

Grid Interoperability: An Experiment in Bridging Grid Islands

Blair Bethwaite
Faculty of Information
Technology,
Monash University,
Clayton, 3800, Victoria,
Australia
blair.bethwaite@infotech.
monash.edu.au

David Abramson
Faculty of Information
Technology,
Monash University,
Clayton, 3800, Victoria,
Australia
david.abramson@infotech.
monash.edu.au

Ashley Buckle
Faculty of Medicine,
Nursing and Health
Sciences,
Monash University,
Clayton, 3800, Victoria,
Australia

Abstract

In the past decade Grid computing has matured considerably. A number of groups have built, operated, and expanded large testbed and production Grids. These Grids have inevitably been designed to meet the needs of a limited set of initial stakeholders, resulting in varying and sometimes ad-hoc specifications. As the use of e-Science becomes more common, this inconsistency is increasingly problematic for the growing set of applications requiring more resources than a single Grid can offer, as spanning these Grid islands is far from trivial. Thus, Grid interoperability is attracting much interest as researchers try to build bridges between separate Grids. Recently we ran a case study that tested interoperation between several Grids, during which we recorded and classified the issues that arose. In this paper we provide empirical evidence supporting existing interoperability efforts, and identify current and potential barriers to Grid interoperability.

1. Introduction

Grid interoperability and Grid interoperation are two common labels often used interchangeably when discussing the challenges involved in spanning Grids. The Grid Interoperability Now community group (GIN-CG) [1] of the Open Grid Forum (OGF) [2] brings together significant experience and expertise in the area. Hence we defer to the community's definition, though others have suggested alternatives [3]. In this paper we are primarily concerned with Grids as de-centralized, cross-organizational infrastructures enabling e-Science.

A recent call for papers [4] summarizing GIN echoed the definition used within the community; describing interoperations as short term efforts addressing what needs to (and can be) done today, using as much existing

technology as possible, to enable Grids to work together. In contrast, interoperability is a longer term, broader goal to achieve native interoperation of Grids and Grid middleware through open standards. In this paper we use these definitions and the umbrella term *interop* when discussion encompasses both definitions.

As e-Science continues to be embraced by the wider research community (i.e. not just the traditional drivers such as computational physics and biology) the demand for resources also increases. The capability for the simple interoperation of Grids is now extremely important for the researchers and virtual organizations (VOs) that constitute their users. Only a decade ago it was common for research institutions to have a single computational cluster, if at all, whereas now we are seeing the formulation of dedicated e-Research and e-Science centres within these institutions.

There are several benefits in achieving Grid interoperation: it increases the resources available to any one particular VO (this is especially important in countries where the national infrastructure restricts researchers' ability to 'think big'); the pooling of resources across Grids provides safety in numbers, decreasing the effect of failure within smaller parts of the combined infrastructure; peering and opportunistic usage of resources can result in better utilization, in turn increasing the cost-return ratio of the often expensive and quickly obsolete infrastructures.

Grid interoperability is a problem today for a number of reasons. Many Grids have evolved based on different and incompatible middleware (common examples include the Globus Toolkit (GT) [5], gLite [6], and UNICORE [7]). Jobs are submitted and described differently, data is stored in archives with differing interfaces, authentication and authorization are handled in customized manners, resource information is published using differing protocols and schemas, policy and usage constraints are not uniform. On top of these potential barriers, the same issues that make using Grids an often time-consuming and

error-prone process are exacerbated when attempting to cross Grid boundaries.

The case study we detail in this paper tested Grid interoperability and interoperation between Australian EnterpriseGrid [8], APAC Grid [9], PRAGMA Grid [10],[11], FermiGrid [12], and Open Science Grid (OSG) [13]. All of these Grids use some version of the GT for basic job and data management, though some have customized various components to meet specific requirements, such as FermiGrid's gateway service. These Grids were chosen partly because of existing relationships and collaborations and also because they represent a wide array of Grid maturity levels. We have deliberately avoided Grids based on other middleware stacks (there is sufficient variation between pre- and web service based versions of the GT), however this will be the subject of extended future work.

