

GADU – Genome Analysis and Database Update Pipeline

Alex Rodriguez* **Dinanath Sulakhe** **Elizabeth Marland**
arodri7@mcs.anl.gov sulakhe@mcs.anl.gov marland@mcs.anl.gov

Veronika Nefedova **Gong Xin Yu** **Natalia Maltsev***
nefedova@mcs.anl.gov gxyu@mcs.anl.gov maltsev@mcs.anl.gov

Mathematics and Computer Science Division, Argonne National
Laboratory

*Corresponding Authors

Abstract

Realizing the enormous scientific potential of exponentially growing biological information requires the development of high-throughput automated computational environments that integrate large amounts of genomic and experimental data, and powerful tools for knowledge discovery and data mining. To assist high-throughput analysis of the genomes, we have developed the Genome Analysis and Databases Update system. GADU efficiently automates major steps of genome analysis: data acquisition and data analysis by a variety of tools and algorithms, as well as data storage and annotation. We are developing a TeraGrid technology-based backend for large-scale computations using GADU. GADU can function in either an automated or interactive mode via a Web-based user interface. Programs monitor every operation in GADU and report the status of the process. This architecture ensures GADU's robust performance and allows simultaneous processing of a large number of sequenced genomes regardless of their size.

Introduction

During the past decade, the scientific community has witnessed an unprecedented accumulation of gene sequence data and data related to the physiology and biochemistry of organisms. To date, more than 120 genomes have been sequenced and genomes of 587 organisms are at various levels of completion [GOLD <http://wit.integratedgenomics.com/GOLD/>]. In order to exploit the enormous scientific value of this information for understanding biological systems, the information should be integrated, analyzed, graphically displayed, and ultimately modeled computationally (1). The emerging systems biology approach requires the development of high-throughput computational environments that integrate (i) large amounts of genomic and experimental data and (ii) powerful tools and algorithms for knowledge discovery and data mining. Most of these tools and algorithms are very CPU-intensive and require substantial computational resources that are not always available to the researchers. The large-scale, distributed computational and storage infrastructure of the TeraGrid offers an ideal platform for mining such large volumes of biological information. Indeed, targeted performance of NSF Distributed Terascale Facility and DOE Science Grid amounts to trillions of floating-point operations per second (teraflops) and storage of hundreds terabytes of data (2).

The first and most crucial step in genome analysis is the assignment of function to genes. Efficiency and accuracy of such predictions are achieved by the use of a variety of state-of-

the-art bioinformatics tools and approaches (e.g., analysis of global similarities (3) (4) (5), domain and motif analysis (6) (7) (8), and analysis of the relevant structural (9) (10) and functional information). Manual acquisition of diverse data inputs, data submission to bioinformatics tools, and collection and processing of the results of analyses required for such high-throughput genome annotations is an extremely tedious and time-consuming effort and is prone to human error.

To address this problem, we have developed the Genome Analysis and Databases Update (GADU), which is an automated, high-performance, scalable computational pipeline for data acquisition and analysis of newly sequenced genomes. GADU allows efficient automation of major steps of genome analysis: data acquisition and data analysis by variety of tools and algorithms, as well as data storage and annotation. GADU can be used as a stand-alone server, or it can be incorporated into established frameworks of other systems for analysis of large volumes of sequence data (e.g., WIT (11) in our case). GADU consists of three conceptual modules: data acquisition module, data analysis, and data storage modules (Figure 1). We are developing a TeraGrid technology-based backend for large-scale computations using GADU. Programs monitor every operation in GADU and report the status of the process and possible errors. This architecture ensures GADU's performance and allows simultaneous processing of a large number of sequenced genomes regardless of their size.

Methods

1. Implementation

Our goal for GADU was to design a flexible architecture that allows modifying and adjusting the genome analysis process according to the needs and requirements of a particular user. The advantage of this system is that it can function both in automated mode and interactively, via a Web-based interface. This feature is especially important for users who are analyzing genomes or using datasets and data libraries that are not found in public databases.

