

Bringing large-scale multiple genome analysis one step closer: ScalaBLAST and beyond

Christopher S. Oehmen¹, Heidi J. Sofia¹, Douglas Baxter², Ernest Szeto³, Philip Hugenholtz⁴, Nikos Kyrpides⁵, Victor Markowitz³, Tjerk P. Straatsma¹

¹Computational Sciences and Mathematics Division, Pacific Northwest National Laboratory (PNNL), 902 Battelle Boulevard, P.O. Box 999, Richland, WA USA

²William R. Wiley Environmental Molecular Sciences Laboratory, Pacific Northwest National Laboratory (PNNL), 902 Battelle Boulevard, P.O. Box 999, Richland, WA USA

³Biological Data Management and Technology Center, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, USA

⁴Microbial Ecology Program, Department of Energy Joint Genome Institute, 2800 Mitchell Drive, Walnut Creek, USA

⁵Microbial Genome Analysis Program, Department of Energy Joint Genome Institute, 2800 Mitchell Drive, Walnut Creek, USA

Genome sequence comparisons of exponentially growing data sets form the foundation for the comparative analysis tools provided by community biological data resources such as the Integrated Microbial Genome (IMG) system at the Joint Genome Institute (JGI). We present an example of how ScalaBLAST, a high-throughput sequence analysis program harnesses increasingly critical high-performance computing to perform sequence analysis which is a critical component of maintaining a state-of-the-art sequence data repository.

The Integrated Microbial Genomes (IMG) system¹ is a data management and analysis platform for microbial genomes hosted at the JGI. IMG contains both draft and complete JGI genomes integrated with other publicly available microbial genomes of all three domains of life. IMG provides tools and viewers for interactive analysis of genomes, genes and functions, individually or in a comparative context. Most of these tools are based on pre-computed pairwise sequence similarities involving millions of genes. These computations are becoming prohibitively time consuming with the rapid increase in the number of newly sequenced genomes incorporated into IMG and the need to refresh regularly the content of IMG in order to reflect changes in the annotations of existing genomes. Thus, building IMG 2.0 (released on December 1st 2006) entailed reloading from NCBI's RefSeq all the genomes in the previous version of IMG (IMG 1.6, as of September 1st, 2006) together with 1,541 new public microbial, viral and eukaryal genomes, bringing the total of IMG genomes to 2,301. A critical part of building IMG 2.0 involved using PNNL ScalaBLAST software for computing pairwise similarities for over 2.2 million genes in under 26 hours on 1,000 processors, thus illustrating the impact that new generation bioinformatics tools are poised to make in biology. The BLAST algorithm^{2,3} is a familiar bioinformatics application for computing sequence similarity, and has become a workhorse in large-scale genomics projects. The rapid growth of genome resources such as IMG cannot be sustained without more powerful tools such as ScalaBLAST that use more effectively large scale computing resources to perform the core BLAST calculations.

ScalaBLAST is a high performance computing algorithm designed to give high throughput BLAST results on high-end supercomputers. Other parallel sequence comparison applications have been developed⁴⁻⁶. However problems with scaling generally prevent these applications from being used for very large searches. ScalaBLAST⁷ is the first BLAST application to be both highly scaleable against the size of the database as well as the number of computer processors on high-end hardware and on commodity clusters. ScalaBLAST achieves high throughput by parsing a large collection of query sequences into independent subgroups. These smaller tasks are assigned to independent process groups. Efficient scaling is achieved by (transparently to the user) sharing only one copy of the target database across all processors using the Global Array toolkit^{8,9}, which provides software implementation of shared memory interface. ScalaBLAST was initially deployed on the 1,960 processor MPP2 cluster in the William R. Wiley Environmental Molecular Sciences Laboratory at Pacific Northwest National Laboratory, and has since been ported to a variety of linux-based clusters and shared memory architectures, including SGI Altix, AMD opteron, and Intel Xeon-based clusters. Future targets include IBM BlueGene, Cray, and SGI Altix XE architectures.