In order to ensure the validity and relevance of our study, the application we ran was 'real' science, a Nimrod/G [14] experiment in structural biology. Specifically, multiple conformations of multiple proteins from the Protein Data Bank were used in a protein crystal structure determination, using the technique of Molecular Replacement (MR). This involved testing the entire Protein Data Bank using the MR application PHASER, as discussed in more detail in the appendix. This experiment generated more than seventy thousand executions across the five Grids, requiring over half a million CPU hours.

In the following sections we discuss related work in the field, detail the methodology of our case study, classify and discuss the Grid interop issues that arose, suggest conclusions that can be drawn from this experience, and summarise the application science.

2. Related work

Previous and current interoperability work includes standardization and Inter-Grid (as in Internet). There has been considerable effort, motivated by various applications, invested in the interoperation of Grid infrastructures. This has resulted in interoperation between Grid software and been fed back into further interoperability efforts.

Standardization work is primarily motivated by the well-known need to overcome the difficulties faced by Grid users and applications developers due to the currently incompatible interfaces of the various Grid middleware and service stacks. The OGF is the de-facto leader of standardization efforts for the Grid. In early 2005 the OGF released The Open Grid Services Architecture (OGSA) (updated 2006) [15], a broad framework and foundation for current and future work (particularly standardization) encompassing the capabilities of "Execution Management, Data, Resource Management, Security, Self-Management, and

Information" [15]. The OGF has working groups focusing on all these areas and a significant number of recommendations and informational documents have been released as a result of these efforts. Some of these standards, for example the Distributed Resource Management Application API (DRMAA) and Job Submission Description Language (JSDL) [16], are now supported in a variety of Grid software stacks [17]. However, in a recent summary work [17] the GIN-CG identified that despite the expertise invested and effort expended in their authoring, standards adoption by Grid middleware providers has been slow. This is compounded by the fact that in the meantime many existing infrastructures have created customised solutions to their particular problems and hence are invested in these non-standards based solutions until motivation enough for change presents itself.

The GIN initiative, as the name suggests, was started to promote interoperations on short time scales. It provides a forum for coordination of these exercises, development of best practice and guidelines, and important feedback into other OGF work including standards. APAC Grid, PRAGMA Grid, and OSG are all GIN contributors and are committed to sponsoring and enabling GIN efforts.

There has been little work on models of interoperability between Grids. Most previously suggested models are based on the concept of a unified worldwide Grid, i.e. Inter-Grid, which has been around since the definition of the Grid (as we use it today) was cemented by Foster and Kesselman in their seminal book [18]. This vision, refined later with Tuecke [19] and also in separate research [20] works, relies heavily on the resolution of existing interop problems. Of particular importance is the widespread adherence to and pervasion of standards across the Inter-Grid in order to enable higher-level services and systems. This seamless, interoperable, high-performance and high-throughput computing environment is full of promise but requires much further investigation in areas such as service brokering, negotiation and virtual market places, to name a few. It would be premature however, to embark on these activities without a thorough understanding of the existing issues.

Many production Grids have embarked on interop projects, for example OSG and TeraGrid, OSG and EGEE, EUC hinaGrid [21], OSG and PRAGMA, NAREGI and EGEE, and NGS and GridPP [22]. Much of this work is ongoing. Examples of middleware efforts include GRIP [23], which resulted in an interoperability layer between UNICORE and GT2, OSG and LCG (as detailed in [24]), and the ongoing OMII Europe [25] project. These middleware and infrastructure interop projects are often pair-wise and typically involve testing

or developing interop between two specific versions of a technology, either with differing or the same vendors.

Interop work is not motivated by purely academic interest. It is the needs of applications and their users that drive progress. Most often interop work is driven by the need to increase the resource pool available for particular applications, either to exploit the aggregated size, improved availability or wider diversity. This is the case for example, with OSG and LCG interop work and federated UK-US Grid efforts (detailed further in [26]).

Much of the existing Grid interop work is characterised by the motivation to get one Grid working with another, so that users from one Grid can submit jobs to the other in a manner compatible with their current use. Related work [17],[3] has suggested that, if we continue interoperations in this fashion, each additional effort brings us another step closer to realising the goal of a unified Grid. However, whilst these efforts are important, particularly in gaining short term solutions, they are clearly not solving the whole problem, as we can see by the duplication of similar efforts around the world. It is doubtful that the continuation of this kind of interop work will lead to a uniform Grid landscape, if only because Grids are not static systems – they have a symbiotic relationship with the ever-changing science and applications that drive and utilise them, so that the requirements for the Grid systems of today will likely be highly inadequate for the Grid systems of the future.