2. Acquisition Module

Information relevant to life sciences has been accumulated in specialized data repositories (e.g., GenBank (12), SwissProt (13), PDB (14), EMBL (15)). These repositories have developed query-processing capabilities and sophisticated data formats enable efficient navigation of specific classes of biological information. In many cases, data from one source often must be combined with data from other sources and warehoused locally, in order to give users information required for efficient analysis of the genomes.

In this section we explain what types of data are acquired and updated by GADU, and we describe the steps of the data acquisition process (Figure 2).

The GADU *data acquisition module* executes periodic updates of the local server with genome data and annotations stored in the public databases such as the NCBI (<http://www.ncbi.nlm.nih.gov>), JGI (<http://www.jgi.doe.gov>), TIGR (<http://www.tigr.org>), PDB, and SwissProt. The frequency of the update search is specified by the user). GADU compares the content of public databases' directories with local genome storage directories, creates a list of the new and updated genomes found in the databases' ftp directories, notifies the user via e-mail. Upon notification, the user may select, via Web

interface, the data to be downloaded and stored locally (Figure 3). Flat files containing DNA and amino acid sequence data in Fasta format, as well as annotations relevant to particular genome, are automatically downloaded to the local directories using the ftp protocol. The sizes of the files vary from genome to genome. The average file size downloaded per genome is 30 MB. Files containing sequences in Fasta format are passed to the *data analysis module* (see below) for submission to bioinformatics tools. Information from the files containing sequence annotations (e.g., NCBI's GenBank .gbk files or SwissProt releases) is extracted by the parsers of the data storage module and automatically deposited in the local Oracle database.

A number of bioinformatics tools (e.g., Blast and InterPro) require use of database files or particular libraries for their performance. GADU periodically updates these files as well and notifies the user, via e-mail, regarding performed updates.

GADU can be also used to download data manually. For manual download the user specifies the location of the data to be incorporated into GADU (figure 3) from the Web interface. A user log is generated containing the latest uploads to the local directory. Protocols of downloads by GADU are automatically submitted to GADU management tables in a database. Use of GADU via the Web interface requires user authentication.

3. Analysis Module

Downloading of the sequences in appropriate format via the *data acquisition module* triggers the *data analysis module*, and the process of submission of the sequence data to bioinformatics tools begins. Currently, GADU allows sequence analysis by three bioinformatics tools: Blast, Blocks, and Pfam. We plan to expand GADU by adding other tools of interest to scientific community (e.g., InterPro, TMHMM (16), and Psort (17)). The user can choose a set of tools for analysis. Every time a new or updated genome is downloaded to the local directory, GADU creates a small file containing information required for automated submission of the genome to the tools (e.g., taxonomic name of the organism, location of the genome's fasta file in the local directory, quantity of sequences in the genome) and informs the user regarding the content of the data (Figure 4),

Once the user chooses the genomes to analyze and the tools to be used for the analysis, the selected data is deposited to a queue for further submission in parallel to the high-performance computing (HPC) resources via PBS jobs. Currently GADU uses two computational resources:

- Chiba City cluster – a 512-CPU Linux cluster with 500 MHz and 512 MB of RAM at Argonne National Laboratory
- Condor pools – 18 Condor pools at the University of Wisconsin-Madison

To increase the scalability of the system, we are implementing a GADU TeraGrid technology-based backend.

For each job that GADU submits to the HPC clusters, GADU creates a working space specific to that job. The working space contains the input sequences to be submitted to the tools and, upon completion of analysis, output from the tools. A number of checkpoints throughout the process ensure that each step of analysis is completed successfully. One critical checkpoint is the point of submission of the batch job to the HPC cluster. At this point, GADU expects to receive a job

identification number from the cluster. If the expected number is not received, the process is terminated, and GADU automatically restarts the submission process. Another checkpoint is triggered once GADU receives a signal that the job has been submitted. GADU must make sure that the job is successfully completed. Depending on the job status received from the checker (active, idle, blocked or completed), GADU performs different functions. If the job is in “active” status, the cluster is still processing the job, and GADU takes no further action at that moment. If the job’s state is “idle,” the cluster is searching for the necessary quantity of nodes for the job to be processed, and once again GADU takes no further action. If the job is found to be in the “blocked” status, something has gone wrong with the job processing, and GADU terminates the job. An e-mail message will notify the GADU administrator that a job was blocked and terminated, so the administrator can correct the error. Finally, if the status of the job is “completed,” the job ID number cannot be found in the cluster’s queue of jobs, and thus GADU assumes that the job has been successfully completed. To ensure that this is the case, GADU checks the working directory that contained the input data files and output of the tools. First, it checks that all expected output files were generated and that there are no empty files. If this check fails, GADU deletes the complete working directory and resubmits the job to the clusters. Once the check is successful, GADU is ready to move on to the *storage module*.

