

A Flexible Grid Framework for Automatic Protein-Ligand Docking

§ David Abramson, † Celine Amoreira, † Kim K. Baldridge,

‡ Laura Berstis, ‡ Chris Kondrick, § Tom Peachey

davida@csse.monash.edu.au, kimb@oci.unizh.ch

§ CSIT,
Monash University,
Clayton, 3800, Victoria,
Australia

† Faculty of Chemistry,
University of Zurich,
Irchel,
Zurich, Switzerland

‡ San Diego Supercomputing
Center,
9500 Gilman Drive, MC 0505
La Jolla, CA 92093-0505, USA

Abstract

Many important and fundamental questions in biology and biochemistry can be better understood through investigations performed at the protein-ligand or drug-receptor level. A variety of techniques have been used over the years, and it is an area of active research. In this paper we illustrate an approach that leverages a number of different computational chemistry approaches, and combines these with non-linear optimization algorithms and grid based high performance computing platforms. The result is a very flexible, high performance method of evaluating protein-ligand interaction algorithms. We illustrate the approach by evaluating a hybrid molecular modeling and quantum theoretical based algorithm.

1. Introduction

Many important and fundamental questions in biology and biochemistry can be better understood through investigations performed at the protein-ligand or drug-receptor level. Experimental research questions involving macromolecular biological complexes are typically directed at understanding the correlation between molecular and structural specialization and functional variations. However, typically, it becomes very difficult to determine the exact mechanism of action at the molecular level without the aid of computational investigations at the atomic level. Additionally, estimating correct three-dimensional atomic structures of complexes between proteins and ligands is an important component of the drug-design process in the pharmaceutical industry. Basic aspects of ligand-protein interactions, categorized under the general term molecular recognition, are concerned with the specificity as well

as the stability of the ligand to bind the binding site. To this end, molecular recognition is central to the development of active substances that could be used as drugs, or designing ligands that can undergo specialized function.

Molecular recognition can be studied by looking at the properties and mechanistic of binding between the protein and a ligand, in the biological environment. Investigations involved in the positioning of the ligand into the macromolecule can be performed very specifically in the case that the researcher is interested in understanding the variance of binding energy as a ligand moves in very specific ways (e.g., around specified dihedral angles), or, they may be interested in probing all degrees of freedom of motion of the ligand in the protein pocket. There has been considerable effort in studies of protein-ligand interactions and many tools and software have been developed. However, much still remains to be done as no single generalized tool has provided a truly reliable and accurate method for understanding detailed ligand-protein interactions. Computational models that perform these calculations typically have significant performance demands, and thus high performance parallel and distributed computers can be desirable.

Rather than equip a single tool to perform all of the functions, we favour a framework in which key tools can be combined. In the area of protein-ligand docking, one benefits from separating the process of computing molecular interaction from a chemical viewpoint, from the non-linear optimization that is performed in order to minimize the binding energy. Moreover, utilization of Grid-based distributed supercomputers can be difficult, and the computational strategies that are required can make software that is already complex even more so.

Our approach for this type of problem involves the coupling of a number of separate codes, each focused on a particular aspect of the total problem, using Grid-oriented strategies. In our case, the chemistry is performed via a combination of quantum chemistry and classical molecular mechanics methods, the non-linear optimization is performed using the package called Nimrod/O [1][3], and Grid based high performance computing is performed with a specialized tool called Nimrod/G [2][4].

In this paper we discuss the approach we have used. Section 2 discusses classical approaches to modeling the protein-ligand docking process. Section 3 describes a new experimental technique that we have been evaluating. Section 4 introduces a framework that leverages the Nimrod family of software tools. In Section 5 we illustrate the results of a case study.

2. Classical approaches to protein-ligand docking

Due to the impact on human health (drug design), huge efforts are devoted to research in areas involving details of protein-ligand interaction, including the development of reliable software tools. The primary objective in ligand-protein interaction investigations is the prediction of the optimal orientation and conformation of a small molecule (e.g., ligand, drug prototype, etc) in a pocket embedded in a protein. Such investigations can be carried out using a variety of different methods, depending on the questions being asked.