3. Methodology

In order to distinguish interop issues from other potential problems during the case study, we followed a workflow as shown in figure 1 (below). This workflow shows the steps taken when adding a computational resource to the Nimrod/G experiment. We began the experiment on our known ‘home’ Grid (EnterpriseGrid) and started the resource discovery, testing, and application deployment steps on the remaining Grids in parallel. Finally, adding each of these Grids to the experiment with intervals of at least a week.

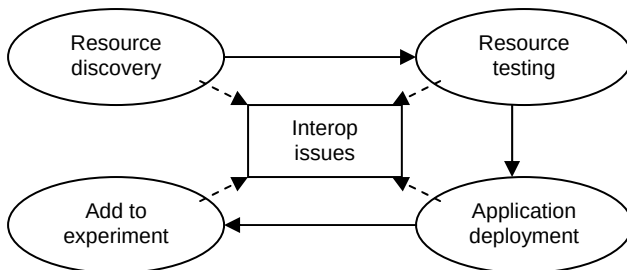


Figure 1: Interoperation workflow

Resource discovery involves finding information about resources so they can be used by Nimrod/G (further details in section 4.2) and communicating with resource providers regarding the details of the interoperation. Resource testing involves checking authorization and running automated resource checks within Nimrod/G. Application deployment (discussed further in [27]) was performed by a separate Nimrod/G experiment run once on each resource. After the successful completion of these steps, and gathering of interop issues along the way, resources from each Grid were added to the parent Nimrod/G PHASER experiment.

4. Identification of Grid interop issues

In this section we detail and discuss the interop issues encountered during our case study. We have identified five areas of interop where we encountered issues, though as we show these areas sometimes overlap and combine before exhibiting problems.

4.1. Access and security

Different Grids have varying authentication, and security mechanisms and processes, which have evolved based on the requirements of the middleware deployed on those Grids. Today the majority of Grid identity management is based on X.509 [28] public key infrastructure, of which the cornerstone is based on trust of well known third party certification authorities (CAs). This trust fabric can present problems for interoperation when crossing into a domain which trusts only a subset of the CAs from the originating domain [17]. Fortunately these sorts of problems are less common since the Tokyo Accord in 2003, subsequent formation of the International Grid Trust Federation (IGTF) in 2005, and continued growth of the member Policy Management Authorities (PMAs). During our case study we used X.509 certificates and proxy credentials signed by the APACGrid CA, which is an IGTF-AP [29] accredited CA and hence included in the IGTF CA distribution which is trusted across our testbed.

A more common issue in Grid interoperations, and even intra-operations, today is management of the virtual organisation memberships necessary for authorization within a Grid. For example, during the course of our case study it was necessary to use three separate Virtual Organization Membership Service (VOMS) [30] proxy credentials in order to be authorized to use APAC, FermiGrid, and OSG resources. Fortunately Nimrod/G binds credentials to resource definitions allowing the use of multiple certificates within a single experiment.

We believe that the use of VOMS in Grid interoperation work highlights some key weaknesses in its usage model. One of the basic assumptions in a VOMS

authorization environment is that VOs remain small (e.g. project group) and tightly bound with regard to their purpose and administration. This is a necessary characteristic to ensure minimal security risk within the VO because the existing implementation maps VOs to shared local accounts on Grid resources.

When coupled with Grid interoperations, VOMS presents a much higher security and accident risk to those within a VO established for interoperations work as they have no idea who the other members are, what their intentions are, and whether their usage could interfere with other members. Whilst VOs within production Grids such as OSG must meet high requirements for entry to the Grid (such as a minimum standard of Acceptable Usage Policy (AUP) and nominated points of contact for administration and incident response) this does not fulfil the very obvious requirement of protecting users from one another. Instead it provides a means to deal with a problem after it has occurred. Of course this does not only apply to VOs used for interoperation activities but also to any VO which is larger or more loosely coupled than the arbitrary ideal. There is an analogy here with the CA problems in the early days of Globus Toolkit development when everybody was using the Globus CA and so concern was raised over the validity of the identities asserted by it.