4. Storage Module

Storage processes in GADU are triggered automatically by completion of the processes by data acquisition module or by the data analysis module. For example, successful download of a new annotated genome from NCBI triggers parsing of related information into the Oracle database. In other cases, successful completion of the Blast analysis of particular genome invokes output-parsing scripts in storage module.

The storage module in GADU is represented by the following three entities:

a. Permanent storage contains sequence data and annotations acquired by GADU from public databases. We have developed an entity relationships model (ERM) that allows integrating and warehousing of various types of annotations in one searchable environment. We have also developed parsers that extract information from the database files and loader programs that deposit the extracted information in the local Oracle database. Parsers for the outputs of the bioinformatics tools (e.g., Blast and Blocks) have also been developed. The information extracted from the output files is further used for analyses by additional algorithms or visualization programs.

b. Data archive stores flat files containing different releases of genomes and annotations in the form of tape archives. Currently GADU has deposited 110 GB of data in our local archive. This information might be especially useful during analyses of incomplete genomes for maintaining continuity between releases and for future use by additional algorithms.

c. Temporary storage contains acquired data and intermediate outputs of the parsers and analysis tools. We are exploring the possibility of using the TeraGrid resources for storing space-consuming files and archives.

The data storage module also accumulates “bookkeeping” information regarding GADU functionality, such as data acquisition protocols and records regarding data processing and analysis.

Results

We present some examples demonstrating the usefulness of GADU's automated system for genome analysis. From our experience, the manual process of analyzing a complete prokaryotic genome (average number of ORFs equals 4,000) from the time of genome acquisition, through analysis by select tools (Blast, Blocks, and Pfam), to depositing of the results into a relational database takes anywhere from 6 to 8 working hours by a qualified staff member. Automation of these processes in GADU minimizes the amount of user intervention (Table 1). Thus, the processes can be run back to back over a 24-hour period without interruption. Automated dataflow in GADU is reliable and stable because of its multiple checking programs, and the user can be assured that the process will be reinitiated in case of an error. This increase in throughput and robustness of the processes of genome analysis is important, in view of the exponential growth of the unannotated sequence data coming from the sequencing projects. Another important application of GADU is periodic updates and reanalysis of data using new information available in the databases.

Using GADU and the Blast, Blocks, and Pfam tools, we have analyzed and deposited in our local (Oracle) database 106 genomes from NCBI. These results will be used for automated assignments of functions to genes and further annotations of these genomes [<http://compbio.mcs.anl.gov>] in WIT3 [<http://compbio.mcs.anl.gov/wit3>].

Conclusions and Future Plans

As shown above, the use of the automated GADU system can substantially decrease the time and the amount of user interaction required for genome analysis. Its modular architecture permits performing different steps of genome analysis efficiently. This feature is especially useful for simultaneous analysis of multiple genomes. For example, the results of one completed genome analysis may enter the data storage process while data analysis module analyzes other genomes.

Availability of new experimental results concerning functions of proteins, previously annotated as hypothetical, as well as improvements in sensitivity and accuracy of bioinformatics tools, requires periodic revisiting of previously annotated genomes and reassignment of function using this newly acquired knowledge. The increased efficiency of genome analysis offered by the GADU system considerably simplifies the analysis of newly sequenced genomes as well as the previously annotated genomes.

We plan to further develop GADU so it will become a high-throughput, batch genome analysis server to be used by the scientific community. Such a server should have a flexible Web-based user interface that will allow users to analyze sequence data of interest in the best way that meets their scientific goals. For this purpose, we are working on the following improvements to the GADU architecture:

Scalability. Addition of the computational resources to the GADU pipeline can substantially increase its performance (Figure 5).