Computational methods for the study of ligand-protein interactions vary considerably in their algorithmic structure and resource requirements, as shown in Figure 1. Quantum mechanical (QM) based methods that operate at the electronic structure level are relatively limiting in terms of molecular size and computational time, but provide high level of accuracy. Classical-based methods use less accurate strategies that neglect the electronic motion, but only require modest compute time. Molecular dynamics and Monte Carlo, both molecular mechanics (MM) based approaches, involve long simulation times but are effective within the classical limits. Molecular docking procedures, widely used for study of ligand-protein interactions, place a ligand in a receptor using local shape features and electrostatic criteria based on a number of different models.

Ab initio QM is the holy grail for highly accurate results for molecular systems and is based on algorithms that solve the Schrödinger equation.

Unfortunately, most problems are too large to be considered fully by QM methods alone. Therefore, approximations that consider less electronic structure detail are incorporated, often emphasizing molecular energy as a function of the nuclear positions only.

The MM-based methods are fully empirical in nature, and are specifically tuned for classes of molecules such as proteins, enzymes, or other macromolecular systems. In MM, molecules are considered collections of masses held together by classical forces. The equations based on classical mechanics and parameters defining the energy surface of a molecule are collectively referred to as force field. The contributions to the molecular energy include, bond stretching, angle bending dihedral deformations, van der Waals, electrostatic interactions, etc. An important issue for force field applications is the feasibility for transferability of the model to a wider range of macromolecules. The MM method is largely applied for basic understanding of low energy conformations on the Potential Energy Surface (PES) through molecular structure refinement, molecular dynamics (MD) simulation, Monte Carlo simulations and/or ligand docking simulation.

Molecular Dynamics, based on the empirical force field and basic assumptions described for MM, aims to reproduce the time-dependent motion of a molecule by solving the Newton's second law for all the atomic degrees of freedom. New positions and velocities of the atoms are predicted and saved as a trajectory file. Performing this process for a range of time steps with additional controls on temperature gives the description of the behavior of the molecular motion as it traverses over the PES. In contrast to QM and MM, both of which attempt to reach the minimum in the energy function, molecular dynamics is able to overcome conformational barriers with the input of temperature. The ability for Molecular Dynamics to search conformational space enables the placement of families of structure on the PES, which can be further minimized for prediction of local stationary points.

Current molecular docking tools attempt to estimate the optimal three-dimensional configurations between a protein and a ligand using a variety of classical means, for example, by invoking an empirical force field function. Typically, docking searches involve a large number of ligands and associated conformations, together with a cost or scoring function, which governs the degree of compatibility between the ligand and the protein in the formation of a reaction complex.

