Grid-enabled SIMAP Utility: Motivation, Integration Technology and Performance Results

Jun Wang, Xuan Liu, Navonil Mustafee, Qian Gao, Simon J E Taylor, David Gilbert
School of Information Systems, Computing and Mathematics
Brunel University, Uxbridge, Middx, UB8 3PH (*firstname.lastname@brunel.ac.uk)*

1. Introduction

A biological system comprises large numbers of functionally diverse and frequently multifunctional sets of elements that interact selectively and nonlinearly to produce coherent behaviours. Such a system can be anything from an intracellular biological process (such as a biochemical reaction cycle, gene regulatory network or signal transduction pathway) to a cell, tissue, entire organism, or even an ecological web. Biochemical systems are responsible for processing environmental signals, inducing the appropriate cellular responses and sequence of internal events. However, such systems are not fully or even poorly understood. Systems biology is a scientific field that is concerned with the systematic study of biological and biochemical systems in terms of complex interactions rather than their individual molecular components. At the core of systems biology is computational modelling (also called mathematical modelling), which is the process of constructing and simulating an abstract model of a biological system for subsequent analysis. This methodology can be used to test hypotheses via *in-silico* experiments, providing predictions that can be tested by *in-vitro* and *in-vivo* studies. For example, the ERbB1-4 receptor tyrosine kinases (RTKs) and the signalling pathways they activate, govern most core cellular processes such as cell division, motility and survival (Citri and Yarden, 2006) and are strongly linked to cancer when they malfunction due to mutations etc. An ODE (ordinary differential equation)-based mass action ErbB model has been constructed and analysed by Chen et al. (2009) in order to depict what roles of each protein plays and ascertain to how sets of proteins coordinate with each other to perform distinct physiological functions. The model comprises 499 species (molecules), 201 parameters and 828 reactions. These *in-silico* experiments can often be computationally very expensive, e.g. when multiple biochemical factors are being considered or a variety of complex networks are being simulated simultaneously. Due to the size and complexity of the models and the requirement to perform comprehensive experiments it is often necessary to use high-performance computing (HPC) to keep the experimental time within tractable bounds. Based on this as part of an EC funded cancer research project, we have developed the SIMAP Utility that allows the SImulation modeling of the MAP kinase pathway (http://www.simap-project.org). In this paper we present experiences with Grid-enabling SIMAP using Condor.

2. IT architecture of the Grid-enabled SIMAP Utility

2.1 Utility Overview

The SIMAP Utility is a platform-independent environment for modelling biochemical networks, and also for simulating and analysing the dynamic behaviour of biochemical models. It has been developed using Java technology and can be run on many platforms that support the JRE. The Netbeans Platform has been chosen for the software development. It has a state-of-the-art modular Swing application framework which includes a windows system, actions, etc. Moreover, it is easy to plug-in other modules and allows easy installation of software updates for separate modules without a need to reinstall the whole package. The SIMAP Utility consists of several plug-in modules which include the *Database Management module, Data Viewer module, Local Simulator module, Model Analysis modules, Grid Access Point*. Additional modules are planned and third party development is actively encouraged.

2.2 Why Grid?

One of the most time-consuming activities in model analysis and construction of this kind is parameter scanning. Parameter scanning permits the exploration of a model's behaviour over different ranges of parameter values. The SIMAP Utility performs parameter scan simulations whereby the same simulation code is executed using different parameter values and each execution of the simulation (subsequently referred to as a job) contributes to a point in the resultant graph. The SIMAP Utility requires non-trivial amounts of computation time to increase the precision of the generated graphs. We followed up the research conducted by Liu et al. (2008) in which a Grid-enabled Biochemical Networks Simulation Environment - BioNessieG has been used to execute large-scale parameter scan on different HPC cluster resources such as National Grid Service (NGS-www.ngs.ac.uk) and

ScotGrid (www.scotgrid.ac.uk). For the current work, because the users of the SIMAP Utility could have some confidential biochemical data and models which are not suitable to be executed and analysed on any shared public HPC resources, we therefore decided to trial the studies using a network of non-dedicated PCs that were installed in our computer labs with Condor-based desktop Grid computing technologies.

