PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A

Phil. Trans. R. Soc. A (2009) **367**, 2705–2716
doi:10.1098/rsta.2009.0046

A computational Grid framework for immunological applications

By Mark D. Halling-Brown, David S. Moss, Clare E. Sansom and Adrian J. Shepherd*

Institute of Structural and Molecular Biology, School of Crystallography, Birkbeck College, Malet Street, London WC1E 7HX, UK

We have developed a computational Grid that enables us to exploit through a single interface a range of local, national and international resources. It insulates the user as far as possible from issues concerning administrative boundaries, passwords and different operating system features. This work has been undertaken as part of the European Union ImmunoGrid project whose aim is to develop simulations of the immune system at the molecular, cellular and organ levels. The ImmunoGrid consortium has members with computational resources on both sides of the Atlantic. By making extensive use of existing Grid middleware, our Grid has enabled us to exploit consortium and publicly available computers in a unified way, notwithstanding the diverse local software and administrative environments. We took 40 000 polypeptide sequences from 4000 avian and mammalian influenza strains and used a neural network for class I T-cell epitope prediction tools for 120 class I alleles and haplotypes to generate over 14 million high-quality protein–peptide binding predictions that we are mapping onto the three-dimensional structures of the proteins. By contrast, the Grid is also being used for developing new methods for class T-cell epitope predictions, where we have running batches of 120 molecular dynamics free-energy calculations.

Keywords: ImmunoGrid; computational Grid; Grid middleware; T-cell epitope prediction

1. Introduction

ImmunoGrid (http://www.immunogrid.eu) is an ambitious systems biology project, funded by the European Union Framework 6 as a STREP, which aims to combine simulations of the human immune system at many levels. Many of its characteristics are shared by other large projects in bioinformatics and systems biology: the involvement of many international partners each with their own computational resources; the need to run large numbers of computations, both large and small; and the need to provide an easy-to-use interface for a user base that is likely to include 'wet' molecular biologists, students or even clinicians. We present here a case study that can show the relevance and effectiveness of a Grid solution to a wide range of biological, and perhaps other scientific, applications.

* Author for correspondence (a.shepherd@mail.cryst.bbk.ac.uk).

One contribution of 16 to a Theme Issue 'Crossing boundaries: computational science, e-Science and global e-Infrastructure II. Selected papers from the UK e-Science All Hands Meeting 2008'.

Many past attempts to exploit Grid resources for scientific computation have been disappointing. This was mainly attributable to the difficulties experienced in setting up and maintaining Grid middlewares and the complexities of the job submission and monitoring processes associated with them. For example, we have had negative experiences with the installation of Globus and Uniform Interface to Computing Resources (UNICORE). The need to install such middlewares on new resources is something we are now able to avoid. While providing mechanisms for accessing existing middlewares, our approach provides alternative methods for adding new resources. Significant developments that facilitate our approach have been made in recent years, notably through the development of lightweight Grid 'upper middlewares' (see §2 below) that insulate users from the extremely complex underlying access technologies. We present here a framework that should allow researchers to construct computational Grids that are easy to modify and develop, and easy for scientists who are not computer specialists to use.

The ImmunoGrid partners required a system that would enable the most complex single simulations that access a large cluster or supercomputer to be run; that would allow large numbers of less complex calculations and simulations, such as epitope predictions, to be carried out in parallel (e.g. to simulate the effects of a given clinical scenario on multiple individuals); and to support smaller scale simulations for demonstration, education or training. No single partner within the consortium could guarantee access to sufficient resources to meet these requirements, so a Grid-based solution is a practical necessity. From experience, we felt it vital to provide a variety of different sized resources in order to allow for the fast processing of small jobs on light resources as well as large resources for the huge batch jobs. We anticipated that a massive number of small jobs would need to be run because of the educational aspects of the project. Queue times on larger resources would typically make the deployment of these small jobs intolerably slow in proportion to the execution time. Multiple resources also provide redundancy in the case of failures.

