A high-throughput bioinformatics distributed computing platform

Thomas M. Keane[1], Andrew J. Page[2], James O. McInerney[1], and Thomas J. Naughton[2]

[1]*Bioinformatics and Pharmacogenomics Laboratory, National University of Ireland, Maynooth, Co. Kildare, Ireland*

[2]*Department of Computer Science, National University of Ireland, Maynooth, Co. Kildare, Ireland*

Homepage: http://www.cs.nuim.ie/distributed

Abstract

In the past number of years the demand for high performance computing has greatly increased in the area of bioinformatics. The huge increase in size of many genomic databases has meant that many common tasks in bioinformatics are not possible to complete in a reasonable amount of time on a single processor. Recently distributed computing has emerged as an inexpensive alternative to dedicated parallel computing. We have developed a general-purpose distributed computing platform that is capable of using semi-idle computing resources to simulate a dedicated computing cluster. We have identified the suitability of a number of bioinformatics tasks to distributed computing. We briefly outline and evaluate two distributed bioinformatics programs, DSEARCH and DPRml, which have been developed for our system.

1. Introduction

In the past number of years, the demand for high performance computing has increased dramatically in the area of bioinformatics. This is mainly due to the rapid increase in the size of genomic databases [1]. Many of the common tasks in bioinformatics are very computationally intensive and can take days, months or even years to complete on a single processor. For instance the two most rigorous database search algorithms are the Needleman-Wunsch [2] and Smith-Waterman [3] algorithms. However for large databases it is not feasible to perform full searches using these algorithms in a reasonable amount of time. Therefore a number of authors have developed heuristic search algorithms in an effort to reduce the search time. However these algorithms reduce the sensitivity of a search and can fail to detect certain matches. When given the choice, most biologists would prefer to use the more rigorous algorithms for their searches. In evolutionary biology, the decision problem associated with searching for the best phylogenetic tree is NP-complete [4]. Therefore it is not feasible to perform an exhaustive search of the tree space for any more than a few taxa. Several authors have attempted to address this problem by proposing heuristic algorithms to reduce the search space (see [5] for a review). These programs have made the process of producing large phylogenetic trees possible using only a single processor. However these programs are often based on greedy heuristic algorithms that often take the best immediate, or local, solution often resulting in a tree that is far from optimal.

In an effort to meet this overwhelming demand for computing power, several vendors have offered specialised and expensive parallel hardware for performing common tasks such as performing complete alignments of genomes [6]. One idea that has become popular in recent years is the concept of taking a number of processors and connecting them together using a high speed network to form a dedicated processing cluster (e.g. [7,8]). These systems have been extremely effective in bringing

supercomputing capabilities to ordinary bioinformatics researchers. However these systems still require a dedicated pool of processors that are physically close together. Furthermore, these systems often require a full-time system administrator to maintain, upgrade, and update the system in the long term. To tackle this apparent failure, the area of distributed computing emerged as a viable alternative to dedicated parallel computing. By harnessing the spare clock cycles of idle machines, it is possible to emulate the computing power offered by a specialised parallel machine at a fraction of the cost. Several successful systems have been developed on this basis, e.g. Seti@Home [9], Folding@Home [10], Genome@Home [10]. However many of these systems are only designed with one application in mind. That is, there are few distributed systems out there that can be programmed by a user to perform arbitrary distributed computations.

We present a general-purpose programmable distributed computing platform suitable for deployment in a typical university environment where many semi-idle desktop PC's are connected via a network. The system is fully cross-platform compatible as it is written entirely in Java. We also describe two distributed bioinformatics applications that have been recently developed to run on our distributed computing system. DSEARCH [11] is a distributed and fully cross-platform database search program that allows the user to utilise the idle clock cycles of machines to perform large searches using the most sensitive algorithms. DPRml [12] is a distributed and fully cross-platform phylogenetic tree building program.

2. Java Distributed Computing Platform

The overall design of the system is based on the client-server model [13]. This model describes a system consisting of a single server computer and a number of client computers. The server controls a resource (such as a database, algorithm, or computer hardware) and the clients initiate requests to the server for access to the resource. Our system, based on the client-server model, is divided into three separate pieces of software: server, client, and remote interface. An overview of the system is illustrated in Fig. 1. The server software stores the problem (for example, genomic data and an algorithm to process it) and breaks the problem down into smaller problems, called work units. The client software is installed on each donor machine and it connects to the server over the Internet. The client requests a data unit, performs the processing, returns the result to the server, and requests another data unit. Multiple clients can make such requests to the server. The server collates the results of the smaller problems from the clients and constructs the result to the larger, original, problem. The remote interface is used to access all functionality on the server and can also be used to remotely update the client software.