The importance of performing high-throughput calculations *rapidly* lies in the rate of growth of sequence data. For a genome sequencing center to provide multiple-genome comparison capabilities, it must keep pace with exponentially growing collection of protein data, both from its own genomes, and from the public genome information as well. As sequence data continues to grow exponentially, this challenge will only increase with time. Solving the BLAST throughput challenge for centralized data resources like IMG has the potential to unlock the power of emerging analysis methods which, until recently, were limited by the availability of multiple genome comparison data. Fig. 1 illustrates how the run-time achieved by efficient scaling in ScalaBLAST enabled the IMG all vs. all BLAST calculations to complete in roughly 1 day. Note that to keep pace with growing IMG database, we will have to double the number of processors used in these calculations during the upcoming year.

Grid-based solutions for improving throughput for BLAST searches has become a popular and attractive option for some centers. The Institute for Genome Research (<http://www.tigr.org/>), for instance, has implemented a grid-based BLAST tool allowing users to submit requests to be farmed out to available computers on an on-demand basis. Likewise open access packages like Squid¹⁰ allow users to setup their own grid-based BLAST engines. When the goal is to enhance throughput for a large collection of users having separate BLAST tasks, Grid-enabled approaches have great potential to improve time-to-solution for individual users.

Another popular alternative is to use a scripting language to schedule independent BLAST runs on a cluster. In this approach, one might submit a single job to a resource scheduler, and when the processors are allocated for that job, attempt to spawn separate BLAST instances on each processor, each with the same database but a separate partition of the user's queries.

ScalaBLAST provides a powerful alternative to grid-based approaches and multiple-instance BLAST scripts for users with very large jobs containing thousands, millions or more queries directed at a large database (though ScalaBLAST is still highly efficient on small databases as well). For grid-based or parallel script users, running separate instances of BLAST requires separate copies of the same database to be present on each processor. For very large databases broken into multiple volumes, this forces each processor to swap in and out *each* database volume repeatedly for *each* query rather than loading each database piece only once, as is done in ScalaBLAST. Also, grid- or script-based approaches often result in a large number of repeated sequential system calls to a BLAST computational core severely limiting data reuse. By contrast, ScalaBLAST takes advantage of the homogeneity of the tasks by only keeping a single ‘shared’ copy of the target database in aggregate memory; and because it encapsulates the collection of tasks, ScalaBLAST does not have to reload the database anew for each separate task. Efficient memory management cuts down dramatically on the memory requirement of the computing platform, and eliminates repeatedly having to pay the startup penalty for launching separate BLAST jobs taking full advantage of the potential for data reuse. When processing millions of sequences, this combination of features results in a substantial improvement in time-to-solution and prevents adverse affects on other users of multi-user systems stemming from repeated swapping, filesystem, and memory bandwidth limitations.

The importance of efficient memory management in ScalaBLAST cannot be overstated as typical datasets continue to grow at an exponential pace—doubling every 18 months on average. One may be tempted to assume that since we can approach some of the larger bioinformatics search problems today, and that since computational processing speed doubles about every 18 months as well, we should be able to accommodate these searches indefinitely. However, this is not at all the case for several reasons. Sequence alignment is much more related to memory moving (pushing sequences past growing databases) than to processing speed (ability to perform floating point arithmetic). It typically takes hardware vendors 3 years to deliver computing systems where the *memory bandwidth* has doubled over previous systems¹¹—so it does not keep pace with the growth of sequence datasets. In that same 3 year span, *latency*, or the lag time associated with random memory operations, only improves by 20%¹¹ creating another barrier to rapid sequence analysis unless careful attention is paid to hiding latency. Another consideration is that the time required for all vs. all search task which is at the core of multiple-genome bioinformatics is proportional to the *square* of the data size. So every 18 months, (using the same computing resources), one should expect a factor of 4 slower time to solution. Fig. 2 illustrates the compound effect of growing databases and lagging improvements in memory bandwidth on the computational run-time required to complete all vs. all BLAST calculations. *Scaling demand* is an indicator of the number of processors required to complete an entire all vs. all BLAST calculation in 1 day for a given database size and memory bandwidth performance. Because of these compounding technical issues, problems which are tractable today will soon be beyond reach unless significant advancements in scalability and computational efficiency can be made. Applications which do not scale efficiently on multi-processor architectures and

effectively manage memory will increasingly be plagued by disparity between computational performance and the computational demands of bioinformatics.

