High-Performance Algorithm Engineering for Large-Scale Graph Problems and Computational Biology

David A. Bader*

Electrical and Computer Engineering Department,
University of New Mexico, Albuquerque, NM 87131
dbader@ece.unm.edu

Abstract. Many large-scale optimization problems rely on graph theoretic solutions; yet high-performance computing has traditionally focused on regular applications with high degrees of locality. We describe our novel methodology for designing and implementing irregular parallel algorithms that attain significant performance on high-end computer systems. Our results for several fundamental graph theory problems are the first ever to achieve parallel speedups. Specifically, we have demonstrated for the first time that significant parallel speedups are attainable for arbitrary instances of a variety of graph problems and are developing a library of fundamental routines for discrete optimization (especially in computational biology) on shared-memory systems.

Phylogenies derived from gene order data may prove crucial in answering some fundamental questions in biomolecular evolution. High-performance algorithm engineering offers a battery of tools that can reduce, sometimes spectacularly, the running time of existing approaches. We discuss one such such application, GRAPPA, that demonstrated over a billion-fold speedup in running time (on a variety of real and simulated datasets), by combining low-level algorithmic improvements, cache-aware programming, careful performance tuning, and massive parallelism. We show how these techniques are directly applicable to a large variety of problems in computational biology.

1 Experimental Parallel Algorithms

We discuss our design and implementation of theoretically-efficient parallel algorithms for combinatorial (irregular) problems that deliver significant speedups on typical configurations of SMPs and SMP clusters and scale gracefully with the number of processors. Problems in genomics, bioinformatics, and computational ecology provide the focus for this research. Our source code is freely-available under the GNU General Public License (GPL) from our web site.

* This work was supported in part by NSF Grants CAREER ACI-00-93039, ITR ACI-00-81404, ITR EIA-01-21377,Biocomplexity DEB-01-20709, and ITR EF/BIO 03-31654; and DARPA contract NBCH30390004.

S.E. Nikoletseas (Ed.): WEA 2005, LNCS 3503, pp. 16–21, 2005.

1.1 Theoretically- and Practically-Efficient Portable Parallel Algorithms for Irregular Problems

Our research has designed parallel algorithms and produced implementations for primitives and kernels for important operations such as prefix-sum, pointer-jumping, symmetry breaking, and list ranking; for combinatorial problems such as sorting and selection; for parallel graph theoretic algorithms such as spanning tree, minimum spanning tree, graph decomposition, and tree contraction; and for computational genomics such as maximum parsimony (see [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]). Several of these classic graph theoretic problems are notoriously challenging to solve in parallel due to the fine-grained global accesses needed for the sparse and irregular data structures. We have demonstrated theoretically and practically fast implementations that achieve parallel speedup for the first time when compared with the best sequential implementation on commercially available platforms.

2 Combinatorial Algorithms for Computational Biology

In the 50 years since the discovery of the structure of DNA, and with new techniques for sequencing the entire genome of organisms, biology is rapidly moving towards a data-intensive, computational science. Many of the newly faced challenges require high-performance computing, either due to the massive-parallelism required by the problem, or the difficult optimization problems that are often combinatoric and NP-hard. Unlike the traditional uses of supercomputers for regular, numerical computing, many problems in biology are irregular in structure, significantly more challenging to parallelize, and integer-based using abstract data structures.

Biologists are in search of biomolecular sequence data, for its comparison with other genomes, and because its structure determines function and leads to the understanding of biochemical pathways, disease prevention and cure, and the mechanisms of life itself. Computational biology has been aided by recent advances in both technology and algorithms; for instance, the ability to sequence short contiguous strings of DNA and from these reconstruct the whole genome and the proliferation of high-speed microarray, gene, and protein chips for the study of gene expression and function determination. These high-throughput techniques have led to an exponential growth of available genomic data.

Algorithms for solving problems from computational biology often require parallel processing techniques due to the data- and compute-intensive nature of the computations. Many problems use polynomial time algorithms (e.g., all-to-all comparisons) but have long running times due to the large number of items in the input; for example, the assembly of an entire genome or the all-to-all comparison of gene sequence data. Other problems are compute-intensive due to their inherent algorithmic complexity, such as protein folding and reconstructing evolutionary histories from molecular data, that are known to be NP-hard (or harder) and often require approximations that are also complex.

3 Phylogeny Reconstruction

A phylogeny is a representation of the evolutionary history of a collection of organisms or genes (known as taxa). The basic assumption of process necessary to phylogenetic reconstruction is repeated divergence within species or genes. A phylogenetic reconstruction is usually depicted as a tree, in which modern taxa are depicted at the leaves and ancestral taxa occupy internal nodes, with the edges of the tree denoting evolutionary relationships among the taxa. Reconstructing phylogenies is a major component of modern research programs in biology and medicine (as well as linguistics). Naturally, scientists are interested in phylogenies for the sake of knowledge, but such analyses also have many uses in applied research and in the commercial arena.