2.1. Installation and Deployment

The entire system consists of three executable Java JAR files corresponding to the server, client, and remote interface. The user is only required to enter the server machine's IP address into a parameter file before running the server with the standard 'java' command. The client application can be run directly from the command line with all necessary start-up parameters passed in as arguments. The remote interface is a stand-alone GUI application that is started without any command line parameters; the user connects to the server using a simple 'point-and-click' interface. There is a full instructions document outlining how to setup and run the software available from the system web page.

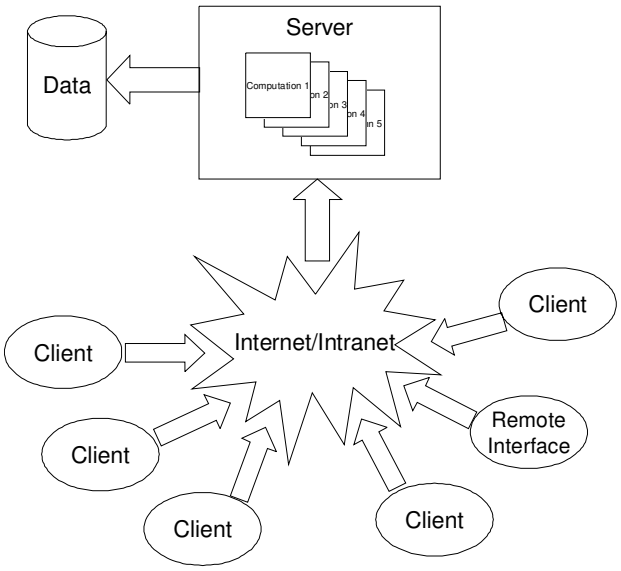

Figure 1: Diagram of the complete system. Although all communication is bi-directional, the arrows indicate the direction of initiation of communication.

To maximise the usage of the semi-idle desktop PC's in the deployment at NUI Maynooth, we chose to run the client as a low priority background service. This means that even if there is nobody logged on at a donor machine, the client software can run in the background 24 hours a day using only the spare clock cycles. We have our client software installed across a number of academic departments running on approximately 250 desktop PC's (various hardware specifications from Pentium II's up to Pentium IV's) running multiple operating systems (Windows 98/NT/2000/XP, Sun Solaris, Mac OSX, and Linux). To illustrate the portability of our system, we have also installed our client on every node of an IBM Linux cluster (32 Dual Pentium IV 1 GHz nodes with between 256 and 768 MB of memory per node) with the desktops and cluster nodes connecting to a single server.

2.2. Suitability of Bioinformatics to Distributed Computing

It has been widely acknowledged that there are a number of clearly identifiable characteristics that a problem should exhibit in order to be suitable for a distributed computing implementation. With the advent of many large scale Internet based supercomputing projects (e.g. [9,10]), a class of algorithmic parallelism referred to as 'coarse-grained parallelism' has emerged as a means of describing the suitability of problems to large scale distributed computing. Coarse grained parallelism refers to the way in which a single large problem can be easily split up into discrete independent sub-blocks that can be processed individually. The second criterion for evaluating the suitability of bioinformatics applications to distributed computing was that the problem must display a high "compute-to-data" ratio to make it worthwhile sending the data over a network rather than computing locally. We have identified these characteristics in several bioinformatics applications and outline two distributed bioinformatics applications that have been developed to run on our system in the following sections.

3. DPRml

One of the great challenges of molecular biology is the completion of the tree of life [14]. The massive accumulation of genomic data has led to increased interest in the production of large and accurate phylogenetic trees. However the decision problem associated with searching for the best tree from a set of taxa is NP-hard [4]. Therefore it is not feasible to perform an exhaustive search of the tree space for trees of a non-trivial size. Maximum likelihood (ML) evaluation has been widely acknowledged as one of the most accurate techniques for reconstructing phylogenies.

In an effort to construct large and accurate phylogenetic trees while still keeping overall processing times reasonable, a number of researchers have developed parallel ML programs that utilise the stepwise insertion approach [15,16]. These programs have been successful in speeding up phylogenetic computations but the overriding problem with these programs is that specialised parallel hardware and software is often required. For most researchers, this can make these programs either prohibitively expensive or simply too complicated to set up. Furthermore these programs are often implemented in a platform specific language which imposes a restrictive limit on the numbers and types of machines that can be used in a parallel computation. It should also be noted that some of these earlier parallel programs only allowed the user to choose from a very limited number of DNA substitution models, which often leads to a poor model fit resulting in sub optimal trees.