From the perspective of the end-users (who will not necessarily be programmers), the system was expected to provide transparent access to the underlying resources, so that individual users would not need to be aware of their organization, or even, necessarily, know where the computers used by a given job were located. Users should be insulated from issues related to systems administration and passwords, and from differences between operating systems. This is best achieved via an easy-to-use graphical interface, with the ubiquitous Web interface particularly appealing, especially as this ensures that end-users do not need to install client software on their local machines.

To fulfil these requirements, we should maximize the range and number of computational resources that can be added to our Grid. These should range from local desktop workstations to the existing national and international Grid services. A partial list of these could include the UK National Grid Service (NGS; http://www.grid-support.ac.uk/), the European supercomputer Grid Distributed European Infrastructure for Supercomputing Applications (DEISA; http://www.deisa.org), PI2S2 Grid and the US TeraGrid (http://www.teragrid.org).

This need to access a diverse range of resources has two practical implications. First, we must aim to support all major existing Grid middleware and platforms. Second, we must make it as easy as possible to add new Grid nodes and

resources, so that any individuals and organizations working in relevant areas that have computational resources suitable for incorporation into our Grid are not deterred from doing so simply because of their complexity.

We have, therefore, sought to reuse existing Grid solutions wherever possible. We use two pieces of upper middleware, the application hosting environment (the AHE; Zasada *et al.* 2006; Coveney *et al.* 2007) and DESHL (DEISA Services for the Heterogeneous management Layer; Sloan *et al.* 2006). Both these tools provide a higher level simplified access to Globus and UNICORE, respectively; however, neither approach provides access to the entire range of Grid resources. AHE provides access to Globus- and UNICORE-enabled resources and AHE-enabled local resources. DESHL provides access to UNICORE-enabled resources. Additionally, we use gLite middleware to access PI2S2, the Sicilian Grid, in collaboration with our collaborators from the University of Catania. None of these tools, alone, will give access to a full range of resources; contrary to sensible expectations, there is no pan-European standard or interoperability. For example, we cannot access resources at CINECA using AHE due to the local administrative policies preventing the installation of the AHE services. Also, DESHL does not provide mechanisms to access local computing resources. However, if used together, they can allow access to the maximum range of resources while shielding users from most of the complexity associated with Globus (http://www.globus.org), UNICORE, gLite and other underlying Grid middleware. For the purposes of the ImmunoGrid project, it was desirable that no installation and management of user-unfriendly middleware would be necessary. Our experience is that AHE is the easiest and most flexible middleware to use in practice.

Our framework also allows the usage of resources directly using a Web Services approach. This provides an application programming interface (API) allowing users to integrate a remotely hosted service with other applications or components of applications. This approach is becoming increasingly popular in major bioinformatics centres such as the European Bioinformatics Institute (http://www.ebi.ac.uk). For ImmunoGrid, instances of our simulators can be wrapped as Web Services, deployed on local machines and accessed via the Grid framework described here. This provides a quick and simple technique to speedily deploy a piece of software through the framework.

By default, specific resources within the ImmunoGrid framework are selected automatically by a simple job broker, but this can be done manually if the user requires. The Grid has been designed to allow for the possibility that different users will have the right to run jobs on different subsets of available resources. For example, a user may have a Grid certificate giving him or her access to a national or international Grid service such as the NGS, DEISA or TeraGrid. The system will select resources that each user has the correct permissions to use transparently, without needing the user to provide further authentication.

Our Grid framework is presented to users via a straightforward Web-based interface, which hides the underlying middleware and resources from the user. The system will automatically select which of the diverse and widely distributed computational resources available within the Grid is most suitable for a particular job, based on the nature of the job and the user's permission status, and allocate the job accordingly to nodes within one or more clusters.