In collaboration with the Globus Project (19) we are working on an implementation of GADU based on distributed computing technology as a computational backend for genome analysis. We plan to use the following TeraGrid resources: DOE Science Grid. We plan to use a variety of distributed computational resources. As the first step we will adopt the GADU system for Argonne's Data Grid Cluster (20 dual-processor PIII 850 MHz nodes with 512 MB RAM, 2 TB of storage, and 2.4.19-rc3-dg kernel). The next step will be to transfer the most computationally

intensive parts of the GADU pipeline (Blast) into the TeraGrid architecture (which is IA64). The use of 64-bit architecture with its native compiler should produce a tenfold speedup of our computations. We have already started preliminary testing on the 64-bit cluster at Pacific Northwest National Laboratory.

Automation and Robustness. We are also working on running GADU under the Chimera [20] system. Much of bioinformatics data is *derived* from other data by the application of computational procedures (such as Blast and Pfam). The explicit representation of these procedures can enable documentation of data provenance, discovery of available methods, and on-demand data generation (so-called virtual data). The Chimera virtual data system (VDS) combines a virtual data catalog, for representing data derivation procedures and derived data, and a virtual data language interpreter that translates user requests into data definitions and query operations on the database. The Chimera system is coupled with distributed data Grid services to enable on-demand execution of computation schedules constructed from database queries. The Chimera VDS provides a catalog that can be used by application environments to describe a set of application programs ("transformations") and then track all the data files produced by executing those applications ("derivations"). Chimera contains a mechanism to locate the "recipe" to produce a given logical file, in the form of an abstract program execution graph. These abstract graphs are then turned into an executable DAG for the Condor DAGman metascheduler by the Pegasus planner which is bundled into the VDS code release. Preliminary testing of the Chimera system was conducted on the Argonne's Data Grid cluster, and the testing of Blast under Condor was done at University of Wisconsin-Madison.

Expansion of Bioinformatics Tools and Services. Numerous excellent bioinformatics tools and algorithms have been developed in the past years. We plan to augment the GADU analysis module by adding a number of publicly available tools for sequence genome analysis: tools for domain and motif analysis (e.g., InterPro, Psort, and TMHMM), sequence analysis (e.g., FastA and ClustalW), and tools for analysis of structural information.

User Interface. We are working on further development of the GADU Web-based user interface, so it will allow users to personalize their GADU workspace. Users will be able to upload and store genome sequences, process them via an user-selected array of tools, and obtain the results of the analyses in various formats (e.g., XML, relational tables, and raw format).

We expect to have a first release of the GADU public server with a TeraGrid backend in May 2003.

Acknowledgments

Special thanks to the following individuals who contributed constructive advice and support: Ian Foster, Miron Livny, Zach Miller, Mike Wilde, Jens Voeckler, Mike Milligan, and Von Welch. This project was funded by NSF ACI-9619019.