Performing of BLAST calculations on an exponentially growing number of genes has long been the computational bottleneck of multiple genome analysis. This problem will become even more difficult to address as thousands of microbial isolate genomes become available together with a rapidly growing number of microbial community aggregate genomes (metagenomes)¹².

The impact of high-throughput technology like ScalaBLAST is that it removes the sequence alignment bottleneck from downstream applications which rely on the availability of BLAST output. Emerging analysis methods which rely on large volume of BLAST output can now be used with regularity on up-to-date multiple genome datasets. As these analysis methods continue to mature, high-throughput sequencing becomes increasingly valuable. ScalaBLAST helps bridge the gap between high-throughput sequencing and advances in multi-genome analysis.

Acknowledgments

This research was supported by the Data-intensive Computing for Complex Biological Systems project funded by the *Office of Advanced Scientific Computing Research*, and under the *LDRD Program* at the Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy under Contract DE-AC06-76RL01830; and by the Director, Office of Science, Office of Biological and Environmental Research, Life Sciences Division, U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research was performed in part using the MSCF in EMSL, a national scientific user facility sponsored by the U.S. DOE, OBER and located at PNNL.

1. Markowitz, V. et al. The Integrated Microbial Genomes (IMG) System. *Nucleic Acids Res* **34**, D344-D348 (2006).
2. Altschul, S., Gish, W., Miller, W., Myers, E. & Lipman, D. Basic local alignment search tool. *J. Mol. Biol.* **215**, 403-410 (1990).
3. Altschul, S. et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389-3402 (1997).
4. Darling, A., Carey, L. & Feng, W.-C. in Proceedings of ClusterWorldSan Jose, CA; 2003).
5. Lin, H., Ma, X., Chandramohan, P., Geist, A. & Samatova, N. in 19th International Parallel and Distributed Processing Symposium (IPDPS) (IEEE CS Press, Denver, CO; 2005).
6. Wang, J. & Mu, Q. Soap-HT-BLAST: high-throughput BLAST based on web services. *Bioinformatics* **19**, 1863-1864 (2003).
7. Oehmen, C. & Nieplocha, J. ScalaBLAST: A scalable implementation of BLAST for High Performance Data-Intensive Bioinformatics Analysis. *IEEE Trans. Parallel. Dist. Sys.* **in press** (2006).

8. Nieplocha, J., Harrison, R. & Littlefield, R. Global Arrays: A nonuniform memory access programming model for high-performance computers. *J. Supercomputing* **10**, 197-220 (1996).
9. Nieplocha, J., Krishnan, M., Palmer, B., Tipparaju, V. & Zhang, Y. in ACM SIGMicro Computing Frontiers2005).
10. Carvalho, P., Gloria, R., de Miranda, A. & Degrave, W. Squid - a simple bioinformatics grid. *BMC Bioinformatics* **6**, 197 (2005).
11. Patterson, D. Latency lags bandwidth: Recognizing the chronic imbalance between bandwidth and latency, and how to cope with it. *Comm. ACM.* **47**, 71-75 (2004).
12. Markowitz, V. et al. An experimental metagenome data management and analysis system. *Bioinformatics* **22**, e359-367 (2006).