2.3 Our approach

Unlike *cluster-based grid computing* that has traditionally been geared towards dedicated, centralized, high-performance clusters running on UNIX flavour operating systems, *desktop-based grid computing* refers to the aggregation of non-dedicated, de-centralised, commodity PCs connected through a network and running (mostly) the Microsoft Windows operating system (Mustafee and Taylor, 2009). Grid-enabling the SIMAP Utility required, first and foremost, the creation of a desktop grid computing infrastructure. This required the installation of a distributed computing middleware that could effectively harness the spare computation cycles available on the PCs. We decided to install Condor because of its large deployment base, its relative ease of use and because it is available free of cost. Condor is an opportunistic job scheduling system that is designed to maximize the utilization of workstations through identification of idle resources and scheduling background jobs on them (Litzkow et al., 1988). It also exploits multiple cores transparently. A collection of such workstations is referred to as a Condor pool.

Condor architecture defines resource providers and resource consumers. The resource providers make their resources available to Condor for the processing of jobs that originate from the resource consumers. The jobs to be processed may have dependencies with regard to the operating system and the physical machines on which the job is to be processed, the memory and disk space required, the available software libraries that are needed and so forth. On the other hand, the resource providers may have certain conditions (e.g. only Java jobs can be run) and preferences (e.g. jobs originating from resource consumer "x" is given priority) based on which access to their resource is granted. Condor allows resource consumers and resource providers to advertise these requirements, conditions and preferences by providing a language called *classified advertisements* (*ClassAds*) that provide a flexible and expressive framework for matching jobs originating from the former with resource offers from the latter (Thain et al., 2004). The *ClassAds* are scanned by a Condor *matchmaker agent* running on only one computer in a Condor Pool, to find a match between the requirements advertised by the resource consumers and the resources advertised by the resource providers. Once a match has been found by the matchmaker agent, it notifies both the resource consumer and the resource providers. Upon receiving this notification, the resource consumer claims the resource advertised by the resource provider through a claiming protocol. The job is executed by the resource provider and the results of the computation are returned back to the resource consumer. Condor allows end-users to *submit* jobs and to *query* job status using two alternative mechanisms, (a) through use of a submit description file, and (b) through use of programming APIs that are exposed by Condor as Web Services (Chapman et al., 2005). The latter approach can integrate Condor capabilities into existing software, and this is the approach used by us to integrate the SIMAP Utility with Condor (through a Java-based job manager utilising Condor Web Services).

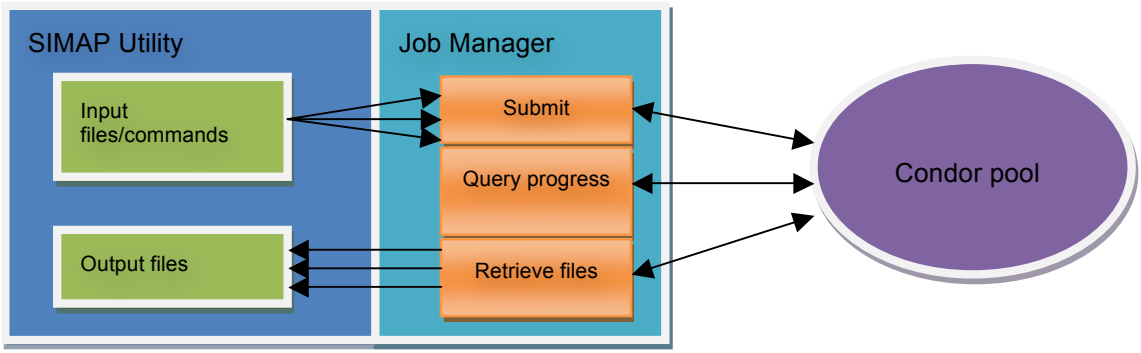

Figure 1: Integration of SIMAP Utility with Condor, through a Java-based Job Manager

As shown in Figure 1, the SIMAP Utility integrates with the Condor pool using our Java-based Job Manager. The Job Manager invokes the Web Services exposed by Condor to *submit* jobs, *query* job progress and to *retrieve* results. For each simulation job, the Utility generates the required input file with a set of scanning parameters. The Utility then wraps the job command (shared by all jobs of a simulation) and the corresponding input files into a job request. Upon submission of job requests, the Utility queries job status, and on completion

of jobs it retrieves the corresponding output files. The experiments we conducted using our grid-enabled SIMAP Utility and the results are discussed next.