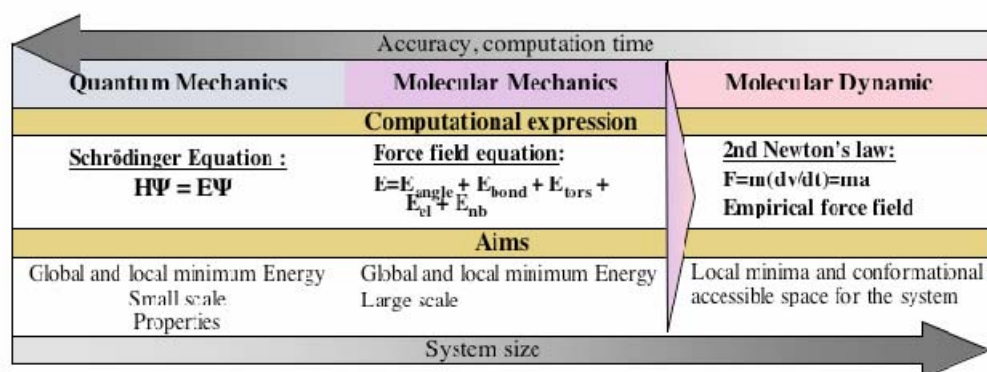


Figure 1: Computational methods for study of protein-ligand interactions

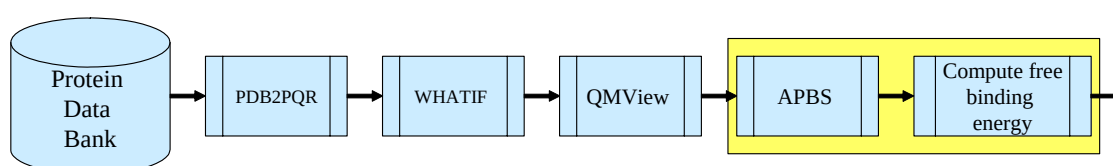


Figure 2: Hybrid protein-ligand docking pipeline

3. A hybrid docking algorithm

Recently, we proposed a new hybrid algorithm that uses both QM and classical methods, and this forms the basis for the experiments we report later in the paper. The overall technique could equally be applied to any of the methods discussed in the previous section. Typically, investigations begin with a structure from the Protein Data Bank (PDB), [9] which provides experimental structures of ligands, proteins, or enzymes. These crystal structures are without hydrogen atoms, which must be added using tools such as Babel or PDB2PQR [7]. In the case that there is a water network attached to the protein, it might be necessary to remove the network, with tools such as WHATIF [8]. After this phase of preparation, a checking point is needed to verify the protonation state of the ligand, and optimal positioning of the hydrogen atoms.

After the initial structures are prepared, the next step is to consider the possible positioning of the ligand within the protein pocket. The researcher may be interested in the variance of binding energy as a ligand moves around only specified degrees of freedom, or, they may be interested in probing all degrees of freedom of motion of the ligand in the pocket. We have a tool, QMView, [5], that we have enhanced to enable the generation of ligand conformations resulting from any particular metric, for example rotation around a torsional angle, and

subsequent placement into the protein by aligning the new conformation with a part of the ligand that remains fixed. A second checking point is then required to delete the non-stable conformations due to unnatural bumping between atoms.

In the various steps, the electrostatic effects can be investigated using a wide variety of available tools. As opposed to using classical force fields for this, we focus on a package that solves the Poisson-Boltzmann equation (PBE) numerically (e.g., APBS [12]). This method describes the protein and ligand as collections of charges and radii within a continuous solution of specified dielectric, salt concentration and temperature. The atomic charges and radii for the ligand, and optionally various levels of structure, are determined using more sophisticated QM methodologies such as the CHARGE method [6], using the ab initio molecular QM program GAMESS [13]. Additionally, QM methods can provide more accurate structural information if need be as well as an estimate of internal energy assessment. The protein charges and radii must also be specified for the electrostatics computation, and are obtained for example from the PARSE [14] parameterization, as they are well defined for estimating the contribution of simple functional groups to solvation.

After solving the PBE, APBS computes the binding free energy, which is used as an objective function value to drive the conformational search. All of the tools discussed to date can be combined in a variety of

ways, ranging from simple shell or Perl scripts, through to complete workflows managed by one of a number of workflow tools, such as Kepler [10][11].

The output of this pipeline, as shown in Figure 2, is the binding free energy for a *particular* protein-ligand configuration. In order to actually minimize the binding free energy, we need to adjust the configuration of the ligand within the protein binding site. In several docking codes, this non-linear optimization phase is embedded in the chemistry code itself, for example, AutoDock [15]. While this is powerful, it is not possible to experiment with different search procedures or different types of chemical calculations without modifying a significant amount of the software package. In this work, just as we are using several different chemistry packages to compute the binding energy, can also combine these with a state of the art nonlinear optimization package to perform searches and other parameterization procedures, the topic of discussion in the next section.

4. A docking framework

As discussed in section 1, our strategy involves separation of three computational aspects of the molecular docking algorithm, namely

- the computational chemistry that computes the binding energy;
- the optimization of the ligand position within the protein and
- the use of high performance computational platforms.

Figure 3 shows a framework that supports this approach using a range of software tools. At the core is the particular protein-ligand docking algorithm, such as the one discussed in the last section. The other two components of the framework are served by two members of our Nimrod tool family, namely Nimrod/O [1][3] and Nimrod/G [2][4].

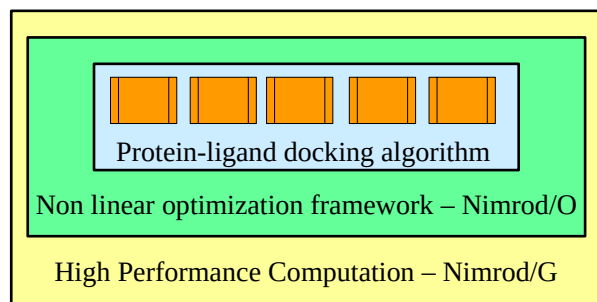


Figure 3 – Docking Framework

Nimrod/O is a generic tool that performs distributed non-linear optimization. This tool can be applied to any problem in which an objective cost function is computed by a computational model, and thus, is an ideal tool to use in performing protein-ligand docking as described in Section 3. Nimrod/O is optimized for computer models that are computationally expensive. Numerical optimization using these models typically needs multiple iterations of the search heuristic, each requiring several runs of the model.