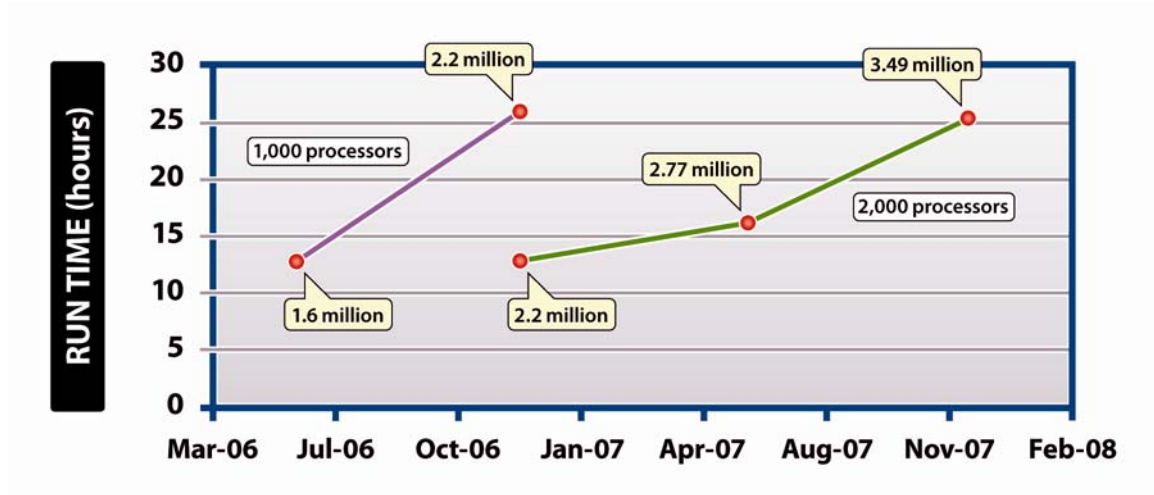


Figure 1. Keeping pace with growing data. ScalaBLAST runs using 1000 processors have enabled all vs. all BLAST runs to be done on 1.6 million and 2.2 million proteins for IMG 1.6 and 2.0 releases, respectively, each in roughly 1 day or less. To maintain IMG updates at a comparable run time, ScalaBLAST will have to be run on 2000 processors, which is within its capacity.

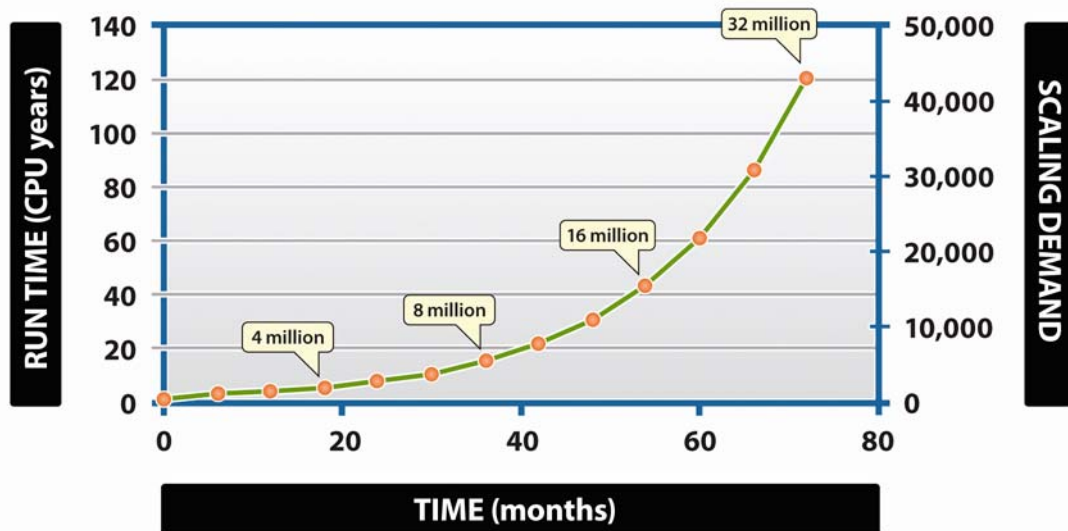


Figure 2. Demand for parallel BLAST. Starting from the approximate current size of IMG at 2 million proteins and assuming a doubling time of 18 months, computing time needed to perform an all vs. all calculation grows exponentially even though compute power increases with time. Scaling demand is calculated as the number of processors required to perform an all vs. all BLAST run within 24 hours at the expected memory bandwidth capacity available at the time of the run. ScalaBLAST scales to thousands of

processors, but increased scaling demand will require running on tens of thousands of processors within 2 years. Callouts indicate anticipated database size over time.