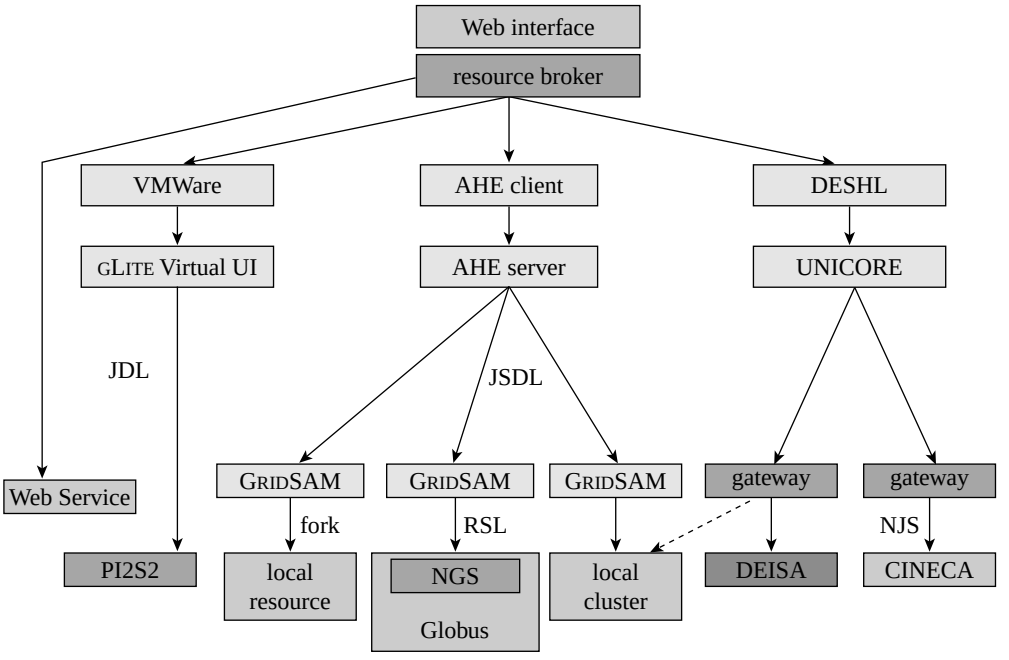

Figure 1. The diagram shows the layered approach adopted in the ImmunoGrid framework and described in the text. The abbreviations used are defined as follows: JMEA, Job Management Enterprise Application; JSDL, Job Submission Description Language; RSL, Resource Specification Language; NJS, Network Job Scheduler; PI2S2, Sicilian Grid; NGS, National Grid Service; DEISA, European Grid; CINECA, Northern Italy Computer Centre (Bologna).

2. Methodology

(*a*) *Overview of Grid architecture*

Here, we describe the functionality of the Grid infrastructure that has been developed and used by the ImmunoGrid project, with particular emphasis on its flexibility, ease of deployment and ease of use. Hence, we focus on the upper layers of our infrastructure and the provision of a generic interface to DESHL, the AHE, gLite and Web Services. The effort required to integrate these solutions within a single, coherent infrastructure is comparatively modest and most of the tasks can be undertaken using the documentation provided. A specialist user will be necessary only when customization of the interface or alterations to the resource broker or other aspects of the Grid infrastructure are required.

A schematic overview of our Grid infrastructure is presented in figure 1. The top level comprises a single Web-based interface coupled to a resource broker/job launcher. The broker/launcher accesses DESHL and the AHE via middleware-specific scripts, gLite via VMWare and Web Services via standard Simple Object Access Protocol protocols. The main benefit of this approach is that it allows us to launch jobs simultaneously on different national/international Grid services, on local computational resources and as Web Services through a single, unified interface. It also allows us to effectively hide the complexity and diversity of the

