word (search target) sizes of the EST database.

Database	AG-BLAST	NCBI-BLAST
NT	26m18s	27m30s
EST	26m40s	33m37s

MPI-BLAST vs. BT-BLAST. We used our bacterial dataset to search against the NCBI NT database with different number of processors in the cluster. Both BLAST versions work in a similar fashion: the database is segmented and each processor searches one (or more) of the segments.

For MPI-BLAST, the database was divided into 14 segments. We may use any number of processors, but we limited this number to 14, equal to the number of segments in our database. MPI-BLAST does not require this limitation; we enforced this setup for comparison purposes. The number of segments and processors can be specified at run-time. Each processor will search a different segment, but some may search more segments than other processors, depending on run conditions and processors speeds. Segments are located locally - on the hard drive of the machine conducting the search of that specific segment.

For BT-BLAST, the database segmentation is more complicated (limited by size) and the number of processors used in the search *must* equal the number of segments. We divided the database into 8, 10, 12, and 14 segments. All segments are located in a single hard drive on the head node. Processors access these data via gigabit Ethernet, conduct the search, and send results to the head node.

MPI-BLAST scales very well with the number of nodes in the cluster, but BT-BLAST perforomace degrades when the nodes number increases beyond 8. There are two reasons for this. First, the first 8 processors in our cluster are G5 processors and the remainder are G4. Mixing G4 processors with G5 processors may degrade the overall performance of the cluster although this was not observed with the MPI-BLAST search. Second, BT-BLAST uses a shared directory to store database segments of. BLAST search is I/O extensive so the shared directory creates a bottleneck. This problem is specifically avoided by MPI-BLAST's distribution of the database.

4 Developing Bioinformatics Tools for Evolutionary Biology

For researchers needing to develop custom programsscripts, new software tools or even full applications, the general hardware requirements for a large cluster are essentially the same as for application-based analyses. Our language of choice for scripting and developing full-fledged bioinformatics tools is Perl. The perl interpreter will function on nearly every modern platform and properly written Perl code will run equally well on all of them. This choice is based upon the widespread use of Perl in bioinformatics research circles and specialized and general modules available through the Comprehensive Perl Archive Network (CPAN at `http://www.cpan.org`). The BioPerl modules (`http://www.bioperl.org`) available on CPAN are especially useful.

Some bioinformatics problems scale up well for parallel processing on a cluster, indeed some problems demand it. A current research effort at ISU underscores this point. We are exploring methods derived from Sequencing-by-Hybridization for their applicability to metagenomic (total environmental nucleic acids) analysis [15, 16]. This effort requires concurrent pair-wise comparisons of dozens of whole microbial genomes and oligonucleotide probe sets. The full-scale application of this method will absolutely demand the power of a cluster.

None of these advances in computing power and software development tools would be possible without the worldwide community of enthusiastic and generous contributors. Often described as the Open Source community, (`http://www.opensource.org` it represents a useful combination of generosity, pragmatism and vision that allows researchers to benefit from the collective efforts of people they may never meet. The Open Source model of software development and licensing works especially well for scientific computing in an academic research setting. The Open Source model is simply an expression of the basic ideals and values of scientific inquiry: transparency, openness and the promotion of ideas based on merit alone.

Evolutionary biology research has reached a point were access to high-powered, high-availability computing is an indispensable requirement for research. We drown in data and thirst for algorithms, tools and applications to make sense of it all. For biologists who analyze these limitless data sets, the challenge is to remain focused on the essential research tasks at hand and not be distracted

or bankrupted by the powerful computing tools at our disposal. Researchers in smaller research institutions can have access to world-class computing resources. Careful attention to the total equation of ownership and maintenance costs plus an understanding of the resources provided by the bioinformatics community and the larger world of scientific computing make it possible.

References

1. Dayhoff, M.: Survey of new data and computer methods of analysis. In Foundation, N.B.R., ed.: Atlas of Protein Sequence and Structure. Volume 5. Georgetown University, Washington, D.C. (1978)
2. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic local alignment search tool. J Mol Biol **215** (1990) 403–10
3. International Human Genome Sequencing Consortium: Finishing the euchromatic sequence of the human genome. Nature **431** (2004) 931–45
4. Imanishi, T., Itoh, T., Suzuki, Y., et al.: Integrative annotation of 21,037 human genes validated by full-length cdna clones. PLoS Biol **2** (2004) E162
5. Thomas, M., Weston, B., Joseph, M., et al.: Evolutionary dynamics of oncogenes and tumor suppressor genes: higher intensities of purifying selection than other genes. Mol Biol Evol **20** (2003) 964–8
6. Jimenez-Sanchez, G., Childs, B., Valle, D.: Human disease genes. Nature **409** (2001) 853–5
7. Feany, M., Bender, W.: A drosophila model of parkinson's disease. Nature **404** (2000) 394–8
8. GO Consortium: Creating the gene ontology resource: design and implementation. Genome Res **11** (2001) 1425–33
9. Stein, L., Mungall, C., Shu, S.Q., et al.: The generic genome browser: A building block for a model organism system database. Genome Res. **12** (2002) 1599–1610
10. Nei, M., Kumar, S.: Molecular Evolution and Phylogenetics. Oxford, New York (2000)
11. Felsenstein, J.: Inferring Phylogenies. Sinauer, New York (2004)
12. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A., Teller, E.: Equation of state calculations by fast computing machines. Journal of Chemical Physics **21** (1953) 1087–1092
13. Hastings, W.: Monte carlo s ampling methods using markov chains and their applications. Biometrika **57** (1970) 97–109
14. Jukes, T., Cantor, C.: Evolution of protein molecules. In Munro, H., ed.: Mamalian Protein Metabolism. Academic Press, New York (1969) 21–132
15. Endo, T.: Probabilistic nucleotide assembling method for sequencing by hybridization. Bioinformatics **20** (2004) 2181–8
16. Venter, J., Remmington, K., Heidelberg, J., et al.: Environmental genome shotgun sequencing of the sargasso sea. Science **304** (2004) 66–74