The objective landscape produced by computational models frequently shows multiple local optima, and the binding energy surfaces explored for protein-ligand interactions are no exception. In addition, discretization errors introduced by these models may introduce many more local optima as artifacts. The local optimum found by a search algorithm will depend on the algorithm used and often on the starting point chosen for the search. Accordingly, Nimrod/O allows multiple concurrent searches from a range of starting points. The tool also allows many of the different optimization methods discussed above to be executed concurrently. If all optimizations converge to the same optimum, then the user can be confident that this is a global optimum. If all optimizations do not converge to the same point, then the range of local optima is an indication of the roughness of the landscape and serves to guide further searches.

parameter x float range from -2 to 2; parameter y float range from -4 to 4; parameter z float range from -4 to 4; parameter xR float range from -3.15 to 3.15; parameter yR float range from -3.15 to 3.15; parameter zR float range from -3.15 to 3.15;
task main node:execute transrad \$x \$y \$z \$xR \$yR \$zR copy root:skeleton node:. node:substitute skeleton apbs.input node:execute apbs apbs.input > apbs.output node:execute extract apbs.output > output.\$jobname copy node:output.\$jobname root:. endtask
method simplex starts 10 named "simplex" starting points random tolerance 0.0 endstarts endmethod
method bfgs starts 10 named "bfgs" starting points random tolerance 0.001 endstarts endmethod

Figure 4: A Nimrod/O schedule file for protein-ligand docking.

In general, a Nimrod/O experiment is performed by preparing a declarative text file called a “schedule”. Figure 4 shows a schedule for performing protein-ligand docking by minimizing the binding energy function, using a number of the packages discussed in Section 3.

The task of minimizing the binding energy as a function of ligand position in the protein pocket is a search within a six dimensional parameter space. Accordingly, six parameters (x , y , z , xR , yR , zR) indicating the position and rotation in three dimensional space are defined in the schedule using the “parameter” statements. The “task” section specifies the operations required to perform one instance of the computational model and produce a numerical output that will be the objective function for the optimization. First, the program “transrad” translates and rotates the ligand, aborting if a collision occurs between the protein and the ligand. Then a “skeleton” file is copied from the root node to the remote computational node where execution will occur. The subsequent “substitute” command edits the skeleton file, replacing the parameters by their current values, producing a file “apbs.input”. Then the program “apbs” acts on the input file producing a file “apbs.output”. This output file is finally filtered to produce the numerical value for the binding energy that is copied back to the root node. The final section of the schedule file specifies the optimization method to be used. In this example, two well-known algorithms are invoked: the downhill simplex and the BFGS gradient descent method. Ten starts are performed with each, giving twenty distinct optimizations.

To improve the performance of the system, batches of jobs are run in parallel using Grid resources. To do this, Nimrod/O uses parallel versions of standard search methods, for example, gradient descent (BFGS), simplex, simulated annealing and evolutionary techniques such as genetic algorithms. The tool also uses a range of computational resources including uni-processor workstations, parallel clusters and distributed super-computers. Nimrod/O does this by leveraging a related tool, Nimrod/G to perform the distribution of computational tasks over a Grid of resources [2][4].

Nimrod/G is used to explore the behaviour of complex systems, modelled by existing computational software packages, as the input conditions change, enabling users to design better procedures, understand complex processes, and/or optimise system behaviour. Nimrod/G incorporates a distributed scheduling

component that manages the scheduling of individual experiments to a wide range of computational resources, including workstations, clusters and high-end supercomputers. The tool sits on top of conventional schedulers like PBS [16] and Condor [17], as well as Grid middleware such as Globus [18]. Together, these features assure that even complex parametric experiments can be defined and run with little programmer effort. In many cases it is possible to establish a new experiment in minutes.

5. A case study

In this section, we show the results of a number of computational experiments as outlined in the previous section for a specific protein and ligand system, for the evaluation of the binding free energy using the APBS software. Three non-linear optimization techniques were selected: simplex, gradient descent (BFGS) and Simulated Annealing (SA). A total of 20 Simplex, 20 BFGS and 10 different SA configurations were executed to explore the behaviour of the different algorithms.