References

1. Ideker, T., Galitski, T., Hood, L. (2001) A new approach to decoding life: Systems biology. *Annu. Rev. Genomics Hum. Genet.*, **2**, 343-372.
2. Mervis, J. (2001) ADVANCED COMPUTING: NSF Launches TeraGrid for Academic Research. *Science*, **293**, 1235-1237.
3. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389-3402.
4. Pearson, W. R. (1994) Using the FASTA program to search protein and DNA sequence databases. *Methods Mol Biol.*, **24**, 307-331.
5. Shpaer, E. G., Robinson, M., Yee, D., Candlin, J. D., Mines, R., Hunkapiller, T. (1996) Sensitivity and selectivity in protein similarity searches: A comparison of Smith-Waterman in hardware to BLAST and FASTA. *Genomics*, **38**, 179-191.
6. Mulder, N. J., Apweiler, R., Attwood, T. K., Bairoch, A., Barrell, D., Bateman, A., Binns, D., Biswas, M., Bradley, P., Bork, P., et al. (2003) The InterPro Database, 2003 brings increased coverage and new features. *Nucleic Acids Res.*, **31**, 315-318.
7. Bateman, A., Birney, E., Cerruti, L., Durbin, R., Eddy, S. R., Griffiths-Jones, S., Howe, K. L., Marshall, M., Sonnhammer, E. L. (2002) The Pfam protein families database. *Nucleic Acids Res.*, **30**, 276-280.
8. Henikoff, S., Henikoff, J. G., Pietrokovski, S. (1999) Blocks+: A non-redundant database of protein alignment blocks derived from multiple compilations. *Bioinformatics*, **15**, 471-479.
9. Pearl, F. M., Bennett, C. F., Bray, J. E., Harrison, A. P., Martin, N., Shepherd, A., Sillitoe, I., Thornton, J., Orengo, C. A. (2003) The CATH database: An extended protein family resource for structural and functional genomics. *Nucleic Acids Res.*, **31**, 452-455.
10. Lo Conte, L., Brenner, S. E., Hubbard, T. J., Chothia, C., Murzin, A. G. (2002) SCOP database in 2002: Refinements accommodate structural genomics. *Nucleic Acids Res.*, **30**, 264-267.
11. Overbeek, R., Larsen, N., Pusch, G. D., D'Souza, M., Selkov, E. Jr., Kyrpides, N., Fonstein, M., Maltsev, N., Selkov, E. (2000) WIT: Integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Nucleic Acids Res.*, **28**, 123-125.
12. Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., Wheeler, D. L. (2003) GenBank. *Nucleic Acids Res.*, **31**, 23-27.
13. Bairoch, A., Apweiler, R. (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res.*, **28**, 45-48.
14. Sussman, J. L., Lin, D., Jiang, J., Manning, N. O., Prilusky, J., Ritter, O., Abola, E. E. (1998) Protein Data Bank (PDB): Database of three-dimensional structural information of biological macromolecules. *Acta. Crystallogr. D. Biol. Crystallogr.*, **54**, 1078-1084.
15. Stoesser, G., Baker, W., Van Den Broek, A., Garcia-Pastor, M., Kanz, C., Kulikova, T., Leinonen, R., Lin, Q., Lombard, V., Lopez, R., Mancuso, R., Nardone, F., Stoehr, P., Tuli, M.A., Tzouvara, K., Vaughan, R. (2003) The EMBL nucleotide sequence database: Major new developments. *Nucleic Acids Res.*, **31**, 17-22.
16. Krogh, A., Larsson, B., von Heijne, G., and Sonnhammer, E. L. L. (2001) Predicting transmembrane protein topology with a hidden Markov model: Application to complete genomes. *J. Mol. Bio.y*, **305**, 567-580.
17. Nakai, K., Horton, P. (1999) PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localization. *Trends Biochem. Sci.*, **24**, 34-

- 36.
18. Litzkow, M., Livny, M., Mutka, M. (1998) Condor: A hunter of idle workstations. In *Proceedings of the 8th International Conference on Distributed Computing Systems*. IEEE Computer Society Press, pp. 108-111.
 19. Foster, I., Kesselman, C., (1997) Globus: A metacomputing infrastructure toolkit. *Intl. J. Supercomputer Applications*, **11**, 115-128.
 20. Foster, I., Voeckler, J., Wilde, M., Zhao, Y. (2002) Chimera: A virtual data system for representing, querying and automating data derivation. In *Proceedings of the 14th Conference on Scientific and Statistical Database Management*, Edinburgh, Scotland.

Figure Legends

Figure 1. The data flow on GADU. The system consists of three separate modules. The first, *data acquisition module*, contains a number of programs that periodically search for data specified by user in public databases and notify the user via email if new data of interest have been found and downloaded locally. The second, *data analysis module*, submits the acquired data to specified bioinformatics tools and algorithms for analysis. Because of the need to process massive amounts by this module, we are developing the GADU computational backend based on scalable TeraGrid technology. Finally, once the outputs from the algorithms and tools have been successfully acquired, they are parsed by the third, *data storage module*, and deposited to a local relational database (Oracle, in our case) for user navigation or for future use by additional algorithms and visualization programs.

Figure 2. Acquisition module flow.

Figure 3. Acquisition module user interface.

Figure 4. Small file containing information required for automated submission of the genome to the tools.

Figure 5. Addition of computational resources to the GADU pipeline can substantially increase its performance.

Table

Table 1. Total Time for Completion of One Prokaryotic Genome (4,000 sequences) in GADU. Average time listed is based on 18-node usage.