4.2. Resource discovery

One of the first steps when attempting to use a new Grid, after gaining authorization, is to discover and test resources. This is often a manual and very time-consuming activity. Service directories and Grid information services have gained attention recently as Grids and their user bases have expanded. Standardization of the information schemas [24] used to publish resource information and development of transformations between the existing incompatible profiles are areas of active work. For example the Grid Laboratory Uniform Environment (GLUE) working group in the OGF has produced the 2.0 version of the GLUE abstract schema.

Historically, however, information services are an area of Grids that have been neglected. For example our experiences developing and using Nimrod/G and attempting to support discovery of resource information from the Globus Toolkit Monitoring and Discovery System (MDS) led us to augment those features with user-specified knowledge and a custom user interface. This was necessary because information contained in the MDS was often missing or incorrect and when present was often not in a reliable format.

Further to the problems of incompatible information services is the difficulty of encouraging Grid managers to provide any information at all! Of the Grids in our testbed, only OSG, FermiGrid and APAC had any published information in a standard schema that could

have been queried programmatically. We should note that the information required to configure a resource for use with Nimrod/G is minimal, for example for a Globus based resource the list is: the URL of the gatekeeper host, the name of the execution jobmanager, the queue name if applicable, and the operating architecture of the gatekeeper, head node, and compute hosts. Because of the lack of this information, we have built special modules into Nimrod/G that probe and query the individual resources in an attempt to discover as much of this information as possible. The fact that we were able to develop a customised solution suggests that Globus should have included a tool for populating the MDS in a consistent manner. The adaptation of monitoring tools such as SCMSWeb [31],[32], recently deployed on PRAGMA Grid, may eventually fill this gap.

It is interesting to note that the production-level Grids lead the way in uniform provisioning of information services. This is possibly because in experimental and testbed Grids the effort required to set up and support these services (which are typically not configured in an automated fashion) is seen as non-essential to the function of the resource within the Grid. This lack of resource information available in standard schemas greatly increases the human effort required to support Grid interoperations and presents a large barrier to non-expert users attempting Grid interoperations.

4.3. Usage policies and guidelines

A further preliminary step when establishing an inter-Grid testbed is to discover and ensure conformance with the usage policies and guidelines of the constituent Grids, assuming it is even possible to do this in a uniform manner. When appropriately populated information services are available much of this conformance could be gained automatically because many of the items typically covered in these documents relate to use of the execution environment (for example file system paths to temporary storage space and deployment areas).

When this information is not available it is typical to fall back to using the Grid home directory set by the job management middleware, indeed without any further guidelines this is all that is possible. This was the case for all Grids in our testbed except OSG and FermiGrid. Because of this, it was necessary to customize application deployment and write bootstrap scripts that allowed the application to be deployed and invoked via Nimrod/G in a uniform fashion.

Fortunately Nimrod/G's remote invocation mechanisms are flexible enough to allow for workarounds like this, but this is yet another area of interoperations requiring specific expert knowledge. The first step to ease this burden would be the creation, dissemination, and uptake of a middleware-independent standard job

execution environment. It is pleasing to see that the first stage in this process has begun [33] under the auspices of the GIN-CG. If this were to become widely supported it would be possible to author simple user-level tools capable of handling many of the application deployment and invocation requirements faced when running across Grids and even within Grids.

4.4. Application compatibility

As discussed in previous work [34] a common issue and barrier encountered when ‘Grid-ifying’ a legacy application is ensuring its portability across processor and operating system architectures. Continued improvements in both hardware and software virtualization support, and experiences with dynamically interpreted and virtual machine (VM) based programming languages, point to the medium term resolution of many of these problems. It is reasonable to expect users will see the benefit in taking an execution performance hit (virtualization overhead) several orders of magnitude lower than the throughput they gain by moving to a Grid environment. Whilst it is becoming more common to see operating system VMs used for sandboxing and consolidation benefits on Grid resources, there is still much work required before VMs are used in a user transparent manner within and across Grids to provide the sort of Dynamic Virtual Environment (DVE) proposed in [20].