5.1. Using a local workstation & local cluster

As discussed in the last section, Nimrod/G is flexible enough to perform the computations on anything from a local workstation, a cluster, or a set of distributed clusters. In this section we report the performance of the experiment on a local workstation and a small cluster. In the first case, the optimization runs are executed sequentially on a uni-processor workstation. In the second case, each optimization algorithms perform a number of functional evaluations in parallel using a cluster. However, the runs are then executed sequentially since there are insufficient resources in the cluster to run them all in parallel.

The results of these simulations are summarized in Table 1. We note that, not all of the simulations were able to complete and find a local optimum. For example, of the 20 Simplex runs, 10 failed to complete because the energy calculation failed for part of the search space. Only 5 BFGS and 5 SA simulations completed. This is not uncommon in the use of computational models that may not be defined in all regions of the search space. In spite of this, a total of 20 different starting points were evaluated, yielding a range of solutions, as summarized in Table 1. “Run#” identifies the particular run. “Optimum” reports the final binding energy. In these experiments, only the APBS electrostatic computation is evaluated. “Iters” shows how many iterations of the heuristic were

performed, whereas “Evals” shows how many times the binding energy was computed. “Batches” reports the number of parallel batches of evaluations, which is a good indication of the variability in wall clock time since this indicates the critical path in the search. “Cluster Time” shows the effect of running each batch in parallel, and is the sum of the batch times for each optimization. “Workstation Execution Time” reports the time that the jobs took when executed sequentially on a uni-processor workstation.

The best result was obtained from one of the SA runs (SA Run 1), and the worst result from one of the Simplex runs (Simplex Run 1), however, there is a wide variability and no single heuristic performed uniformly poorly. Moreover, each of the runs took a different number of iterations of the search heuristics, which in turn caused a varying number of objective function evaluations and parallel batches. It is worth observing that, while SA produced the best result, this method also required the longest time to complete.

5.1. Using the grid

To demonstrate the ability to run the experiments on the Grid, all these optimizations were repeated in parallel. This time, Nimrod/G was configured with a small Grid testbed, and shown in Table 2. On all these resources, the experiment was competing for CPUs with other users, and thus even though a given resource may have had many processors, we were not able to utilize all of them at the same time. Figure 5 shows how the number CPUs used varied as the experiment progressed. As optimizations complete, or fail, the resource requirements attenuate.

The results of this case study indicate that the overall approach is viable – clearly it is possible to couple the many different packages that we used to successfully explore different protein-ligand bindings. By using this Grid, we were able to run the experiment some 33 times faster than the workstation, and about 9 times faster than the small cluster.

6. Conclusions

In this paper we have reviewed a range of techniques that can be used to explore the dynamics of protein-ligand binding, and have evaluated a particular approach involving a hybrid between classical molecular dynamics and quantum theoretic approaches.

A major problem in docking is the determination of a realistic energy cost function. This function is usually assumed to consist of an electrostatic energy

contribution, a nonelectrostatic energy and entropic contribution, and an internal energy part. The electrostatic binding energy between a protein and a ligand can be relatively accurately computed by dielectric continuum solvation methods based on the Poisson-Boltzmann equation. Since in contrast to the protein, the charge distribution on the ligand is usually unknown, quantum chemical methods can be used. This provides more accurate structure, charge information, and estimation of internal energy. The non-electrostatic and entropic contributions are fit to experimental results and included. Future goals involve exploration of more complex techniques to produce more accurate models of the protein-ligand interaction in a general procedure.

The approach presented here uses a range of different software packages for study of ligand/protein interactions, enabling prediction of binding energy for a ligand docking into a binding site. Further, because we separated the non-linear optimization component from the chemistry evaluation component, we are able to utilize a Grid enabled package to perform the latter task. Overall, this generates considerable flexibility and will allow us to explore a range of modeling techniques in the future.

Acknowledgements

The projects are supported by a variety of funding agencies, including the Australian Research Council, the US National Science Foundation, the Swiss National Science Foundation, and PRAGMA.