Existing phylogenetic reconstruction techniques suffer from serious problems of running time (or, when fast, of accuracy). The problem is particularly serious for large data sets: even though data sets comprised of sequence from a single gene continue to pose challenges (e.g., some analyses are still running after two years of computation on medium-sized clusters), using whole-genome data (such as gene content and gene order) gives rise to even more formidable computational problems, particularly in data sets with large numbers of genes and highly-rearranged genomes.

To date, almost every model of speciation and genomic evolution used in phylogenetic reconstruction has given rise to NP-hard optimization problems. Three major classes of methods are in common use. Heuristics (a natural consequence of the NP-hardness of the problems) run quickly, but may offer no quality guarantees and may not even have a well-defined optimization criterion, such as the popular *neighbor-joining* heuristic [13]. Optimization based on the criterion of *maximum parsimony* (MP) [14] seeks the phylogeny with the least total amount of change needed to explain modern data. Finally, optimization based on the criterion of *maximum likelihood* (ML) [15] seeks the phylogeny that is the most likely to have given rise to the modern data.

Heuristics are fast and often rival the optimization methods in terms of accuracy, at least on datasets of moderate size. Parsimony-based methods may take exponential time, but, at least for DNA and amino acid data, can often be run to completion on datasets of moderate size. Methods based on maximum likelihood are very slow (the point estimation problem alone appears intractable) and thus restricted to very small instances, and also require many more assumptions than parsimony-based methods, but appear capable of outperforming the others in terms of the quality of solutions when these assumptions are met. Both MP- and ML-based analyses are often run with various heuristics to ensure timely termination of the computation, with mostly unquantified effects on the quality of the answers returned.

Thus there is ample scope for the application of high-performance algorithm engineering in the area. As in all scientific computing areas, biologists want to study a particular dataset and are willing to spend months and even years in the process: accurate branch prediction is the main goal. However, since all exact algorithms scale exponentially (or worse, in the case of ML approaches) with the

number of taxa, speed remains a crucial parameter—otherwise few datasets of more than a few dozen taxa could ever be analyzed.

As an illustration, we briefly discuss our experience with a high-performance software suite, GRAPPA (Genome Rearrangement Analysis through Parsimony and other Phylogenetic Algorithms) that we developed, *GRAPPA* extends Sankoff and Blanchette's breakpoint phylogeny algorithm [16] into the more biologically-meaningful inversion phylogeny and provides a highly-optimized code that can make use of distributed- and shared-memory parallel systems (see [17, 18, 19, 20, 21, 22] for details). In [23] we give the first linear-time algorithm and fast implementation for computing inversion distance between two signed permutations. We ran *GRAPPA* on a 512-processor IBM Linux cluster with Myrinet and obtained a 512-fold speed-up (linear speedup with respect to the number of processors): a complete breakpoint analysis (with the more demanding inversion distance used in lieu of breakpoint distance) for the 13 genomes in the Campanulaceae data set ran in less than 1.5 hours in an October 2000 run, for a *million-fold* speedup over the original implementation. Our latest version features significantly improved bounds and new distance correction methods and, on the same dataset, exhibits a speedup factor of *over one billion*. We achieved this speedup through a combination of parallelism and high-performance algorithm engineering. Although such spectacular speedups will not always be realized, we suggest that many algorithmic approaches now in use in the biological, pharmaceutical, and medical communities can benefit tremendously from such an application of high-performance techniques and platforms.

This example indicates the potential of applying high-performance algorithm engineering techniques to applications in computational biology, especially in areas that involve complex optimizations: our reimplementation did not require new algorithms or entirely new techniques, yet achieved gains that turned an impractical approach into a usable one.

References

1. Bader, D., Illendula, A., Moret, B.M., Weisse-Bernstein, N.: Using PRAM algorithms on a uniform-memory-access shared-memory architecture. In Brodal, G., Frigioni, D., Marchetti-Spaccamela, A., eds.: Proc. 5th Int'l Workshop on Algorithm Engineering (WAE 2001). Volume 2141 of Lecture Notes in Computer Science., Århus, Denmark, Springer-Verlag (2001) 129–144
2. Bader, D., Moret, B., Sanders, P.: Algorithm engineering for parallel computation. In Fleischer, R., Meineche-Schmidt, E., Moret, B., eds.: Experimental Algorithmics. Volume 2547 of Lecture Notes in Computer Science. Springer-Verlag (2002) 1–23
3. Bader, D., Sreshta, S., Weisse-Bernstein, N.: Evaluating arithmetic expressions using tree contraction: A fast and scalable parallel implementation for symmetric multiprocessors (SMPs). In Sahni, S., Prasanna, V., Shukla, U., eds.: Proc. 9th Int'l Conf. on High Performance Computing (HiPC 2002). Volume 2552 of Lecture Notes in Computer Science., Bangalore, India, Springer-Verlag (2002) 63–75