We have identified the suitability of phylogenetic analysis to heterogeneous distributed computing and have developed a fully cross-platform distributed application, DPRml [12], which we believe to be one of the most general and powerful likelihood-based phylogenetic tree building programs currently available. DPRml is, to our knowledge, the first distributed phylogenetic tree building program to satisfy each of the three requirements outlined above. The user has a very straightforward configuration file with which to tailor the computation and can choose from one of the most extensive ranges of DNA substitution models currently available. DPRml implements an already proven tree building algorithm and uses the popular Phylogenetic Analysis Library (PAL) v1.4 [17] for all its likelihood calculations.

Multiple DPRml computations can be submitted to the server, which allows users to always make optimal use of the available donor machines. To investigate the effect on speedup of running multiple DPRml computations in the distributed system, we completed a speedup graph using a university computing laboratory consisting of 40 desktop PC's (see Fig. 2) based on the running time of six simultaneous DPRml computations. For this test, we used one of the datasets that was used to test parallel fastDNAml (Stewart *et al.*, 2001), consisting of 50 taxa (1858 nucleotides per taxa), and ran six simultaneous computations with varying numbers of clients. Figure 2 demonstrates that DPRml achieves near linear speedup when speedup is measured with multiple DPRml computations running simultaneously. The above results fit well with the expected usage of the program. Typically a researcher would repeat the entire tree building process with several different randomisations of the taxon addition order and then compare the best of the resulting trees to determine a consensus tree.

3. DSEARCH

Database searching for similar sequences is one of the fundamental tasks in bioinformatics. One way to significantly reduce the runtime of sensitive database searches is to parallelise the search process across multiple processors. Many approaches to parallelising database searching have been investigated because database searching is both computationally intensive and easily parallelised. However the

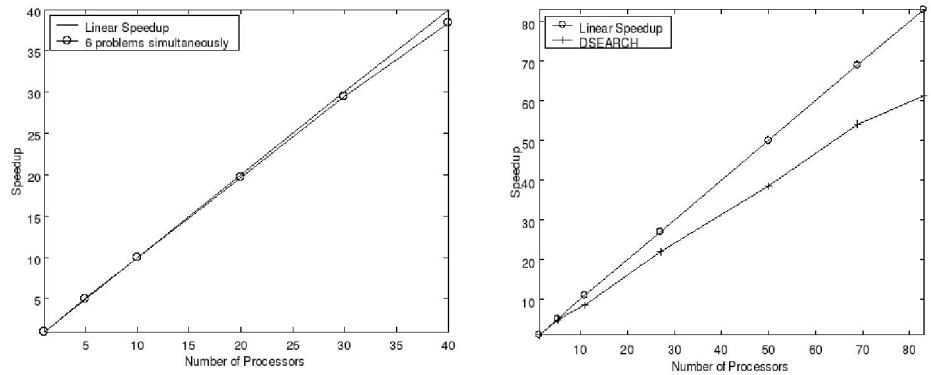

Figure 2: Speedup achieved by running 6 simultaneous DPRml problems using between 1-40 semi-idle processors. Speedup achieved by DSEARCH running on between 1-80 semi-idle processors.

overriding problem with many of these programs is that specialised parallel hardware and software is often required, making these programs either prohibitively expensive or simply too complicated to set up. DSEARCH is a fully cross-platform parallel database search program that does not require any specialised parallel hardware or software.

DSEARCH [11] operates in a master-slave environment and the search is parallelised by splitting the database into fixed sized units that are subsequently searched on the donor machines. The parallel granularity is dynamically controlled during each search to match the processing abilities of the current set of donor machines. The user edits a straightforward configuration file to tailor their computation and chooses one of the built-in search algorithms [2,3,18]. The inputs to the program are a FASTA database file, a FASTA query sequences file, a scoring scheme, and a configuration file. We completed a speedup graph using a university computing laboratory consisting of 80 desktop PC's (see Fig. 2) that shows how DSEARCH scales with increasing numbers of processors.

5. Conclusion

The explosion in the size of genomic databases in recent years has led to major computational challenges for bioinformatics researchers. Despite the development of faster and more efficient algorithms, it is not feasible to perform many common bioinformatics tasks on a single processor. Several vendors have offered specialized dedicated processor clusters in order to meet these computational challenges. However the cost of this hardware is often quite prohibitive for an ordinary researcher operating on a limited budget. To tackle this apparent failure, the area of distributed computing emerged as a viable alternative to dedicated parallel computing. By harnessing the spare clock cycles of idle machines, it is possible to emulate the computing power offered by a specialised parallel machine at a fraction of the cost.