References

- [1] Abramson, D, Lewis A, Peachey T, Fletcher, C., “An Automatic Design Optimization Tool and its Application to Computational Fluid Dynamics”, SuperComputing 2001, Denver, Nov 2001.
- [2] Abramson, D., Sosic R., Giddy J. and Hall B., “Nimrod: A Tool for Performing Parametrised Simulations using Distributed Workstations”, The 4th IEEE Symposium on High Performance Distributed Computing, Virginia, August 1995.
- [3] Abramson, D, Lewis, A. and Peachy, T., “Nimrod/O: A Tool for Automatic Design Optimization”, The 4th International Conference on Algorithms & Architectures for Parallel Processing (ICA3PP 2000), Hong Kong, 11 - 13 December 2000.
- [4] Abramson, D., Giddy, J. and Kotler, L. “High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?”, International Parallel and Distributed Processing Symposium (IPDPS), Cancun, Mexico, May 2000, pp 520- 528.
- [5] Baldridge, K.K.; Greenberg, J.P. QMView: “A

Computational 3-D Visualization Tool at the Interface Between Molecules and Man". *J. Mol. Graphics* 1995, 13, pp 63-66.

[6] Breneman, C.M., Wiberg K.B., "Determining atom-centered monopoles from molecular electrostatic potentials", *J. Comp. Chem.*, 1990, 11, pp 361-373.

[7] Dolinsky, TJ, Nielsen JE, McCammon JA, Baker NA. "PDB2PQR: an automated pipeline for the setup, execution, and analysis of Poisson-Boltzmann electrostatics calculations". *Nucleic Acids Research*, 32 W665-W667 2004, <http://pdb2pqr.sourceforge.net/>

[8] Vriend, G., "WHAT IF: A molecular modeling and drug design program", *J. Mol. Graph* 8, 1990, pp 52-56, <http://swift.cmbi.kun.nl/whatif/>

[9] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne, "The Protein Data Bank", *Nucleic Acids Research*, 28 pp. 235-242 (2000), <http://www.rcsb.org/pdb/>

[10] <http://kepler-project.org>

[11] Ludäscher, B. Altintas, I. Berkley, C. Higgins, D. Jaeger-Frank, E. Jones, M. Lee, E. Tao J. and Zhao Y., "Scientific Workflow Management and the Kepler System",

Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows, 2005.

[12] M. Holst and F. Saied, "Multigrid solution of the Poisson-Boltzmann equation", *J. Comp. Chem.* 14, 1993, pp 105-113, <http://apbs.sourceforge.net/>

[13] Schmidt, M.W., Baldridge, K.K., Boatz, J.A., Elbert, S.T., Gordon, M.S., Jensen, J.H., Koseki, S., Matsunaga, N., Nguyen, K.A., Su, S., Windus, T.L., "The General Atomic and Molecular Electronic Structure System" *J. Comp. Chem.* 14, 1993, pp 1347-1363.

[14] Sitkoff, D.; Sharp, K.A.; Honig, B. "Calculation of Hydration Free Energies Using Macroscopic Solvent Models", *J. Phys. Chem*, 98, 1994, pp 1978-1988.

[15] Morris, G. M., Goodsell, D. S., Halliday, R.S., Huey, R., Hart, W. E., Belew, R. K. and Olson, A. J. "Automated Docking Using a Lamarckian Genetic Algorithm and and Empirical Binding Free Energy Function", *J. Comp. Chem* 19, pp 1639-1662.

[16] <http://www.openpbs.org/>

[17] <http://www.cs.wisc.edu/condor/>

[18] <http://www.globus.org/>

Algorithm	Run#	Optimum	Iters	Evals	Batches	Cluster Execution Time (mins)	Workstation Execution Time (mins)
Simplex	1	80.8	27	114	29	1044	2089
Simplex	2	No result		7	1	2	14
Simplex	3	No result		7	1	2	14
Simplex	4	73	21	90	22	663	1729
Simplex	5	No result		7	1	2	14
Simplex	6	68.2	63	277	68	2364	6148
Simplex	7	No result		7	1	54	225
Simplex	8	68.4	65	265	67	3023	6301
Simplex	9	69.6	67	262	72	3077	6428
Simplex	10	No result		7	1	56	236
Simplex	11	No result		7	1	56	237
Simplex	12	69.3	67	259	72	2402	6265
Simplex	13	70.8	43	139	43	1052	2836
Simplex	14	No result		7	1	59	251
Simplex	15	No result		7	1	59	253
Simplex	16	70.3	67	244	69	2488	5856
Simplex	17	No result		7	1	60	259
Simplex	18	72.7	46	207	50	2179	5202
Simplex	19	No result		7	1	60	267
Simplex	20	70.5	41	146	43	1397	3441
BFGS	1	78.3	3	62	11	400	2202
BFGS	2	No result		1	1	2	2
BFGS	3	No result		1	1	2	2
BFGS	4	No result		13	2	259	1109
BFGS	5	No result		1	1	3	3
BFGS	6	No result		13	2	539	1662
BFGS	7	No result		1	1	1	1
BFGS	8	No result		13	2	537	2088
BFGS	9	75.1	3	55	9	733	3435
BFGS	10	No result		1	1	1	1
BFGS	11	No result		1	1	1	1
BFGS	12	70.9	7	173	28	968	4885