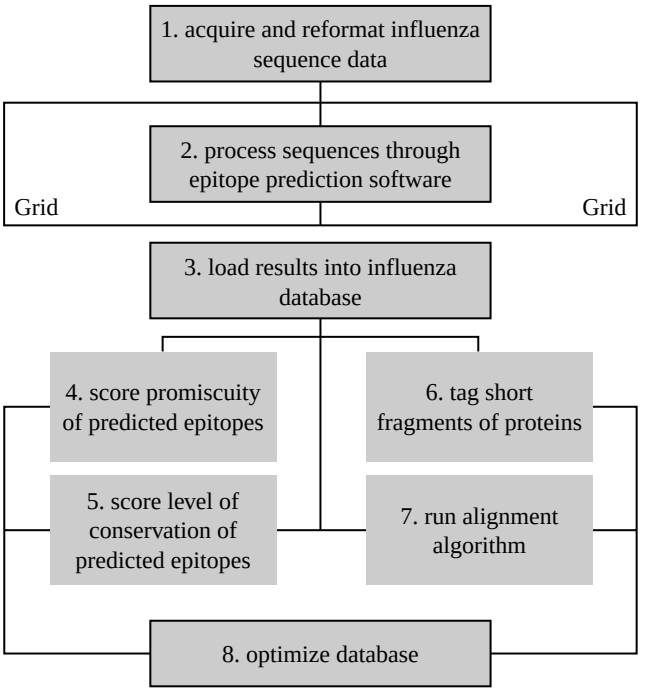

Figure 2. The flow diagram for the analysis of the polypeptide sequences obtained from mammalian and avian influenza isolates. The Grid framework described in the text is used for running neural network prediction tools to assess the likelihood of peptide sequences binding to human MHC molecules. This is a key step in generating an immune response. This process is almost fully automated.

underlying middleware and resources from the end-user. An example of how the Grid framework has been used within the context of a specific application is shown in figure 2.

(*b*) *Upper middleware*

Arguably the most important ingredient in our framework is the role played by upper middleware (AHE and DESHL), as it hides much of the complexity of the Grid both from the Grid user and from those developing a new Grid infrastructure. Without upper middleware, the whole enterprise would be prohibitively complicated and time-consuming for most scientific institutions or consortiums to undertake. AHE and DESHL play a key role in the deployment of software to different computational resources and in the management of jobs.

AHE and DESHL both provide a command-line interface via which a job can be launched, its progress monitored and its output (both intermediate and final) retrieved. There are, however, some important differences.

The AHE is a lightweight environment designed to run unmodified applications on diverse, distributed Grid resources. An explicit design goal of the AHE is to hide the underlying complexity (of the 'lower' Grid middleware, of the host environment and of how executables are set-up) from the end-user. Currently, this is achieved using GridSAM (http://gridsam.sourceforge.net/2.0.1;

Grid Job Submission and Monitoring Web Service), but a (UNICORE (Almond & Snelling 1999)) plug-in has recently been developed and has been used by the Coveney group (Zasada *et al.* 2007). The UNICORE plug-in is not available as part of the AHE package. After the initial deployment of AHE, a simple Job Submission Description Language file must be produced for each combination of software and resource that the AHE will have access to. This is the only manual intervention required, and it needs to be done only once for each software/resource combination. Thereafter, the AHE provides a list of available resources upon which the software has been installed. Jobs can then be run by simply selecting resources from the list, as detailed below; there is no need to access any resources directly. The AHE is easy to install as part of the OMII (http://www.omii.ac.uk) stack.

Although DESHL is somewhat less flexible than the AHE, it does provide essential mechanisms for accessing European supercomputers via UNICORE. In the context of the ImmunoGrid project, such resources are available via DEISA and at CINECA (a member of the ImmunoGrid consortium). Setting up access to a supercomputer via DESHL is somewhat less transparent than adding a resource using the AHE, as scripts need to be written that manage access to a named user space.

With some customization and coding, a similar mechanism was constructed for the gLite submission process. This was undertaken as proof of concept, given the involvement of a Sicilian partner on the ImmunoGrid project, and was more complicated to set-up than AHE or DESHL. Access to the gLite PI2S2 Grid services is made using a virtual user interface (VUI). The virtual machine is run by VMWare application installed on the Web server. The authentication step from the job launcher to the VUI is made by Secure Shell without password using public key authentication with an agent. The credentials to use the Grid services are obtained by pk12 GILDA CA certificates. The integration between the job launcher and PI2S2 Grid is made by a pool of scripts that are able to initialize the VOMS proxy for the user, create the environment needed to submit a job, wait for the 'Done' status of the job and retrieve the output of the submitted job.

(*c*) *Accessing local resources*

There are two ways that a local resource can be accessed via our system—using the AHE or as a Web Service. The fundamental difference between these two approaches is that a given resource is made available for any application using the AHE route, whereas the Web Services approach makes a specific application available. The practical differences between these two approaches are relatively minor.