This heterogeneity in operating environment concerns more than just differences in architecture and operating systems, but also issues as simple as machine configuration. Virtual memory limitations, for example, are a common problem even in intra-Grid operations let alone interoperations. During our case study we encountered several issues in this regard when moving into the inter-Grid testbed.

Our starting Grid (EnterpriseGrid) was highly homogeneous as the machines for the four sites were all purchased at the same time and from the same vendor, the compute nodes containing 8Gb of RAM and a further 6Gb of swap space for a total of 14Gb of virtual memory shared across eight processor cores. This is, relative to modern standards, a reasonably memory rich configuration for worker nodes and as such we did not encounter any memory problems. However, once we expanded execution into the PRAGMA Grid and OSG it became clear that some jobs were attempting to use more virtual memory than available on machines within those Grids.

There are two aspects to this problem. First, in many scenarios, as with our case study, it is not possible to know the resource requirements of any one particular job ahead of time and it would be unacceptably limiting to request only resources able to fulfill the upper limit when the execution profile is so varied. This means we had to

accept that some jobs would fail due to exceeding virtual memory limits. We were able to augment Nimrod/G to retry these jobs on different resources, but this still resulted in the loss of many thousands of hours of CPU time (much to the disdain of those providing the cycles).

Second, as illustrated by several resources on the PRAGMA Grid during our case study, queuing and batch systems are often not fully or sufficiently configured. In many cases the configuration of limits on basic system consumables such as virtual memory and wall time were neglected, though these can be restricted in most queuing system implementations. During our case study, the memory requirements of some jobs on such resources exhausted the available virtual memory and caused the operating systems on those machines to thrash and eventually become unresponsive. Getting those machines back into operation required physically rebooting them – not a simple process when those machines are geographically distributed and under local autonomous control! This lack of configuration is not uncommon or surprising, especially given many of the resources on testbed Grids are operated and maintained by researchers who may have little system administration experience let alone expertise with Grid software. This further highlights the need for features or tools to automate and simplify the configuration of the Grid infrastructure.

4.5. Middleware compatibility

Much previous Grid interoperation work has sought to address incompatibilities in existing Grid middleware and infrastructures. This focus is mirrored in standardization efforts within the OGF; however middleware interoperability is not a silver bullet for achieving Grid interoperability. Simply ensuring that all clients and services are communicating in a standard fashion does not guarantee that those interactions will have the same effects across varying implementations.

Problems arise because these standards, such as the OGSA Basic Execution Service [35], do little to specify non-functional requirements on the interfaces, potentially resulting in unforeseen side effects in the concrete implementations. We encountered an example of this during our case study when the same GT4 job submission interface had a side effect on some resources but not others.

To illustrate the problem it is necessary to understand some of the details of the Nimrod/G remote execution mechanism. Nimrod/G deploys a remote agent to computational resources which contacts the Nimrod/G server to collect work and deposit results. It is not necessary to collect the output from the Globus Resource Allocation Manager (GRAM) job which launched the agent, except when debugging the agent itself, because Nimrod/G uses its own mechanisms for file transfer. As

such, Nimrod/G does not request this output from Globus, and hence, nor can it explicitly clean it up because as far as Nimrod/G is concerned it does not exist. Also for this reason, the agent submission does not specify any particular initial working directory for the agent.

These two simple details of the Nimrod/G submission exposed an inconsistency between GRAM implementations used on different Grids. It is common for local resource managers, such as PBS and SGE, when used directly, to automatically create standard output and error files when they are not explicitly requested or specified by the job. Unfortunately the GT4 interface to PBS does not deal with these automatically generated files. This resulted in a build up of empty output files in the initial working directory of the job, which Globus had defaulted to the home directory. When coupled with VOMS (which meant the local user account was shared by many users) the eventual build up of files in this directory caused an effective denial of service for other users attempting to use that file system.

Thus, a combination of poorly specified side effects of a given interface, plus assumptions in other Grid infrastructure like the VOMS software, caused unexpected errors. This is relevant to current standardization and interoperability efforts because this issue occurred within a single software stack with an identical specification. Clearly ensuring middleware interfaces are interoperable by adhering to a single functional specification can not solve these sorts of problems. Of course, this is a general software engineering program, and is not just applicable to Grid computing. However, the wide distribution and lack of central control makes it far more significant in an open distributed system like the Grid, because the components being connected are not controlled by a single organisation.