We have presented a general-purpose distributed computing platform that is suitable for deployment in a typical university environment where semi-idle PC's are connected via a network. Our system is fully cross-platform compatible and has already been deployed across a number of academic departments at NUI Maynooth. We have identified a number of bioinformatics applications as being suitable for a distributed computing implementation. To date we have developed two distributed applications, DSEARCH and DPRml, which enable researchers to use the idle clock cycles of many machines simultaneously to perform

computationally intensive bioinformatics tasks. Our main goal for the future is to improve and expand the range of bioinformatics applications for the system.

6. Acknowledgements

This research has been funded by the Embark Initiative from the Irish Research Council for Science, Engineering and Technology: funded by the National Development Plan.

7. References

[1] Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Rapp, B.A. and Wheeler, D. L., "GenBank", Nucleic Acids Research, 2000, 28, 15-18

[2] Needleman, S.B and Wunsch, C.D., "A general method applicable to the search for similarities in the amino acid sequences of two proteins", Journal of Molecular Biology, 1970, 48, 443-453

[3] Smith, T.F. and Waterman, M.S., "Identification of common molecular subsequences", Journal of Molecular Biology, 1981, 147, 195-197

[4] Bodlaender, H., Fellows, M., and Warnow, T., "Two strikes against perfect phylogeny", Proceedings of the 19th International Colloquium on Automata, Languages, and Programming, Lecture Notes in Computer Science, 1992, 623, 273-283, Springer-Verlag, NY

[5] Felsenstein, J., "Inferring Phylogenies", Sinauer Associates Incorporated, Sunderland, MA, 2004, USA, ISBN- 0878-9317-75

[6] Hughey, R., "Parallel hardware for sequence comparison and alignment", Computer Applications in the Biosciences, 1996, 12, 473-479

[7] OpenMosix: A Linux kernel extension for single-system image clustering which turns a network of ordinary computers into a supercomputer, http://openmosix.sourceforge.net

[8] Haumacher, B., Moschny, T., Reuter, J., and Tichy, W.F., "Transparent Distributed Threads for Java", Proceedings of the 5th International Workshop on Java for Parallel and Distributed Computing in conjunction with the International Parallel and Distributed Processing Symposium, 2003, 136, Nice, France, IEEE Computer Society, ISBN-0769-5192-61

[9] Korpela, E., Werthimer, D., Anderson, D., Cobb, J., and Lebofsky, M., "SETI@home-Massively Distributed Computing for SETI", IEEE: Computer Science and Engineering, 2001, 3(1), 77-83

[10] Larson, S.M., Snow, C.D., Shirts, M.R., and Pande. V.S., "Folding@Home and Genome@Home: Using distributed computing to tackle previously intractable problems in computational biology", to appear in Computational Genomics, Richard Grant editor, Horizon Press

[11] Keane, T.M. and Naughton T.J., "DSEARCH: sensitive database searching using distributed computing", Bioinformatics, 2005, in press, doi:10.1093/bioinformatics/bti163

[12] Keane, T.M., Naughton, T.J., Travers, S.A.A, McInernery, J.O., McCormack, G.P., "DPRml: Distributed Phylogeny Reconstruction by maximum likelihood", Bioinformatics, 2005, 21(7):969-974

[13] Keane, T.M., "A General-Purpose Heterogeneous Distributed Computing System", M.Sc. Thesis, Department of Computer Science, National University of Ireland, Maynooth, 2004

[14] Crandall, K.A. and Buhay, J.E., "Genomic databases and the tree of life", Science, 2004, 306(5699), 1144-1145

[15] Stewart, C.A., Hart, D., Berry, D.K., Olsen, G.J., Wernert, E.A., and Fischer, W., "Parallel implementation and performance of fastDNAml – a program for maximum likelihood phylogenetic inference", Proceedings of SC2001, 2001, Denver, CO, USA

[16] Stamatakis, A.P., and Ludwig, T., "Phylogenetic Tree Inference on PC Architectures with AxML/PAxML", Proceedings of IPDPS2003 (High Performance Computational Biology workshop), 2003, 157, Nice, France

[17] Drummond, A. and Strimmer, K., "PAL: An object-oriented programming library for molecular evolution and phylogenetics", Bioinformatics, 2001, 17, 662-663

[18] Crochemore, M., Landau, G., and Ziv-Ukelson, M., "A Subquadratic Sequence Alignment Algorithm for Unrestricted Scoring Matrices", SIAM Journal of Computing, 2003, 32 (6), 1654-1673