In order to access a new local resource using the AHE, an Apache Tomcat (http://www.tomcat.apache.org) server needs to be installed on the local machine together with an instance of GridSAM (an open-source job submission and monitoring Web Service). These are automatically installed and configured (without recourse to special administrative rights) when the OMII stack is installed on the machine. The final step is simply to edit the configuration of the GridSAM instance, so that it points to the locally installed software that we wish to use.

Providing access to a local resource using Web Services is slightly more complicated. It can be achieved either by 'wrapping' the software in a simple Web Service shell or by pointing a Web Service execution at the local software. In either case, this involves writing some code, such as a Web Service Definition Language file. An application server or Web server is required to host the Web Service. Both of these approaches to the incorporation of local resources into a Grid are documented in detail on the portal website (http://igrid-ext.cryst.bbk.ac.uk/portal_guide).

(*d*) *Security*

UNICORE, AHE and gLite handle security and authorization using X.509 digital certificates (http://www.faqs.org/rfcs/rfc2510.html). This largely enables us to manage the security of our Grid using a single uniform approach. Users who have their own Grid certificates for accessing NGS and DEISA resources are able to upload them, thereby gaining access to those resources. The same certificate can be used to access NGS and DEISA. However, we anticipate that the majority of end-users will not have their own certificates. To allow such users access to local Grid resources, a self-signed certificate can be generated for the portal or each user. This self-signed certificate will provide access to local Grid resources only. For the purposes of the ImmunoGrid project, we have acquired portal certificates that are effectively group passes, preventing the need for users to acquire their own certificates.

Both UNICORE and AHE (together with most X.509 certificate authorized (CA) middlewares) require that the certificate is presented on a MyProxy server (Novotny *et al.* 2001; this includes the self-signed certificates). This ensures that the certificate's password needs to be entered only once during the submission process. To enable certificates to be deployed in this manner, the Web interface to our system has a link to the Java Web Start (http://java.Sun.com/products/javawebstart), Java Network Launching Protocol (http://java.Sun.com/products/javawebstart) and MyProxy Upload Tool. Although this is not a fully automated solution (as it requires the user to manually enter the location of the certificate as well as enter the password), it is currently the most reliable. The self-signed certificates are handled in exactly the same manner as the CA certificates so provide the same security features.

(*e*) *Resource brokering and job launching*

Currently, our infrastructure uses a simple PHP script to handle resource brokering and job launching. Jobs are allocated to resources according to whether the user has a Grid certificate, and by taking into account the anticipated job length compared with any limits imposed by specific resources. For example, when the job involves running the ImmunoGrid simulator, specific settings within the simulator configuration file (such as the maximum number of iterations it will run for, and the length of the bit string used to represent molecular interactions) are used to estimate job length. From the list of resources deemed to be both available and appropriate for running a given job, specific resources are allocated at random. Alternatively, the end-user may select which of the available resources he/she wishes to use for running a given batch of jobs.

Although this approach is sufficient for our current requirements, a more sophisticated resource broker that seeks to optimize total execution time and ensure fairness within the context of agreed policies on resource usage will be appropriate for many applications. A wide range of approaches are possible (Kertész & Kacsuk 2006). Unfortunately, deploying an existing Grid resource broker within our framework is currently problematic, as each broker supports only a subset of possible middlewares. However, this situation may change in the future, given the present interest in Grid resource meta-brokering (Kertész *et al.* 2007).

Jobs are launched by a simple job launcher. This executes the appropriate launch command for a given job and resource (this is different for AHE, DESHL, gLite and Web Services). The launcher also records the details of the job both on the server's file system and in a local database, and executes the appropriate command-line script corresponding to the resource that is selected. The state of the job is stored in the local database along with any information required to uniquely identify that job. This allows the appropriate scripts to be called when the user requests the job state to be refreshed.

(*f*) *Web interface*

The Web interface to our infrastructure provides the end-user with simple mechanisms for uploading, launching and monitoring the progress of jobs, as well as for retrieving and displaying results. The interface comprises PHP Web pages, with AJAX and DHTML used to give them a modern look and feel. This interface is available from http://www.immunogrid.eu.