5. Conclusions

Successful interoperation of Grids for large scale science continues to present challenges and generally requires expert technical knowledge in the field. In this paper we have discussed and illustrated some of the issues that define Grid interoperability and make Grid interoperation difficult. However, we have also shown through the application science of our case study that interoperation is possible and indeed a worthwhile pursuit. The structural biology workflow that we performed to drive interoperation efforts is an early example of techniques that are likely to become common place as emerging fields such as metagenomics continue to grow.

We identified five categories that the interop issues encountered fall into: access and security, resource discovery, usage policies and guidelines, application

compatibility, and middleware compatibility. Following is a summary of the contributions of our paper.

In 4.1, Access and security, we provided evidence supporting current authentication schemes but noted that the VOMS authorization implementation deployed on some of the Grids had the potential to cause problems, especially in scenarios such as interoperation where VOs can become very loosely coupled.

In 4.2, Resource discovery, we discussed the importance of the uniform provision of information services. We also identified the need for tools which simplify the configuration and maintenance of these services, an area of potential future work.

In 4.3, Usage policies and guidelines, we noted the lack of a standard base level execution environment and showed an example use case, application deployment, which could be greatly simplified by its development and uptake. We noted that the initial development [33] of such a guideline is under way within GIN and recommend that it be pursued and published as a best practice.

In 4.4, Application compatibility, we showed that machine architecture and operating system are not the only issues requiring consideration when running an application in such a heterogeneous environment as a Grid. Good progress has been made in match-making when resource requirements are well known, but this does little to help when those requirements are highly variable and dynamic. Here we also gave examples where this was problematic for the resources as well as the application. This provides further support for future efforts to ease configuration and management of Grid computing software infrastructure.

In 4.5, Middleware compatibility, we gave an example of an interoperation problem concerning middleware that could not be solved by current standards. Many related works (e.g. [3],[17],[20],[26],[19]) have concluded with calls for the increased support of standards across various middleware. We agree with this in principle though we caution that standardization in its current form does not guarantee successful interoperations. A simple remedy to the kinds of problems we encountered might be for standards to include specification of pre- and post-conditions at interface boundaries between systems. Future work could apply formal software engineering techniques to middleware standards.

Our findings on the state of Grid interoperations are directly applicable to the areas of the GIN working groups and will, we hope, provide valuable direction for their future work. The PRAGMA Grid is actively involved in GIN efforts and we expect some of the issues we have highlighted here to spurn discussion within the PRAGMA resources working group.

6. Acknowledgements

This research was done using a large number of resources supported by various funding sources. We would like to acknowledge:

The resources provided by the Open Science Grid, which is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science.;

The resources provided by the Pacific Rim Applications and Grid Middleware Assembly and its member organizations, PRAGMA member institutes are funded by funding agencies of their respective governments. PRAGMA is supported by NSF Grant No. INT-0216895, INT-0314015 and OCI-0627026.; and

The Australian Research Council's provision of the Australian EnterpriseGrid resources, including our colleagues at Deakin University, RMIT and QUT.

A great number of people have contributed in various ways and been involved in the efforts detailed herein. We would like to give particular thanks to Cindy Zheng, Neha Sharma, Mats Rynge, Nadya Williams, and Steve Androulakis.

7. Appendix: Molecular replacement Grid

The most common method of crystallographic protein structure determination is molecular replacement (MR). This technique involves using the structure of a protein that shares significant sequence similarity with the protein of unknown structure as a starting point in the structure determination (otherwise known as solving the *phase problem*). The process generally involves four steps: (1) Using sequence-based searching methods such as PSI-BLAST [36] to identify suitable structures that can be used for MR; (2) modification of structures to yield *search models*; (3) Finding the orientation and position of the search model in the unit cell of the target crystal; (4) Refinement of the model.