Data acquisition	1 minute
Data analysis	4 hours
Data storage	1 hour
Total Time	5 hours

Figures

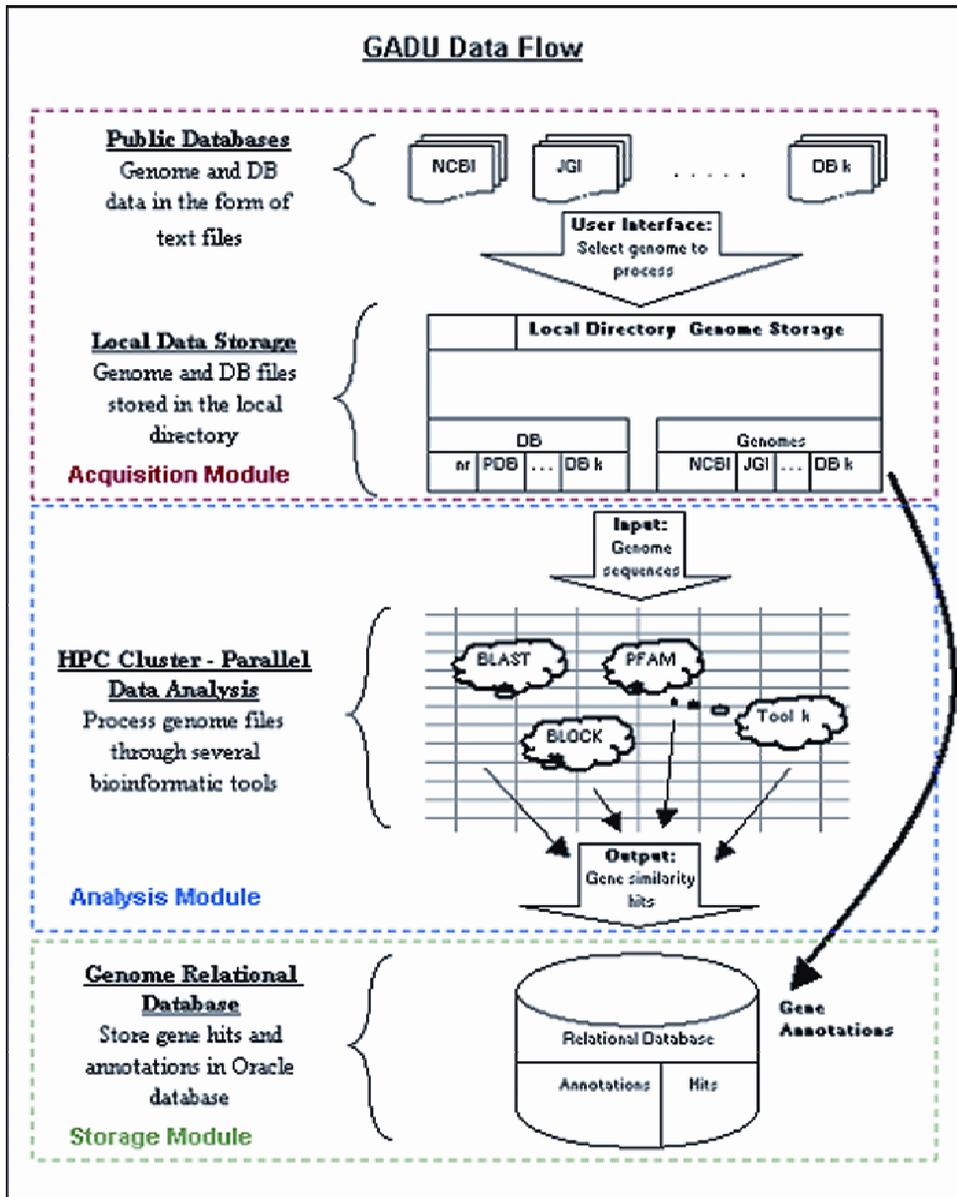


Figure 1. The data flow on GADU.