3. Results

The Grid infrastructure described here has been used to run a number of different applications within the ImmunoGrid project. The two case studies presented here are both in the area of molecular immunology. The first concerns the large-scale prediction of class I T-cell epitopes in influenza virus isolates for use in a database, using local installations of the prediction software developed at the Center for Biological Sequence Analysis (CBS), Technical University of Denmark (DTU), Copenhagen, Denmark (Nielsen *et al.* 2003). The second is a novel method for class II T-cell epitope prediction that uses molecular dynamics simulations rather than sequences and empirical binding data.

All timings are measured in wall clock time, as this represents the most relevant measurement for the end-user. Both these applications demonstrated the amount of time that could be saved by using a Grid-based approach in handling large-scale applications. The first is an example of an application that involves multiple instances—up to millions—of similar, simple calculations, with a single class I T-cell epitope prediction taking only a fraction of a second. Conversely, the second is an example of a very complex, if easily parallelizable, application, requiring wall clock times of approximately 2 days for a single molecular dynamics simulation. Access for these case studies was restricted to resources in London (UK) via AHE, Bologna (Italy) via DESHL and Boston (USA) via AHE. This was to evaluate the potential benefits of our novel approach to combining local resources at multiple sites.

(*a*) *Class I T-cell epitope prediction in influenza*

There is a resurgence of interest in research into the influenza virus following the emergence of the highly pathogenic H5N1 strain of avian influenza as a severe potential threat to human health. The only drugs available against influenza, neuraminidase inhibitors, are only partially effective: vaccination is by far the most effective prophylactic treatment for this disease, but current vaccines are developed from inactivated or attenuated live virus, which can itself be dangerous, and must be reformulated every season in response to antigenic drift. The development of subunit vaccines from antigenic fragments of influenza virus proteins (epitopes) could be simpler and safer, particularly if these peptides are also conserved between many different viral strains. Relatively few such epitopes are yet to be characterized experimentally (Bui *et al.* 2007). However, there is a wealth of influenza virus sequence data available that can be mined for predicted epitopes; this *in silico* approach has gained popularity for vaccine design in recent years. In this case study, we concentrate on the prediction of peptides that bind to MHC class I and II molecules; such binding is necessary, although not significant, for a T-cell-based immune response.

Over 40 000 influenza protein sequences extracted from the public NCBI databases (Miotto *et al.* 2008) were collated and analysed using the CBS T-cell epitope prediction software. This provided predictions for a total of 137 class I and II MHC alleles, giving a total of over 5 400 000 jobs. We estimated (from timings for a subset of 86 552 jobs) that running the entire batch on a single server at Birkbeck would take approximately 155 hours. Using the Grid infrastructure described here, the total number of jobs was split equally between three resources: the Birkbeck local cluster, the Dana–Farber local cluster and the CINECA supercomputer. In this case, splitting jobs over these three resources were performed by hand. This task can also be performed automatically by the resource broker, as long as an appropriate schedule has been implemented. Subsets of the total number of nodes were used at each resource in order to comply with fair usage guidelines. In the case of the AHE-enabled resources, the prediction tool software was bundled with the Grid job. For local administration reasons, the tools were preinstalled on the CINCA cluster.

This distribution of jobs proved highly successful, with a saving of approximately 6 days (over 90%) in wall clock time compared with the time anticipated to be used by a single machine. One important caveat, however, is that the jobs sent to the CINECA supercomputer were not held in a queue for a significant period of time. This could not have been guaranteed, and other batches of jobs submitted to CINECA have not been so fortunate. Queue length can be determined only by directly logging onto the supercomputer at run time.

(*b*) *Use of molecular dynamics in T-cell epitope prediction*

Sequence-based prediction of T-cell epitopes depends on empirical binding data, and as much more data are available for class I than for class II epitopes, prediction methods for class I binding are more accurate than those for class II binding (Lundegaard *et al.* 2007). Using molecular dynamics simulations to predict the binding energy of a peptide–MHC complex is an alternative method that does not rely on empirical data, but is very much more computationally

intensive; it will be feasible only by using a Grid (Davies *et al.* 2003). Simulation requires an experimentally determined crystal structure, or, more controversially, a homology model of the required MHC allele.