3. Results

We used a Condor pool with 32 machines in Brunel University as the test bed. Each machine is configured with dual 2.1GHz cores and 2 Gigabyte memory. All the machines are connected to the network at 100Mbps. The test unit is the computational model of the *mammalian ErbB signaling pathway* (Chen et al., 2009) with 21 seconds' computation time (25 minutes' simulated time) and input/output files of about 1 Megabyte each. We selected a range of cores (4-32) to run jobs (32-4096). The time matrix is shown on Table 1. The row stands for the number of jobs per simulation, and the column stands for sequential run or the number of dual core machines. From the test results, we can see our Grid enabled SIMAP Utility can achieve high throughput on interactive jobs with a speedup to about 12 using 16 machines (32 cores).

	sequential run	**2**	**4**	**8**	**12**	**16**
32	672s	401s	220s	138s	103s	103s
64	1344s	859s	490s	245s	175s	154s
128	2688s	1708s	811s	498s	288s	234s
256	89.6m	44.18m	22.65m	14.18m	7.83m	7.33m
512	179.2m	79.95m	40.38m	23.27m	14.6m	13.03m
1024	5.97h	2.67h	1.2h	0.7h	0.52h	0.48h
2048	11.95h	5.32h	2.67h	1.41h	0.94h	0.96h
4096	23.89h	10.5h	5.33h	2.81h	1.93h	1.9h

Table 1: Running time when using different number of jobs and different numbers of 2-core machines

4. Conclusions

This SIMAP project uses mathematical modelling techniques and information technologies to simulate and analyse biochemical models. We have presented the Grid-enabled SIMAP Utility which can perform large-scale parameter scans for biochemical analysis. Our lightweight condor-based approaches can use existing campus Microsoft Windows desktops to support up to thousands of interactive simulation jobs. This approach has achieved an acceptable speedup performance. However, it has been the ease of deployment and integration with SIMAP Utility that has made these uses of Condor extremely interesting. The next stages of this research will be to profile SIMAP models to determine minimum model size against expected speedup and to further develop the job submission system for biologists.

5. References:

Chapman, C., Goonatilake, C., Emmerich, W., Farrellee, M., Tannenbaum, T., Livny, M., Calleja, M. and Dove, M. (2005). Condor Birdbath-Web Service interfaces to Condor. In *Proceedings of the UK e-Science All Hands Meeting 2005*, pp.737-744. Available online: *http://archive.niees.ac.uk/documents/ AHM_Birdbath_2005.pdf*. Last accessed on 9 July, 2009.

Chen, W.W., Schoeberl, B., Jasper, P.J., Niepel, M., Nielsen, U.B., Lauffenburger, D.A. and Sorger, P.K. (2009). Input–output behavior of ErbB signaling pathways as revealed by a mass action model trained against dynamic data. *Molecular Systems Biology*, **5**:239. Available Online: *http://www.nature.com/msb /journal/v5/n1/pdf/msb200874.pdf*. Last accessed on 9 July, 2009.

Citri, A., Yarden Y. (2006) EGF-ERBB signalling: towards the systems level. Nat Rev. 1, 505-516.

Litzkow, M., Livny, M. and Mutka, M. (1988). Condor - a hunter of idle workstations. In *Proceedings of the 8th International Conference of Distributed Computing Systems*, pp.104-111. IEEE Computer Society, Washington, DC, USA.

Liu, X, Jiang, J., Ajayi, O., Gu, X., Gilbert, D., Sinnott R., (2008), 'BioNessie(G) - A Grid Enabled Biochemical Networks Simulation Environment'. *Studies in Health Technology and Informatic*s, IOS Press, 2008, **138**: 147-157.

Mustafee, N. and Taylor, S.J.E. (2009). Speeding Up Simulation Applications Using WinGrid. *Concurrency and Computation: Practice and Experience,* **21**(11): 1504-1523.

Thain, D., Tannenbaum, T. and Livny, M. (2004). Distributed computing in practice: the Condor experience. *Concurrency and Computation: Practice and Experience*, **17**(2–4): 323–356.