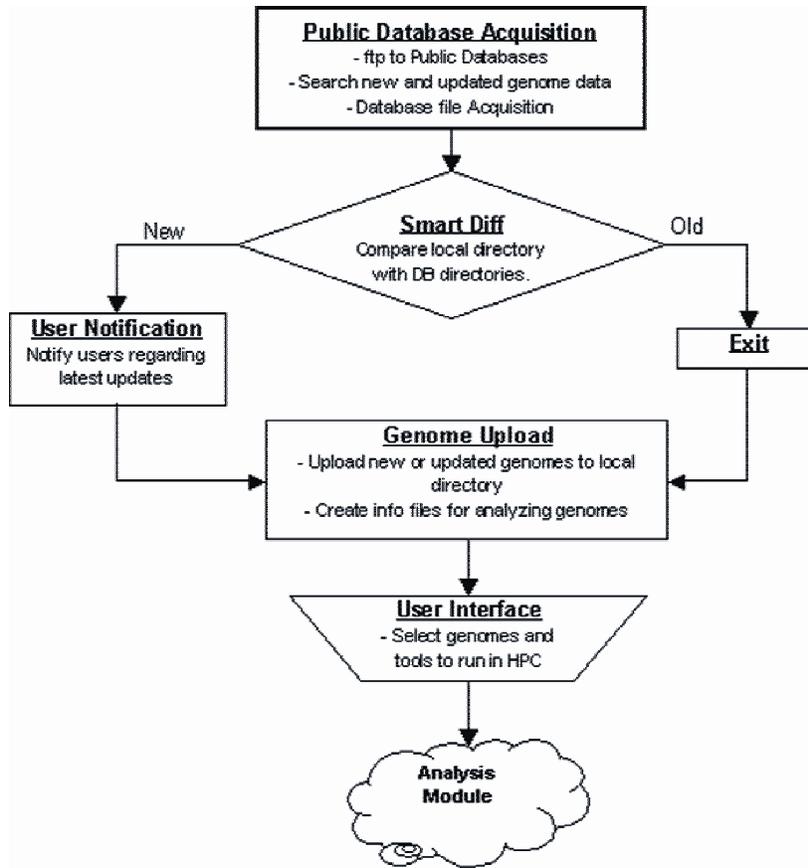


Figure 2. Acquisition module flow.