In this case study, we simulated the binding of peptides to class I MHC proteins using NAnoscale Molecular Dynamics (NAMD; Phillips *et al.* 2005) using the adaptive biasing force option (Hénin & Chipot 2004). We studied the diffusion of 120 peptides out of the binding groove of the MHC protein for experimentally determined strong, weak and non-binders. We estimated (from timings for a single job) that running a batch of 120 jobs on the Birkbeck server would take approximately 5760 hours. For this case study, we sent half of the jobs to the Birkbeck cluster, and the other half to the supercomputer at CINECA. The appropriate software (NAMD) was already available on both resources. We achieved a substantial saving in wall clock time compared with the anticipated time on a single machine, this time of approximately 87 per cent, equivalent to over 200 days CPU time. In this case, however, the majority of the time used at CINECA was not CPU time, but queuing time.

4. Discussion and conclusion

A large and increasing number of research projects in the biological sciences—and others—would benefit from having access to large-scale computational resources such as Grids. This need is a characteristic particularly of projects in systems biology, which typically need to run complex simulations of biological processes at many levels and rely on large- and high-throughput datasets. These resources may be available locally, in the form of a supercomputer or, more usually, a large CPU cluster, or be made available on a national or international Grid service following a successful application for access time. However, these resources will not always be adequate. In any case, it is risky for a large and complex project to rely on a single source for computational resources, as these will be vulnerable to down time and maintenance periods. Long-term access to production-quality Grids is difficult for academic users in particular, and lengthy queuing times are commonplace.

We present here a lightweight Grid framework that can allow research workers who are unfamiliar with the mechanics of Grid applications to access a wide range of computational resources, located in several countries and using different infrastructures. We have demonstrated that our framework will allow access to a wide range of such resources, from local workstations to supercomputers and national Grids. The case studies described here, together with the development of an immune system simulator (Pappalardo *et al.* 2008), are illustrations of its successful use.

This framework has been shown to be flexible and easy to install and use, and, as the case studies here illustrate, it can generate useful scientific results within the framework of a large systems biology project. Such an approach can be seen as particularly appropriate for use by consortia of research groups working on large-scale projects, as it allows those groups to connect their own resources to the Grid easily and at low cost. However, it is essential to mention some caveats. Our Grid has been designed to be a research tool, rather than a production Grid. It is not yet feasible to develop such a Grid quickly, both if it is to have the

flexibility to enable it to incorporate a wide range of infrastructures and if it is to be of production quality. Production-quality Grids require certain features: constant monitoring of services, sophisticated error-handling methods and dedicated user support (Richards *et al.* 2004). These limitations, however, are also found to some extent in national Grids. No off-the-shelf solution is available, and the long development time still limits the construction of dedicated Grids for any but large and long-running projects. We have deliberately designed our system to be modular, so that we will be able to use a suitable meta-broker when one becomes available. Nevertheless, although our Grid solution does not necessarily represent the optimum use of available resources, we believe that it provides an acceptable compromise between development time, efficiency and ease of use, and that it contains features that will be useful for other large research consortia that require the flexible use of distributed computational resources by a diverse user community.

We acknowledge the assistance of Dr Matthew Davies in the molecular dynamics work and the financial support received under the EC contract FP6-2004-IST-4, no. 028069 (ImmunoGrid).