4. Bader, D.A., Cong, G.: A fast, parallel spanning tree algorithm for symmetric multiprocessors (SMPs). In: Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS 2004), Santa Fe, NM (2004)
5. Bader, D.A., Cong, G.: A fast, parallel spanning tree algorithm for symmetric multiprocessors (SMPs). Journal of Parallel and Distributed Computing (2004) to appear.
6. Bader, D.A., Cong, G.: Fast shared-memory algorithms for computing the minimum spanning forest of sparse graphs. In: Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS 2004), Santa Fe, NM (2004)
7. Cong, G., Bader, D.A.: The Euler tour technique and parallel rooted spanning tree. In: Proc. Int'l Conf. on Parallel Processing (ICPP), Montreal, Canada (2004) 448–457
8. Su, M.F., El-Kady, I., Bader, D.A., Lin, S.Y.: A novel FDTD application featuring OpenMP-MPI hybrid parallelization. In: Proc. Int'l Conf. on Parallel Processing (ICPP), Montreal, Canada (2004) 373–379
9. Bader, D., Madduri, K.: A parallel state assignment algorithm for finite state machines. In: Proc. 11th Int'l Conf. on High Performance Computing (HiPC 2004), Bangalore, India, Springer-Verlag (2004)
10. Cong, G., Bader, D.: Lock-free parallel algorithms: An experimental study. In: Proc. 11th Int'l Conf. on High Performance Computing (HiPC 2004), Bangalore, India, Springer-Verlag (2004)
11. Cong, G., Bader, D.: An experimental study of parallel biconnected components algorithms on symmetric multiprocessors (SMPs). Technical report, Electrical and Computer Engineering Department, The University of New Mexico, Albuquerque, NM (2004) Submitted for publication.
12. Bader, D., Cong, G., Feo, J.: A comparison of the performance of list ranking and connected components algorithms on SMP and MTA shared-memory systems. Technical report, Electrical and Computer Engineering Department, The University of New Mexico, Albuquerque, NM (2004) Submitted for publication.
13. Saitou, N., Nei, M.: The neighbor-joining method: A new method for reconstruction of phylogenetic trees. Molecular Biological and Evolution **4** (1987) 406–425
14. Farris, J.: The logical basis of phylogenetic analysis. In Platnick, N., Funk, V., eds.: Advances in Cladistics. Columbia Univ. Press, New York (1983) 1–36
15. Felsenstein, J.: Evolutionary trees from DNA sequences: a maximum likelihood approach. J. Mol. Evol. **17** (1981) 368–376
16. Sankoff, D., Blanchette, M.: Multiple genome rearrangement and breakpoint phylogeny. Journal of Computational Biology **5** (1998) 555–570
17. Bader, D., Moret, B., Vawter, L.: Industrial applications of high-performance computing for phylogeny reconstruction. In Siegel, H., ed.: Proc. SPIE Commercial Applications for High-Performance Computing. Volume 4528., Denver, CO, SPIE (2001) 159–168
18. Bader, D., Moret, B.M., Warnow, T., Wyman, S., Yan, M.: High-performance algorithm engineering for gene-order phylogenies. In: DIMACS Workshop on Whole Genome Comparison, Piscataway, NJ, Rutgers University (2001)
19. Moret, B., Bader, D., Warnow, T.: High-performance algorithm engineering for computational phylogenetics. J. Supercomputing **22** (2002) 99–111 Special issue on the best papers from ICCS'01.
20. Moret, B., Wyman, S., Bader, D., Warnow, T., Yan, M.: A new implementation and detailed study of breakpoint analysis. In: Proc. 6th Pacific Symp. Biocomputing (PSB 2001), Hawaii (2001) 583–594

21. Moret, B.M., Bader, D., Warnow, T., Wyman, S., Yan, M.: GRAPPA: a high-performance computational tool for phylogeny reconstruction from gene-order data. In: Proc. Botany, Albuquerque, NM (2001)
22. Yan, M.: High Performance Algorithms for Phylogeny Reconstruction with Maximum Parsimony. PhD thesis, Electrical and Computer Engineering Department, University of New Mexico, Albuquerque, NM (2004)
23. Bader, D., Moret, B., Yan, M.: A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. Journal of Computational Biology **8** (2001) 483–491