Molecular replacement has been used to determine the structure of approximately half of the 50,000 structures deposited in the Protein Data Bank (PDB; 67% of 2006 releases were solved by MR [37]). It is anticipated that the proportion determined by MR will grow for three reasons: First, the probability that the unknown target structure belongs to readily identifiable fold is steadily increasing, due to the rapid growth of the PDB. Second, the emergence of more sophisticated sequence searching algorithms, such as profile-profile matching [38], improve the probability of finding a suitable search model, even in cases of very low similarity (<20% identity). Third, MR algorithms consistently improve.

Where the sequence similarity between the unknown target and the search model is high (sequence identity >40%) the success rate of MR is very good, even without optimization of the search model. However, in cases where sequence similarity is low (identity <30%) MR, and subsequent structure refinement becomes non-trivial,

and emphasis must be placed on the optimization of the search model. A key breakthrough in successfully applying the MR approach to situations where sequence identity is low was the development of the PHASER maximum likelihood approach [39].

Even in cases where a MR solution with low overall sequence identity can be obtained these solutions are commonly challenging to refine (the so called "model bias" trap). This situation occurs where errors in regions of the starting model cannot be adequately identified and corrected due to model bias. There are several criteria that affect the outcome of the MR calculation; 1) structural similarity between search model and target structure (measured by root mean square deviation (RMSD)); 2) percentage of residues missing from the search model (*coverage*); 3) the amount of conserved side chains that are expected to remain structurally conserved (for example in the protein interior). These factors, and thus the outcome of the MR calculation, can be influenced by improvement of the search model. The simplest approach is to remove regions of the structure that are predicted to be different in the search model and target, typically loops. However, this process is a subjective one and relies on sequence alignments, which are often incorrect, particularly at low sequence identity. Thus it is often unclear which loops should be removed and how much of the loop should be removed, and each model must be tested. In some cases the true space group is unknown and several alternatives must be tested in the MR calculation. In addition the estimated RMSD between the search model and unknown structure can affect the outcome of the MR calculation, leading in the worst case to probable solutions being missed. Therefore, the combination of multiple models, space groups and RMSD values makes MR time and labor intensive, and puts an emphasis on the availability and power of computational resources.

Utilizing PHASER discussed above, we used Nimrod/G [14] to distribute thousands of MR calculations in parallel, thereby increasing the likelihood of finding a structure. The system takes as input the structure factor data and a compressed file of search models, and parses the input and distributes jobs to available nodes on an available grid resource. Each job runs to completion independent of all other jobs, and results of the submission are returned to the user.

Using this approach we have been able to perform MR calculations using the entire PDB (~80,000 structures) in a reasonable timeframe. This feat was inconceivable three years ago (and many in our community consider it still is). Our approaches have lead to the structural solution of several key medically important proteins in our laboratories, many of which involve collaborations with researchers from Monash University and several international laboratories, which would otherwise have

remained unsolved. Much of this data is currently being published in top international journals.