BFGS	13	73.3	7	161	26	1314	6508
BFGS	14	No result		1	1	1	1
BFGS	15	No result		1	1	1	1
BFGS	16	No result		1	1	2	2
BFGS	17	No result		1	1	2	2
BFGS	18	69.6	4	108	19	502	2818
BFGS	19	No result		1	1	2	2
BFGS	20	No result		13	2	1550	8846
SA	1	66.5	16	815	160	3705	12746
SA	2	No result		226	12	225	3874
SA	3	No result		221	12	228	3874
SA	4	72.5	3	338	40	1000	5699
SA	5	No result		218	12	228	3873
SA	6	67	14	725	142	3282	11227
SA	7	No result		225	12	228	3878
SA	8	68.5	4	341	50	1181	6325
SA	9	69	8	495	84	1992	8655
SA	10	No result		223	12	228	3878
Best		66.5	3	1	1	1	1
Worst		80.8	67	815	160	3705	12746
Ave		71.2	29	130	24	784	2913
SD		3.6	26	180	36	1035	3230
Total (mins)						39216	145665
Speedup on Grid						9	33

Table 1 –Experimental results

Name	Institution	Processor Speed (MIPS)	No. CPUs
mahar.infotech.monash.edu.au	Monash Univ., Australia	5977	50
tgc.trecc.org	NCSA, USA	3985	13
brecca-2.vpac.org	VPAC, Australia	5571	180
pragma001.grid.sinica.edu.tw	ASGC, Taiwan	4771	11

Table 2 –Computational resources

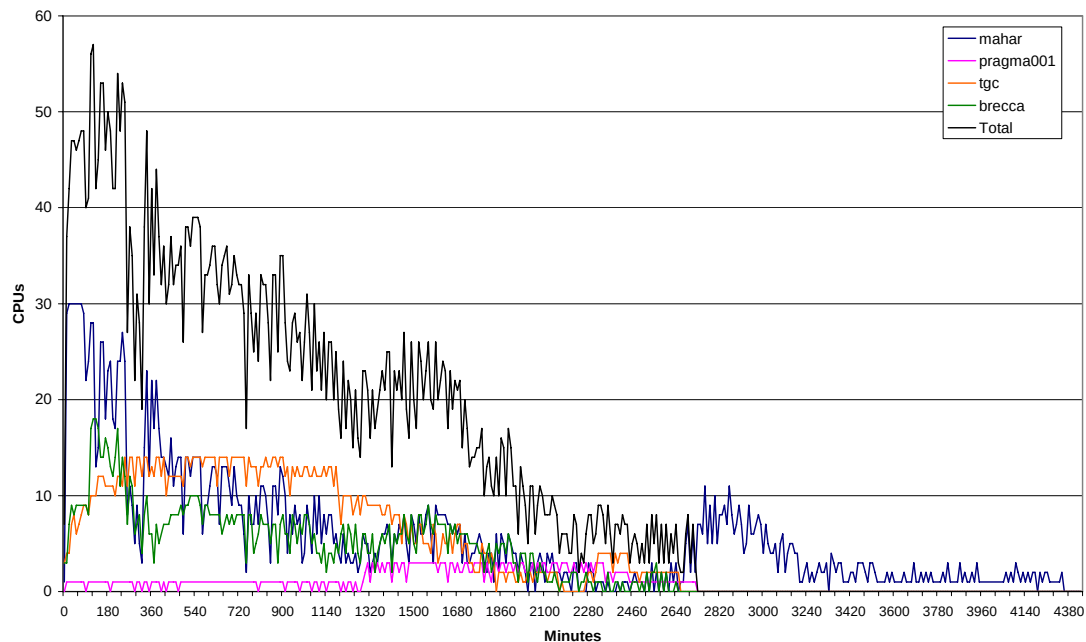


Figure 5 – Progress of jobs on grid testbed