References

Almond, J. & Snelling, D. 1999 UNICORE: uniform access to supercomputing as an element of electronic commerce. *Future Gen. Comput. Syst.* **15**, 539–548. (doi:10.1016/S0167-739X(99)00007-2)

Bui, H.-H., Peters, B., Assarsson, E., Mbawuike, I. & Sette, A. 2007 Ab and T-cell epitiopes of influenza A virus, knowledge and opportunities. *Proc. Natl Acad. Sci. USA* **104**, 246–251. (doi:10.1073/pnas.0609330104)

Coveney, P. V., Saksena, R. S., Zasada, S. J., McKeown, M. & Pickles, S. 2007 The application hosting environment: lightweight middleware for grid-based computational science. *Comp. Phys. Commun.* **176**, 406–418. (doi:10.1016/j.cpc.2006.11.011)

Davies, M. N., Sansom, C. E., Beazley, C. & Moss, D. S. 2003 A novel predictive technique for the MHC class II peptide-binding interaction. *Mol. Med.* **9**, 220–225. (doi:10.2119/2003-00032.Sansom)

Hénin, J. & Chipot, C. 2004 Overcoming free energy barriers using unconstrained molecular dynamics simulations. *J. Chem. Phys.* **121**, 2904. (doi:10.1063/1.1773132)

Kertész, A. & Kacsuk, P. 2006 A taxonomy of Grid Resource Brokers. In *Proc. 6th Austrian–Hungarian Workshop on Distributed and Parallel Systems* (*DAPSYS 2006*) *in conjunction with the Austrian Grid Symposium, Innsbruck, Austria 2006*, pp. 21–23.

Kertész, A., Kacsuk, P., Rodero, I., Guim, F. & Corbalan, J. 2007 Meta-Brokering requirements and research directions in state-of-the-art Grid Resource Management. CoreGRID Technical report no. TR-0116.

Lundegaard, C., Lund, O., Kesmir, C., Brunak, S. & Nielsen, M. 2007 Modeling the adaptive immune system: predictions and simulations. *Bioinform.* **23**, 3265–3275. (doi:10.1093/bioinformatics/btm471)

Miotto, O., Tan, T. W. & Brusic, V. 2008 Rule-based knowledge aggregation for large-scale protein sequence analysis of influenza A viruses. *BMC Bioinformatics* **9**(Suppl. 1), S7. (doi:10.1186/1471-2105-9-S1-S7)

Nielsen, M., Lundegaard, C., Worning, P., Lauemoller, S. L., Lamberth, K., Buus, S., Brunak, S. & Lund, O. 2003 Reliable prediction of T-cell epitopes using neural networks with novel sequence representations. *Protein Sci.* **12**, 1007–1017. (doi:10.1110/ps.0239403)

Novotny, J., Tuecke, S. & Welch, V. 2001 An online credential repository for the Grid: MyProxy. In *Proc. 10th Int. Symp. on High Performance Distributed Computing* (*HPDC-10*), pp. 104–111. Piscataway, NJ: IEEE Press.

Pappalardo, F., Musumeci, S. & Motta, S. 2008 Modeling immune system control of atherogenesis. *Bioinformatics* **24**, 1715–1721. (doi:10.1093/bioinformatics/btn306)

Phillips, J. C. *et al.* 2005 Scalable molecular dynamics with NAMD. *J. Comput. Chem.* **26**, 1781–1802. (doi:10.1002/jcc.20289)

Richards, A., Schmidt, J., Dew, P. M., Youhanaie, F., Ford, M. & Geddes, N. 2004 Production quality e-Science Grid. In *Proc. UK e-Science All Hands Meeting, Nottingham, UK, 31st August–3rd September 2004*.

Sloan, T. M., Menday, R., Seed, T., Illingworth, M. & Trew, A. S. 2006 DESHL—standards based access to a heterogeneous European supercomputing infrastructure. In *Proc. 2nd IEEE Int. Conf. on e-Science and Grid Computing—eScience 2006, 4th–6th December, Amsterdam*, p. 91.

Zasada, S. J., Saksena, R. S., Coveney, P. V., McKeown, M. & Pickles, S. 2006 Facilitating user access to the Grid: a lightweight application hosting environment for grid enabled computational science. In *Proc. 2nd IEEE Int. Conf. on e-Science and Grid Computing*, p. 50.

Zasada, S. J., Cheney, B. G., Saksena, R. S., Suter, J. L., Coveney, P. V. & Essex, J. W. 2007 Production level scientific simulation management on international federated Grids. In *Proc. 2nd TeraGrid Conference, Madison, WI, 4–8 June 2007*.