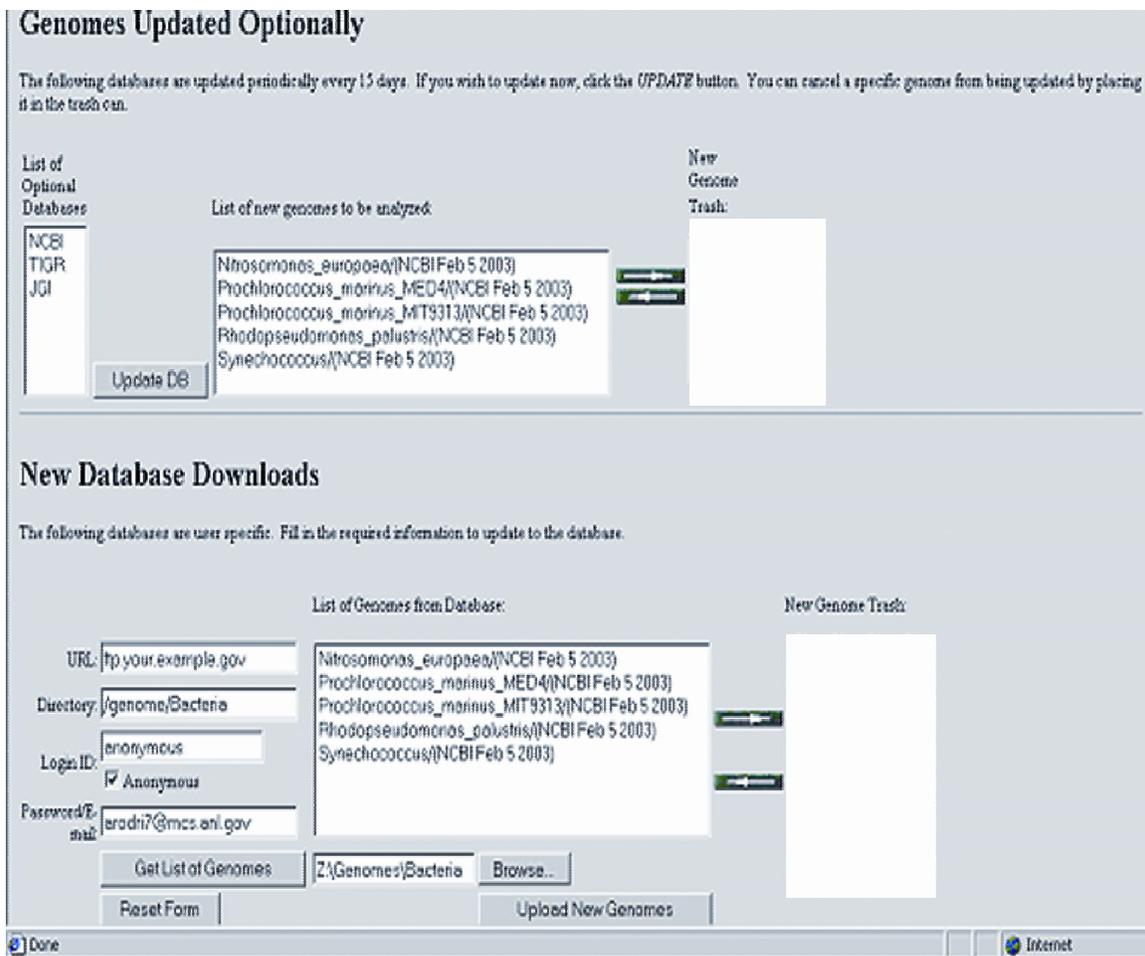


Figure 3. Acquisition module user interface.

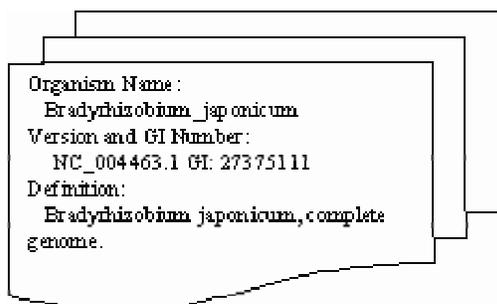


Figure 4. Small file containing information required for automated submission of the genome to the tools.

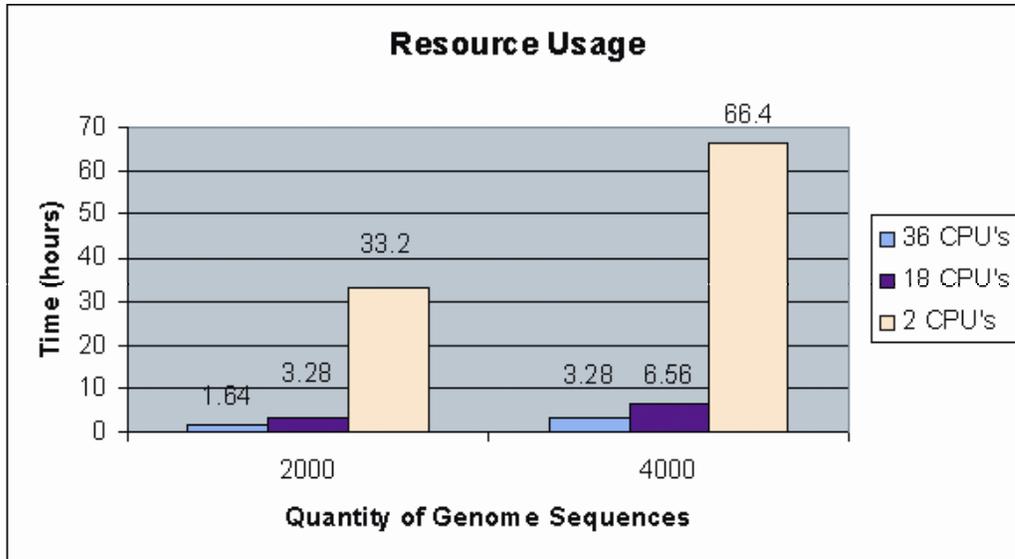


Figure 5. Addition of computational resources to the GADU pipeline can substantially increase its performance.