8. References

- [1] "Grid Interoperation Now Community Group (GIN-CG)"; <https://forge.gridforum.org/sf/projects/gin>.
- [2] "Open Grid Forum"; <http://www.ogf.org/>.
- [3] L. Field and M. Schulz, "Grid interoperability: the interoperations cookbook," *Journal of Physics: Conference Series*, vol. 119, 2008, p. 012001.
- [4] "Journal of Grid Computing"; <http://www.springer.com/computer/communications/journal/10723>.
- [5] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *International Journal of Supercomputer Applications*, vol. 11, 1997, pp. 115--128.
- [6] E. Laure et al., "Programming The Grid with gLite," *Computational Methods in Science and Technology*, vol. 12, 2006, pp. 33-46.
- [7] D.W. Erwin, "UNICORE - A Grid Computing Environment," *Concurrency and Computation: Practice and Experience*, vol. 14, 2002, pp. 1395-1410.
- [8] "Enterprise Grid"; <http://www.enterprisegrid.edu.au/>.
- [9] "ARCS - Australian Research Collaboration Service"; <http://www.arcs.org.au/>.
- [10] C. Zheng et al., "The PRAGMA Testbed - Building a Multi-Application International Grid," *Cluster Computing and the Grid Workshops, 2006. Sixth IEEE International Symposium on*, 2006, p. 57.
- [11] "PRAGMA — Pacific Rim Applications and Grid Middleware Assembly"; <http://www.pragma-grid.net/>.
- [12] "FermiGrid"; <http://fermigrd.fnal.gov/>.
- [13] "Open Science Grid"; <http://www.opensciencegrid.org/>.
- [14] D. Abramson, J. Giddy, and L. Kotler, "High performance parametric modeling with Nimrod/G: killer application for the global grid?," 2000, pp. 520-528.
- [15] I. Foster, H. Kishimoto, and A. Savva, eds., "GFD-I.080 The Open Grid Services Architecture, Version 1.5," Jul. 2006; <http://www.ogf.org/documents/GFD.80.pdf>.
- [16] "OGF Documents"; <http://www.ogf.org/gf/docs/?final>.
- [17] Morris Riedel et al., "Interoperation of World-Wide Production e-Science Infrastructures," *Concurrency and Computation: Practice and Experience*, In-press. ; http://www.ogf.org/OGF_Special_Issue/cpeginpaper_riedel.pdf.
- [18] I. Foster and C. Kesselman, eds., *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, 1998.
- [19] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of High Performance Computing Applications*, vol. 15, Aug. 2001, pp. 200-222.
- [20] M.D.D. Assunc, ~ao, R. Buyya, and S. Venugopal, "InterGrid: a case for internetworking islands of Grids," *Concurrency and Computation: Practice and Experience*, vol. 20, 2008, pp. 997-1024.
- [21] "EUCINA GRID"; <http://www.euchinagrid.org/>.
- [22] J. Jensen et al., "Practical Grid Interoperability: GridPP and the National Grid Service," Nottingham, UK: 2007, pp. 119-126.
- [23] "Project GRIP"; <http://www.fz-juelich.de/jsc/cooperations/grip>.
- [24] M. Flechl and L. Field, "Grid interoperability: joining grid information systems," *Journal of Physics: Conference Series*, vol. 119, 2008, p. 062030.
- [25] "Open Middleware Infrastructure Institute (OMII) Europe"; <http://omii-europe.org/>.
- [26] B. Boghosian et al., "NEKTAR, SPICE and Vortronics: using federated grids for large scale scientific applications," *Cluster Computing*, vol. 10, 2007, pp. 351-364.
- [27] D. Abramson, "Applications Development for the Computational Grid," *Frontiers of WWW Research and Development - APWeb 2006*, 2006, pp. 1-12; http://dx.doi.org/10.1007/11610113_1.
- [28] D. Cooper et al., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," May. 2008; <http://tools.ietf.org/html/rfc5280>.
- [29] "The International Grid Trust Federation"; <http://www.igtf.net/>.
- [30] R. Alfieri et al., "From gridmap-file to VOMS: managing authorization in a Grid environment," *Future Generation Computer Systems*, vol. 21, Apr. 2005, pp. 549-558.
- [31] "OpenSCE Project - SCMSWeb"; <http://www.opensce.org/components/SCMSWeb>.
- [32] Napat Chalakornkosol and Putchong Uthayopas, "Monitoring the Dynamics of Grid Environment using Grid Observer," May. 2003; <https://www.opensce.org/components/SCMSWeb/Members/b4305127/Document/GridObserverPoster>.
- [33] Morris Riedel, "Execution Environment," *OGF Gridforge GIN-CG*; <http://forge.ogf.org/sf/go/doc15010?nav=1>.
- [34] W. Goscinski and D. Abramson, "An Infrastructure for the Deployment of e-Science Applications," *High Performance Computing (HPC) and Grids in Action*, L. Grandinetti, ed., 2008, p. 540.
- [35] I. Foster et al., "GFD-R-P.108 OGSA Basic Execution Service," Aug. 2007; <http://www.ogf.org/documents/GFD.108.pdf>.
- [36] S.F. Altschul et al., "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, Sep. 1997, pp. 3389-402.
- [37] F. Long et al., "BALBES: a molecular-replacement pipeline," *Acta Crystallographica Section D*, vol. 64, 2008, pp. 125-132.
- [38] L. Jaroszewski et al., "FFAS03: a server for profile-profile sequence alignments," *Nucleic Acids Research*, vol. 33, Jul. 2005, pp. W284-288.
- [39] A.J. McCoy et al., "Phaser crystallographic software," *Journal of Applied Crystallography*, vol. 40, Aug. 2007, pp. 658-674.